VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

METHODS AND TOOLS FOR IMAGE AND VIDEO QUALITY ASSESSMENT

METODY A PROSTŘEDKY PRO HODNOCENÍ KVALITY OBRAZU

DISERTAČNÍ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE AUTHOR	Ing. MARTIN SLANINA
VEDOUCÍ PRÁCE SUPERVISOR	prof. Ing. VÁCLAV ŘÍČNÝ, CSc.

BRNO 2008

Licenční smlouva
poskytovaná k výkonu práva užít školní dílo

uzavřená mezi smluvními stranami:

1. Pan

Jméno a příjmení:	Ing. Martin Slanina
Bytem:	Zikova 628/15, 77900, Olomouc - Nové Sady
Narozen (datum a místo):	15.4.1982, Moravská Třebová

(dále jen autor)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 244/53, 60200 Brno 2
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. RNDr. Vladimír Aubrecht, CSc.

(dále jen nabyvatel)

Čl. 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

   ☒ disertační práce

   ☐ diplomová práce

   ☐ bakalářská práce

   ☐ jiná práce, jejíž druh je specifikován jako ....

   (dále jen VŠKP nebo dílo)

Název VŠKP:	Methods and Tools for Image and Video Quality Assessment
Vedoucí/ školitel VŠKP:	prof. Ing. Václav Říčný, CSc.
Ústav:	Ústav radioelektroniky
Datum obhajoby VŠKP:	neuvedeno

   VŠKP odevzdal autor nabyvateli v[1]:

   ☐ tištěné formě — počet exemplářů 3

   ☐ elektronické formě — počet exemplářů 3

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

[1] hodící se zaškrtněte

Abstrakt

Disertační práce se zabývá metodami a prostředky pro hodnocení kvality obrazu ve videosekvencích, což je velmi aktuální téma, zažívající velký rozmach zejména v souvislosti s digitálním zpracováním videosignálů. Přestože již existuje relativně velké množství metod a metrik pro objektivní, tedy automatizované měření kvality videosekvencí, jsou tyto metody zpravidla založeny na porovnání zpracované (poškozené, například komprimací) a originální videosekvence. Metod pro hodnocení kvality videosekvení bez reference, tedy pouze na základě analýzy zpracovaného materiálu, je velmi málo. Navíc se takové metody převážně zaměřují na analýzu hodnot signálu (typicky jasu) v jednotlivých obrazových bodech dekódovaného signálu, což je jen těžko aplikovatelné pro moderní komprimační algoritmy jako je H.264/AVC, který používá sofistikovené techniky pro odstranění komprimačních artefaktů.

V práci je nejprve podán stučný přehled dostupných metod pro objektivní hodnocení komprimovaných videosekvencí se zdůrazněním rozdílného principu metod využívajících referenční materiál a metod pracujících bez reference. Na základě analýzy možných přístupů pro hodnocení video sekvencí komprimovaných moderními komprimačními algoritmy je v dalším textu práce popsán návrh nové metody určené pro hodnocení kvality obrazu ve videosekvencích komprimovaných s využitím algoritmu H.264/AVC.

Nová metoda je založena na sledování hodnot parametrů, které jsou obsaženy v transportním toku komprimovaného videa, a přímo souvisí s procesem kódování. Nejprve je provedena úvaha nad vlivem některých takových parametrů na kvalitu výsledného videa. Následně je navržen algoritmus, který s využitím umělé neuronové sítě určuje špičkový poměr signálu a šumu (peak signal-to-noise ratio – PSNR) v komprimované videosekvenci – plně referenční metrika je tedy nahrazována metrikou bez reference. Je ověřeno několik konfigurací umělých neuronových sítí od těch nejjednodušších až po třívrstvé dopředné sítě. Pro učení sítí a následnou analýzu jejich výkonnosti a věrnosti určení PSNR jsou vytvořeny dva soubory nekomprimovaných videosekvencí, které jsou následně komprimovány algoritmem H.264/AVC s proměnným nastavením kodéru.

V závěrečné části práce je proveden rozbor chování nově navrženého algoritmu v případě, že se změní vlastnosti zpracovávaného videa (rozlišení, prokládání, střih), případně kodéru (počtu snímků kódovaných v jedné skupině, formát skupiny). Chování algoritmu je analyzováno až do plného vysokého rozlišení zdrojového signálu (full HD – 1920×1080 obrazových bodů).

Abstract

The doctoral thesis is focused on methods and tools for image quality assessment in video sequences, which is a very up-to-date theme, undergoing a rapid evolution with respect to digital video signal processing, in particular. Although a variety of metrics for objective (automated) video sequence quality measurement has been developed recently, these methods are mostly based on comparison of the processed (damaged, e.g. with compression) and original video sequences. There are very few methods operating without reference, i.e. only on the processed video material. Moreover, such methods are usually analyzing signal values (typically luminance) in picture elements of the decoded signal, which is hardly applicable for modern compression algorithms such as the H.264/AVC as they use sophisticated techniques to remove compression artifacts.

The thesis first gives a brief overview of the available metrics for objective quality measurements of compressed video sequences, emphasizing the different approach of full-reference and no-reference methods. Based on an analysis of possible ideas for measuring quality of video sequences compressed using modern compression algorithms, the thesis describes the design process of a new quality metric for video sequences compressed with the H.264/AVC algorithm.

The new method is based on monitoring of several parameters, present in the transport stream of the compressed video and directly related to the encoding process. The impact of bitstream parameters on the video quality is considered first. Consequently, an algorithm is designed, employing an artificial neural network to estimate the peak signal-to-noise ratios (PSNR) of the compressed video sequences – a full-reference metric is thus replaced by a no–reference metric. Several neural network configurations are verified, reaching from the simplest to three-layer feedforward networks. Two sets of video sequences are constructed to train the networks and analyze their performance and fidelity of estimated PSNRs. The sequences are compressed using the H.264/AVC algorithm with variable encoder configuration.

The final part of the thesis deals with an analysis of behavior of the newly designed algorithm, provided the properties of the processed video are changed (resolution, cut) or encoder configuration is altered (format of group of pictures coded together). The analysis is done on video sequences with resolution up to full HD (1920 $\times$ 1080 pixels, progressive)

Klíčová slova

H.264/AVC, MPEG-4 Part 10, umělá neuronová sít', kvalita videa, objektivní hodnocení, metrika pro hodnocení kvality, PSNR

Keywords

H.264/AVC, MPEG-4 Part 10, artificial neural network, video quality, objective assessment, quality metric,PSNR

SLANINA, M. *Methods and Tools for Image and Video Quality Assessment.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 80 s. Vedoucí dizertační práce prof. Ing. Václav Říčný, CSc.

Declaration

I hereby declare this thesis describes my own research and I composed the thesis entirely myself.

Martin Slanina
Brno, August 14th 2008

Acknowledgements

First of all, I would like to thank my supervisor, professor Václav Říčný, for his patience, recommendations, suggestions, and generous help. A big "Thank you" also belongs to everyone I had the chance to consult my thesis with, among these at the first place professor Robert Forchheimer from Sweden, who actually helped me start getting a certain direction in the research. Spending two months at his institution was an invaluable professional and personal experience. Thank you, Robert, and everyone involved in my stay.

It was not only the professional support that I needed to accomplish my thesis. A huge support from my parents was invaluable in many aspects. Thank you.

Contents

Introduction 1

In the days that *analog* signal's form was dominant in video processing, scanning, transmission and recording systems, determining video quality was not really a challenging issue. Several simple features sufficed to measure the overall video quality [10] . Among these measurements, let us mention the frequency response, conveniently measurable in transmission channels using special lines in television broadcast signals, signal to noise ratio, etc. Several test signals could easily be used to determine the overall performance of the whole system. With the recent advent of digital video processing systems, these measures are no longer usable. For *digital* video systems, the quality evaluation methods must be changed. Performance of a digital video processing (or transmission) system can vary significantly, depending on the actual video content.

Since its outset, commonly dated to be around the year 1950, digital video image coding research has been growing constantly. As uncompressed digital video samples represent a huge amount of data, an obvious goal of researchers ever since the digital video processing came into play was to reduce the statistical and perceptual redundancies of the video image data to either comply with a certain storage or communications bandwidth restrictions with the best possible picture quality, or to provide a certain picture quality service with the lowest possible bandwidth used.

In lossless compression systems, only the statistical redundancy of the digital video data is reduced. This approach has the advantage of not reducing the visual quality of the video content (compared to the uncompressed *digital* original), however the resulting bit rate reduction of such systems is often insufficient. This is why lossy compression techniques overwhelm in present-day systems. Lossy compression always introduces compression artifacts – encoder-specific degradations of the video image, e.g. block artifacts and staircase effect for DCT-based techniques (such as MPEG-1, MPEG-2, H.261, H.263), blur (for MPEG-1, MPEG-2, MPEG-4 – H.264/AVC), etc. Although these artifacts do not necessarily have to be noticed by the observer, it is always a challenging task to determine where the bound of visibility lies. Anyway, it is not only video compression that introduces video image quality degradation. With digital video scanning, recording and transmission systems, different types of errors may occur. Depending on the system properties and error types, these errors may have different impact on the resulting perceived video quality.

As human observer is the target consumer of the video content, the one and only perfect quality measure is always the observer's opinion on the perceived video quality. However, human observers can hardly be used every time quality needs to be evaluated. Human observers are used in so-called *subjective* tests. Recent research in digital video quality measurement methods is aiming to develop algorithms, which are able to estimate subjective results automatically. These computational techniques are also referred to as the *objective* quality tests. There has been a great deal of research in the area of objective quality assessment methods. However, no such method has been developed so far to fully substitute the objective testing in terms of performance.

In Chapter 2 of this doctoral thesis, the contemporary objective methods and the corresponding subjective tests are briefly summarized. The state-of-the-art in digital video quality is described and the pros and cons of the available metrics are discussed. The last section of Chapter 2 defines the objectives of the doctoral thesis. Chapter 3 describes a new metric for no-reference video quality assessment, starting with the theoretical background needed for the understanding of the new solution in Sections 3.1 and 3.2. From Sec. 3.3 on, the novel solution and the actual original contribution of the thesis is dealt with. Chapter 4 describes the set of video sequences that were used for the design and performance testing of the newly designed algorithm. Chapter 5 deals with performance analysis of the algorithm when the characteristics of the inputs are changing. The doctoral thesis concludes in Chapter 6.

State of Knowledge 2

As mentioned in Chapter 1, the perfect video quality measure is the observer's opinion on the perceived quality. Although a minor research is still active in the area of *subjective* quality [19], these methods are quite well understood and standardized. However, they will be briefly mentioned in this text as they are used as a benchmark for the objective tests. As such, their understanding is necessary, even though my research is oriented in *objective* quality assessment methods.

2.1 Subjective Quality Assessment

The routines for subjective image and video quality assessment are formalized in recommendation ITU-R BT.500-11 [11] and ITU-T P.910 [13]. In these recommendations, the viewing conditions are described, the material and observer selection, assessment routines and data analysis methods are defined. While the ITU-T P.910 is intended for multimedia applications, the ITU-R BT.500-11 should be used for television system quality assessment.

Basically, there are two manners in which the subjective tests may be defined: with *single stimulus* methods, only one image or one video sequence is available for quality rating, i.e. no original is given for comparison. The *double stimulus* methods give access to an "original" or undistorted material, and thus comparison is possible.

2.2 Full-Reference Objective Methods

The *full-reference* (FR) objective quality assessment methods have one idea in common. For the quality evaluation, an original material is always provided for comparison. Obviously, this is the feature of *double stimulus* subjective methods, which may be used as a benchmark of FR objective metrics.

Pixel based metrics

The first method to be mentioned here is, of course, the *peak signal-to-noise ratio* (PSNR). It is a very simple quality measure, given by [47]

$$\mathrm{PSNR} = 10\log_{10}\frac{m^2}{\mathrm{MSE}}, \qquad [dB] \tag{2.1}$$

where m is the maximum value a pixel can take and MSE is the *mean squared error*, given by [47]

$$\mathrm{MSE} = \frac{1}{\mathrm{T}\cdot\mathrm{M}\cdot\mathrm{N}}\sum_{k=1}^{\mathrm{T}}\sum_{i=1}^{\mathrm{M}}\sum_{j=1}^{\mathrm{N}}[f(k,i,j)-\tilde{f}(k,i,j)]^2, \qquad [-] \tag{2.2}$$

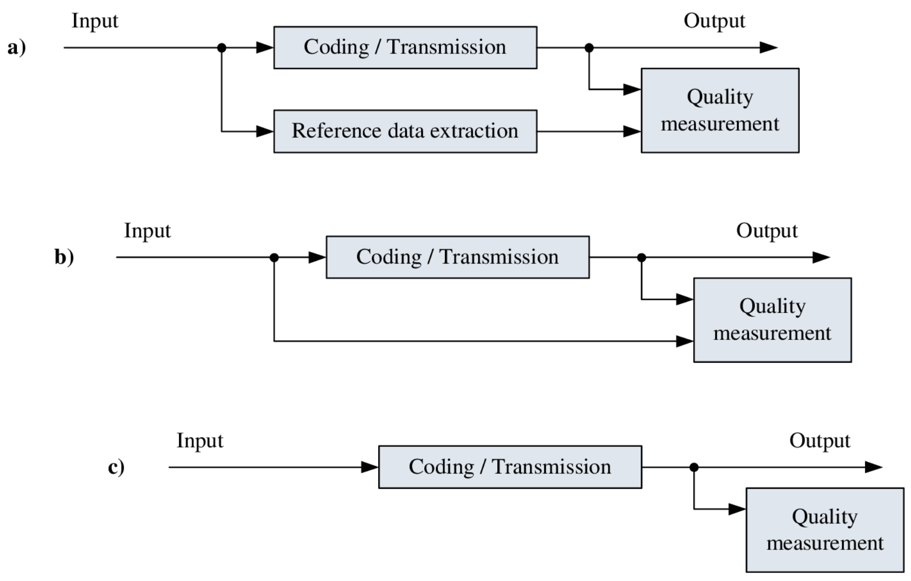

Figure 2.1: Quality metrics operating with a) full reference, b) reduced reference, c) no reference.

for a video sequence consisting of T frames of M × N pixels. The symbols $f(k,i,j)$ and $\tilde{f}(k,i,j)$ represent the luma pixel values of the original and the distorted video, respectively.

Human visual system modeling

The PSNR, yet widely used for its simplicity, is often criticized for not correlating well with the subjective tests [22]. More sophisticated methods have been developed, using different approaches. In Fig. 2.2, a framework of one such approach is shown. It is called the *error sensitivity approach*. This framework covers some of the basic features of the *human visual system* (HVS) and is used in many quality assessment methods [3, 21, 34, 42, 43, 46, 47]. The input of such system is created by the luma pixel values of both the original and the degraded image or video. Please note some of these metrics are originally designed for static images only, but can easily be used for video sequences when applied frame-by-frame [3, 21, 34, 46]. In the *preprocessing* block, usually color space conversion and gamma correction is performed. After this, one of the human visual system features is modeled - the *contrast sensitivity function* (CSF). In the *channel decomposition* block, usually a bank of filters is used to divide the data into different frequency bands (orientation is very often considered as well). In each of these bands, *masking* – another typical feature of the HVS – is simulated. At the end, the error data is pooled to form a quality measure.

The objective quality metrics that use human visual system modeling are in their nature very universal. They do not need any prior knowledge about the distortions within the video image. Consequently, they do not need any information about the video processing, recording or transmission system, which introduces video quality degradation. However, this universality requires a considerably high price to be paid: an original for comparison has to be provided and the human visual system must be understood well enough. Unfortunately, the HVS is so complex that we are not yet able to capture all of

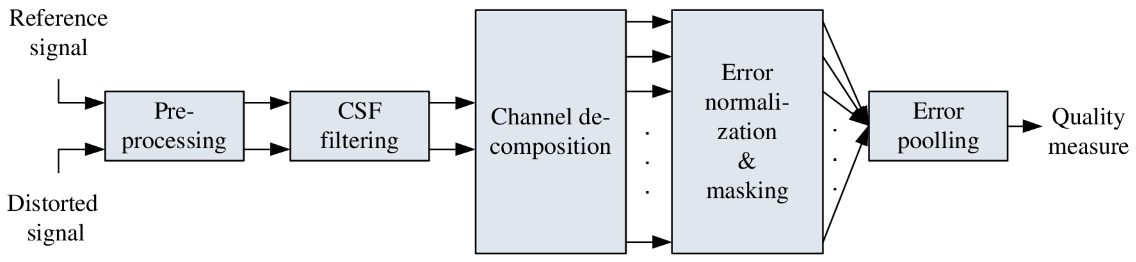

Figure 2.2: A framework of error sensitivity based quality metrics.

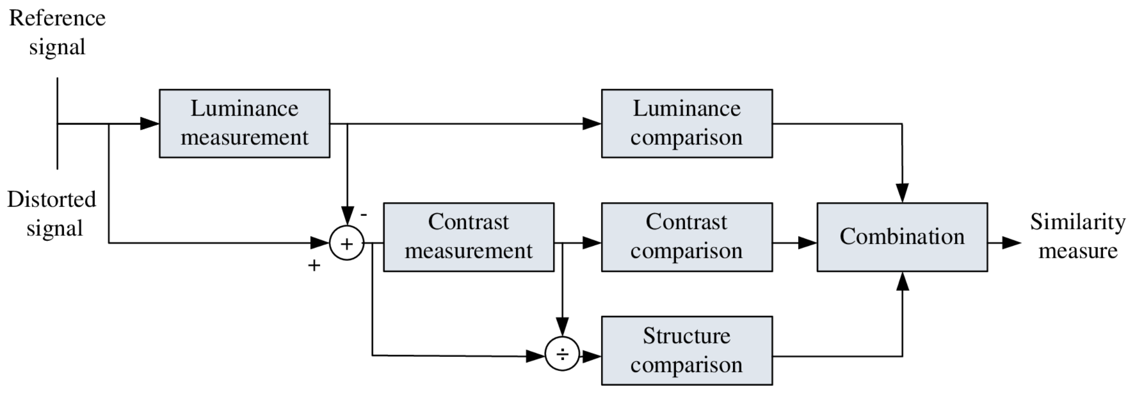

Figure 2.3: Diagram of the SSIM system.

its features. Any progress in such modeling is closely linked to vision research. No HVS metric has yet been developed to satisfy the needs of FR quality assessment – subjective tests are still used for system performance evaluation (see Section 2.5).

