Seminar Report

H.264/MPEG-4 Advanced Video Coding

Alexander Hermans

Matriculation Number: 284141
RWTH

September 11, 2012

Contents

1 Introduction

Since digital images have been created, the need for compression has been clear. The amount of data needed to store an uncompressed image is large. But while it still might be feasible to store a full resolution uncompressed image, storing video sequences without compression is not. Assuming a normal frame-rate of about 25 frames per second, the amount of data needed to store one hour of high definition video is about 560GB[1]. Compressing each image on its own, would reduce this amount, but it would not make the video small enough to be stored on today's typical storage mediums. To overcome this problem, video compression algorithms have exploited the temporal redundancies in the video frames. By using previously encoded, or decoded, frames to predict values for the next frame, the data can be compressed with such a rate that storage becomes feasible. This approach to compress videos is also called the motion compensation approach.

Like earlier standards, such as MPEG-1 [4], MPEG-2 [5] and MPEG-4 advanced simple profile (ASP) [6], the MPEG-4 advanced video coding (AVC) standard [7] is a lossy compression standard and also based on the motion compensation approach. In many ways these standards are very similar, but each consecutive standard offered a higher compression, while maintaining a similar quality. This has been achieved by improving the steps in the compression algorithm, often at the cost of a higher complexity. In general it is said that MPEG-4 AVC gives double the compression rate, at the same quality, when compared to MPEG-4 ASP.

MPEG-4 AVC was developed by the Moving Picture Experts Group (MPEG), together with the ITU-T Video Coding Experts Group (VCEG). Since both groups have their own way of naming new compression standards, MPEG-4 AVC is also called H.264, which is the name given by VCEG. Both groups have competed in creating better video compression standards since the early nineties. VCEG created H.261 in 1988, which was the first practical video compression standard. Many of today's video compression algorithms are similar to H.261. Over the years both the VCEG and MPEG created their own standards. In 1996 they released MPEG-2 Part 2/H.262, which was also created under cooperation of both groups. Then in 2003 they released MPEG-4 AVC/H.264.

In 2012 the two teams released a first draft of the High Efficiency Video Coding (HEVC) standard. This will be the successor to MPEG-4 AVC and again it should provide double the compression rate, when compared to MPEG-4 AVC [1].

[1] $\underbrace{1920 \times 1080}_{\text{pixels}} \times \underbrace{3}_{\text{byte/pixel}} \times \underbrace{25}_{\text{framerate}} \times \underbrace{3600}_{\text{seconds}} = 559.872.000$ Bytes

1.1 MPEG-4 AVC/H.264 Overview

The MPEG-4 AVC standard was created to offer a higher compression rate, but often at the cost of a higher complexity. This compression gain is achieved by many small improvements, when compared to earlier standards. Some of these changes are better motion compensation, an image segmentation into finer blocks, improved entropy encoding schemes and a mandatory deblocking filter.

It is however not a single standard, but rather a family of standards. It defines a set of compression tools, which are then used or not used, based on a selected profile. There are several profiles, designed to match the standard to the user's needs. For example the baseline profile offers a low compression rate and some error resilience, while maintaining a low complexity. The main profile, on the other hand, offers high compression gain, at the cost of a high complexity.

Based on these profiles the standard can be used in many fields. For example it is used to store videos on Blu-ray discs, but it is also used to stream videos on websites such as Youtube or Vimeo.

1.2 Structure of this paper

Section 2 will describe a general encoder and decoder pair based on the MPEG-4 AVC standard. Here the important differences will be pointed out. These are then explained in more detail in section 3. Section 4 explains the profiles and levels, which are used to define the required capabilities of a decoder. An evaluation of the standard will be given in section 5, followed by a short discussion of some patenting issues in section 6.

2 Codec overview

Before looking at the changes in detail, this section will give a high level overview of how the general encoding and decoding structure has changed in the MPEG-4 AVC standard. First the coding structure of actual data will be discussed and then a general overview of an encoder and a decoder will be shown.

2.1 Coding structure

The structure in which a video is described is similar to that of earlier standards. A video sequence is split into frames, these are dissected into slices, which are groups of macroblocks. A macroblock (MB) is, as in earlier standards, a block of 16×16 pixels. Each macroblock contains further sub-macroblocks of 16×8, 8×16 or 8×8 pixels. Unlike earlier standards, these sub-macroblocks can further be refined into 8×4, 4×8 and 4×4 blocks.

The 4×4 pixel block is the smallest pixel block and the basic encoding unit in the MPEG-4 AVC standard. In previous standards, the 8×8 blocks were the smallest encoding units.

Depending on the encoding format of a pixel, the luminance and chrominance components of a pixel change. Considering the human eye is more sensible to the luminance information, the chrominance information is stored in a sub-sampled way. For each 2x2 pixel block, each pixel has one byte of luminance information and the whole pixel block share one byte of chrominance for each chrominance channel (Cb and Cr). This is called the 4:2:0 format, which is also used by most of the earlier standards. For a macroblock this means that there are 16×16 pixels, with each one luminance values and together these pixels share two 8×8 blocks of chrominance values. Each macroblock thus consists of sixteen 4×4 blocks, containing luminance values and four 4×4 blocks, containing chrominance values for each of the two channels. This gives a total of 24 4×4 blocks per macroblock. Initially this was the only supported format, but in later updates of the standard other encoding formats were added, such as the 4:2:2 or 4:4:4 encoding formats [8]. In the 4:4:4 format, a pixel has one byte for each of the channels.

As in earlier standards, the notion of Intra-frames (I-frames), Predictive-frames (P-frames) and Bidirectional-frames (B-frames) still exists. In the MPEG-4 AVC standard the different encoding types are no longer decided on a frame level, but on a slice level. This means there are I-slices, P-slices and B-slices. I-frames, P-frames and B-frames can easily be simulated. This can be done by either just using one slice per frame, or by only using the same slice type for all the slices in a frame. Frames only containing I-slices are called instantaneous decoder refresh (IDR) frames. Once an IDR frame is decoded, no previous frames will be used for motion prediction. These frames can thus be used as random entry points.

The I-slices do not use any motion prediction and offer the least compression gain. The macroblocks in P-slices can use motion prediction, but only from one slice in the future or the past. In previous standards, P-frames were only allowed to use the previous I-frame as a references frame for motion prediction. Macroblocks in B-Slices use one slice in the future and one slice in the past for motion prediction. Unlike previous standards, it is also allowed to have both predictions in the future or the past. Furthermore B-Slices can be stored and also used for motion prediction. Macroblocks in both P-Slices and B-Slices can also be encoded without using motion prediction, if this gives better results.

