NTNU
Innovation and Creativity

# User-Perceived Quality of Service in Video on Demand Services

**Arnfinn Flo**

Master of Science in Communication Technology
Submission date: August 2006
Supervisor: Steinar Bjørnstad, ITEM
Co-supervisor: Kimsås Andreas, ITEM

Norwegian University of Science and Technology
Department of Telematics

# Problem Description

Video on Demand (VoD) is an Internet service with an increasing customer base and a potentially substantial importance to Internet Service Providers' and other Internet actors' revenue. Simultaneously, VoD is a demanding service with respect to network resource usage.

In this Master's Thesis, the student will evaluate how changes in network-QoS-parameters affect the quality of video delivered to the end-user through the Internet. The student will determine how the quality perceived by the end-user changes as a function of packet-loss rate and the available bandwidth. This will be accomplished through:

- Defining a VoD-scenario with realistic usage patterns for residential users
- Recreating these patterns in a lab-environment
- Performing a test on a group of individuals in order to assess their perceived quality in accordance with well-documented methodology
- Analyzing the results in a statistical and comparative consistent way

Assignment given: 25. April 2006
Supervisor: Steinar Bjørnstad, ITEM

# Abstract

Video on Demand (VoD) is an Internet service with a growing appeal to the mass market, and is of increasing importance to Internet service providers' revenue. This master's thesis presents a subjective assessment on the user-perceived quality of service of an imaginary VoD service. By implementing the SAMVIQ methodology of subjective video quality assessment, the state of the art video codec H.264/MPEG-4's resilience to packet loss is examined.

Through the recreation of several residential usage scenarios, different amounts of packet loss is added to H.264/MPEG-4 content encoded at diversified bitrates. The results suggest that random packet loss rates above 0,1% deteriorates the perceived quality to such an extent that it is not acceptable to the end-user. High-bitrate encoded content is relatively more affected than low-bitrate content, and bursty packet loss is preferred to loss categorized as non-bursty.

# Preface

This master's thesis documents the achievements of the 10th semester of the Master of Science study in Communications Technology at the Norwegian University of Science an Technology (NTNU). It was carried out at the Department of Telematics in the time span of March - August 2006.

I would like to thank my supervisors Andreas Kimsås and Adjunct Associate Professor Steinar Bjørnstad for valuable advice and comments during the process. PhD student Odd Inge Hillestad deserves a thank-you for help setting up and controlling the network emulator used, as does all the participants in the test that formed the base of this thesis.

Last, a special thank-you to Gunhild, for making me smoothies in the exceptional hot Norwegian summer of 2006.

Trondheim,
August 17. 2006

Arnfinn Flo

# Table of Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **ASO** | Arbitrary Slice Order |
| **AVC** | Advanced Video Coding |
| **DP** | Data Partitioning |
| **DSCQS** | Double Stimulus Continuous Quality Scale |
| **DSIS** | Double Stimulus Impairment Scale |
| **DSL** | Digital Subscriber Line |
| **DVB** | Digital Video Broadcasting Group |
| **EBU** | European Broadcasting Union |
| **FMO** | Flexible Macroblock Order |
| **GMS** | Grand Mean Score |
| **HD-TV** | High Definition TeleVision |
| **IETF** | Internet Engineering Task Force |
| **IMS** | Individual Mean Score |
| **IP** | Internet Protocol |
| **ISP** | Internet Service Provider |
| **ITU-T** | International Telecommunication Union |
| **JMF** | Java Media Framework |
| **JVT** | Joint Video Team |
| **MOS** | Mean Opinion Score |
| **MPEG** | Motion Picture Experts Groups |
| **MPEG TS** | MPEG Transport Stream |
| **NAL** | Network Abstraction Layer |
| **OSI** | Open System Interconnection |
| **PES** | Packetized Elementary Stream |
| **PSNR** | Peak Signal to Noise Ratio |
| **QoS** | Quality of Service |
| **QTJava** | QuickTime for Java |
| **RTCP** | Real Time Transport Control Protocol |
| **RTP** | Real Time Transport Protocol |
| **SAMVIQ** | Subjective Assessment Methodology for Video Quality |
| **SDSCE** | Simultaneous Double Stimulus for Continuous Evaluation |
| **SSAT** | SAMVIQ Subjective Assessment Tool |
| **SSCQE** | Single Stimulus Continuous Quality Evaluation |
| **TCP** | Transmission Control Protocol |
| **UDP** | User Datagram Protocol |
| **VCEG** | Video Coding Expert Group |
| **VCL** | Video Coding Layer |
| **VoD** | Video on Demand |
| **VoIP** | Voice over IP |

# Chapter 1

# Introduction

**Contents**



This chapter presents the motivation and the objective of this master's thesis. It justifies the importance of a user-perceived view on quality of service and presents the state of the art video codec H.264/MPEG-4 as a key factor in modern video on demand systems. Finally, the structure of this report is described.

## 1.1 Motivation

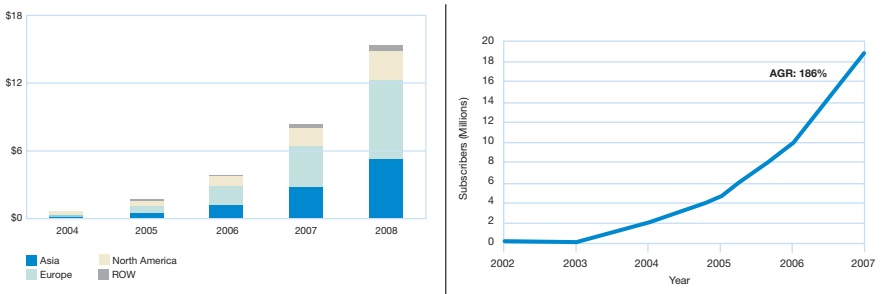
> As Internet has evolved, from an experimental four-node network in 1969 to the global information highway of today, it's importance and relevance has increased dramatically. The number of Internet users is continually rising, along with the number of networked applications. Because of improved technology and a strong competitive telecommunication-marked, Internet Service Providers (ISPs) strive to offer constantly higher bandwidth-capacities to the end-user. This ever-increasing carousel has resulted in an explosion of new services, in which **Video on Demand (VoD)** is one of the most appealing to end-users.

This section presents the motivation behind this master's thesis. It describes the importance of VoD to service providers revenue, and shows how the end-user's perception of video-quality is an important factor in the telecommunication-marked. In addition, it is explained why the H.264/MPEG-4 video codec is believed to play a substantial role in the deployment of VoD services and why it

is important to survey the codec's resilience to packet loss in best-effort networks like Internet.

### Market Trends

Higher bandwidth-capacities fuel the deployment of new bandwidth-demanding services in Internet. Voice over IP (VoIP) has gone from being an unreliable experimental service, to an application transforming the telecommunication-marked completely. While VoIP now is considered "yesterday's news", academia and commercial service providers turn their attention to other areas. VoD is an Internet service attracting considerable interest from these actors, and is by many ISPs considered to be the most promising service for new future revenues [2]. Figure 1.1 shows the predicted growth in this marked, which undoubtedly can be described as substantial.




(a) Projected Revenue for Video Services

(b) Projected Video Subscriber Growth

Figure 1.1: Video Services Marked Trends [45]

### The Economy of Subjective Quality of Service

In unreliable networks like Internet, the quality of real-time services can never be guaranteed. As the number of real-time services delivered through best effort IP networks increases, so does the need to asses this quality in order to understand which factors that influence it. As it is the end-user who ultimately decides whether a delivered service is of good quality or not, it is important to carry out *subjective* assessments focusing on the end-user's perception of the quality delivered. This is called a subjective view on quality of service, and is the basic paradigm on which this thesis is based.

A subjective view on quality of service is not only important as a theoretical term, it is of vital importance to the service providers' revenue. As Internet matures, the overall traffic growth will moderate and access prices will need to stabilize and firm up to secure the network providers' revenue [48]. This means that other factors than price will be important for the customers. This shift in customer preferences can be described as a transition from a price-oriented behavior to a quality-concerned one. In other terms, user-perceived quality of service becomes an important factor that network and service providers must respond to in order to secure and increase their customer base.

### Video Quality - the Codec Revolution

In the "pre Internet age", video compression schemes - or codecs - were originally developed to reduce the disc-space needed on the storage medium. These codecs are very efficient in terms of bitrate reduction. However, they are less suited for video streamed over best-effort networks, where packet loss and delay are introduced as substantial quality degradation factors. The packet loss experienced is caused either by packets dropped in the network due to congestion, or by delayed packets arriving at the receiver end after their play-time has expired. Video is especially vulnerable to degradation because of packet loss, mostly because of the exploitation of the temporal interdependencies between video frames [7].

In May 2003 the ISO/IEC Moving Picture Experts Group (MPEG) and the ITU Telecommunication Standardization Sector (ITU-T) issued the final draft on their common effort, the H.264/MPEG-4 video codec standard. The aim of this standard was to create a codec capable of providing good quality video at substantial lower bitrates than previous codecs, and that it should work well on a wide variety of networks and systems, including IP-networks like Internet.

Since it's release the H.264/MPEG-4 codec has gained a huge interest from many parties. The MPEG-2 standard, which has been the industry standard for digital TV-distribution since the mid 1990's, is about to be outdistanced by H.264/MPEG-4 and numerous VoD actors (e.g Apple and Sony) are making the transition.

### Packet Loss Influences Video Quality

In the project work preceding this master's thesis [20], it was argued that the packet loss rate is the dominant degrading parameter when a VoD service is subjected to unreliable environments like Internet.

While the packet loss rate in the fixed Internet have continuously decreased, the deployment of wireless access have once again raised the overall end-to-end loss rate. The improved technology and convenience of wireless access have gotten more and more Internet users to connect to the Internet through wireless solutions, and the data loss rate in wireless networks are much higher than in wired networks [37].

While several studies have shown H.264/MPEG-4's superiority over other codecs, little is known about it's resilience to packet loss. The few studies that have been published concerning this issue, are by great majority objective tests based on objective metrics. It is therefore believed that a subjective test, asserting H.264/MPEG-4's performance in lossy environments will be of interest, both to VoD providers and other actors interested in the codec's performance.

## 1.2 Objectives

The main objective of this master's thesis is to:

> Determine how the quality perceived by the end-user changes as a function of the packet loss rate and the available bandwidth in a typical VoD service.

This will be accomplished through the following steps:

- Defining a VoD-scenario with realistic usage patterns for residential users.
- Recreating these patterns in a laboratory environment.
- Performing a test on a group of individuals in order to assess their perceived quality in accordance with well-documented methodology.
- Analyzing the results in a statistical and comparative consistent way.

The term *typical VoD service* is defined in detail in Section 3.1, as are the specific goals of the subjective test itself.

## 1.3 Scope and Limitations

#### Preliminary Work

This master's thesis is the continuation of a project work titled User Perceived Quality of Service in Packet Based Internet Services [20], delivered at the Norwegian University of Science and Technology in March 2006. The main objective of the project work was to survey the effect that different network parameters imposed on the perceived quality of Internet services. This was accomplished through an extensive literature study. In addition, a test setup was suggested that would be suitable for conducting a test on the user-perceived quality of service of a VoD service.

This master's thesis is an independent work and the reader is not expected to be familiar with [20]. However, some background information is omitted from this report due to coverage in [20], information which may enlighten the reader further on some subjects. This is commented on when encountered in the report.

#### State of the Art - H.264/MPEG-4

Today, different VoD service providers employ a variety of different video codecs to encode the content delivered. These include RealNetwork's RealVideo, Microsoft's Windows Media Video, Apple's QuickTime and different implementations of the MPEG standard [36]. However, as was argued for in Section 1.1, the

current marked trend is a biasing toward the H.264/MPEG-4 implementation of MPEG. The main reason for this shift in preferences is the technological superiority of this standard, both in terms of coding efficiency (see Section 2.4.1) and robustness to network impairments (see for example [36, 52, 20]).

Because a subjective test assessing all common video codecs would be to extensive and resource-demanding for the scope of this master's thesis, it was decided to concentrate on the state of the art - the H.264/MPEG-4 video codec.

## 1.4 Structure of this Report

This report is structured as follows:

**Chapter 1** presents the motivation, objective and structure of the report.

**Chapter 2** presents important background theory needed to fully grasp the content of this master's thesis.

**Chapter 3** describes the design and implementation of the subjective test conducted.

**Chapter 4** presents and analyzes the results obtained in the subjective test.

**Chapter 5** concludes the report and gives suggestion on further work.

**Appendices A-D** add source material and documentation not necessary for the immediate grasping of the master's thesis' results, but useful for in-depth analysis and further work.

# Chapter 2

# Background Theory

**Contents**



This chapter describes important background theory needed to fully grasp the concepts in this thesis. It is expected of the reader that he[1] is familiar with basic computer network theory at minimum undergraduate level.

## 2.1 Video on Demand

### 2.1.1 Historical Review and Service Description

Video on Demand (VoD) is the collective term describing services where the end-user (customer) can select and watch video-content over a network, independently of TV-schedules. VoD is sometimes compared to an electronic video

[1] In the remainder of this report, *he* should be treated as *he or she*.

rental store, where the user can watch the content ordered online, either on his computer or on his TV-set [50].

The first commercial VoD service was launched over cable by the Hong Kong Telephone Company in the early 1990's. However, the service was no success, mainly because of high prizes, complicated user-interface, and difficulties getting the public to grasp the concept of pay-per-view. VoD has in general been anything but a success-history when delivered over cable [2].

With the growing success of Internet and the continually increasing access-capacity offered to end-users through xDSL, fiber and other technologies, the VoD marked opened up to other actors than cable-operators. ISPs are constantly seeking new revenue-opportunities in the deployment of new services, and as was shown in Section 1.1, VoD over IP promised a potential huge revenue. This fueled the development of VoD services and it is now possible to choose among a huge number of service providers, both traditional cable operators and Internet service providers, offering video content cheap and on demand[2]. Figure 2.1 shows a typical VoD user interface. The user can browse or search for available content and view information about the video currently playing. When satisfied, he may enter full-screen mode for the best possible experience.


Figure 2.1: User Interface of AOL's VoD Service

[2]See for instance `http://en.wikipedia.org/wiki/Video_on_demand` for an overview of VoD providers.

### Near Video on Demand

VoD allows the user to stop, start and rewind videos at will. Giving the user this ability forces the service provider to transmit a single stream to each customer (unicast). This is however very resource-demanding. Near Video on Demand is a scheme where the provider starts every video, say, every 10 minutes, running the video-content non-stop. By doing this, the service provider may utilize possible resource savings by employing multicast instead of unicast. Near Video on Demand has received a lot of interest from commercial providers in the past, but is now fading in popularity, mainly because of a better understanding of customer-preferences [9].

## 2.1.2 Architecture of VoD systems

The basic architecture of any VoD system consists of three major parts; the client, the network and the server. While each part can be divided into smaller components, they all fit into this model, commonly known as the "client-server" architecture [50]. The client-server architecture is the most widespread network architecture in Internet and is depicted in Figure 2.2.


Figure 2.2: The Client-Server Network Architecture

In the simplest VoD system possible, one computer would act as server, streaming media to a client computer on request through some sort of network. (This is known as *pull VoD*, in opposite to *push VoD* where the server initiates the streaming.) By (possibly) extending the number of clients and adding media archives to supply the servers with content, a basic *centralized* VoD system is achieved. Figure 2.3 illustrates such a centralized system, which is delivering its content through a public IP network, typically Internet.

Centralized systems have the characteristic property of easy managing. However, centralized systems often suffer from poor scalability and possible long delays in the global Internet, especially when the client is located far away from the server. The redundancy offered is small, and link-error may dramatically lower performance. To counter these effect, it is possible to distribute the system to a chosen extent. By adding local media buffers which holds popular content at different physical and logical locations in the network, congestion and delay experienced by central servers can be diminished [33]. Such a distributed approach is shown in Figure 2.4.

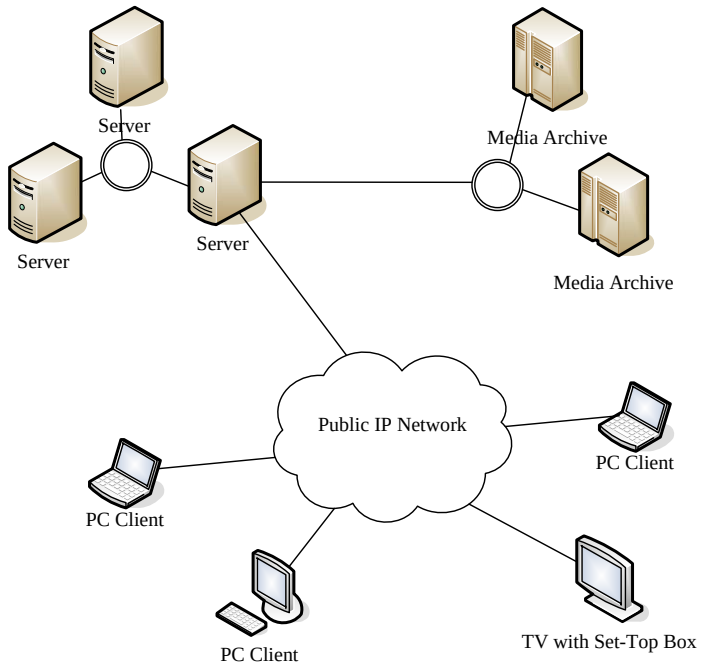



Figure 2.3: A Centralized VoD System

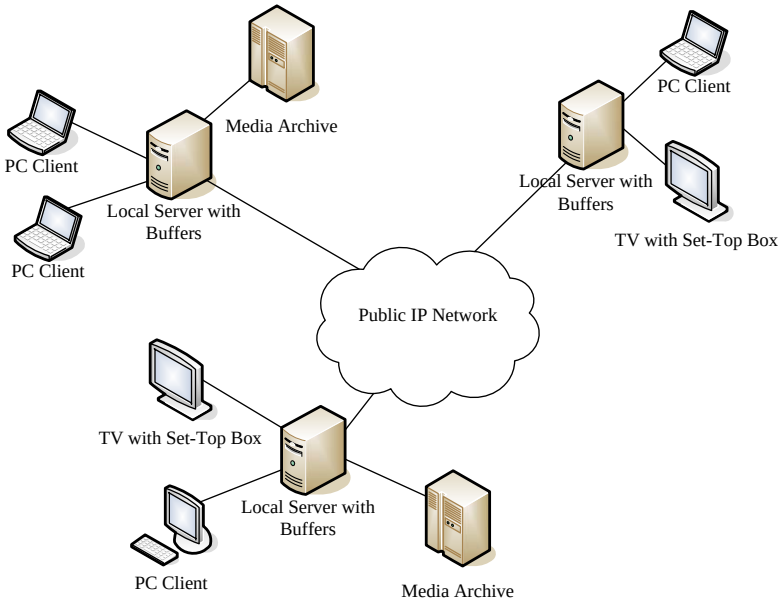



Figure 2.4: A Distributed VoD System

### The VoD Server

In any VoD system, distributed or not, it's heart lies in the server(s). The server controls the storage system, performs admission control and controls the traffic characteristics so to optimize server performance.

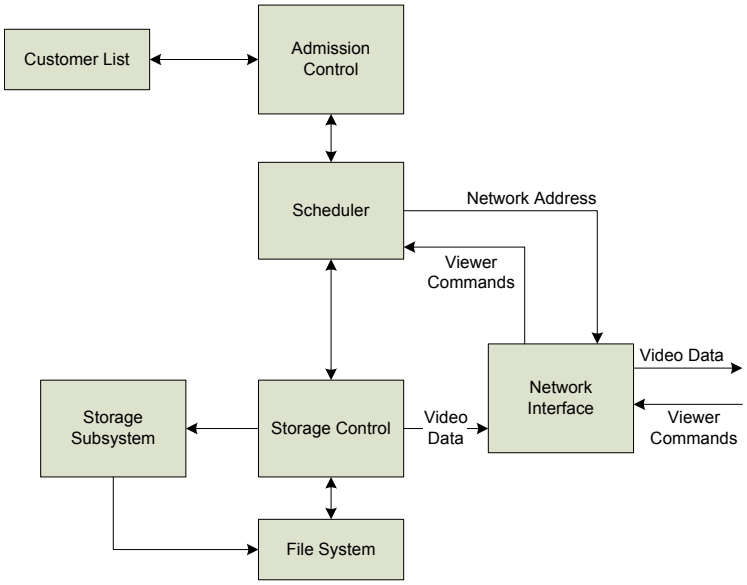



Figure 2.5: Data Flow of a VoD Server [33]

Figure 2.5 shows a possible logical architecture of a VoD server. Any clients who want to use the VoD service, must first of all request and set up a connection. The request is handled by the Admission Control Unit and depending on the user's privileges, the request is granted or not. When the customer requests a certain media content, the Storage Control unit checks if the content is available in the (local) file system. If not, a request to the Storage Subsystem (which may be distributed) is made, and the content is loaded into internal server memory. The streaming is then initiated by the server at the bit-rate specified by the (possible) Service Level Agreement (SLA) between the customer and the service provider [33].

Depending on the customer's preferences and available equipment, the streamed media content is either watched directly on the customer's computer or terminated in set-top boxes connected to a television set. Such set-top boxes decode and prepare the media for TV-display and are, in fact, powerful, specialized personal computers with (usually) hardware-encoders capable of fast real-time decoding [50].

## 2.2 Quality of Service

The term Quality of Service (QoS) has gained new interest and value as new networking technology has been introduced. Multiple definitions exist, each focusing on different areas of a certain field of interest. This is also the case in a communication-oriented context, as is reviewed in the preliminary project work [20]. The consecutive section defines the term as appropriate for the problem scope of this thesis.

### 2.2.1 User-Perceived Quality of Service

Quality of Service can be defined at several different levels in a communication-hierarchy model. Figure 2.6 shows QoS as a term relating to the end-user's perception of the quality only. In other words, QoS is defined as a subjective parameter, not exactly determined by objective metrics in the network.

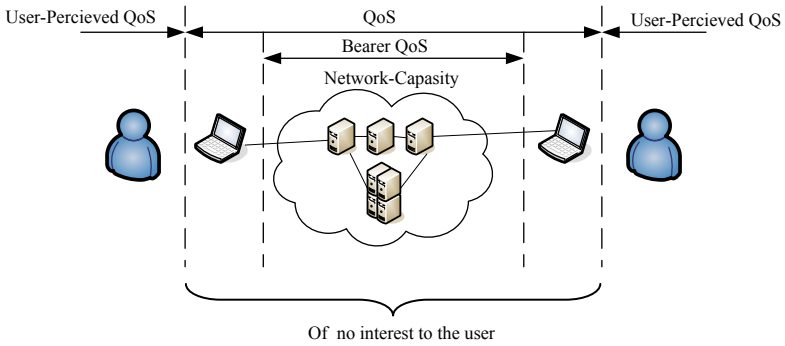



Figure 2.6: User-Perceived Quality of Service

This way of viewing QoS is first encountered in [29], and is known as **User-Perceived Quality of Service**. Such an user-perceived, or subjective, view on QoS is important for numerous reasons. With QoS being a decisive factor for a service and a service providers's success (as discussed is Section 1.1), assessments and knowledge of the perceived quality experienced by the user becomes a valuable resource. Deep knowledge of the user's preferences becomes a competitive advantage, which ultimately may decide whether a service provider survives in the marked or not [2]. With the concept of user-perceived QoS in mind, Quality of Service will in this thesis be defined as follows:

> Quality of Service (QoS) is the degree of end-user satisfaction with the service.

User-perceived QoS is not only interesting from an economic view-point. With network resources being sparse, optimal utilization of carriage capacity is important. With a deep insight of user-perceived QoS it is possible for researchers

and other actors to gain a deep understanding of network threshold values. There is no point in improving, let's say the sending bitrate of a streaming video server, if the increase does not have a positive effect on the end-user's perceived quality [55]. This way of thinking links user-perceived QoS to the metrics commonly used to measure objective QoS. The next section describes such objective QoS-metrics in Internet.

### 2.2.2 Quality of Service in Internet

IP networks, like Internet, are by nature best-effort. Packets sent over these kind of networks don't follow a designated path and are delivered only if the network-capacity is not exceeded. This is in contrast to the public telephone network or cable-TV networks, where data follows predetermined paths and is guaranteed a certain transmission-rate. Multiplexing in IP-networks is statistical, not guaranteeing any consistency in packet loss-rate, delay or delay variation (jitter). That is, Internet is *unreliable* [50].

There are five objective parameters contributing to user-perceived QoS in Internet [17]. Ideally, there should exist, for a certain service, a function $F(param1, ..., param5) = UPQoS$, where $UPQoS$ is some sort of measure of the perceived quality. Section 2.3 discusses the search for such mappings. Next, a brief introduction to these five parameters is given, focusing on packet loss which is the most important parameter in the test described later in this thesis. For a thorough discussion on all parameters, the reader is referred to the preliminary project work, [20].

#### Packet Loss

IP-packets can be lost in transmission due to two main reasons [3]:

**Congestion** When the traffic offered at a certain network-node exceeds the capacity of that node, packets are buffered in queues of limited length. Severe congestion may, because of the limited queue-length, result in queue-exhaustion which in turn leads to packets being dropped. Severe congestion could either mean that the condition is held for a period of time, in which the packet loss rate is said to be *distributed*, or consist of a sudden and short-lived traffic-increase, in which case the packet loss is characterized as *bursty*.

**Error** Errors at the transmission path, or corruption, is another reason for loss of packets. When noisy links etc. modifies the content of a packet, this is usually detected by a link-layer checksum at the receiving end, which in turn discards the packet. Link-error is very rare on high-capacity mediums such as fiber, but more common in wireless environments like IEEE 802.11(x). The latter has become the de-facto standard for wireless access in domestic environments. Because of the increasing popularity of wireless access, losses due to corruption is becoming more common [37].

*Characteristics of packet loss in Internet*

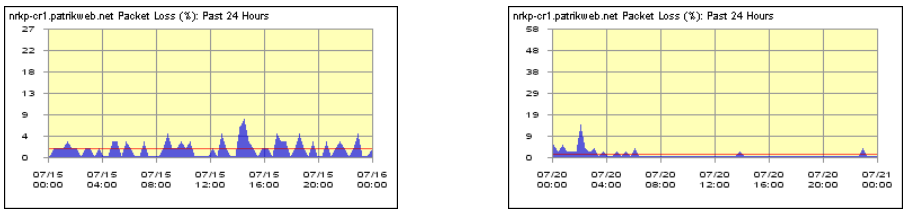
As was noted above, packet loss can either be characterized as distributed or bursty. Below, the characteristics of each type of impairment is summarized. This information was later used to model different realistic usage scenarios in the test described in Chapter 3.

