Still Picture Encoding for Digital Video Broadcasting

Implementation of MPEG-4 AVC Still Picture encoding for broadcasting using a MPEG-2 transport stream and evaluating from a bit efficiency point of view

N I C K L A S L U N D I N

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2010
Still Picture Encoding for Digital Video Broadcasting

Implementation of MPEG-4 AVC Still Picture encoding for broadcasting using a MPEG-2 transport stream and evaluating from a bit efficiency point of view

N I C K L A S L U N D I N

Master’s Thesis in Media Technology (30 ECTS credits) at the School of Media Technology Royal Institute of Technology year 2010
Supervisor at CSC was Lars Kjelldahl
Examiner was Nils Enlund

TRITA-CSC-E 2010:036
ISRN-KTH/CSC/E--10/036--SE
ISSN-1653-5715
Still Picture Encoding for Digital Video Broadcasting

Implementation of MPEG-4 AVC Still Picture encoding for broadcasting using a MPEG-2 transport stream and evaluating from a bit efficiency point of view.

Abstract

With the demands of more services and High Definition TV in the Terrestrial Digital Video Broadcast (DVB-T) network, the need for bandwidth increases. It is therefore essential to make broadcasting more efficient when possible.

As for today, the still images sent in the Swedish DVB-T network are encoded just like regular video, with 25 frames per second. Demand for more bit efficient ways to send still images over the DVB-T network has been heard from actors like Boxer and Sveriges Radio (SR).

This master thesis investigates the method of sending still pictures using the AVC still pictures method, available with the introduction of MPEG-4 Advanced Video Coding (AVC). It also evaluates the bitrate efficiency of this method.

The AVC still picture method has its base in that you remove the redundant data which is needed only when encoding moving video. I-, P- and B pictures are removed, keeping only IDR pictures, which are sent with a long interval, for example 2 seconds.

During this master thesis, an AVC still picture concept stream was developed as a project in cooperation with Anton Alila, another master thesis student, and that stream is what the results are based on. The bitrate of this concept stream had a maximum value of 220 Kbit/s, with a picture quality comparable to a conventional non-moving video stream encoded at 700 Kbit/s. I.e. the resulting stream uses less than a third of the original bandwidth.

The concept stream was then used during a controlled observation on 16 consumer receivers where 4 receivers were able to acceptable display the AVC still picture stream. This number should increase if the method is put to use and receiver manufacturers could then update existing receivers over the air.

Despite the low number of receivers currently supporting the AVC still pictures, the method is to be considered a good and bit efficient way of broadcasting still picture material.
Stillbildskodning för utsändning

Implementering av MPEG-4 AVC Still Pictures för utsändning i en MPEG-2 transportström och utvärdering från en biteffektiv synvinkel.

Sammanfattning

I och med kravet om fler kanaler och högupplöst TV ökar även efterfrågan på bandbredd. Det är därför viktigt effektivisera utsändningen av TV där det är möjligt.

I dagsläget sänds stillbilder i det svenska digitala marknätet med samma videokodning som rörlig video, med 25 bilder per sekund. Biteffektiva sätt att sända ut stillbilder i det digitala marknätet har efterfrågats av aktörer som Sveriges Radio (SR) och Boxer.

Den här exjobbsrapporten utreder en metod att sända stillbilder vid namn ”AVC still pictures” som blir tillgänglig i och med introduktionen av videokodningen ”MPEG-4 Advanced Video Coding” (AVC). Rapporten utvärderar även denna metods biteffektivitet och jämför dess bandbreddsanvändning med den konventionella metoden att sända stillbilder.

”AVC still pictures” har sin grund i att man tar bort redundant data som endast är nödvändig när rörlig video kodas. I-, P- och B-bilder tas bort ur videoströmmen och endast IDR-bilder lämnas kvar. Dessa sänds med ett längre intervall, exempelvis 2 sekunder.


Trots det låga antalet mottagare som för närvarande stödjer ”AVC still picture”-strömmar är metoden att betrakta som ett bra och biteffektivt sätt att sända stillbilder i det digitala marknätet.
# Table of Contents

1  Introduction .................................................................................................................. 1  
   1.1  Background ................................................................................................................. 1  
   1.2  Problem Statement ...................................................................................................... 1  
   1.3  Aim ............................................................................................................................... 2  
   1.4  Delimitations .............................................................................................................. 3  

2  Knowledge Base ........................................................................................................... 4  
   2.1  The Human Eye .......................................................................................................... 4  
   2.2  Analog Television ....................................................................................................... 5  
   2.3  Digitizing TV .............................................................................................................. 6  
   2.4  Broadcasting ............................................................................................................. 7  
   2.5  Data Compression ...................................................................................................... 9  
   2.6  Image Compression .................................................................................................... 11  
   2.7  Video Compression ..................................................................................................... 13  
   2.8  Packetizing the Compressed Video .......................................................................... 18  

3  Method ............................................................................................................................ 23  
   3.1  Iterative Development ............................................................................................... 23  
   3.2  The Iterative Development Model in Practice ......................................................... 24  
   3.3  Goals ......................................................................................................................... 24  
   3.4  Reliability and Validity ............................................................................................... 24  

4  Implementation ............................................................................................................... 26  
   4.1  First Thoughts on the Problem .................................................................................. 26  
   4.2  Iterations ................................................................................................................... 27  
   4.3  Tools and Utilities ..................................................................................................... 31  
   4.4  Practical Verification ................................................................................................. 33  

5  Results .............................................................................................................................. 35  
   5.1  Consumer Receivers ................................................................................................. 35  
   5.2  Bitrate Efficiency ...................................................................................................... 36  
   5.3  Picture Quality .......................................................................................................... 38  

6  Conclusions ..................................................................................................................... 39  
   6.1  AVC Still Pictures vs. Conventional Method ............................................................ 39  
   6.2  Introduction on the Swedish Market ....................................................................... 40  
   6.3  Discussion .................................................................................................................. 41  
   6.4  Further Research ....................................................................................................... 41  

7  References ....................................................................................................................... 42  

Appendix A ......................................................................................................................... 43  
Appendix B ......................................................................................................................... 44  
Appendix C ......................................................................................................................... 45
### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ASI</strong></td>
<td>Asynchronous Serial Interface. A streaming data format used to carry the MPEG-2 TS.</td>
</tr>
<tr>
<td><strong>AVC</strong></td>
<td>Advanced Video Coding. Part 10 of the MPEG-4 standard. The part used in this master thesis.</td>
</tr>
<tr>
<td><strong>BER</strong></td>
<td>Bit Error Ratio. The ratio of unharmed bits to erroneous bits.</td>
</tr>
<tr>
<td><strong>DCT</strong></td>
<td>Discrete Cosine Transform. A mathematical transform that describes a signal in a sum of cosines.</td>
</tr>
<tr>
<td><strong>DVB-C</strong></td>
<td>Digital Video Broadcast Cable. The standard used to send digital TV in a cable network.</td>
</tr>
<tr>
<td><strong>DVB-S</strong></td>
<td>Digital Video Broadcast Satellite. The standard used to send digital TV in a satellite network.</td>
</tr>
<tr>
<td><strong>DVB-T</strong></td>
<td>Digital Video Broadcast Terrestrial. The standard used to send digital TV in a terrestrial network.</td>
</tr>
<tr>
<td><strong>ES</strong></td>
<td>Elementary Stream. An endless bit stream of data representing images.</td>
</tr>
<tr>
<td><strong>FEC</strong></td>
<td>Forward Error Correction. Measures taken to be able to correct errors arisen during transmission.</td>
</tr>
<tr>
<td><strong>FPS</strong></td>
<td>Frames Per Second. The number of complete frames used during a one second time window.</td>
</tr>
<tr>
<td><strong>GOP</strong></td>
<td>Group Of Pictures. The number of frames from one I picture to the next.</td>
</tr>
<tr>
<td><strong>HEX</strong></td>
<td>Hexadecimal. Numbers in base 16.</td>
</tr>
<tr>
<td><strong>IDTV</strong></td>
<td>Integrated Digital TV. A television with an integrated digital receiver/decoder.</td>
</tr>
<tr>
<td><strong>MPEG</strong></td>
<td>Moving Picture Experts Group. A group formed to set standards for video compression and transmission.</td>
</tr>
<tr>
<td><strong>MPTS</strong></td>
<td>Multiple Program Transport Stream. A TS with more than one program/channel.</td>
</tr>
<tr>
<td><strong>NAL</strong></td>
<td>Network Abstraction Layer. A package layer in MPEG-4. Contains the ES and is packetized in PES-packets.</td>
</tr>
<tr>
<td><strong>NIT</strong></td>
<td>Network Information Table. One of the PSI tables. Carries information on the network.</td>
</tr>
<tr>
<td><strong>PAT</strong></td>
<td>Program Association Table. A PSI table. Contains links to the program map tables.</td>
</tr>
<tr>
<td><strong>PCR</strong></td>
<td>Program Clock Reference. An embedded timestamp to ensure proper audio-video synchronization.</td>
</tr>
<tr>
<td><strong>PES</strong></td>
<td>Packetized Elementary Stream. The elementary stream is packetized in to PES units.</td>
</tr>
<tr>
<td><strong>PID</strong></td>
<td>Packet Identifier. Each ES or table in the transport stream is identified by a 13 bit number.</td>
</tr>
<tr>
<td><strong>PMT</strong></td>
<td>Program Map Table. Contains information on a program. At which PID streams are located.</td>
</tr>
<tr>
<td><strong>PPS</strong></td>
<td>Picture Parameter Set. A NAL unit for sets of parameters for one or more pictures inside a SPS.</td>
</tr>
<tr>
<td><strong>PSI</strong></td>
<td>Program Specific Information. Information other than audio/video/data streams that needs to be sent in the transport stream.</td>
</tr>
<tr>
<td><strong>QEF</strong></td>
<td>Quasi Error Free. A bit error ratio threshold at which the consumers experience is not tainted.</td>
</tr>
<tr>
<td><strong>RBSP</strong></td>
<td>Raw Byte Sequence Payload. An ordered sequence of bytes that contain a string of data bits.</td>
</tr>
<tr>
<td><strong>RGB</strong></td>
<td>Red Green Blue. An additive color model.</td>
</tr>
<tr>
<td><strong>SEI</strong></td>
<td>Supplemental Enhancement Information. A NAL unit containing information not essential for decoding.</td>
</tr>
<tr>
<td><strong>SPS</strong></td>
<td>A NAL unit where parameters for the whole video sequence is kept.</td>
</tr>
<tr>
<td><strong>SPTS</strong></td>
<td>Single Program Transport Stream. A TS with only one program/channel.</td>
</tr>
<tr>
<td><strong>STB</strong></td>
<td>Set Top Box. A stand alone receiver used to decode digital signals.</td>
</tr>
<tr>
<td><strong>TS</strong></td>
<td>Transport Stream. PSI and PES packets in 188 byte TS packets.</td>
</tr>
<tr>
<td><strong>TSA</strong></td>
<td>Transport Stream Analyzer. Software used analyze transport streams.</td>
</tr>
<tr>
<td><strong>VLC</strong></td>
<td>Variable Length Coding. A theory where a symbol can be mapped to a variable number of bits.</td>
</tr>
</tbody>
</table>
1 Introduction

This chapter is an introduction to why this master thesis is interesting, relevant and pressing. The chapter was written in cooperation with Anton Allila.

