A Prototype Application for Objective Video Quality Assessment

Johan Moberg
Preface

The master thesis was carried out by Johan Moberg at Neava Consulting AB, Luleå 2006/2007. It is a part of the Master of Science program in Media Technology at Luleå University of technology (LTU).

I would like to thank my supervisors Staffan Johansson at Neava Consulting and Kåre Synnes at LTU. Also I’d like to thank all of those who proof read the thesis and all who participated in the subjective testing. But especially I would like to thank my family who supported me throughout all the work on the thesis.

Thank you.
Abstract

The area of mobile communications has evolved in the last years and with the introduction of 3rd generation cellphones users can use features like video calls and mobile television. There are various bottlenecks in this kind of communication that introduce distortions to the transmitted video; therefore it is desirable to measure the video quality of these services. A subjective measurement takes long time to conduct, time which often isn’t available. Hence objective measurements with good correlation with the subjective quality evaluations are needed.

The purpose of this master thesis is to develop a proof-of-concept application which objectively measures video quality.

Traditional video quality algorithms are based on error-sensitivity. The human visual system on the other hand is adapted to extract structural information from the viewing field. In this thesis the Structural Similarity Index Measurement (SSIM) algorithm is used. This algorithm is based on structural distortions and considers image degradations as perceived structural information loss. It is a full reference algorithm and thus needs access to the reference sequence in order to generate a quality rating.

To be able to test the SSIM algorithm two video sequences were selected; one of a woman talking on the phone and the other with a rotating chessboard. Different distortions were then applied to the sequences to get a testing material. In the end ten distorted sequences from each reference were chosen, making a total of 20 sequences.

A subjective testing was conducted to get a reference quality measure for all generated video sequences. In total, there were 45 participants from Neava in Luleå and Ericsson in Lund in the testing which in the end rendered a mean opinion score (MOS) for each of the 20 sequences. Later the MOS from the subjective tests were compared to the objective quality rating from the SSIM algorithm.

In order to obtain the objective quality ratings an application prototype was developed. The prototype implements the SSIM algorithm and displays the quality for each frame individually in a graph and as a mean of the entire video sequence. In addition the application offers dual playback of both the reference and deformed sequences at the same time, and for demonstrational purposes there is also a graph which can show for instance block errors or received signal strength. These parameters can then be compared to the video quality rating. By then examining the results from the subjective- respectively objective tests the thesis hopes to answer questions about the reliability of objective measurements.

The results look promising, but as discussed in the thesis further improvements can be made in different areas to improve the quality ratings.
1 Introduction

1.1 Background

When the third generation mobile networks were introduced on the market the mobile communications was improved with higher data rates which made it possible to stream video. It is now possible to make video calls and watch live television on a cellular phone. This triggers the need of being able to objectively determine the video quality in different testing scenarios. To make these quality analyses subjectively cost lots of money and time. This is not feasible in a large scale.

In this thesis an application is developed which implements an algorithm for this purpose. The goal with such an algorithm is to, as close as possible, resemble how a real person would perceive video quality. Since humans are the ultimate receivers of the transmitted data we are also the ones who can best determine what good and bad quality is. The goal of the objective video quality algorithms is to match the quality a real person would perceive.

At the time of this writing the mobile networks are upgraded with higher speeds, like High-Speed Downlink Packet Access (HSDPA), in the downlink and the cellphones are equipped with larger screens with higher resolutions. Due to this the video quality can be enhanced with compression algorithms that don’t destroy as much data and displays the video in a higher resolution.

In today’s wireless networks transmission errors may result in distortions in the received video data. A video quality metric can be used to optimize the transmission of video data and minimize transmission errors. The most reliable way to determine the quality is to conduct a subjective evaluation but in most cases it is too time consuming and expensive to be a realistic option. Instead an objective video quality metric can be implemented to automatically determine the perceived video quality.

Three examples of usage scenarios are presented to show how an objective metric can be used:

- Test new phone models to verify that they meet certain standards for video playback under various conditions.
- In the case of mobile video the video quality could be monitored with different radio conditions and scenarios. This way the base stations could be optimized in the area of video performance.
- Benchmark video processing systems and algorithms for video compression.

These are some areas of usage, but of course it can be used in many other situations as well.

1.2 Problem description

In most cases the most reliable way to determine video quality is by conducting a subjective evaluation. In such an evaluation a large group of people are collected, preferably of different backgrounds, to judge the video quality. A mean opinion score (MOS) is produced from the result.

However, this is a very costly and time consuming operation to perform. An objective evaluation which is performed in just a matter of seconds or minutes would be a great improvement.
The objective evaluation should match the MOS as close as possible in order to be an alternative to the subjective evaluation.

A prototype application has been developed that uses an algorithm to calculate a quality rating for a video sequence, provided that it has access to the original sequence. Questions about how objective quality measurements can be used and how they relate to a MOS score are answered in this thesis.

1.3 Limitations

The application and especially the algorithm are designed with the assumption that two video sequences are already synchronized. The deformed sequence is compared to the original frame-by-frame, starting at the first frame in both sequences and presumes that they should be identical.

In the case of cellphones and the MPEG-4 video format the sync layer could probably be utilized to achieve frame synchronization.

In the application the resolution of the video windows is fixed to 320*240 pixels. It can handle smaller and larger video resolutions but they will be up- or downscaled to fit the window.

When conducting a subjective testing a large test group consisting of persons with varying background is preferable. A larger test group makes the MOS more accurate and there is less variance. Due to time aspects the test group couldn’t be as large as intended and mainly consist of people with a technical background.
2 THEORY

2.1 Digital images and video introduction

The purpose of this chapter is to give a basic introduction to digital images and video, a more detailed introduction can be found in [4]. Naturally the signals of images are analog but they can be converted to a digital format in an analog-to-digital conversion process where they are first sampled and then quantized. Another word for this is that they are digitized.

The first step in the digitization is the sampling process. It takes a continuous analog signal and samples that signal and indexes each sample using integer numbers. Each of these sampled numbers is often referred to as a picture element or pixel for short. The pixels are stored in a matrix, thus a digital image signal is typically represented by a two-dimensional array of discrete signal samples. The dimension of a signal is the number of coordinates that are required to index a given point in the image (see figure 1). As a consequence digital image processing is quite data intensive, this means that significant storage and computational resources are needed when working with them. The second part of image digitization is quantization. In this step the image signal intensities is represented using a finite precision. When the grey-level of an image pixel is quantized, it is assigned to be one of a finite set of numbers which is in the grey-level range. In other words a value is assigned to the pixel, describing the color or grey-level in this pixel.

![Digital Image](image.png)

Figure 1. Image dimensions.