Structural similarity

A different approach was presented in [41], which compares the structure of the respective images, rather than simulating the responses of the HVS. The scheme of such system is shown in Fig. 2.3 [41, 49]. In principle, three measures are compared for the two images: luminance (represented by pixel mean luma value), contrast (represented by standard deviation) and structure. This metric, as shown in [26], performs very well compared to the simplest pixel based metric. However, the correlation with subjective tests is still not high enough.

2.3 Reduced-Reference Objective Methods

The *reduced-reference* (RR) metrics lie half-way between the FR and *no-reference* (NR) metrics. In principle, some features are extracted from the original image/video, transmitted along with the signal and used for the quality measurement at the receiver end. The bottleneck of these methods is in that additional data must be stored or transmitted, which requires changes in the way video data is stored. Of course, RR metric can be used as FR metrics, with the only difference in data they use for comparison. An example of one such metric can be found in [48].

2.4 No-Reference Objective Methods

The *no-reference* video quality assessment metrics cannot rely on any information about the original material. What information is then available at the receiver side and can be used for measurement? Usually, no-reference metrics use some a-priori information about the processing system. For example, a usual DVB-T broadcasting system using MPEG-2 source coding is known to have the block artifacts as the most annoying impairment [6]. Tracking these artifacts down in the video image may provide enough information to judge the overall quality.

No-reference analysis using pixel values

The block artifact detection approach is used in *DVQ analyzer* – video quality measurement equipment supplied by Rohde & Schwarz and is briefly described in [6]. The principle of the method is in assumption that block artifacts create a regular grid with constant distances. Neighboring pixel differences are computed for the whole image and averaged in such manner that only 16 values remain (since MPEG-2 is supposed to create 16×16 blocks). If the average pixel value difference is significantly larger on block boundaries, a statement can be made that block artifacts are present in the image. Such process was implemented and described in [27]. In [18], a no-reference algorithm was presented capable of detecting block artifacts in a block-by-block manner and, as an extension, detects flatness of the image. A no-reference block and blur detection approach is introduced in [9], designed to measure quality of JPEG and JPEG2000 images. Another no-reference algorithm for block artifact detection was described in [40], extending the conventional approach with masking effect implementations.

Another common distortion, blur, can also be used for quality evaluation. Of course, depending on the characteristics of the processing system (whether or not the system is likely to introduce blur). An interesting metric was presented in [17]. A very similar approach is also used in [14]. In principle, these metrics analyze how steep the changes in pixel values within the line are. The main difference is that [17] analyzes not only the horizontal direction, but measures blur in four directions instead.

A interesting no-reference approach was used in [35], using a learning algorithm to assess the overall quality of an image. The metric uses pixel values of a decoded picture, which was subject to JPEG or JPEG2000 compression.

No-reference analysis using transform coefficients and encoded stream values

In [23], a metric was presented for JPEG2000 compressed static images. The JPEG2000 standard uses wavelet transform. The authors analyze the wavelet coefficients to gain a quality measure. An observation was made that in natural images, these coefficients have some characteristic properties. If the wavelet coefficients do not behave in a desired manner, a quality degradation can be expected. However, this metric is only applicable for wavelet transform compressed images, and thus not applicable for any of the wide-spread present-day video compression standards.

Anyway, coefficient analysis for video sequences is also possible. In [7], such analysis was performed for MPEG-2 compressed video sequences. First of all, a statistical distribution analysis was performed to say which of the features available in the MPEG-2 transport stream may be used for evaluation. Over twenty features are then used to feed an artificial neural network for learning and consequently for quality evaluation. For this metric, a correlation as high as 0.85 was achieved by the authors.

A different approach was published recently [5], where the authors compute PSNR values of a H.264/AVC (see Sec. 3.1) using transform coefficient and quantization parameter values, which means computation can be done on the encoded bit stream only. In fact, it is a similar approach as what I am working on. However, I use a different set of parameters and a different means of computation. It is worth mentioning that the framework of the system described in Chapter 3 [26] was published earlier than [5].

2.5 Standardization Efforts

To establish standards in the field of video quality assessment, *Video Quality Experts Group* (VQEG) was formed in 1997. The majority of participants are active in the International Telecommunication Union (ITU). VQEG combines the expertise and resources found in several ITU Study Groups to work towards a common goal.

So far, two phases of tests were performed to define and recommend procedures for *full-reference* quality assessment. As Phase I (2000) was completed with limited success [37], Phase II tests were conducted in 2003 [38]. In this phase, out of seven proposed FR metrics, at least five were stated to be performing reasonably well (average correlation with subjective test scores as high as 0.91).

The VQEG will continue its work with testing of HDTV and *Multimedia* video sequences. Furthermore, *reduced-reference* and *no-reference* tests are planned. However, for all of these test, only test plans have been defined so far [39].

2.6 Aims of Dissertation

The goal of the doctoral thesis is to bring a new approach to objective digital video quality assessment. Lots of research has already been done in full-reference and reduced-reference quality assessment. Originally, my research was oriented in evaluation of the performance of existing metrics [26, 27, 24, 25]. However, not enough information to implement the existing metrics is often available. In addition, such work has already been done by the VQEG (see Chapter 2.5).

Anyway, the reduced-reference objective methods are still not very well defined and understood, especially for the emerging video compression standards. The objectives of the doctoral thesis can thus be stated as follows:

- develop a framework of a new metric suitable for no-reference objective video quality assessment of compressed video conforming to one of the most recent video compression standards – the H.264/AVC [12]. Design in detail and implement a metric capable of replacing a full-reference metric with a no-reference approach,
- construct a video sequence database for metric performance evaluation,
- performance analysis methodology. Analyse the performance of the new metric for different encoder configurations and different video sequence characteristics.

2.7 Conclusion

We have made an overview of the available video quality metrics and defined the state-of-the-art in digital video quality assessment. An evaluation of some of the available metrics was published in [24, 26, 27, 25]. Based on this, the aims of the doctoral thesis were defined.

A New Metric for H.264/AVC. 3

3.1 The H.264/AVC Video Encoding

The “Advanced Video Coding” standard is known as ITU-T Recommendation H.264 [12, 44] and as ISO/IEC 14496 (MPEG-4) Part 10[1]. These two documents are formally identical, as the standard was developed by a Joint video team (JVT) of ISO/IEC Motion picture experts group (MPEG) and the ITU-T Video coding experts group (VCEG) [45]. These two groups have a very fertile background in the development of video coding techniques – let us mention at least the MPEG-1 and the very successful MPEG-2 by the MPEG group, vastly used today in digital television broadcasting, DVD-video, etc., and, on the other hand, the VCEG’s H.261 videoconferencing standard and its more efficient successor H.263. The Joint Video Team efforts concluded in 2003, when the H.264/MPEG-4 Part 10 standard was first published. The standard will be noted as the H.264/AVC throughout the remaining text of this thesis.

The H.264/AVC was designed to cope with a broad range of applications, such as broadcasting (cable, satellite, terrestrial, DSL), video storage, conversational services, video-on-demand streaming, multimedia messaging services [20]. Obviously, for such a diverse set of applications, the range of bit rates must be correspondingly broad to cover applications from mobile device video (like DVB-H, for instance) to HDTV video broadcasting. The system is designed as flexible to meet such requirements.

In the following sections, the basic principles of the H.264/AVC will be described to provide a ground for a description of the new quality metric

3.1.1 The data format

Basically, all the coding of the H.264/AVC is divided into two layers: the *video coding layer* (VCL) and the *network abstraction layer* (NAL) [12]. The video encoding itself is all done in the VCL, and the output is VCL data. This data is consequently mapped into NAL units, containing a header and a payload - the RBSP (raw byte sequence payload). The purpose of such separation into two layers is such that we can easily make a distinction between the coding-specific features and procedures and the way the encoded data is treated afterwards (storage, transmission, etc.).

The advantage for our purpose of designing a quality assessment algorithm is then such that we can completely look away from the NAL operations (we do not make analysis of erroneous bit streams) and only focus on the VCL itself.

[1]The MPEG-4 is describing much more than only the MPEG-4 Part 10. A brief list of its current Parts can be found in Table 3.1. The parts 23 to 25 are not yet finished (December 2007).

Table 3.1: Parts of the MPEG-4 (ISO/IEC 14496) standard. [45]

Part No.	**Description**
Part 1	Systems: Scene description, multiplexing of audio, video and related information, synchronisation, buffer management, intellectual property management.
Part 2	Visual: Coding of 'natural' and 'synthetic' visual objects.
Part 3	Audio: Coding of natural and synthetic audio objects.
Part 4	Conformance Testing: Conformance conditions, test procedures, test bit streams.
Part 5	Reference Software: Publicly-available software that implements most tools in the Standard.
Part 6	Delivery Multimedia Integration Framework:Asession protocol for multimedia streaming.
Part 7	Optimised Visual Reference Software: Optimised software implementation of selected Visual coding tools. This Part is a Technical Report (and not an International Standard).
Part 8	Carriage of MPEG-4 over IP: Specifies the mechanism for carrying MPEG-4 coded data over Internet Protocol (IP) networks.
Part 9	Reference Hardware Description: VHDL descriptions of MPEG-4 coding tools (suitable for implementation in ICs). This Part is a Technical Report.
Part 10	*Advanced Video Coding*: Efficient coding of natural video. International standard.
Part 11	Scene description and Application Engine.
Part 12	ISO Base Media File Format.
Part 13	Intellectual Property Management and Protection Extensions.
Part 14	MPEG-4 File Format.
Part 15	AVC File Format.
Part 16	Animation Framework Extension.
Part 17	Timed Text subtitle format.
Part 18	Font Compression and Streaming.
Part 19	Synthesized Texture Stream.
Part 20	Lightweight Scene Representation.
Part 21	MPEG-J Graphical Framework eXtension (GFX).
Part 22	Open Font Format Specification (OFFS) based on OpenType.
Part 23	Symbolic Music Representation.

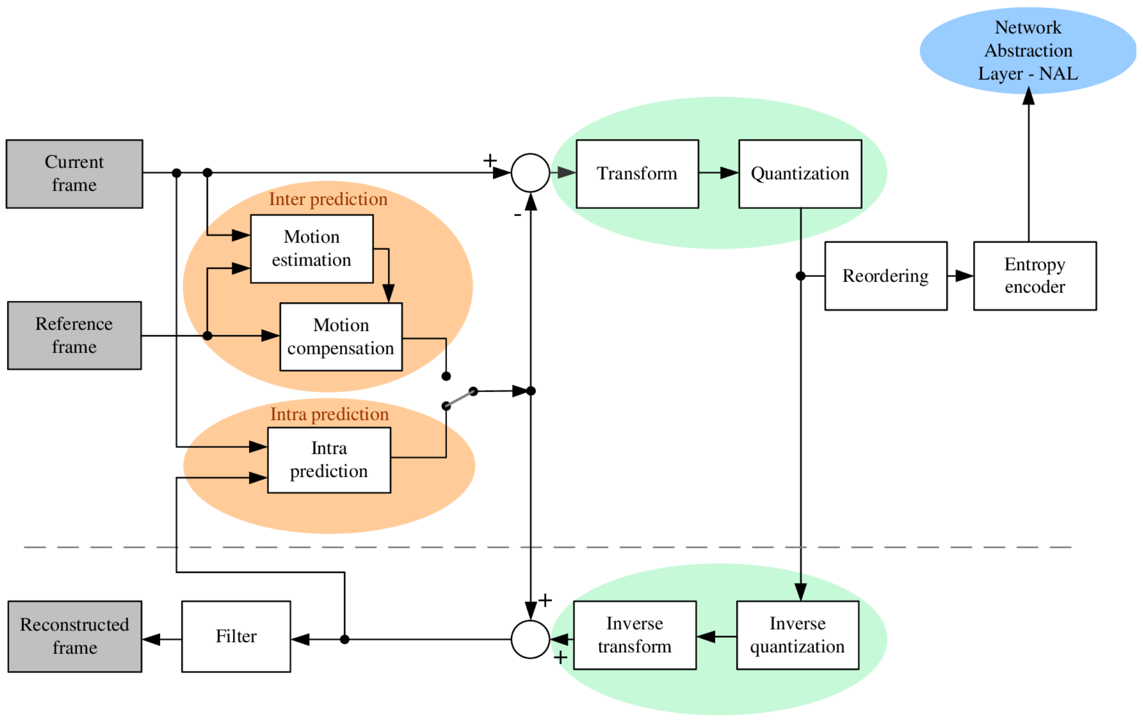

Figure 3.1: H.264/AVC encoder block diagram.

3.1.2 The encoding process

Before we move on to detailed description of the encoder features that we use for quality assessment, it is necessary to briefly describe the function of the encoder / decoder of H.264/AVC as a whole. Please note that the standard only defines the syntax and semantics of the conforming bit streams, along with instructions on the decoding process [12, 20]. The encoder and decoder structure is not strictly given and can vary in different implementations. The structure described in the following paragraphs is, however, very likely to be common for most of the implementations of the standard.

Let us have a look at the encoder structure in Fig.3.1 (the figure is taken from [20] with modifications). We can observe two paths in the encoder: a “forward path” – from left to right, and a “reconstruction path” – from right to left. The paths are divided with a dashed line in Fig. 3.1. The following description is based on an explanation given in [20]. We will only describe the inputs as frames, however when processing interlaced video, fields may be encoded instead of frames.

Encoder

Now consider the *forward path* of the encoder. The frames are not encoded as a whole – blocks of 16 by 16 pixels within them (macroblocks) are processed one by one (see Section 3.1.3). For each macroblock, a prediction P is formed first. To achieve this, the encoder must decide whether the macroblock will be Intra predicted or Inter predicted (more in Sections 3.1.4 and 3.1.5, respectively). The main difference is what data is used for prediction: Inter predicted macroblocks are derived using pixels in different frames (one such frame is depicted in the figure as Reference frame), whereas Intra predicted

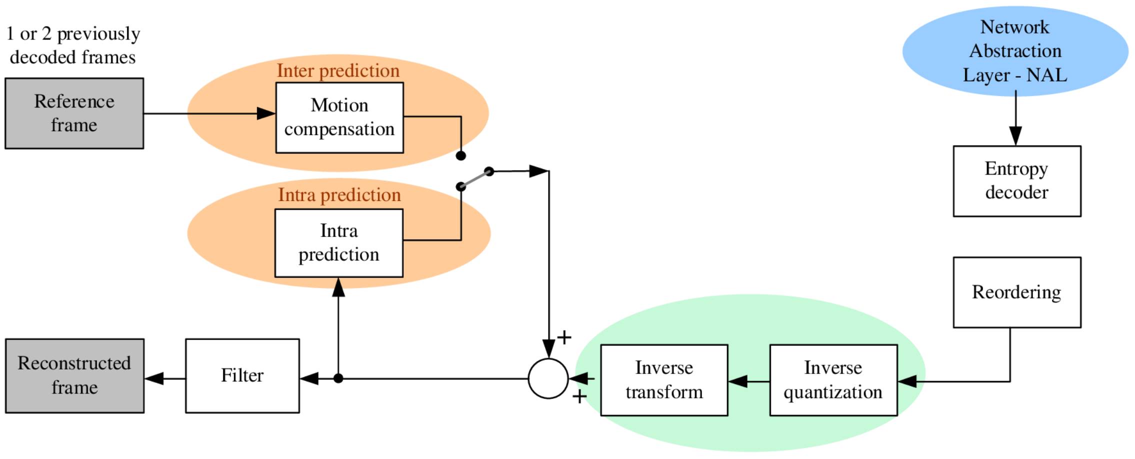

Figure 3.2: H.264/AVC decoder block diagram.

macroblocks can only use neighboring pixels within the same frame (or, more precisely, take advantage of pixel data of an encoded and subsequently decoded neighbors within the frame being processed – notice in Fig. 3.1 that the reference is taken after going through the forward path as well as the reconstruction path).

After forming the prediction P, the predicted values are substracted from the corresponding samples in the original frame, which forms a residual block. The residual block is further transformed and the transform coefficients are quantized to reduce redundancy (see Section 3.1.6). At the end, the quantized transform coefficients are reordered and entropy coded using two different algorithms [12]. The entropy coded bit stream is then sent to the Network Abstraction Layer, which takes care of proper transmission or storage of the compressed data.

The *reconstruction path* of the encoder provides a reconstructed frame, which shows what the frame after the whole proccess of compression and decompression will look like. It is necessary for providing the output of the whole encoding and decoding process. In Intra prediction, the blocks in the reconstructed frame are further used for prediction. The block named Filter in the block diagram represents an adaptive deblocking filter, intended to smooth areas where block artifacts caused by compression are likely to appear. It improves the viewer's impression on the reconstructed video quality [20].

Decoder

To illustrate the correspondence between the encoder and the decoder of H.264/AVC, the decoder in Fig. 3.2 [20] is drawn in an unusual direction from right to left. The bit stream taken from the NAL is first subject to entropy decoding and reordering of transform coefficients. Inverse quantization and inverse quantization are then performed to produce a difference block. A prediction block is then added to it to form a reconstructed block, which is at the end adaptively smoothed by the deblocking filter.

Table 3.2: Slice types and allowed macroblock types.

Slice type	Description	Profile(s)
I (Intra)	Only Intra predicted macroblocks (each predicted from previously coded data within the same slice).	All
P (Predicted)	Contains Inter predicted macroblock and Intra predicted macroblocks.	All
B (Bi-predictive)	Contains Inter predicted macroblocks using two reference pictures and Intra predicted macroblocks.	Extended and Main
SP (Switching P)	Facilitates switching between coded streams. Contains Inter predicted and/or Intra predicted macroblocks.	Extended
SI (Switching I)	Facilitates switching between coded streams. Contains a special type of intra coded macroblocks.	Extended

3.1.3 Macroblocks, slices, frames

As noted earlier, the frames (fields) are not encoded as a whole at once, but the processing is done on blocks of a fixed size of 16 by 16 luma samples (and the corresponding block of chroma samples, a blocks of 8 by 8 samples of chroma Cr and a blocks of 8 by 8 samples for chroma Cb samples). These blocks are called *macroblocks*. They are numbered in raster scan order within a frame [20].