2.2 Encoder

As in previous standards, the encoder is not part of the actual standard. How the resulting implementation works is left to the developer, but the resulting bitstream has to be in a format specified by the standard. Still,

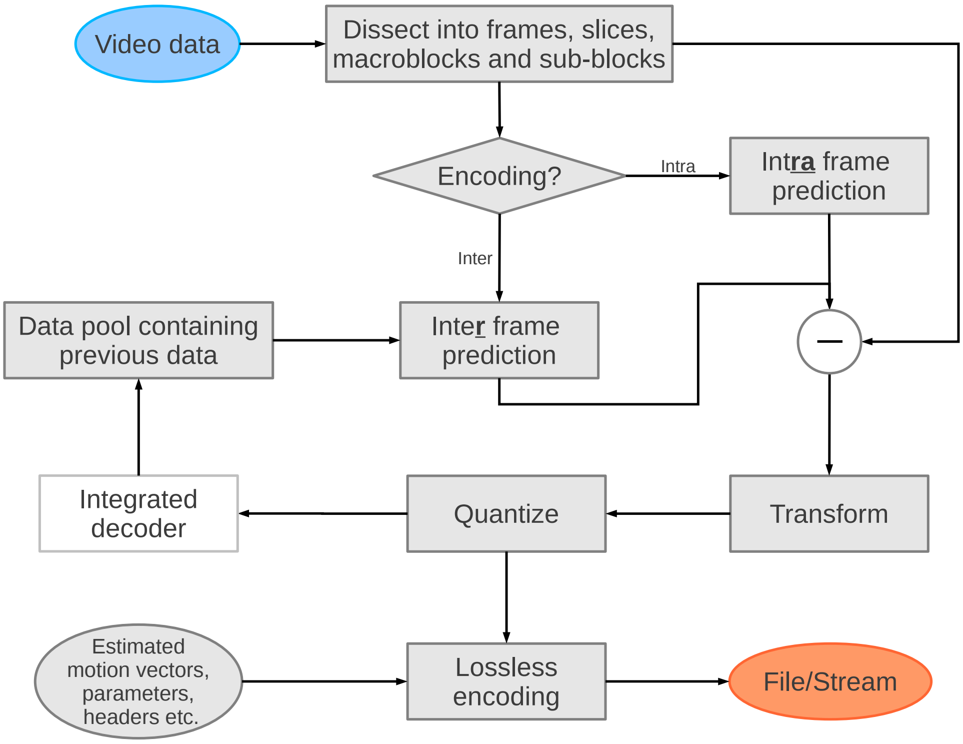

Video data
Dissect into frames, slices, macroblocks and sub-blocks
Encoding?
Intra
Intra frame prediction
Inter
Data pool containing previous data
Inter frame prediction
−
Integrated decoder
Quantize
Transform
Estimated motion vectors, parameters, headers etc.
Lossless encoding
File/Stream

Figure 1: A general MPEG-4 AVC encoder flowchart. The raw video data is first dissected down to the 4×4 blocks. For each block a prediction is created by either inter or intra frame prediction. The prediction is subtracted from the original data to obtain the residual data. This data is then transformed, quantized and encoded. Together with parameters, headers and other data which needs to be encoded in a lossless way, this results in the video file. Within the encoder there is also an integrated decoder. This decoder decodes the data, to assure both the encoder and decoder have the same data for the inter frame prediction.

the general idea of an encoder is clear, although a different implementation could be possible. The basic structure of an encoder (Fig. 1) is very similar to that of earlier standards. The video data is split into frames, slices, macroblocks and down to the 4×4 blocks. Each of these is then encoded using either intra or inter frame prediction, to get a predictions for the values based on earlier data. (Although intra frame prediction can also predict a whole macroblock at once and inter frame prediction can predict coarser segmentations of the macroblock. Since the transform only works on 4×4 blocks though, we will assume these for simplicity. The actual choice between inter and intra frame prediction is made at the macroblock level though.) The prediction is subtracted from the initial data in the block. The residual data is transformed and quantized and together with the parameters, which specify for example the prediction method, it is encoded in a lossless way. Furthermore the encoder has an integrated decoder, which is used to decode the previously encoded data. This ensures that the encoder

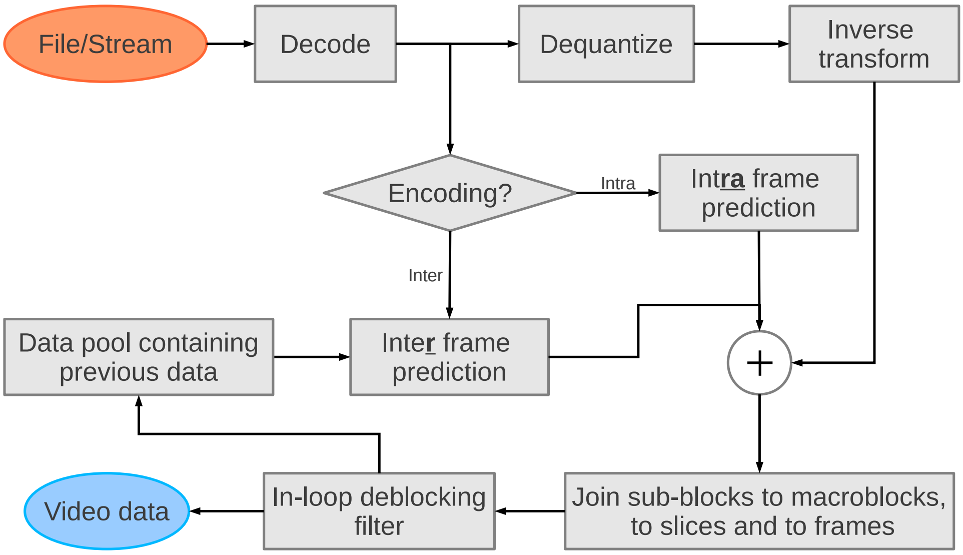


Figure 2: A general MPEG-4 AVC decoder flowchart. The encoded video data is first decoded using the correct lossless decoding technique. The obtained data is dequantized and an inverse transform is applied to obtain a residual. Based on parameters in the data, a prediction is created. The prediction is added to the residual data to obtain one decoded 4×4 block. Multiple sub-blocks are then joined together to a macroblock, which are in turn joined together to a slice and finally to a frame. The in-loop deblocking filter is applied to each of the 4×4 borders, to obtain the final video frame. Depending on the type of frame, the data is stored for later inter frame predictions.

and the decoder use the same data for motion prediction.