Digital processing of video requires that the video is in digital format, thus it needs to be sampled and quantized. The quantization process is essentially the same as for images, but video sampling involves a new dimension which images don’t have. Namely time.

To handle the time dimension, digital video processing uses frames to simulate moving pictures (see figure 2). At a given rate images are captured from the videostream and when they are played in sequence it becomes a video. Each of those captured images is called a frame and the term framerate is used to describe the rate at which the frames are captured. A frame rate of 25 fps (frames per second) means that 25 images are captured each second. A
higher fps will make the video more smooth and a lower makes it seem jerky. Usually an fps around 25-30 is sufficient for most users. This is because the human eye is limited and might not detect any flicker. The human visual system (HVS) is further described in [2, 10].

2.1.1 Video compression and transmission errors

This chapter is a brief introduction to video compression with focus on MPEG-2 since it is basic and easy to understand. It is also discussed how transmission errors might affect video playback. A more exhaustive explanation about compression techniques and video formats can be found in [4].

Video compression is all about removing or reduce redundant information. There are two types of redundancies when dealing with moving images; spatial and temporal.

The spatial correlation deals with individual images and reduces the redundancies within each frame. An example of this is JPEG compression.Temporal correlation identifies redundancies between successive frames in a video sequence. It is based on the fact that frames over a certain time interval are similar and can be partially constructed from other frames.

In most implementations of video compression both spatial and temporal techniques are used since higher compression ratios can be achieved. However, when the transmission is unreliable the image may suffer from more artifacts.

MPEG-2 is a commonly used video format which uses both spatial and temporal correlation. One example of this is the DVD standard which uses MPEG-2.
There are three types of frames used in MPEG-2, they are called I, P and B-frames.

- I-frames are called *intra-coded* frames. They contain information to describe the entire frame.
- P-frames or *predictive-coded* frames relies on information from a previous I or P frame in order to recreate the scene.
- B-frames are bi-directional; hence they are called *bi-directionally* coded frames. They need information from both the previous and following frames in order to draw the scene fully.

P-frames provide more compression than an I-frame because they take advantage of the data in the previous I- or P-frame. Both I and P-frames are called reference frames. Figure 3 shows an example of how the video frames can be encoded.

When video is streamed over a wireless interface there will be bottlenecks that generate transmission errors. Transmission errors will appear when data packets are lost or don’t arrive in time for decoding. This will result in artifacts in the video.

The artifacts will be more or less visible depending on which data is lost. The impact of the lost data will also be larger if an I-frame is lost.

### 2.2 Video quality introduction

One of the most successful and widespread inventions of the twentieth century is the moving picture. In recent years the video signal has moved from the analog to the digital domain. With powerful compression algorithms and video processing equipment this has been made possible. A priority when designing the digital video systems has been to provide a quality which is superior to the analog systems while reducing the bandwidth and storage requirements.

The digital video bitstreams may be received imperfectly when the information is sent over error-prone channels such as wireless networks. The quality can also be downgraded if lossy
compression techniques are used. Therefore guaranteeing a certain level of quality is an important concern for the content providers.

Who decides what good quality is then? A computer is able to analyze video sequences but even if it tells us that the quality is perfect a human might not agree if there are visible errors. In subjective evaluations humans evaluate the quality of images or video sequences while in an objective evaluation a computer uses different algorithms to calculate a quality value. The goal of an objective evaluation is to simulate the human visual system to a degree where it produces the same results as a subjective evaluation would.

2.3 Video quality analysis

When dealing with video there are two methods of determining the quality; either a subjective- or objective quality measurement is performed. In a subjective measurement humans determine the quality and in an objective the quality analysis is performed by, for example, a computer. The objective measurement could also be made automatically as will be seen later on in the thesis.

2.3.1 Subjective quality measurement

The most reliable way of assessing the perceived quality of an image or a video is to conduct a subjective evaluation. A subjective quality evaluation is obtained from a number of human observers and results in a MOS score. However the MOS method is slow and expensive so there is a great need for accurate objective quality measurements.

There are different factors to consider when conducting a subjective measurement. The viewers all have individual interests and expectations. There are many factors that will have an effect on the subjective result, a few of them are listed here.

- Different display types will render different video qualities. Also the size, resolution of the display and viewing distance affects the result.
- The quality bar rises with time. A VHS recording, which was of good quality a couple of years ago, will now be considered to have lesser quality by everyone that has a DVD player.
- A sports fan that attentively watches a game of some kind would probably have different quality requirements then a person with no interest in the sport.

More factors are listed in [2].

2.3.2 Objective quality measurement

Subjective measurements are costly, both in time and money. Therefore there is a need to be able to execute these quality measurements automatically. The problem is that the results still need to be accurate when compared to a MOS score.

There are three different scenarios when dealing with objective video quality; full reference (FR), reduced reference (RR) and no reference (NR). It is called full reference when the reference video can be accessed for comparison (see figure 4). This way an algorithm can compare an impaired video sequence against a reference (which is considered of perfect quality). The Structural Similarity Index Measurement (SSIM) algorithm (see chapter 3.1.1) which this thesis is based on is a FR metric. The second type is reduced reference; here the original image or video signal is not fully available. Instead
certain information are extracted from the reference and transmitted to the quality assessment system as side information to help in the evaluation. There are situations where no reference exists, but there is still a need to evaluate the quality. This is called no reference, for a human this is no difficult task. He or she doesn’t need a reference to evaluate the quality but for a computer this is very difficult. Currently the most widely used FR video quality metrics are mean squared error (MSE) and peak signal-to-noise ratio (PSNR). They are further described in chapter 2.4.

Figure 4. Full reference perceptual quality measurement system.

2.4 Objective quality metrics

This chapter describes the two most commonly used video quality metrics; MSE and PSNR. Both metrics are very common and simple to both implement and use.

2.4.1 Mean squared error

Consider two images; the original \( x \) and the distorted image \( y \). If \( i \) defines the i-th pixel in the respective image MSE can be calculated like this:

\[
MSE = \frac{1}{N} \sum_{i=1}^{N} (x_i - y_i)^2
\]  

(1)

Where \( N \) is the total number of pixels in the original image. For the MSE result to be good it should be as close to zero as possible. A zero result means that there is a perfect match between the images.