The macroblocks are organized in *slices*. The number of macroblocks in a slice is not fixed and doesn't even have to be the same within one picture. A slice may contain an integral number of macroblocks from one to the total number of macroblocks within a picture. The important thing about slices is that they can only contain certain types of macroblock depending on the slice type. The slice types and the respective allowed macroblock types are listed in Table 3.2 [20].

For motion compensated (Inter) prediction, one or several previously encoded pictures can be used as reference. The encoder and decoded maintain one or two lists of previously encoded and decoded pictures. In is worth noting that the time order of pictures (frames) in the original video doesn't imply the availability of pictures for prediction. In other words, we can predict from "future" blocks in time order – the time order of frames doesn't have to be the same as the decoding order.

3.1.4 Intra prediction

The H.264/AVC encoder is still operating on blocks like the MPEG-2 encoder [12, 20]. However, its function is more sophisticated. In the AVC, every block in the frame is predicted using previously encoded and decoded data. Every frame is divided in pixels called macroblocks, which are treated separately(see Section 3.1.3). Macroblocks are organized in slices, in which only a specified prediction can be used according to slice type. Basically, two different means of prediction can be used: for *Intra* prediction mode, prediction is done from previously decoded samples in the same slice (neighboring data).

For *Inter* prediction, motion vectors are used to predict from areas in previously coded data or even from future frames.

For *Intra* prediction, only the data in the current slice (and frame) can be used. This means, in the decoder all the predicted pixel values are only determined by the neighboring previously decoded pixels. There are different modes in which the Intra prediction of luma samples[2] can be performed [12]:

- Intra 16×16 luma prediction modes,
- Intra 8×8 luma prediction modes,
- Intra 4×4 luma prediction modes.

Intra 4 x 4 luma prediction

For Intra 4×4 luma prediction, the available modes together with simple sketches can be seen on Fig.3.3 [20]. Let us describe the process of the prediction for one of the schemes - the Intra 4×4 vertical prediction mode. As inputs for the prediction, 8 samples above, 4 samples to the left and one diagonal sample are taken. These luma samples are marked with blue color and indexed with capital letters A to M in Fig. 3.3. The output of the prediction process is a block of 16 samples, marked orange in Fig. 3.3. For vertical prediction mode, the predicted samples are derived only from the samples above the predicted area. In each column, the predicted luma values are copies of the reference samples right above the columns – all the predicted samples in first column are exact copies of sample A, for instance. A similar derivation is done for all the modes, however for some of them the computation of the predicted samples may be rather more complicated.

Intra 8 x 8 luma prediction

For Intra 8×8 luma prediction, again one of nine modes can be selected. As the mode schemes are similar to those shown in Fig 3.3 for Intra 4×4 prediction, no closer description will be given here. Of course, as we need to predict larger blocks, we also need more reference samples to use as inputs for the prediction. Hence we need twice as long row and twice as long column of reference samples, compared to the configuration for 4×4 prediction [12].

Intra 16 x 16 luma prediction

The last option of Intra prediction is to predict the whole macroblocks of 16×16 samples without further dividing them into smaller areas. As defined in H.264/AVC, only four prediction schemes are available, as we can only use the 16 neighboring samples to the top of the predicted area and 16 neighboring samples to the left. The situation is illustrated in Fig. 3.4, with the four different modes shown in a) to d) [20].

Further and more detailed description of the respective prediction modes and predicted sample calculations is beyond the scope of this thesis and is not necessary for our later considerations.

[2]Likewise, such modes are defined for chroma sample prediction. However, we will only consider luma coding now.

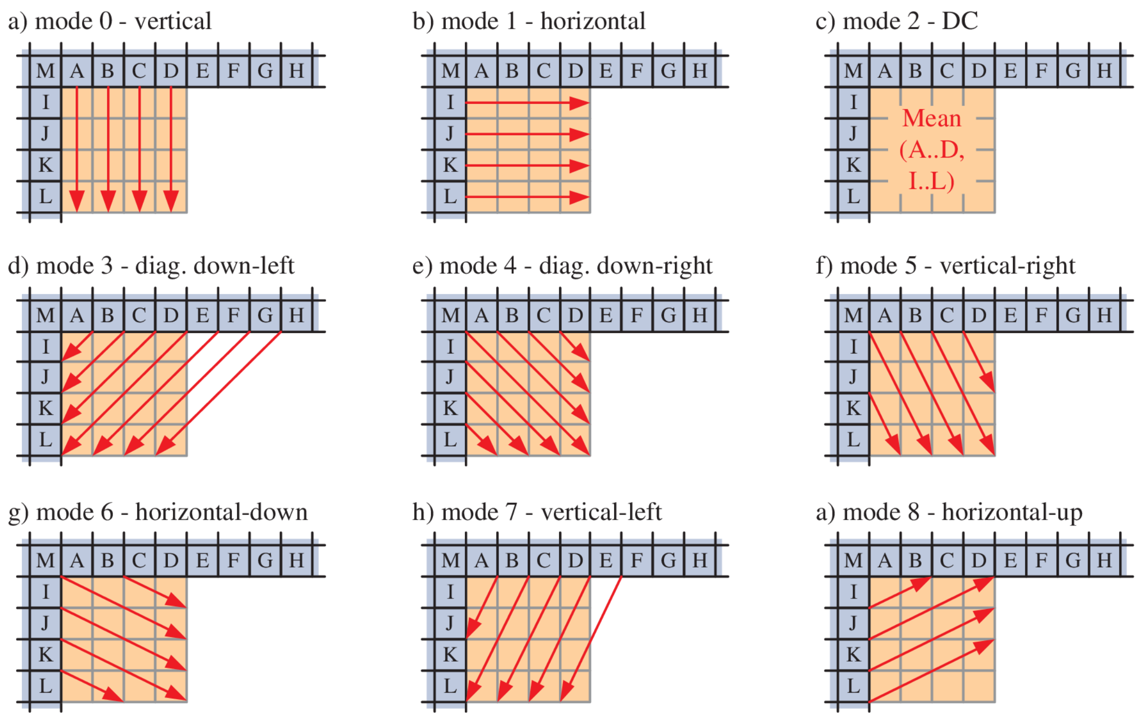
a) mode 0 - vertical
b) mode 1 - horizontal
c) mode 2 - DC
d) mode 3 - diag. down-left
e) mode 4 - diag. down-right
f) mode 5 - vertical-right
g) mode 6 - horizontal-down
h) mode 7 - vertical-left
a) mode 8 - horizontal-up
M A B C D E F G H
I J K L
Mean (A..D, I..L)

Figure 3.3: Intra 4 × 4 prediction modes.

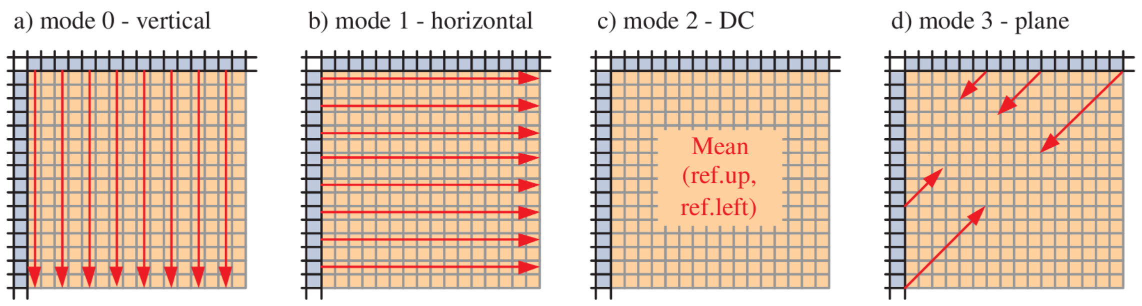

Figure 3.4: Intra 16 × 16 prediction modes.

There is one more mode to be mentioned - the "I_PCM" mode, which enables the encoder to transmit the sample values directly, without undergoing the chain of Intra prediction, transformation, quantization and entropy coding [12]. This scheme may be advantageous in some situations.

3.1.5 Inter prediction

Similarly to Intra prediction, the block size for Inter prediction is not fixed in H.264/AVC. It can be changed from 16×16 down to 4×4. This scheme is called the *tree structured motion compensation.* In addition to this, prediction can be used in quarter-pixel accuracy. As described in Section 3.1.4, *Intra* prediction of a luma macroblock can be performed on a whole 16×16 macroblock, four 8×8 blocks or sixteen 4×4 blocks. For *Inter* coded macroblock, the luminance component of each macroblock (16×16 samples) may be split up in four ways and motion compensated either as one 16×16 macroblock partition, two 16×8 partitions, two 8×16 partitions or four 8×8 partitions. If the 8×8 mode is chosen, each of the four 8×8 sub-macroblocks within the macroblock may be split in further 4 ways. These partitions and sub-macroblocks give rise to a large number of possible combinations within each macroblock. The way macroblock partitions and sub-macroblock partitions can be organized is shown in figures 3.5 and 3.6, respectively. In addition to all the prediction modes, a macroblock may be *Direct* coded, which means no information is transmitted and the macroblock si simply copied from the reference picture.

A separate motion vector is required for each partition or sub-macroblock. Each motion vector must be coded and transmitted, and the choice of partition(s) must be encoded in the compressed bit stream. Choosing a large partition size means that a fewer bits are required to signal the choice of motion vector(s) and the type of partition but the motion compensated residual may contain a significant amount of energy in areas with high detail. Choosing a small partition size may give a lower-energy residual after motion compensation but requires a larger number of bits to signal the motion vectors and choice of partition(s). The choice of partition size therefore has a significant impact on compression performance. In general, a large partition size is appropriate for homogeneous areas of the frame and a small partition size may be beneficial for detailed areas.

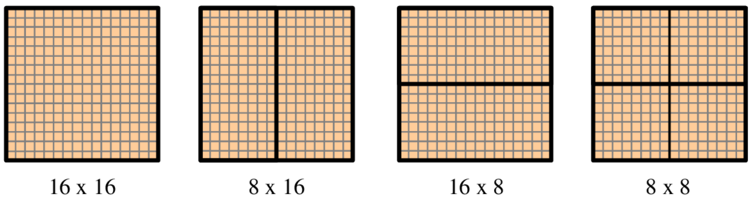

Figure 3.5: Macroblock partitions.

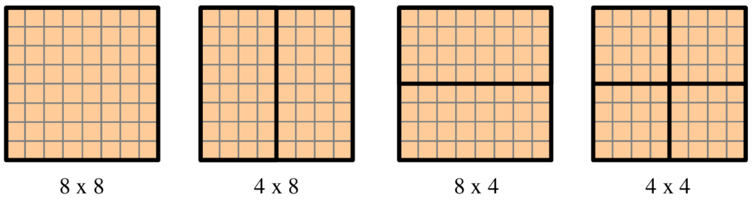

Figure 3.6: Sub-macroblock partitions.

3.1.6 Quantization

After prediction, the residuals must be encoded. The H.264/AVC standard uses three transforms depending on the residual data to be coded:

- Hadamard transform for the 4×4 array of luma DC coefficients in Intra macroblocks predicted in 16×16 mode,
- Hadamard transform for the 2×2 array of chroma DC coefficients (in any macroblock),
- DCT-based transform for all other 4×4 blocks in the residual data.

H.264/AVC assumes scalar quantization. The basic forward quantizer operation is [20]

$$Z_{i,j} = round(Y_{i,j}/Qstep), \tag{3.1}$$

where $Y_{i,j}$ are the transform coefficients, $Qstep$ is a quantizer step size and $Z_{i,j}$ are the quantized coefficients. A total of 52 $Qstep$ values are supported by the standard, indexed by a *Quantization Parameter* – QP.

3.2 Artificial Neural Networks

This Section describes the basic and essential theory on artificial neural networks. In later Sections, we will use them as a tool for video quality measurements, which cannot be achieved without loss of continuity unless we define the basic terms.

In their infancy, artificial neural networks (ANNs), were technical tools trying to model the biological neural networks. Over the years, however, they have developed into a strong instrument for data processing, classification and clustering. They are used in a wide range of applications, reaching from signal processing to financial prognoses. One common thing for all ANNs is that they consist of very simple units, but often very many units are used. The network structures may vary significantly – from single neuron units to complex structures with feedbacks. The most common type, however, are feedforward backpropagation networks, which will be described in this chapter. The word backpropagation in the name is related to the way the networks are trained - errors propagate back throught the network to adjust the neuron parameters.

The power of artificial neural networks is in their adaptability – we can train the networks to adapt to some given inputs. We should define two ways in which the ANNS may be trained, the supervised and unsupervised learning. In *unsupervised* learning, the network is given a set of training examples with no target outputs. In this case, the network is trained to classify similar inputs to groups, for instance. In *supervised* learning, the ANN is given a set of inputs with the desired outputs. During the training process, the network parameters are adjusted in order to get close or equal to the targets. We will only focus on supervised learning here, as it is suitable for our application.

3.2.1 Neuron model

The basic model of a neuron (a network unit) is shown in Fig. 3.7. It can be described by a series of functional transformations. First of all, assume a linear combination of M input variables $x_1, x_2, \ldots, x_M$ as [2]

$$a = \sum_{i=1}^{M} w_i x_i + b, \tag{3.2}$$

or in vector representation as [2, 4]

$$a = \mathbf{wx} + b. \tag{3.3}$$

Here $\mathbf{w}$ is a row vector and $\mathbf{x}$ is a column vector. The symbol b in Eq. 3.2 and Eq. 3.3 represents *bias*, a constant which is added to the weighted sum of inputs. In some literature, the Equation 3.2 is written as [16]

$$a = \sum_{i=0}^{M} w_i x_i, \tag{3.4}$$

where the bias seems to have disappeared. However, the bias is still present, but is denoted as a value of the weight vector w_0 with a fixed corresponding input $x_0 = 1$. The value a may be denoted as *output unit activation.* It is used as an input to the *transfer function* of *activation function* f to form the neuron output y as [4]

$$y = f(a) = f\left(\sum_{i=1}^{M} w_i x_i + b\right) = f\left(\mathbf{w} \cdot \mathbf{x}^T + b\right). \tag{3.5}$$

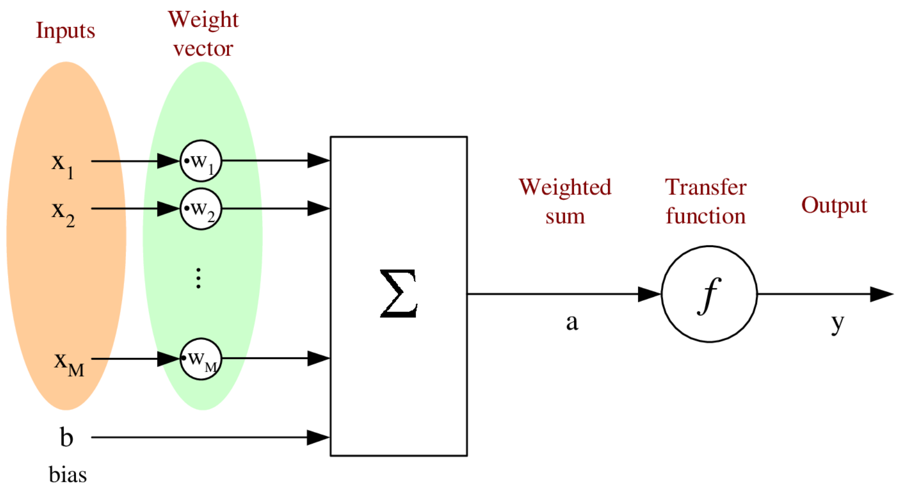

Figure 3.7: Neuron model.

The function f is not limited to a certain function, many variants can be used. The most common choices are as follows:

Hard-limit transfer function. This is the simplest case and the neuron using hard-limiting transfer function is called *perceptron*. The function can be written as follows [16]:

$$f_1(a) = \begin{cases} 1 & \text{for} \quad a \geq 0 \\ 0 & \text{otherwise} \end{cases} . \tag{3.6}$$

Alternatively, a symmetric hard-limiting function can be used, given by [4]

$$f_2(a) = \begin{cases} 1 & \text{for} \quad a \geq 0 \\ -1 & \text{otherwise} \end{cases} . \tag{3.7}$$

Linear transfer function. The hard-limiting transfer function has its output limited to only two values and can thus be used for clustering or data classification. However, if a range of output values is needed rather than a binary output, using a linear transfer function in the form [4]

$$f_3(a) = a \tag{3.8}$$

may be beneficial. An interesting point about linear networks is mentioned in [4]: any multi-layer artificial neural network (only consisting of neurons with linear transfer functions) can be replaced by a single-layer network with linear units. Thus, for one desired output, we only need one linear neuron unit to examine the capabilities of a linear network.

Linear neurons can be trained to learn an affine function of their inputs, or to find a linear approximation to a nonlinear function. A linear network cannot, of course, be made to perform a nonlinear computation.

Sigmoid transfer function. In case the artificial neural network is supposed to represent nonlinear functions of its inputs, a linear network does not suffice and so introduce a nonlinear transfer function of the neurons is introduced. A neuron whose output is a nonlinear function of its inputs is needed, and, at the same time, the output needs to be a differentiable function of the inputs. One such solution is the *sigmoid* unit, whose transfer function is given by [2]

$$f_4(a) = \sigma(a) = \frac{1}{1+e^{-a}}. \tag{3.9}$$

The sigmoid transfer function is very much like the perceptron hard-limiting transfer function – it applies a threshold to the linear combination of the neuron inputs. However, the threshold output is a continuous function of its inputs [16]. This is beneficial the for gradient descent learning algorithm, described in Section 3.2.2.

3.2.2 Network training

The real power of the artificial neural networks lies in their ability to adapt in order to give desired outputs. However most problems will probably be simulated by multi-layer networks, let the training algorithms for the simplest networks – single neuron units – be described first.

Perceptron training rule

Although the applicability of a perceptron is limited to linearly separable problems, let us begin with understanding how to train this basic unit. There are several training algorithms available, out of which the *perceptron training rule* [4, 16] will be described at this point. Other algorithms, such as *delta rule*, can be found in [15, 16], and some will be noted in later sections. These algorithms provide the basis for training networks of many units.