**Distributed packet loss** The global Internet backbone seldom suffers from any distributed packet loss of major extent. Highly utilized links typically average around 0.02%, while a loss rate <0.0001 % is common for others. When the acces-network is included in the equation however, the numbers change. UNINETT, the national supplier of network and network services for Norwegian acedemic institutions, grades distributed packet loss rates according to Table 2.1.

| Average Packet Loss Rate (%) | Grading |
|---|---|
| $< 0.1$ | Good |
| $< 0.5$ | Average |
| $> 0.5$ | Poor |

Table 2.1: Distributed Packet Loss Rate Gradings

For shorter period of times, the distributed packet loss rate may however increase dramatically. Figure 2.7 shows the packet loss rate of a major Swedish network node at two different 24 hours intervals. We see that the packet loss rate averages at 2% and 1% respectively[3]. Such variations are not uncommon, and contribute to the uncertainty of Internet QoS [7].




(a) July 15 - July 16 2006

(b) July 20 - July 21 2006

Figure 2.7: Variation in packet loss rate

**Bursty Packet Loss** Multiple studies show that a considerable amount of packet loss in Internet can be categorized as bursty [40, 10, 5]. This is especially the case on highly utilized links, where router-buffers are likely to be congested. Burst-length vary from network to network, but [10] shows that a burst-length distribution of 5 - 30 consecutive lost packets may be a realistic estimate on an average utilized link.

*On the effect of Packet Loss on Perceived Quality*

Packet loss is the dominant parameter influencing perceived video quality. In addition, studies suggest that a bursty loss environment is preferred over a

[3]Metrics obtained from `http://www.internettrafficreport.com`

non-bursty one. This has to do with the interdependency of consecutive video-frames, especially in MPEG coded video [6]. A distributed packet loss distribution affects a greater percentage of the total video-frames, because the video-stream is not allowed to *settle* after each loss episode.

### Packet Size

For multimedia traffic, like video, knowledge of the distributed packet loss rate is not sufficient to determine the impact of loss on the perceived quality with objective methods. The packet size influences the perceived quality in an indirect way, in that the impact of a lost packet is higher if the packet is larger. A small packet size, however, means that more packets have to be transmitted, which increases the overhead and decreases the throughput [15].

The burstiness of the packet loss pattern is also affected by the packet size distribution [14]. Thus, it is possible to find an optimum packet size in terms of throughput and resistance to packet loss. However, such an optimum size depends both on the video-codec deployed and the underlying physical network.

### Delay

The delay, or end-to-end delay of a network path, indicates the time it takes a packet to travel from the sender's application to the receiver's application. Even though the end-to-end delay may be significant in Internet, it has little effect on the perceived quality of VoD services, as long as it is held constant. The reason for this is that irrespective of the delay conditions in the network, the packets get relatively offset as they traverse the path [7].

### Delay Jitter

Jitter is the variation in end-to-end delay, caused by queuing, contention and serialization effects on the network path from sender to receiver [50]. Buffers in today's video applications have grown to such a scale that the direct effects of jitter is neglectable. Even so, the delay jitter may influence the perceived quality of streamed video to some extent due to possible reordering of packets and an indirect influence on the overall packet loss rate. This issue is addressed in detail in Section 3.1.2.

### Bandwidth

Irrespective of the above listed QoS parameters; the one factor that influences the perceived quality of any video service the most, is the amount of available bandwidth. That is, the bitrate the customer is able to receive through his access-network. As the bitrate increases, so does the overall perceived quality, due to the encoding-mechanisms of video compression. But, the relative effect of packet loss increases as well, due to the added number of packets lost [23]. This

interesting relationship is surveyed and assessed in detail through the subjective test presented in this thesis.

## 2.3 Methods of Subjective Quality Evaluation

Subjective methods of measurements of QoS refer to any method or methodology where a sufficient number of humans assess some specific test sequence under controlled conditions [36]. The process of assessing user-perceived QoS is both time- and resource-demanding. While objective QoS-metrics, like the Peak Signal to Noise Ratio (PSNR), can be calculated, or even simulated, subjective metrics need to be determined through some sort of subjective test.

### 2.3.1 ITU-T Methodologies

The traditional subjective methodologies used in assessment of video-quality are given in ITU-T Recommendation BT.500, Methodology for the subjective assessment of the quality of television pictures [27], and ITU-T Recommendation P.910, Subjective video quality assessment methods for multimedia applications [30].

These methodologies are however subject to some criticism. They all belong in a strict telecommunication tradition and some industry and academic actors criticize the ITU for being too conservative, not addressing the change in user-preferences fueled by the deployment of new multimedia services. The European Broadcasting Union (EBU) has developed new methodologies for testing both multimedia audio and video, and their framework for subjective video quality evaluation, SAMVIQ, has challenged the ITU-models.

The reader is referred to [20] for a review of the methodologies developed by the ITU, while SAMVIQ is described in the following section. Table 2.3.1 sums up the most important differences between the methodologies of [27] and SAMVIQ, and served as guide when deciding on SAMVIQ as test methodology to implement the subjective test in this thesis according to. (See Section 3.1.3 for a more detailed discussion on the choice of SAMVIQ.)

### 2.3.2 SAMVIQ

SAMVIQ, Subjective Assessment Methodology for Video Quality, defined in [35], is a test methodology specially developed by the European Broadcasting Union (EBU) to cope with multimedia content. It was designed to take into account a wide range of codecs, image formats, and network-specific parameters such as bitrate and packet loss. In addition, care was taken to provide for excellent reproducibility and repeatability of the tests, making verification and extending of the test results an easier task than was the case with older methodologies [35].

Any subjective test methodology expects the test-subject to rate individual video clips in accordance with a pre-defined scale. In SAMVIQ the-test subjects

| Parameter | DSIS[a] | DSCQS[b] | SSCQE[c] | SDSCE[d] | SAMVIQ |
|---|---|---|---|---|---|
| **Explicit reference** | Yes | No | No | Yes | Yes |
| **Hidden reference** | No | Yes | No | No | Yes |
| **High anchor** | No | Yes | No | No | Hidden reference |
| **Low anchor** | No | Yes | No | No | Yes |
| **Scale** | Bad to Excellent | Bad to Excellent | Bad to Excellent | Bad to Excellent | Bad to Excellent |
| **Sequence length** | 10s | 10s | 5 min | 10s | 10s |
| **Two simultaneous stimuli** | No | No | No | Yes | No |
| **Presentation of test material** | 1: Once 2: Twice in succession | Twice in succession | Once | Once | Several concurrent (multi-stimuli) |
| **Possibility to change the vote before proceeding** | No | No | No | No | Yes |
| **Minimum accepted votes** | 15 | 15 | 15 | 15 | 15 |
| **Display** | Mainly TV | Mainly TV | Mainly TV | Mainly TV | Mainly PC |

[a]Double Stimulus Impairment Scale
[b]Double Stimulus Continuous Quality Scale
[c]Single Stimulus Continuous Quality Evaluation
[d]Simultaneous Double Stimulus for Continuous Evaluation

Table 2.2: Differences between subjective assessment methodologies [35]

asses the overall video quality of each video clip by assigning a grade ranging from 0-100. With such a fine grained scale, the rating system is in effect continuous. To better compare the scale to the more common discrete ones, it has been divided into five equal lengths as specified in Table 2.3. A mapping to the common Mean Opinion Score - MOS, is included for comparability as well.

The key essence of SAMVIQ is the way the test-clips are presented to the test-subject. The subject can choose when and in what order he wants to view the different video clips. Each video clip is to be rated by the assessor, comparing it to an explicit reference clip rated 100 by definition. In addition to the explicit reference, a hidden reference is included, in order to test the consistency of the test-subject. Each clip, except for the reference and hidden reference, is degraded in some way, for example by adding network-specific impairments such as packet loss. Through some sort of interactive test interface, the subject is

| SAMVIQ grade | MOS-grade | Description |
|---|---|---|
| 80 to 100 | 5 | Excellent |
| 60 to 80 | 4 | Good |
| 40 to 60 | 3 | Fair |
| 20 to 40 | 2 | Poor |
| 00 to 20 | 1 | Bad |

Table 2.3: SAMVIQ rating scale and mapping to MOS

able to view the different clips and rate them according to his preferences. The implementation of such a test interface is described in section 3.6. Figure 2.8 shows a possible test-organization, where the subject can press different access-buttons to view different video-clips subject to different test-conditions. The subject is unaware of which test condition he is assessing at the time.

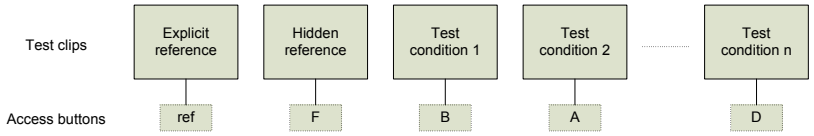



Figure 2.8: A possible test organization in SAMVIQ

SAMVIQ is a new methodology, but is rapidly becoming an important standard used in several subjective assessments. More detailed information on the SAMVIQ methodology is provided in the remaining of this report; both in the methodical process of evaluating the test results (Chapter 4) and in the preliminary test design (Chapter 3).

## 2.4 Video Encoding

One of the key aspects when delivering video over any sort of network, is the encoding-scheme used to encode the raw video data. This is referred to as video compression standards, or video codecs. A video codec enables the use of compression for digital video, usually by some sort of lossy data compression. There is a complex balance between the video quality, the codec's resistance to errors in the distribution network, and the amount of data needed to represent it. Different codecs address this problem in different ways, employing complex algorithm-designs to reduce the bitrate and at the same time minimizing quality degradation [17].

In the academic world there are two major standardization bodies in setting video compression standards [13]; The International Telecommunications Union (ITU)[4] and The MPEG (Motion Picture Engineering Group) Group[5]. Over the years, these two organizations have proposed numerous standards for the encoding and decoding of video content, summarized in Figure 2.9.

[4] `http://www.itu.int/ITU-T/`
[5] `http://www.chiariglione.org/mpeg/`

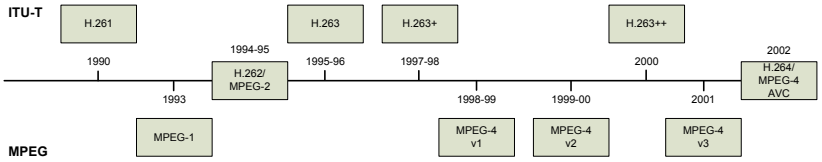



Figure 2.9: Evaluation of Encoding Standards [13]

Below, a brief introduction to MPEG, ITU and the predecessors of H.264/MPEG-4 is given. H.264/MPEG-4 itself is treated in detail in Section 2.4.1

### MPEG

The Motion Picture Engineering Group, MPEG, was established in 1988 as a working group of ISO/IEC. It was charged with the development of international standards for voice and video encoding standards. As of 2006, MPEG has grown to include approximately 350 members from various industries, universities an research organizations.

**MPEG-1:** The initial video format developed by MPEG. It was used by the Video CD format and was designed with the goal of achieving acceptable video quality at 1.5 mbps bitrates and 352x240 resolution. MPEG-1 is the most compatible video format of today, playable on almost all computers and VCD/DVD players. In addition, the popular audio-format .mp3 is derived directly from the audio layer of MPEG-1 [13].

**MPEG-2:** Approved in 1994, MPEG-2 is the current de-facto standard for digital television, typically used to encode audio and video for broadcast applications as satellite- and cable-TV. It is also the encoding format used by the standardized DVD format. MPEG-2 video is similar to MPEG-1 video, but outperforms MPEG-1 at 3 Mbit/seconds and above, making it an obvious choice for bandwidth demanding services such as HDTV [13].

**MPEG-3:** Designed to handle HDTV signals in the 20 - 40 mbps range, MPEG-3 was abandoned when discovered that MPEG-2 could offer similar results through minor modifications.

**MPEG-4:** A newer standard that includes a more modern video encoder than the one used MPEG-2. Designed to work well within a wide range of bit rates and to support different services as streaming media, CD distribution and broadcasted television. MPEG-4 scales well and is able to transport media at any data-rate, from data suitable for delivery over dial-up modems to high-bandwidth HDTV deliverance. Commonly described as offering twice the coding efficiency than that of MPEG-2. MPEG-4 consists of several standards, termed ”parts”, in which part 10 is the equivalent of H.264/MPEG-4 [21].

#### ITU-standards

The ITU standards is primarily targeted at video carriage over low bit rate networks.

**H.261:** The first practical digital video coding standard launched in 1991. Initially targeted at teleconferencing applications and intented to carry video over ISDN. H.261 is a low complexity, low latency video standard optimized for bit rates of nx64 kbps [28]. Even though it is an outdated standard, it is still in use for backward compatibility in video conferencing scenarios.

**H.263:** A similar standard to H.261, but with some improvements and changes to improve both bit rate efficiency and error recovery. H.263+ and H.263++ further improves on these parameters.

**H.264:** Similar to the H.264/MPEG-4 standard described in the section to come.

In addition to ITU and MPEG, several commercial actors have proposed their own standards. This includes Microsoft's Windows Media Video (WMV) and RealNetworks' RealVideo.

### 2.4.1 H.264/MPEG-4

Defined in [32], the H.264/MPEG-4 (Part 10) Advanced Video Coding (commonly referred to as H.264/MPEG-4), is the result of a joint effort between the ITU-T's Video Coding Expert Group (VCEG) and ISO/IEC's Moving Picture Experts Group (MPEG) called the Joint Video Team (JVT). H.264/MPEG-4 is currently the video codec scheme offering the best balance between coding efficiency, implementation complexity and cost, growing in popularity by the day and expected to become the standard of almost all video-streaming solutions within few years [21].

#### Technical Overview

The H.264/MPEG-4 specification is divided into a Video Coding Layer (VCL), which includes the encoding-algorithms, and a Network Abstraction Layer (NAL) which prepares the VCL-representation for transport by above transport layers. The NAL-layer will not be treaded here, and the reader is referred to [32] for details concerning this part of the codec.

*Video Coding Layer*

The video coding layer of H.264/MPEG-4 is in principal not different from the other MPEG-standards, it is the efficiency of the coding that is improved. For a detailed description of the technical concepts of H.264/MPEG-4, please refer to [26, 32, 21, 46]; a high-level explanation is provided below.

Figure 2.10 shows a structural block diagram of the VLC-layer, in which the input video signal is a *macroblock* of 16x16 pixels of the original video picture [46].

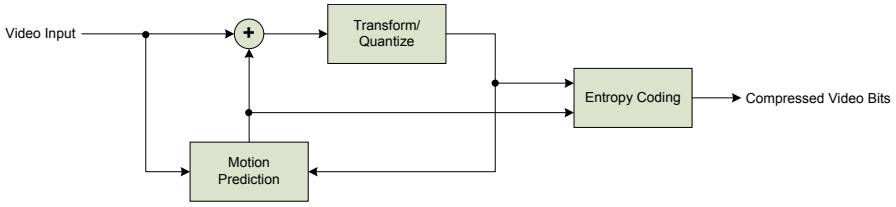



Figure 2.10: The H.264/MPEG-4 Encoding Process

*Motion Prediction/Compression*

The macroblocks are the basic building blocks of the standard, which again are organized in larger *slices*. Each slice represents a larger subset of the *video picture* and is to be encoded and decoded independently. Slices can be categorized by their encoding algorithms as follows [32]:

**I-slice** (Intra-slice) All macroblocks coded without referring to other pictures within the video sequence.

**P-slice** (Predicted-slice) All macroblocks coded with motion-prediction from prior pictures.

**B-slice** (Bi-predicted-slice) All macroblocks coded with a weighted average of prior and future pictures.

The hierarchical order of data blocks in H.264/MPEG-4 may be summarized as: video[picture[slice[macroblock]]], and a possible ordering of pictures is shown in Figure 2.11. Motion prediction on this form is called *temporal* prediction, while internal prediction within I-slices, known as *spacial* prediction, also is employable.

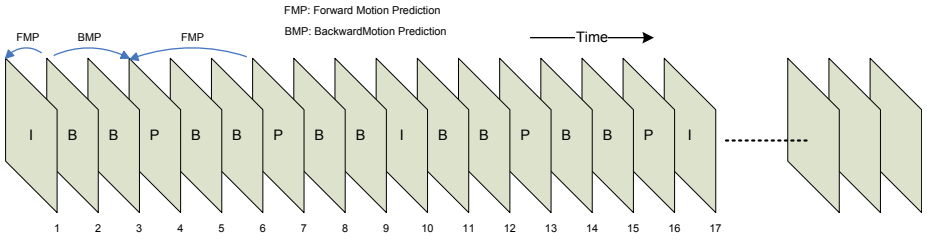



Figure 2.11: Possible ordering of I- P- and B-pictures

While I-slice coded pictures provide a low compression rate, P- and B-pictures substantially contribute to a high compression grade.

*Transform and Quantization*

Referring to Figure 2.10, a mathematical transform is applied to decorrelate the data after motion-prediction. For details, please refer to [21]. In addition,

the coefficients produced by the transform are quantized using a quantization parameter that is independently set for each macroblock, allowing for further optimal data-encoding.

*Entropy Coding*

The last step in the encoding-process is known as entropy coding and relates to the process of assigning codes to symbols so as to match code lengths with the probabilities of the symbols. Entropy coding in itself can only reduce the data size modestly, but in combination with the other predictions, quantizations and transformations, it significantly reduces the data size [21].

### Error Resilience

Due to the design goals of H.264/MPEG-4, which included high error resilience in networks like Internet, the basic implementation of H.264/MPEG-4 includes several tools to overcome lossy and error prone environments. In addition to the basic strength of the design, there are essentially four additional tools which can be employed to further protect the bitstream from network transmission problems [21]:

- Flexible Macroblock Order (FMO)
- Arbitrary Slice Order (ASO)
- Redundant Slices (RS)
- Data Partitioning (DP)

FMO randomizes the data prior to transmission, so in case of a lost packet, the errors are more distributed over the video pictures. This minimizes the chance of relevant adjacent data loss, which could be very quality degrading due to high inter-picture dependence. When ASO is employed, pictures are allowed to arrive out of order, which is especially useful in best-effort networks like Internet. RS adds to the overall resilience by adding redundant representation of pictures, while DP categorizes coded slice data by importance to the picture fidelity [21].

### Profiles and Levels

H.264/MPEG-4 is a complicated standard and contains a wide variety of video coding tools possible to employ. However, depending on the application in question, not every tool is needed. In a network with very little data corruption and loss, for example, complicated error resilience tools are not needed and would only lead to an unnecessarily complex decoder if employed. To deal with such diversified needs, subsets of coding tools, or *profiles* are defined. In the original standard, three such profiles exist[6]:

[6]Several other profiles was defined in the Fidelity Range Extension of H-264/MPEG-4, see for instance [21].

- Baseline profile
- Main profile
- Extended profile

The baseline profile is primary used in videoconferencing applications, the main profile is intended for broadcast and storage applications while the extended profile has a relatively high compression capability and is very error-prone, making it suitable for streaming video-applications.

In addition to profiles, H.264/MPEG-4 defines 16 different *levels*. Levels define the picture size, frame rate, number of reference pictures (I-slice coded) and the maximum compressed bit rate that can be used. For a listing of all levels, see for example [26].

## 2.5 Streaming Video in Internet

VoD-services is a subset of every possible Internet service, and a yet another subset of what may be referred to as streaming services. Streaming services include such diversified services as e-learning, video conferencing, live broadcasting and VoD, and can be categorized by being real-time. Real-time services are services ...

> ... for which time constrains exist between the transmission and the reception and/or between the transmission and the presentation of the data [17].

Real-time services places different demands on the underlying network than non-real time services, and therefore employ a different protocol-stack. This section describes the transport protocols used by streaming real-time services, such as VoD, summarized in Figure 2.12.

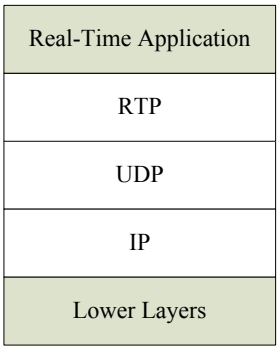



Figure 2.12: The Protocol Stack of a Real-Time Application

### 2.5.1 TCP vs. UDP in Real-Time Streaming Environments

Internet has two widely employed protocols in the transport layer of the Open System Interconnection (OSI) seven-layer network reference model: The Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). TCP is a *connection-oriented* protocol while UDP is *connectionless*[7].

TCP provides a reliable end-to-end byte stream over an unreliable network like Internet. Because the stream is to be reliable, i.e. all data sent over the network is required to reach it's destination, TCP retransmits any lost or corrupted packets. The process of retransmission is rather complex, and involves such mechanisms as slowing down the transmission-rate when the packet loss rate is high (TCP Congestion Control). Such schemes are fine for applications like mail and http, but for real-time streaming services as VoD, a decrease in transmission-rate is not acceptable. A decrease in bitrate would lead to a decrease in the perceived quality, and for multicasted streaming services retransmission to a single source it is not even possible.

UDP, on the contrary, discards lost packets and does not lower the bitrate. To keep the real-rime constraints of real-time streaming application satisfied, UDP is therefore the most commonly employed transport protocol for real-time services. UDP is a very simple protocol; basically IP with a extended header to control multiplexing and error-detection. In particular, there are no special guarantees made about deliverance of packets. Because of this, packet loss may be surveyed as an independent parameter which does not influence the end-to-end delay due to extended buffering in the routers [40].

### 2.5.2 RTP/RTCP

#### Real Time Transport Protocol

The Real-Time Transport Protocol (RTP), defined in [47], provides end-to-end delivery-services for real-time interactive applications. It's basic function is to multiplex several real-time streams onto a single stream of UDP packets. RTP was developed when it was discovered that most real-time applications was reinventing, to a certain extend, the same real-time transport protocol [44].

It is a matter of definition where to put RTP in the protocol stack. The multimedia application deploying RTP feeds it's audio and video streams into RTP's library, which in effect belongs to the application itself. This library then multiplexes the streams, encodes them in RTP-packets and delivers them through a socket-interface which again embeds the RTP-data in UDP-packets. This design suggests that RTP is an application protocol. But, on the other hand, it is an application-*independent* protocol that basically provides transport facilities, which suggests it is a transport protocol. A possible description would be that it is a transport protocol implemented in the application layer of the OSI-model [50].

[7] For an introduction to Internet Protocols and the OSI-model, see for example [50].

RTP gives each packet sent in the RTP-stream a number-identifier one higher than it's predecessor, which allows the receiver to determine if any packets are missing. If so, the receiver could approximate the missing data by interpolation. Retransmission is, as for UDP, not an option because the retransmitted packet most probably would arrive too late for it's playing-time. Each RTP packet may by timestamped, to allow the receiver to do a small amount of buffering, and to play each individual sample the right number of milliseconds after the start of the stream. This timestamping is also useful for the synchronization of voice and video [44].

#### Real Time Transport Control Protocol

The Real Time Transport Control Protocol (RTCP) is an embedded part of RTP and handles feedback, synchronization and the user interface. But, it does not transport any data. RTP control is achieved by periodically transmitting and receiving RTCP packets to and from all recipients of the media content. RTCP adds to the functionality of RTP through the following mechanisms [47]:

**QoS Feedback** Provides feedback on QoS parameters as packet loss rate and delay time. Enables sources to adjust their sending settings in reply to the change in network QoS conditions.

**Canonical Name** Adds a persistent transport-level identifier for each RTP source.

**Rate Adjustment** Makes RTP scalable when the number of participants of a multicast session is large.

**Session Control Information** Provides session control information.

### 2.5.3 MPEG Transport Stream

The H.264/MPEG-4 encoding format has, as described earlier, been accepted by the majority of service providers as the video codec to use when streaming media content through Internet. In spite of this agreement, no common specification for the transport of H.264/MPEG-4 over IP networks exists today [38].

Because of the experience with the MPEG-2 format, some service providers and application developers have utilized MPEG-2 Transport Streams (TS), described in [31], for the carriage of MPEG-4 data. This is for example the case with the VideoLAN Streaming Server which was used to stream MPEG-4 data in the subjective test performed in this thesis.

The MPEG-2 Transport Stream is so called, to signify that it is the input to the Transport Layer in the ISO network reference model. It is not, in itself, a transport layer protocol and no mechanism is provided to ensure the reliable delivery of the transported data. Such services are handled by the underlying layers, typically RTP when deployed.

The primary task of a Transport Stream is to multiplex data, typically audio and video, as well as possible subtitles and associated program information. The Transport Streams are composed of 188 byte TS Packets with an associated 4 byte header. In addition, some TS packets may contain an optional Adaption Field, making the actual size of a TS Packet somewhat varying.

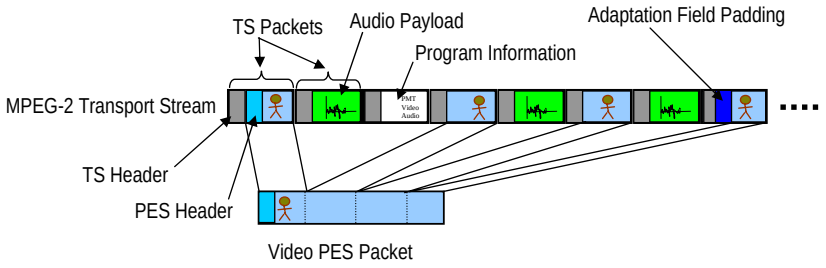



Figure 2.13: MPEG TS multiplexing video, audio and program information [38]

The actual media data is contained in individual Packetized Elementary Streams (PES) broken into 184 bytes pieces to fit into the TS Packet payload. Each PES packet contains one type of media-stream, either video, data or other associated control data. Figure 3.5.2 of Section 3.5.1 shows how the MPEG TS packets are encapsulated by lower network protocols, while Figure 2.13 of this section shows the possible configuration of a MPEG TS stream.

# Chapter 3

# Test Setup

## Contents

This chapter describes the test setup designed to survey the perceived quality of a VoD-application. Section 3.1 defines the scope of the test and argues for the choice of methodology and test parameters. Section 3.2 describes the values of the test parameters, while Section 3.3 discusses the choice of test sequence. In Section 3.4 and 3.5 it is explained how the test sequence was encoded and degraded into individual test clips. Section 3.6 describes the implementation of a test interface in Java, while Sections 3.7 and 3.8 describes the physical environment and how he test was conducted.

## 3.1 Main Principles

### 3.1.1 Questions to be Answered

There are many possible approaches on how to conduct a subjective assessment of perceived video-quality. This report arranges for the study of the effect of packet loss across different limited bandwidth-capacities. The main goal of the test is:

> To survey the perceived degradation-effect of packet loss on a typical VoD-service delivered to the end-user through the Internet. The media-content is to be encoded with the H.264/MPEG-4 video coding standard.