2.4.2 Peak signal-to-noise ratio

For digital images the PSNR is the ratio between maximum possible value that each pixel can assume and the noise in the image.

\[
PSNR = 10 \cdot \log_{10} \left( \frac{L^2}{MSE} \right)
\]  

(2)

Here \( L \) is the number of possible values each pixel can have. For a greyscale image this would be the number of steps in the greyscale. This metric is mostly used to measure the quality of reconstruction in image compression. A study by the ITU [9], where objective quality metrics are compared to each other, shows that no other model outperforms PSNR in all conditions. Although one problem with this kind of
error-sensitivity based metric is that it doesn’t consider how the human eye perceives the images. Hence two images with the same PSNR quality rating can have very different errors. In the next chapter an algorithm called SSIM is described which is based on structural distortions instead.
3 Methods

3.1 Image quality analysis

"The main function of the human visual system is to extract structural information from the viewing field, and the human visual system is highly adapted for this purpose. Therefore a measurement of structural distortion should be a good approximation of perceived image distortion." – Zhou Wang

A major difference between a philosophy based on structural distortions and traditional error-sensitivity based philosophies is that image degradations are considered as perceived structural information loss instead of perceived errors. Another difference is that this philosophy is a top-down approach which simulates the overall function of the HVS. While the error-sensitivity based philosophy uses a bottom-up approach which tries to simulate the function of each component in the HVS and then combine them together.

3.1.1 Image quality using SSIM

SSIM is based on structural distortions and hence consider the structural information in an image as those attributes that reflect the structure of the objects in the scene, independent of the average luminance and contrast. This leads to an image quality assessment approach that separates the measurement of luminance, contrast and structural distortions. SSIM is a full reference quality assessment since it has access to both the original and the impaired image. Let \( x \) and \( y \) be two images of the same width and height and define \( x = \{ x_i \mid i = 1, 2, 3, \ldots, N \} \) and \( y = \{ y_i \mid i = 1, 2, 3, \ldots, N \} \). Then the measure between \( x \) and \( y \) can be defined as a function of the luminance, contrast and structure comparison measures:

\[
S(x, y) = l(x, y) \cdot c(x, y) \cdot s(x, y)
\]  

Let the mean of \( x \) and the mean of \( y \) be defined as:

\[
\mu_x = \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i, \quad \mu_y = \bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_i
\]

The variance of \( x \) and the variance of \( y \) as:

\[
\sigma_x^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2, \quad \sigma_y^2 = \frac{1}{N-1} \sum_{i=1}^{N} (y_i - \bar{y})^2
\]

And the covariance, or the measure of how much one signal has changed nonlinearly towards the other signal, is defined as:

\[
\sigma_{xy} = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})
\]  

Then the luminance, contrast and structure measure comparisons can be defined as:
\begin{align*}
l(x, y) &= \frac{2 \cdot \mu_x \cdot \mu_y}{\mu_x^2 + \mu_y^2}, \\
c(x, y) &= \frac{2 \cdot \sigma_x \cdot \sigma_y}{\sigma_x^2 + \sigma_y^2}, \\
s(x, y) &= \frac{\sigma_{xy}}{\sigma_x \cdot \sigma_y}.
\end{align*}

These terms are independent in such a way that if the luminance or the contrast is changed in either image it will not affect the third term. Geometrically \( s(x, y) \) corresponds to the cosine of the angle between \( x - \mu_x \) and \( y - \mu_y \). It reflects the similarity between two image structures and equals one only if the structures of the compared images are exactly the same.

The similarity index measure between \( x \) and \( y \) can be calculated if (7) is inserted into (3):

\begin{equation}
S(x, y) = \frac{4 \cdot \mu_x \cdot \mu_y \cdot \sigma_{xy}}{(\mu_x^2 + \mu_y^2) \cdot (\sigma_x^2 + \sigma_y^2)}
\end{equation}

One problem with (8) is that if the denominator of either the luminance or the contrast is zero or close to zero the measurement will be unstable. To avoid this problem, two constants \( C_1 \) and \( C_2 \), are added. The resulting measure is named Structural Similarity Index Measurement:

\begin{equation}
SSIM(x, y) = \frac{(2 \cdot \mu_x \cdot \mu_y + C_1) \cdot (2 \cdot \sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1) \cdot (\sigma_x^2 + \sigma_y^2 + C_2)}
\end{equation}

The constants, \( C_1 \) and \( C_2 \) are given by

\begin{equation}
C_1 = (K_1 \cdot L)^2, \quad C_2 = (K_2 \cdot L)^2
\end{equation}

where \( L \) is the dynamic range of the pixel values, or the color depth of the image. For 8 bits/pixel greyscale images \( L \) is 255. \( K_1 \) and \( K_2 \) has small values so that they will only affect equation (9) when the denominators are small. In [1] these values are set to 0.01 and 0.03 for \( K_1 \) and \( K_2 \) respectively.

The SSIM index satisfies three conditions:

1. \( SSIM(x,y) = SSIM(y,x) \)
2. \( SSIM(x,y) \geq 1 \)
3. \( SSIM(x,y) = 1 \) if and only if \( x = y \).

The SSIM algorithm is applied for quality assessment of still images using a sliding window approach. In [1] the window size is fixed to 8x8 pixels. The SSIM index is calculated within the sliding window, which moves, row by row from the top-left to the bottom-right corner of the image. Then the overall quality value is calculated as the average of all the windows.

### 3.1.2 SSIM versus MSE

In an error-based algorithm such as MSE certain structured images will get low ratings even though a human will not see the degradations. The structure based SSIM algorithm on the other hand is much better in handling those distortions which will be shown here.

The figures and quality ratings used in this subchapter are taken from [10].

In the first image, Tiffany (figure 5), an image has been compressed to save space. The left side is the original and the right side is the distorted image. The distortions are clearly visible in the face of the woman.
In the other image, Mandrill (figure 6), the distortions in the right image are barely visible to the human eye. But when comparing the MSE quality ratings one will see that they are about the same; 165 for Tiffany and 167 for Mandrill. SSIM though will rate the two figures more like we perceive them. Tiffany rates 0.37 and Mandrill 0.8 out of a maximum of 1 so in this case SSIM is a lot more accurate than MSE.

**Figure 5 – Tiffany. Original to the left and distorted to the right. MSE= 165 and SSIM=0.37.**

**Figure 6 – Mandrill. Original to the left and distorted to the right. MSE=167 and SSIM=0.8.**
3.2 Video quality using SSIM

When measuring the quality of video there is not only one frame but a range of frames to be considered. The quality of the distorted video is thus measured in three levels: the local region (window) level, the frame level and sequence level (see figure 7).

![Figure 7. A system for measuring video quality using SSIM.](image)

To be able to do a comparison between two sequences they first need to be aligned, the frames in the distorted video needs to be compared to the same frames in the original. SSIM assumes that they are already aligned.

To limit the number of calculations the algorithm can select only a proportion of all possible windows in the frame, either random or by making the window move more then one pixel at the time. This reduces the computational cost but still maintains rather robust measurement results.