The initial step in the perceptron training rule is to set up random weights and a random bias. Then, for each of the training examples, the output is computed and the weights are modified in case the perceptron micslassifies the training example. This process is repeated, iterating through the training examples as many times as needed until the perceptron classifies all the training examples correctly. The weights are updated according to the perceptron training rule, which updates the weight w_i associated with input x_i. One update step is then [16]

$$w_i \Leftarrow w_i + \Delta w_i \tag{3.10}$$

for one weight. For the whole weight vector, the expression can be written as

$$\mathbf{w} \Leftarrow \mathbf{w} + \Delta\mathbf{w}. \tag{3.11}$$

Similarly, the bias will be updated as [16]

$$b \Leftarrow b + \Delta b. \tag{3.12}$$

For each training example, the input vector $\mathbf{x}$ and the corresponding correct (target) output t is available. After setting the initial random weights and random bias, the perceptron output y can be computed using Eq. 3.5. There are then three cases that can occur for a single neuron [4]:

1. If the output is correct ($y = t$), the weight vector is not updated, $\Delta\mathbf{w} = 0$.
2. If the neuron output is 0 and should have been 1 ($y = 0$ and $t = 1$), the transposed input vector $\mathbf{x}^T$ is added to the weight vector $\mathbf{w}$, $\Delta\mathbf{w} = \mathbf{x}^T$.
3. If the neuron output is 1 and should have been 0 ($y = 1$ and $t = 0$), the transposed input vector $\mathbf{x}$ is substracted from the weight vector $\mathbf{w}$, $\Delta\mathbf{w} = -\mathbf{x}^T$.

These three cases can be written with a single expression as [4]

$$\Delta\mathbf{w} = (t - y)\mathbf{x}^T = e\mathbf{x}^T, \tag{3.13}$$

where the term e is the classification error. Now recall the Eq. 3.4, where the neuron bias was denoted as a member of the weight vector with a constant corresponding input of 1. Obviously, the bias update step is

$$\Delta b = (t - y)(1) = e. \tag{3.14}$$

The degree to which the weights are changed at each step may be controlled by a parameter called *learning rate*, denoted by η [2, 4, 16]. It is a small positive constant, reducing the weight update speed. With the learning rate, the Equations 3.13 and 3.14 get the form [4]

$$\Delta\mathbf{w} = \eta(t - y)\mathbf{x}^T = \eta e\mathbf{x}^T \tag{3.15}$$

and

$$\Delta b = \eta(t - y)(1) = \eta e, \tag{3.16}$$

respectively. The perceptron training rule is guaranteed to find a solution in a finite number of iterations, provided the solution exists.

Gradient descent algorithm

The perceptron learning rule is only applicable for perceptron training, i.e. only hard-limiting units can be trained. However, we might need a different unit trained. For this, the *gradient descent algorithm* (or the delta rule) will now be introduced, based on the *LMS* (least mean squares) *algorithm*. For simplicity, only a *linear* unit will be considered, whose transfer function is pure linear function, and so its output y is equal to the weighted sum a (see Fig. 3.7). We will see that this algorithm can be used for any unit whose transfer function is differentiable. For simplicity, the notation introduced in Eq. 3.4 will be used, where the bias is a member of the weight vector $\mathbf{w}$ as w_0.

Let us define the *training error* as [16]

$$E(\mathbf{w}) = \frac{1}{D}\sum_{d=1}^{D}(t_d - y_d)^2, \tag{3.17}$$

where D is the set of training examples, t_d is the desired (target) output for training example d and y_d is the linear neuron output for the training example d. The error function represents the mean squared error for all the training examples. The error, of course, depends on the set of the training examples as well, but as it is assumed that this set is not changing during training, it is not explicitly written in Eq. 3.17.

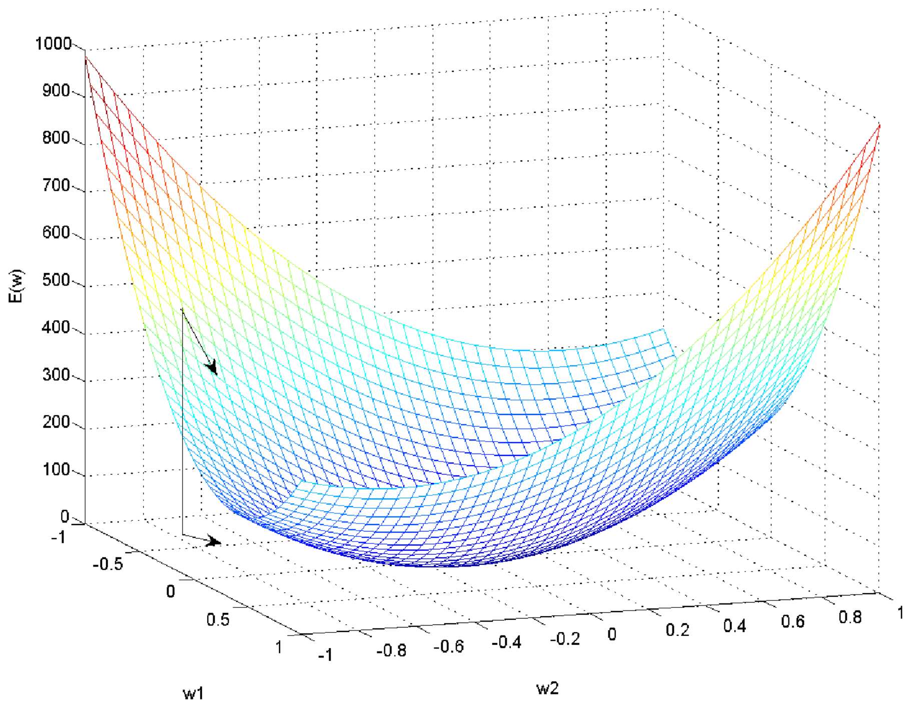

Figure 3.8: Error of different hypotheses.

Now imagine the whole hypothesis space with the possible weight vectors and their E values. For a linear unit with two inputs, the situation may be as shown in Fig. 3.8. The hypothesis space of a linear unit is always parabolic with a single minimum. Of course, the specific parabola will depend on the particular choice of training examples [16].

As the goal is find the weight vector for which the error is minimal, the best choice is to alter the weights in the direction of the steepest descent of E in each iteration. To calculate the direction of the steepest descent along the error surface, the derivative of E with respect to each component of weight vector $\mathbf{w}$ [16] can be used:

$$\nabla E(\mathbf{w}) = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_M}\right]. \tag{3.18}$$

The gradient can then be used to compute the training step $\Delta\mathbf{w}$ (see Eq. 3.11) in the form

$$\Delta\mathbf{w} = -\eta\nabla E(\mathbf{w}), \tag{3.19}$$

where η is the learning rate. For each weight w_i of the weight vector $\mathbf{w}$ we can write

$$\Delta w_i = -\eta\frac{\partial E}{\partial w_i}. \tag{3.20}$$

To be able to efficiently update the weight vector in each iteration, way to compute the gradient at each step is needed. For this, consider the derivative of E from equation 3.17 with respect to one weight component w_i as [16]

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i}\frac{1}{D}\sum_{d=1}^{D}(t_d - y_d)^2, \tag{3.21}$$

which leads to

$$\frac{\partial E}{\partial w_i} = \frac{1}{D}\sum_{d=1}^{D} 2\left(t_d - y_d\right)\frac{\partial}{\partial w_i}\left(t_d - y_d\right). \tag{3.22}$$

Substituting y_d from Eq. 3.5 and considering the transfer function of a linear unit is identity results in

$$\frac{\partial E}{\partial w_i} = \frac{2}{D}\sum_{d=1}^{D}\left(t_d - y_d\right)\frac{\partial}{\partial w_i}\left(t_d - \mathbf{w}\cdot\mathbf{x}_d\right) \tag{3.23}$$

For a linear unit, we can write [16]

$$\frac{\partial}{\partial w_i}\left(t_d - \mathbf{w}\cdot\mathbf{x}_d\right) = -x_{id}, \tag{3.24}$$

and thus

$$\Delta w_i = \frac{2}{D}\eta\sum_{d=1}^{D}\left(t_d - y_d\right)x_{id}. \tag{3.25}$$

The number of training examples D is a constant and as such can be absorbed into the learning rate η. Finally, the weight training step is derived in the form

$$\Delta w_i = \eta\sum_{d=1}^{D}\left(t_d - y_d\right)x_{id}. \tag{3.26}$$

Recall that the notation introduced in Eq. 3.4 is used, where the bias is denoted as w_0 and its corresponding input is always $x_{0d} = 1$. For the bias training step, this leads to

$$\Delta b = \Delta w_0 = \eta\sum_{d=1}^{D}\left(t_d - y_d\right). \tag{3.27}$$

The Equations 3.26 and 3.27 define the training steps of weights and the bias for a linear unit in one iteration, i.e. in one presentation of the set of training examples. For a unit with a different transfer function, Eq. 3.22 would have to be used and the derivative according to the particular transfer function computed.

3.3 PSNR Estimation

In the following, the description of the new video quality metric is provided, which is the actual original contribution of the thesis.

Consider a configuration shown in Fig. 2.1a). A video material undergoes a process, at the end of which we are trying to evaluate the quality of the resulting video. What we have available is the original and the processed video sequence, so we can easily compare them to get a quality measure such as the simplest one – the PSNR, described by Eq. 2.1. Although the PSNR does not give a perfect information about the perceived quality [22], it is the most common metric and is well understood and very often used.

In the configuration shown in Fig. 2.1c), we do not have access to the original material when trying to evaluate the quality. Consequently, it is much more difficult to get an objective, exactly defined score.

For the H.264/AVC, the number of available no reference quality metrics is very limited (see Sec. 2.4). Objective metrics for image quality measurement depending on the compression itself are still in their infancy. What will be described in the following sections is a no-reference metric operating on the encoded bit stream only, which estimates the PSNR – a value which needs the original as well as the decoded material as inputs to be computed. What are the possible applications of such "PSNR Estimator"? Apart from fast quality evaluation of compressed bit streams before or after their transmission, it may easily be used to evaluate the quality and the weak spots of a statistically multiplexed broadcast channel, for instance.

3.3.1 The framework

In Fig. 3.9, a simple scheme of the new metric is shown [31], [32]. The metric operates on the H.264/AVC compressed bit stream, which brings the benefit of quick processing without the necessity of decoding the actual pixel content. For simplicity, in the beginning it works with files stored on a computer hard disk. An extension for streaming video will then be straightforward.

In the block called *Feature extractor*, the bit stream is parsed and all but the desired parameter values is discarded. The selected parameters will be described in detail in Section 3.4. After extracting the parameter values, a *Mapping* algorithm comes into play to form a quality measure.

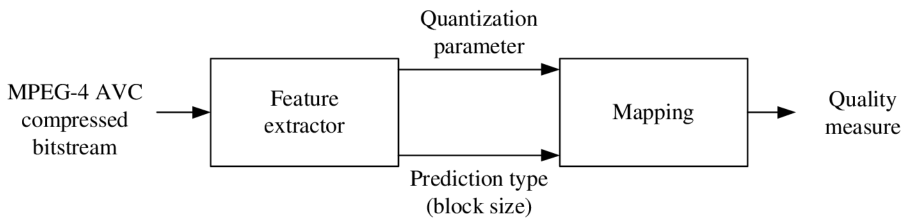

Figure 3.9: Scheme of the PSNR Estimator.

3.4 Feature Extractor

The Feature extractor forms the first part of the new quality metric. An application has been developed to parse the bit stream and to store the desired parameter values. As no user interaction is needed for its function, the application is written in C language as a command line program.

Basically, the H.264/AVC defines two levels in which the data is handled - the *video coding layer* (VCL) and the *network abstraction layer* (NAL), see Section 3.1.1. While the VCL deals with the information and processes needed to decode a video sequence, the NAL is defined to handle encoded data in terms of transmission and storage. To be able to extract the bit stream parameters, both the NAL and VCL decoding needs to be considered.

To achieve both NAL and VCL decoding, the JVT JM11 reference decoder [33] is used and modified, implementing several functions to store the extracted data. The reference software is written in C language and includes H.264/AVC encoder and decoder. According to the license, users are free to use and/or modify the reference software for their purposes. Practical applicability of the software for video encoding and decoding is not very high as the software implementation is very slow. As encoder input, raw YUV video sequences are used. The encoding process can easily be controlled by modifying input parameters. Decoding fully relies on data present in the encoded bit stream.

When running the modified decoder, the parameter values are stored in a separate file for each frame. Special care was taken when modifying the decoder to gain access to the desired data *before* the actual pixel decoding starts. Consequently, the final video quality metric will rely on the data present in the bit stream and thus computational complexity will be reasonably low. The output format is MATLAB compatible *.mat* file. This will make the development of the mapping algorithm much easier using the MATLAB toolboxes.

3.4.1 Intra predicted pictures

Figures 3.10 – 3.13 represent the first frames of the two encoded sequences with different target bit rates. These sequences were downloaded from [1] and are in CIF resolution (352×288 pixels). In each of these images, the top left (a) part shows the decoded image. The top right part (b) displays a block type grid, illustrating what block sizes are used throughout the frame. The bottom right part of the figure shows the same data in a different representation: the lighter the color of the area, the smaller blocks are used for prediction. Please note all of the frames are *Intra* coded only (first slice of the sequence must always be Intra coded, in this case the whole frame consists of only one slice), only three different sizes are thus possible as only 16×16, 8×8 and 4×4 modes are available for Intra coded slices. The same analysis is possible for *Inter* coded slices as well, and will be described later. However, it is obvious that the quality rating of *Inter* coded video image areas strongly depends on the data from which the area is predicted. As such, the analysis on *Intra* coded slices is the first step.

To sum up, the list of extracted parameters for intra coded pictures (frames, fields or slices) will be as follows:

- % of Intra 16×16 predicted luma blocks,

- % of Intra 8×8 predicted luma blocks,
- % of Intra 4×4 predicted luma blocks,
- % of I_PCM luma blocks (directly encoded - see Section 3.1.4),
- Quantization parameter[3] for the actual picture.

As an example, the parameter values for frames displayed in Figures 3.10 – 3.13 are listed in Table 3.3. Note the changing size of encoded blocks. It is obvious that for higher bit rates (where higher PSNR is achievable), finer structure is used for prediction.

There are different profiles defined in the H.264/AVC, the Baseline profile being the simplest and implementing only a limited set of features [12]. The sequences analyzed in this section are encoded using the High profile as it implements the widest range of options. Regarding the Intra coded frames, the difference between the respective profiles is in that only the High profile is capable of using Intra 8×8 block prediction.

Table 3.3: Parameters of the encoded sequences, frame 0.

sequence	Foreman	Foreman	Tempete	Tempete
PSNR [dB]	33.83	40.08	30.59	36.33
bit rate [kbps]	106	1004	156	827
Intra 16x16	17.2 %	15.4 %	4.8 %	2.8 %
Intra 8x8	49.5 %	29.5 %	52.0 %	19.7 %
Intra 4x4	33.3 %	55.1 %	43.5 %	77.5 %
I_PCM	0.0 %	0.0 %	0.0 %	0.0 %
QP	35	25	35	28
Figure	3.10	3.11	3.12	3.13

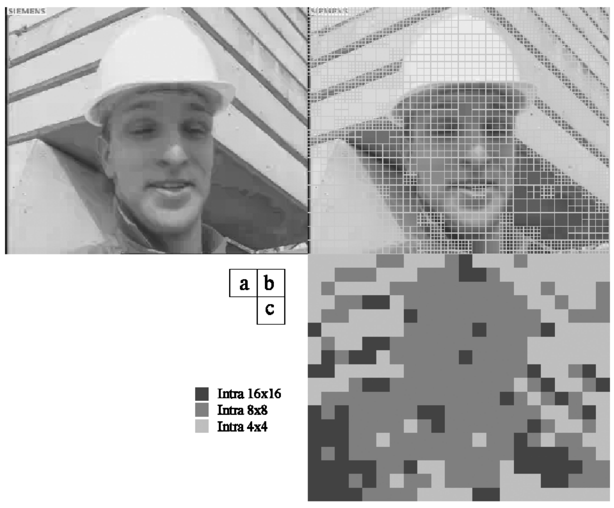

Figure 3.10: Foreman sequence, frame 0, PSNR = 33.83 dB, QP = 35.

[3]Although the H.264/AVC defines variability of the quantization parameter within a slice, the JM11 reference encoder does not use this feature. This is why only one number of the quantization parameter is given in the description and only one number is extracted from the encoded sequence.

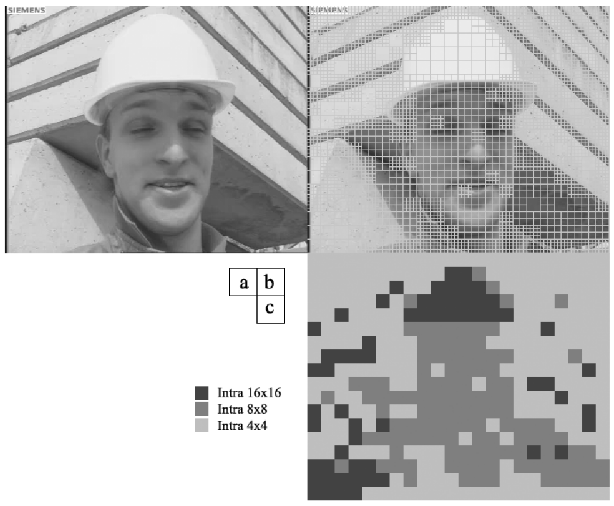

Figure 3.11: Foreman sequence, frame 0, PSNR = 40.08 dB, QP = 25.