1.1 Background

The development of the television broadcast network in Sweden has been done in several stages. One important quality of the network is that it should be backwards compatible. The backwards compatibility is important to not render older technology worthless and thus force consumers to upgrade. Although, sometimes this approach is not entirely possible. The closing of the analog broadcasting network in Sweden (during the years 2005-2007) is one of these exceptions. This time, the consumer could still use her TV set but had to buy a digital receiver to decode the digital transmissions.

The television broadcast network uses a set of fixed frequencies given by the Swedish Post and Telecom Agency (Post- och Telestyrelsen). These frequencies are inherited from the time when only analog TV were broadcasted and the frequencies set the capacity for bandwidth and the amount of channels in the terrestrial digital video broadcast (DVB-T) network.

The decision to migrate to a digital way of broadcasting came with some benefits, for example, the amount of channels could be increased as the digital technology is more efficient.

Along with the channel distributor Boxer's range of TV-channels, the Swedish Radio's (SR) radio channels P1, P2, P3 and P4 are distributed over the DVB-T network. These channels were first broadcasted without a video stream but SR has made requests to also send still pictures along with their radio shows. The still pictures would be in form of a logotype or a tableau of upcoming programs.

The project initiator of this master thesis is Teracom AB, the company responsible for the TV and radio broadcasting infrastructure.

1.2 Problem Statement

With the demands of more TV-channels and High Definition TV, the need for bandwidth increases. It is therefore essential to make broadcasting more efficient when possible.

Today, video streams with still picture material are broadcasted in the same way as motion picture video streams, i.e. with 25 frames per second (in Sweden). This is because the Moving Picture Experts Group (MPEG) compression standard centers around moving images and is therefore not very well optimized for still picture material. This means that conventional still picture video streams require the same technology and somewhat similar amount of bandwidth as motion picture video.

A more efficient technique of sending still pictures can also be used by channel companies that today share their bandwidth between two channels. I.e. channel A broadcasts in the AM and channel B broadcasts in the PM. When one channel broadcasts the other one does not have the sufficient bandwidth to send a video stream. This situation is represented by Figure 1. The channel which is not sending at the moment still wants an opportunity to market their channel, show a TV tableau or a commercial slideshow.

This is where the need for this master thesis arises, when you need to compress a video stream of only still pictures.
The investigation is pressing since the market of MPEG-4 receivers still is very young but growing fast. It is important to develop a concept stream that receiver manufacturers can test and comply with. It is specified in the NorDig (2009) standard that the Advanced Video Coding (AVC) still picture method should be supported by receivers but as there is no reference stream applying this method, it cannot be tested.

The developed concept stream should be verified to work on digital receivers (which have support for MPEG-4 AVC) currently on the market. The ambition is that receiver manufacturers can use a concept stream to develop support for AVC still pictures in their receivers and DVB operators could use this method of sending still images to create a more bandwidth efficient broadcasting of content.

The questions that will be answered in this report are:

- Is it possible to send still pictures using the AVC still picture method?
- How can this be implemented?
- How bit efficient is this in comparison with the conventional way?
- Is there support for AVC still pictures in the MPEG-4 AVC receivers on the market?

1.3 Aim

As we are two students on this thesis project the literary study, resulting in the theory chapter has been written in cooperation and will therefore be the same in both reports. Also, as the background of the problem is the same and this has also been written in cooperation. We have worked very close together developing the concept stream and all testing has been collaborative.

There is one aim for the complete project, founded by the project initiator Teracom AB and this aim has been divided between Anton Alila and me into two viewpoints, to allow two different master thesis papers.
1.3.1 The Project
The aim of this project is to investigate if it is possible to send still pictures in a bit efficient way in the digital terrestrial network using MPEG-4 AVC (H.264) video compression and the MPEG-2 transport layer. And if it is possible; how the method can be implemented and how it would work in a multiplexed transport stream with other video streams. It is also very important to know how much bandwidth it is possible to save.
To be able to do this, a concept or example stream needs to be developed during the project and this is where most of the practical work will be done.

1.3.2 This Thesis
This thesis report is specialized in comparing the bit efficiency of the AVC still picture video stream to the conventional way of sending still images. It will evaluate the developed concept stream regarding bit efficiency, to know whether or not it is an efficient way of sending still pictures. It will also cover which variables the bandwidth efficiency of the AVC still picture method depend on.

1.3.3 Dividing the Project
Initially, there were no obvious ways to divide the project to fit two master theses. At the end of the development process, it became clear that a natural division would be for one student to evaluate the bitrate efficiency of the AVC still picture video stream and the other student to evaluate the homogenization of the transport stream. This way, both master theses would get a proper technical depth and good scientific content.

1.4 Delimitations
This master thesis is not about, and will not cover, the algorithms for picture and video compression. The necessary modifications of the video stream will be done in syntax only.
The only broadcast standard this thesis will cover is the DVB-T. Although the method might work well with other broadcast standards as well.
The result of the master thesis project is not meant to be a complete and finished product. It is an experiment, a proof of concept and an investigation on the possibilities to develop such a product or service.
The verification test or observation of the concept stream on consumer receivers is not meant to be a scientific experiment. The aim of the verification is to observe and gather knowhow on how receivers react in practice. The verification is called a controlled observation, as not all variables in the test environment could be controlled.
The other paper in this master thesis project has to do with the bandwidth homogeneity, i.e. how to keep the bandwidth from spiking and to keep as constant bandwidth as possible. Thus, this will only be covered very briefly in this paper.

---

1 H.264 is another name for the MPEG-4 AVC standard
2 Knowledge Base

A literary review was conducted to understand the concepts of video compression and broadcasting. To understand the problem completely, we decided to start from the basics. This chapter was written in cooperation with Anton Alila.

2.1 The Human Eye

2.1.1 Acuity in Color Vision

To understand the theories used in the compression process it is important to have a basic understanding of the human eye and the way in which human perception works. The human eye consists of a lens focusing light from the environment onto the back surface of the eye itself. The surface is covered with light sensitive receptors of two kinds; rods and cones. The rods are not sensitive to color, only luminance information is sensed. They are responsible for night vision, motion detection and peripheral vision. The color information is provided by the cones which are divided into three types, one of each senses red, green and blue colors. Studies show that the populations of these types vary greatly. 64% of the cones are red perceiving cones, 32% green perceiving, and 2% blue. The light sensitivity of the rods is more than a thousand times more receptive than the cones. This makes the human visual system much more sensitive to variations in brightness than color (Goldstein, 2009).

These conditions can be used to greatly improve the compression process which is briefly covered in chapter 2.2.2.

2.1.2 Temporal Factors in Vision

The human mind is quite easy to trick. This is something that has been thoroughly investigated by scientists and technological developers alike. The eye communicates through nerve impulses to the brain about 1000 times per second. Although this is not the same rate as to which the rods and cones can perceive changes in stimuli. Instead a slight lag is present. This is the temporal property of the eye called persistence of vision; the retention of the stimuli after it is removed or changed (Whitaker, 2001). This is the phenomenon upon which the whole technology of moving pictures is built.

By showing a series of still pictures which depict some type of movement that can be perceived by the mind as a logical flow of events, the mind quickly begins to interpret the still images as a constant flow of actual movement. For this to be perceived correctly it is necessary to update the picture with a frequency of 25 Hz, or 25 frames per second. Yet there is still another problem. The persistence of vision is reverse proportional to the brightness (intensity) of the image viewed. The stronger the stimulus to the eye is, the shorter the persistence of vision will become. In practice this means that update frequency needs to be improved in order to sustain the illusion of moving pictures, without introducing flickering, when the brightness of the images is increased. This is usually done by showing the same image several times, with a black frame in-between (Whitaker, 2001).
2.2 Analog Television

2.2.1 Interlaced
Interlaced video has its origin in one of the first television inventions. The Scottish inventor John Baird used the technological progress of the 1920s and applied it to the Nipkow disk. By using a disk with 30 holes, the image was recorded successively, hole by hole. Placing the disk in front of a selenium photocell, the signal produced from the photocell showed the recorded image divided over 30 holes (Röjne, 2006). The images were then reproduced at another location using the signal and basically the same equipment in a reverse setup.

Later in the development of television, a cathode ray tube canon was used to draw the pictures using horizontal lines over the TV monitor. The canon was not fast enough to draw each line without the first lines fading out; the eye noticed this. Using the eyes properties in persistence of vision, this problem was overcome by using interlaced scanning.

Interlaced scanning work by capturing every other line from left to right by the camera and shown on the monitor or TV. First all odd lines are drawn and then the even ones fills the space between. Each picture with either the even or the odd lines is called a field. When two fields are shown in rapid succession, they are integrated by the eye and called a frame. One frame represents all lines in two succeeding fields.

In Sweden, and most of the world, the line voltage uses a frequency of 50 Hz. Using this frequency as a good reference to keep a constant speed, a new field is scanned by the camera or shown by the TV 50 times per second. But as two fields form one frame, the video runs at 25 frames per second (fps). In North America, 60 Hz is used in the line voltage and thus the video runs at (about) 30 fps (Ascher, Pincus, 1999).

The North American system was the National Television System Committee (NTSC), with 525 lines per frame and the format used in Sweden (and many other European countries) were Phase Alternating Line (PAL). PAL, as well as "Séquentiel couleur à mémoire" (SECAM) uses 625 lines per frame (Röjne, 2006).

2.2.2 Color Signal
When TV first made its entrance on the consumer market in Sweden, in the 1950s, there was no color signal at all. The signal was black and white. Every grey tone in the picture corresponded to a voltage level in the signal; this level was and is called luminance. When the luminance signal is +1 volts, the luminance is 100%, i.e. a white picture (Röjne, 2006).

As described in chapter 2.1.1, the eye has three color receptors, red, green and blue. The additive color model, Red Green Blue (RGB), which is used in most professional cameras, mimics the receptors as the camera uses a prism to divide the light onto 3 separate Charge-coupled device (CCD) sensors, one for red, one for green and one for blue.

When the color TV made its appearance it used three cathode ray tubes, red, green and blue. The two main reasons a signal with these three colors could not be broadcasted were the backward compatibility issue; black and white TV sets would not understand the new signal. The other reason is the problem of broadcasting; the three signals would consume three times the amount of bandwidth compared to the black and white system (Röjne, 2006).
A solution was developed from the knowledge of human perception. Part of which is described in chapter 2.1.1. Using the additive color model; the luminance, Y, can be defined by:

\[ Y = 0.30R + 0.59G + 0.11B \]
\[ R - Y = 0.70R - 0.59G - 0.11B \]
\[ B - Y = -0.30R - 0.59G + 0.89B \]

This is just another way of describing the RGB color space and R-Y and B-Y are color difference signals, Chrominance. This way of encoding RGB is called YPbPr.

We already learned that the eye is less sensitive to variations in color; this means we can use less bandwidth for the chrominance information; we achieve an analog image compression. The luminance information is filtered down and the chrominance information placed above the luminance in frequency. In the PAL-standard, which is used in Sweden, the luminance uses 5MHz bandwidth and the chrominance uses 2 x 1 MHz (B-Y, R-Y), this means that the color in analog color TV uses only 40% of the information compared to the luminance (Röjne, 2006).

### 2.3 Digitizing TV

#### 2.3.1 Digitizing

The analog signal transmits a lot of redundancy and unimportant information. By representing the signal digitally, bandwidth requirement and issues regarding noise can be reduced.

Going from the analog to the digital world, we start by taking small samples from the analog waveform, making it discrete. The digital world is built around the binary system, i.e. a bit is either 1 (one) or 0 (zero) and a sequence of 8 bits constitutes a byte.

The digitizing consists of two phases. First we make the continuously analog signal discrete. This is done by reading the value of the analog signal at a constant frequency. This is called sampling, and will disrupt the analog signal's time dimension continuity.