When measuring video each frames color space is divided into $Y$, $C_b$ and $C_r$ color components and combined into a quality measure using a weighted summation. Let $SSIM_{ij}$ be the SSIM index value for the $j$-th sampling window in the $i$-th frame. Then the local quality index is given by:

$$SSIM_{ij} = W_y \cdot SSIM_{ij}^Y + W_{Cb} \cdot SSIM_{ij}^{Cb} + W_{Cr} \cdot SSIM_{ij}^{Cr}$$

Where $SSIM_{ij}^Y$, $SSIM_{ij}^{Cb}$ and $SSIM_{ij}^{Cr}$ denote the index values of the $Y$, $C_b$ and $C_r$ components of the $j$-th window in the $i$-th frame respectively. In [1] the weights are set to $W_y=0.8$ and $W_{Cb}=W_{Cr}=0.1$. The weight of the luminance component is highest since humans notice errors in the luminance channel much more then in the other.

Another difference when measuring video is that each window is weighted with consideration to its average luminance. Humans tend to have a higher acceptance to errors in dark areas (areas with low luminance) then in brighter areas. Dark regions usually don’t attract eye fixations and are thus assigned smaller weighting values. This results in a frame-level quality index, $Q_i$ for the $i$-th frame:

$$Q_i = \frac{\sum_{j=1}^{M} (w_y \cdot SSIM_{ij})}{\sum_{j=1}^{M} w_y}$$

(12)
Where \( w_{ij} \) is the weight for the \( j \)-th window in the \( i \)-th frame and \( M \) is the number of windows. The mean value of \( x \) in (4) is used to calculate the local window weights. The weight is adjusted as

\[
w_{ij} = \begin{cases} 
0 & \mu_x \leq 40 \\
(\mu_x - 40) / 10 & 40 < \mu_x \leq 50 \\
1 & \mu_x > 50
\end{cases}
\] (13)

In [1] a weighting on frame-level describes the case when large global movements occur in between frames. A block-based motion estimation algorithm is used to evaluate one frames motion with respect to its adjacent next frame. If there is a large amount of motion the frame will get a lower weighting. Since this weighting isn’t implemented in this project it will not be further discussed.

Finally the overall quality of the entire video sequence is given by

\[
Q = \frac{\sum_{i=1}^{F} Q_i}{F}
\] (14)

where \( F \) is the total number of frames in the sequence. \( Q \) is the average quality over the sequence.

## 3.3 Application

The purpose of the application development is to demonstrate a prototype of an objective video quality metric. The goal is to find out how the metric performs compared to a subjective evaluation. The application was developed in Java. More details on the application implementation can be found in appendix A.1.

### 3.3.1 Interface and functions

The application (see figure 9) is designed with the capability to show two different video sequences at the same time. This way the user easily can compare the sequences to each other. Playback controls from JMF was added to make the navigation easier and also a separate button to start playback of both sequences at the same time was added. The playback controls is used to navigate in the sequence and another way to navigate is to use the left- and right arrows of the keyboard to step frame-by-frame in the video.

![Figure 8. The file menu.](image)
Under the file menu (see figure 8) the following options can be found:

- **Analyze** – this is used to determine the currently loaded deformed sequence using the SSIM algorithm. The original video must also be loaded.
- **Load original** – loads a video sequence to the left (original) video window. Both this option and the **Load deformed** option open a file dialogue.
- **Load deformed** – loads a video sequence to the right (deformed) video window.
- **Save result** – used to save the resulting graph(s) to a csv-file (comma separated values file) which can be opened in Microsoft Excel. Both the save- and load file menu options open a file dialog. A filename is generated which describes the analysis, but the user can also enter a filename of choice.
- **Load result** – loads csv-files and displays them in the graphs.
- **Exit** – quits the application.

The graphs in the application display information about the quality and other information that might be of interest. The top graph displays the results from the SSIM quality metric for each separate frame. The user can select a certain frame of interest in this graph and the application will move the video to this frame so it can be easily evaluated. It is also possible to zoom in the graphs by right-clicking in the graph. A marker in the graphs shows the user which frame that is being evaluated.

In the prototype application the graphs below the quality measurement is there for demonstrational purposes. In this case the bottom graph displays the signal strength and packet loss as if the video was transmitted over a wireless packet switched network. The purpose is to get a comparison between the quality of the video and the different parameters in a network that can influence the transmission rate and quality. For instance the user could see what kind of effect the signal strength has on the quality of the video transmission. The application is designed so it is possible to display any data the user captures during the video transmission.

In this thesis no real values has been used in this graph, instead random values has been used as a demonstration of how it can be used and what it would look like. This is since there are no real data to present due to the fact that the video sequences used are produced in a controlled environment, see chapter 4.1.2.
Figure 9. This snapshot of the application is taken after an analyze has been completed and the user has clicked in the graph to evaluate a certain frame in the sequences.
When the analysis of the two files is complete, i.e. when the SSIM algorithm is finished comparing the quality of the two sequences, statistics from the algorithm will be shown in the *Statistics* window which is then opened (see figure 10). This includes the mean value of the SSIM quality measure of the complete sequence and also minimum-, maximum- and median value. The information in the statistics window is also stored in the csv-file if the user chooses to save the result.

![Statistics Window](image.png)

**Figure 10.** Statistics window from the application. This window is opened after the quality analysis is completed.
4 Evaluation

4.1 Testing

To evaluate the SSIM algorithms quality measurement against the HVS a subjective testing was arranged. The results of this test were then compared to the quality measures which SSIM generated for the same video sequences.

Two different video sequences were used in the testing, one with a chessboard that starts to rotate around the centre and finally stops the rotation. The second sequence has a woman talking in the phone. To compare the algorithm with real results a subjective testing was made with the two sequences, in these tests different distortions was applied to the sequences. The subjective testing resulted in a MOS which can be seen as an average.

In the objective testing the algorithm was run on the same sequences as the subjective testing, the results can be read in chapter 4.2.

4.1.1 Testing material

The same video sequences were used in both the subjective and objective testing. As a base there were two reference sequences. These were then distorted in different ways to get an impaired or reduced quality.

![Figure 11. The first frame in the reference Chess sequence.](image)

The first sequence, Chess, is a rotating chessboard (see figure 11). It is seven seconds long and consists of 175 frames.
The second sequence, Susi, is a close up of a woman talking on the phone (see figure 12) and is ten seconds (shortened from 15 in the original) and 250 frames. This sequence comes from the Tektronix website [3]. The resolution in both sequences was changed to 320x240 pixels since mobile phones rarely handle higher resolutions and it makes the application interface easier to handle in this prototype.