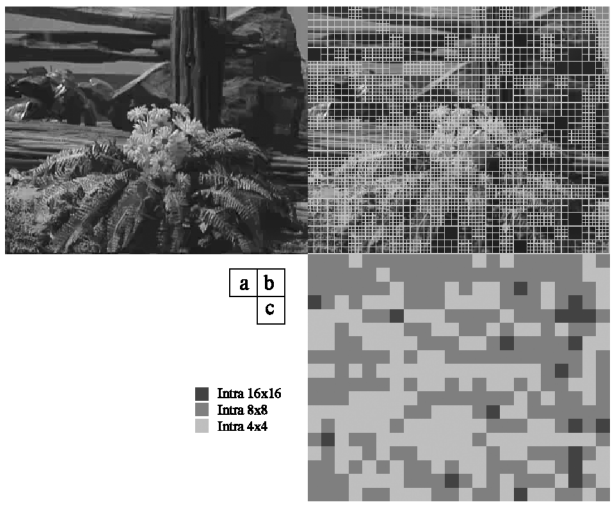

Figure 3.12: Tempete sequence, frame 0, PSNR = 30.59 dB, QP = 35

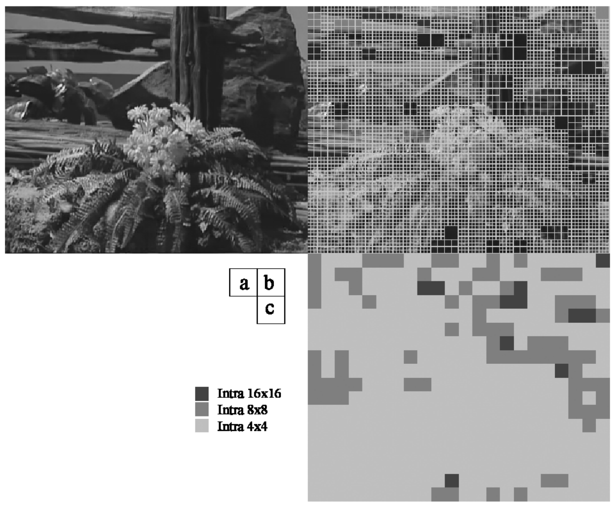

Figure 3.13: Tempete frame, PSNR = 36.66 dB, QP = 28

3.4.2 Inter predicted pictures

In Figures 3.14 – 3.17, block type maps for Inter predicted pictures are given. The used sequences are the same as for Intra predicted pictures and the Figures are similar to those for Intra predicted pictures. Of course, in Inter predicted pictures, blocks may be predicted from other pictures using Inter prediction and motion compensation. A whole set of Inter block types is then available. The Figures are organized as follows: The top left part shows the decoded frame, the top right right part (b) displays a block type grid, part (c) shows the block type map for Intra coded blocks and the part (d) shows the block type map for Inter coded blocks. I_PCM and Direct coded macroblocks are not displayed for simplicity (they will appear as black in both block type maps). Again, the lighter the color of the blocks in parts (c) and (d), the smaller blocks are used for prediction.

According to the H.264/AVC specification [12], in Inter predicted pictures blocks may be encoded using either Intra or Inter prediction modes, which means for PSNR prediction of Inter coded pictures, the same parameters as defined in previous subsection for Intra coded pictures should be used, i.e.

- % of Intra 16×16 predicted luma blocks,
- % of Intra 8×8 predicted luma blocks,
- % of Intra 4×4 predicted luma blocks,
- % of I_PCM luma blocks (directly encoded - see Section 3.1.4),
- Quantization parameter for the actual picture.

In addition, parameters describing the Inter predicted part of the picture will be used. These are

- % of Inter 16×16 predicted luma blocks,
- % of Inter 16×8 or 8×16 predicted luma blocks,
- % of Inter 8×8 predicted luma blocks,
- % of Inter 8×4 or 4×8 predicted luma blocks,
- % of Inter 4×4 predicted luma blocks,
- % of Direct luma blocks (directly encoded - see Section 3.1.5).

Note that as only block sizes for each type of prediction are considered in our approach, no distinction is made in the shape of the block – 16×8 blocks are treated same as 8×16 blocks, for instance.

Observe the behavior of the block types within the coded pictures. For low quality, highly compressed pictures, the PSNR remains low and the quantization parameter is high, which is a signal of rough quantization and a significant loss of details in transform coefficients. Regarding the block types, low quality pictures tend to be predicted using larger blocks, both for Intra predicted and Inter predicted area. This effect is very illustratively shown in Figure pairs 3.14 – 3.15 and 3.16 – 3.17

Table 3.4: Parameters of the encoded sequences, frame 5.

sequence	Foreman	Foreman	Tempete	Tempete
PSNR [dB]	30.83	37.61	28.03	31.15
bit rate [kbps]	33.01	500.63	48.36	500.76
Intra 16x16	3.03 %	1.77 %	0.51 %	0 %
Intra 8x8	0 %	0 %	0 %	0 %
Intra 4x4	0 %	4.04 %	0.25 %	1.01 %
I_PCM	0 %	0 %	0 %	0 %
Inter 16x16	20.71 %	28.53 %	29.29 %	35.86 %
Inter 16x8 (8x16)	2.78 %	21.21 %	4.04 %	19.70 %
Inter 8x8	0 %	11.17 %	0.19 %	8.33 %
Inter 8x4 (4x8)	0 %	5.49 %	0.06 %	5.93 %
Inter 4x4	0 %	0.76 %	0 %	1.14 %
Direct	73.48 %	27.02 %	65.66 %	28.03 %
QP	43	27	43	33
Figure	3.14	3.15	3.16	3.17

Again, let us give an example of the extracted parameter values for encoded sequence frames, see Tab. 3.4.

As Inter predicted blocks are derived from other blocks in other pictures (see Sec. 3.1.5), the resulting signal-to-noise ratio of the predicted block will strongly depend on the SNR of the reference data. This brings results in a necessity of taking another set of parameters into account - the *referenced PSNR*. Fortunately, the situation is quite simple as long as only the Baseline profile is considered. Each of the Inter predicted blocks is derived from a reference block in one of the previously decoded pictures [12]. In the H.264/AVC decoder, the decoded pictures are stored for reference in two lists (List0 and List1, [12, 20]). An index to a list is then carried in the bit stream to tell which picture to predict from and is our clue to the PSNR of the reference picture. Unfortunately, when applying our quality metric as a no-reference tool, real PSNR of the reference picture will not be available. In the phase of designing the algorithm, the PSNR values computed using comparison with the original will be used. After that, the PSNR values our algorithm estimated for the previous (reference) pictures will be used. More on this topic will follow in Sec. 3.5.4.

It was stated that for the Inter predicted blocks, the PSNR should be derived from the reference picture PSNR as well. Thus, for each Inter block type ratio (i.e. 16×16 inter blocks, 16×8 or 8×16 inter blocks, 8×8 inter blocks, 8×4 or 4×8 inter blocks, 4×4 inter blocks, direct blocks) an average reference PSNR (refPSNR) from the reference pictures needs to be computed. In addition to the parameters listed in Tab. 3.4, for each inter predicted frame the following set of parameters can thus be defined:

- 16×16 block reference PSNR,
- 16×8 or 8×16 block reference PSNR,
- 8×8 block reference PSNR,
- 8×4 or 4×8 block reference PSNR,
- 4×4 block reference PSNR.

Rather than averaging the PSNR directly, it is desirable to calculate the PSNR using the mean squared error of the area used for prediction. Expressing MSE from Eq. 2.1 leads to

$$\text{MSE} = \frac{m^2}{10^{\frac{\text{PSNR}}{10}}}. \tag{3.28}$$

Calculating an overall mean squared error $\overline{\text{MSE}}$ of an area consisting of N_1 blocks within a picture having the mean squared error MSE_1 and peak signal-to-noise ratio PSNR_1, N_2 blocks within a picture with MSE_2 and PSNR_2, etc. yields

$$\begin{aligned}\overline{\text{MSE}} &= \frac{N_1 \cdot \text{MSE}_1 + N_2 \cdot \text{MSE}_2 + \ldots + N_x \cdot \text{MSE}_x}{N_1 + N_2 + \ldots + N_x} = \\ &= \frac{\frac{N_1 \cdot m^2}{10^{\frac{\text{PSNR}_1}{10}}} + \frac{N_2 \cdot m^2}{10^{\frac{\text{PSNR}_2}{10}}} + \ldots + \frac{N_x \cdot m^2}{10^{\frac{\text{PSNR}_x}{10}}}}{N_1 + N_2 + \ldots + N_x} = \\ &= \frac{m^2 \left(\frac{N_1}{10^{\frac{\text{PSNR}_1}{10}}} + \frac{N_2}{10^{\frac{\text{PSNR}_2}{10}}} + \ldots + \frac{N_x}{10^{\frac{\text{PSNR}_x}{10}}} \right)}{N_1 + N_2 + \ldots + N_x} = \\ &= \frac{m^2 \sum_{i=1}^{x} \frac{N_i}{10^{\frac{\text{PSNR}_i}{10}}}}{\sum_{j=1}^{x} N_j},\end{aligned} \tag{3.29}$$

where x is the total number of different PSNRs in the reference pictures the prediction is done from.

Substituting the result of Eq. 3.29 back into Eq. 2.1 gives the overall reference PSNR for one block size as

$$\begin{aligned}\text{refPSNR} &= 10 \log_{10} \frac{m^2}{\overline{\text{MSE}}} = \\ &= 10 \log_{10} \frac{m^2 \sum_{i=1}^{x} N_i}{m^2 \sum_{i=1}^{x} \frac{N_i}{10^{\frac{\text{PSNR}_i}{10}}}} = \\ &= 10 \log_{10} \sum_{i=1}^{x} N_i - 10 \log_{10} \sum_{j=1}^{x} \frac{N_j}{10^{\frac{\text{PSNR}_j}{10}}}.\end{aligned} \tag{3.30}$$

Main profile

While the Baseline profile does not support bi-directional Inter prediction, such feature is available in the Main profile. For each bi-directionally predicted block, the only thing that gets complicated compared to the Baseline profile setting is that there is not only one reference PSNR as defined in Eq. 3.30 for each predicted block size, but there are two references – for one prediction direction each.

Consider all blocks within a picture that are Inter coded using one block size (8×8, for instance). Each such block can thus be predicted from one reference picture (in one direction) or from two reference pictures (bi-directionally). To put the correct PSNR of the reference into Eq. 3.30, the PSNR_B of the block B should be derived as follows:

1. The block is predicted only in one direction – the PSNR of its reference picture will be used as the block PSNR_B

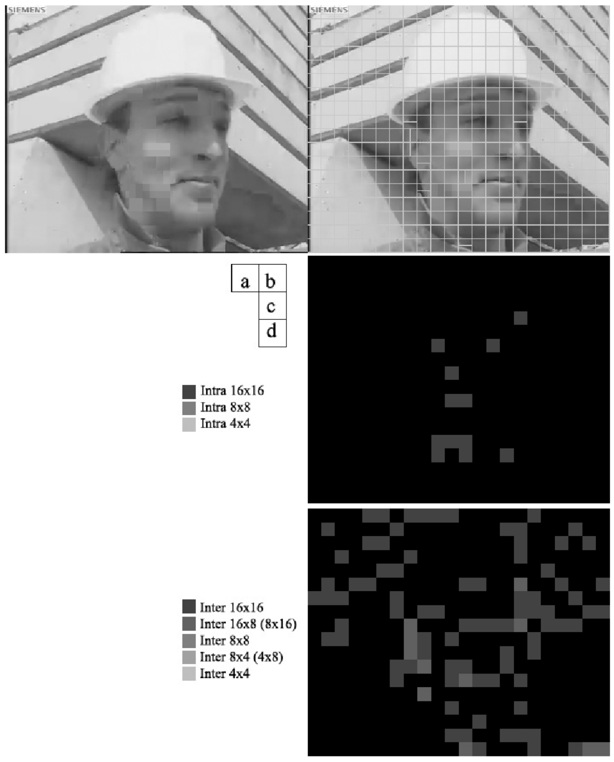

Figure 3.14: Foreman sequence, frame 5, PSNR = 30.83 dB, QP = 43.

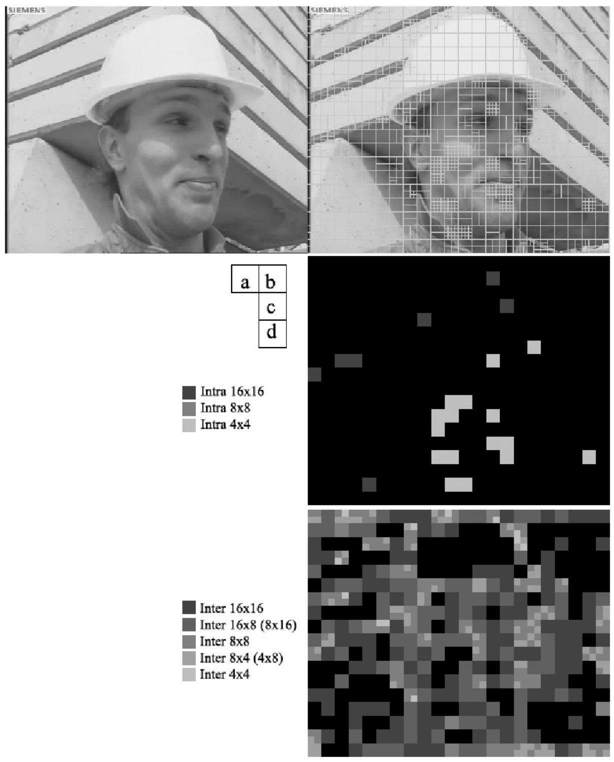

Figure 3.15: Foreman sequence, frame 5, PSNR = 37.61 dB, QP = 27.

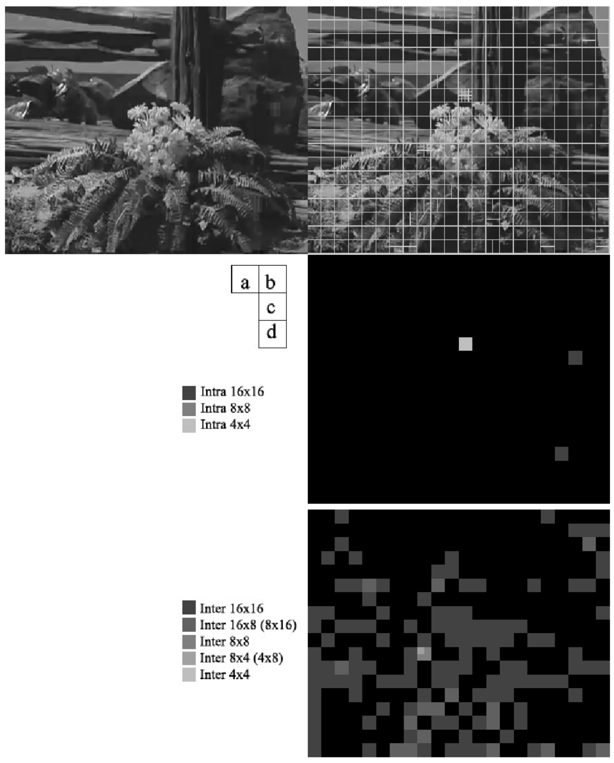

Figure 3.16: Tempete sequence, frame 5, PSNR = 28.03 dB, QP = 43

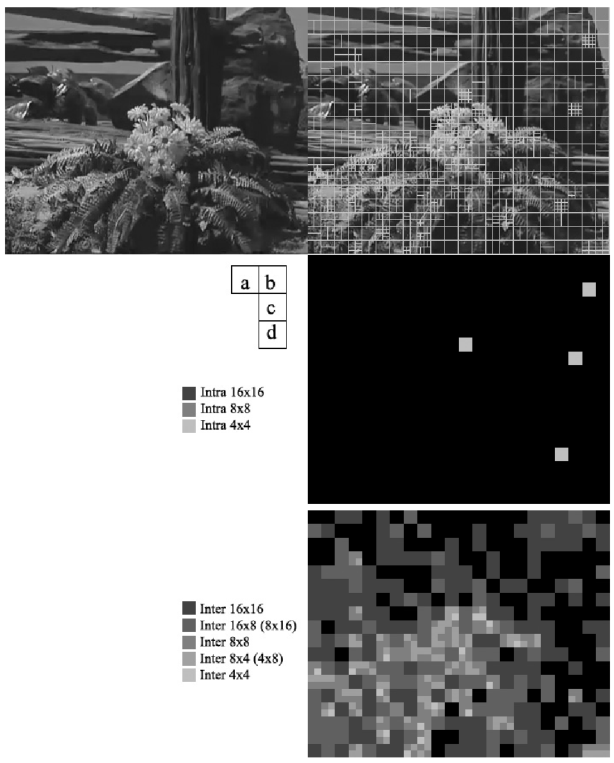

Figure 3.17: Tempete sequence, frame 5, PSNR = 31.15 dB, QP = 33

2. The block is predicted bi-directionally – the PSNR of the reference area will be calculated similarly to the expression in Eq. 3.30. Assume the block B is predicted from two blocks $B1$ and $B2$. The PSNR of the reference pictures is known (PSNR_{B1} and PSNR_{B2}). Based on the expression in Eq. 3.30, the reference PSNR for the block B is

$$\mathrm{PSNR_B} = 10 \log_{10} 2 - 10 \log_{10} \frac{2}{10^{\frac{\mathrm{PSNR_{B1}}}{10}} + 10^{\frac{\mathrm{PSNR_{B2}}}{10}}}. \qquad (3.31)$$

Extended and High profiles

The H.264/AVC defines the Extended profile, High profile, High 10 profile, High 4:2:2 profile and High 4:4:4 profile apart from those described in previous sections (Baseline and Main profile) [12]. However, there are no additional features to take care of in our approach – the only interesting feature in the Extended profile is that it introduces special macroblock types for switching purposes. The possibility of block size analysis is, however, still available. The parameters and calculations described above for all these profiles will thus be used.

3.5 Mapping

Now that all the necessary parameters are extracted, let us move forward to the design of the mapping algorithm. For this, a set of training video sequences can be used, each encoded with the JM11 encoder software. The selected training set will be described later. For each sequence in the training set, the parameters defined in Section 3.4 are extracted. Furthermore, the target PSNR value for every frame is available, as the encoded and decoded sequences can be compared with the original. The desired mapping algorithm thus uses *supervised learning* - in other words, for each presentation of inputs, the desired output is known as well.

In the following chapters, *artificial neural networks* (see Chapter 3.2) will be considered as the mapping algorithm, trying to reach maximum correlation of the estimated and real PSNR and to minimize the mean squared error of PSNR values. As the inputs differ vastly in case of Intra and Inter predicted pictures, each group will be treated separately, designing two different artificial neural networks.

Two sets of encoded sequences were constructed:

Training set. A set of 10 original sequences is used. Each sequence is encoded using the JM11 reference encoder with varying target bit rates. As the encoder allows us to define the initial quantization parameter for the sequence, this feature is used as well. This results in a set of the total 40 different encoded sequences. For each frame in each sequence, the PSNR is computed (the JM11 reference encoder computes the PSNR by default, the JM11 reference decoder computes the PSNR provided the original sequence is available).