To represent the analog signal correctly, the samples must be made at high enough frequency. This frequency can be calculated using the Nyquist sampling theorem which in its basic states that the signal should be sampled at a frequency of at least twice the signal bandwidth. Sampling with a lower frequency will result in an incorrect representation of the signal, a problem called aliasing. For example, to correctly describe a picture bandwidth of 5 MHz, digitally, we would need to sample at a frequency of 10 MHz, i.e. 10 000 000 samples per second.

We also need to round the samples to predetermined values, this process is called quantization. In this stage we lose the continuous signal values and approximate them to fit the nearest of our fixed values. An everyday example of this is a ladder, which steps quantize the height. The number of predetermined steps usually used in video is 256, which is the same as 8 bits. The differences between the measured analog values and the discrete digital steps are called quantization errors (Watkinson, 2004).

Using the example values above, the digitizing of one black and white PAL TV-channel without sound would need the transmit speed of 10 000 000 bit x 8 levels = 80 Mbit/s. The total bandwidth of the DVB-T network in Sweden today is about 110 Mbit/s. Using more than 70% of the entire bandwidth for only one channel is not an option; we need to compress the data. This is covered in chapter 2.5, 2.6 and 2.7.
2.3.2 Benefits

The benefits of digitizing TV is mainly the ability to implement powerful compression algorithms and thus reduced bandwidth needed for each channel. This in turn, leads to room more channels to be broadcasted and the possibility to send other data to be used by the receiving set top box.

Digitizing also comes with a better picture quality as the noise is not contaminating the signal in the same way. The digital signal is also more robust and can handle interference in a better way compared to the analog signal. The digital signal needs less signal strength (Teracom, n.d) and thus the transmitter can use less power which is good, both from an environmental and an economic point of view.

2.4 Broadcasting

2.4.1 Digital Video Broadcast

Digital Video Broadcast (DVB) is a group formed in 1993 with aim to create a standard for transmitting digital video. Their standards are published by “The European Telecommunications Standards Institute” (ETSI) and most parts of the world has agreed to use different kinds of DVB to broadcast digital video. The standards are known as DVB and a suffix, depending on what the standard covers.

The standards of DVB differ depending on which medium the broadcasts move through. The reason for this is that different mediums have different demands or need different error protection. For example; a transmission from satellite needs a robust but not very efficient modulation due to a noisier channel. The Satellite DVB standard can use a less efficient modulation because of the satellite’s higher bandwidth.

Some standards from DVB does not concern transmitting, but focuses around subtitling, service information or conditional access (Röjne, 2006).

Amongst many other things important for this master thesis, the DVB-T standard reads that “in the case of still pictures the fixed_frame_rate_flag shall be equal to 0” (DVB standard, 2007). This allows bypassing the requirement of 25 frames per second.

2.4.2 Bit Errors

A broadcasted signal is always in risk of being exposed to noise of different kinds. Therefore we need to protect the signal against both static noise and noise bursts.

Noise is unwanted, random interference with the signal. Static noise usually causes bit errors scattered over a major part of the signal. Noise-, or error bursts are a bit different. These can be caused by thunder, voltage spikes or electronic equipment and will wipe out a series of subsequent bits.

When the decoder receives the damaged signal it needs to be fixed, otherwise the decoder will not be able to understand it. Unlike other distribution forms, the DVB does not have a return channel, so the receiver cannot ask the sender to resend the package. Luckily some precautions are made before sending the signal to make sure the decoder has a good chance of correcting the errors that might arise during the transmission, these precautions are called Forward Error Correction (FEC). Some of the FEC used in DVB are Reed-Solomon error correction, interleaving and punctured coding.

The amount of bit errors in relation to non erroneous bits is called Bit Error Ratio (BER) and is strived to be as low as the threshold $1 \times 10^{-11}$ which is called Quasi Error Free (QEF). When the BER is below the QEF threshold, the consumer experience of the broadcasted service is not tainted. The BER of $1 \times 10^{-11}$ is one bit error every 20 000 seconds (Röjne, 2006).
2.4.3 Multiplexing

Multiplexing is a way to transmit several signals, or services, over one medium. The medium could be a cable or in the case of DVB-T, a radio frequency. In theory, multiplexing uses the capacity of the low level channel to create many high level logical channels. What it does in practice is mixing the packets from, in this case, different TV channels in time over the transport medium.

Multiplexing over time is called Time Division Multiplexing. The physical transmission channel is chopped up into time slots. During one time slot, only one sender can use the channel. The other senders have to wait for their turn.

The multiplexers used by Teracom in Sweden are statistical multiplexers. These are dynamic and can communicate and adjust the MPEG encoders to fit each channel's need at a specific moment. Suppose that Channel A and B are residing in the same Time Division Multiplex and shows different TV programs. If the program on Channel A is hard to code, i.e. needs more bandwidth, and the program on Channel B needs less, the multiplexer will inform the encoders of this. Channel A will then be able to use more time slots per time unit. I.e. the bandwidth of Channel A increases and Channel B decreases. A representation of the bandwidth in a Time Division Multiplex is shown in Figure 2. More about how this affects this thesis in chapter 2.8.5.

The statistical multiplexers make sure to keep the output transport stream bandwidth fixed at, for example, 22 Mbit/s.

![Figure 2: Three channels sharing a statistical multiplex](image)

2.4.4 Receiver

The receiver hardware and software at the consumer end has to understand all the transmission modes and compression methods used. Most new TV's today has built-in DVB decoders, some for DVB-T, some for Cable and some for all three; Terrestrial (-T), Cable (-C) and Satellite (-S). The TV's with built in receiver is called Integrated Digital Television (IDTV). If the TV doesn't have an integrated DVB decoder a set top box has to be used to decode the signal.

The standard of DVB has a lot of alternatives, for example; the structure of the broadcast networks in different countries affects the radio performance. Also the encryption system for pay TV differs. This leads to different DVB decoders for different regions of the world, or even countries.

NorDig is an interest organization who has set up some ground specifications and minimal requirements for DVB decoders on (primarily) the Nordic market. There are more of these interest organizations, but not as many as there are countries using the DVB standard (Hyvärinen, Teracom course, 2009).
The receiver basically consists of the inverse of all the pieces in the encoding and transmitting equipment. Its job is to take the incoming radio signal, demodulate it, repair potential errors, decode the transmitted MPEG-2 transport stream and show the images on the TV screen.

The decoder does not have to be as intelligent as the encoder. This feature is implemented and planned from the start of DVB with the choosing of algorithms that are optimized as being easy to decode. The calculations and the effectiveness of the video stream and signal are decided by the encoder and this is the reason for the enormous price difference between encoders and decoders.

### 2.5 Data Compression

Data compression, just like compression in the physical sense, is all about figuring out ways to fit something into a smaller container than before. In data compression, it is not done by brute force, as in the physical world; instead one substitutes the way the data is represented. In the real world, the receiver of a compressed physical medium can decompress, without knowing or understanding what algorithm or technique was used to compress. This is not true for data compression. Both parties must know what algorithm was used during the data compression process in order to understand how to decompress the data. This can easily be illustrated with languages. If you don't understand the written language the spoken words have been "compressed" into; you cannot decompress them back into words. The same is true for all data compression.

There are several different approaches to compressing data, mainly divided into two groups; lossy and lossless compression.

Lossless compression does not alter the compressed data. This means that to the exact bit, one can recreate the compressed data back to its original form when decompressing it. This is true for all .zip\(^2\)-like formats, in the sense that when you zip a word document, and then unzip it, the text inside is intact. You are not missing letters and words that have been compromised in the compression process.

Lossy compression does not bother trying to recreate the data compressed to the exact bit. Instead the aim is to compress the data in such a way, that the receiver cannot tell it apart from the original. This is done, for example, by using all of the limitations of the human visual system that we described earlier.

A plain example of this is the following. We are trying to compress the number

\[
56.77777777
\]

A lossless approach would be to utilize the redundancy in the trailing sevens with an algorithm or syntax that takes up less space. One such algorithm called Run Length Encoding gives the result

\[
56.[8]7
\]

We have introduced a syntax that has to be understood by the receiver in order to decompress the number back into decimal form. Where \([n]x\) means that upon this sign are \(n\) instances of trailing \(x\).

The lossy compression approach would simply be

\[
57
\]

---

\(^2\) Read more about the zip format at: http://en.wikipedia.org/wiki/ZIP_(file_format)
Now, the observant reader would of course doubt the previous statement that the receiver or decompressor of this lossy coding doesn’t really have to know the algorithm used in order to understand this data. This might be true for this example, but if the receiver is awaiting a number with 8 decimals and instead picks up an integer, he is still in need of the algorithm in order to de-compress the data.

2.5.1 Variable Length Coding

The basic lossless compression technology is possible because of statistical redundancies in the data. For example, when compressing text, one could use a shorter bit-representation of characters that are used more often than others and in such a way get a smaller file-size (Watkinson, 2004). This is called Variable Length Coding (VLC). An example of VLC would be to represent a picture with 4 different colors, black, white and two grayscales. Using the system in Table 1, each color will get an equal length bit code.

<table>
<thead>
<tr>
<th>Color</th>
<th>Bit code</th>
</tr>
</thead>
<tbody>
<tr>
<td>White</td>
<td>00</td>
</tr>
<tr>
<td>Light grey</td>
<td>01</td>
</tr>
<tr>
<td>Dark grey</td>
<td>10</td>
</tr>
<tr>
<td>Black</td>
<td>11</td>
</tr>
</tbody>
</table>

*Table 1: Equal length bit-code representation of 4 color image.*

Without knowing the statistical redundancies in the image, this is the obvious way to encode the image. The bit per pixel quota in this equal length bit-code will always be two. If we know the statistical redundancies of the image we can introduce a variable length coding pattern and lower the quota. For example, if we know images compressed with this pattern have the occurrence percentages shown in Table 2 we can construct the bit-code system in Table 3.

<table>
<thead>
<tr>
<th>Color</th>
<th>Occurrence percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>White</td>
<td>35%</td>
</tr>
<tr>
<td>Light grey</td>
<td>20%</td>
</tr>
<tr>
<td>Dark grey</td>
<td>5%</td>
</tr>
<tr>
<td>Black</td>
<td>40%</td>
</tr>
</tbody>
</table>

*Table 2: Statistics of pixel colors in example image.*

<table>
<thead>
<tr>
<th>Color</th>
<th>VLC bit code</th>
</tr>
</thead>
<tbody>
<tr>
<td>White</td>
<td>10</td>
</tr>
<tr>
<td>Light grey</td>
<td>110</td>
</tr>
<tr>
<td>Dark grey</td>
<td>111</td>
</tr>
<tr>
<td>Black</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 3: Variable length coding pattern.*
Using the VLC bit code in Table 3 will, unlike in Table 1, give us a bit per pixel quota of:

\[ 1\times0.40 + 2\times0.35 + 3\times0.20 + 3\times0.05 = 1.85 \]

The problem of variable length coding is that since we don't know how many bits that represent a pixel, we need another way of distinguishing when the color code of one pixel ends and where another one begins. This has been done in the above example by always ending the bit code of each pixel with 0. This way the decompression algorithm can read until the next 0 value, or a maximum of 3 bits, and then match the code to the pattern and know what color it represents.

Huffman coding is a well known variable length coding scheme published in 1952. The length of each bit-code in the coding pattern, used to encode the input symbol sequence, is proportional to the probability of the symbol in the sequence (Watkinson, 2004). This is done by inserting the probabilities into a frequency-sorted binary tree. This will give the most common symbols the shortest bit-codes. The length of the bit-code can be roughly calculated through the negative logarithm of the probability for the symbol encoded. The frequency-sorted binary tree for the previous example is shown in Figure 3.