The two sequences were selected because of their different image structures. The chess board has very defined structures with straight angles while the woman in the phone is more natural. Errors, or distortions, of the same nature will not appear in the same way in images with different structures due to the HVS. Another reason for choosing the woman is the fact that this sequence resembles a video call using a cellphone. Together the two clips spans a larger group of different image types, this way the robustness of the SSIM algorithm can be better evaluated then if only one were used. A larger selection of sequences would produce a better result but that is out of the scope of this thesis since the subjective testing would take to long to perform.

4.1.2 Applied distortions

To evaluate how the quality measurement behaves under different circumstances, different known distortions were added to the two sequences. Some of the distortions were chosen to resemble a video call and other to test more basic quality degradations, like a change in brightness for example. A subjective test was then conducted to get a MOS score for each distortion. Read more about how the testing was setup in chapter 4.1.3. The purpose of the evaluation is to evaluate the SSIM algorithm and thus the MOS was compared to the quality measure which SSIM calculated. Comparing these measures gives a good estimation of how SSIM corresponds with the HVS. The effects were added to the sequences using Adobe Premiere Pro. Here follows pictures and a short description of the distortions that was introduced to the sequences.
The brightness of each individual frame was increased in figure 13.

Here, in figure 14, both the brightness and contrast of the sequences was adjusted to higher values.

An effect which simulates an image leaving the focal range of the camera and blurring the clip was added in figure 15.
The two clips in figure 16 have a higher contrast than the original.

A rather large Gaussian blur was introduced to the sequences in figure 17.

The sequences in figure 18 were exposed to a sharpening effect.
Random noise was added to the sequences in figure 19 and figure 20.

Noise was added to this clip as well.

In figure 21 each image in the sequence was pixellated to create a blockier image.
A larger pixellation then in figure 21 was introduced to the clips in figure 22.

4.1.3 Subjective testing

A subjective testing was conducted to get a proper comparison between the SSIM algorithm and the HVS. Since the persons testing the sequences in a subjective testing are the ultimate receivers of this kind of media it is the best way to get a good estimation. For the testing a program called MSU Perceptual Video Quality tool was used. This program lets the test coordinator choose between a few different methods and in this case double stimulus impairment scale (DSIS) best fit the purpose. This method is described in ITU-R BT.500-11. In DSIS the video are shown consequently in pairs: the first one is the reference, and the subject is informed about it, and the second one is impaired. After the playback the subject is asked to give his opinion using the impairment scale in table 1.

<table>
<thead>
<tr>
<th>Subjective MOS score</th>
<th>Objective quality measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>5, Imperceptible</td>
<td>1,0</td>
</tr>
<tr>
<td>4, Perceptible</td>
<td>0,8</td>
</tr>
<tr>
<td>3, Slightly annoying</td>
<td>0,6</td>
</tr>
<tr>
<td>2, Annoying</td>
<td>0,4</td>
</tr>
<tr>
<td>1, Very annoying</td>
<td>0,2</td>
</tr>
</tbody>
</table>

Table 1. The impairment scale used in the subjective tests.

Since the SSIM quality measure ranges between [-1, 1] (with only inverted images resulting in negative values) a conversion had to be made. This can be done in a number of ways but one has to keep in mind what the impairment scale says, not only look at the numbers. For instance, is “very annoying” a zero on a scale from [0,1] or is it maybe 0,2? The mapping between the subjective MOS score and the objective quality measure is shown in table 2.

The negative values of SSIM are left out since there is only a remote chance, if any, that a complete sequence will become inverted. And the reason for mapping the MOS score “very
“annoying” to “0.2” is that a zero in this case only could be generated from a sequence without resemblance to the original. And this is considered as more then “very annoying” which is the lowest MOS score.

The subjective testing consisted of two test groups, one at Neava Consulting in Luleå and one at Ericsson Mobile Platforms (EMP) in Lund. Totally 45 persons were engaged in the tests.

4.1.4 Objective testing using SSIM

In the objective testing the application was used to calculate a SSIM quality measure for each video sequence. Only the mean value is later used in the comparison with the subjective testing. The other generated values can be of interest for other purposes but not when compared to the subjective testing since the MOS score is a mean value.

4.2 Results

This section discusses the results from the subjective- and objective tests and compares them to each other. The goal is to get an understanding of how well the quality ratings of the SSIM algorithm corresponds to the subjective MOS score.

To make the graphs in the chapter easier to understand the distortions were indexed from A-J as in table 3 (the distortions can be seen in figure 14 to figure 22):

<table>
<thead>
<tr>
<th>Clip</th>
<th>Distortion</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Brightness</td>
</tr>
<tr>
<td>B</td>
<td>Brightness and contrast</td>
</tr>
<tr>
<td>C</td>
<td>Camera blur</td>
</tr>
<tr>
<td>D</td>
<td>Contrast</td>
</tr>
<tr>
<td>E</td>
<td>Gaussian blur 25</td>
</tr>
<tr>
<td>F</td>
<td>Gaussian sharpen</td>
</tr>
<tr>
<td>G</td>
<td>Noise 50</td>
</tr>
<tr>
<td>H</td>
<td>Noise his auto 100</td>
</tr>
<tr>
<td>I</td>
<td>Mosaic 50</td>
</tr>
<tr>
<td>J</td>
<td>Mosaic 100</td>
</tr>
</tbody>
</table>

Table 3. An index over the different distortions.

4.2.1 Results from the subjective testing

The average age of the test groups were 35 years at Neava and 28 years at EMP in Lund. Counting both groups the average age was 31 years. Most participants were male and with a technical background, this might make the MOS score lower then a more evenly distributed group would have. Table 4 shows the average age and total number of participating test subjects in the two groups.

<table>
<thead>
<tr>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subjects</td>
<td>45</td>
</tr>
<tr>
<td>Average age</td>
<td>31</td>
</tr>
</tbody>
</table>

Table 4. Test subjects and average age for the test groups.
A comparison between the groups shows that Neava’s MOS rates are generally higher than EMP (see figure 23). The people at EMP might be more experienced in video quality and therefore the MOS is lower than Neava’s. The biggest difference is in clip C (camera blur) where the MOS from Neava is about twice as high as EMP.

From here on the two test groups are combined as one when the results are discussed.