Evaluation set. For evaluation of the designed mapping algorithm, an evaluation set is constructed. It is a set of 7 different video sequences, each encoded with four different encoder settings. This results in a set of 28 different sequences. Again, the desired PSNR is computed, but the mapping algorithm (the artificial neural network) does not have access to these values. After the network makes its estimation of PSNR, the estimated value will be compared to the real one.

Both the training set and the evaluation set consist of short sequences in CIF resolution (352×288 pixels). More details on the sequences can be found in Sec. 4.1.

3.5.1 Intra Predicted pictures, linear network

Let us begin with the simplest configuration of an artificial neural network. As the outputs should be in a continuous range of values rather than discrete, the perceptron configuration with thresholding transfer function is not usable. A linear unit will be used instead. As noted in Sec. 3.2.1, every network made up of linear units may be substituted by a single linear unit with equal performance. For the input parameters defined in Sec. 3.4.1, the situation is shown in Fig. 3.18. At this point, with video sequences compressed in the High profile will be considered as it is the only profile that uses all the macroblock types inlcuding the Intra 8×8 predicted blocks. Extension to other profiles will be discussed later. The used H.264/AVC reference encoder settings are identical to those given with the software as examples for the High profile video encoding [33]. All the input parameters are normalized to stay in the range from 0 to 1

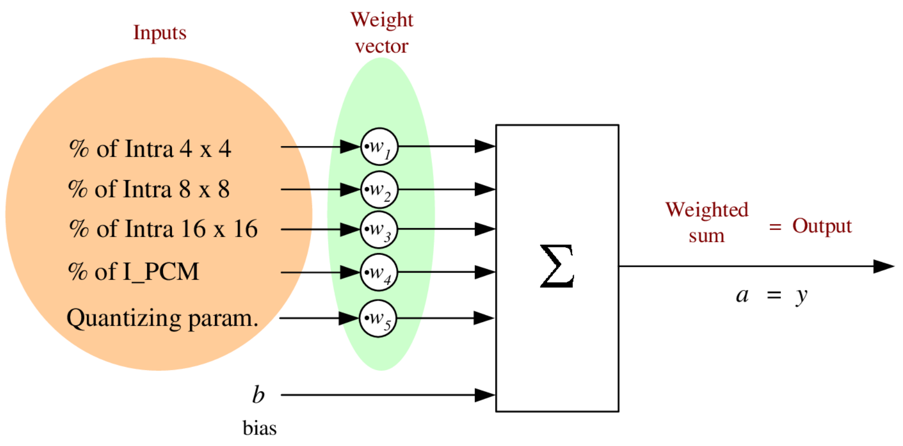

Figure 3.18: Linear neuron unit estimating PSNR for Intra predicted pictures.

Table 3.5: Weight and bias values for a linear unit trained with Intra predicted pictures

Input description	Input	Weight	Weight value
% of Intra 4 $\times$ 4 blocks / 100	x_1	w_1	11.75
% of Intra 8 $\times$ 8 blocks / 100	x_2	w_2	13.10
% of Intra 16 $\times$ 16 blocks / 100	x_3	w_3	22.41
% of I_PCM blocks / 100	x_4	w_4	0.000
Quantization parameter / 52	x_5	w_5	-43.98
		b	47.26

– the percentage inputs are divided by a factor of 100 and the quantization parameter is divided by 52, as it is the maximum that the quantization parameter can take according to the H.264/AVC standard. To train the linear unit, the MATLAB Neural Network toolbox is used, namely the function `train`, which trains the linear network using the gradient descent algorithm (see Sec. 3.2.2). Suppose the input vectors are available in matrix `input` and the desired network outputs (targets) are stored in vector `target`. The MATLAB code to train such network may then be as follows:

```
net = newlin([0 1;0 1;0 1;0 1;0 1],1);
net.trainParam.epochs = 2500;
net = train(net,input,target);
```

Table 3.5 shows the weight and bias values after 2500 epochs (iterations), where the minimal mean squared error for the training set is reached. The learning rate is $\eta = 0.01$.

For the estimated PSNR, we can write

$$PSNR_{Est} = 47.26 + x_1 \cdot 11.75 + x_2 \cdot 13.10 + x_3 \cdot 22.41 + x_5 \cdot (-43.98). \tag{3.32}$$

The description of inputs x_1 to x_5 can be found in Tab. 3.5. Obviously, the input x_4 has no effect as its weight is equal to zero. In the High profile configuration of the JM11 encoder that is used, no I_PCM blocks are present. However, this input is not discarded at this point as it may be useful when a different training set is used.

Fig. 3.19 shows how the mean squared error for the training and the evaluation set is changing with the increasing number of epochs. After 2500 epochs, we reach the mean squared error of the training set 0.681. For the evaluation set, the MSE after 2500 epochs

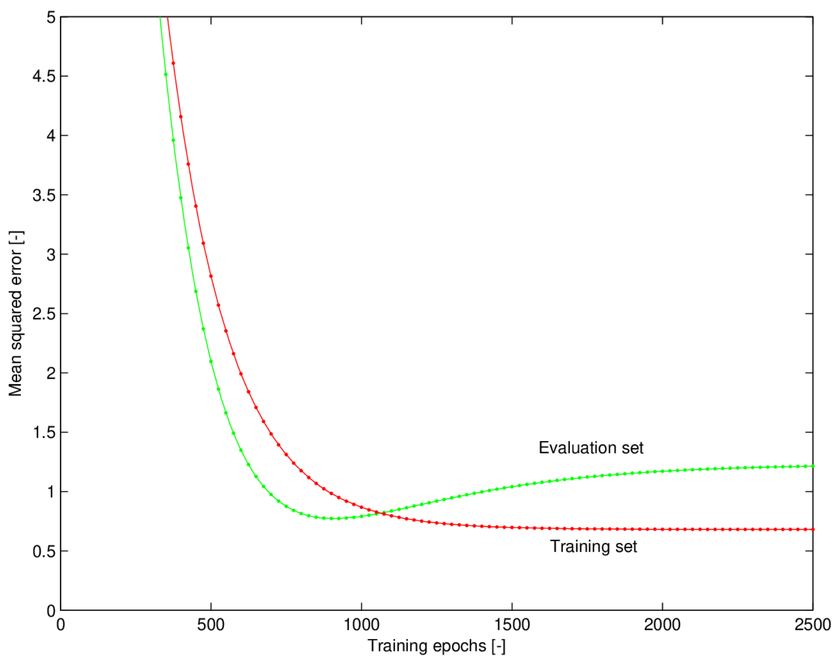

Figure 3.19: Mean squared error with increasing number of training epochs. Linear unit, Intra frames.

is as low as 1.214. It is obvious that after about 800 epochs, the minimum MSE for the evaluation set is reached and then it increases as the training goes on.

Fig. 3.20 shows a similar graph for the correlation coefficient between the real and the estimated PSNR. The correlation ceofficient for the training set is 0.983 after 2500 epochs. For the evaluation set, its value is 0.971 after 2500 epochs. Finally, Fig. 3.21 shows the scatter plot diagram of the real and the estimated PSNR for the evaluation sequence set.

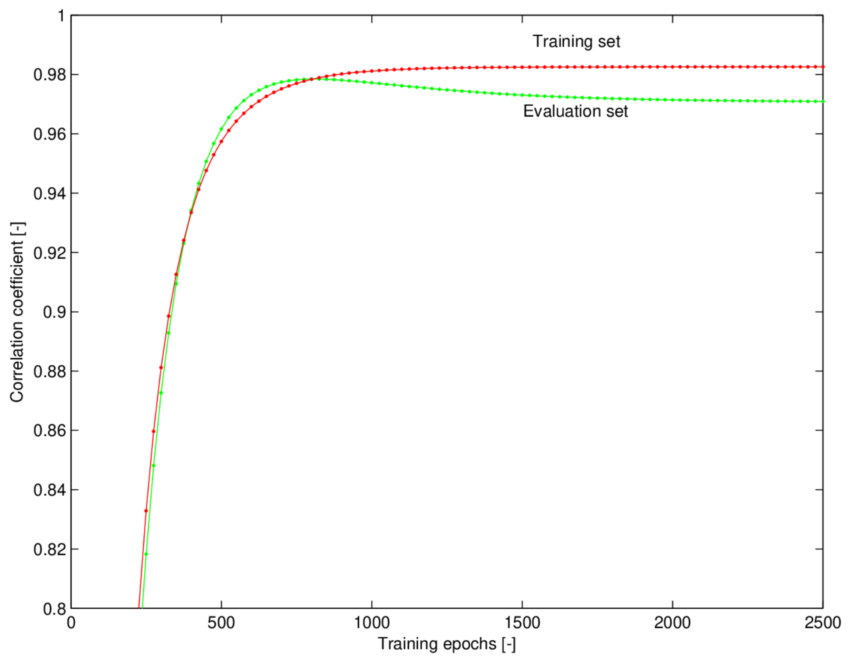

Figure 3.20: Correlation coefficient with increasing number of training epochs. Linear unit, Intra frames.

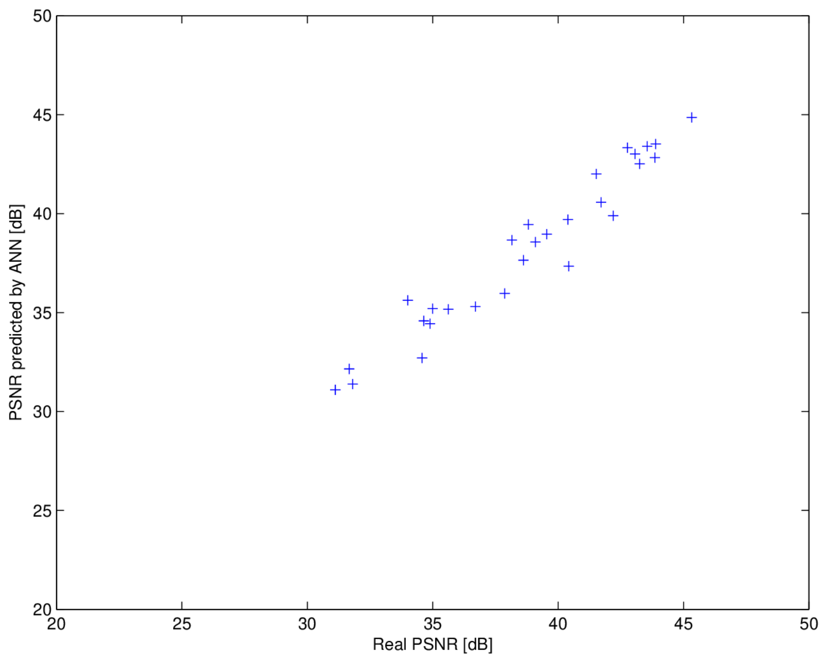

Figure 3.21: Scatter plot diagram: Estimated PSNR values plotted versus real values for evaluation set. Linear unit, Intra frames.

Table 3.6: 2-layer sigmoid unit network: results.

	Training set	Evaluation set	
Configuration	MSE	MSE	Correlation coef.
1 logsig - 1 lin	0.682	1.408	0.9657
2 logsig - 1 lin	0.703	1.513	0.9627
3 logsig - 1 lin	0.702	1.457	0.9641
4 logsig - 1 lin	0.711	1.547	0.9616
5 logsig - 1 lin	0.621	1.059	0.9728
1 tansig - 1 lin	2.722	2.073	0.9494
2 tansig - 1 lin	0.702	1.217	0.9673
3 tansig - 1 lin	0.681	1.386	0.9663
4 tansig - 1 lin	0.690	1.431	0.9610
5 tansig - 1 lin	0.656	1.192	0.9702

3.5.2 Intra predicted pictures, sigmoid unit network

As linear neuron units have only a limited ability to get along with more complicated classification tasks, let us experiment with a more complicated network using different neuron units and see if we can reach an improvement in prediction performance.

Several two-layer artificial neural networks were constructed, with the first layer made up of a variable number of sigmoid units and the second layer created by a single linear unit (only one unit is in the second layer as we want the network to output only one value). Again, the networks are trained in MATLAB, using the gradient descent algorithm. The performance of the network, measured as the correlation coefficient of the estimated and target PSNRs of the evaluation set can be seen on Fig. 3.22 and Fig. 3.23 developing over the training process. Please note that the networks are trained to minimize the mean squared error over the training set of video sequences. The correlation coefficient is used as a measure of generalization ability of the network. Networks having one to five sigmoid units in the first layer [15] are considered. Furthermore, two different sigmoid transfer functions are used, both available in MATLAB [4]: the `logsig` transfer function, given by Eq. 3.9, and the `tansig` transfer function, given by

$$\sigma(a) = \frac{2}{1+e^{-2n}} - 1, \tag{3.33}$$

where a is the weighted sum in the unit (see Fig. 3.7). The two functions are quite similar in shape. The main difference is that the tansig function may have negative outputs.

The training process is quite slow, results in the first 6000 epochs are shown in the graphs. However, all the networks are trained until 20000 epochs are reached.

It is important to note that there is a lot of unstability in the training process as every time the network is created, different initial weights and biases are set. Therefore the convergence can not always be assured for the gradient descent algorithm.

3.5.3 Intra predicted pictures, multi-layer perceptron

Adding one more layer to the network presented in the previous section leads to the multi-layer perceptron configuration. The first – input layer, commonly has as many units as

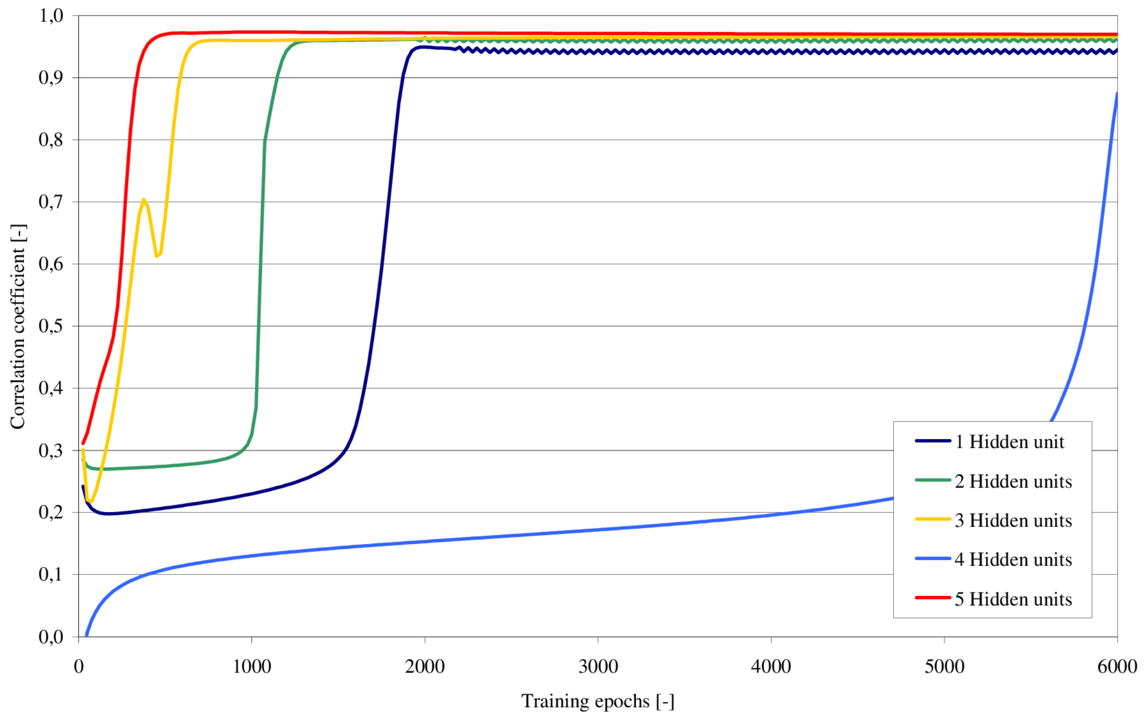

Figure 3.22: The training process of an ANN having a variable number of *tansig* units in the hidden layer.

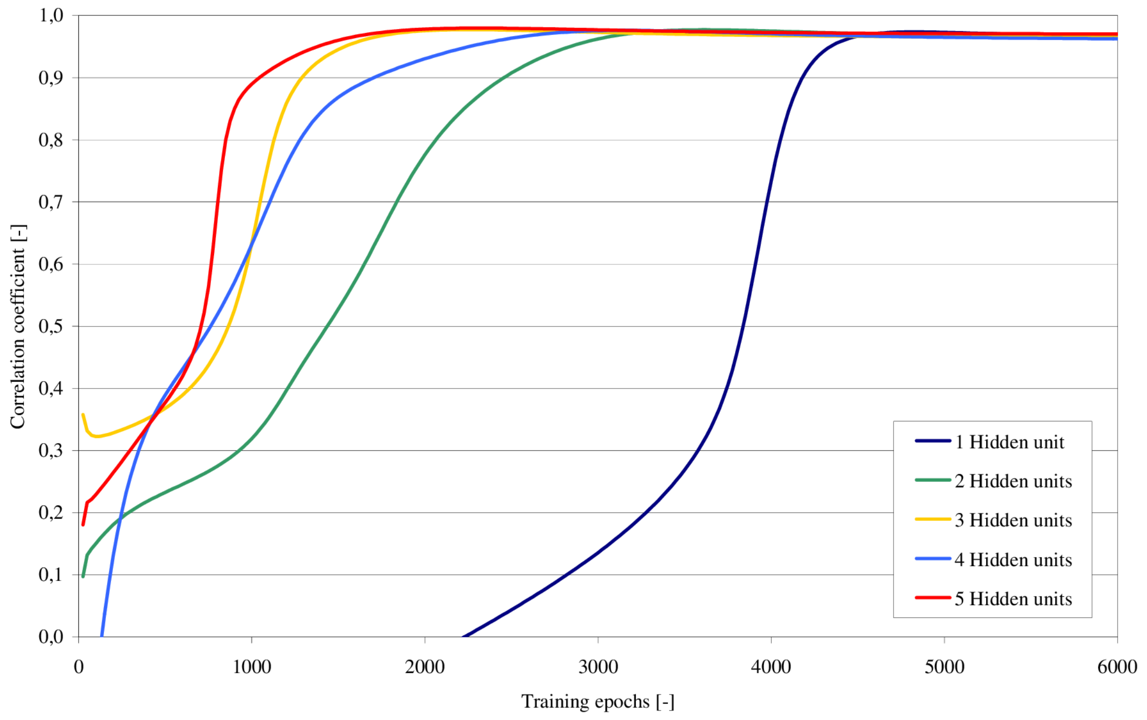

Figure 3.23: The training process of an ANN having a variable number of *logsig* units in the hidden layer.

there are inputs to the network [2]. The second – hidden layer has variable number of units trying to get the best possible results. In this section, let us experiment with the number of hidden units and try to reach the best possible results. Finally, the last – output layer has as many units as there are desired outputs from the system, in this case only one value is required thus a single unit will be used.