Compared to MPEG-4 ASP, this looks very similar. A clear difference is that the intra prediction now serves as an alternative to inter prediction. In MPEG-4 ASP, it was used after the data was transformed to predict some of those values. Apart from that, the real differences are within the components. The intra frame prediction has become more accurate and the inter frame prediction is more flexible and uses more prediction frames. The transformation is no longer a Discrete Cosine Transform (DCT), but an integer approximation of the DCT. For this approximation to work efficiently, the quantization has been adapted. Additionally there are new adaptive lossless encoding techniques, that give a better compression.

2.3 Decoder

As the decoder is part of the standard, the structure is well-defined (Fig. 2). The decoder structure is also very similar to decoders of previous standards. Considering the intra frame prediction shifted it's place in the encoder, it does as well in the decoder. Furthermore a mandatory in-loop deblocking filter is added, which is applied to each of the 4×4 block borders. This filter is one of the main reasons why the compression is better than that of earlier

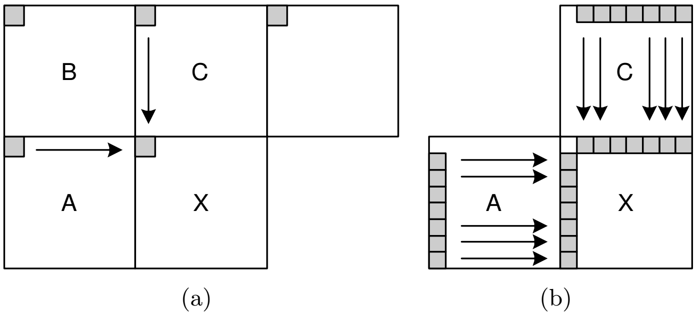


Figure 3: Intra frame prediction for an 8×8 block in MPEG-4 ASP. (a) Prediction of the DC value. It is either predicted from A or from C, depending on the gradient between the blocks B and C and the blocks B and A. (b) Prediction of the AC values. Seven AC values are predicted from the same block as the DC value was predicted from. Images from [11].

standards. The filter results in an overall better quality of the decoded image. Because of this, to achieve the same quality a stronger compression can be applied. Apart from these changes, all the components are adapted corresponding to the changes in the encoder components.

3 Main differences

As already pointed out in section 1, most of today's video encoding standards are very similar in structure. MPEG-4 AVC also has a similar structure as described in section 2, but still there are many changes when compared to earlier standards, like MPEG-2 or MPEG-4 ASP. This section will only describe some of these changes. It will focus on the those changes, that improve the compression efficiency of the standard. These changes and additions can be split up into two categories:

- **Improved prediction:** Changes in the prediction algorithms which yield a better prediction for a block of data. This means the residual error is smaller and thus there is fewer data, which needs to be encoded in a lossless way.

- **Improved coding efficiency:** Changes to the transform functions and lossless encoding techniques. These result in a better compression of the data.

3.1 Intra frame prediction

The intra frame prediction predicts the values of a block, by using previously decoded data in a frame. Typically the blocks to the left and the blocks above the current one are used for this prediction. The values in a block

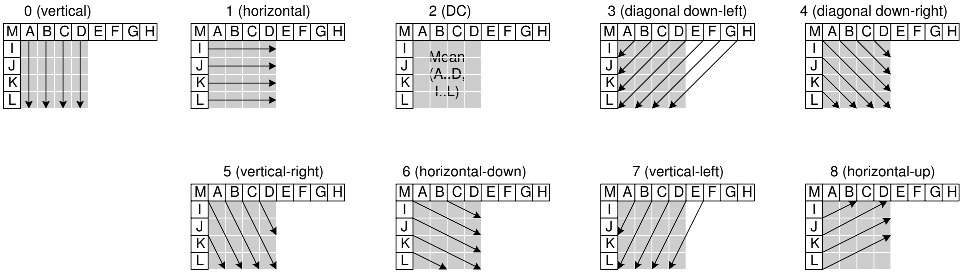


Figure 4: Nine different modes for intra frame prediction of 4 × 4 luminance blocks in MPEG-4 AVC. Here A through M stand for previously decoded entries in adjacent blocks. Image from [11].

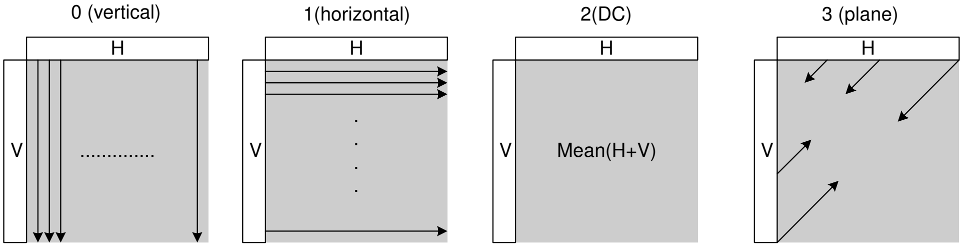


Figure 5: Four different modes for intra frame prediction of 16 × 16 luminance blocks in MPEG-4 AVC. Here H stands for the 16 luminance values of the macroblock above and V for the 16 values of the macroblock to the left. Image from [11].

can be between 0 and 255. When a good prediction is obtained, the values in the residual data block will most likely be fairly small values. By then exploiting the higher frequency of small values, the residual data blocks can be compressed in a more efficient way.

As already pointed out in section 2, when compared to MPEG-4 ASP, the intra frame prediction has been moved to a different location in the encoder and decoder. In MPEG-4 ASP the intra prediction is only used to compute eight of the values in an 8 × 8 block after the DCT was applied. This means that unless a block is predicted using inter frame prediction, there is no prediction for all of the values of a block prior to the DCT. For a block (X), the prediction is either made from the block to the left (A), or from the block above (C), as seen in Figure 3(a). If $|DC_A - DC_B|$ is larger than $|DC_A - DC_C|$, the prediction is made from A, otherwise from C. After this decision has been made, the DC value can be predicted and either seven AC values can be predicted from A, or from C as seen in Figure 3(b).