**Figure 23. A comparison between the two subjective test groups.**

An interesting, and also expected, result is that the MOS differ between the two sequences (Susi and Chess) even though the same filters were added to both (see figure 24). This probably has to do with the human visual system. When the face of a human is distorted (as in the Susi sequences) it looks more unordinary than if a chessboard is distorted. The biggest difference between the sequences is when the camera blur filter is used (clip C), the straight lines in the chessboard doesn’t feel as out of place as when the face of the woman is blurred. This sequence was also the one where the MOS ratings from the two test groups disagreed most.
4.2.2 Results from the objective testing

The SSIM quality ratings can be seen in figure 25. To make it easier to compare the results with the subjective MOS scores the ratings has been multiplied by ten. So instead of a range from [0,1] the range is [0,10].

The two test sequences show some differences although the same distortions are applied. The biggest difference can be seen in clip G (called mosaic 50) where Susi rates about two points higher than Chess.
4.2.3 Comparison of the results

This subchapter will compare the results from the subjective and objective tests in order to evaluate the performance of the SSIM algorithm.

In figure 26 the subjective and objective quality evaluations of the Susi sequences is compared. It can be seen that the objective quality metric has best performance when there are small distortions, like changes in brightness and contrast. When the distortions become larger the metric rates the sequences to high compared to the subjective ratings. Clip C with the camera blur is an example of this.
Figure 26. A comparison between the subjective MOS and objective SSIM quality rating for the Susi sequence.

Figure 27 shows the same comparison but for the chess sequence. Also here it can be seen that the metric is to kind in its evaluation. But the MOS scores are higher here which helps the SSIM algorithm when it rates the sequences to high.

Figure 27. A comparison between the subjective MOS and objective SSIM quality rating for the Chess sequence.
As the figures show, SSIM in this state shows some imperfections in detecting the structural differences. The sequences with brightness and contrast distortions are handled well. But figure 26 and figure 27 shows that clip C, E, G and H was very overrated by the algorithm while J was underrated. A look at table 3 shows that these clips are in fact the clips with blurring, mosaic and noise effects which all mostly effect the structural information in the images.

In an effort to correct this, a new weighting on frame level was added in the algorithm. [4, 8] describes a weighting for the structure, luminance and contrast separately. If the images \(x\) and \(y\) are considered the equation looks like this:

\[
SSIM (x, y) = \left[ l(x, y) \right]^\alpha \cdot \left[ c(x, y) \right]^\beta \cdot \left[ s(x, y) \right]^\lambda
\]  

Where \(\alpha > 0\), \(\beta > 0\) and \(\lambda > 0\) are parameters that adjust the relative importance of the three components and the functions are from equation (7). This definition still satisfies the three conditions given in chapter 3.1.1.

4.2.4 Results with image component weighting

The new frame level weighting, described in the previous chapter, was implemented into the algorithm and the weighting for structure (ie \(\alpha\) in equation 15) was set to a higher value then the other weightings. This chapter shows the results with this new weighting, and in figure 28 and figure 29 it can be seen that there is a visible improvement. The algorithm is still not perfect but has improved the quality rating for many sequences, especially those which were overrated. Some of the ratings are also worse with the weighting then before.

The results show that the algorithm can be tuned to closer match a subjective evaluation. This is a good option when mobile video calls are considered, since most distortions will be structural changes.
Figure 28. A comparison between the two SSIM algorithms and the subjective test result using the Susi sequence.

Figure 29. A comparison between the two SSIM algorithms and the subjective test result using the Chess sequence.
5 Discussion

In this chapter the results of the subjective- and objective testing is discussed and also if SSIM can be used as an objective quality measurement. There is also a discussion about the handling of dropped frames and what enhancements that can be made to the application and to the algorithm.

5.1 SSIM as an objective quality metric

The results in chapter 4.2.4 show that the image component weighting is an improvement to the algorithm, at least when dealing with these kinds of errors and sequences. Even for other types of errors (or sequences) this weighting is a better method since it offers a means to fine tune the algorithm.

SSIM in this state could be used to monitor the video quality in a mobile network where the original sequence exists. Quality degradations will be seen in the tests with the SSIM algorithm, though the quality rating has to be tuned further. Right now the ratings will not tell the user much about the quality without first making extensive tests to find out what each grade on the scale relates to in video quality.

The results demonstrate that the algorithm isn’t perfect, but one has to consider that also the subjective evaluation has flaws. Bearing this in mind the algorithm is a good start and it could be used in many scenarios where video quality is concerned. A subjective evaluation will always be the most accurate way to get the MOS, but in time the objective metrics will be closing in.

5.2 Dealing with dropped frames

In the case when video is transmitted over a packet switched wireless network dropped frames can become an issue. In wireless networks it is common that packets are lost or arrive too late. When the data rate over the network becomes to slow or even stops the video will also stop or become jerky. This becomes interesting in the perspective of objective video quality. How can one handle these frame freezes when performing a FR analysis of the two sequences? Here is an example of two ways to handle it; either the comparison continues as if nothing happened (see alternative 2 in figure 30) or the original video is halted until the video stream is received again (as in alternative 1 in figure 30).

The first alternative could be used when streaming a music video for example, since the application can wait for the video buffer to fill up again before continuing. If this alternative is used a weighting will have to be introduced that measures the impact that the dropped frames will have on the total quality of the sequence.

The second alternative is more fitting for a video conversation where only a small delay is accepted. If the application would wait for the missing frames a delay, as long as the dropped frame interval, would be introduced. The dropped frames could be dealt with in two ways in this case; either use a weighting as in alternative 1 or just let the quality metric run as nothing happened and it will compare the last frame that arrived with the corresponding missed frame. A subjective evaluation is needed for both alternatives if the weighting solution were to be implemented.
5.3 Future work

Different ideas and solutions have come up during the work with this thesis but the timeframe hasn’t allowed for everything to be implemented. This chapter describes some of the suggestions that haven’t been tested but might come in handy or at least be worth considering when designing this kind of application. The section is split into two parts; applications improvements and improvements for the SSIM algorithm.

5.3.1 Application

The application was designed to make it easy to apply new algorithms and this is also true for sound, not only video. Some improvements in the GUI to show the sound waves and a “sound grabber” similar to the frame grabber can be implemented for sound analysis. The QualityIndex class can then be used to present data in different ways.

Another good implementation would be a batch mode for the analysis. In the case of wireless streaming an original file could be streamed several times with different controlling parameters and produce a large quantity of deformed video sequences. It would be good if the user then could select an original and then compare all the deformed sequences against it in one swift operation. This could produce an Excel document with the result, and then the user could check into certain interesting deformed files manually with the application. This would save a lot of time in many test scenarios.

The Java Native Interface (JNI) can be used to code the algorithm in a language with better performance. The code for performing the algorithm calculations could, for example, be coded in C++ and still run from within the Java application. This way the time it takes to calculate the quality ratings can be greatly reduced.
5.3.2 SSIM algorithm

To further improve the SSIM quality measure in different situations a frame weighting based on movements can be implemented. This is described in [1].