Similarly to the two-layer networks, units with `tansig` and `logsig` transfer functions are considered. Rather than constructing a whole network from units of one type, one transfer function is used for the input layer neurons and the other for the hidden layer neuron units. The training process is the same as for the previous network configurations. The results obtained for networks with 5 tansig units in the input layer, a variable number of logsig units in the hidden layer and one linear unit in the output layer are listed in Tab. 3.7. The best results are reached with eight units in the hidden layer. The results for such network are typed in bold in Tab. 3.7. Such network can be created and trained in MATLAB using the following commands:

```
net = newff([0 1;0 1;0 1;0 1;0 1],[5 8 1],
    'tansig' 'logsig' 'purelin','traingd','learngdm','mse');
net.initFcn = 'initlay';
net.layers1.initFcn = 'initnw';
net.trainParam.epochs = 20000;
net.trainParam.lr = 0.005;
net = train(net,input,target);
```

However, initial conditions influence the training a lot and even though special care is taken to set appropriate initial conditions before the training (using the settings of `net.initFcn` and `net.layers1.initFcn` parameters), the results are not the same when performing the training twice for a selected network configuration. We also experimented with 5 logsig units in the input layer and a variable number of tansig units in the hidden layer. The results are far not as good as those obtained with tansig units in the input and logsig units in the hidden layer (networks having 1-3 hidden tansig units could not be trained to give reasonable outputs, best results were achieved with nine hidden units – MSE over the training set 0.5698, MSE over the evaluation set 0.5956 and the correlation coefficient for the evaluation set 0.9833).

Matrices 3.34 – 3.39 give the network weights and biases corresponding to the network that achieved the best results: A multi-layer perceptron with 5 tansig units in the input layer, 8 logsig in the hidden layer and one linear unit in the output layer. The notation corresponds to Eq. 3.3.

$$\mathbf{W_1} = \begin{bmatrix} -0.5359 & -2.0997 & -0.2520 & -3.1786 & -0.3937 \\ 0.0554 & -1.3242 & 2.4851 & -0.8467 & -1.3157 \\ -1.3493 & 0.6811 & -0.6804 & 0.1193 & -3.1576 \\ 0.5849 & 2.9838 & 2.4966 & 1.6820 & 0.1193 \\ 2.4489 & 2.1919 & 1.6106 & 0.4137 & -3.8296 \end{bmatrix}, \tag{3.34}$$

$$\mathbf{b_1^T} = \begin{bmatrix} 5.1307 & 1.2507 & 2.5854 & -2.2952 & -0.9445 \end{bmatrix}, \tag{3.35}$$

Table 3.7: Multi-layer perceptron: results.

	Training set	Evaluation set	
Configuration	MSE	MSE	Correlation coef.
5 tansig - 1 logsig - 1 lin	0.860	1.700	0.9521
5 tansig - 2 logsig - 1 lin	0.593	0.729	0.9786
5 tansig - 3 logsig - 1 lin	0.658	0.511	0.9839
5 tansig - 4 logsig - 1 lin	0.621	1.480	0.9647
5 tansig - 5 logsig - 1 lin	0.567	0.789	0.9792
5 tansig - 6 logsig - 1 lin	0.638	0.543	0.9836
5 tansig - 7 logsig - 1 lin	0.570	0.698	0.9818
5 tansig - 8 logsig - 1 lin	**0.541**	**0.584**	**0.9835**
5 tansig - 9 logsig - 1 lin	0.647	0.907	0.9760
5 tansig - 10 logsig - 1 lin	0.670	0.880	0.9805

$$\mathbf{W_2} = \begin{bmatrix} -1.8854 & 2.8590 & 0.4646 & 3.3502 & 3.9028 \\ 2.7528 & -0.3113 & 3.4268 & -2.7093 & 2.7022 \\ 2.0718 & 1.3019 & -1.0678 & 3.1970 & -1.8585 \\ 4.0367 & -0.0438 & -2.7313 & 0.2466 & -0.1005 \\ 1.1069 & 2.6695 & 1.3508 & 1.3090 & 3.3722 \\ -2.5610 & 1.4260 & 2.4003 & -1.8859 & -0.6846 \\ 1.5111 & -2.0741 & -1.7041 & -0.9599 & -2.8877 \\ 2.0874 & -0.3491 & -1.6028 & -2.7243 & -1.9105 \end{bmatrix}, \tag{3.36}$$

$$\mathbf{b_2^T} = \begin{bmatrix} 3.3692 & -2.2772 & -1.5902 & 0.6252 & 1.3665 & -1.5697 & 3.8134 & 4.2850 \end{bmatrix}, \tag{3.37}$$

$$\mathbf{W_3} = \begin{bmatrix} 7.1935 & 7.1380 & -1.1475 & 6.1820 & 5.3185 & 1.0553 & 6.6130 & 7.8723 \end{bmatrix}, \tag{3.38}$$

$$\mathbf{b_3} = \begin{bmatrix} 6.324 \end{bmatrix}. \tag{3.39}$$

For the first layer of neurons, the matrix in Eq. 3.34 gives the input weights, each row in the matrix representing the weights of one neuron. Similarly, each value in Eq. 3.35 represents a bias for one neuron unit. To obtain outputs $\mathbf{l_1}$ from the input layer (a column vector), multiplication is performed as

$$\mathbf{l_1} = \mathbf{W_1 x} + \mathbf{b_1}, \tag{3.40}$$

where the elements in the input vector $\mathbf{x}$ are defined in Tab. 3.5. The outputs of subsequent layers are then computed in a similar manner.

So far, only sequences coded in High profile H.264/AVC were considered. It is the only profile that uses 8×8 blocks for Intra prediction. The other profiles are limited to the remaining modes, however they are quite unlikely to use the I_PCM prediction mode in most encoder implementations. For all but the High profile, we come to three parameter determining the PSNR for Intra frames - the percentage of 16×16 blocks, percentage of 4×4 blocks and the quantization parameter value. The number of input parameters is limited compared to the High profile and a certain loss of PSNR prediction accuracy can be expected even though the considerations and networks mentioned above are suitable universally for all the profiles.

Let us now use the best performing network from Tab. 3.7 to estimate PSNR of Intra frames coded in the Baseline, Main and Extended profiles. To make the encoder work in the selected profiles, several parameters need to be modified in the input configuration file. For the Extended profile, for instance, the parameters would be as follows:

```
ProfileIDC = 88 # Profile IDC
   #(66=baseline, 77=main, 88=extended; FREXT Profiles:
   # 100=High, 110=High 10, 122=High 4:2:2, 144=High 4:4:4)
LevelIDC = 20 # Level IDC (e.g.  20 = level 2.0)
Transform8x8Mode = 0 # (0:  only 4x4 transform,
   # 1:  allow using 8x8 transform additionally, 2:  only 8x8 transform)
```

In order to make the encoder work for Baseline profile, bi-directional prediction has to be disabled for Inter coded frames using the parameter

```
NumberBFrames = 0 # Number of B coded frames inserted (0=not used).
```

For further details on the encoder settings and encoder control using the file `encoder.cfg`, see [36].

In our experiments, the Baseline, Main, and Extended profile all code the Intra frames with the same results. Using the multi-layer perceptron network derived in 3.5.3, PSNR is estimated for all the profiles. The resulting correlation coefficient of the real and estimated PSNR for the evaluation set of sequences is 0.9600, and the mean squared error of the predicted values is 4.5294. These results are far not as good as those achieved for the High profile, but they are reasonable assuming the input set of parameters is in fact reduced by one quarter. The scatter plot diagrams of the real and the estimated PSNR values for video sequence Intra frames coded in the High and in the Extended profiles are shown in Fig. 3.24 and 3.25, respectively.

Finally, let us try how the linear unit derived in Sec. 3.5.1 behaves with the Extended profile coded frames, for instance. The correlation coefficient is lower compared to the results of the multi-layer perceptron and reaches 0.9424. On the other hand, the mean squared error decreases to 2.6949. The corresponding scatter plot diagram is given in Fig. 3.26.

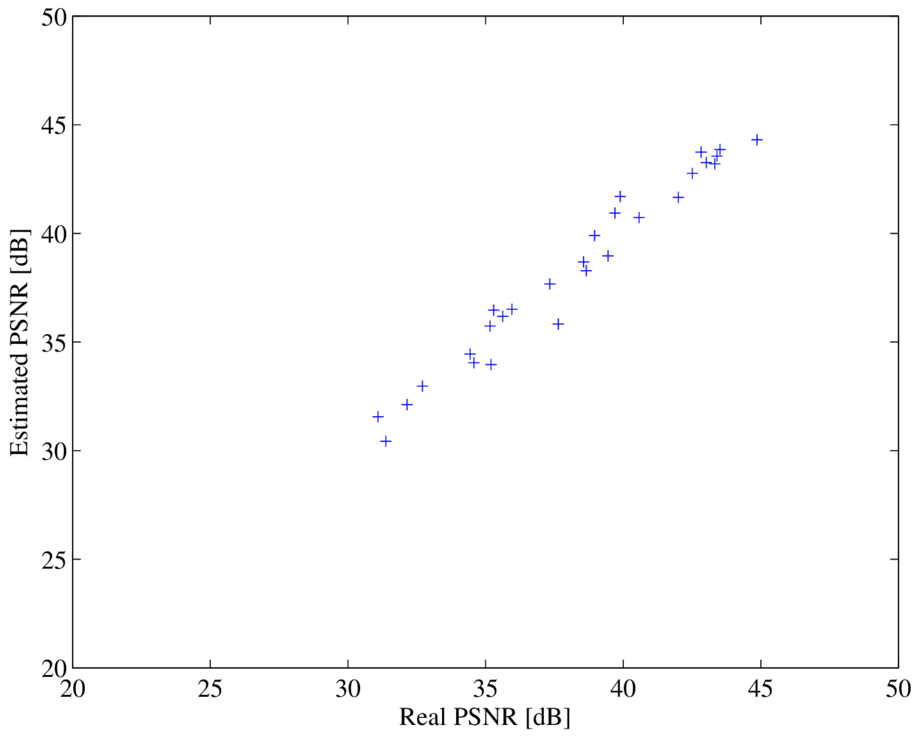

Figure 3.24: Scatter plot diagram: Real versus estimated PSNR values for Intra coded pictures from the evaluation set. High profile, multi-layer perceptron network.

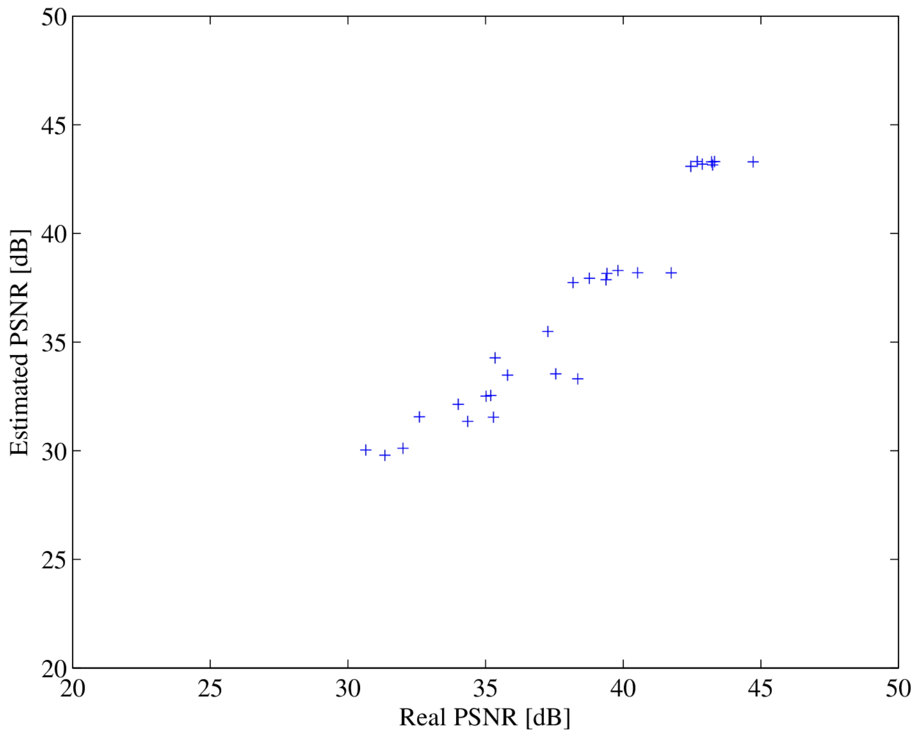

Figure 3.25: Scatter plot diagram: Real versus estimated PSNR values for Intra coded pictures from the evaluation set. Extended profile, multi-layer perceptron network.

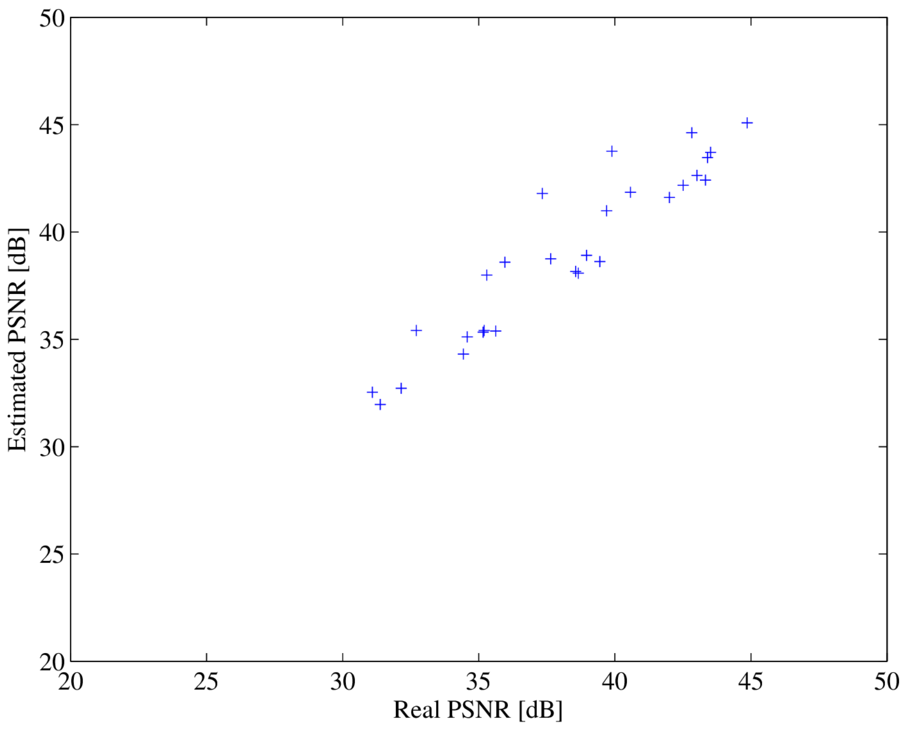

Figure 3.26: Scatter plot diagram: Real versus estimated PSNR values for Intra coded pictures from the evaluation set. Extended profile, linear network.

3.5.4 Inter predicted pictures

For the Inter predicted pictures, again two sets of video sequences are used - the training set and the evaluation set. However, as there is a slight difference in the required variability, the encoder settings are different from those used for the Intra frame network training. More details on the encoder settings can be found in Sec. 4.1.

When estimating the PSNR for Inter predicted pictures, the PSNR of the first frame in a sequence needs to be estimated first, as it is consequently used for reference. For this, the multi-layer perceptron is used as it reached the best results in the previous section. The artificial neural network inputs for Inter frame PSNR estimation are shown in Fig. 3.27.

To design the network, the reference PSNR approach presented in Sec. 3.4.2 is used (Eq. 3.30, 3.31). A flowchart of the process of preparing network inputs and outputs for training is displayed in Fig. 3.28. For Inter frames, the situation is complicated compared to the Intra frames, as the reference PSNR of those frames, from which the blocks within the actual frame are predicted, are needed. In the training phase, the real PSNR of each frame is available. The real PSNR can thus be put in Eq. 3.30 and 3.31. The algorithm shown in Fig. 3.28 is performed for each frame in each sequence within the training set. The calculated parameters are then stored in a matrix to be used for batch network training.

When the artificial neural network is trained, its verification over the evaluation set can be done. The estimated PSNR values are reached with an algorithm similar to the one shown in Fig. 3.28. The most important difference is that the real PSNR of the sequence frames is not available any more, so what is used to calculate the reference PSNR for the Inter predicted data in Eq. 3.30 and 3.31 is the network's own output (estimated PSNR) related to the reference frame. Of course, the last block in the flowchart – real PSNR calculation – is replaced by neural network PSNR estimation when verifying the network performance. Unlike for Intra predicted pictures, the linear network configuration does not suffice for the estimation of PSNR for inter predicted pictures. In the Baseline profile, prediction is done only in one direction, and the reference PSNR for each of the Inter

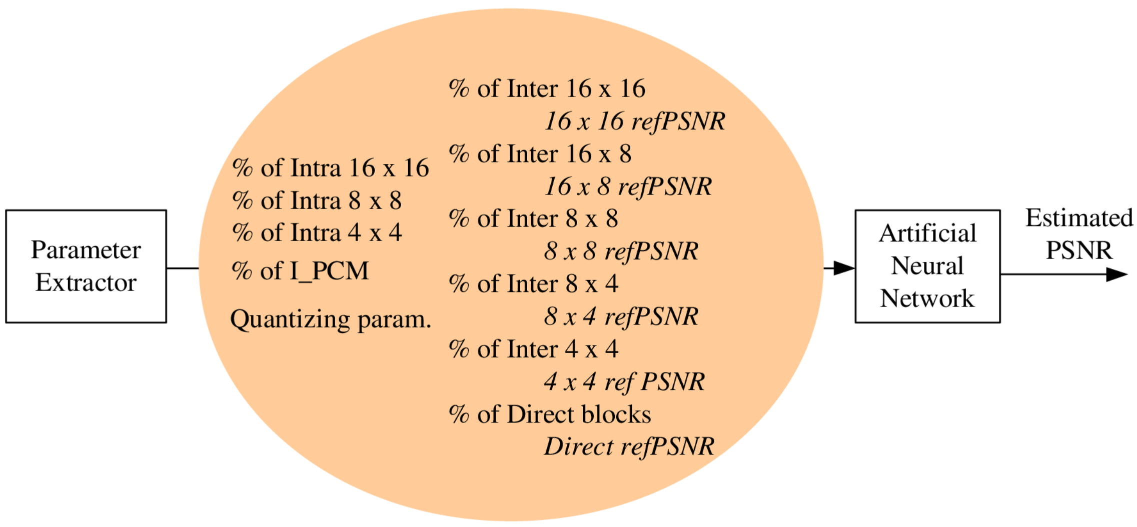

Figure 3.27: Parameters needed for Inter picture PSNR estimation.