In MPEG-4 AVC, the intra frame prediction is an alternative to the inter frame prediction. If a block is part of a P-slice or a B-slice, the encoder will have to decide which prediction gives a better result. If no good motion prediction can be made, inter prediction is used. There are two types of intra prediction in MPEG-4 AVC. Both give more flexible predictions than the prediction in MPEG-4 ASP.

Each 4 × 4 luminance block can be encoded using one of nine different modes, as seen in Figure 4. The arrows indicate in which direction the predictions are made. For the horizontal or vertical mode, the data is simply copied in the given direction, based on the luminance values of the surrounding blocks. For the DC mode, the average of the values A through D and I through L is taken for all of the luminance values. For the other six modes, a linear weighted average of the values is taken based on the direction of the mode.

Next to the prediction of the 4 × 4 blocks, the whole 16 × 16 luminance values of a macroblock can be predicted at once. (See Figure 5.) Here the prediction methods are a little less flexible as for the 4 × 4 blocks. The first three modes work the same as for the 4 × 4 blocks. The fourth, the plane mode, fits a plane function to the data from the neighboring macroblocks.

Both of these prediction types are only used for the prediction of luminance values. The two 8 × 8 blocks for the chrominance values of each macroblock are predicted in a similar way as the 16 × 16 blocks.

3.2 Inter frame prediction

The main compression gain of video encoding standards is achieved by predicting the values from a similar frame, from the past or the future. Because of this, inter frame prediction is an important part of a video compression standard. Many improvements in the MPEG-4 AVC standard have been done in the inter frame prediction. These create a more accurate prediction of the actual values, at the cost of increased complexity.

Smaller block size:
Unlike earlier standards that only allowed macroblocks to be dissected into sub-blocks with a size of 8 × 8 pixels, MPEG-4 AVC allows a finer dissection into 8 × 4, 4 × 8 and 4 × 4 pixel blocks. This means that a finer segmentation of a frame can be achieved if needed and thus a better prediction can be made. Consider a moving object in front of a static background, as seen in Figure 6. In such a case, a very good prediction for the background, as well as for the object, can be made. However for a macroblock containing both the object and the background, it is not possible to make a good prediction. Either the background or the object would not be predicted correctly. If it is now possible to create a very fine dissection of a macroblock, the blocks that contain both the object and the background can be made smaller. This means there are fewer pixels with a bad prediction.

Multiple reference frames:
Depending on the encoding type of the slice a macroblock is contained in, the number of reference frames changes. If a macroblock is in a P-slice, only one prediction is allowed per sub-block. For B-slices two predictions are

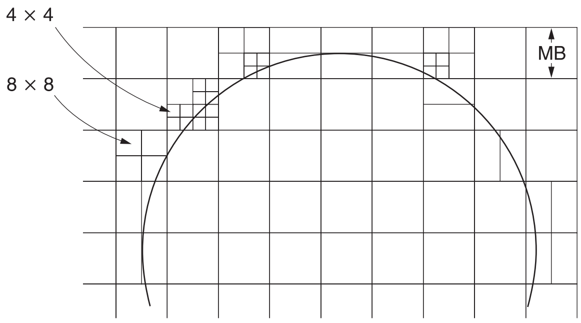


Figure 6: Possible dissection of macroblocks into sub-blocks in MPEG-4 AVC. Here the circle represents a moving object in front of a static background [14].

allowed. In earlier standards, each frame is only allowed to have one, or in the case of a B-frame, two reference frames. All macroblocks in the frame have to use these frames for the inter frame prediction. This restriction is no longer used and thus every sub-block can reference a different frame for prediction. In case of a macroblock in a P-Slice, each of the possibly 16 sub-blocks in a macroblock can reference a different frame, which means a maximum of 16 reference frames. For a macroblock in a B-Slice, this maximum is doubled to 32, as each 4 × 4 block has two predictions. This gives the option of getting a better prediction from different frames when needed. In practice, the number of reference frames is limited by the buffer size of the decoder, as this buffer needs to store the previously decoded frames for prediction.

Weighted and scaled predictions:
As blocks in B-slices have two predictions, a combination has to be made between the predictions. In earlier standards a simple average was used to create the final prediction. In MPEG-4 AVC, a weight can be assigned to both predictions. Furthermore an offset can be added to a prediction, both in B- and in P-slices. This can give a significant compression gain when encoding scenes that are fading in or out.

Quarter pixel accuracy:
In MPEG-4 AVC, a quarter pixel accurate motion prediction is mandatory. This is realized by first getting half pixel values using a 2D FIR filter. This filter calculates the value in the middle of two full pixels by taking into account the values of three full pixels at both sides of the half pixel. Using this filter, first all half pixel between two full pixels are calculated, in horizontal and vertical direction. Then the pixel in the center, of a four pixel square is calculated. This gives all the half pixel accurate values. The quarter pixel accurate values are then calculated by averaging the two adjacent values.

As the chrominance values only have half the resolution of the luminance values, these are actually predicted with one-eight pixel accurate positions using bilinear interpolation.

Using a more accurate prediction methods will give better results, but it also means there is a larger search space for good matches. This means both the complexity of the encoder and decoder increases. The encoder needs to find a match in a larger search space, but the decoder also has to calculate the additional values. In MPEG-4 ASP, quarter pixel accurate predictions are possible as well, but not mandatory. This means depending on the implementation of the encoder it might not be used.

3.3 Transform function

Considering the basic block size has changed in MPEG-4 AVC, the transform function also needs to be adapted. But other than reducing the size of the discrete cosine transform (DCT), the transform itself is also changed. MPEG-4 AVC uses an integer approximation of the 4×4 DCT. While larger block sizes would allow the transform to exploit more global correlations in the image, a smaller block size is better for local adaptivity and reduces the complexity.