With a typical VoD-service it is understood a service where the end-user is able to gain access to a selection of video-content stored by the service-provider on centralized or distributed media-servers. The content is streamed (not downloaded) by the end user through the common Internet; that is, not only trough an internal network. In addition, the service should be available to all Internet-subscribers, independently of Internet access-providers. However, the service provider may be able to deny users with an insufficient network-connection access to the service.

There are numerous examples of such services as described above. In Norway, NRK's free of charge WEB-TV[1] or the commercial VoD-provider SF Anytime[2] both fit the description. In the USA, ABC is offering it's most popular television series through a similar solution[3]. The latter has gained huge interest, both from the media and the public[41].

The main goal is further divided into the following set of questions which is to be answered in the context described above:

- Does packet loss influence the perceived quality of H.264/MPEG-4, and if so, to what extend?

[1] `http://www1.nrk.no/nett-tv/forside`
[2] `http://www.sf-anytime.com/`
[3] `http://dynamic.abc.go.com/streaming/landing`

- If the packet loss rate is kept constant, how does an increase in the available bandwidth affect the perceived quality?
- Is there a difference in the perceived quality when the packets are dropped in a burst compared to an unbursty loss rate?

### 3.1.2 Omitting the Delay and the Delay Jitter

Even though delay and delay jitter are among the five Internet specific issues associated with the quality of received media-content (See section 2.2.2), the emphasis of this test is on the effect of packet loss and available bandwidth. The reasons for concentrating on losses, while ignoring delay aspects, are basically three:

A constant end-to-end delay does not influence the end-users perception of quality when dealing with non interactive real-time media streaming services. The reason is, of course, that a constant delay affects all sent packets the same way, resulting in no re-ordering of packets [17].[4] A large end-to-end delay may be an indicator of congestion in the network-path, and is therefore linked to the packet loss rate through their common shared causes [50]. However, it is not a direct source of quality-deterioration, and is therefore not included in this test.

The delay jitter may cause packets to arrive at the receiver end both out of order and after their play-out time has expired. As these packets are discarded, this is an indirect cause of loss. We can therefore say that the effect of jitter can be directly mapped to packet loss [51]. A good introduction to the quantitative correlation between delay and loss is given in [42], and will not be addressed in this report.

Today's video streaming application buffers are in addition sufficiently large to accommodate the majority of network delays [23], even though these (sometimes) over-dimensioned buffers contribute to slow setup and channel-change time [11]. Because end-users tend to compare real-time video services as VoD with traditional TV-services where setup and channel-change time are not noticeable, the buffer size may in itself affect the perceived quality. This kind of discussion is however outside the scope of this test. It is sufficient to assume that buffers minimizes the direct effect of delay and that any excess jitter and delay maps to packet loss, which is surveyed.

### 3.1.3 Methodology to be Used - SAMVIQ

In Section 2.3, an introduction to the most commonly employed subjective assessment methodologies was given. Table 2.3.1 sums up the findings in a comparative manner, making it possible to make a considered decision on which methodology to use in this test. For our assessment, **SAMVIQ** was chosen as test methodology, and the test setup will follow the recommendations made by

[4] When dealing with interactive services such as videoconferencing, this added delay may on the other hand depreciate the quality, by adding pauses in the conversation.

EBU in [35], SAMVIQ - A new EBU Methodology for Video Quality Evaluations in Multimedia.

#### Why Choose SAMVIQ?

Even though the SAMVIQ methodology initially was developed for multimedia codec comparison, it is also able to cope with real-time multimedia issues such as packet loss and bandwidth-restrictions [35]. The most fundamental difference between SAMVIQ and the other methodologies presented in 2.3, is the way test clips are presented to the test subjects. In the SAMVIQ methodology, the user can choose the order of test clips and correct their votes, as appropriate. This approach minimizes the effect of premature decisions due to lack of concentration. Because of this uncontinuous way of presenting the test clips, it is believed that the SAMVIQ methodology is able to offer higher reliability and smaller standard deviations than the other methodologies [36].

The SAMVIQ methodology also differs with respect to user-interaction. While others methodologies passively present the test-clips to the user, SAMVIQ requires an active user, making the voting-procedure a more interactive process. With this approach, the SAMVIQ methodology forces the test-subject to be more active, decreasing the percentage of decisions made by accidental occurrence.

To further enhance the validity of the final results, SAMVIQ is the only methodology that introduces both an explicit and a hidden reference clip. By comparing the vote given to the hidden reference with the high anchor vote set by definition on the explicit reference, a measure on the consistency of the test-subject can be achieved. This measure may be used to remove test-subjects that don't vote consistently.

The above reasons, joined by the fact that SAMVIQ is the only methodology that is specially developed for multimedia-content displayed on a computer-screen, tipped the decision on witch methodology to use in SAMVIQ's favor.

### 3.1.4 The Subjective Quality Evaluation Process

SAMVIQ divides the process of subjective quality evaluation into eight simple steps, listed below[35].

1. Select the parameters under test (See section 3.2)
2. Select the test sequence(s) (See section 3.3)
3. Define the degradation settings (See section 3.2)
4. Degrade the test sequence(s) (See section 3.5)
5. Select the evaluation methodology and establish a reproducible test environment (See section 3.7)

6. Organize test sessions, invite test subjects, present them with the degraded test sequences and ask them to determine the quality as they perceive it (See section 3.8)

7. Collect the evaluation results, perform some statistical analysis on the voting data and remove inconsisting subjects (See chapter 4)

8. *Publish a test report*

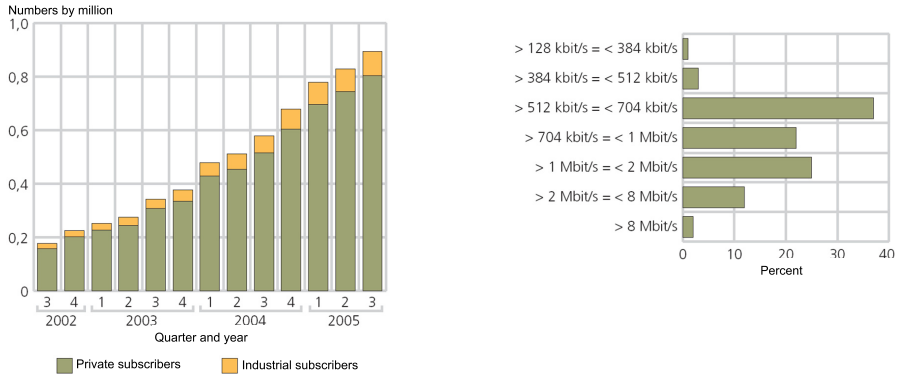
Items one through seven are handled explicitly in this report. Item eight is commented on in Section 5.3, Further Work.

## 3.2 The Test Parameters

The two test parameters in this test are the *packet loss rate* and the amount of *available bandwidth* or *bitrate*. These two parameters could be chosen to reflect several possible usage scenarios, recreating the conditions of different usage patterns. In coherence with the discussion in Section 3.1, however, a typical domestic usage scenario will be replicated.

### 3.2.1 Domestic Usage Pattern Scenario

VoD is a service with a growing appeal to the mass marked. Through the Internet, members of normal households are able to view multimedia content independently of TV-schedules, either on their personal computer or on their TV-set via a set-top box.




(a) Number of Subscribers

(b) Distribution of Available Bandwidth

Figure 3.1: Norwegian broadband statistics, 3rd quarter 2005 [24]

Figure 3.1 shows some statistics on the Norwegian broadband access growth[5]. The annual number of broadband-subscribers have increased dramatically since

[5]Event though this is Norwegian numbers, according to [24], the same trend can be spotted in most OECD-countries, see e.g. [43].

2002. and studies (e.g [24]) predicts this trend to continue. The distribution of available bandwidth is also observed, which averages in the 512 kbps - 2 mbps interval. It is, however, expected a substantial biasing toward the high-end of this scale in the future years, mainly due to lowered prices as a result of improved technology [12].

Armed with this knowledge it is possible to define realistic usage scenarios which later is to be recreated in the test setup. In these scenarios we therefore assume households with broadband Internet Access through ADSL, provided by common ISPs.

Table 3.1 sums up the different values used for the QoS parameters during the testing.

| Usage Scenario | Bandwidth (kbps) | Random Packet Loss (%) | Bursty Packet Loss (# of pkts. lost in each burst) |
|---|---|---|---|
| 1 / 13 / 25 | 800 / 2048 / 5000 | 3 | - |
| 2 / 14 / 26 | 800 / 2048 / 5000 | 2 | - |
| 3 / 15 / 27 | 800 / 2048 / 5000 | 1 | - |
| 4 / 16 / 28 | 800 / 2048 / 5000 | 0,75 | - |
| 5 / 17 / 29 | 800 / 2048 / 5000 | 0,5 | - |
| 6 / 18 / 30 | 800 / 2048 / 5000 | 0,25 | - |
| 7 / 19 / 31 | 800 / 2048 / 5000 | 0,1 | - |
| 8 / 20 / 32 | 800 / 2048 / 5000 | 0,05 | - |
| 9 / 21 / 33 | 800 / 2048 / 5000 | 0,01 | - |
| 10 / 22 / 34 | 800 / 2048 / 5000 | - | 30 |
| 11 / 23 / 35 | 800 / 2048 / 5000 | - | 20 |
| 12 / 24 / 36 | 800 / 2048 / 5000 | - | 10 |

Table 3.1: Test scenarios and parameter values

### Bandwidth Limitations

Usage scenarios 1 through 12 model an Internet-connection with a relatively low bandwidth-capacity (800 kbps). As can be seen from Figure 3.1, this is not an unlikely scenario. However, end-users who are interested in multimedia content (e.g. VoD) may not be satisfied with such a limited rate. Usage scenarios 13-24 are intented to model such users. Here the available bandwidth is the commonly offered (by June 2006) 2048 kbps link[6]. It is also possible to think of scenarios 1-12 as a 2048 kbps link experiencing bandwidth-drops due to multiple users[7].

Usage scenarios 25-36 model a user with a very good Internet-connection (5000 kbps). This amount of available bandwidth is not common today, nor possible everywhere. It is on the other hand likely to be a normal rate in the near future. In addition, 5000 kbps is close to the threshold value assumed to be used in HD-

[6]See for instance `www.telenor.no`

[7]Even though the drop in available bandwidth may be temporarily, it may affect the video-stream continuously due to congestion at the time of stream-setup negotiation [18]

TV delivered over xDSL by Norwegian operators[8], and is thus a value not likely to be exceeded for video transmission any time soon.

#### Packet Loss Rates

In Section 2.2.2, the characteristics of packet loss in Internet were reviewed. Based on these statistics it was decided to apply unbursty packet loss rates from 3% and downwards. Due to the short length of the test-clips (11 sec) and consequently limited number of packets, the minimum packet loss rate was set to 0,01% (Discussed in detail in section 3.5).

The bursty loss profiles, also based on the findings in Section 2.2.2, were defined as: Lose $n$ packets in every burst, $n \in \{10, 20, 30\}$, with two bursts per 11 second video clip.

To gain an understanding of how similar packet loss rates affect the perceived quality on different bitrates, the same packet loss profiles were combined with each bitrate, adding up to a total of 36 usage scenarios to be tested.

## 3.3 The Test Sequence

The choice of test sequence(s) is crucial to the success of any test. It should meet the demands of the SAMVIQ methodology and be an ordinary video clip not designed by any codec-manufactures to optimize their coding algorithms and technologies.

### 3.3.1 The Original High Definition Clip

When selecting a video clip to use as test sequence, the degree of motion is a crucial factor. Most video-codecs, H.264/MPEG-4 especially, works by some sort of motion-prediction (See Section 2.4) to compress the data. In general, more and faster motion place higher demands on the codec. In some tests it is customary to use several different video-clips with different motion-profiles, each with the same impairments added, in order to address this issue[9]. Due to the large number of usage-scenarios defined in this test, it was however decided to use only one clip. If more clips was to be tested, the length of the test would exceed the recommendations made in the SAMVIQ methodology on test-length.

The chosen clip has a very high degree of motion, mainly because scenes with a high level of motion often is the main factor of annoyance for a viewer of compressed video [53], but also since it puts the highest demands on the codec. Table 3.2 sums up the technical data, while Figure 3.2 depicts frame number 172/275. A compressed version of the clip is available for download at `http://folk.ntnu.no/flo/wedding.mp4`.

[8] A joint effort between TV2 and NRK assumes a maximum bitrate of 5000 kbps for the delivery of H.264/MPEG-4 content through Internet.

[9] This is especially important when comparing different video-codecs.

Figure 3.2: The Original Uncompressed Video Clip

| Parameter | Value |
|---|---|
| **Name** | Wedding |
| **Format** | YUV |
| **Resolution** | 4CIF (704x576) |
| **Length** | 11 s |
| **# of frames** | 275 |
| **Frame rate** | 25 fps |
| **Description** | Newlyweds arriving at a party. Many people, heavy movement, highly detailed |

Table 3.2: Technical Data - Original Clip

## 3.4 Encoding of the Test Sequence

To encode the reference clip into the MPEG-4/H.264 file format, a multi-stage process as shown in Figure 3.3 was carried out.

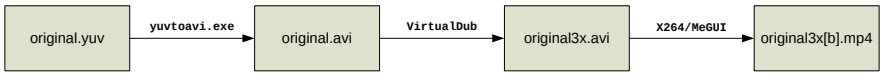



Figure 3.3: Encoding Process

Most of the test sequences used in image coding communities are distributed in YUV DA/601 format, which is a professional lossless format without any compression. This is also the case for the reference clip used in this test. To make the video file compatible with Windows-based codecs and utilities, the clip was converted to uncompressed AVI using the publicly available program, `yuv2avi.exe`, from StreamCrest[10].

The sequence was further extended to consist of three identical replicas of it self. Because a codec needs time to stabilize, this ensured that when encoding the sequence, the last replica would be be a good representation of an, in theory, infinite long encoding process [53]. The last repetition was later cut from the sequence and used as material in the test. VirtalDub[11], an open source solution for video editing was used to accomplish this task.

The encoding itself was carried out using `x264`[12], a non-proprietary H.264/MPEG-4 implementation ranked #1 in MSU's annual H.264/MPEG-4 Video Codec Comparison [52]. To control x264, the open source encoding frontend `MeGUI`[13] was used. The sequence was encoded with Baseline Profile and one added B-frame, according to QuickTime standards. The target bitrate was set to 800 kbps, 2048 kbps and 5000 kbps respectively, and both picture format (4CIF) and framerate (25 fps) was kept as the original clip. The obtained bitrates are summarized in Table 3.3. The complete encoding-logs from `x264` are shown in Appendix A.

| Target Bitrate (kbps) | Obtained Bitrate (kbps) |
|---|---|
| 800 | 801,8 |
| 2048 | 2047,0 |
| 5000 | 5011,2 |

Table 3.3: Target Bitrates vs. Obtained Bitrates

[10] `http://www.streamcrest.com`
[11] `http://www.virtualdub.org`
[12] `http://developers.videolan.org/x264.html`
[13] `http://megui.sourceforge.net/`

## 3.5 Degradation of the Test Sequences

When the original reference clip was extended and encoded, the next step was to degrade the clip further, reflecting the different network conditions described in the usage scenarios. This section describes this process. The reader is referred to Chapter 2 for extensive explanation of the technical terms encountered.

### 3.5.1 Modeling a VoD System

As described in Section 2.1, a VoD system is in principle a client-server system running over some sort of network. To model this three-component system, the following equipment was chosen, and organized according to Figure 3.4:

- **Server** - VideoLAN Media and Streaming Server
- **Network** - Empirix Packetsphere Network Emulator
- **Client** - VideoLAN Media and Streaming Server

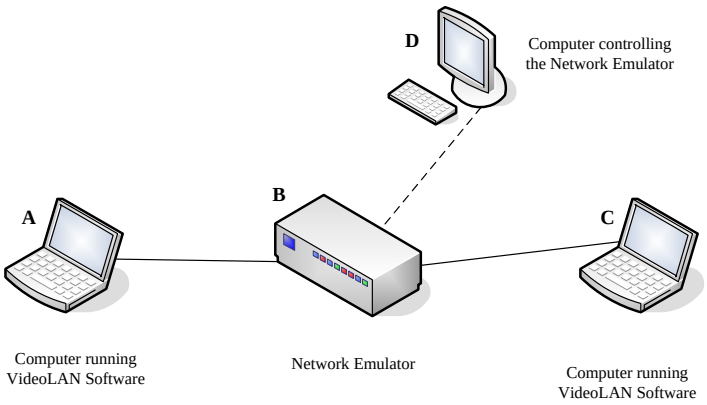
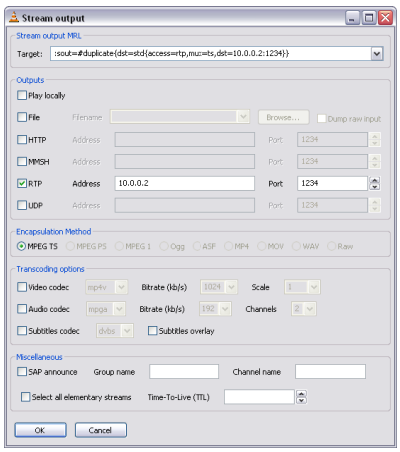



Figure 3.4: Logical Setup - the Degradation Process

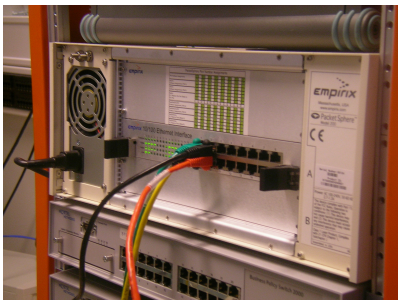
#### Equipment

VideoLAN[14] is a complete open source solution for video streaming, developed under the GNU General Public License. Through it's interface (Figure 3.5(a)) it is possible to stream media-content in a variety of ways, depending on the format of the media-content. The Empirix Packetsphere Network Emulator (Figure 3.5(b)) is a hardware network emulator that is capable of adding typical packet network-impairments to traffic routed through it. It's main goal is to provide a way to efficiently emulate the many possible conditions in unreliable packet networks, such as environments with a heavy packet loss rate.

[14] `http://www.videolan.org/`




(a) The VideoLAN interface




(b) The Empirix Packetsphere Network Emulator

Figure 3.5: Equipment used in modeling the VoD system

### The Streaming Process

When streaming any MPEG-4 content with VideoLAN, H.264 included, it is mandatory to multiplex the media-content into a MPEG Transport Stream (MPEG TS), as described in Section 2.5.3). There are two methods currently utilized for the carriage of MPEG TS over IP and applicable in VideoLAN; *MPEG TS over UDP over IP* or *MPEG TS over RTP over UDP over IP*.

Even though the latter method adds more redundant information through an extra level of packetization, it is the most recommended. The Digital Video Broadcasting Project (DVB)[15], for example, specifically warns users not to make use of the direct UDP transport mechanism [38]. It was therefore decided to use the protocol stack MPEG TS/RTP/UDP(/IP), transported over Ethernet, as specified by the IETF in [25] and by DVB in [16].

The streaming-process in VideoLAN is quite straightforward. Referring to Figure 3.4, computer A(IP:10.0.0.1) was set up to stream to computer C(IP:10.0.0.2) on port 1234, manually routing the stream through the network emulator, B. The actual command line parameter passed to VideoLAN to ensure that the media-content was streamed with the proper protocol-stack employed was:
`:sout=#duplicate{dst=std{access=rtp,mux=ts,url=10.0.0.2:1234}}`

Correspondingly, the client (C) was set up to receive the stream, dumping it's content raw to disk for further processing. While the IP/UDP/RTP headers are automatically removed by VideoLAN and lower network peripherals, the MPEG TS headers are still appended. This means that the stream is still multiplexed, calling for later remultiplexing into playable MPEG-4 files (See Section 3.5.4).

[15] `http://www.dvb.org/`

### 3.5.2 Modeling Packet Loss Behavior

#### Packet Size

The Local Area Network at NTNU is Ethernet-based, and such networks have a Maximum Transmission Unit (MTU) of 1500 bytes[50]. MPEG TS Packets each have a payload of 188 bytes (included a 4 byte header). This implies that $1500/188 \approx 7$ MPEG TS packets fit in one MTU. When we add to this the header size of RTP, UDP and IP (Table 3.4), we can estimate the IP packet-size: 7*188 + 12 + 8 + 20 = 1356 bytes (Figure 3.5.2). This calculation ignores the headers and adaption field sizes added from the packetized elementary stream (PESs), since these are not evenly distributed amongst the MPEG TS packets (See Section 2.5.3 for a detailed discussion). We can thus expect a slightly larger packet size than 1356 bytes. This corresponds with Table 3.5 which shows the observed metrics when video was streamed in the test setup with no packet loss added[16].

| Packetization | Header Size (bytes) |
|---|---|
| IP | 20 [a] |
| UDP | 8 |
| RTP | 12 |
| TS | 4 |
| PES | 8, 13 |

[a] An option exists within the IP header that allows further optional bytes to be added, but this is not normally used [50].

Table 3.4: Header Size for Various Packetization Methods [38, 50]

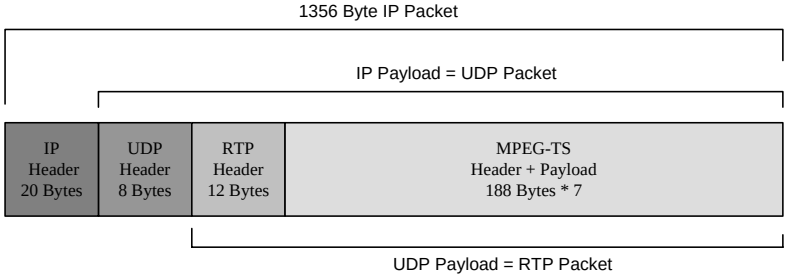



Figure 3.6: IP Packet Size Calculation

| Encoded at | Bytes streamed | # packets | Avg. packet size (bytes) |
|---|---|---|---|
| 5000 kbps | 21 589 062 | 15 851 | 1 362 |
| 2048 kbps | 9 042 406 | 6 639 | 1 362 |
| 800 kbps | 3 735 966 | 2 743 | 1 362 |

Table 3.5: Observed Statistics when Streamed With No Packet loss

[16] Metrics obtained from the network emulator's log files

### Unbursty Packet Loss

Packet loss can be categorized in several ways. The most widely used model to describe unbursty or random packet loss is the *Bernoulli* or *Independent* Model. Here, a packet is lost with a probability $p$, and received with a probability $1-p$. The Bernoulli model is illustrated in Figure 3.7, where State 0 denotes "received packet" and State 1 denotes "dropped packet".

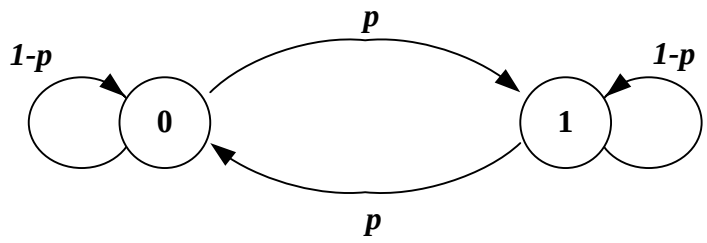



Figure 3.7: The Bernoulli Loss Model

When modeling packet loss behavior, a random distribution may however be problematic. Because different packet loss rates will be applied to different video clips, a random model is useless unless it *guarantees* the desired packet loss rate, at least to some extent. To see this, consider the following argument [19]:

Let the random variable $X$ be the number of packets lost in $n$ independent Bernoulli trials. $X$ then follows a binomial distribution with mean

$$\mu = n \cdot p \tag{3.1}$$

and variance

$$\sigma^2 = n \cdot p(1-p) \tag{3.2}$$

To obtain a statistical coherent model of the packet loss rate, we need $n$ to be big enough that we with a certain probability can say we have reached a rate of $p$ percent. If $n$ is large enough and the skew of the distribution is not too great, then an excellent approximation to the binomial distribution is given by the normal distribution [54]. To determine the minimum value of $n$, we can use the sample size formula based on this normal approximation to the binomial distribution [8]. This gives

$$n = \left(\frac{2Z_{\alpha/2}}{w}\right)^2 p(1-p) \tag{3.3}$$

where $Z_{\alpha/2}$ is the upper $100(1-\alpha/2)$ percentile of the normal distribution, $w$ is the width, or margin of error, and $n$ is the sample size required. We would

in this test like $w$ to be 1 percent of $p$, that is; we allow a margin of error $w = p/100$. This gives

$$n = \frac{40000 Z^2_{\alpha/2}(1-p)}{p} \tag{3.4}$$

Table 3.6 shows the number of packets that need to be streamed to obtain a certainty of 95% that the desired packet loss rate is reached within the defined 1% margin of error.

| Random packet loss rate (%) | Required number of packets sent |
|---|---|
| 3 | 4 968 470 |
| 2 | 7 529 536 |
| 1 | 15 212 736 |
| 0,75 | 20 334 870 |
| ⋮ | ⋮ |
| 0,01 | 1 536 486 336 |

Table 3.6: The Required Number of Packets Sent (Sample Size), Random Packet Loss Rate.

The number of packets sent in the 5000 kbps case was 15 851 (Table 3.5). We see that this is well below the required numbers. This implies that it is impossible to make a statistical consistent random model of packet loss when dealing with such short video-clips as is the case in this test. It was therefore decided to use a constant model for modeling unbursty loss-patterns.

The most fundamental of all constant models is supported by the Packetsphere Network Emulator, and can be described as: *Lose every nth packet.* For a given packet loss rate, $p$, $n$=$1/p$. For the usage-pattern scenarios with unbursty packet loss, this model, illustrated by Figure 3.8 was deployed.

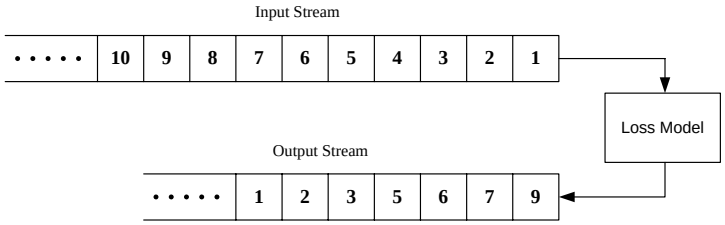



Figure 3.8: Lose every $n$th packet loss model. $n = 4, p = 0,25$

Even though a constant model is a very simplistic model, assuming no temporal correlation between consecutive losses and deploys a constant inter-loss rate, it is well suited for this subjective test. The primary goal of this test is to investigate the effect of packet loss on the perceived quality. Keeping this in mind, a somewhat simplified reality could be accepted, as is suggested in [51] and [23]. An added positive effect is the improved verification- and extension-possibilities by other researchers.