![Figure 3: Frequency-sorted binary tree for Huffman encoding. White on black backdrop is the probability and black numbers on white backdrop is the bit sequence.](image)

### 2.6 Image Compression

There are several different approaches to image encoding. The most common and most successful from a compression point of view, for encoding photos, is the Joint Photographic Experts Group (JPEG) compression algorithm. This is the only image compression algorithm that will be covered in this master's thesis as the same algorithms are used when coding inter coded video frames in MPEG-2 and -4.

#### 2.6.1 JPEG Compression

Although there are lossless versions of the JPEG compression algorithm, most of the strongpoint's of the algorithm are based on the ideas of lossy compression. The JPEG algorithm is used in image compression, and in essence also in video compression. Several steps and mathematical systems are used in order to achieve a high compression rate. First and foremost the image is split into 8x8 pixel macroblocks. These blocks are then processed independently.

In order to understand the transformation algorithms used in JPEG compression we need to understand the basics features of the frequency domain. Any single waveform in a one dimensional spectrum, with a finite number of discreet points, can be represented by a weighted sum of cosine functions oscillating at different frequencies and amplitudes. This is called a discreet cosine transformation (DCT). This conversion is a spectral transformation, where you transform your waveform representation from the time domain to the frequency domain.
In the same way, each and every combination of shapes and patterns in a two dimensional 8x8 macroblock can be represented as a weighted sum of 64 sub “images”. This is essentially converting the image into a set of weights, which describes how much of the original macroblock can be represented by each of the sub images. These sub images are a set pattern of the cosine waves oscillating with increasing horizontal and vertical spatial frequencies. This pattern is shown in Figure 4.

![Figure 4: The two-dimensional DCT wave table.](image)

Entering the spatial frequency domain will grant independent control over each and every frequency in the macroblock. This has two main advantages. Since the human eye is more tolerant to noise in the high frequencies, there is more room for lossy compression in this area. As the frequency gradually rise, the amplitude usually decline. With lower amplitude, fewer bits are needed to represent the value. Entering the frequency domain is done mathematically by transforming the image using a two dimensional DCT transformation matrix. The result is an 8x8 coefficient matrix, previously referred to as a set of weights, that holds the information as to how much of the image that is represented by each and every one of the sub “images”. This process is shown in Figure 5.

![Figure 5: Pixel values converted into DCT coefficients.](image)

The top left coefficient is called the DC value, which holds the mean brightness of the macroblock. The DCT transformation will concentrate the majority of the information to the upper left of the coefficient matrix. As the horizontal and vertical frequencies rise, moving towards the bottom right corner, the amplitudes of these cosine patterns decline.
The next step is the quantization process. Since, again, the human eye is less sensitive to high frequency brightness variations and more tolerant to noise in the high frequencies, the quantization matrix will cut these frequencies with a higher denominator value as shown in Equation 1. Equation 1 also contain a strength multiple; \( S \) which control the image size.

\[
F'(u,v) = \frac{1}{S} \frac{F(u,v)}{Q(u,v)}
\]

Finally the quantized DCT coefficient matrix is read using a zig zag pattern. This groups similar frequencies together and the objective is to get to as many consecutive zeros as possible in order to use Run Length Coding to represent them (Watkinson, 2004).

The variable length coding used for the final bit-sequences is usually Huffman. The Huffman encoding table used can either be calculated for the frequency distribution of a specific image, or picked from the JPEG standard general-purpose Huffman tables. Before and after compression pictures are shown in Figure 6.

![Figure 6: The original picture to the left and compressed with 23:1 to the right.](image)

### 2.7 Video Compression

The typical video compression algorithm exploits the temporal redundancies that generally exist in between moving picture frames. This is done by representing a frame in a moving picture sequence as the difference between the previous and present frame. Figure 7 shows an illustrated example of how such a residual frame is calculated.

The sky in the background is redundant in the images and gives zero residual data. A slight camera movement changes the position of the hydrostatic wind gage. The position of the rotating blades is also changed. By using this approach there is usually a 3:1 ratio compression in picture size, between full picture and the residual picture (Röjne, 2006), although this is largely dependent on the input video material, the encoder used and the compression algorithm used by the encoder.

![Figure 7: Frame N is subtracted from N+1 giving the residual frame D.](image)
To reconstruct the picture the residual data is added to the previous picture, this is shown in Figure 8.

![Figure 8: The reconstructed frame N+1 is calculated from N+D](image)

The problem with this approach arises when you cut from one scene to the next. When this happens, it is no longer viable from a compression viewpoint to keep on with the residual frame encoding approach. To solve this you present another full frame, from which you can calculate the succeeding residual pictures. In MPEG-4 (chapter 2.7.2), this frame is called an Instantaneous Decoder Refresh (IDR) frame which clears the content of the reference picture buffer and instantly decodes the IDR picture (Richardson, 2003).

### 2.7.1 MPEG-2

In MPEG-2, moving picture encoding, there are three types of frames. The full frame, which is encoded without reference to another picture in the sequence, is called an I-frame (Intra-coded picture). The second type is the P-frame (Predicted picture or Inter frame prediction) which were described earlier as the residual picture with reference to another I-frame. The third type is the B-frame (Bi-Predicted, Inter frame picture) that is calculated from both preceding and succeeding P- and I-frames. The expression intra-coded picture refers to the fact that the coding is done using only information from the current frame. Inter-coded pictures refer frames other than the current frame (Röjne, 2006).

One full sequence, spanning from one I-frame to the next, is called a Group Of Pictures (GOP) (Röjne, 2006). A normal size of GOP used is 15. If the frame rate is 25, this would lead to almost two GOP’s every second. The GOP size of a MPEG-2 encoded sequence is described with N, representing the number of frames in the sequence and M, representing the frequency of the P-frames. A typical GOP shown in Figure 9 has the GOP size of N=9 and every third frame is a P-frame; P=3.

![Figure 9: A GOP structure for MPEG-2 video.](image)

As we can see, the I-frame is the reference for the first B- and P-frames. The B-frames are also referenced in reverse time from the P-frame, hence bi-predicted frame.

The next feature of the MPEG-2 standard is the motion compensation process. To even further compress an image, such as the previous example of the hydrostatic wind gage, the encoder estimates the movement of objects in the picture. This is done by searching for each macro block in the N+1 frame, for a match in the reference picture. In Figure 10 we are searching for the highlighted block in the N+1 frame. This block can be found
displaced in the previous image $N$, highlighted in red. Now we can represent this entire macro block as the vector describing the displacement of this block in the reference picture. This is shown in Figure 11.

![Figure 10: The object in the highlighted macro block in frame $N+1$ can be found displaced in the previous frame $N$ highlighted in red](image1)

The exact algorithm for this process is not standardized. It’s up to the encoder manufacturer to figure out how this search is done in practice. Only the method for decoding the video data is specified, and this process is indifferent to how the matching motion estimation was done during encoding.

![Figure 11: The displacement is described by the motion vector in blue.](image2)

The actual encoding process in MPEG compression is a bit different from the order that has been described earlier. First, all motion compensation is done using the previous image, trying to match the current one. Then the motion compensated, predicted, image is compared to the actual image, creating a prediction error, also called a residual frame. This prediction error together with the motion compensation vectors is now sufficient information to describe the entire frame. An example of the benefits of using this approach is shown in Figure 12.
During encoding, each frame is given both a presentation timestamp and a decoding timestamp. This is because the order in which frames are transmitted and decoded is different from the order in which they are displayed. Bi-predicted frames need all reference pictures to be present in memory before they are decoded since B-frames reference both preceding and succeeding frames. An example of this is shown in Figure 13.

The aim is always to reach a high compression ratio. Figure 14 displays an example of the difference in data amount between I-, P- and B-frames. Note that this example is for moving video.
2.7.2 MPEG-4 AVC

The MPEG-4 AVC compression algorithm improves the compression even further. The main differences from MPEG-2 compression is that the macroblocks are encoded in fields of 4x4 instead of the previous 8x8 for JPEG and 16x16 for MPEG-2. The new 4x4 transform matrix is fully reversible. This means that an image encoded without quantization can be restored to its original form, without rounding errors. This makes the algorithm lossless in its core. Furthermore the motion compensation can be done with anywhere between 4x4 and 16x16 macroblocks, including combinations like 8x4, 8x16 and so forth. An individual frame can reference up to 16 previously encoded frames, in comparison to one or two in MPEG-2. This makes the Hierarchical GOP structure in Figure 15 possible. Dynamic GOP length is also introduced in MPEG-4 (Watkinson, 2004).

![Hierarchical GOP](image)

*Figure 15: Example of Hierarchical GOP in a MPEG-4.*

The previous Variable Length Coding is replaced with Context-adaptive Variable-Length Coding (CAVLC). This makes it possible to change the VLC in accordance with the context of the pictures. A more complex version of this is the Context-adaptive binary arithmetic coding (CABAC) (Watkinson, 2004).

Recurrent information is sent using a Network Abstraction Layer (NAL) which holds all the information beyond video coding data. This is covered further in chapter 2.8.2.

2.7.2.1 MPEG-4 Profiles and Levels

The MPEG-4 encoding standard has distinct profiles which define how video is encoded and which tools and syntax may be used. These profiles target different classes of application, ranging from Constrained Baseline Profile for mobile and video conferencing to High 4:4:4 Intra Profile with only intra frame coding. To define the video further different levels are introduced, specifying the maximum macroblocks per second, per frame and also maximum video bit rate. The profile and level is conventionally written as profile@level.
2.7.2.2 AV Two Still Pictures

The MPEG-4 AVC standard supports still pictures in the video stream. How this is used in practice is one of the aims of this master thesis and is covered in chapter 4. The definition of an AVC still picture from H.222\textsuperscript{3} is:

An AVC still picture consists of an AVC access unit containing an IDR picture, preceded by SPS and PPS NAL units that carry sufficient information to correctly decode the IDR picture. Preceding an AVC still picture, there shall be another AVC still picture or an End of Sequence NAL unit terminating a preceding coded video sequence unless the AVC still picture is the very first access unit in the video stream. (MPEG-2 standard, 2006)

According to the NorDig (2009) Unified Requirements standard, the receiver should, when accepting an AVC still picture stream, ignore the potential buffer underrun and display the still image until another still image or another video stream is decoded. The use of AVC still pictures should be signaled in an AVC Video Descriptor in the Program Map Table.

2.8 Packetizing the Compressed Video

The audio, video, and data in the DVB-T system is carried, whether it is a MPEG-2 encoded or a MPEG-4 encoded video, using a MPEG-2 transport stream. The compressed audio- and video stream is to be sent over a low level medium and therefore we need to packetize it to be able to parse the bit stream information.

When receiving a bit stream, we need to be able to understand when one packet ends and another starts and also find the synchronizing patterns. Luckily, as the DVB standards use the Reed-Solomon FEC, it is already parsed due to the error correction algorithm, so what we get out from the Reed-Solomon decoder is a byte stream where the word boundaries are already known (Watkinson, 2004).

2.8.1 The Elementary Stream

The video elementary stream (ES) is an endless bit stream of raw data representing encoded video frames in the decoding order. The ES is built up from sequences and only contains one type of data, e.g., video, audio, or data. Each sequence also has information about the video height, width, picture format, frame rate, and data rate (Röjne, 2006).