The transform, which is used in MPEG-4 AVC, is called a high correlation transform (HCT) [3]. This transform approximates the DCT by two simple matrix multiplications. By decomposition of the matrix into a very simple matrix and a vector of scale factors, the resulting transform can be realized in an easy way. The matrix which needs to be multiplied twice, only contains 1's and 2's. The decomposition of the matrix is given by:

$$[HCT] = \begin{pmatrix} 1/2 \\ 1/\sqrt{10} \\ 1/2 \\ 1/\sqrt{10} \end{pmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}. \tag{1}$$

Given a 4×4 block X, the transformed values of Y are obtained by:

$$Y = [HCT] \cdot X \cdot [HCT]^T. \tag{2}$$

The scaling factors to the left in equation 1 can be left out in equation 2 and instead, it is incorporated in the quantization step. This means, the transformation step is reduced to a set of multiplication and additions of 16 bit integers. This also means, there is an exact inverse function. Depending on the implementation of float variables, the inverse DCT can actually give slightly different resulting values, than the input values given to the DCT. Since these small errors will be amplified, if they are used for prediction of a later frame, this effect can lead to worse quality when decoding videos. So not only is the complexity of the transform function reduced, it also has an exact inverse.

To further reduce the amount of data which needs to be encoded, a secondary transform is applied to the data. Each of the 16 smaller luminance blocks now have one DC value. Since the DC value represents the average color in the block, this will have a rather big value in many cases. Often the 16 DC values in one macroblock are rather similar. The DC values are joined into one 4×4 matrix. This is then transformed with a Walsh Hadammard transform (WHT)[3], which is similar to the HCT. The two chrominance blocks, which are part of a macroblock, are handled in the same way. Here each 8×8 block is split up into four 4×4 blocks as described earlier. The four DC values of these blocks are then joined together into a 2×2 block. A smaller version of the WHT is applied to these blocks. After the transform, the 16, or in case of the chrominance blocks 4, values are typically smaller. As with the values, that are obtained by applying the DCT, the values in the lower right of the matrix tend to be very small. Exploiting this, the data can be more efficiently encoded using variable length coding.

3.4 Quantization

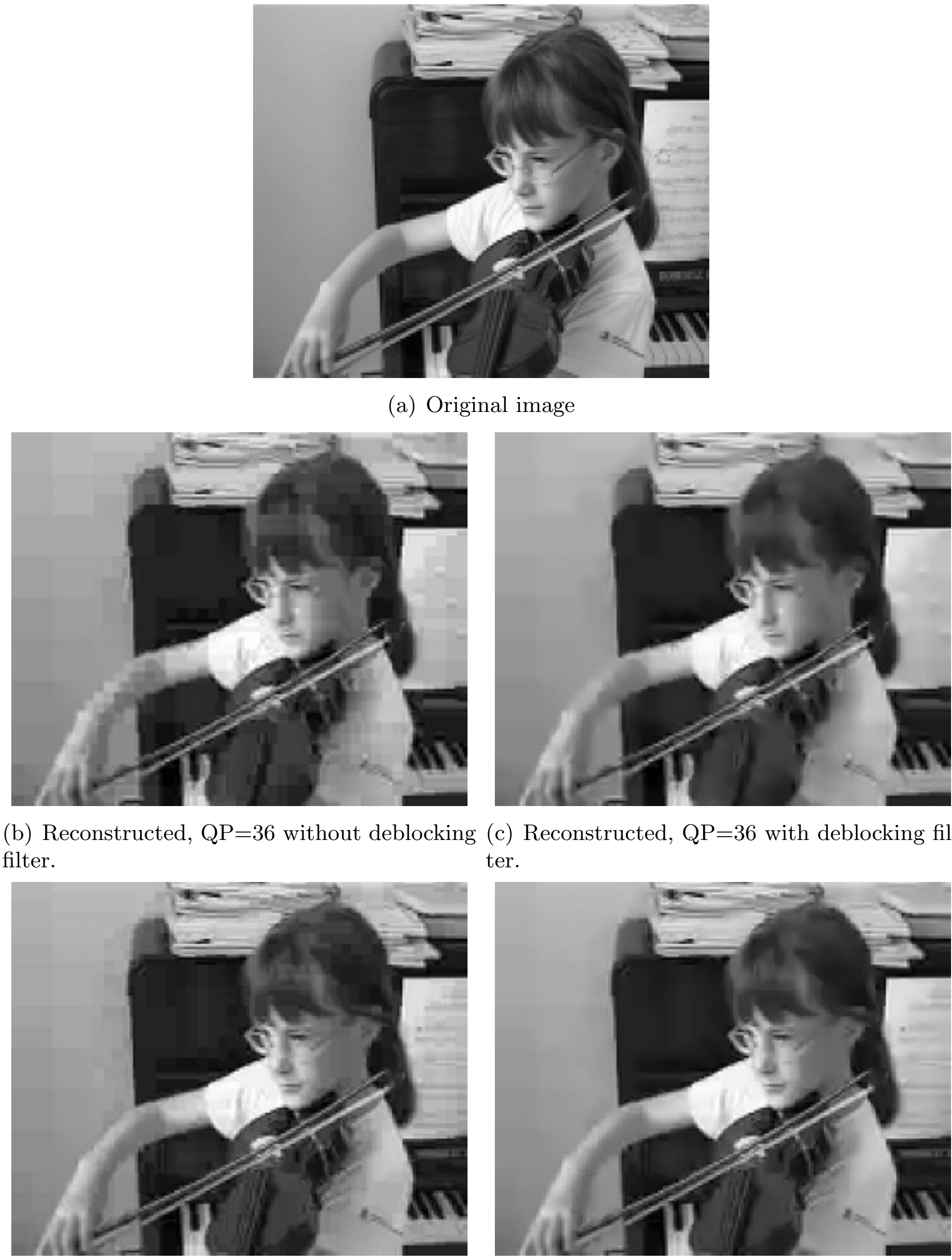
Earlier standards apply different kinds of quantization matrices, which can be applied to a resulting 8×8 block by using a scalar product. This way, each of the 64 values can be quantized with an individual value. In MPEG-4 AVC, one quantization value is used for all of the 16 values of a 4×4 block. The standard requires the quantization to be implemented without floating point operations and without divisions. Furthermore it needs to include the scaling factor, which was left out during the transformation.

Based on a desired bit-rate for the resulting encoded video and the currently achieved bit-rate, a quantization parameter (QP) is specified for each macroblock. The QP, which ranges from 0 to 51, specifies the actual quantization factor (Qstep). Qstep ranges from 0.625 to 224 and doubles each time QP increases by 6. Figure 7 (b) and (d) shows results for QP=36 and QP=32. Figure 7 (c) and (e) show the same results, but with the additional inloop-deblocking filter enabled.