#### Bursty Packet Loss

A general model describing bursty packet loss is depicted in Figure 3.9. This is called a Gilbert Model and is a simplified version of a burst model suggested by E. Gilbert in [22]. This simplified model is widely used in the literature, as is documented in [51]. In the model, State 0 denotes "received packet" and State 1 "dropped packet". $p$ is the probability that a packet will be lost, given that the previous packet was received. $q$ is the probability that a packet will be lost, given that the previous packet was lost. $q$ is also known as the *conditional loss probability, clp* while the probability of being in State 1 is called the *unconditional loss probability, ulp*. It is given by $ulp = p/(p + 1 - q)$, and serves as the most common measure of the average packet loss rate when in a bursty environment [13].

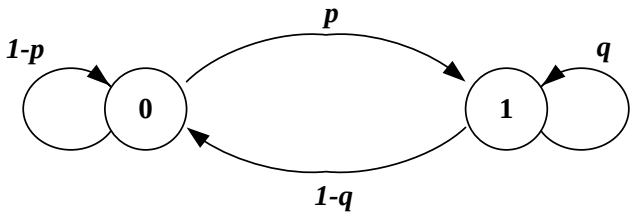



Figure 3.9: Gilbert burst model

The Packetsphere Network Emulator, however, ships with only one bursty loss model. This model does not fit the profile of the Gilbert burst model.

We want a loss model where $n$ packets is consecutively lost in every burst, not a conditional probability of consecutive loss. To address this kind of loss-profile, we need to modify the Gilbert model. This new model can be illustrated with the Bernoulli loss model depicted in Figure 3.7. In this case, State 0 denotes as before "received packet". State 1, however, now describes the consecutive loss of $n$ packets. $p$ is here the probability that $n$ packets will be lost, given that the previous packet was received. The random overall loss rate, $y$, can then be defined as $y = pn$, whereupon the unconditional loss probability, ulp, is given by $ulp = \frac{y}{n} = p$

### 3.5.3 Adding the Packet Loss

When the loss models of the previous section was decided on, the next step was to implement them in the network emulator and carry out the physical degradation process.

#### Unbursty loss

For the unbursty loss model, the implementation in Packetsphere was straightforward. Figure 3.10 shows how the graphical user interface was set up to model a constant loss rate of $p = 3\%$.

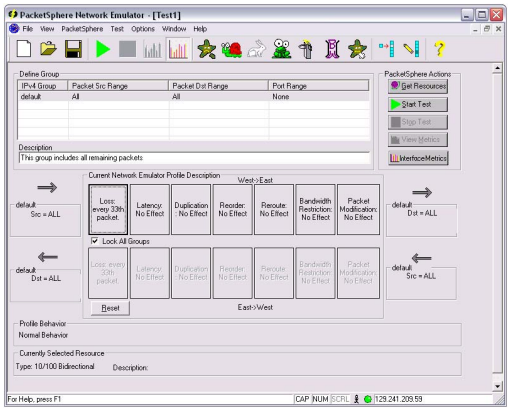



Figure 3.10: The network emulator set up to simulate a packet loss rate of 3%

Table 3.7 shows the number of lost packets for the different usage scenarios employing unbursty loss. The metrics were taken from the network emulator's log files.

| | 3% | 2% | 1% | 0,75% | 0,5% | 0,25% | 0,1% | 0,05% | 0,01% |
|---|---|---|---|---|---|---|---|---|---|
| 5000 kbps | 480 | 317 | 158 | 119 | 79 | 39 | 15 | 7 | 1* |
| 2048 kbps | 201 | 132 | 66 | 49 | 33 | 16 | 6 | 3 | 1* |
| 800 kbps | 83 | 54 | 27 | 20 | 13 | 6 | 2 | 1* | na |

Table 3.7: Number of lost packets, unbursty packet loss

Because of the limited video file sizes, the 0,01% packet loss rate was only applicable for the 5000 and 2048 kbps cases. Because the usage scenarios defined earlier calls for such a loss rate even for the 800 kbps case, usage scenario #9 was modeled as the hidden reference clip, that is with no packet loss added. In addition, when only one packet was lost (marked with an asterix in table 3.7), it was ensured that this packet was lost in the last third of the video file, making sure it's effect was noticeable when the files were later trimmed to original length. The effect of the degradation is illustrated in Figure 3.11.

(a) the Reference Clip

(b) added 1% unbursty packet loss

Figure 3.11: The effect of degradation, 5000kbps

### Bursty Loss

It was a somewhat trickier task to employ the bursty loss model, than is was to employ the unbursty one. Figure 3.12 shows the dialog box where the burst parameters is set in the Packetsphere Network Emulator.

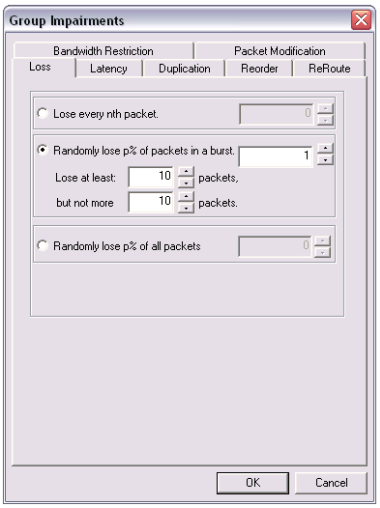



Figure 3.12: Adding bursty loss profile

As can be seen, the only available burst model is a random one, where the overall packet loss rate $p$ and the maximum and minimum number of lost packets can be set. By setting maximum = minimum = $n$, it is possible to implement the bursty packet loss model described in Section 3.5.2.

Because this approach is random, it was necessary with a process of trial and error to obtain loss profiles as those defined in the usage scenarios. The following example describes this approach for usage scenario #34.

**Desired loss characteristics:** Lose 30 packets in two bursts per 11 seconds
**Number of packets per 11 sec:** $15851/3 = 5284$
**Overall packet loss rate:** $60/5284 = 0.011 = 1.1\%$

By setting $p = 1.1$ and minimum/maximum lost packets to 30, the network emulator will, on average, lose 60 packets in two burst per 11 seconds[17]. Because consistent data was needed, it was required that two and only two bursts occurred in the last 11 seconds of the extended video clip, which later was to be extracted. This was accomplished by monitoring the loss statistics in real time, selecting only those clips that fitted the requirement. Figure 3.13 shows the loss characteristics approved for usage scenario #34. The reminding profiles can be found in Appendix B.

[17]Referring to the modified Gilbert Model of the prior section, $p$ corresponds to the random overall loss rate, $y$.

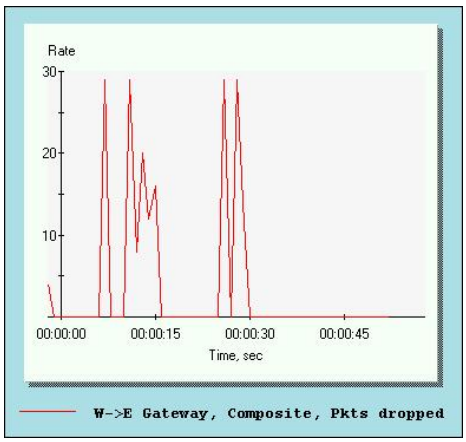



Figure 3.13: Loss profile usage scenario #34[a]

[a]Because of the limited horizontal resolution, it may look as if other amounts than 30 packets were lost in every burst. This is not the case. In the last 11 seconds of the video clip, we observe the two required bursts.

### 3.5.4 Post Processing

#### Remultiplexing the streams

As was discussed in the beginning of this section, VideoLAN encapsulates the raw MPEG-4 data in a MPEG Transport Stream in order to transport the data over a network. Because the received data was dumped raw to disk, a remultiplexing process to make playable MPEG-4 files was necessary.

Figure 3.14 shows the graphical user interface of the commercial Elecard XMuxer Pro[18], which was used to remultiplex the raw TS streams into pure H.264/MPEG-4 streams (files). We see that the original stream only consists of one elementary stream, video (PID 68).

#### Trimming of the video clips

When encoded, the original video files was extended to consist of three replicas of itself. The last operation in the degradation process was to extract the third and last copies from these extended versions. Figure 3.15 shows the original merging, 33 seconds long and consisting of 825 frames.

QuickTime Pro was used to extract the third copy, by manually locating the first frame of the third replica.

[18]`http://www.elecard.com/products/products-pc/consumer/xmuxer-pro/`

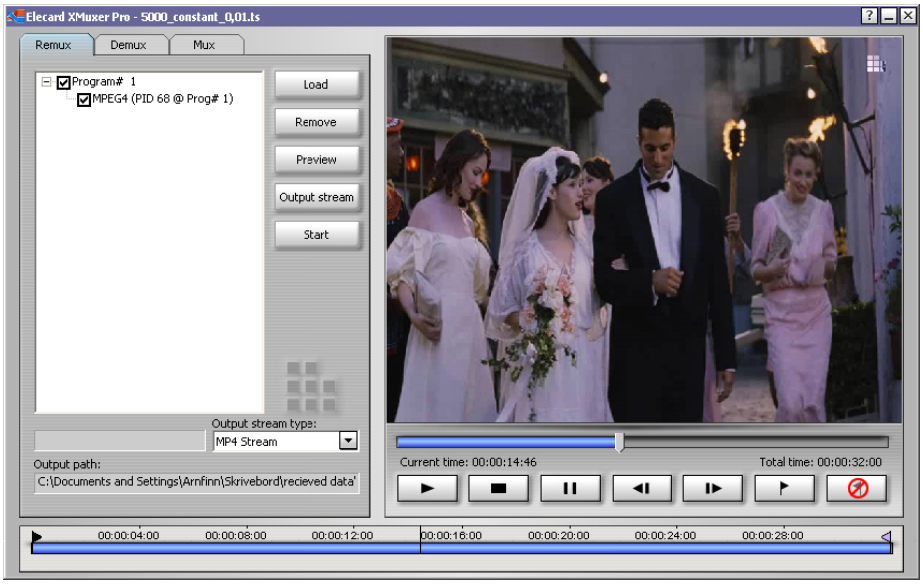



Figure 3.14: Demuxing the MPEG Transport Stream

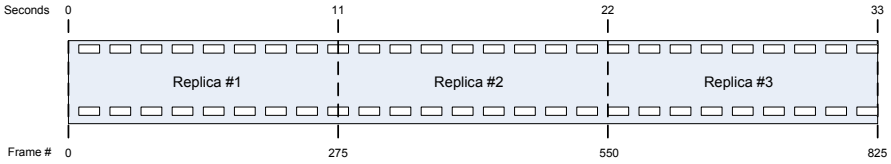



Figure 3.15: Original merging of the video clips

## 3.6 Implementing the SAMVIQ Test Interface - SSAT

To conduct the test, some sort of test interface was needed. Very few subjective assessment tools are available on the open marked. There exist, to the author's knowledge, only one tool that implements all the methodologies described Section in 2.3; the *MSU Perceptual Video Quality Tool* [53]. However, it lacks important support for widely deployed video codecs[19], and has an unintuitive approach to the SAMVIQ methodology. An interface implementing only the SAMVIQ methodology has been developed by France Telecom, but is not open source nor available to other than researchers within the EBU [35].

Because of the reasons mention above, it was decided to design and implement a brand new assessment tool. In the following subsection, the preliminary system requirements which formed the basic of the software development process, are outlined, as are the technological choices made. Please refer to Appendix D for detailed documentation and source code. The tool was named **SSAT - SAMVIQ Subjective Assessment Tool.**

[19]Currently supports only .avi, .avs and .yuv video formats.

### 3.6.1 General Description

When developing SSAT, the following vision served as the basic guideline for what the system should be able to do.

> SSAT shall implement the SAMVIQ methodology of perceived video quality assessment in an intuitive and self-explanatory way. The test-subjects should be able to compare and rate different video clips on a 0-100 scale, without assistance and in an interactive manner. SSAT should record the results in a consistent way, allowing for later statistical analysis.

It is worth noting that this vision doesn't call for any intuitive managing-functionality. That is, there is no interface to decide which video clips to include in the test, how many clips to include in each test etc. This information is passed to SSAT through config-files, and is described in Appendix D. The emphasis of SSAT has in the context of this report been to allow intuitive and user-friendly functionality for the *test-subjects*, not the *test-organizers*. This is further commented on in Section 5.3, Further Work.

### 3.6.2 Functional Requirements

To allow for the fulfilling of the vision stated in the above section, the functional requirements of SSAT were identified and listed. These requirements are cited in Table 3.8.

| FR | Description | Priority |
|---|---|---|
| 1 | Secure authentication of test-subject prior to test-start (login). *To ensure uniqueness of each test-subject. However, anonymity should be maintained.* | High |
| 2 | Test-subjects should be tested for color-blindness prior to test-start. *Using standard Ishihara test charts.* | Medium |
| 3 | The reference-clip should be rated 100 by definition. *It should thus be impossible to alter the reference-clip's rating.* | High |
| 4 | The test-subject should be able to view the test-clips in his/her preferred order and multiple times. | High |
| 5 | Grading of the clips should be possible during playing of the clip as well as afterward. *On a continuous 0-100 scale with guiding labels (bad, poor, fair, good, excellent).* | High |
| 5 | The test-subject should be constantly updated on which test-clips he/she has graded/not graded. | Medium |
| 6 | The test-results should be saved in a consistent manner. | Medium |
| 7 | It should not be possible to exit SSAT without saving the test-results. | Medium |

Table 3.8: Functional requirements of SSAT

### 3.6.3 Technological Challenges and Choices

SSAT was developed in Java™(J2SE)[20], and is thus platform independent. Multimedia support in Java is however limited. The following paragraphs describe the native Java multimedia framework, JMF, and the QuickTime for Java framework, and explain why QuickTime for Java was chosen over JMF.

#### Java Media Framework

SUN has issued a multimedia API called Java Media Framework (JMF) [49], intended to be the standard for multimedia support in Java. This project has more or less been abandoned by developers, last updated in March 2003, resulting in inferior support. JMF lacks support for widely employed media codecs, MPEG-4 included, depending on non-proprietary add-ons to widen it's scope; an example being OmniVidea's FOBS [21]. The most serious flaw of JMF is however, irrespective of media support, it's performance. In the early development-process of SSAT, test-runs showed that JMF was unable to deliver stable decoding and playback of video-files above certain bit-rates (1200 kbps for MPEG-4 encoded video with 4CIF resolution). These performance issues combined with limited documentation and lack of native video-support resulted in the abandonment of JMF as multimedia framework.

#### QuickTime for Java

QuickTime for Java (QTJava) [39], is a set of cross-platform APIs which allow Java developers to build multimedia into Java applications independently of the inferior Java Media Framework. JMF is pure Java, while QTJava allows developers to access native multimedia libraries, making it both fast and reliable. QTJava is a powerful framework allowing for both streaming, editing and playback of multimedia content. In addition, QTJava performed very well in early test-runs, easily playing back MPEG-4 encoded video encoded at 5000 kbps and with 4CIF resolution. Based on this, QTJava was chosen as media framework in SSAT.

QTJava is a powerful API, whose entirety is extensive. Loading and playing a movie, which is all that is needed in SSAT, is on the other hand quite straightforward. The process of loading and playing a movie is the adding of a `quicktime.app.display.QTCanvas` to a `javax.swing.JPanel`, which "punches a hole" through Java's normal graphical hierarchy, accessing the system's native multimedia mechanisms. The next steps are:

1. Create a `quicktime.std.movies.Movie` object.
2. Create a `quicktime.app.view.MoviePlayer` object that will play the movie just created.

[20] `http://java.sun.com/javase/`
[21] `http://fobs.sourceforge.net/`

3. Create a `quicktime.app.players.QTPlayer` object that displays the movie progress-bar.
4. Add the `QTPlayer` to the `QTCanvas`.
5. Start the movie.

These steps are illustrated in the example below:

```
(1) Movie movie = Movie.fromFile(OpenMovieFile.asRead(file));
(2) MoviePlayer moviePlayer = new MoviePlayer(movie);
(3) QTPlayer qtPlayer = new QTPlayer(new MovieController(movie));
(4) QTCanvas.setClient(qtPlayer, true);
(5) moviePlayer.setRate(1);
```

QTJava is closely linked to Apple's QuickTime Media Player[22]. Any Java application employing QTJava will be able to play any video-format supported in QuickTime. An extensive support of H.264/MPEG-4 is build into QuickTime, featuring one of the most versatile decoders on the marked today [4]. In the latest version (by July 2006), QTJava is embedded in the Quick Time Player and is thus installed on all machines that have QuickTime installed.

### 3.6.4 Mode of Operation

This section describes how the vision and functional requirements of Table 3.8 resulted in a fully operational test tool, and how the test subjects encountered SSAT when carrying out the test.

#### Logging in

When the test subject first starts SSAT, he or she must log in in accordance with functional requirement #1. This is to ensure uniqueness of each test subject so that the test values committed by each subject is saved independently. User names and passwords are handed out by the test organizer, and stored in a separate file, `users.sam`.

After a successful login, the test subjects are presented with a color-blindness test in accordance with the SAMVIQ methodology and functional requirement #2 (Figure 3.16). The test subjects will, independently of the result, be able to carry out the main test. The color-test result is saved together with the main test results in files with names on the form `user + test ID`, allowing for later analysis.

#### Loading the Tests

When the color-blindness test is passed, the next step is to load one of several test-scenarios. Depending on the choices made by the test-designer, the number

[22] `http://www.apple.com/quicktime/`

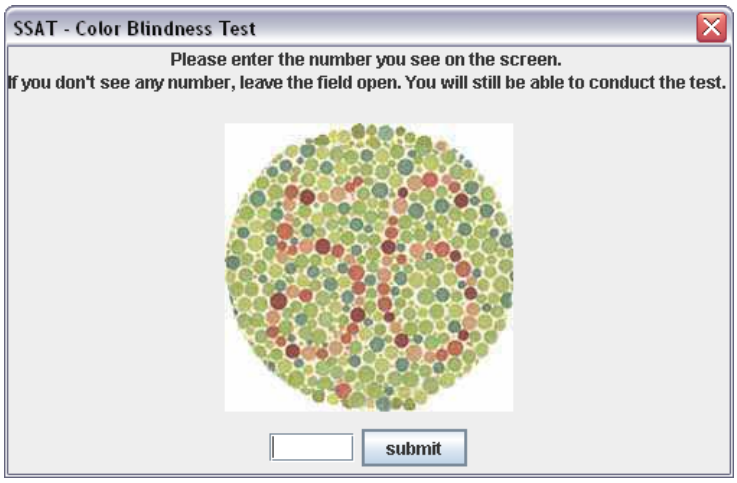



Figure 3.16: SSAT - Color Blindness Test

of test-scenarios may range from 1 to n. In the SAMVIQ methodology, a pretest is mandatory, and the test subject is therefore required to load the pretest-scenario first.

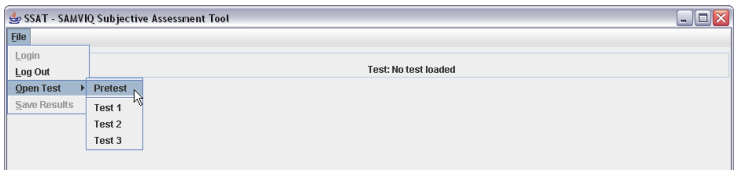



Figure 3.17: SSAT - Loading a Test Scenario

### Rating of the test clips

The essence of any subjective test is the rating of the individual test clips. After finishing the pretest, one of the main test scenarios should be loaded, and the test subject may choose which clip he would like to watch. In accordance with the SAMVIQ methodology, it is possible to view and rate each test clip multiple times. The rating itself is done by moving a continuous slider on a 0-100 scale, with labels indicating the different quality-ranges.

Figure 3.18 illustrates a typical test-scenario, with 9 different test clips in addition to the reference clip. The test subject has viewed and rated clip A, B and D, as can be seen by the green color of their respective buttons, while the other clips are still to be rated. It is not possible to finish a test before all test clips are rated, and the test subject can not close SSAT without saving unless he confirms to do so.

When one test-scenario is finished and saved, the user is free to load any other test-scenario, or to exit SSAT if all scenarios are finished.

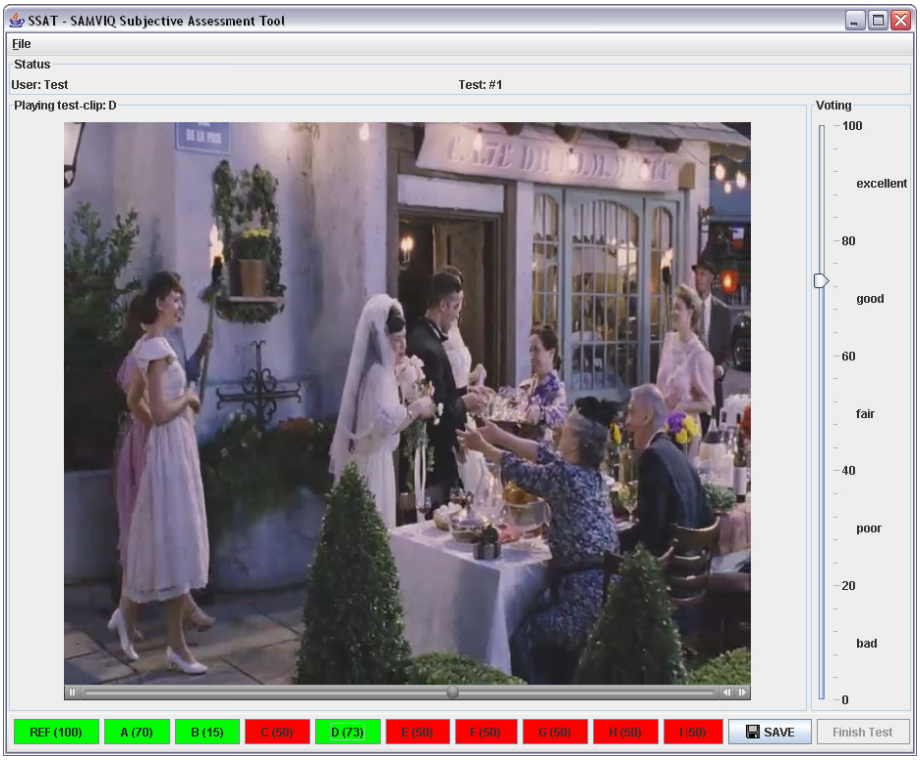



Figure 3.18: SSAT - Rating of individual test clips

## 3.7 Test Environment

When conducting subjective tests it is important to present the test subjects with similar viewing conditions. If this criteria is not satisfied, the validity of the obtained results would suffer to some extent. In addition; when care is taken to document and implement a consistent test environment, comparisons with other tests are better arranged for, and the test is possible to recreate and extend at other physical locations [30].

This section describes the physical environment under which the test was performed. All arrangements was done in accordance with the SAMVIQ methodology, which again bases it's environment recommendations on ITU-R Recommendation BT.500-11, Methods for the Subjective Quality Assessment of the Quality of Television Pictures [27].

### 3.7.1 Laboratory Environment

The test was performed at the Sahara Internet Laboratory[23] at the Norwegian University of Technology and Science (NTNU). The Sahara Internet Lab is usually an open computer lab, but during the tests, other activities were kept

[23] `www.item.ntnu.no/lab/sahara`

to an absolute minimum, ensuring a quiet and non-disturbing test environment.

Figure 3.19: The Sahara Internet Laboratory

The room was lit artificially from above, any direct sunlight eliminated by drawing white curtains, concealing the windows. This also had the added effect of removing any disturbance from window reflection. Three computers were used in the test. This allowed three test subjects to simultaneously conduct the test, even though this seldom was the case. The computers were aligned according to Figure 3.19, and the test subjects were seated in normal office chairs which could be independently set to suit the test subjects personal preferences.

### 3.7.2 Hardware Equipment

The computers used in the test were all specially set up. No additional software was installed, even access to the Internet was removed to eliminate any exploration by over-enthusiastic test subjects. This sort of preparation is especially important when implementing interactive methodologies where the test subject is left to him self for a longer period of time [35].

The hardware equipment used is summarized in Table 3.9.

| Equipment | Specifications |
|---|---|
| Computer | Dell Optiplex GX640 |
| Processor | Pentium D 3Ghz |
| RAM | 2GB |
| OS installed | Windows XP Professional 2002 (SP2) |
| Monitor | Dell 1950FP[a] |

Table 3.9: Hardware Equipment Used

[a]19-inch flat panel configured to a graphics mode of 1280x1024 True Color and a refresh rate of 60Hz. The luminance is 250 cd/m$^2$, which is well above the recommended minimum value of 200 cd/m$^2$.

### 3.7.3 Viewing Distance

While the traditional subjective test methodologies recommend a fixed viewing distance, SAMVIQ does not. This is natural, because the traditional methodologies was developed for TV display, where the viewer is assumed to have a permanent set up optimized for his viewing pleasure. When dealing with computer display, however, the viewer tends to interact more with the media content and the optimal viewing distance may vary from user to user. A *Punctum Proximum* is defined as the nearest viewing distance, subjectively determined by the viewer's eye, for optimum accommodation to a given display [36]. In the SAMVIQ methodology, the test subjects is assumed to optimize their punctum proximus continuously and individually. Specific instructions on viewing distance were therefore not given.

## 3.8 Conducting the Test

This section describes how the test was conducted and carried out, and how the test subjects were presented with the test material. The test took place between June 5 and June 24 2006, within the context described in section 3.7, Test Environment. The number of test subjects was 30.

### 3.8.1 Test Organization

In Section 3.2, 36 different usage scenarios were defined (Table 3.1). These scenarios were divided into 3 test cases, each which held 12 usage scenarios. Test case #1 included all test scenarios where the video clips were coded at 800 kbps, test case #2 the 2048 kbps coded ones and test case #3 included the clips coded at 5000 kbps.