Another weighting that could be tested and maybe implemented is a frame rate based weighting. This might be an issue since bad frames are exposed longer time to the viewer in a sequence with lower frame rate, then in one with higher. Especially when dealing with mobile phones since the frame rate differ a lot depending on what the user is watching. A subjective evaluation can determine if this is an issue or not.

Since SSIM calculates the overall quality rating of the entire sequence, as in equation (14), a counter for frame freezes could be interesting. In long sequences with a couple of frame freezes the SSIM quality would still be really high. The freezes will not make a large impact in those cases and therefore a counter would be a good complement.

As discussed in [8] there is a multi-scale implementation of the SSIM which also takes into consideration image resolution and viewing conditions. This approach could be implemented for a greater accuracy in the metric.

Lip sync, or sound synchronisation in general, is another issue that will affect the quality result. The viewer will be annoyed if the sound is out of sync and this can affect the perceived quality of the video. A subjective testing might show that video with the sound out of sync will get lower MOS scores than the same video with no sound at all. In countries where they dub movies this issue will be of lower importance since they are not used to lip sync.

The weighting of dark areas can make the algorithm unstable since dark images will get low scores if the whole image is dark. This happens because the dark areas get low mean values and thus are weighted with a zero in equation (13). The solution might be to adjust the weighting table.

5.4 Conclusions

The image component weighting improved the objective quality rating but it is not perfect. The distortions used in the video sequences are not realistic from a network transmission point of view. If real sequences could be used in the objective evaluations the algorithm can be tuned to better fit real life scenarios.

When the subjective and objective quality ratings are compared there is a mapping between them in order to be able to make the comparison. In this thesis the mapping is linear but this could be investigated further, maybe another mapping will generate better results.

In the subjective evaluation all the test subjects got to view a number of sequences. The viewing distance and environment was not controlled in any way so there might have been slight differences which might affect the MOS score. Also most participants were of the same gender and have the same education, so the selection doesn’t represent the wide range of people one could wish for.

Video quality is an interesting field and much have yet to be accomplished before there are perfect objective algorithms, if there can be one. The SSIM algorithm is easy to comprehend and has a low cost in computational time. It is far from perfect in this state but hopefully it will evolve, maybe merge with other algorithms to form even more accurate metrics. It will be interesting to see where the research will take this area in the future.
6 References


7 Glossary

A list of the abbreviations used in the thesis.

API Application Program Interface
AVI Audio-Video Interleaved
CSV Comma Separated Values
CVS Concurrent Versions System
DSIS Double Stimulus Impairment Scale
DVD Digital Versatile Disc
EMP Ericsson Mobile Platforms
FOBS Ffmpeg OBjectS
FPS Frames Per Second
FR Full Reference
GUI Graphical User Interface
HSDPA High-Speed Downlink Packet Access
HVS Human Visual System
ITU International Telecommunication Union
JMF Java Media Frameworks
JNI Java Native Interface
JPEG Joint Photographic Expert Group
MOS Mean Opinion Score
MPEG Moving Pictures Expert Group
MSE Mean Squared Error
NR No Reference
PSNR Peak Signal-to-Noise Ratio
QT QuickTime
RR Reduced Reference
RSCP Received Signal Code Power
SSIM Structural Similarity Index Measurement
UMTS Universal Mobile Telecommunication System
VHS Video Home System
Appendix

A.1 Application development

A.1.1 Design

Java was the programming language of choice because of its simplicity. There are many useful application program interfaces (API) in java which saves time. And because this is a prototype designed to provide a proof of concept, the improved performance of other languages isn’t as important as the simplicity Java provides.

Figure 31 shows the class structure in a simple way, and then follows a brief description of the different classes.

TestGraph
This is the main class of the project. It contains the main method which starts the application and it also builds the graphical user interface (GUI).

FrameGrabber
Contains methods for grabbing frames from a video sequence. Used by the algorithm to fetch the frames and then compare them.

Algorithm
This is an interface class for the algorithms. It contains certain methods that are common for all algorithm classes.
AlgorithmSSIM
This is where the SSIM algorithm is implemented. It grabs frames using the FrameGrabber and then performs computations and produces a quality value for each frame. The quality values are stored in a QualityIndex which the graphs use to present the result. It implements the Algorithm class.

SSIMReader
This class is used by AlgorithmSSIM to read and parse the settings-file (see chapter A.1.3).

AlgorithmPSNR
Used to calculate the Peak Signal-to-Noise ratio which is a simple quality value. It implements the Algorithm class.

QualityIndex
Stores the quality information provided by the algorithm classes.

GraphDefault
This is an interface class for the graphs. It contains certain methods that are common for all algorithms.

GraphQuality
Reads the QualityIndex and displays the values in a graph. It implements the GraphQuality class.

GraphTEMSData
This class reads the QualityIndex to find out how many frames that were analyzed. It then produces random signal strength and packet loss for demonstrational purposes. Implements the GraphQuality class.

A.1.2 Application flow
When the user clicks the Analyze item in the file menu a chain of events is started, the flow can be seen in figure 32. If the video sequences are loaded a FrameGrabber is assigned to each sequence. Later on, when the analysis begins, these grabbers are passed on to the SSIM algorithm so it can access the frames.

The algorithm then extracts the information from the frames and stores it in two matrices, one for the original and one for the deformed sequence. Now it can begin to calculate the quality values for the windows and frames. The results are then stored in a QualityIndex which is returned to the application (and later used by the graph classes) when the analysis is completed.
A.1.3 Parameters and settings

There are certain parameters used by the algorithm which the user can change in order to better fit the algorithm to the users needs. To set the different SSIM weights and change other parameters a settings-file is implemented. This way the user doesn’t have to recompile the source code to make changes. A sample of the file is shown in appendix A.2.

A.1.4 Implementation

This chapter describes different choices made when implementing the algorithm and also mentions which java APIs were used.

In chapter 3.1.1 and 3.2 a sliding window which uses random window positions is described. This is implemented in another way; instead of using random windows a stepping is introduced in the algorithm. A stepping of one pixel means that the window just moves one pixel forward in each slide. During the testing a stepping of the same size as the window has been used, the window size has been 8x8 pixels and thus the stepping was 8 pixels. Both the window size and stepping size can be set using the settings-file.

In java there is an API for handling video called Java Media Framework (JMF) [5] which is used in the application, both to play the video and to access the frames and convert them to matrices. In appendix B.1 there is a list of audio- and video formats that is supported by JMF.