Given a 4×4 block Y, after transformation, the quantization idea is the following:

$$Z_{ij} = round\left(Y_{ij} \cdot \frac{SF_{ij}}{Qstep(QP)}\right). \tag{3}$$

Here SF_{ij} stands for the scaling factor, which needs to be applied to the data, in order to complete the transformation. A very close approximation of this equation can be implemented only using integer multiplications, additions and bit shifts [11].



(a) Original image

(b) Reconstructed, QP=36 without deblocking filter.

(c) Reconstructed, QP=36 with deblocking filter.

(d) Reconstructed, QP=32 without deblocking filter.

(e) Reconstructed, QP=32 with deblocking filter.

Figure 7: Results of a decoded image with different quantization parameters (QP) and with, or without the in-loop deblocking filter enabled. Images from [11]

After this step, the data can be encoded using a lossless encoding technique. In order to encode it efficiently, the blocks are first linearised. In previous standards, different scanning orders are possible. MPEG-4 AVC uses a fixed zig-zag scanning order to linearise the block. This preferably yields chains of consecutive zeros.

3.5 Entropy encoding

For the encoding of the video data, two different lossless encoding techniques are available. Depending on the profile, either the context adaptive variable length coding (CAVLC), or the context adaptive binary arithmetic coding (CABAC) is used. The second gives a higher compression ratio at the cost of a higher complexity. Both are adaptive encoding techniques, which means by adapting to the current context, a better compression ratio can be achieved. In MPEG-4 ASP, based on the context, one of two different tables are used for encoding. While this is somewhat context adaptive, it is still fairly fixed. In MPEG2, context adaptive encoding is not available.

CAVLC:
Based on the number of non-zero values in the neighbouring blocks, one out of four variable length coding tables is picked. The smaller the amounts of non-zero values, the more the picked table will be biased towards smaller numbers. Then the number of trailing ones is counted. This means the number of consecutive 1's or -1's. Here only zeros between the ones are allowed, otherwise they are not counted as consecutive. These ones are encoded in a special way, but only the last three ones in a set of values can be encoded this way. The data is then encoded in the following way:

1. The total number of non-zero values, including the trailing ones.
2. The signs of the trailing ones, up to three per block. Listed in a reversed order.
3. The levels of the values, one entry for each non-zero value, excluding the trailing ones. Listed in a reversed order.
4. The total number of zeros.
5. The length of the zero runs before each of the non-zero values. Listed in a reversed order.

Depending on the chosen table, the actual values of these numbers might be encoded in a different way. While decoding, first the list of trailing ones is decoded, each time inserting the next one in front of the current output. Then the other non-zero values are inserted and the zero runs are inserted before the values.

Because the choice of tables is limited, this method might not always be able to adapt to the actual distribution of used values, still it gives a better encoding efficiency than only using one or two tables.

CABAC:
When computational complexity is not the main concern, CABAC can be used to achieve a higher coding efficiency. CABAC works on binary data, this means prior to encoding the linearised block, this is first converted into a binary format. This data is then encoded in a stepwise manor. Based on previously encoded data, one out of about 400 context models is chosen and based on this table, one bit is encoded. Then the frequency of 1's and 0's is updated, based on the previously encoded bit, and a new model might be chosen.

The main advantage of arithmetic encoding, compared to variable length coding, is that symbols can be coded with a higher fractional accuracy in bits. While a symbol in variable length coding is always mapped to an integer amount of bits, arithmetic coding can encode a symbol also with a fraction of bits, meaning arithmetic encoding can get a lower bit-rate per symbol. Furthermore CABAC has a higher number of adaptive models than CAVLC does. Because of this, CABAC can achieve a higher encoding efficiency at the cost of a higher complexity.

3.6 In-loop deblocking filter

The mandatory in-loop deblocking filter is one of the main changes in the MPEG-4 AVC standard. While earlier standards did already encourage post processing of the image to remove the block artefacts, it was not mandatory. The in-loop deblocking filter is applied to each edge of a 4×4 block, unless it is at an image boundary. Given a macroblock, first the deblocking filter is applied to all the vertical edge of the 4×4 blocks. Here for each block only the left vertical edge is filtered. The right one will be filtered once the adjacent block is handled. Then, the horizontal edges are filtered. As with the vertical edges, only the upper edge is filtered. An in-loop deblocking filter has several advantages:

- By moving the filter into the decoding loop, a video cannot be decoded while simply ignoring the filter. Using a simple post-processing filter at the end of the decoding process, it can easily be ignored by a decoder. This means there is no guarantee of quality after the decoding.

- There is no need for an extra frame buffer. Depending on the implementation of a post-processing step, these frames can no longer be used for prediction. Since the encoder does not know which filter is used, it can only predict from frames that are not yet filtered. In order to access these frames, a decoder will need to store both the unfiltered frames, as well as the filtered frames for displaying.

- Generally, an in-loop deblocking filter gives better quality results. This is mainly caused by the fact that an encoder can now use filtered frames for prediction and thus get better predictions.

The in-loop deblocking filter in MPEG-4 AVC filters each edge with a different strength. An encoder can decide to turn the filter off for a complete slice. If it is turned on, every edge of the macroblock within that slice is filtered using a certain filter strength based on:

- **The boundary strength:** There are five different boundary strengths. One of these is chosen, based on the type of block encoding and if the edge is also a macroblock edge or not.

- **The gradient of the block values:** Considering two samples at both side of the edge, the gradient is determined. Based on two thresholds, defined by the standard, it is checked if there is a significant change across the edge. In such a case, it is likely that there is an actual edge in the imge and thus the filter is not used.

The boundary strength is determined for a whole block, but the gradient is checked for each two pixels at the boundaries. Depending on the outcome, either zero, one or three pixels are changed at both sides of the edge. Since the output of one filter operation, is used as the input for the next filter operation, each value in a block can be changed by the deblocking filter.

While the actual compression gain of the in-loop deblocking filter is rather small, the improvement of quality is rather big. Based on this, the encoder can use a stronger quantization and thus achieve a higher compression. The quality difference resulting from an enabled or disabled deblocking filter can be seen in Figure 7.

4 Profiles and levels

As MPEG-4 AVC is not one encoding standard which is used in all cases, but rather a set of tools, the profiles define which tools are used. For example an encoder needs to decide which lossless encoding technique is used, or if B-slices are supported or not. An encoder will compress a video using a specific profile, this will then define which tools the decoder will need in order to decompress a video. A decoder may support some profiles, while it does not support others. Since the constrained baseline profile is the simplest profile, it needs to be supported by all decoders. Besides the profiles, a level specifies the required performance of a decoder.