2.8.2 The Network Abstraction Layer

The Network abstraction layer (NAL) is something which is new in MPEG-4. In MPEG-4 AVC, the video coding layer, where all the coding is done, is separate from the transport layer and NAL is a step between the two. The ES are mapped to NAL units before transmission or storage (Richardson, 2003). The coded video sequence is thus represented by a series of NAL units. Each NAL unit packet has a header with information about what kind of raw byte sequence payload (RBSP) it carries. The RBSP is the type of information in the NAL unit.

---

\textsuperscript{3}H.222 is the standard document for MPEG-2 transmission and multiplexing.
The types of available RBSP types can be seen in Appendix A (MPEG-4 standard, 2008). The most important ones for this master thesis, except for the coded slices, are:

- Sequence parameter set (SPS), where parameters for whole video sequences are kept, such as limits on frame numbers, picture order count and whether field (interlaced) coding, or frame coding was used.
- The picture parameter set (PPS) is more or less the same as the SPS, but applies to one or more pictures inside a SPS. The PPS has, amongst other, information on what kind of entropy coding is used, the number of slice group used and a list, and number, of reference pictures.
- Supplemental Enhancement Information (SEI) messages are not essential for encoding the video sequence but can contain information on buffering time, picture timing and the deblocking filter properties.
- The end of sequence (EOS) unit indicates the end of a video sequence and that the next picture in the decoding order is an IDR picture.

The parameter sets are a way for the encoder to signal ahead for important changes in the coding, as for example, slice coding type. (MPEG-4 standard, 2008)

The PPS is activated by a referral from a coded slice header; this PPS stays active until another one is called from another slice header. A SPS is activated by a call from the PPS header in the same manner (MPEG-4 standard, 2008).

The coded slice data partition units consist of three different forms, A, B and C. Partition A holds the headers for all the macro blocks in the slice. Partition B contain intra coded slice data, and C contain inter coded slice data. NAL unit type number five is the coded slice of an IDR picture (MPEG-4 standard, 2008).

### 2.8.3 The Packetized Elementary Stream

The elementary streams, in MPEG-2, or the NAL units, in MPEG-4 AVC, are packetized for sending over a medium. Both transmission and storage prefer discrete blocks of data (Watkinson, 2004), these packages are called PES packets, for packetized elementary stream, and can only contain a video ES, an audio ES or a data ES.

The size of the PES packets vary but one video PES normally contains one whole picture. An audio PES would normally contain about 24 ms of sound (Röjne, 2006). The PES packet has a header and a payload as described in Figure 16.

```
<table>
<thead>
<tr>
<th>Start code</th>
<th>Stream id</th>
<th>PES packet length</th>
<th>PAYLOAD DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>24 bit</td>
<td>8 bit</td>
<td>16 bit</td>
<td></td>
</tr>
</tbody>
</table>
```

*Figure 16: The PES packet mandatory header information*

The PES packet headers often contain time stamps for synchronization, decoding and presentation purposes. The PES header for a video ES can also contain information about various trick modes and their properties.

The whole PES has to be received and put in the decoding buffer for the receiver to be able to start decoding it.
2.8.4 The MPEG-2 Transport Stream

The transport stream (TS) is the last step in packetizing and is what gets passed on to the modulator for transmission over the medium.

A transport stream is audio, video and data PES packets multiplexed onto a stream constructed for transmission purposes. A transport stream with multiple programs or services is called a Multiple Program Transport Stream (MPTS) (shown in Figure 17) and a transport stream with one, and only one program or service is called a Single Program Transport Stream (SPTS). Both of these have a fixed packet length and are constructed robust, for transmission.

![Figure 17: A MPEG-2 MPTS is created from an elementary stream](image)

The length of a transport stream packet is always 188 bytes to facilitate multiplexing and error correction, but the payload data can vary as the packet header has optional fields (Watkinson, 2004).

The packet header contains a lot of information needed to demultiplex and decode the stream. Appendix B shows the layout of the transport stream header. The PID, which is the packet identifier, is one of the information chunks in the transport stream packet header, as is the continuity counter. The continuity counter exists to make sure all packets are received and received in the correct order.

The PID is a unique 13 bit code which is static for each video, audio or data elementary stream (Watkinson, 2004). Some PID values in a transport stream are predetermined as the receiving demultiplexer has to know where to start looking for a program. This procedure is represented in Figure 18. The Program Specific Information (PSI) helps with that by using a Program Association Table (PAT), a Program Map Table (PMT) and also a Network Information Table (NIT) (Watkinson, 2004). The PSI is an umbrella term for transport stream essential information packets, like PAT, PMT and NIT.

The PAT is always at PID 0 and this is the only thing the decoder knows when it is powering up. In the PAT, there is a list of all the programs and PSI the transport stream carries and a PID “link” (as seen in Figure 18) to each program’s PMT, this link is called a program number. The PMT contains the information about at which PID the program video, audio and data streams can be found (Watkinson, 2004).

The audio- or video streams in the PMT can contain different descriptors which are used to store standard- or user defined data that describe the stream. One of these descriptors important for this master thesis is the AVC Video Descriptor.
2.8.5 Bandwidth Use in a Multiplexed Transport Stream

As we talked about in chapter 2.4.3, to fit more than one program or service on a transport stream, we need to multiplex them.

The transport stream bandwidth is always constant but the bandwidth of the video or audio PIDs can vary depending on how advanced or difficult to compress the input material is. To fill up the constant bandwidth of the transport stream, null packets are used. Null packets are packets with the PID number 8191. These packets are discarded by the receiver (or potential next multiplex).
A basic and low level representation of a multiplexed transport stream would be to use the number 1 for a packet with picture information and 0 for a null packet. This example leaves out all audio and other PID necessary but the principle is the same. In this example, a single video PES is multiplexed on to a transport stream. There is 20 subsequent transport stream time slots to dispose for a single I-, P- or B picture. For an easily encoded I picture, or a P or B picture, the transport stream would look like this:

11100000000000000000.

Only a small amount of the average bandwidth of the transport stream is used. For a picture harder to encode the transport stream would look like this:

11111111111111111110.

Almost all of the total bandwidth is used. The transport stream bandwidth is constant while the actual picture/video bandwidth is varying. In a transport stream with multiple PID’s, how often a packet with a specific PID occurs in the transport stream decides the bitrate of that PID. If 70% of the packages in a transport stream with bitrate 10 Mbit/s are video packages, the average video bitrate would be 7 Mbit/s.

The important thing to understand here is that the two examples above have the same peak bandwidth use. The peak bandwidth is what matters as too much varying bandwidth of the individual video stream leads to extensive load on statistical multiplexers, with loss of efficiency as a result. For statistical multiplexers to be efficient, the video stream bitrate variation must be kept at a minimum to prevent loss of picture quality in the other streams.

### 2.8.6 Picture Timing

In order to synchronize the playback rate with the broadcast rate, a Program Clock Reference (PCR) timestamp is embedded in the Transport Stream packets. This timestamp helps to keep the receiver System Time Clock in sync with the transmitting one. This is needed to ensure that a picture is displayed at the correct time in the receiver's display unit. If these clocks get out of sync, either the pictures are shown too fast and the picture buffer will underflow, or they are shown too slow and the buffer will overflow, both causing interference with the playback.

In order to control the display time of broadcasted pictures, every picture has a Presentation Timestamp (PTS). The receiver looks to this value in order to decide at what time the picture shall be displayed. When the System Time Clock, controlled by the PCR timestamps, is equal to a specific pictures PTS, the picture is displayed. Because the complexity of the picture decoding process varies between pictures, it is also possible to set a Decoding Timestamp (DTS), which controls when a picture shall be decoded. This is also used to force pictures, needed for referencing, into the buffer ahead of their presentation time, in order to decode pictures referencing to them. The DTS is needed when bi-directional encoded pictures are used.
3 Method

This chapter describes the project method, the iterative development model used and some thoughts on why this method was chosen. The reliability and validity of this development model is also discussed.

3.1 Iterative Development

To not risk finishing the project with a half way done product, a form of iterative development model was used both for the master thesis and for the project. The term iterative development model can be considered an umbrella term for a number of project models. An example of one of these project models is the Scrum development model, used since the early 1990s (Schwaber, K., Sutherland, J., 2009).

To be able to go back, do a lot of testing and make changes depended on what was learned, was crucial in this project as we had no idea on what was going to be the right approach or where the project would take us. This is the main reason an iterative development model was chosen for this project.

The iterative development model has the benefit of being able to go from an idea to implementing and testing that idea in a quick and easy way. It also has the advantage of developing the software, or in this case the concept stream, incrementally. It was evident early on that a lot of prototyping would need to be done.

According to the iterative development model, a minimum of the software requirements should be implemented at first and with each iteration, more features are added. If the implementation of or analyzing a problem is too difficult, a step back is taken and the problem is re-thought and re-designed. Figure 20 describes the basic layout of an iterative development model.

![Diagram of iterative development model]

*Figure 20: A representation of the iterative development model with the initial planning phase, the iterative cycle and the deployment phase*
3.2 The Iterative Development Model in Practice

Most of the knowledge acquired during the literary study was very useful and essential to understand what needed to be done. Unfortunately, almost no literature apart from what was written in the standards was found on the subject of AVC still pictures. This literature was only on a more general level which meant a lot of the knowledge needed to be acquired during the development process.

Beginning to work with this thesis problem in practice, the depth of knowledge needed was unexpected. This meant going back and revisiting the theory stage of the process and acquiring new knowledge had to be done from time to time. Also, almost all the work was done with the standard documents at hand.

The iterative development model in Figure 20 is a universal figure. For this master thesis; the initial planning stage was the 4 week extensive literary study where video encoding and broadcasting were studied. The literary study also included a two day Digital TV systems course. The knowledge obtained during this literary study was essential to come up with theories and to understand the problem fully.

The iterative cycle of the iterative development model corresponds to where the practical work was done. The iterations consisted of an original stream being altered with a hex editor and java programs (these are covered in chapter 4.3.2 and 4.3.3) to create a prototype stream. This prototype stream was then tested using analyzing software and a professional MPEG decoder. The practical work and the implementations in the iterations are covered in chapter 4.

The deployment stage in this master thesis project is when the concept stream has been tested on the consumer receivers. The evaluation of the project is then done and the reports for the project initiator and the master thesis are written.

3.3 Goals

It was completely clear, going in to the project, that, even if only sending one frame per second, the resulting bandwidth use would not be even near 1/25 of the original bandwidth. If the video encoding used didn’t use the inter frame coding, i.e. referring to other frames instead of intra coding them again, the resulting bandwidth could be down to 1/25 of the original bandwidth use. But as almost all video encoding uses inter predictive encoding, we cannot hope to save that much bandwidth.

A wish from the thesis initiator, Teracom, is for a successful prototype stream to use less than 100 Kbit/s of bandwidth, still with an adequate image quality.

3.4 Reliability and Validity

It can be difficult to ensure a high validity and reliability in development projects. We have taken the preventative measures we hope were necessary to keep as high validity and reliability as possible when choosing the development method. By choosing an iterative development model and doing an extensive literary study before starting the development, we hope to reach both a high validity and reliability.

The model used in this project will ensure a high validity mainly as we have put a good deal of research behind the development theories and set out to understand both the specific problem and the background of the problem.

The validity is further ensured by keeping the developed product inside the roams of standards. This is illustrated by the Venn diagram in Figure 21. The standards are kept close at hand at all times and every change in syntax is validated to comply with the standard documents. The choice of modifying an existing stream, instead of building a new
one, adds to the validity as well, as we then can eliminate some sources of errors. More on this choice in chapter 4.1.