For the graphs a tool called JFreeChart [6] was used (version 1.0.1), it is fairly easy to use and has many different graph implementations. The graphs used in the application are called XYSeries graphs.
A.1.5 Limitations

In this chapter the current limitations of the application is presented.

The JMF version available at the time of this writing (version 2.1.1e) supports these video formats among others:
AVI, MPEG-1, QT, H.261 and H.263. For a complete listing of formats see appendix B.1.
There is a project called Omnividea FOBS [7] which has a plugin for JMF called *Fobs4JMF* that adds compatibility for a lot of new formats like H.264 and XVID. At the time of this writing FOBS (version 0.4.0) only has support for frame grabbing in the concurrent version systems (CVS) repository but it will be included in future versions of FOBS.

When designing the GUI, the resolution of the video windows in the application was hard coded to be 320x240 pixels for demonstrational purposes. Higher and lower resolutions are automatically scaled to proper size. All demonstrational sequences have a 320x240 resolution; no larger resolutions were needed since this was made with a cellphone in mind.
Since the video is scaled the application should not be used to conduct subjective evaluations.

The application is only tested in Windows XP, changes might have to be made in the code if another operating system is used.
A.2 Settings file

Here follows a replica of the file used by the application to import the settings.

```
# This file is used to set weights and other parameters that
# SSIM (Structured Similarities Index Measurement) will use.
#
# Usage:
# Don't use spaces. Ex 'Y=0.8' works but 'Y = 0.8' doesn't.
# Only use comments on new lines. Ex 'Y=0.8 #comment' doesn't work.
# Author Johan Moberg

[weights]
# These weights are used by the algorithm.
# Y, Cb and Cr are weights for each colorchannel. They should add up to one.
# K1 and K2 are used in the denominator so it won't come to close to zero.
# Y=0.8
# Cb=0.1
# Cr=0.1
# K1=0.01
# K2=0.03

[colorformat]
# This is used to set which conversion to use between RGB->YCbCr.
# 1=YCbCr REC 709
# 0=YCbCr REC 601.1
colorformat=1

[windowparameters]
# These parameters change how the sliding window acts.
# Size is the side of one side of the square window.
# Stepping is how long steps it takes, ex 1 just moves the window one pixel.
# WindowWeighting is whether or not weighting on windowlevel is turned on. 1=on
# Size=8
# Stepping=8
windowWeighting=1
```
B.1 JMF Supported formats

This information is taken from Sun’s JMF homepage.

Supported Media Formats

JMF supports audio sample rates from 8KHz to 48KHz. Note that cross-platform version of JMF only supports the following rates: 8, 11.025, 11.127, 16, 22.05, 22.254, 32, 44.1, and 48 KHz.

The JMF 2.1.1 Reference Implementation supports the media types and formats listed in the table below. In this table:

- **D** indicates the format can be decoded and presented.
- **E** indicates the media stream can be encoded in the format.
- **read** indicates the media type can be used as input (read from a file)
- **write** indicates the media type can be generated as output (written to a file)

<table>
<thead>
<tr>
<th>Media Type</th>
<th>JMF 2.1.1 Cross Platform Version</th>
<th>JMF 2.1.1 Solaris/Linux Performance Pack</th>
<th>JMF 2.1.1 Windows Performance Pack</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIFF (.aiff)</td>
<td>read/write</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>8-bit mono/stereo</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>16-bit mono/stereo</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>G.711 (U-law)</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>A-law</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>IMA4 ADPCM</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>AVI (.avi)</td>
<td>read/write</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>Audio: 8-bit</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: 16-bit</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: DVI ADPCM</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: G.711 (U-law)</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: A-law</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Audio: GSM mono</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: ACM**</td>
<td>-</td>
<td>-</td>
<td>D,E</td>
</tr>
<tr>
<td>Video: Cinepak</td>
<td>D</td>
<td>D,E</td>
<td>D</td>
</tr>
<tr>
<td>Video: MJPEG (422)</td>
<td>D</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Video: RGB</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Video: YUV</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Format</td>
<td>Read/Write 1</td>
<td>Read/Write 2</td>
<td>Read/Write 3</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Video: VCM**</td>
<td>-</td>
<td>-</td>
<td>D,E</td>
</tr>
<tr>
<td>GSM (.gsm)</td>
<td>read/write</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>GSM mono audio</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>HotMedia (.mvr)</td>
<td>read only</td>
<td>read only</td>
<td>read only</td>
</tr>
<tr>
<td>IBM HotMedia</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>MIDI (.mid)</td>
<td>read only</td>
<td>read only</td>
<td>read only</td>
</tr>
<tr>
<td>Type 1 &amp; 2 MIDI</td>
<td>-</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>MPEG-1 Video (.mpg)</td>
<td>-</td>
<td>read only</td>
<td>read only</td>
</tr>
<tr>
<td>Multiplexed System stream</td>
<td>-</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Video-only stream</td>
<td>-</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>MPEG Layer II Audio (.mp2)</td>
<td>read only</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>MPEG layer 1, 2 audio</td>
<td>D</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>QuickTime (.mov)</td>
<td>read/write</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>Audio: 8 bits mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: 16 bits mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: G.711 (U-law)</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: A-law</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Audio: GSM mono</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: IMA4 ADPCM</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Video: Cinepak</td>
<td>D</td>
<td>D,E</td>
<td>D</td>
</tr>
<tr>
<td>Video: H.261</td>
<td>-</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Video: H.263</td>
<td>D</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Video: JPEG (420, 422, 444)</td>
<td>D</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: 8 bits mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>Audio: 16 bits mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>G.711 (U-law)</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>A-law</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Wave (.wav)</td>
<td>read/write</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>Sun Audio (.au)</td>
<td>read/write</td>
<td>read/write</td>
<td>read/write</td>
</tr>
<tr>
<td>8 bits mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>16 bits mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>G.711 (U-law)</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>8-bit mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>16-bit mono/stereo linear</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>G.711 (U-law)</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>A-law</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Format</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>GSM mono</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>DVI ADPCM</td>
<td>D,E</td>
<td>D,E</td>
<td>D,E</td>
</tr>
<tr>
<td>MS ADPCM</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>ACM**</td>
<td>-</td>
<td>-</td>
<td>D,E</td>
</tr>
</tbody>
</table>

Notes:

- ACM** - Window's Audio Compression Manager support. Tested for these formats: A-law, GSM610, MSNAudio, MSADPCM, Truespeech, mp3, PCM, Voxware AC8, Voxware AC10.
- VCM** - Window's Video Compression Manager support. Tested for these formats: IV41, IV51, VGPX, WINX, YV12, I263, CRAM, MPG4.