4.1 Profiles

In the original MPEG-4 AVC standard only three profiles were defined. The baseline profile, the main profile and the extended profile. Later on the standard was extended with several other profiles, for example a series of high profiles, or the scalable profile.

- **Baseline profile:** This profile supports I- and P-slices and it uses CAVLC for entropy encoding. It targets low-delay applications, such as video conferencing or platforms with low processing power. Because of this it needs a low complexity, which also means it offers the least encoding efficiency. This was the most basic profile until later on the constrained baseline profile was defined. These two are very similar, but the baseline profile offers some extra features for data loss robustness.

- **Extended profile:** The extended profile extends the baseline profile with several error resilience techniques. It also uses B-slices and it supports interlaced video coding. This profile is targeted at streaming of video. It offers a higher compression at the cost of a higher complexity. Unlike the other profiles, this profile supports SI- and SP-slices. These are special slices designed for streaming, which allow the server to switch between different bit-rate streams when needed.

- **Main profile:** The main profile uses I-,P- and B-slices. It supports CABAC entropy encoding, but also CAVLC. It lacks some of the error resilience techniques supported by the extended profile. This was the profile to offer the highest possible quality at the highest complexity. It was targeted to be used for digital- and HDTV. Since the later introduced high profiles offered an even higher compression, this profile is now mainly used for digital non-HD television broadcast.

- **High profile(s):** This profile offers a higher compression than the others, while only slightly increasing the computation or implementation complexity [8]. It is used to store HD videos on Blu-ray discs and it is used for HDTV broadcasts. Later on several other high profiles were introduced. Some of them support a higher number of bits per sample and different pixel formats such as the 4:4:4 format.

- **Scalable profile:** This profile enables several quality video stream to be multiplexed into one[12]. This means, there is one main quality stream, with one or more subset streams. This could, for example, be used for streaming to both desktop computers and mobile platforms at the same time.

4.2 Levels

A level specifies the size of the video a decoder must be able to handle. It specifies a maximum bit-rate for the video and a maximum number of macroblocks per second. (These values are profile dependant.) Based on the number of macroblocks per second, the other parameters can be derived. There is no real limit to the frame size or frame rate. Once a certain frame size has been chosen though, the frame rate can be derived from the maximum number of macroblocks per second. The levels also specify the maximum number of stored frames, based on the total memory needed to store these frames.

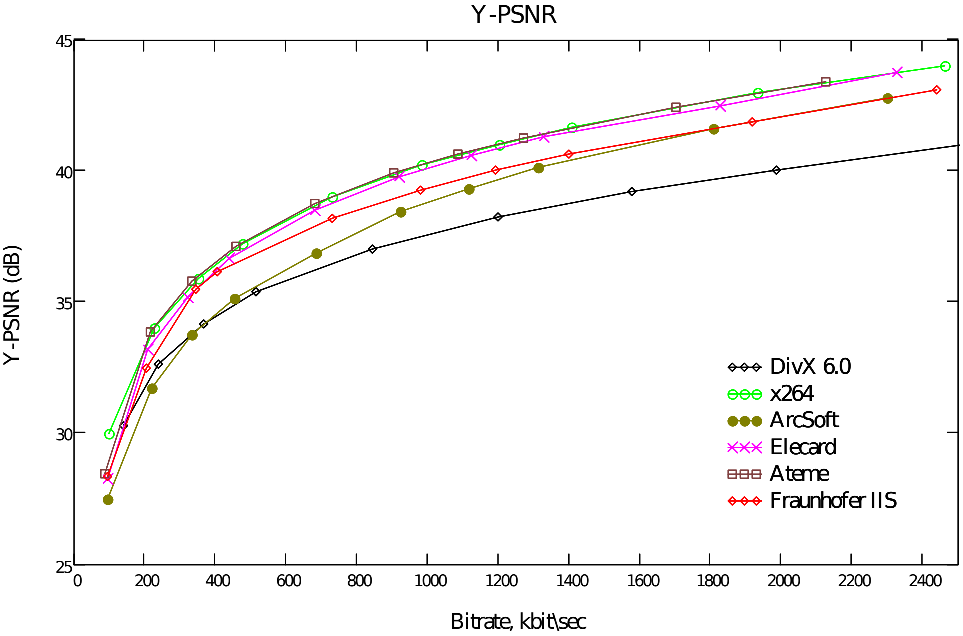


Figure 8: The PSNR for different bit-rates after encoding and decoding of a test video. The values are obtained for different implementations of the MPEG-4 AVC standard. Here only DivX 6.0 represents an implementation of the MPEG-4 ASP standard. Image from [13].

5 Evaluation

A general comparison of the MPEG-4 AVC standard can be seen in Figure 8. Here several implementations are compared to each other and to the MPEG-4 ASP standard, which is represented by the DivX 6.0 codec. The others are implementations of the MPEG-4 AVC standard. The graph shows the peak signal-to-noise ratio (PSNR) for different bit-rates. It is a metric for quality measurements, often used for videos. It describes the ratio between

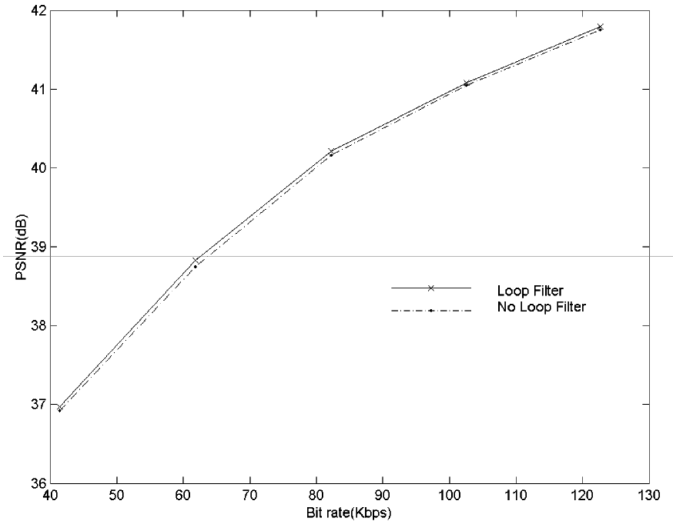


Figure 9: The difference of the PSNR when a test video is compressed with the in-loop deblocking filter turned on or off. Image from [10].

the maximal possible power of a signal and the noise. Often it is seen as good approximation of how humans perceive quality, but this is not true in all cases.