I would say that the fact that there are two of us developing the stream favors the validity. Also, the possibility of observing how receivers react to the stream also ensures that we are developing what we actually mean to develop.

The reliability is a little bit trickier. Everybody interprets stuff differently and someone else may have entered the development project with a different view of things and with other expectations on what is possible. This could lead to a different development phase. But as the standard has set up what the results should comply with, the end result, the concept stream, should therefore be the same if another study or development was done independent of the one described here. Therefore, the Venn diagram in Figure 21 can be used to describe both the validity and the reliability of the result.

Figure 21: To ensure the result has a high validity and reliability, the resulting stream must be kept within the area where all circles intersect.
4 Implementation

This chapter contains a short, practical, description on how the concept stream was developed and why some steps were, had to be, or weren’t taken.

4.1 First Thoughts on the Problem

The encoders available during this project are not able to construct a video stream using the AVC still pictures method. This does not necessarily mean there are no encoders that support and can construct AVC still picture streams. If such encoders exist, the technical advisor at the project initiator did not know of them and they were not available to us in this project.

Two implementation methods on how to approach the development problem were elaborated:

Method A. A still picture stream could be created from scratch using the theoretical knowledge and pre-encoded MPEG-4 images.

Method B. A pre-recorded video stream could be modified to a video stream only containing AVC still pictures.

To know which of these methods was most likely to succeed, a video stream consisting of still image material was constructed using the video editing software Final Cut pro. The video stream was 25 frames per second. It consisted of a slide show with a "scene" change every two seconds and an end picture with color bars, the images can be seen in Figure 22. The image encoding difficulty is varying as the slide show progresses.

![Figure 22: A representation of the input slide show material](image)

Sound was also added to know if the prototype streams actually played but no images where shown.

The video and audio streams were fed to a Thomson ViBE EM2000 MPEG-4 AVC encoder which encoded the input material at High Profile @ Level 3, in standard definition resolution (720 x 576 pixels) and then multiplexed the ES streams onto a MPEG-2 transport stream. A number of recordings with different encoder settings were done. GOP lengths 1, 12 and 64, constant and variable bit rate (CBR/VBR) were recorded and examined.

With the first prototypes, the video stream was encoded at 3 Mbit/s with the PCR clock carried by the video PID. Though, later in the project, a 1 Mbit/s transport stream with video encoded at 700 Kbit/s CBR and GOP-length 24 was recorded and used as the original stream. This stream had no PCR clock embedded in the video PID, why this had to be done is described in chapter 4.2.2. The lower bit rate led to smaller files and less unnecessary data to work with. The images maintained an acceptable quality even at a 700 Kbit/s bit rate.

The streams were recorded using a device named JDSU DTS-330, which recorded the raw transport stream. This device could then, with software described in chapter 4.3.1, analyze the transport stream and interpret the MPEG-2 and -4 syntax. After recording the stream, some analyzing and practice were done, to get a feel of how it was to work with the data and to know what implementation method to use. Two "prototypes" were developed by hand, more about these in chapter 4.2.1.
The huge amount of data was a surprise. Normally, a 10 MB file is considered a pretty small in today's world. Though, it is roughly 10 million bytes, 80 million bits. Knowing that byte and even bit level modifications would be necessary was mainly the reason to go with implementation method B\(^4\).

Other advantages of modifying a pre-recorded video stream was that most variables and syntax would be correct to start with and the parts not interesting to this thesis would not need as much time, and not be a source of error, as they could be in implementation method A\(^5\).

### 4.2 Iterations

It was evident early that many prototype streams were going to be developed and also that they could not be done by hand, as in editing all the bits and bytes manually. Editing the prototype streams by hand would be a both monotone and not very scientific method. Instead it was decided that detecting patterns, using the standard documents as guidelines and building programs that used these patterns to alter the prototype streams, would be a good way of proceeding.

The subheadings in this chapter are only some of the prototype streams developed, but should be considered the main iterations where breakthroughs in the development process were made. The streams are referred to as prototype streams until the “final” stream, the concept stream, is developed.

#### 4.2.1 The First Two Prototypes

The first two prototypes if they can be called prototypes were made by hand and were more of a try-it-out process before deciding whether or not to build a new stream from scratch.

These prototypes were disasters in functionality. What was done here was cutting and pasting data from one place in the transport stream file to another place. By cutting from the first transport stream packet in an IDR picture PES to the first in the next PES a complete picture was acquired. By pasting those bytes after another picture we hoped to see this picture at another place in the stream of pictures. A simple illustration of this is presented in Figure 23.

![Figure 23: Cut-and-pasting in transport streams are not a good idea.](image)

This reckless method meant messing up the transport stream packet continuity counter as well as accidentally moving packets with other PID’s. Transport stream packets adhering to audio PES’s were moved and this caused the presentation timestamps and decoding timestamps to be completely off, if the whole audio PES even got sent. Program Specific Information (PSI) was also moved which caused additional errors.

---

\(^4\) Method B: Pre-recorded video stream modified.

\(^5\) Method A: Still picture stream created from scratch.
Although most of the syntax was correctly set, as the AVC_still_present flag in the AVC video descriptor in the PMT, the fixed_frame_rate_flag and end of sequences, nothing worked as expected. The first two prototype streams were big failures but led us in the right direction.

Due to the amount of errors and severity of these errors, it was at this point clear that we would need to build programs to do the dirty work for us. The first program constructed was used to reset all transport stream packet continuity counters, and the second one to alter all the PMT’s in the stream. Later versions are described in chapter 4.3.3.

4.2.2 Stripping the Stream

One theory, emerging from the H.222 standard document, was that the stream would only consist of IDR pictures and as each new scene begins with an IDR picture, why not use this to our advantage?

The idea for this next major prototype was to use the existing transport stream structure, keeping all the PSI and only remove the redundant video data. This meant finding the I-, P-, B-pictures and replacing all their transport stream packets with null packets. By removing these pictures, only IDR pictures where present in the video PID. This lead to a couple of problems; one of which had to do with something called the idr_pic_id. The error occurred because two consecutive IDR images may not have the same idr_pic_id value. The bit sequence that needed to be altered was in the slice header and was represented with Exponential Golomb (Exp-Golomb) coding.

Every even IDR picture was to be set from “idr_pic_id 0” to “idr_pic_id 1”. But as shown in Table 4, the Exp-Golomb representation of 1 instead of 0 uses two more bits and there had to be room for this change in the slice header. There was no room in the header of one picture, this lead to severe decoding errors. Changing the odd IDR pictures instead, there was room and the alteration succeeded.

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Binary</th>
<th>Exp-Golomb</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>010</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>011</td>
</tr>
</tbody>
</table>

Table 4: Conversion table between decimal, binary and Exp-Golomb coded bits

An even more challenging problem had to do with the PCR clock. When stripping the video stream, transport stream packets which carried the PCR clock were removed and the TS stream was now invalid as the PCR clock occurred infrequently and had wrong values.

Leaving the TS packets containing the PCR clock was no option as we then had the headers of some of the I-, P-, B pictures remaining and caused decoding and reference errors. It was therefore decided to record a stream without an embedded PCR clock and to add the clock in a later stage.

Figure 24 shows us the difference between the amount of video transport stream packets before and after the stripping of I-, P- and B-pictures. This is a good way of illustrating the average bandwidth gain but also the issue if bandwidth peaks in this stream.
This prototype stream will henceforth be referred to as the stripped prototype stream.

### 4.2.3 Fixing the Timing

With the use of the new stream (1 Mbit/s transport stream with video encoded at 700 Kbit/s CBR), without the PCR clock embedded, a correct syntax AVC still picture stream was constructed using the above steps. The problem for the next iteration was to deal with the picture timing. As no PCR clock had been used, the pictures were presented at the moment the receiver had decoded them. And the decoding started when all the TS packets for the PES had been received. This meant that easily encoded ("small") pictures had a long display time and pictures hard to encode ("big") pictures just flashed by.

The definition of which PID is carrying the PCR clock is found as a post in the PMT. By signal a new, unused PID as the PCR PID in the PMT and inject packets with this PID in to the transport stream at a predetermined interval (20 ms was used) the PCR clock was added to the prototype stream.

The PCR clock value had then to be set in these packets. By constructing a start value for the PCR clock and from that start value calculate the correct value for the PCR clock in each packet, the timing accuracy could be kept within the ±500 nanoseconds needed. Equation 2 was used to calculate the PCR clock value of packet\((i)\).

\[
PCR(i) = ValueOfLastTimestamp + (bytesSinceLastTimestamp \times PCRTicksPerByte)
\]  

(2)

The PCRTicksPerByte value was elaborated using a transport stream, with the same bitrate as the prototype stream, but containing PCR clocks. It was calculated by using Equation 3 and 4. The value is connected to the transport stream bitrate.
The presentation timestamp (PTS) and decoding timestamp (DTS) for the PES packets needed to be set correctly with the use of the produced PCR clock. This was done using the Equation 5 and 6. The DTSC and PTSC are values to ensure the picture is completely received and ready to be decoded and presented when the PCR timestamp occurs.

\[
PCR(i) = PCR_{base}(i) \times 300 + PCR_{ext}(i)
\]  

(3)

\[
\frac{\Delta PCR}{\Delta BYTE_{POS}} = PCRTicksPerByte
\]  

(4)

The presentation timestamp (PTS) and decoding timestamp (DTS) for the PES packets needed to be set correctly with the use of the produced PCR clock. This was done using the Equation 5 and 6. The DTSC and PTSC are values to ensure the picture is completely received and ready to be decoded and presented when the PCR timestamp occurs.

\[
DTS = ValueOfLastTimestamp + (\text{bytesSinceLastTimestamp} + \text{bytesUntilEndOfCodedPicture}) \times PCRTicksPerByte + DTSC
\]  

(5)

\[
PTS = ValueOfLastTimestamp + (\text{bytesSinceLastTimestamp} + \text{bytesUntilEndOfCodedPicture}) \times PCRTicksPerByte + DTSC + PTSC
\]  

(6)

By doing this, the timing could be controlled and the problem of too long or too short presentation time was solved. A side effect of letting the PCR clock be carried in a separate PID is the unnecessary data being transferred with every PCR clock packet. With every PCR clock packet, there are 12 “useful” bytes and 176 bytes of stuffing (unnecessary data). This leads to a higher bitrate for the total transport stream.

### 4.2.4 Smoothing the Stream

As the stream should function well together with other streams in a multiplex, it is important that the bandwidth utilization is homogenized, without peaks. This view is covered extensively in the master thesis report by Anton Alila (submitted 2010), but will be covered very briefly in this report as it still is a vital part of the project.

The bandwidth utilization homogenization is done by sending the video packets for the IDR picture during a longer time frame. As the transport stream now only contains the IDR pictures, there is a “window” with null packets after each IDR picture. We decided to name this window the “packet window”. In the stripped transport stream, the IDR picture is sent during the beginning of this window, we want to use the whole window to send it.

During the second packet window at which picture(i) is displayed, the TS packets adhering to picture(i+1) is sent with a “distribution ratio” calculated using Equation 7. The distribution ratio decides how often to send a video TS packet.

\[
\frac{\text{VideoPackets} + \text{NullPackets}}{\text{VideoPackets}} = \text{DistributionRatio}
\]  

(7)

By inserting the video data with a calculated ratio we have basically remultiplexed the transport stream, and while the average bitrate will be the same as the stripped prototype stream, the concept transport stream will now have less bandwidth peaks. Figure 25 shows the original transport stream, the stripped- and the homogenized prototype stream. The last of which also shows how the distribution ratio is varying.
The final transport stream, the “concept stream” is developed using the methods described in these iterations. The resulting stream was used in the observation verification on consumer receivers in chapter 5.1. How this stream performed, from a bit efficiency point of view is described in chapter 5.2.