Each test case was presented to the user through SSAT as individual test cases, which were loaded on request. Within the different test cases, the individual usage scenarios together with the reference and the hidden reference were randomly named according to table 3.10.

| Usage Scenario | Test Case | Name |
|---|---|---|
| reference | 1 / 2 / 3 | REF |
| hidden reference | 1 / 2 / 3 | K / E / B |
| 1 / 13 / 25 | 1 / 2 / 3 | I / M / H |
| 2 / 14 / 26 | 1 / 2 / 3 | A / G / D |
| 3 / 15 / 27 | 1 / 2 / 3 | L / H / I |
| 4 / 16 / 28 | 1 / 2 / 3 | H / D / A |
| 5 / 17 / 29 | 1 / 2 / 3 | M / I / E |
| 6 / 18 / 30 | 1 / 2 / 3 | J / K / F |
| 7 / 19 / 31 | 1 / 2 / 3 | D / J / G |
| 8 / 20 / 32 | 1 / 2 / 3 | E / L / L |
| 9 / 21 / 33 | 1 / 2 / 3 | F / F / M |
| 10 / 22 / 34 | 1 / 2 / 3 | B / A / C |
| 11 / 23 / 35 | 1 / 2 / 3 | G / C / K |
| 12 / 24 / 36 | 1 / 2 / 3 | C / B / J |

Table 3.10: The randomization and naming of the individual test clips

#### Session Length

According to the SAMVIQ methodology, no subjective test case should last for more than 30 minutes. This is to avoid boredom and to ensure concentrated subjects that grade the video clips according to their perception, not randomly due to exhaustion. Each clip in the test last for 11 seconds, adding up to a total playing time of 154 seconds per test case. Because SAMVIQ calls for multiple viewings and clip comparison, the expected time each test subject spends on each test case was calculated by multiplying the total playing time by a factor of 6. Add to this two five minutes breaks and time for training of the test subject, and it is possible to make an estimated time line for the entire test. This time-line is shown in Figure 3.20 and served as a guide when the test was conducted.

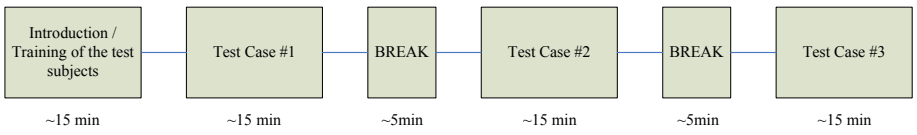



Figure 3.20: Estimated time line for the entire test

### 3.8.2 Registration and Training of the Test Subjects

#### Registration

When the test subjects showed up for the test, they were first given a short oral introduction explaining the context and goal of the test. Then they were asked to fill out a test form designed to collect personal information and preferences, later to be used in the analysis of the test results. The test form is depicted in Figure 3.21.

### Subject Training

The SAMVIQ methodology requires that the test subjects are experienced. This does not mean that the subjects should be professionally involved in video quality assessments on a daily basis, but that they are properly trained prior to the test.

It is important that the subjects acquire experience about the content, type of impairment, the quality levels likely to occur during the test session and the test interface. If this is not provided for, some of the time that could be spent assessing during the test cases would instead be spent making the subject familiar with the content and test equipment.

When the test subject had filled out the test forms, they were seated by one of the three dedicated test computers. The instructor handed out the written test instructions (Figure 3.22), and supervised the test subjects while they started the SSAT test interface. The subjects then loaded the pretest case, which was designed to make the subjects familiar both with SSAT, the reference clip and the type of quality deterioration packet loss may result in. When the pretest case was finished, the test subjects was left alone to complete test cases #1, #2 and #3.

username: asbj2
password: asbj2

NTNU
Innovation and Creativity

SUBJECTIVE TEST OF VIDEO QUALITY

**General Information**

This test is part of Arnfinn Flo's master thesis**, User-Perceived Quality-of-Service in Video-on Demand Services**. Its main goal is to assess how different network-parameters affect the perceived quality of streamed video delivered through the Internet.

Your participation is completely voluntary, and you may withdraw from the test at any time. The data you provide can and will not be connected to your name. The submitted scores will only be reported as a statistical average among the entire group, or sub-groups of the test participants.

---

**Pre-test questions**
*Please answer these questions when the instructor tells you to*

1. Sex:

   Male……☐ Female……☐

2. Age:

   Under 18…..☐ 30-49……...☐
   18 – 29…….☐ 50 or older……...☐

3. Highest completed education

   Primary/elementary school……☐
   High school……☐
   University/University College, <3 yrs…☐
   University, >3 yrs……...☐

   Field of specialization (if applicable)

   ……

4. What type of Internet connection do you have at home?

   None……☐
   Dial-Up……..……☐
   xDSL or Cable Modem……..☐
   LAN……☐
   Don't know……☐

5. Have you ever watched video streamed over the Internet?

   Yes……..☐
   No……...☐
   Don't know……☐

6. If YES, how would you rate the quality of the perceived video?

   Excellent……☐
   Good……...☐
   Fair……..☐
   Poor……....☐
   Bad……..☐

7. How many hours a week do you spend watching TV?

   none……☐
   up to 5 hours ……...☐
   5-15 hours……...☐
   15-30 hours……☐
   over 30 hours…….☐

---

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Telematics

Figure 3.21: The test form presented to the test subjects prior to the test

username: asbj2
password: asbj2

NTNU
Innovation and Creativity

SUBJECTIVE TEST OF VIDEO QUALITY

**Test Instructions**

Your main task in this test is to view several video clips through the custom developed SSAT – tool. You will rate individual video clips on a continuous 0-100 scale, comparing each clip with a reference clip rated 100 by definition. You may view each clip as many times and in any order you prefer. The test consists of three test cases, each consisting of thirteen test clips.

**SSAT – Instructions**

1. Start SSAT by clicking on the SSAT-icon located at the desktop of your computer.
2. Log in with the username and password supplied to you.
3. Proceed through the color-blindness test.
4. Load the pre-test scenario from the file-menu.
5. Start by viewing the reference-clip, then rate the other clips according to your perception of the quality. ***Remember that you can view each clip several times, and in any order you prefer***.
6. Save your results by pressing the SAVE-button.
7. Press the Finish Test button.
8. Feel free to take a five minute break.
9. Load Test 1 from the file menu
10. Repeat steps 5-8
11. Load Test 2 from the file menu
12. Repeat steps 5-8
13. Load Test 3 from the file menu
14. Repeat steps 5-8
15. Close SSAT and report to the instructor.

**Please consult the instructor if you encounter any problems. Do not close the program before all test-scenarios are saved and finished.**

**Thank you for participating.**

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Telematics

Figure 3.22: The test instructions presented to the test subjects

# Chapter 4

# Test Results

## Contents



This chapter presents and analyzes the results obtained in the subjective test. To ensure comparability and reproducibility, EBU has proposed a SAMVIQ evaluation protocol which includes the following information [35]:

1. Test configuration
2. Test sequences
3. Type of picture source and display monitor
4. Reference systems used
5. Number and type of assessors
6. The grand mean score for the experiment
7. Original and adjusted mean scores with 95% confidence intervals

Items one through four are treated in Chapter 3; this chapter deals with items five through seven and is organized as follows: First, a presentation of the statistical presumptions needed to analyse the test results in a consistent manner is given. Then, the subjective test results are uncritically presented followed by a discussion on the validity of these results.

## 4.1 Theoretical Framework

The statistical tools needed to analyse the results are not very complicated. Below, an explanation of these tools are presented, together with the presentation of the basic metrics used to describe the outcome of the test. A more in-depth discussion on the prerequisites for the statistics used is given in the preliminary project work [20].

### 4.1.1 Mean Scores and Confidence Intervals

#### Individual Mean Scores

The subjective test performed produced $N = 30$ different scores for each usage scenario, each score ranging from 0-100. The individual scores vary within each usage scenario due to the differences in judged perceived quality, producing a distribution with mean $\bar{\mu}$. This mean score of each usage scenario is the single most important parameter derived, and is formally given by equation 4.1 [27]:

$$\bar{\mu}_j = \frac{1}{N} \sum_{i=1}^{N} \mu_{ij} \tag{4.1}$$

where:

$\mu_{ij}$ : score of test-subject $i$ for usage scenario $j$

$N$ : number of test-subjects

In the remainder of this report, the individual mean scores of each usage scenario will be referred to IMSj, where j is the usage scenario of table 3.1, or just IMS when it is clear which usage scenario referred to.

#### Grand Mean Score

The grand mean score (GMS) is the mean produced when all IMSs are added and averaged. The grand mean score is not a very important value when it comes to practical analysis of the results, because it does not describe the perceived quality of any individual usage scenario. It is however required by the SAMVIQ methodology because of two reasons:

1. The grand mean score describes quantitatively the test-subjects overall perception of the quality, providing the researcher with useful feedback on the test design. A very low GMS (say 20) may indicate that the usage scenarios were too biased toward the low quality end of the scale, and vice versa.

2. The grand mean score serves as a signature for the test itself, making it a useful parameter when comparing identical tests conducted at different locations and by different personnel.

The grand mean score is defined according to equation 4.2:

$$GMS = \bar{\mu} = \frac{1}{J}\sum_{j=1}^{J}\bar{\mu}_j = \frac{1}{NJ}\sum_{j=1}^{J}\sum_{i=1}^{N}\mu_{ij} \tag{4.2}$$

where:

$\bar{\mu}_{ij}$ : score of test-subject $i$ for usage scenario $j$

$N$ : number of test-subjects

$J$ : number of usage scenarios

Correspondingly, an *individual* GMS can be calculated for each test subject, a parameter especially useful when screening the test subjects. Such an individual GMS, denoted gms, is for test subject $i$ defined according to equation 4.3:

$$gms = \bar{\mu}_i = \frac{1}{J}\sum_{j=1}^{J}\mu_{ij} \tag{4.3}$$

### Confidence Intervals

The SAMVIQ methodology requires all means to be presented with an associated 95% confidence interval. Any confidence interval is derived from the standard deviation and size of each sample, and is for each usage scenario $j$ given by [54]:

$$[\bar{\mu}_j - t_{\alpha/2(v)}\delta_j, \bar{\mu}_j + t_{\alpha/2(v)}\delta_j] \tag{4.4}$$

where:

$$\delta_j = \frac{S_j}{\sqrt{N}} \tag{4.5}$$

and:

$t_{\alpha/2(v)}$ is found by lookup in the Student's t distribution with $\alpha = 0.05$ and $v = N - 1$.

The standard deviation for each usage scenario, $S_j$, is given by equation 4.6:

$$S_j = \sqrt{\sum_{i=1}^{N} \frac{(\bar{\mu_j} - \mu_{ij})^2}{(N-1)}} \tag{4.6}$$

The textual interpretation of the confidence interval is [27]:

> With a probability of 95%, the absolute value of the difference between the experimental mean score and the "'true"' mean score (for a very high number of observers) is smaller than the 95% confidence interaval, on condition that the distribution of the individual scores is independent and identicly distributed.

### 4.1.2 Rejection Criteria

The SAMVIQ methodology requires that all test-subjects who have taken part in the subjective test, must be screened in order to establish the consistency of their scores. Inconsistent subjects, that is subjects who produce unstable or even contradictory scores, are to be removed from the final statistical analysis of the results. Thus, the final number of assessors that contribute to the mean scores may be below the original number, resulting in new calculations of the test means.

The rejection criteria used by SAMVIQ is the Pearson's product-moment coefficient or the Pearson decision criteria, "Pearson's r". Depending on the value of r, a test subject may be discarded or not. Pearson's r reflects the degree of linear relationship between two random variables X and Y. It ranges from +1 to -1. A correlation of +/-1 means that there is a perfect linear relationship between the variables, while a r-value of 0 indicates complete independency.

In SAMVIQ, the two random variables X and Y represent the individual score of the test-subject and the corresponding mean score from all the subjects respectively. With $J$ usage scenarios the sample correlation coefficient, or Pearson's r is defined as [54]:

$$r = \frac{\sum_{i=1}^{J} (x_i - \bar{x})(y_i - \bar{y})}{(J-1)S_x S_y} \tag{4.7}$$

where $x_i$ and $y_i$ are the individual-test subject score and the IMS of usage scenario $i$ respectively, and $\bar{x}$ and $\bar{y}$ are the and individual grand mean scores (gms)

and grand mean score (GMS). $S_x$ and $S_y$ are the sample standard deviations, calculated according to equation 4.6

This may be rewritten as:

$$r = \frac{J\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{J\sum x_i^2 - (\sum x_i)^2}\sqrt{J\sum y_i^2 - (\sum y_i)^2}} \tag{4.8}$$

which was the form used when calculating the Pearson's r values in the subjective test.

The SAMVIQ methodology states that a test subject should be discarder if $r$ is less than the correlation threshold, which is usually set to 0,85.

## 4.2 Usage Scenario Test Results

On the following pages are the complete test results depicted. The IMSs of the different usage scenarios, together with their 95% confidence intervals, are plotted against the loss profile of each scenario. 0% packet loss refers to the IMSs of the hidden reference scenarios. Please refer to Table C.1 of Appendix C for exact numerical values.

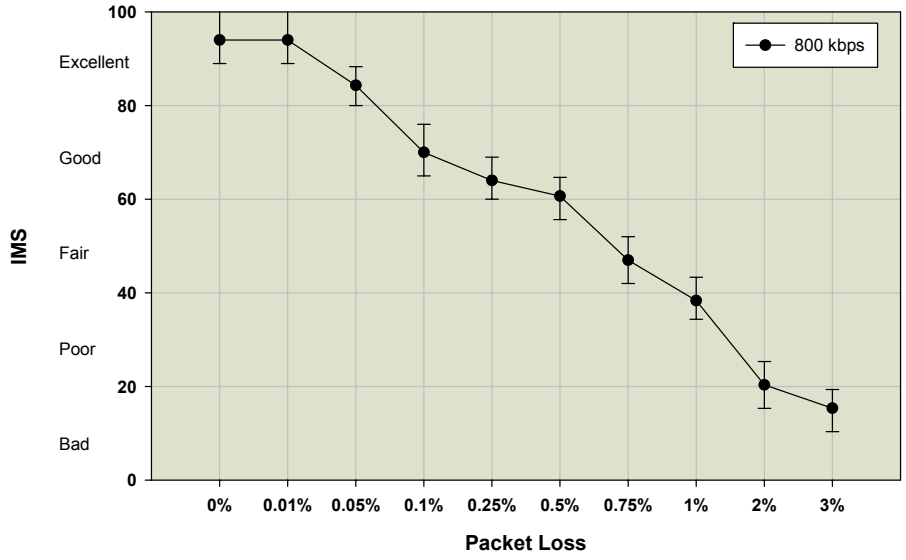



Figure 4.1: Unbursty Loss 800 kbps, Usage Scenarios 1-9

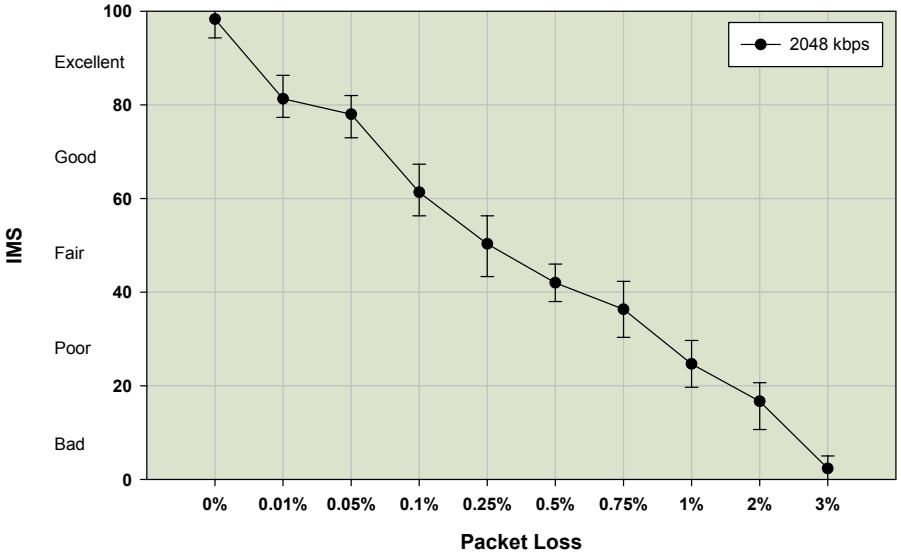



Figure 4.2: Unbursty Loss 2048 kbps, Usage Scenarios 13-21

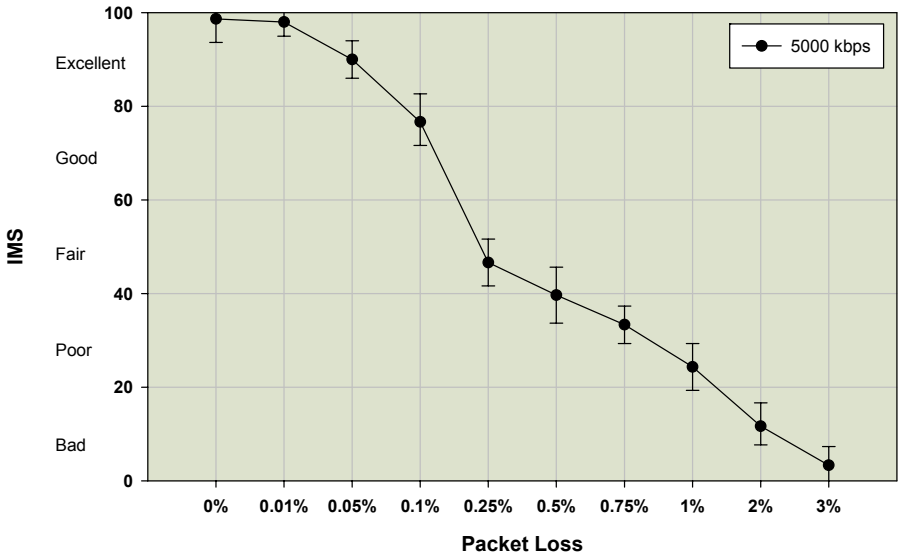



Figure 4.3: Unbursty Loss 5000 kbps, Usage Scenarios 25-33

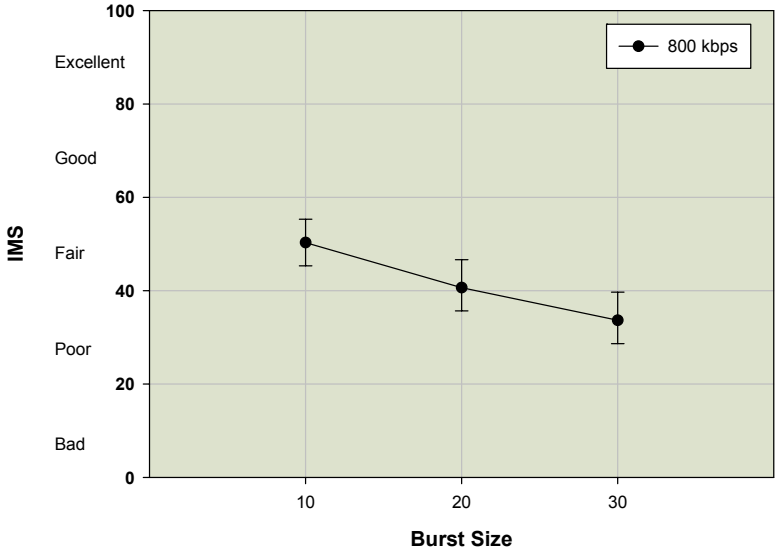



Figure 4.4: Bursty Loss 800 kbps, 2 bursts pr 11 second clip, Usage Scenarios 10-12

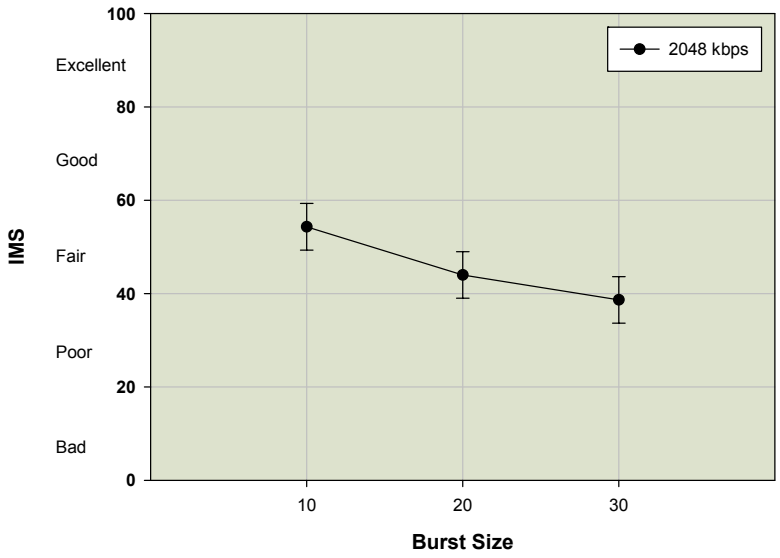



Figure 4.5: Bursty Loss 2048 kbps, 2 bursts pr 11 second clip, Usage Scenarios 22-24

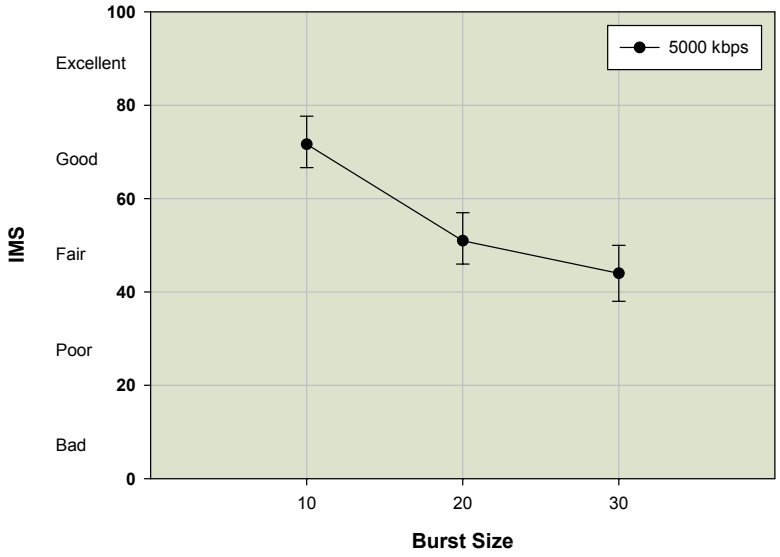



Figure 4.6: Bursty Loss 5000 kbps, 2 bursts pr 11 second clip, Usage Scenarios 34-36

## 4.3 Discussion

The main goal of the subjective test carried out was defined in Section 3.1. The overall objective was to survey the perceived degradation-effect of packet loss on VoD-content encoded with H.264/MPEG-4 and delivered to the end-user through Internet. In this section, the results obtained is discussed in the form of a continuous text. For a simplified and concluding summary, please refer to Chapter 5.

### 4.3.1 General Observations

In the figures of Section 4.2, the IMSs of each usage scenario together with their 95% confidence intervals were presented. Figures 4.7 and 4.8 summarize these results by plotting the IMSs of the different bitrates in the same diagram. The confidence intervals are ommitted for improved readability.

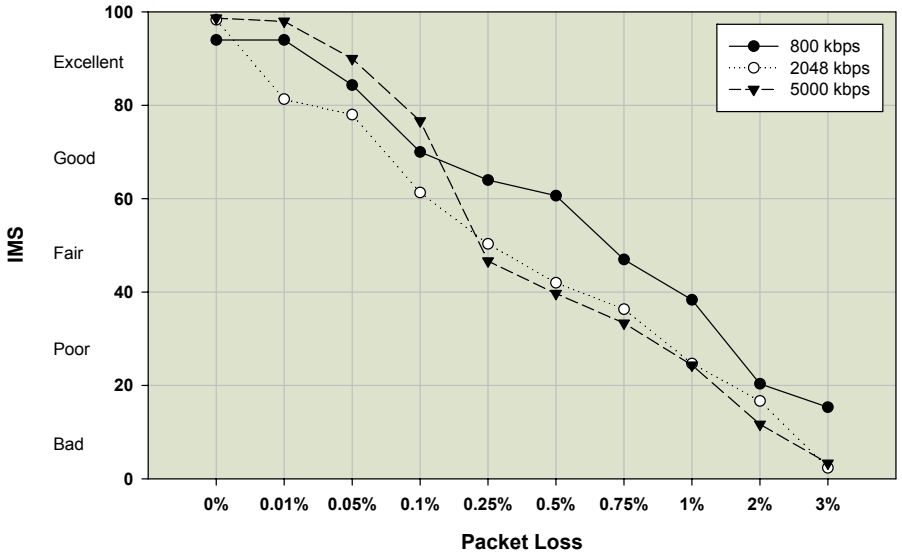



Figure 4.7: Summarized Results, Unbursty Loss

It seams clear from the results that an increase in the packet loss rate, bursty or not, significantly reduces the test subjects perception of the video-quality. This is of no surprise. On the contrary; if loss of information would not degrade the quality of the received content to any extent, the test would have to be discarded. Multiple surveys, e.g [37, 6] show that an increasing packet loss rate decreases the quality. These studies consider MPEG encoded video, even though MPEG-4 is not treated especially.

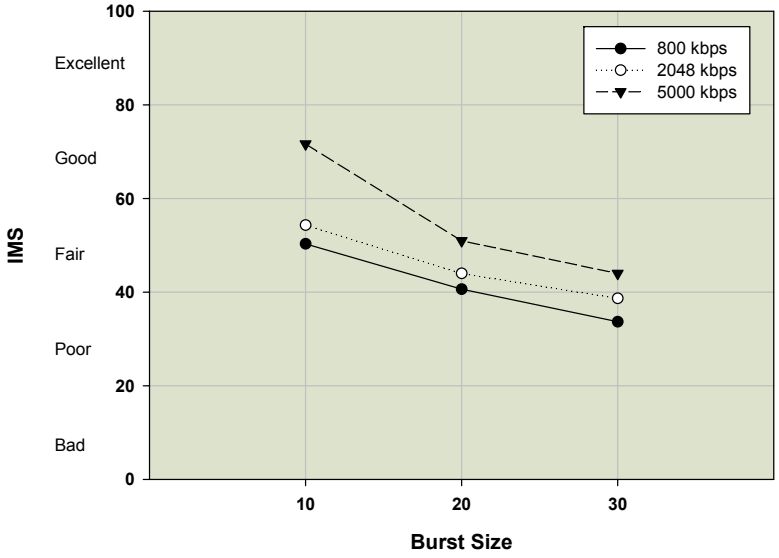



Figure 4.8: Summarized Results, Bursty Loss

### 4.3.2 The Hidden Reference Clip

It is interesting to note the small, but still noticeable, decrease in perception when no packet loss is added whatsoever. The 0% case corresponds to the *hidden reference* in SAMVIQ methodology (See Section 2.3.2). In an ideal world, the hidden reference which is identical to the standard of reference, should be rated accoringly, that is with a perfect IMS of 100. For the 5000 and 2048 kpbs test cases the difference is very small (2 and 3 points respectively), but for the 800 kbps case a 6 points degration is observed. This observation suggests that the test-subjects find the lowered quality of the 800 kbps encoded clip to be noticable, even though the reference clip looks the same. Because the 800 kbps test case was the third and last case shown to the test-subjects, they may still be influenced by the better quality of the 2048 and 5000 kbps cases. This further suggests that the 800 kpbs case should suffer from lower scores through all packet loss rates, *if* packet loss influences all bitrates the same. This is however, not the case, as is shown later.