The graph shows that the different implementations of the MPEG-4 AVC standard can vary in quality at similar bit-rates. What can be seen though, is that the MPEG-4 ASP implementation performs worse than all of the MPEG-4 AVC implementations, after a certain bit-rate (∼500kbit/s). When comparing the bit-rate of the MPEG-4 ASP implementation at a given quality, to the bit-rates of the MPEG-4 AVC implementations, the claimed encoding efficiency boost can indeed be seen. For many quality levels, the MPEG-4 AVC implementations achieve a bit-rate somewhere between two to three times as low as the MPEG-4 ASP implementation can achieve, at the same quality level.

One of the changes with the most impact is the deblocking filter. As explained in section 3, the deblocking filter both increases the quality of the decoded image and it improves the results from inter frame prediction. When looking at Figure 9 though, the change in PSNR can be minimal. However, the difference of the deblocking filter can clearly be seen in Figure 7 (b)-(e). So although the PSNR changes are minimal, the perceived quality difference is fairly large. This points to the fact that human perception of quality does not always match the results of the PSNR metric. Because of this it is often hard to get an objective quality comparison. The actual change in bit-rate, based on the deblocking filter, is rather small. Disabling the deblocking filter, leads to an increase in bit-rate around 1% and in the worst case up to 2% [9].

Further experimental results in [9], show that disabling the use of sub-blocks, smaller than 8×8 pixels, results in bit-rate increases between 1% and 6%. This depends on the encoded video, the number of reference frames and the number of B-frames. This indicates, that smaller block sizes for inter frame prediction gives better results and thus it is an improvement when compared to earlier standards.

6 Patents

The MPEG-4 AVC standard builds upon a large set of patented technologies. Because of this, it is not a free standard. These patents are owned by a set of different companies, such as Microsoft or Apple. A company called MPEG LA gathered all the patents involved in MPEG-4 AVC and made contracts with the owners of the patents [2]. Because of that, they are able to sell a license for the whole patent pool, which enables you to use MPEG-4 AVC. Although the patent pool seems to be complete, there is no guarantee that it covers all the patents used in MPEG-4 AVC. They also sell licenses for other standards such as MPEG-4 ASP or MPEG2. MPEG LA is not connected to MPEG in any way though.

A license is needed for the distribution of MPEG-4 AVC decoders and encoders, but also for the distribution of videos encoded with the MPEG-4 AVC standard. When distributing encoders or decoders the first 100.000 units can be distributed for free, if more are distributed, a fee of 0.20$ has to be paid per unit. Videos are free when shorter than 12 minutes, for longer videos, 0.02$ per video has to be paid. However in 2010 MPEG LA announced that internet videos would be free forever.

These patenting issues have caused different opinions about MPEG-4 AVC. For example the HTML5 community is split into two groups. Supporters of MPEG-4 AVC and those that would rather use a free standard such as ogg Theora in HTML5. This has also taken effect on web browsers. Some support MPEG-4 AVC decoding in HTML5, while others do not.

7 Summary

MPEG-4 AVC gains better compression results, when compared to previous standards. In general, it keeps the same quality at twice the compression rate. This is done by the many changes in the standard. Most notable are:

- The improvements made to the inter frame prediction, that exploit the redundancies in video frames in a better way.

- The in-loop deblocking filter, which manages to remove the block artefacts very well.

- The context adaptive entropy encoding schemes, which improve the lossless compression rates.

As the standard is rather a set of tools, the profiles and levels can be used to adjust the standard in such a way that it fits the user's needs. Based on this, it has a wide range of possible applications. From low bit-rate streaming to high quality storage.

The standard relies on many patented technologies, this means that it might not be the best choice for everyone. When used in small scales it will be free, but for large scale commercial use a license needs to be acquired from MPEG LA.

In 2013 it will supposedly be succeeded by the HEVC standard, which again should give double the compression rate, while maintaining the same quality.

References

[1] HEVC website, August 2012. `http://hevc.hhi.fraunhofer.de/`.

[2] MPEG LA website, August 2012. `http://www.mpegla.com`.

[3] R. J. Clarke. *Transform Coding of Images.* Academic Press, Inc., 1985.

[4] ISO/IEC JTCI/SC29. *Coding of moving pictures and associated audio for digital storage media up to about 1.5Mbit/s.* ISO/IEC 11172-2, November 1992.

[5] ISO/IEC JTCI/SC29. *Generic coding of moving pictures and associated audio.* ISO/IEC 13818-2, November 1994.

[6] ISO/IEC JTCI/SC29. *Coding of audio-visual objects.* ISO/IEC 14496-2, January 2000.

[7] Joint Video Team of ITU-T and ISO/IEC JTC1. *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification.* ISO/IEC 14496-10 AVC, March 2003.

[8] Detlev Marpe, Thomas Wiegand, and Stephen Gordon. H.264/MPEG4-AVC fidelity range extensions: tools, profiles, performance, and application areas. In *ICIP (1)*, pages 593–596, 2005.

[9] Atul Puri, Xuemin Chen, and Ajay Luthra. Video coding using the H.264/MPEG-4 AVC compression standard. *Signal Processing: Image Communication*, 19(9):793 – 849, 2004.

[10] Gulistan Raja and Muhammad Javed Mirza. In-loop deblocking filter for JVT H.264/AVC. In *Proceedings of the 5th WSEAS International Conference on Signal Processing, Robotics and Automation*, pages 235–240, 2006.

[11] Iain E. Richardson. *H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia.* Wiley, 2003.

[12] Heiko Schwarz, Detlev Marpe, and Thomas Wieg. Overview of the scalable H.264/MPEG4-AVC extension. In *Proceedings of the IEEE International Conference on Image Processing, ICIP '06*, pages 161–164, 2006.

[13] Dmitriy Vatolin, Dmitriy Kulikov, Alexander Parshin, Artem Titarenko, and Stanislav Soldatov. *Second Annual MSU MPEG-4 AVC/H.264 Video Codec Comparison.* 2005.

[14] J. Watkinson. *The MPEG Handbook*. Focal Press, Woburn (MA), USA, 2001.