4.3 Tools and Utilities

4.3.1 Analyzing Hardware and Software

The main software used for analyzing the stream was made by Interra Systems. The software used for analyzing the NAL units and thus SPS, PPS and slices was the Interra Vega H264 Analyzer 7.6 (Vega). This software decoded the stream of pictures and if a picture encoding was faulty, we could see it with this software.

Though the Vega software was able to interpret and give us some information on transport stream level, the DVB Transport Stream Analyzer 6 (TSA) software from Acterna was a more potent choice in these situations. The TSA gave a good overview of what was wrong in the stream; a screenshot from TSA, taken in an early development stage i.e. with lots of errors can be seen in Figure 26.

Figure 25: The original TS on top followed by the stripped prototype TS and the concept stream at the bottom. Black pixels are video TS packets, white pixels are null packets and gray pixels are audio or PSI packets.
The TSA was also able to show us graphs of the bandwidth utilization for different PID's. Many of the bandwidth diagram screenshots in this paper is produced using this software. This software was housed in the JDSU DTS-330 hardware. This is a computer with feature to record and play transports stream using an ASI interface.

4.3.2 Hex Editor

Some work still had to be done by hand; adding the end of sequence NAL units for example. This was best done in a hexadecimal (hex) editor. The hex editor interprets the byte stream (transport stream) and presents the bytes in base 16. The hex editor software used was Hex Workshop v6, developed by BreakPoint Software Inc. This software has the feature of color mapping which lets the user add filters to highlight certain byte series. It was used mainly for highlighting different TS packet headers.

The Hex Workshop software also has a comparison feature which compares two files, both in a byte-to-byte technique (Simple) and in an "intelligent" technique (Resynchronizing). Both of these techniques were used during the development to compare transport streams before and after a java program (see chapter 4.3.3) was used on the file. The “intelligent” comparison technique worked well for less complicated changes in the stream. But as the modification of the TS files grew more complex, like when starting to remultiplex/smooth the stream, the intelligent comparison technique went haywire and the simple technique had to be used.

Hex Workshop was also used when debugging and troubleshooting the java programs with a sort of reverse engineering method. The java program was used on the original stream and the result was then analyzed. The byte area where the program had malfunctioned was analyzed with Hex Workshop to understand what the cause of the malfunction was.

4.3.3 Java

During the development a number of java programs were developed and used as tools to modify the transport stream. Developing programs to do the job was decided early in the development process. It became evident that the amount of modification needed to be made on a transport stream would be too much to do by hand, and most important, it would be monotone work and not very scientific. The java programs were developed using the Java EE in the Eclipse Integrated Development Environment.

Seven programs were developed and all of them are some kind of parsers. The programs could be merged to one, but we chose not to do so for two reasons:

1. The programs were developed at different iterations in the development process and the later programs are more skillfully programmed and better structured.
2. We felt it would provide a better overview and would be easier to handle, keeping the programs separate.

Using prior knowledge from the transport control protocol (TCP); I implemented a sliding window method early in the development process. By using the sliding window, patterns in the streams can be detected. The sliding window approach is explained in Figure 27.

```
Sliding window length: 3   Find pattern: 47 40 6D

47 40 6D 31 01 60 00 00 01 E0 00 00 8B CO
XX XX 47 ≠ 47 40 6D

47 40 6D 31 01 60 00 00 01 E0 00 00 8B CO
XX 47 40 ≠ 47 40 6D

47 40 6D 31 01 60 00 00 01 E0 00 00 8B CO
47 40 6D ≠ 47 40 6D -→ Match!
```

*Figure 27: Using a sliding window (red marking) to match the pattern 47 40 6D*

Most of the programs read from one TS file, while simultaneously writing only the desired packets or the modified packets to a new TS file. The exception is the program to remultiplex the TS or homogenize the bandwidth peaks. This program reads the complete input TS file and creates an array of TS packets (a data type implemented by me). The other programs could be modified to use this approach as well, but there was no need. This particular program needed to keep all TS packets in the memory as it scatters packets. It still writes to a new file.

Constructing the java programs was the phase where most of the project work was done. The programming and code itself is nothing new or ground breaking; it was used to fit a purpose and to build the tools needed for a specific task or iteration.

### 4.4 Practical Verification

During the development of the concept stream, the prototype streams had to be tested. The observations done on the professional decoder is considered a part of the development process. The consumer receiver observation can be considered to be in a gray area between the development- and the result phase.

#### 4.4.1 Professional Decoder

The prototype streams were observed on a Tandberg RX1290 (shown in Figure 28), a professional decoder for both MPEG-2 and -4 audio and video, during the development. It was the visual results from this receiver, in collaboration with the results from the Vega and TSA software that decided on how to proceed with the development.
The RX1290 was set up with the JDSU DTS-330 feeding a prototype transport stream by an Asynchronous Serial Interface (ASI) directly to the decoder. This way, no RF modulation had to be done at this point. The RX1290 was then hooked up to a monitor that displayed the decoded material. The web interface of the decoder was the only way of getting any information, from the decoder other than the visual information from the monitor. Unfortunately, the web interface did not give much feedback. More on this in chapter 6.3.

4.4.2 Consumer Receivers

When the concept stream was developed it had to be observed whether or not it really worked on consumer receivers, as this was a part of the problem statement and an aim of the project.

It is important to point out that this was not a scientific experiment as not all of the variables could be controlled. The correct term to use is a controlled observation.

A lot of testing for the NorDig standard is done at Teracom AB, taking advantage of this; we were able to set up an observation environment at their labs. The observation subjects consisted of 6 TVs with integrated digital receiver (IDTV) and 10 set top boxes which were connected to a display (a TV). All receivers had support for MPEG-4 as this encoding was used for both audio and video.

The concept stream was multiplexed on to a transport stream with three live TV channels to be able to switch channels. The MPTS was then modulated on to a RF channel and the receivers received the signal by RF. Not all the receivers could be run simultaneously. First the IDTVs were observed and the TV that performed best was kept as a “reference” monitor when testing the set top boxes.

The following aspects were observed, both for set top boxes and IDTV receivers:

1. Audio playing (any sound is OK)
2. Flawless audio
3. Video playing (any image is OK)
4. Flawless video
5. Shows first picture
6. Shows last picture
7. Correct sync, audio-video
8. Correct picture timing (two seconds)

The receivers were then graded “pass” or “fail” depending on how well they managed an aspect. The aspects are in no specific order. For a working product, all of the aspects above must be acceptable managed by the receiver.

The terms “flawless audio” and “flawless video” is used to describe the class of the output material. If the receiver could present “video” or audio in the same class as the input material, it was considered flawless. The correct synchronization between audio and video was established by using the last picture of color bars; the correct audio should then be a sinus tone.
5 Results

An AVC still picture stream was achieved. This stream is called the concept stream as it implements the concept of AVC still pictures. The stream sends proper still pictures, one intra coded picture every two seconds. The results of the consumer receiver observation and the concept stream bitrate results will be presented in this chapter.

5.1 Consumer Receivers

One of the tasks of this master thesis project was to evaluate how the concept stream worked on consumer MPEG-4 AVC receivers which are currently on the market. To not risk putting a manufacturer in a bad spotlight, I have chosen to censor the results of the consumer receiver observation. There is no need for the receiver name to be included in this report. This will therefore be an overview of the results. However, Appendix C contains the complete observation sheet. How the observation was performed is explained in chapter 4.4.2.

There were 6 IDTV receivers, 3 of them demonstrated results which we considered to be satisfactory for AVC still pictures. The 10 set top box (STB) receivers did not do as good. Only one receiver was able to present the concept stream satisfactory and this only with a very prolonged channel zapping time.

The receiver which was graded to have “passed” on the most aspects of the observation was an IDTV receiver, with a "score" of 7 (out of 8). This IDTV was then used as the reference monitor when observing the set top boxes. A "scoreboard" of the top 6 receivers is shown in Figure 29.

![Top Score](image)

*Figure 29: The “top 6” consumer receivers rated on how many aspects passed in the observation*

When observing the set top box which "scored" highest (6 out of 8), the audio stream came to use. This receiver had trouble with the correct audio and video synchronization. With the help of the audio stream we could then notice this receivers synchronization problem as the color bar picture did not appear at the same time as the sinus tone.

In total; only 4 out of 16 receivers were able to display the still pictures in the concept stream satisfactory. Three of them were IDTV receivers. The one set top box that managed to present AVC still pictures was not aimed for the Swedish market.
What this means for a possible introduction of AVC still pictures on the Swedish market is discussed in chapter 6.2.

### 5.2 Bitrate Efficiency

The main aim of this master thesis is to evaluate the bitrate for a concept stream using the AVC still picture method. Comparing the results between this new method of sending still pictures and the conventional method is a bit difficult as the conventional method adds stuffing to the P and B frames to maintain a constant bitrate. Where the original stream will have an almost constant bandwidth use, the bandwidth of the concept stream on the other hand will vary with the size of the frame encoded.

The original stream had video bitrate of 0.7 Mbit/s. This is a very low bitrate if one is talking about moving video, which normally would be between 3 and 4 Mbit/s per service (including audio). The bandwidth history graph for the original, unmodified stream can be seen in Figure 30.

![Original stream graph](image)

*Figure 30: The original stream. Video bitrate is approximately 0.7 Mbit/s.*

The TSA software, described in chapter 4.3.1, has been used to create the graphs in this chapter. But to make a graph of the stripped prototype stream with this software would be misleading. The true peak bitrate value of the stripped prototype stream would not be correctly represented in such graph.

The bitrate is measured as the number of bits processed per time unit. To do this, the size of a “window” over which the measurement is done has to be defined. The size of this window is not foretold for the TSA software, but believed to be one second. The TSA software will therefore calculate the average bitrate over one second.

The stripped prototype stream has a very irregular momentary bitrate. In fact, during the short moment the IDR picture is sent, it is sent with a bitrate of 700 Kbit/s. It is the same as in the original stream, but after the picture has been sent, the stream is “silent” for up to two seconds. This phenomenon can be seen in Figure 24 in chapter 4.2.2.

---

6The picture display time for the concept stream is two seconds and depending on the picture size, the time of “silence” will vary.
A one second window would thus provide us with a faulty bitrate as it calculates the average bit rate over the window and not the actual peak bitrate. Instead, a representation of the bandwidth used by the stripped prototype stream is shown in Figure 31.

![Figure 31: An illustration of the theoretical bitrate used by a stripped prototype stream](image)

In theory, the peak bitrate for an AVC still picture transport stream can never be below the size of the largest IDR picture divided by the time this image is sent. As seen in Figure 32, the largest IDR picture for the concept stream is somewhere around 408 Kbit. It would therefore take 0.58 seconds for this IDR picture to be transmitted at 700 Kbit/s. The rest of the 2 second picture display time no video is being sent and the video bitrate is 0 Kbit/s. The concept stream is constructed to use the whole 2 seconds to send the IDR picture, at a lower bitrate. It uses a constant bitrate for 2 seconds and then changes the bitrate depending on the IDR picture size; the graph in Figure 33 is thus a little misleading as it does not represent the sudden changes in bitrate. The complete theory behind this method is covered in Alila’s (submitted 2010) master thesis paper.