### 4.3.3 Unbursty vs. Bursty Loss

To compare the unbursty loss scenarios with the bursty ones, a mapping from *number of consecutive lost packets* in the bursty model to the *random overall loss rate* of the unbursty model is needed. Armed with the statistics and models of Section 3.5, and bearing in mind that two bursts occur per 11 second video clip, this mapping takes on the form of Table 4.1.

The most general of observations is that a bursty loss behavior is preferred over an unbursty one, if the number of consecutive lost packets over is kept the same

| Bitrate | Burst Size | | |
|---|---|---|---|
| | 10 | 20 | 30 |
| 800 kbps | 2,2% | 4,4% | 6,6% |
| 2048 kbps | 0,9% | 1,8% | 2,7% |
| 5000 kbps | 0,4% | 0,8% | 1,1% |

Table 4.1: Random Overall Loss Rates, Bursty Loss Model

(the random overall loss percent). This can for example be seen by comparing the IMSs of the 5000 kbps case. A burst size of 10, corresponding to a random loss percent of 0,4% rates 72, or in the middle of the *good*-interval. The closest unbursty packet loss percent (0,5%) rates 40, or on the border between *poor* and *fair*. For the unbursty loss scenario to score as good as the bursty one, noe more than 0,1% packet loss could be tolerated[1]. Such relations hold true for all of the bursty loss scenarios. Figure 4.9 shows this by plotting the relationship between the perceived quality (IMS) of the bursty and non bursty usage scenarios and corresponding packet loss rates. For the unbursty rates the loss percentage closest to the bursty IMS was used, which for all but one case was within the confidence interval borders of that usage scenario.

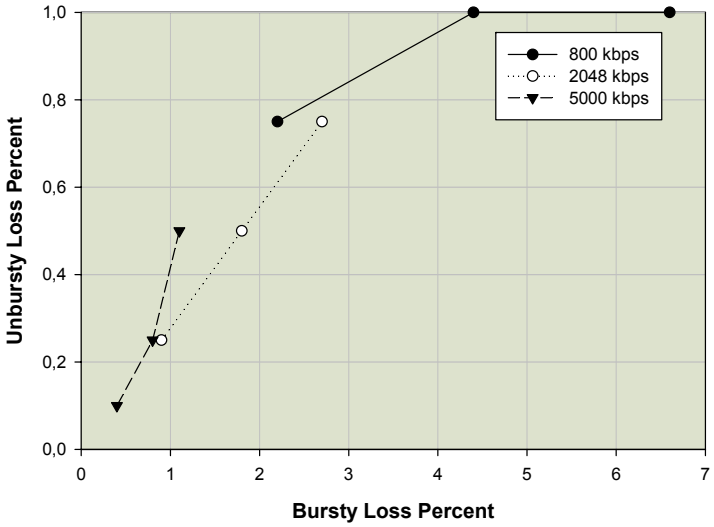



Figure 4.9: Corresponding loss percentages for same IMS, bursty and unbursty loss profiles

We observe that the relationship between the bursty and unbursty loss percentages resembles a linear one, at least for the 2048 and 5000 kbps cases[2]. With

[1] 0,1% unbursty loss scores 77 which is 5 points above the IMS of the bursty loss scenario. The actual loss percent may thus be somewhat higher.

[2] The 800 kbps case is also believed to hold semi-linear qualities, but due to the large gap in tested loss percentage between 1% and 2% (as apposed to the finer grained intervals in the rest of the scale), this does not show in the plot. While the 1% scenario rates 38(43-34) the 2% case rates 20(25-15) suggesting that the "'true"' value of an IMS of 34 may lay higher than 1%.

only three bursty loss scenarios per bitrate, however, not enough empirical data is availible to state so within accepteble levels of confidence.

### 4.3.4 The Importance of Bitrate

An increasing bitrate is equivalent to an increasing number of packets streamed per time unit. Table 3.5 summarized the streaming-statistics of the different bitrates, from which the metrics of Table 4.2 are obtained:

| Bitrate | Packets per clip | Packets per second |
|---|---|---|
| 800 kbps | 914 | 83 |
| 2048 kbps | 2213 | 201 |
| 5000 kbps | 5284 | 480 |

Table 4.2: Packets Streamed per Time Unit

#### Bitrate - Unbursty Loss Scenarios

Because of the higher number of packets streamed in the 5000 kbps case than in the 2048 and 800 kbps cases, more packets will be lost per time unit. A higher number of lost packets could be expected to result in an increased relative quality-degradation, as explained among others in [40, 7].

This assumption, however, only holds true to some extent. For the unbursty packet loss scenarios, we see an increased perception-degradation for the higher bitrates at packet loss rates above 0.1%. At 0,75%, for example, the 800 kbps case rates 47 while the 2048 and 5000 kbps cases rate 36 and 33 respectively. This does not imply that the *overall* quality of the 800 kbps clip is better than the 5000 kbps one, only that *relative* to the reference clip (which is of lower quality), the quality degradation is less. That is, the 800 kbps case is less vulnerable to unbursty packet loss than the 2048 and 5000 kbps ones. The higher number of packets lost per time unit is in other words not countered by the increased bitrate. Increasing the encoding-bitrate adds to the redundancy and error-resiliance of H.264/MPEG-4 in addition to increasing the overall quality [26], but this effect is shown to be quantitative inferior to the effect of packet loss, at least in the scenarios considered in this test.

As can be seen from Figure 4.7, the relative quality-degradation takes on different characteristics at packet loss rates less than 0,25%. Most noticable is the 5000 kbps case, which goes from being the best ranked case (at loss percentages from 0 through 0.1%) to the worst. The H.264/MPEG-4 codec is designed to cope with data loss and other errors [4], and these mechanisms are belived to counter the effect of packet loss to some extent at low packet loss rates. At 0.25% and above, however, the packet loss rate becomes too high, and the user experiences the “full” effect of the lost data. Another reason for the shiht in mutual ranking is that the superior quality of the 5000 kbps case counter the effect of packet loss to some extent, but fails to do so when reaching a threshold value somewhere between 0,1% and 0,25%. This does not, however, explain

why the 2048 kbps case is ranked the worst in the low loss percentage range. This unexpected and rather strange result may suggest that the subjective test carried out is inferiour and unaccurate in the low percentage range, or that the randomly lost packets in the 2048 case accidentally was of greater importance than those in the other cases. This could for example happen if a high number of B-frames were lost. Either way, the results from 0% to 0,1% are to be treated with great suspicion.

What *does* seems clear form the collected data, is that a low bitrate H.264/MPEG-4 stream better sustains it's original quality than a high bitrate stream. But as will be shown; on the condition that the packet loss is characterized in an unbursty way.

### Bitrate - Bursty Loss Scenarios

When categorizing the bursty scenarios with respect to burst size (Figure 4.8), the results are quite clear and understandable. A burst size of 10 consecutive lost packets results in an IMS of 72 for the 5000 kbps case, 54 for the 2048 kbps case and 50 for the 2048 case. This is of no surprise, since such consecutive lost packets result in a burst length of 0,12 seconds, 0,05 second and 0,02 seconds respectively[3]. A shorter burst length makes the perceived effect of the packet loss less noticeable, and we can thus conclude that a higher bitrate diminishes the effect of bursty packet loss, when the bursts are modeled as described in Section 3.5.3.

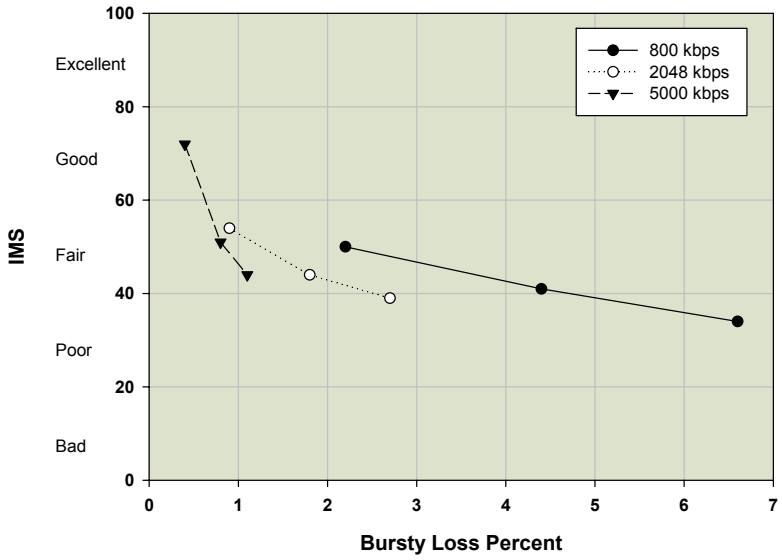



Figure 4.10: Overall Loss Percentage, Bursty Scenarios

If we substitute the burst size with the random overall loss percentages of Table 4.1, on the other hand, the results become quite different. Figure 4.10 shows

[3] $BurstLength(s) = \frac{packets/burst}{packets/second}$ , data taken from Table 4.2

the burst sizes converted to overall loss percentage and their corresponding IMSs. We observe for example that a loss percentage of 4,4% in the 800 kbps case rates (almost) as good as loss percentages of 1,8% and 1,1% in the 2048 and 5000 kbps cases. Such results suggest that lower bitrates is more resilient to bursty packet loss, *if* measured by overall loss percentage.

## 4.4 Validity of the Results

The results presented and discussed in the above sections are affected by several constraints worth noting. In this section, the validity of the obtained results are discussed in the context of these constraints.

### 4.4.1 Grand Mean Scores

The Grand Mean Score (GMS), as defined in Section 4.1, gives in addition to serving as a test-signature, a clue on how successful the test design has been. A very low GMS suggests that too many test scenarios with a high degree of degradation was employed and vice versa. The SAMVIQ methodology stresses the importance of a medium sized GMS (that is in the 40-60 area), because well diversified test scenarios, in where the whole IMS-scale is used, tend to produce more accurate results. Table 4.3 shows both the GMS of the whole experiment, and the individual GMSs of each bitrate.

| Test Condition | GMS |
|---|---|
| 800 kbps | 59 |
| 2048 kbps | 48 |
| 5000 kbps | 53 |
| Total | 53 |

Table 4.3: Grand Mean Scores of the Experiment

We observe that all GMSs are well within the recommended interval (40-60). The test design could therefore be categorized as diversified and balanced, not adding any excess uncertainty to the obtained results.

### 4.4.2 Test Sample - Classification of Subjects

Of vital importance in any empirical test is the selection of test sample. A test sample could be defined as:

> A segment of the population that has been selected to represent the population as a whole [34].

When selecting such a sample, three main factors come into play: The *sample unit*, the *sample size* and the *sampling procedure* used.

The choice of sampling unit is equivalent to picking subjects who are likely to possess the information needed [34]. Since the process of evaluation video clips is a pure subjective one; there is no wrong or right, merely a state of opinion and perception, samples could basically by drawn from the entire grown up population. The SAMVIQ methodology, however, advises against choosing subjects who are professionally involved in picture quality assessments, as they often have preconceived judgements of video artifacts, resulting in somewhat biased scoring [35]. Such subjects were thus avoided, placing some restriction on the sample unit.

The SAMVIQ methodology requires, in order to obtain statistically valid results, a sample size of at least 15 after removing inconsistent subjects [35]. In the subjective assessment performed in this Master's Thesis, 30 test subjects carried out the test, later reduced to 23 after post-screening (See Section 4.4.3). The sample size is therefore considered satisfactory, adding to the validity of the test itself.

The choice of sampling procedure is, on the other hand, one of compromise. The best way to draw a limited sample from a finite population is by using some sort of *probability sample* [34]. Due to stringent time limitations and the accessibility of willing subjects, the sample drawn in this test hold the characteristics of a*convenience sample*. A convenience sample is defined as:

> (...) a sample where the researcher selects the easiest population members from which to obtain information [34].

The demographic distribution of the test sample is shown in Table 4.4.

Table 4.4: Classification of Subjects

(a) Gender

| Gender | # |
|---|---|
| Male | 22 |
| Female | 8 |

(b) age

| Age | # |
|---|---|
| under 18 | 2 |
| 18-29 | 21 |
| 30-49 | 5 |
| 50 or older | 2 |

(c) Eduacation

| Highest completed education | # |
|---|---|
| Primary/Elementary school | 1 |
| High School | 1 |
| University/University College <3 yrs | 11 |
| University >3 yrs | 17 |

(d) Experience

| Watched video over Internet before? | # |
|---|---|
| Yes | 30 |
| No | 0 |

(e) Internet Connection at Home

| Connection | # |
|---|---|
| None | 9 |
| Dial-Up | 0 |
| xDSL or Cable Modem | 16 |
| LAN | 5 |
| Don't know | 0 |

In addition, all but four test subjects were picked from students at the Norwegian University of Science and Technology, 15 of which specialize in communication technology. Hence, both age and educational background are relatively similar, as is the level of experience. It is difficult to make any judgements on how this conformist demographic distribution affects the validity of the test results. However, since all assessors were well experienced in computer-technology, this may have added to the validity because of a diminished psychological barrier when dealing with the test interface.

### 4.4.3 Inconsistent Subjects

The SAMVIQ methodology requires that test subjects who vote inconsistently and uncorrelated with the mean scores, are removed from the final test results. The Pearson's r product-moment coefficient was defined and explained in Section 4.1.2, and served as refection criteria:

$$\text{If } |r_i| < 0,85, \text{ remove subject } i \text{ from the calculation of the results.}$$

This criteria led to the removal of 7 test subjects. Table 4.4 shows the distribution of r-values among the test subjects, which in the $[0,85 - 1 >$ interval averaged at 0,91.

| r-value interval | # |
|---|---|
| $[0 - 0,2 >$ | 1 |
| $[0,2 - 0,8 >$ | 2 |
| $[0,8 - 0,85 >$ | 4 |
| $[0,85 - 1 >$ | 23 |

Table 4.4: Distribution of r-values

It is interesting to note the few low r-values. To score as low as 0,09, which was the score of the $[0 - 0,2 >$ occurrence, complete ignorance, or even intented sabotage, has to be assumed. The removal of such subjects add to the validity of the test, and a mean vale of 0,91 in the $[0,85 - 1 >$ interval suggests a more than satisfactory correlation among the unremoved subjects.

# Chapter 5

# Conclusion

**Contents**



This final chapter concludes the master's thesis. It summarizes the most important findings and relates them to other works and assessments. Last, suggestions on further work is presented.

## 5.1 Main Results

The main objective of the subjective test performed in this master's thesis was defined in Section 3.1 and discussed in detail in Section 4.3, quote:

> The main objective of the test is to survey the perceived degradation-effect of packet loss on a typical VoD-service delivered to the end-user through Internet. The media-content is to be encoded with the H.264/MPEG-4 video coding standard.

In conjunction with this objective, a set of questions were defined which answers have been sought throughout the report:

- Does packet loss influence the perceived quality of MPEG-4/H.264, and if so, to what extend?

- If the packet loss rate is kept constant, how does an increase in the available bandwidth affect the perceived quality?
- Is there a difference in the perceived quality when the packets are dropped in a burst compared to an unbursty loss-rate?

It can, without doubt, be stated that packet loss influences the perceived video quality. The results obtained show a stringent relationship between increased packet loss rates and lowered perception of quality. Further, if the packet loss rate is kept below 0,1%, the results suggest that the video quality can be described as acceptable. Any increase in the packet loss rate above this threshold level, and the perceived quality droppes to a level that is unsatisfactory to the end-user.

An increase in the end-user's available bandwidth allows for increased sending-bitrate at the server side. This improves the quality of the video, but at the same time increases it's vulnerability to packet loss. At packet loss rates above 0,1% the relative quality degradation of the different bitrates tested suggests such a coherence. At loss rates below 0,1%, the inter-bitrate relations are unconsistent, even though it is believed that the improved quality of the high bitrate cases counters the effect of a small amounts of packet loss.

The results described above relates to packet loss characterized as unbursty, or random. When applying bursty packet loss, the results differ. When the burst size is measured in consecutive lost packets, the perceived quality deteriorates the most at low bit rates and the least at high. This is natural, because more packets are sent per time unit in the high bitrate cases. When converted to overall loss percentage, on the other hand, the results are similar to that of unbursty packet loss: High bitrates are more vulnerable to packet loss than low.

Further, the results show that a bursty packet loss behavior is preferred to an unbursty one. It may also be the case that a linear relationship exits between bursty and unbursty loss percentages that result in the same quality degradation, but not enough data is available to state so with certainty.

## 5.2 Related Works

As noted in Section 1.1, very few subjective assessments as the one described in this report have been published. The majority of available subjective video tests concentrate on the comparison of different video codecs at different bit rates, not their resilience to packet loss. Examples of such works include [36, 56, 52]. For more information, the reader is referred to the preliminary project work, [20].

Of the published work directly relating to the results of this master's thesis, some are worth noting: [37] shows, in coherence with the results obtained in this report, that MPEG video is less affected by bursty packet loss compared to unbursty loss, if the overall loss rate is kept the same. In [1] it is concluded that increasing the packet burst size produces improved opinions of quality if the packet loss rate is kept constant. [35] introduces the SAMVIQ methodology,

and shows that the relative quality degradation due to packet is smaller for low bit rates than for high.

Also worth noting is an ongoing research project at the Centre for Quantifiable Quality of Service in Communication Systems (Q2S), at the Norwegian University of Science and Technology. The work relates to the visual quality of H.264/MPEG-4 when transported in error-prone environments like Internet. Please refer to `http://www.q2s.ntnu.no/` for updates and publications when available.

## 5.3 Further Work

This master's thesis has been the ultimate academic goal of the author since finishing the preliminary project work, [20], in March 2006. Reaching this goal does not mean that there is no more work to consider, nor that every aspects of user-perceived quality in VoD-services are covered. In this section, some suggestions on further research topics are presented, along with thoughts on how to remove some of the test's constrains.

### 5.3.1 Different Usage Scenarios

This work put 36 different usage scenarios to the test. In a retrospective view, more bursty scenarios would have been an interesting approach, both since packet loss in Internet best can be categorized as bursty, and because several different burst-models could be employed. The mostly unbursty scenarios modeled here may by some be considered unsuccessful in modeling Internet packet loss behavior. Irrespectively of this, the test has described H.264/MPEG-4's resilience to packet loss in an universal context. However, to better survey the codec's resilience in a realistic Internet-scenario, more work should be put into different bursty scenarios.

### 5.3.2 QoS-mechanisms in Internet

Even if Internet is a best-effort network, several efforts have been made to employ different QoS-mechanisms in the core network. These mechanisms include over-provisioning of the network, traffic engineering and IP QoS mechanisms such as IntServ and DiffServ [50]. Because commercial VoD providers sometimes are able to prioritize individual customers or even individual streams, an interesting approach would be to employ one or more of these QoS-mechanisms in the test setup. This would further enhance the commercial value of the test results, due to the not neglectable cost of Internet QoS.

### 5.3.3 Transport Mode

When streaming H.264/MPEG-4 in the test setup described in Chapter 3, a choice was made to employ the MPEG-4/MPEG-TS/RTP/UDP/IP protocol stack. An other approach would be to use native RTP, that is to packetize the raw H.264/MPEG-4 data directly into RTP packets, avoiding the MPEG Transport Stream layer. The resulting protocol stack can then be described as MPEG-4/RTP/UDP/IP. It has been shown that this approach may add to the error resilience of the codec itself [38]. An interesting task would therefore be to perform such tests as performed in this master's thesis with the new protocol stack employed, and compare the results.

### 5.3.4 Developing SSAT

The development of SSAT, SAMVIQ Subjective Assessment Tool, is not stated as one of the Master Thesis' objectives. It is however an achievement part of the thesis, and a noticeable time was spent developing it. When carrying out the subjective tests, SSAT proved itself as an user friendly and intuitive tool, adding to the convenience of the test implementation. It was stable at runtime and is not believed to have had any negative influence on the assessors experience. Even though the source code is not optimized, nor very well commented at present time, further development could result in a tool that incorporates the SAMVIQ methodology in a fully satisfactory way. It's most serious drawback as today, is the lack of intuitive management functionality and the ease of developing extension to further enhance it's features.

### 5.3.5 Test Report

As described in Section 3.1.4, the SAMVIQ methodology divides the process of subjective quality evaluation into eight discrete steps. The last step describes the publishing of a test report. Even though this report can be viewed as such a report, a more concentrated presentation would be of value to those only interested in the core results of the thesis. The publishing of an independent test report is not planned for, but could be considered at a later stage if the results prove themselves to be of substantial interest to academia or commercial actors.

# Bibliography


[1] *Report on Video Streaming Tests*, 1999. JUPITER II, Joint Usability, Performability and Interoperability Trials in Europe.

[2] Tom Adams. *Video-on-Demand: The future of media networks, Strategic lessons from the US experience.* Adams Media Research, 2005.

[3] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Packet Loss Metric for IPPM, 1999. Internet Engineering Task Force RFC 2680.

[4] Apple Computer Inc. *Technology Brief; QuickTime and MPEG-4*, 2005.

[5] M. Borella, D. Swider, S. Uludag, and G. Brewster. Internet packet loss: Measurements and implications for end-to-end QoS. In *Proceedings of the International Conference on Parallel Processing*, August 1998.

[6] Jill M. Boyce and Robert D. Gaglianello. Packet loss effects on MPEG video sent over the public Internet. In *Proceedings of the sixth ACM international conference on Multimedia*, 1998.

[7] Prasad Calyam and Chang-Gun Lee. Characterizing voice and video traffic behavior over the Internet. Technical report, The Ohio State University, 2005.

[8] K. Charloner and I. Verdinelly. Bayesian experimental design: a review. *Statistical Science*, 10:273–304, 1995.

[9] D. Choi, E. Biersack, and G. Urvoy-Keller. Cost-optimal dimensioning of a large scale video on demand system. In *Proceedings of NGC, 2002*, August 2002.

[10] Seung-Hwa Chung, Deepali Agrawal, Myung-Sup Kim, James W. Hong, and Kihong Park. Analysis of bursty packet loss characteristics on under-utilized links using snmp, 2004.

[11] Managing Delay in IP Video Networks, 2005. A Cisco System White Paper.

[12] Tim Cox. World broadband statistics: Q1 2006. Technical report, Point Topic Ltd, June 2006.

[13] Nicola Cranley. *User-Perceived Quality-Aware Adaption of Streamed Multimedia over Best-effort IP Networks.* PhD thesis, National University of Ireland, March 2004.

[14] György Dán, Viktória Fodor, and Gunnar Karlsson. Packet size distribution: an aside? In *QoSIP 05*, pages 75–87, February 2005.

[15] György Dán, Viktória Fodor, and Gunnar Karlsson. On the effects of the packet size distribution on the packet loss process. In *Telecommunication Systems Journal*, volume 32, pages 31–53, May 2006.

[16] ETSI TS 102 034 DVP IP Phase 1 Handbook. Digital video broadcasting (dvb); transport of mpeg-2 based dvb services over ip based networks. Technical report, Digital Video Broadcasting Project, March 2005.

[17] Hans-Detlef Shultz (ed.). Understanding Real-Time Internet Services. Technical report, European Institute for Research and Strategic Studies in Telecommunications, 2000.

[18] Juanita Ellis, Charles Pursell, and Joy Rahman. *Voice, Video, and Data Network Convergence*. Academic Press, 2003.

[19] Peder J. Emstad, Poul E. Heegaard, and Bjarne E. Helvik. *Pålitelighet og ytelse i informasjons- og kommunikasjonssystem*. NTNU, 2003. Lecture Notes in subject TTM4110 - Quality of Service.

[20] Arnfinn Flo. *User-Perceived Quality of Service in Packet-Based Internet Services*. Norwegian University of Science and Technology, March 2006. Project Work.

[21] Gary J. Sullivan and Pankaj Topiwala and Ajay Luthra. The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions. In *SPIE Conference on Applications of Digital Image Processing XXVII*, August 2004.

[22] E. Gilbert. Capacity of a burst-loss channel. *Bell Systems Technical Journal*, 5(39), 1960.

[23] David Hands and Miles Wilkins. A study of the impact of network loss and burst size on video streaming quality and acceptability. In *IDMS '99: Proceedings of the 6th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services*, pages 45–57. Springer-Verlag, 1999.

[24] Mads Hansen-Møllerud, Annette Kalvøy, Ole-Petter Kordahl, Geir Martin Pilskog, and Anne-Hege Sølverud. Nøkkeltall om informasjonssamfunnet 2005. Technical report, Statistisk Sentralbyrå (Statistics Norway), June 2006.

[25] D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar. RTP Payload Format for MPEG1/MPEG2 Video, 1998. Internet Engineering Task Force RFC 2250.

[26] International Organisation For Standardisation. *Overview of the MPEG-4 Standard*, 2002. Edited by Rob Koenen.

[27] ITU-R. Methodology for the subjective assessment of the quality of television pictures, ITU-R Recommendation BT.500-11. Technical report, the International Telecommunication Union, 2002.

[28] ITU-T. Video Codec for Audiovisual Services at px64 kbit/s, ITU-T Recommendation H.261. Technical report, the International Telecommunication Union, 1993.

[29] ITU-T. Terms and Definitions Related to Quality of Service and Network Performance including Dependability, ITU-T Recommendation E.800. Technical report, the International Telecommunication Union, 1994.

[30] ITU-T. Subjective Video Quality Assessment Methods for Multimedia Applications, ITU-T Recommendation P.910. Technical report, the International Telecommunication Union, 1999.

[31] ITU-T. Generic coding of moving pictures and associated audio information: Systems, ITU-T Recommendation H.222.0. Technical report, the International Telecommunication Union, 2000.

[32] ITU-T. Advanced video coding for generic audiovisual services, ITU-T Recommendation H.264. Technical report, the International Telecommunication Union, 2005.

[33] Miranda Ko and Irene Koo. An Overview of Interactive Video On Demand System. Technical report, The University of British Columbia, 1996.

[34] Philip Kotler and Gary Armstrong. *Principles of Marketing.* Prentice Hall, eleventh edition, 2005.

[35] F. Kozamernik, V. Steinmann, P. Sunna, and E. Wyckens. SAMVIQ - a New EBU Methology for Video Quality Evaluations in Multimedia. Technical report, The European Broadcasting Union, 2005.

[36] Franz Kozamernik, Paola Sunna, Emmanuel Wyckens, and Dag Inge Pettersen. Subjective quality of Internet video codecs - phase 2 evaluations using SAMVIQ, 2005. EBU Technical Review.

[37] Cheng-Han Lin, Chih-Heng Ke, Ce-Kuen Shieh, and Naveen K. Chilamkurti. The packet loss effect on mpeg video transmission in wireless networks. In *AINA '06: Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06)*, pages 565–572, Washington, DC, USA, 2006. IEEE Computer Society.

[38] Alex MacAulay, Boris Felts, and Yuval Fisher. Whitepaper - ip streaming of mpeg-4: Native rtp vs mpeg-2 transport stream. Technical report, Envivio, October 2005.

[39] Tom Maremaa and William Stewart. *QuickTime for Java: A Developer's Reference*, 1999.

[40] Velibor Markovski, Fei Xue, and Ljiljane Trajkovic. Simulation and analysis of packet loss in video transfers using User Datagram Protocol. Technical report, Kluwer Academic Publishers, 2001.

[41] Paul R. La Monica. Online Video: Must-free TV. Technical report, CNN, 2006. Retrieved June 23 2006 from `http://money.cnn.com/2006/04/10/news/companies/abconline_free/index.htm?cnn=yes`.

[42] S. B. Moon, J. Kurose, P. Skelly, and D. Towsley. Correlation of packet delay and loss in the internet. Technical report, Amherst, MA, USA, 1998.

[43] OECD Broadband Statistics, December 2005. Retrieved June 25 2006 from `http://www.oecd.org/document/39/0,2340,en_2825_495656_36459431_1_1_1_1,00.html`.

[44] Colin Perkins. *RTP: Audio and Video for the Internet*. Addison-Wesley Professional, 2003.

[45] Telegeography Research. Global Internet Geography 2006, 2005.

[46] Ralf Schäfer, Thomas Wiegard, and Heiko Schwartz. *The emerging H.264/AVC standard*. Heinrich Heinz Institute, 2003. EBU technical review.

[47] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications, 2003. Internet Engineering Task Force RFC 2250.

[48] Varian Shapiro. *Information Rules. A Strategic Guide to the Network Economy*. Harvard Business School Press, 1999.

[49] Sun Microsystems. *Java$^{TM}$Media Framework API Guide*, 1999.

[50] Andrew S. Tanenbaum. *Computer Networks*. Pearson Education International, fourth edition, 2003.

[51] Martin Varela, Ian Marsh, and Björn Grönvall. A systematic study of PSEQ's behavior (from a networking perspective). Technical report, Swedish Institute of Computer Science (SICS), June 2006.

[52] Dmitry Vatolin, Dmitriy Kulikov, Alexander Parshin, Artem Titarenko, and Stanislav Soldatov. MPEG-4 AVC/H.264 Video Codec Comparison. Technical report, CS MSU Graphics &Media Lab, December 2005.

[53] Dmitry Vatolin and Oleg Petroc. *MSU Perceptual Video Quality tool 1.0*. CS MSU Graphics & Media Lab, February 2006.

[54] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability & Statistics for Engineers & Scientists*. Prentice Hall, 2001.

[55] Anna Watson and M. Angale Sasse. Measuring Perceived of Speech and Video in Multimedia Conferencing Applications. In *Proceedings of the sixth ACM international conference on Multimedia*, pages 55–60, 1998.

[56] Stefan Winkler and Ruth Cambos. Video Quality Evaluation for Internet Streaming Applications. Technical report, Swiss Federal Institute of Technology (EPFL), 2004.

# Appendix A

# Encoding Logs

The encoding logs obtained from `x264`.

## A.1 800 kbps

```
Starting job job1-1 at 21:51:41
Job is a video job. encoder commandline:
--pass 1 --bitrate 800 --stats "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats" --bframes 1
--direct auto --subme 1 --analyse none --me dia --progress --no-psnr --output NUL "C:\Documents and
Settings\Arnfinn\Desktop\Video Clips\STEM\800.avs"
successfully started encoding
Processing ended at 21:52:32
-------------------------------------------------------------------------------------------------------------

Log for job job1-1

avis [info]: 704x576 @ 25.00 fps (825 frames)
x264 [info]: using cpu capabilities MMX MMXEXT SSE SSE2
x264 [info]: slice I:9     Avg QP:30.22  size: 20451
x264 [info]: slice P:426   Avg QP:33.50  size:  4949
x264 [info]: slice B:390   Avg QP:35.67  size:  1242
x264 [info]: mb I  I16..4: 41.8%  0.0% 58.2%
x264 [info]: mb P  I16..4:  9.7%  0.0%  0.0%  P16..4: 42.3%  0.0%  0.0%  0.0%  0.0%    skip:48.0%
x264 [info]: mb B  I16..4:  0.8%  0.0%  0.0%  B16..8: 13.2%  0.0%  0.0%  direct: 5.3%  skip:80.8%
x264 [info]: final ratefactor: 32.10
x264 [info]: direct mvs  spatial:21.0%  temporal:79.0%
x264 [info]: kb/s:807.0

Actual bitrate after encoding without container overhead: 807.22

-------------------------------------------------------------------------------------------------------------
Job completed successfully and deletion of intermediate files is activated
job job1-1 has been processed. This job is linked to the next job: job1-2
Starting job job1-2 at 21:52:32
Job is a video job. encoder commandline:
--pass 2 --bitrate 800 --stats "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats" --ref 5
--bframes 1 --b-rdo --direct auto --subme 6 --trellis 1 --analyse p8x8,b8x8,i4x4,p4x4 --me umh --progress
--no-psnr --output "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.mp4"
"C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.avs"
successfully started encoding
Processing ended at 21:56:02
-------------------------------------------------------------------------------------------------------------

Log for job job1-2

avis [info]: 704x576 @ 25.00 fps (825 frames)
x264 [info]: using cpu capabilities MMX MMXEXT SSE SSE2
mp4 [info]: initial delay 166833 (scale 5000000)
x264 [info]: slice I:9     Avg QP:30.00  size: 18618
x264 [info]: slice P:426   Avg QP:32.92  size:  4761
x264 [info]: slice B:390   Avg QP:34.96  size:  1444
x264 [info]: mb I  I16..4: 42.0%  0.0% 58.0%
x264 [info]: mb P  I16..4:  7.1%  0.0%  3.9%  P16..4: 33.7%  9.6%  2.0%  0.3%  0.0%    skip:43.4%
x264 [info]: mb B  I16..4:  0.3%  0.0%  0.2%  B16..8: 30.8%  1.3%  2.5%  direct: 0.5%  skip:64.3%
x264 [info]: direct mvs  spatial:25.1%  temporal:74.9%
x264 [info]: ref P  85.0%  7.4%  4.2%  1.6%  1.8%
x264 [info]: ref B  93.2%  3.6%  1.8%  0.8%  0.7%
x264 [info]: kb/s:801.8

Actual bitrate after encoding without container overhead: 802.07
desired video bitrate of this job: 800 kbit/s - obtained video bitrate: 805,153692978044 kbit/s
-------------------------------------------------------------------------------------------------------------
```

```
Job completed successfully and deletion of intermediate files is activated
Found intermediate output file 'C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats', deleting...
Deletion succeeded.
```

## A.2 2048 kbps

```
Starting job job1-1 at 22:00:45
Job is a video job. encoder commandline:
--pass 1 --bitrate 2048 --stats "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats" --bframes 1
--direct auto --subme 1 --analyse none --me dia --progress --no-psnr --output NUL "C:\Documents and
Settings\Arnfinn\Desktop\Video Clips\STEM\800.avs"
successfully started encoding
Processing ended at 22:01:49
--------------------------------------------------------------------------------------------------------------

Log for job job1-1

avis [info]: 704x576 @ 25.00 fps (825 frames)
x264 [info]: using cpu capabilities MMX MMXEXT SSE SSE2
x264 [info]: slice I:9     Avg QP:23.78  size: 37928
x264 [info]: slice P:426   Avg QP:26.52  size: 12367
x264 [info]: slice B:390   Avg QP:28.69  size:  3901
x264 [info]: mb I  I16..4: 25.3%  0.0% 74.7%
x264 [info]: mb P  I16..4:  9.8%  0.0%  0.0%  P16..4: 60.7%  0.0%  0.0%  0.0%  0.0%    skip:29.5%
x264 [info]: mb B  I16..4:  0.9%  0.0%  0.0%  B16..8: 23.7%  0.0%  0.0%  direct:11.9%  skip:63.5%
x264 [info]: final ratefactor: 25.30
x264 [info]: direct mvs  spatial:93.3%  temporal:6.7%
x264 [info]: kb/s:2072.5

Actual bitrate after encoding without container overhead: 2072.72

--------------------------------------------------------------------------------------------------------------
Job completed successfully and deletion of intermediate files is activated
job job1-1 has been processed. This job is linked to the next job: job1-2
Starting job job1-2 at 22:01:49
Job is a video job. encoder commandline:
--pass 2 --bitrate 2048 --stats "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats" --ref 5
--bframes 1 --b-rdo --direct auto --subme 6 --trellis 1 --analyse p8x8,b8x8,i4x4,p4x4 --me umh --progress
--no-psnr --output "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\2048.mp4" "C:\Documents and
Settings\Arnfinn\Desktop\Video Clips\STEM\800.avs"
successfully started encoding
Processing ended at 22:06:19
--------------------------------------------------------------------------------------------------------------

Log for job job1-2

avis [info]: 704x576 @ 25.00 fps (825 frames)
x264 [info]: using cpu capabilities MMX MMXEXT SSE SSE2
mp4 [info]: initial delay 166833 (scale 5000000)
x264 [info]: slice I:9     Avg QP:22.56  size: 40733
x264 [info]: slice P:426   Avg QP:25.46  size: 12233
x264 [info]: slice B:390   Avg QP:27.63  size:  3758
x264 [info]: mb I  I16..4: 26.8%  0.0% 73.2%
x264 [info]: mb P  I16..4:  5.6%  0.0%  7.0%  P16..4: 41.2% 14.3%  5.5%  1.4%  0.7%    skip:24.3%
x264 [info]: mb B  I16..4:  0.4%  0.0%  0.6%  B16..8: 37.0%  3.2%  6.1%  direct: 1.2%  skip:51.6%
x264 [info]: direct mvs  spatial:69.5%  temporal:30.5%
x264 [info]: ref P  79.1% 10.1%  5.6%  2.7%  2.5%
x264 [info]: ref B  90.8%  5.3%  2.3%  1.0%  0.7%
x264 [info]: kb/s:2047.0

Actual bitrate after encoding without container overhead: 2047.22
desired video bitrate of this job: 2048 kbit/s - obtained video bitrate: 2050,30550419772 kbit/s
--------------------------------------------------------------------------------------------------------------
Job completed successfully and deletion of intermediate files is activated
Found intermediate output file 'C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats', deleting...
Deletion succeeded.
```

## A.3 5000 kbps

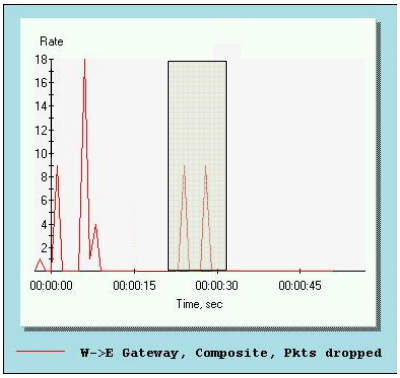
```
Starting job job1-1 at 22:10:50
Job is a video job. encoder commandline:
--pass 1 --bitrate 5000 --stats "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats" --bframes 1
--direct auto --subme 1 --analyse none --me dia --progress --no-psnr --output NUL "C:\Documents and
Settings\Arnfinn\Desktop\Video Clips\STEM\800.avs"
successfully started encoding
Processing ended at 22:12:02
--------------------------------------------------------------------------------------------------------------

Log for job job1-1

avis [info]: 704x576 @ 25.00 fps (825 frames)
x264 [info]: using cpu capabilities MMX MMXEXT SSE SSE2
x264 [info]: slice I:9     Avg QP:18.00  size: 71409
x264 [info]: slice P:426   Avg QP:20.10  size: 29324
x264 [info]: slice B:390   Avg QP:22.29  size: 11338
x264 [info]: mb I  I16..4: 14.7%  0.0% 85.3%
x264 [info]: mb P  I16..4: 10.1%  0.0%  0.0%  P16..4: 79.0%  0.0%  0.0%  0.0%  0.0%    skip:10.9%
x264 [info]: mb B  I16..4:  1.0%  0.0%  0.0%  B16..8: 29.7%  0.0%  0.0%  direct:32.8%  skip:36.5%
x264 [info]: final ratefactor: 19.02
```

```
x264 [info]: direct mvs  spatial:99.5%  temporal:0.5%
x264 [info]: kb/s:5102.3

Actual bitrate after encoding without container overhead: 5102.56

-----------------------------------------------------------------------------------------------------------
Job completed successfully and deletion of intermediate files is activated
job job1-1 has been processed. This job is linked to the next job: job1-2
Starting job job1-2 at 22:12:02
Job is a video job. encoder commandline:
--pass 2 --bitrate 5000 --stats "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats" --ref 5
--bframes 1 --b-rdo --direct auto --subme 6 --trellis 1 --analyse p8x8,b8x8,i4x4,p4x4 --me umh --progress
--no-psnr --output "C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\5000.mp4" "C:\Documents and
Settings\Arnfinn\Desktop\Video Clips\STEM\800.avs"
successfully started encoding
Processing ended at 22:17:45
-----------------------------------------------------------------------------------------------------------

Log for job job1-2

avis [info]: 704x576 @ 25.00 fps (825 frames)
x264 [info]: using cpu capabilities MMX MMXEXT SSE SSE2
mp4 [info]: initial delay 166833 (scale 5000000)
x264 [info]: slice I:9     Avg QP:17.00  size: 74103
x264 [info]: slice P:426   Avg QP:19.01  size: 29891
x264 [info]: slice B:390   Avg QP:21.18  size:  9854
x264 [info]: mb I  I16..4: 22.6%  0.0% 77.4%
x264 [info]: mb P  I16..4:  4.2%  0.0%  8.8%  P16..4: 47.6% 18.9%  9.3%  2.7%  2.0%    skip: 6.6%
x264 [info]: mb B  I16..4:  0.4%  0.0%  1.0%  B16..8: 37.8%  5.1% 11.2%  direct: 5.3%  skip:39.2%
x264 [info]: direct mvs  spatial:85.6%  temporal:14.4%
x264 [info]: ref P  75.9% 11.3%  6.4%  3.4%  3.0%
x264 [info]: ref B  85.5%  7.4%  3.9%  1.9%  1.3%
x264 [info]: kb/s:5011.2

Actual bitrate after encoding without container overhead: 5011.49
desired video bitrate of this job: 5000 kbit/s - obtained video bitrate: 5014,57857785203 kbit/s
-----------------------------------------------------------------------------------------------------------
Job completed successfully and deletion of intermediate files is activated
Found intermediate output file 'C:\Documents and Settings\Arnfinn\Desktop\Video Clips\STEM\800.stats', deleting...
Deletion succeeded.
```

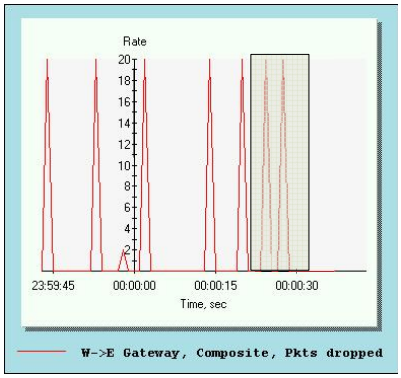
# Appendix B

# Bursty Loss Profiles

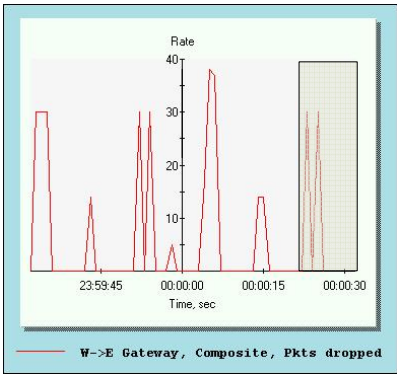
The following pages show the loss characteristics approved for the bursty usage scenarios. The somewhat poor quality is due to the Packetsphere Network Emulator's limited graphical support.



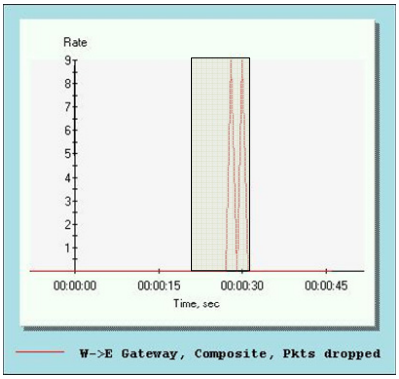

(a) 800 kbps, burst size 10



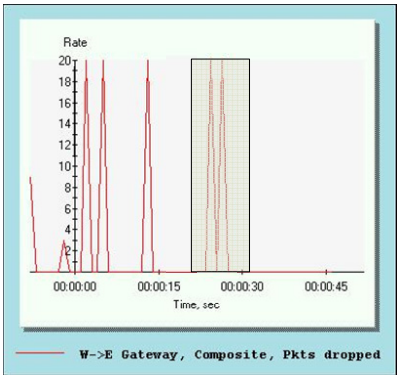

(b) 800 kbps, burst size 20



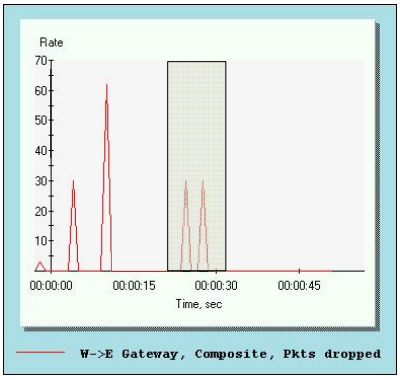

(c) 800 kbps, burst size 30




(d) 2048 kbps, burst size 10

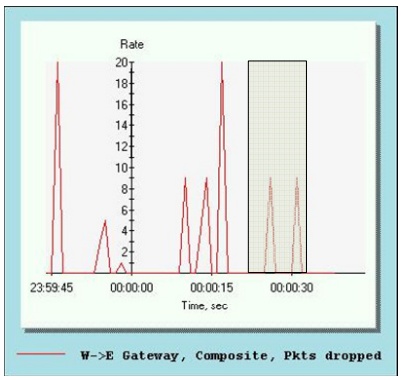



(e) 2048 kbps, burst size 20

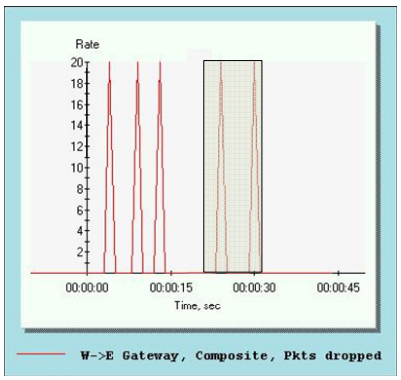



(f) 2048 kbps, burst size 30

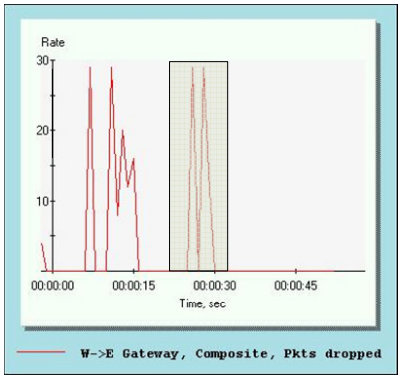
Figure B.1: Bursty Loss Profiles




(a) 5000 kbps, burst size 10




(b) 5000 kbps, burst size 20




(c) 5000 kbps, burst size 30

# Appendix C

# Complete Test Results

Table C.1 on the following page contains the IMSs and related confidence interval borders for all usage scenarios tested.

| Usage Scenario | Bandwidth (Kbps) | Packet Loss | IMS | HCB[a] | LCB[b] |
|---|---|---|---|---|---|
| 1 | 800 | 3 % | 15 | 19 | 10 |
| 2 | 800 | 2 % | 20 | 25 | 15 |
| 3 | 800 | 1 % | 38 | 43 | 34 |
| 4 | 800 | 0,75 % | 47 | 52 | 42 |
| 5 | 800 | 0,5 % | 61 | 65 | 56 |
| 6 | 800 | 0,25 % | 64 | 69 | 60 |
| 7 | 800 | 0,1 % | 70 | 76 | 65 |
| 8 | 800 | 0,05 % | 84 | 88 | 80 |
| 9 | 800 | 0,01 % | 94 | 100 | 89 |
| 10 | 800 | 30 pckts (burst) | 34 | 40 | 29 |
| 11 | 800 | 20 pckts (burst) | 41 | 46 | 37 |
| 12 | 800 | 10 pckts (burst) | 50 | 56 | 44 |
| hidden reference | 800 | 0 | 94 | 100 | 89 |
| 13 | 2048 | 3 % | 2 | 0 | 5 |
| 14 | 2048 | 2 % | 17 | 21 | 11 |
| 15 | 2048 | 1 % | 25 | 30 | 20 |
| 16 | 2048 | 0,75 % | 36 | 42 | 30 |
| 17 | 2048 | 0,5 % | 42 | 46 | 38 |
| 18 | 2048 | 0,25 % | 50 | 56 | 43 |
| 19 | 2048 | 0,1 % | 61 | 67 | 56 |
| 20 | 2048 | 0,05 % | 78 | 82 | 73 |
| 21 | 2048 | 0,01 % | 81 | 86 | 77 |
| 22 | 2048 | 30 pckts (burst) | 39 | 43 | 34 |
| 23 | 2048 | 20 pckts (burst) | 44 | 50 | 39 |
| 24 | 2048 | 10 pckts (burst) | 54 | 59 | 49 |
| hidden reference | 2048 | 0 | 98 | 100 | 94 |
| 25 | 5000 | 3 % | 3 | 7 | 0 |
| 26 | 5000 | 2 % | 12 | 17 | 8 |
| 27 | 5000 | 1 % | 24 | 29 | 19 |
| 28 | 5000 | 0,75 % | 33 | 37 | 29 |
| 29 | 5000 | 0,5 % | 40 | 46 | 34 |
| 30 | 5000 | 0,25 % | 47 | 52 | 42 |
| 31 | 5000 | 0,1 % | 77 | 83 | 72 |
| 32 | 5000 | 0,05 % | 90 | 94 | 86 |
| 33 | 5000 | 0,01 % | 98 | 100 | 95 |
| 34 | 5000 | 30 pckts (burst) | 44 | 50 | 38 |
| 35 | 5000 | 20 pckts (burst) | 51 | 57 | 46 |
| 36 | 5000 | 10 pckts (burst) | 72 | 78 | 67 |
| hidden reference | 5000 | 0 | 99 | 100 | 94 |

[a] High 95% confidence interval border
[b] Low 95% confidence interval border

Table C.1: Complete Test Results

# Appendix D

# SSAT

## D.1 Documentation

### Introduction to SSAT

**SSAT**, SAMVIQ Subjective Assessment Tool, is a partially developed tool for subjective video quality evaluation. It can be used to evaluate the individual quality of different video clips in accordance with the SAMVIQ methodology [1].

SSAT was developed in conjunction with the master's thesis "User-Perceived Quality of Service in Video on Demand Services", and has been put to the test in a 30 person subjective assessment. During this assessment, SSAT proved itself to be user-friendly, stable at runtime and intuitive.

SSAT is not to be considered a fully developed application; it is not optimized, nor is the source code well documented. In addition, it lacks functionality for easy management of test scenarios and analysis of the results collected. However, it demonstrates an efficient way of presenting video-content to test-subjects and may serve as a guide for a more complete development of a subjective assessment tool at a later stage.

An example implantation of SSAT can be downloaded from `http://folk.ntnu.no/flo/SSAT`. It requires the latest version of QuickTime installed, downloadable from `http://www.apple.com/quicktime/download/`, and includes the video clips used in the master's thesis. For questions or comments, send an e-mail to `arnfinn.flo@gmail.com`.

### Basic Architecture

Multimedia support in SSAT is incorporated via QuickTime for Java (QTJava). QTJava is a set of cross-platform APIs which allow Java developers to build

[1] For an introduction to the SAMVIQ methodology, see for example `http://www.ebu.ch/en/technical/trev/trevindex-xz.html`

multimedia into Java applications independently of SUN's inferior Java Media Framework. QTJava requires the latest version of the QuickTime player to be installed on the host machine, and is then able to control all file-formats supported by QuickTime.

Three main classes makes up the basic structure of SSAT:

**Application.java** is the main control class, and handles all GUI interaction. It creates as many instances of `PlayAction.java` as there are video clips in each test-case.

**PlayAction.java** controls the playback of each video clip. It initializes a QuickTime-session and remains in control of it's performance. Each instance of `PlayAction` stores it's own current voting-score, and sends information to the `Application` class so that it can update it's user-feedback.

**Test.java** handles all file reading and writing. It reads information from config-files, which in turn is used to set up individual tests. An instance of the `Test` class is created each time the user loads a test-case.

In addition, the current version of SSAT includes two more classes; `LoginDialog.java` which handles basic admission control and `ColorTest.java`, which adds a color blindness test as required by the SAMVIQ methodology.

### Step by Step Process of Prepering a Subjective Test

Even with the current version of SSAT, it is very easy to make subjective quality evaluations. The following steps describe the process in steps:

**1** Download SSAT from `http://folk.ntnu.no/flo/SSAT`. Make sure you have the latest version of QuickTime installed.

**2** Copy the video clips you want do put under test into the following directory: `ssat\Tests\Test#\Clips`, substituting # with 0 for a pretest case or any other number for a real test case. The test case will show as `Test #` in the file menu of SSAT. Repeat the process with different values of # for more loadable test cases. Please note that each test case must consist of video files of the same resolution and with the same frame rate for optimal performance in SSAT.

**3** Prepare the `config.sam` file for each test case, located in `ssat\Tests\Test#\Clips`. The file format is as follows:

**Line 1** Number of test clips (including the reference clip)

**Line 2** Horizontal resolution

**Line 3** Vertical resolution

**Line 4** Frame rate

**Line 5** Name of reference clip

**Line 6** Location of reference clip

**Line 7** Name of test clip 1

**Line 8** Location of test clip 1

Repeat line 7 and 8 for the reminding test clips.

An example is provided below. Here, four clips in addition to the reference clip is to be rated. The resolution is 704x576 pixels and the frame rate is 30 frames/second.

```
5
704
576
30
REF
/Tests/Test1/Clips/5000_constant_0.mp4
A
/Tests/Test1/Clips/5000_constant_2.mp4
B
/Tests/Test1/Clips/5000_burst_30(60).mp4
C
/Tests/Test1/Clips/5000_burst_10(20).mp4
D
/Tests/Test1/Clips/5000_constant_0,1.mp4
```

**4** Edit the `users.sam` file, located as `ssat\Tests\` to include user names and passwords in order to allow for authentication of the assessors. File format as follows:

**Line 1** Number of assessors

**Line 2** User name, assessor 1

**Line 3** Password, assessor 1

Repeat line 2 and 3 for the remanding assessors. There are no restrictions on the user name or password format. An example file for three assessors is provided below.

```
3
Mnadsdf
143243
Lkjdfsf
1230fk
Geir
tullepassord
```

**5** Start SSAT by evoking ssat.exe, present the assessor(s) with the user interface and make them log in by choosing `Log in` from the File-menu. The assessor can then load any test case defined in step 3, and rate the individual clips as appropriate. When all test cases are saved and finished, the assessor may close the program.