The peak bitrate of the concept stream has a theoretical maximum calculated in Equation 8.

\[
\text{TheoreticalPeakBitrate} = \frac{\text{SizeOfLargestIDR}}{\text{SendingTime}} = \frac{408 \text{ Kbit}}{2 \text{ seconds}} = 204 \text{ Kbit/s}
\]
Figure 33: AVC still picture stream with remuxed video packets to minimize “peaks”. Video bitrate is max 0.22 Mbit/s. The main thing is the difference in smoothness.

As seen in Figure 33, the achieved peak bitrate of the concept stream is about 0.22 Mbit/s. This value is very close to the theoretical value calculated in Equation 8. The difference can be explained by the implementation of the distribution ratio variable in one of the java programs. This variable was lowered to ensure all packets would reach their destination in time to be decoded and presented at the correct timestamp.

5.3 Picture Quality

Since there still are intra coded pictures from the original stream left in the concept stream, no picture quality will be lost. On the contrary; the conventional method of displaying still pictures will have problems with image jerkiness in low bitrate still picture material. This jerkiness has its source in the fact that no encoder is perfect and will detect small (non-existing changes) between frames, these changes will be encoded in to the B- and P pictures. When watching the low bitrate conventional still pictures, one can detect jerkiness in the pictures; they are not completely “still”.

With the AVC still picture method, these B- and P pictures do not exist and no non-existing changes are sent. This will lead to a completely still picture, without jerkiness.
6 Conclusions

This chapter will explain the results of this project and discuss if there is a future for the AVC still picture method. It will also refer back to the aim and problem statements of this master thesis.

6.1 AVC Still Pictures vs. Conventional Method

It is possible to send still picture material using the AVC still picture method using an MPEG-2 transport stream in the DVB-T network. The method can be very bandwidth efficient if used correctly, but could have big negative effect on the viewing experience if used too greedily.

The bandwidth saved will depend on how often an IDR picture is sent and the size of that IDR picture. During this project, we managed to lower the video bandwidth used by the original transport stream from 700 Kbit/s peak bitrate to 220 Kbit/s peak bitrate. Without losing any picture quality, alternatively, one could say we gained picture quality as the jerkiness of the still pictures disappeared. The concept stream still had some issues with an uneven bitrate.

A way to avoid the uneven bitrate would be to start with a fixed encoded picture size of let’s say 200 Kbit per IDR picture. The picture quality would then vary while the picture size stays permanent. This would give us a more constant bitrate and the momentary bitrate could be managed without difficulty.

Still, there is a minimum bitrate for a transport stream; the program specific information (PSI) must be sent, and the PCR clock as well. As stated in the DVB standard (2007), a packet with the PCR clock must be received every 100 ms, as must the PMT and the PAT. This minimum average bitrate for a transport stream containing only PAT, PMT and PCR packets is described in Equation 9. This value is too low as there are other PSI that are needed and using exactly the maximum time interval of 100 ms is walking the line.

\[
\text{MinimumBitrate} = \frac{(\text{PATPacketSize} + \text{PMTPacketSize} + \text{PCRPacketSize})}{\text{MaximumTimeInterval}}
\]

\[
= \frac{188 \text{ byte} + 188 \text{ byte} + 188 \text{ byte}}{0.1 \text{ second}} = 5640 \text{ byte/s} \approx 45 \text{ Kbit/s}
\]

(9)

Upon the bandwidth of 45 Kbit/s, the AVC still image stream bandwidth is added. The graph in Figure 34 is a bandwidth graph of a theoretical AVC still picture stream with a constant IDR picture size of 200 Kbit. This looks pretty good, by sending an IDR picture every 4 seconds, we will get a bitrate below 100 Kbit/s. Though, sending IDR pictures this rarely is not such a good idea.
Using the method too greedily, by sending the IDR pictures too far apart, cause's problems for the end consumer. As she switches to a channel with AVC still pictures, the receiver will not display anything before an IDR picture is received and decoded. For example; 4 seconds between the IDR pictures would lead to the receiver displaying 4 seconds of "nothingness" in the worst case scenario.

The problem of how long a consumer is willing to wait in this “nothingness” is not covered in this thesis. In my opinion; 4 seconds would probably be way too long to wait. On the other hand, if the option is to always have the “nothingness” on this channel when not sending, 4 seconds might be “better late than never”.

The AVC still picture streams could have one more weakness; if the transport stream should be compromised by a burst or noise error and a bit error occur, the compromised IDR picture received cannot be decoded properly. We would then lose this frame entirely, or have some sort of decoding error. If the conventional method was used, there would be an intra coded picture sent, and hopefully received, within a GOP length’s time, i.e. in about one second.

6.2 Introduction on the Swedish Market

Based on the observation performed in this master thesis project, I draw the conclusion that there is a high probability that very few of the MPEG-4 AVC receivers on the market today has support for AVC still picture streams. This is the biggest problem of the method.

During the observation in this project, IDTV receivers were found to provide better support for the AVC still picture stream than the set top boxes. The main reason for this is probably the fact that the IDTV’s are built for a wider market. The set top boxes have tighter economic margins and therefore, only the most essential features are implemented.

The IDTV’s that are built for a wider market will probably comply better with the MHEG-5 standard, used mainly in Great Britain. A form of AVC still pictures, very similar to the one covered in this thesis, has been used there and this is probably why some IDTV's performed very well in presenting AVC still pictures.

The fact that no receiver crashed during the project means the AVC still picture streams probably could be introduced in the Swedish DVB-T network, without any major difficulties, in a near future. Only the consumers with receivers supporting AVC still pictures would then see the images. Bear in mind that this is then only meant to be used on the radio channels, SR P1-P4 and maybe on channels with shared bandwidth.
6.3 Discussion

Probably the biggest problem throughout the development process was the inability of the professional decoder, the Tandberg RX1290 (presented in chapter 4.4.1), to show proper debugging information.

During observations of unsuccessful prototype streams on the RX1290, the decoder web interface would simply state “Video: Fail”. There was no useful information available. Hardware with a proper debugging feature, or the possibility to observe the stream on more than one professional decoder, would have made the development process easier and definitely more straightforward.

If another original file had been chosen to start with, the concept stream would probably look a little different in terms of bandwidth usage. I think the stream we chose to work with was a good proof of concept. A lower original stream bitrate would of course lead to a concept stream with a lower peak bitrate, but not as good picture quality.

Using an original stream with a longer GOP period would result in fewer I-pictures and thus allowing the IDR-pictures to be encoded with a higher quality and size. This would have lead to a hypothetic concept stream with better picture quality but higher bitrate as not as many I-pictures could have been removed. An original stream with shorter GOP period would consequently have lead to a poorer picture quality but less bitrate.

In the concept stream, the PCR clock is carried in a separate PID. This is very inefficient as it generates a large overhead in the PCR PID TS packets (explained in chapter 4.2.3). If the AVC still picture method would be implemented in the DVB-T network, the PCR clock would most definitely be embedded in another PID, such as the PMT or the video PID.

As this concept stream was developed to be a proof of concept, placing the PCR clock in an individual PID was considered OK as it was a way of minimizing sources of error.

6.4 Further Research

The concept stream or an equivalent stream must be properly tested on a wider array of consumer receivers to document which decoder chipset have support for AVC still pictures. The manufacturers of the chipset and receivers must then be contacted and informed about the AVC still picture method so that they can implement support for it. The receiver firmware can then be updated using an Over The Air (OTA) update method available in DVB-T. This is something the consumer wouldn't even notice, let alone having to deal with the firmware update themselves.

A system of constructing the AVC still picture streams is also needed. This should, according to me, be a server solution where the content provider can upload images to a web interface and construct a slideshow from there. The system would then encode the images using MPEG-4 AVC and create an AVC still picture transport stream with settings chosen by the content provider. The system would then feed this transport stream directly to the multiplex. The content provider would then be able to broadcast still images in mere minutes.

Research on how long a consumer is willing to wait for their receiver to present an image has to be done. This knowledge is essential to know exactly how efficient the AVC still picture method really is. This would be a good subject for another master thesis.
7 References


Appendix A

A NAL unit is a package in the Network Abstraction Layer in MPEG-4 AVC. The contents of a NAL unit is called the Raw Byte Sequence Payload. This appendix contains a list of NAL units available in MPEG-4 AVC. The NAL units used in this master thesis project are mainly: 5 to 10 and 12.

<table>
<thead>
<tr>
<th>NAL unit type</th>
<th>RBSP data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unspecified</td>
</tr>
<tr>
<td>1</td>
<td>Coded slice of a non-IDR picture</td>
</tr>
<tr>
<td>2</td>
<td>Coded slice data partition A</td>
</tr>
<tr>
<td>3</td>
<td>Coded slice data partition B</td>
</tr>
<tr>
<td>4</td>
<td>Coded slice data partition C</td>
</tr>
<tr>
<td>5</td>
<td>Coded slice of an IDR picture</td>
</tr>
<tr>
<td>6</td>
<td>Supplemental enhancement information (SEI)</td>
</tr>
<tr>
<td>7</td>
<td>Sequence parameter set</td>
</tr>
<tr>
<td>8</td>
<td>Picture parameter set</td>
</tr>
<tr>
<td>9</td>
<td>Access unit delimiter</td>
</tr>
<tr>
<td>10</td>
<td>End of sequence</td>
</tr>
<tr>
<td>11</td>
<td>End of stream</td>
</tr>
<tr>
<td>12</td>
<td>Filler data</td>
</tr>
<tr>
<td>13</td>
<td>Sequence parameter set extension</td>
</tr>
<tr>
<td>14</td>
<td>Prefix NAL unit in scalable extension</td>
</tr>
<tr>
<td>15</td>
<td>Subset sequence parameter set</td>
</tr>
<tr>
<td>16</td>
<td>Reserved</td>
</tr>
<tr>
<td>17</td>
<td>Reserved</td>
</tr>
<tr>
<td>18</td>
<td>Reserved</td>
</tr>
<tr>
<td>19</td>
<td>Coded slice of an auxiliary coded picture without partitioning</td>
</tr>
<tr>
<td>20</td>
<td>Coded slice in scalable extension</td>
</tr>
<tr>
<td>21</td>
<td>Reserved or unspecified</td>
</tr>
<tr>
<td>...</td>
<td>Reserved or unspecified</td>
</tr>
<tr>
<td>31</td>
<td>Reserved or unspecified</td>
</tr>
</tbody>
</table>
A transport stream consists of packets with a fixed length; 188 byte. Every one of these packets has a packet header. This header can look different depending on the stream and the content of the TS packet. This appendix contains a picture from Watkinson (2004) where the transport stream packet is explained. The numbers below the labels are the bit length of that position.
Appendix C

This appendix shows a censored observation sheet, the result of the practical observation. I have chosen to censor the observation sheet to not risk putting a receiver or decoder manufacturer in a bad spotlight.

IDTV = Integrated Digital Television Receiver
STB = Set Top Box Receiver
P = Pass
F = Fail

<table>
<thead>
<tr>
<th>Type</th>
<th>#1</th>
<th>#2</th>
<th>#3</th>
<th>#4</th>
<th>#5</th>
<th>#6</th>
<th>#7</th>
<th>#8</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDTV</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>IDTV</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>P</td>
<td>F</td>
</tr>
<tr>
<td>IDTV</td>
<td>P</td>
<td>F</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>P</td>
<td>F</td>
</tr>
<tr>
<td>IDTV</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>P</td>
<td>F</td>
</tr>
<tr>
<td>IDTV</td>
<td>P</td>
<td>F</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>P</td>
<td>F</td>
</tr>
<tr>
<td>IDTV</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>P</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>STB</td>
<td>P</td>
<td>P</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>