**6** Collect the results from `ssat\Tests\Test#\Results`. Results are saved in files named `user name + test case id`.

## D.2 Source Code

The source code of the main classes `Application.java`, `PlayAction.java` and `Test.java` are depicted on the following pages. The complete source code is downloadable from `http://folk.ntnu.no/flo/SSAT`.

### Application.java

```
1  /**
2   * The Application class is the main control class of SSAT.
3   * It controls the GUI and links to instances of the PlayAction class,
4   which in turn
5   * hold the different video clips.
6   *
7   * As today, it includes a main method for startup.
8   */
9
10 package ssat;
11
12 import java.awt.BorderLayout;
13 import java.awt.Container;
14 import java.awt.GridLayout;
15 import java.awt.event.ActionEvent;
16 import java.awt.event.ActionListener;
17 import java.awt.event.WindowAdapter;
18 import java.awt.event.WindowEvent;
19 import java.util.Dictionary;
20 import java.util.Hashtable;
21
22 import javax.swing.ImageIcon;
23 import javax.swing.JButton;
24 import javax.swing.JDialog;
25 import javax.swing.JFrame;
26 import javax.swing.JLabel;
27 import javax.swing.JMenu;
28 import javax.swing.JMenuBar;
29 import javax.swing.JMenuItem;
30 import javax.swing.JOptionPane;
31 import javax.swing.JPanel;
32 import javax.swing.JSlider;
33 import javax.swing.JToolBar;
34 import javax.swing.border.TitledBorder;
35 import javax.swing.event.ChangeEvent;
36 import javax.swing.event.ChangeListener;
37
38 import quicktime.app.anim.Compositor;
39 import quicktime.app.display.QTCanvas;
40
41 public class Application extends JFrame implements ActionListener,
42 ChangeListener {
43
44   Container contentPane;
45   JToolBar jtb;
46   JPanel playerPanel, statusPanel, buttons;
47   JMenuBar menuBar;
48   JMenu loadMenu, loginmenu;
49   JMenuItem loginItem, logoutItem, saveItem, testMenuItem[];
50   JLabel userLabel, testLabel;
51   JDialog frame;
```

```
52  JButton saveButton, finishButton;
53  TitledBorder playerPanelBorder;
54
55  static JSlider slider;
56  boolean first, saved, testInProgress, colorTest;;
57  String filename, buttonText;
58  int hres, vres, numberOfClips, numberOfTests;;
59  static String user, root;
60  static boolean testFinished;
61
62  QTCanvas myQTCanvas;
63  Compositor compositor;
64
65  Test activeTest;
66  PlayAction [] videoArray;
67
68  public static int STATE = 0;
69  public final static int TEST_IN_PROGRESS_NOT_SAVED = 1;
70  public final static int TEST_IN_PROGRESS_SAVED = 2;
71  public final static int TEST_NOT_IN_PROGRESS = 0;
72
73  public Application() {
74
75    super("SSAT - SAMVIQ Subjective Assessment Tool");
76
77    first = true;
78    numberOfTests = 4;
79    setDefaultCloseOperation(DO_NOTHING_ON_CLOSE);
80
81    addWindowListener(new WindowAdapter(){
82      public void windowClosing(WindowEvent e){
83
84        if (STATE==TEST_IN_PROGRESS_NOT_SAVED) {
85          setEnabled(false);
86          int action = JOptionPane.showConfirmDialog(null, "Save Test-Scenario
87          before exiting?", "SAMVIQ",
88              JOptionPane.YES_NO_CANCEL_OPTION);
89
90          if (action == JOptionPane.YES_OPTION) {
91            activeTest.save();
92            System.exit(0);
93          } else if (action == JOptionPane.NO_OPTION) {
94            System.out.println("Exit uten á save");
95            System.exit(0);
96          } else if (action == JOptionPane.CANCEL_OPTION) {
97                    setEnabled(true);
98                    toFront();
99          }
100       } else if (STATE==TEST_IN_PROGRESS_SAVED){
101         setEnabled(false);
102         int action = JOptionPane.showConfirmDialog(null, "Are you sure you
103         want to exit?", "SAMVIQ",
104             JOptionPane.YES_NO_OPTION);
105         if (action == JOptionPane.YES_OPTION) {
106           System.exit(0);
107         } else if (action == JOptionPane.NO_OPTION) {
108           setEnabled(true);
109                   toFront();
110         }
111       } else {
112         System.exit(0);
113       }
```

```
114       }
115     });
116
117     contentPane = getContentPane();
118     saved = true;
119     testInProgress = false;
120
121     String userdir = System.getProperty("user.dir");
122     System.out.println("Current user directory " + userdir);
123     root = userdir + "/ssat";
124
125     //Window size, default is fullscrean
126     setExtendedState(JFrame.MAXIMIZED_BOTH);
127
128     //Menu
129     makeJMenuBar();
130
131     //Statusline (NORTH)
132     makeStatusLine();
133
134     //Invoke QuickTime
135     PlayAction.openSession();
136   }
137
138   //Create all menus
139   public void makeJMenuBar() {
140
141     menuBar = new JMenuBar();
142
143     //File-menu
144     JMenu menu = new JMenu("File");
145     menu.setMnemonic('F');
146     menuBar.add(menu);
147
148     //Login-submenu
149     loginItem = new JMenuItem("Login");
150     loginItem.setMnemonic('L');
151     menu.add(loginItem);
152     loginItem.addActionListener(this);
153
154     logoutItem = new JMenuItem("Log Out");
155     logoutItem.setMnemonic('L');
156     menu.add(logoutItem);
157     logoutItem.addActionListener(this);
158     logoutItem.setEnabled(false);
159
160     //Load Test-submenu
161     loadMenu = new JMenu("Open Test");
162     loadMenu.setMnemonic('O');
163     loadMenu.setEnabled(false);
164     loadMenu.getPopupMenu().setLightWeightPopupEnabled(false);
165
166     testMenuItem = new JMenuItem[numberOfTests];
167
168     testMenuItem[0] = new JMenuItem("Pretest");
169     loadMenu.add(testMenuItem[0]);
170     testMenuItem[0].addActionListener(this);
171     testMenuItem[0].setActionCommand("Test " + 0);
172     loadMenu.addSeparator();
173
174     for (int i=1; i<numberOfTests ; i++) {
175       testMenuItem[i] = new JMenuItem("Test " + i);
```

```
176       loadMenu.add(testMenuItem[i]);
177       testMenuItem[i].addActionListener(this);
178     }
179     menu.add(loadMenu);
180
181     //Save Test - submenu
182     saveItem = new JMenuItem("Save Results");
183     saveItem.setMnemonic('S');
184     menu.add(saveItem);
185     saveItem.addActionListener(this);
186     saveItem.setEnabled(false);
187
188     menuBar.add(menu);
189     menu.getPopupMenu().setLightWeightPopupEnabled(false);
190     setJMenuBar(menuBar);
191   }
192
193   //Invoke a test-scenario
194   public void startTest(Test t) {
195
196     String text = t.getStringTestID();
197     if (text.equals("0")) text = "Pretest";
198     else text = "#" + text;
199
200     testLabel.setText("Test: " + text);
201     activeTest = t;
202     testInProgress = true;
203
204     //Draws the player panel (CENTER)
205     if (playerPanel !=null) {
206       playerPanel.remove(myQTCanvas);
207       playerPanelBorder = new TitledBorder("Playing test-clip: none");
208       playerPanel.setBorder(playerPanelBorder);
209
210     } else {
211       playerPanel = new JPanel(true);
212       playerPanelBorder = new TitledBorder("Playing test-clip: none");
213       playerPanel.setBorder(playerPanelBorder);
214       contentPane.add(playerPanel, BorderLayout.CENTER);
215     }
216
217     //Adds the QuickTime Canvas on which to display the video
218     myQTCanvas = new QTCanvas();
219     playerPanel.add(myQTCanvas);
220     playerPanel.setVisible(true);
221
222     //Draws out the playback-buttons (SOUTH)
223     makeButtons();
224
225     //Draws the static voting slider (EAST) and initializes it to a
226     //score of 100
227     if (first) {
228       makeSlider();
229       first = false;
230     } else {
231       slider.setValue(100);
232       slider.setEnabled(false);
233       slider.setVisible(true);
234     }
235     //Redraw Screen
236     setVisible(true);
237     slider.setVisible(true);
```

```
238   }
239
240   //Draws the status line (NORTH)
241   public void makeStatusLine(){
242
243     statusPanel = new JPanel();
244     statusPanel.setBorder(new TitledBorder("Status"));
245     statusPanel.setLayout(new GridLayout());
246
247     userLabel = new JLabel("User: " + "No user logged in", JLabel.LEFT);
248     statusPanel.add(userLabel);
249
250     testLabel = new JLabel("Test: " + "No test loaded", JLabel.LEFT);
251     statusPanel.add(testLabel);
252
253     contentPane.add(statusPanel, BorderLayout.NORTH);
254
255   }
256
257   //Initializes the voting slider
258   public void makeSlider() {
259
260     slider = new JSlider(JSlider.VERTICAL,0,100,100);
261     slider.setMinorTickSpacing(4);
262     slider.setMajorTickSpacing(20);
263     slider.setPaintTicks(true);
264     slider.setPaintLabels(true);
265
266     // Creates the label table
267     Dictionary labelTable = new Hashtable();
268     labelTable.put(new Integer(0), new JLabel("0"));
269     labelTable.put(new Integer(10), new JLabel("     bad"));
270     labelTable.put(new Integer(20), new JLabel("20"));
271     labelTable.put(new Integer(30), new JLabel("     poor"));
272     labelTable.put(new Integer(40), new JLabel("40"));
273     labelTable.put(new Integer(50), new JLabel("     fair"));
274     labelTable.put(new Integer(60), new JLabel("60"));
275     labelTable.put(new Integer(70), new JLabel("     good"));
276     labelTable.put(new Integer(80), new JLabel("80"));
277     labelTable.put(new Integer(90), new JLabel("     excellent"));
278     labelTable.put(new Integer(100), new JLabel("100"));
279
280     slider.setLabelTable(labelTable);
281
282     TitledBorder titled = new TitledBorder("Voting");
283     slider.setBorder(titled);
284
285     slider.addChangeListener(this);
286     slider.setEnabled(false);
287
288     contentPane.add(slider, BorderLayout.EAST);
289   }
290
291   //Makes the playback-buttons and links each to an indiviual
292   //PlayAction-object
293   public void makeButtons(){
294
295     String videofile = null;
296       videoArray = new PlayAction[activeTest.getNumberOfClips()];
297
298       if (jtb == null) jtb = new JToolBar();
299       else jtb.removeAll();
```

```
300
301       jtb.setLayout(new BorderLayout());
302
303       buttons = new JPanel();
304
305       for (int i=0; i<activeTest.getNumberOfClips(); i++){
306        buttonText = activeTest.getButtonName(i);
307        videofile = activeTest.getFileName(i);
308        System.out.println(videofile);
309        videoArray[i]= new PlayAction(this, playerPanel, videofile,
310        buttonText, videoArray, i, myQTCanvas);
311        buttons.add(videoArray[i]);
312     }
313
314       jtb.setFloatable(false);
315       jtb.add(buttons, BorderLayout.WEST);
316
317       //Save-button
318       ImageIcon saveIcon = new ImageIcon(Application.root + "/graphics/save.
319       gif");
320       saveButton = new JButton("SAVE", saveIcon);
321       saveButton.addActionListener(this);
322       saveButton.setEnabled(false);
323       buttons.add(saveButton);
324
325       //Finish Test Button - only enabled when all test clips have
326       //been viewed
327       finishButton = new JButton("Finish Test");
328       finishButton.addActionListener(this);
329       finishButton.setEnabled(false);
330       buttons.add(finishButton);
331
332       jtb.add(buttons, BorderLayout.WEST);
333
334      if (first) contentPane.add(jtb, BorderLayout.SOUTH);
335      else jtb.setVisible(true);
336       jtb.addSeparator();
337       setVisible(true);
338   }
339
340   //Controls the menu actions
341   public void actionPerformed(ActionEvent e) {
342
343     String cmd = e.getActionCommand();
344
345     //Load Test Scenario
346     if (cmd.regionMatches(0, "Test", 0, numberOfTests))  {
347       //Determines which test is to be loaded
348       int testnr = Integer.valueOf(cmd.substring(5, cmd.length()));
349
350       if (STATE==TEST_IN_PROGRESS_NOT_SAVED) {
351         //If yes or no
352         if (saveDialog()) {
353           STATE = TEST_IN_PROGRESS_SAVED;
354           startTest(new Test(testnr, root + "/Tests/Test"+testnr+"/Clips/config.
355           sam", this));
356         }
357       } else {
358         startTest(new Test(testnr, root + "/Tests/Test"+testnr+"/Clips/config.
359         sam", this));
360       }
361     } else if (e.getActionCommand().equals("Finish Test"))  {
```

```
362       finishTest();
363     }
364
365     //Login
366     else if (e.getActionCommand().equals("Login"))  {
367       setEnabled(false);
368       new LoginDialog(this);
369
370     //Logout
371     } else if (e.getActionCommand().equals("Log Out")) {
372       if (STATE==TEST_IN_PROGRESS_NOT_SAVED) {
373         if (saveDialog()) {
374           setUser(null);
375           dispose();
376           Application app = new Application();
377           app.setVisible(true);
378         }
379       } else {
380         setUser(null);
381         dispose();
382         Application app = new Application();
383         app.setVisible(true);
384       }
385     }
386
387     //Save results
388     else if (e.getActionCommand().equals("Save Results") || e.
389     getActionCommand().equals("SAVE"))  {
390       STATE = TEST_IN_PROGRESS_SAVED;
391       saved=true;
392       activeTest.save();
393       saveButton.setEnabled(false);
394       saveItem.setEnabled(false);
395     }
396
397   }
398
399   //Checks if the user really want to finish the current test-scenario
400   private void finishTest() {
401
402     int id = activeTest.getIntTestID();
403     videoArray[PlayAction.activeVideo].stop();
404     setEnabled(false);
405
406     int action = JOptionPane.showConfirmDialog(null, "This will make your
407     results final. Are you sure you want to finish this test-scenario?",
408     "Confirm",
409         JOptionPane.YES_NO_OPTION);
410
411     if (action == JOptionPane.YES_OPTION) {
412       testMenuItem[id].setEnabled(false);
413       testMenuItem[id].setText("Test " + id + " - FINISHED");
414       finishButton.setEnabled(false);
415       jtb.setVisible(false);
416       slider.setVisible(false);
417       playerPanel.setVisible(false);
418       testLabel.setText("Test: " + "No test loaded");
419
420       setEnabled(true);
421            toFront();
422
423     } else if (action == JOptionPane.NO_OPTION) {
```

```
424       setEnabled(true);
425            toFront();
426            videoArray[PlayAction.activeVideo].start();
427     }
428
429   }
430
431   //Returns false if cancelled
432   public boolean saveDialog() {
433     setEnabled(false);
434
435     int action = JOptionPane.showConfirmDialog(null, "Save Test before
436     exiting?", "SAMVIQ",
437         JOptionPane.YES_NO_CANCEL_OPTION);
438
439     if (action == JOptionPane.YES_OPTION) {
440       activeTest.save();
441       setEnabled(true);
442            toFront();
443       return true;
444     } else if (action == JOptionPane.NO_OPTION) {
445       System.out.println("Exit uten å save");
446       setEnabled(true);
447            toFront();
448       return true;
449     } else if (action == JOptionPane.CANCEL_OPTION) {
450            setEnabled(true);
451            toFront();
452     }
453     return false;
454   }
455
456   //Invoked if the slider value changes. Sets video clip score
457   //and disables the save option.
458   public void stateChanged(ChangeEvent arg0) {
459     saveButton.setEnabled(true);
460     saveItem.setEnabled(true);
461     finishButton.setEnabled(false);
462     saved = false;
463     STATE = TEST_IN_PROGRESS_NOT_SAVED;
464     if (PlayAction.activeVideo!=-1) activeTest.setRating(PlayAction.
465     activeVideo, slider.getValue());
466   }
467
468   //Returns the grade of the current playing video clip.
469   public int getRating(int videoID) {
470     return activeTest.getRating(videoID);
471   }
472
473   //Registrates or removes a user
474   public void setUser(String user) {
475
476     if (user!=null) {
477       Application.user=user;
478       userLabel.setText("User: " + Application.user);
479       loadMenu.setEnabled(true);
480       loginItem.setEnabled(false);
481       logoutItem.setEnabled(true);
482       STATE = TEST_NOT_IN_PROGRESS;
483     } else {
484       Application.user=null;
485       userLabel.setText("User: " + "No user logged in");
```

```
486       loginItem.setEnabled(true);
487       logoutItem.setEnabled(false);
488       loadMenu.setEnabled(false);
489
490       contentPane.remove(slider);
491       jtb.setVisible(false);
492
493       testLabel.setText("Test: " + "No test loaded");
494       setVisible(true);
495       STATE = TEST_NOT_IN_PROGRESS;
496     }
497   }
498
499
500   public void setColorTest(boolean colorTest) {
501     this.colorTest = colorTest;
502   }
503
504   public boolean getColorTest() {
505     return colorTest;
506   }
507
508   public static void main(String[] args) {
509     Application app = new Application();
510     app.setVisible(true);
511   }
512 }
```

## PlayAction.java

```
1  /**
2   * The PlayAction class holds the basic QuickTime video functionality.
3   * Each instance of this class is visualized as a button in the GUI and
4   controls
5   * it's own QuickTime for Java movie player.
6   */
7  package ssat;
8
9  import java.awt.Color;
10 import java.awt.Component;
11 import java.awt.event.ActionEvent;
12 import java.awt.event.ActionListener;
13 import java.io.IOException;
14
15 import javax.swing.JButton;
16 import javax.swing.JPanel;
17 import javax.swing.border.TitledBorder;
18
19 import quicktime.QTException;
20 import quicktime.QTSession;
21 import quicktime.app.anim.Compositor;
22 import quicktime.app.display.QTCanvas;
23 import quicktime.app.players.MoviePlayer;
24 import quicktime.app.players.MoviePresenter;
25 import quicktime.app.players.QTPlayer;
26 import quicktime.io.OpenMovieFile;
27 import quicktime.io.QTFile;
28 import quicktime.std.movies.Movie;
29 import quicktime.std.movies.MovieController;
```

```
30
31 public class PlayAction extends JButton implements ActionListener{
32
33   JPanel playerPanel;
34   String videofile, buttonText;
35   static int activeVideo = -1;
36   int videoID;
37   Component comp, activecomp, control;
38
39   Application manager;
40   PlayAction [] videoArray;
41
42   Movie movie;
43   MoviePlayer moviePlayer;
44   MoviePresenter md;
45   QTCanvas myQTCanvas;
46   Compositor compositor;
47   QTPlayer qtPlayer;
48
49   public PlayAction(Application manager, JPanel playerPanel, String
50   videofile, String buttonText, PlayAction[] videoArray, int videoID,
51   QTCanvas myQTCanvas){
52
53     super(buttonText + " ("+manager.activeTest.getRating(videoID)+")");
54
55     this.compositor = compositor;
56     this.myQTCanvas = myQTCanvas;
57     this.manager = manager;
58     this.playerPanel = playerPanel;
59     this.videofile = videofile;
60     this.buttonText = buttonText;
61     this.videoArray = videoArray;
62     this.videoID = videoID;
63     this.addActionListener(this);
64
65     activeVideo=-1;
66     setBackground(Color.RED);
67
68   }
69
70   //Sets green button color
71   public void setGreen(){
72     setBackground(Color.GREEN);
73   }
74
75   //Prints the current grade on the button
76   public void setButtonText(int grade){
77     setText(buttonText + " ("+String.valueOf(grade)+")");
78   }
79
80   //Plays the video when the button is pressed
81   public void actionPerformed(ActionEvent ae) {
82
83   if (!buttonText.equals("REF")) Application.slider.setEnabled(true);
84   else {
85       Application.slider.setEnabled(false);
86       setGreen();
87   }
88
89   try {
90       //Stops current playing video (if any)
91       if (activeVideo != -1) videoArray[activeVideo].stop();
```

```
92
93        //Displays the name of the video clip
94        TitledBorder titled = new TitledBorder("Playing test-clip: "+button-
Text)
95        ;
96        playerPanel.setBorder(titled);
97
98        //Sets the slider value to the current video's grade
99        activeVideo = videoID;
100       Application.slider.setValue(manager.getRating(activeVideo));
101       System.out.println("Aktiv video settes nå til: " + activeVideo + " og
102       starter");
103
104       //Loads and displays the video
105       movie = makeMovie(new QTFile(videofile));
106       moviePlayer = new MoviePlayer(movie);
107       qtPlayer = new QTPlayer(new MovieController(movie));
108       myQTCanvas.setClient(qtPlayer, true);
109       //Starts the video
110       moviePlayer.setRate(1);
111
112
113   } catch (Exception e) {
114         System.out.println(e.toString());
115     }
116
117   }
118
119   //Returns a movie from a file, handles all file-types that
120   //the installed QuickTime Player does.
121   protected Movie makeMovie(QTFile f) throws IOException, QTException{
122
123     OpenMovieFile movieFile = OpenMovieFile.asRead(f);
124     Movie m = Movie.fromFile(movieFile);
125     return m;
126    }
127
128     //Starts a movie
129   public void start() {
130     try {
131       moviePlayer.setRate(1);
132     } catch (QTException e) {
133       e.printStackTrace();
134     }
135   }
136
137   //Stops a movie
138   public void stop() {
139        try {
140         movie.stop();
141        } catch (Exception e) {}
142   }
143
144   // Open the QuickTime session
145   protected static void openSession() {
146      try {
147        QTSession.open();
148      } catch( Exception e ) {
149       e.printStackTrace();
150         return;
151      }
152   }
```

```
153
154     // Close the QuickTime session
155   public static boolean close() {
156      try {
157          QTSession.close();
158      } catch( Exception e ) {
159          e.printStackTrace();
160      }
161          return true;
162    }
163 }
```

### Test.java

```
1  /**
2   * The Test class holds and controls the individual test data.
3   * An instance of the class is created everytime the user loads a
4   * test scenario, and writes the data submitted by the testers to file.
5   */
6
7  package ssat;
8
9  import java.io.BufferedReader;
10 import java.io.BufferedWriter;
11 import java.io.FileReader;
12 import java.io.FileWriter;
13 import java.io.IOException;
14 import java.util.Calendar;
15
16 public class Test {
17
18   int testID, hres, vres, rate, numberOfClips, rating[];
19   String user, buttonName[], fileName[], filename;
20   Boolean [] isRated;
21   PlayAction [] videoArray;
22   Application app;
23
24   public Test(int testID, String filename, Application app) {
25
26     this.testID = testID;
27     BufferedReader input = null;
28     user = Application.user;
29     this.app = app;
30
31     //Reads test data from file
32     try {
33         input = new BufferedReader(new FileReader(filename));
34         numberOfClips = Integer.parseInt(input.readLine());
35
36         videoArray = new PlayAction[numberOfClips];
37         buttonName = new String[numberOfClips];
38         fileName = new String[numberOfClips];
39         rating = new int[numberOfClips];
40         isRated = new Boolean[numberOfClips];
41
42         hres = Integer.parseInt(input.readLine());
43         vres = Integer.parseInt(input.readLine());
44         rate = Integer.parseInt(input.readLine());
45
```

```
46          //REF-clip
47          buttonName[0] = input.readLine();
48          fileName[0] = Application.root + input.readLine();
49          rating[0] = 100;
50          isRated[0] = false;
51
52          //Test-clips
53          for (int i=1; i<numberOfClips; i++){
54            buttonName[i] = input.readLine();
55            fileName[i] = Application.root + input.readLine();
56            rating[i] = 100;
57            isRated[i] = false;
58          }
59      } catch (Exception e) {
60
61      }
62      try {
63        if (input!= null) input.close();
64        } catch (IOException ex) {
65              ex.printStackTrace();
66        }
67  }
68
69  //Saves test data to file. File name: "User"+testID
70  public void save() {
71    Calendar cal = Calendar.getInstance();
72
73    try {
74          BufferedWriter out = new BufferedWriter(new FileWriter(
75          Application.root+"/Tests/Test"+testID+"/Results/"+Application.
76          user+testID));
77          out.write("Test saved: " + cal.getTime());
78          out.newLine();
79          out.write("User: " + Application.user);
80          out.newLine();
81          out.write("Test ID: " + testID);
82          out.newLine();
83          out.write("Passed Color Test? " + (app.getColorTest()?"Yes":"No"))
84          ;
85          out.newLine();
86
87          for (int i=0; i<numberOfClips; i++){
88          out.write("Clip: " + buttonName[i] + ": " + rating[i]);
89          out.newLine();
90        }
91
92          Application.STATE = Application.TEST_IN_PROGRESS_SAVED;
93
94          //Checks if test is finished
95          if (isFinished()) app.finishButton.setEnabled(true);
96
97          out.close();
98      } catch (IOException e) {
99      }
100
101 }
102
103 //Sets individual test clip ratings
104 public void setRating(int videoclip, int grade){
105
106   rating[videoclip] = grade;
107   if (isRated[videoclip])  {
```

```
108         app.videoArray[videoclip].setGreen();
109         app.videoArray[videoclip].setButtonText(grade);
110       }
111       isRated[videoclip] = true;
112     }
113
114     //Returns true if all clips are rated
115     public boolean isFinished(){
116
117       for (int i=1; i<numberOfClips; i++) if (!isRated[i]) return false;
118       return true;
119     }
120
121     //Standard get-methods
122     public int getIntTestID(){
123       return testID;
124     }
125
126     public String getStringTestID(){
127       return String.valueOf(testID);
128     }
129
130     public int getNumberOfClips(){
131       return numberOfClips;
132     }
133
134     public String getButtonName(int i){
135       return buttonName[i];
136     }
137
138     public String getFileName(int i){
139       return fileName[i];
140     }
141
142     public int getRating(int i){
143       return rating[i];
144     }
145
146     public int getHres() {
147       return hres;
148     }
149
150     public int getVres() {
151       return vres;
152     }
153
154     public int getRate() {
155       return rate;
156     }
157 }
```