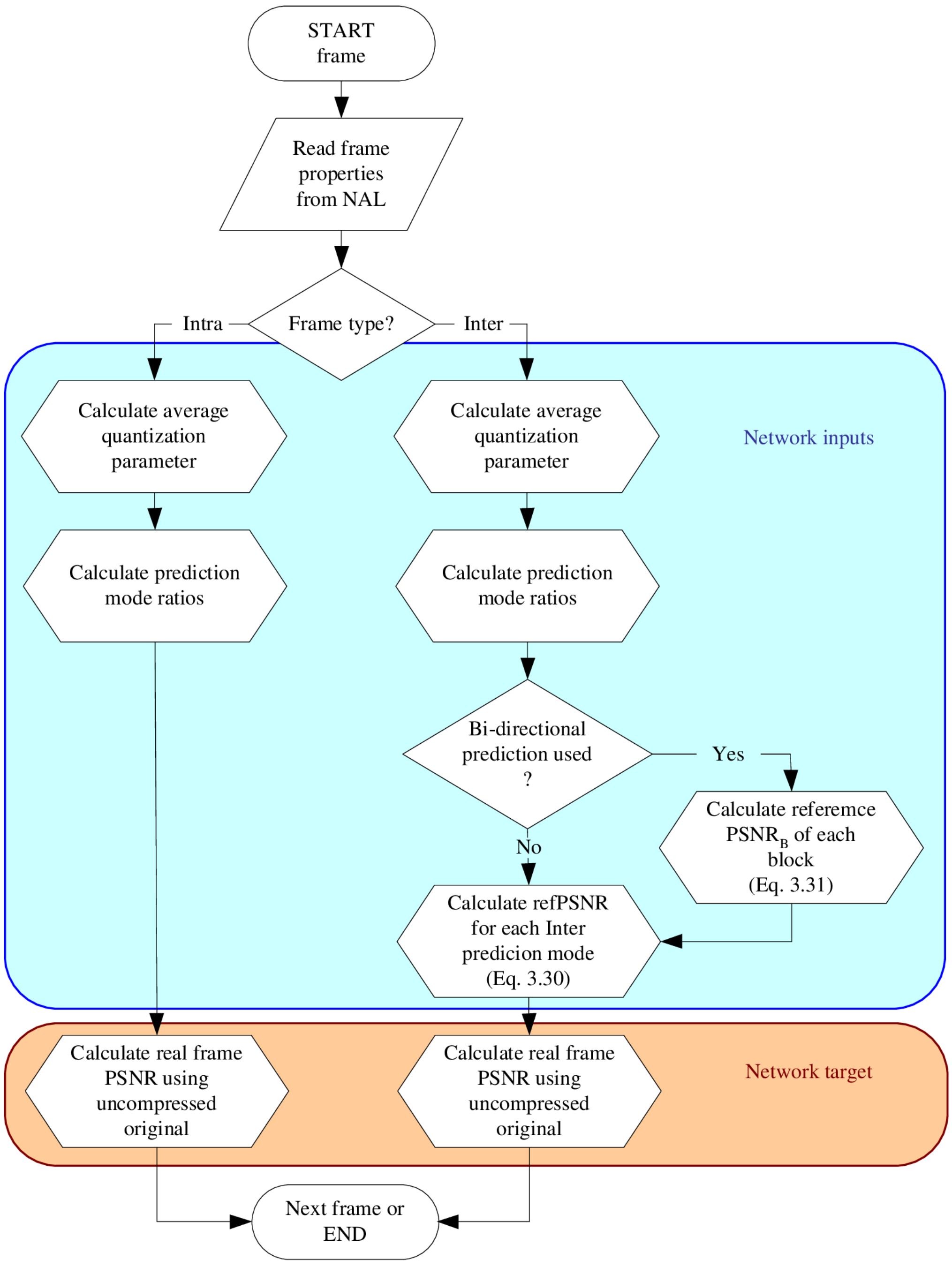
START frame
Read frame properties from NAL
Frame type?
Intra
Inter
Network inputs
Calculate average quantization parameter
Calculate prediction mode ratios
Calculate average quantization parameter
Calculate prediction mode ratios
Bi-directional prediction used ?
Yes
No
Calculate referemce PSNR$_B$ of each block (Eq. 3.31)
Calculate refPSNR for each Inter predicion mode (Eq. 3.30)
Network target
Calculate real frame PSNR using uncompressed original
Calculate real frame PSNR using uncompressed original
Next frame or END

Figure 3.28: Preparing inputs for training of an artificial neural network.

prediction block sizes is computed using Eq. 3.30. In the higher profiles, Eq. 3.31 is used first for each block in a frame predicted from two reference pictures. The consequent operations are the same for all profile configurations.The multi-layer perceptron is used as a mapping tool, with varying number of units in the hidden layer.

The first fifteen frames of the training set sequences are encoded and used for the training. The GOP (Group of pictures) format is IBPBPBP... (I – Intra coded frame, P – Inter coded frame, B – Inter coded frame with bi-directional prediction) Only one GOP per sequence is used. To verify the results, 15 frames of the evaluation set sequences are encoded with the same GOP settings. The GOP size of 15 frames is selected as it is the shortest GOP usually used for IPTV, for instance (the typical GOP lengths here are 15 – 250 frames). To verify the network performance for longer GOPs, the trained networks are used to estimate PSNRs of 30-frame GOPs as well. Different networks for each H.264/AVC profile are considered, and generalization ability of a network trained with the High profile is tested - whether or not it gives satisfactory results for different profile configurations.

One network per profile

In this configuration, both the training and the evaluation set sequences are encoded in the Baseline, Main, Extended and High profiles. The main aspects, as noted above, are that the Baseline profile does not use bi-directional prediction, and that the High profile is the only one to use Intra 8×8 predicted blocks.

For each profile, the total of 10 network configurations are trained. As the multi-layer perceptron with `tansig` transfer function units in the input layer and `logsig` transfer function units in the hidden layer reaches best results in our consideration for Intra frames, the same approach is used for Inter frames. As there are 17 parameters (see Fig. 3.27) as inputs to the network for Intra frames, all the considered networks have 17 `tansig` units in the input layer. The number of `logsig` units in the hidden layer is variable, reaching from one to ten. Finally, the output layer only includes one linear unit.

The results for the *Baseline* profile are listed in Tab. 3.8. There is one difference compared to all the other profiles - as the Baseline profile doesn't support bi-directional prediction, the GOP format is IPPPPP... instead of IBPBPBP... As expected, the mean squared error of estimated PSNRs in the training set is decreasing as the number of hidden units in the network increases. Rather more interesting are the values of the mean squared error and the correlation coefficient for the estimated PSNRs in the evaluation set. The MSE first decreases as new units are added to the hidden layer of the network. When a certain number of units is present in the hidden layer, the MSE values do not decrease monotonically any more, but stay in a certain range. Adding more units into the hidden layer is not likely to strongly improve the results for the evaluation set. This is why the tests are not performed for more than ten units in the hidden layer (it will be more obvious for other profiles that adding hidden units does not necessarily lead to improvement). However, for the Baseline profile, the best results are reached with ten units in the hidden layer. Fig. 3.29 shows the scatter plot diagram of the real and estimated PSNR values for the first 15 frames of the evaluation set sequences encoded in the Baseline profile. In this case, the evaluation set sequences are used in the same encoder configuration as the training sequences, having 15 frames in a GOP. To test the algorithm's stability for longer GOPs, let us test its behavior for twice as long GOPs –

Table 3.8: Inter coded pictures, Baseline profile: results.

	Training set	Eval. set 15 frames		Eval. set 30 frames	
Hidden units	MSE	MSE	Corr coef.	MSE	Corr coef.
1	0.2204	13.377	0.8460	17.710	0.8460
2	0.0514	28.954	0.6951	39.303	0.7295
3	0.0460	18.209	0.7640	18.573	0.8354
4	0.0351	9.910	0.8647	10.956	0.8978
5	0.0292	8.046	0.8921	8.987	0.9204
6	0.0286	9.524	0.8692	12.160	0.8872
7	0.0229	5.179	0.9318	6.562	0.9411
8	0.0235	6.198	0.9207	5.933	0.9497
9	0.0165	7.672	0.8991	8.962	0.9237
10	0.0141	4.931	0.9350	4.721	0.9576

Fig. 3.30 shows the scatter plot diagram in this case. The network performs reasonably well here, too. Ass seen in Tab. 3.8, both the correlation coefficient and the MSE have improved compared to the 15-frame GOP (this doesn't mean the algorithm generally performs better for longer GOPs, this depends on the video sequence itself).

Let us describe the *Main* and *Extended* profiles in a single paragraph. The reason is simple – both profiles are using the same features we are dealing with, and – with the encoder settings we are using – give similar, if not identical results. In our configuration, the encoder behaves identically for both profiles, which means it selects the same prediction modes and, moreover, the resulting PSNRs are equal. Tab. 3.9 and Tab. 3.10 give the PSNR estimation results for both profiles, taking both 15-frame and 30-frame GOPs into account. It is obvious that increasing the number of units in the hidden layer does not necessarily lead to improvement - the best results are reached for three hidden units in the Main profile and seven units in the Extended profile, but as the encoded sequences are identical, the difference is caused by different initial weights in the artificial neural networks rather than anything else. The results are something worse than those for the Baseline profile - for bi-directionally predicted frames, the algorithm's performance is lower.

The *High* profile supports bi-directional prediction and 8×8 Intra prediction. The results for the High profile are given in Tab. 3.11. The algorithm performs best with three hidden units, with the correlation coefficient reaching 0.8979 and the MSE as low as 5.523 for 15-frame GOPs. The scatter plot diagram for such configuration and 30-frame GOP is shown in Fig. 3.31. Fig. 3.32 shows how the real and estimated PSNR develops in time for the 30-frame GOPs. All the three plots represent values for the "coastguard sequence" (see Sec. 4), with different encoder settings. The plots a), b), c) conform to encoder settings no. 1, 2, 3 in Tab. 4.2, respectively.

One network over different profiles

In the previous considerations, the artificial neural networks were trained separately for each H.264/AVC profile. Let us now try to use one single network through different profiles and observe its performance. As the High profile supports in fact all the prediction modes available in the H.264/AVC standard, it is straightforward it should be used to

Table 3.9: Inter coded pictures, Main profile: results.

	Training set	Eval. set 15 frames		Eval. set 30 frames	
Hidden units	MSE	MSE	Corr coef.	MSE	Corr coef.
1	0.0856	9.927	0.7903	13.482	0.8319
2	0.0719	9.725	0.7990	26.159	0.6370
3	0.0626	7.912	0.8379	9.319	0.8842
4	0.0530	10.028	0.7909	13.679	0.8258
5	0.0424	9.452	0.8030	20.348	0.7347
6	0.0379	11.969	0.7465	16.321	0.7851
7	0.0338	10.006	0.7887	19.604	0.7379
8	0.0283	10.280	0.7821	15.740	0.8094
9	0.0297	7.790	0.8443	13.156	0.8342
10	0.0290	10.544	0.7793	23.787	0.7070

Table 3.10: Inter coded pictures, Extended profile: results.

	Training set	Eval. set 15 frames		Eval. set 30 frames	
Hidden units	MSE	MSE	Corr coef.	MSE	Corr coef.
1	0.0844	10.898	0.7686	15.566	0.8125
2	0.0770	10.334	0.7837	11.643	0.8576
3	0.0549	9.154	0.8088	12.908	0.8409
4	0.0530	11.456	0.7555	17.172	0.7920
5	0.0445	7.448	0.7599	13.976	0.8271
6	0.0377	11.169	0.8005	14.030	0.8220
7	0.0348	9.870	0.8813	8.862	0.8930
8	0.0292	5.988	0.7041	18.996	0.7463
9	0.0280	7.393	0.8496	8.509	0.8971
10	0.0249	11.137	0.7627	16.907	0.7829

Table 3.11: Inter coded pictures, High profile: results.

	Training set	Eval. set 15 frames		Eval. set 30 frames	
Hidden units	MSE	MSE	Corr coef.	MSE	Corr coef.
1	0.0730	7.106	0.8680	8.479	0.8962
2	0.0712	6.975	0.8695	7.735	0.9058
3	0.0533	5.523	0.8979	5.719	0.9314
4	0.0484	7.763	0.8552	10.720	0.8759
5	0.0463	7.448	0.8522	11.778	0.8568
6	0.0412	5.680	0.8997	6.290	0.9258
7	0.0322	6.975	0.8631	9.219	0.8865
8	0.0293	5.920	0.8986	8.993	0.8896
9	0.0298	8.597	0.8337	11.161	0.8624
10	0.0170	18.560	0.7320	21.450	0.7926

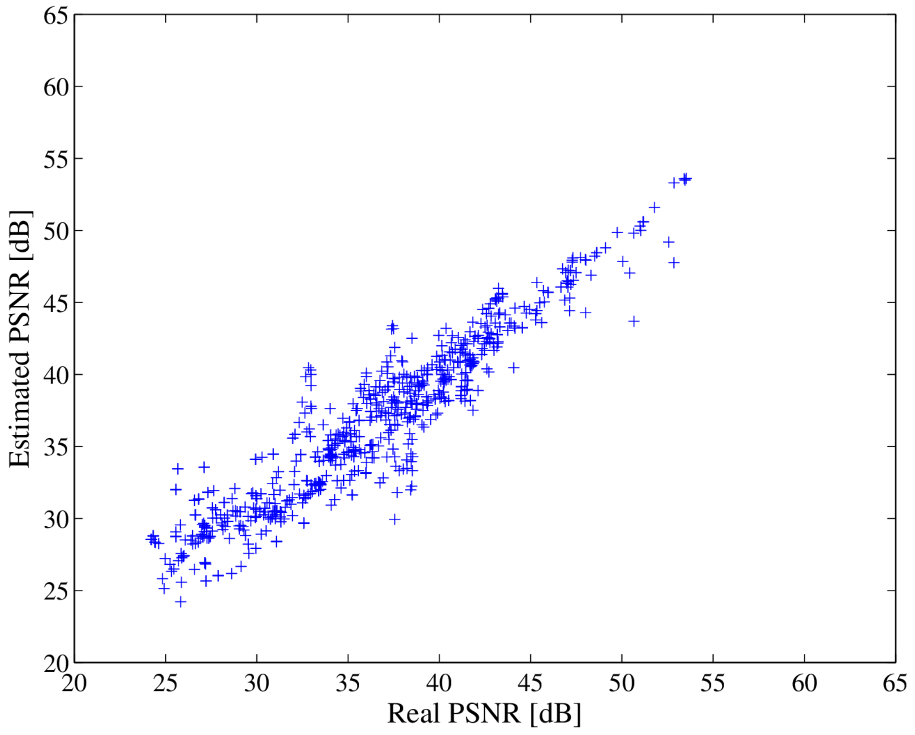

Figure 3.29: Scatter plot diagram: Estimated versus real PSNR values for Inter frames, 15 frames in a GOP. Baseline profile.

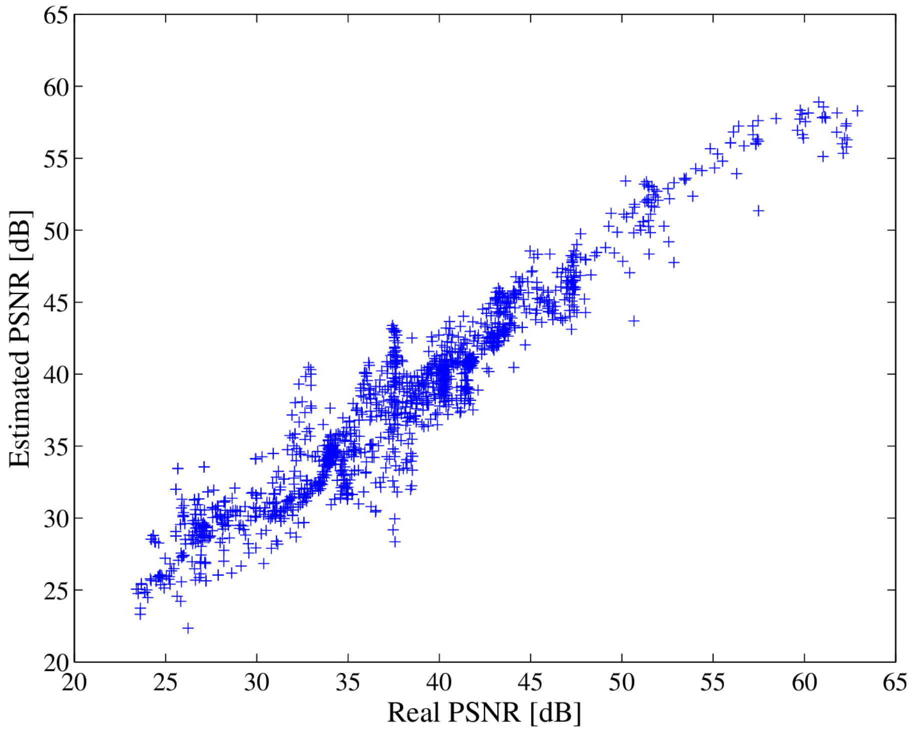

Figure 3.30: Scatter plot diagram: Estimated versus real PSNR values for Inter frames, 30 frames in a GOP. Baseline profile.

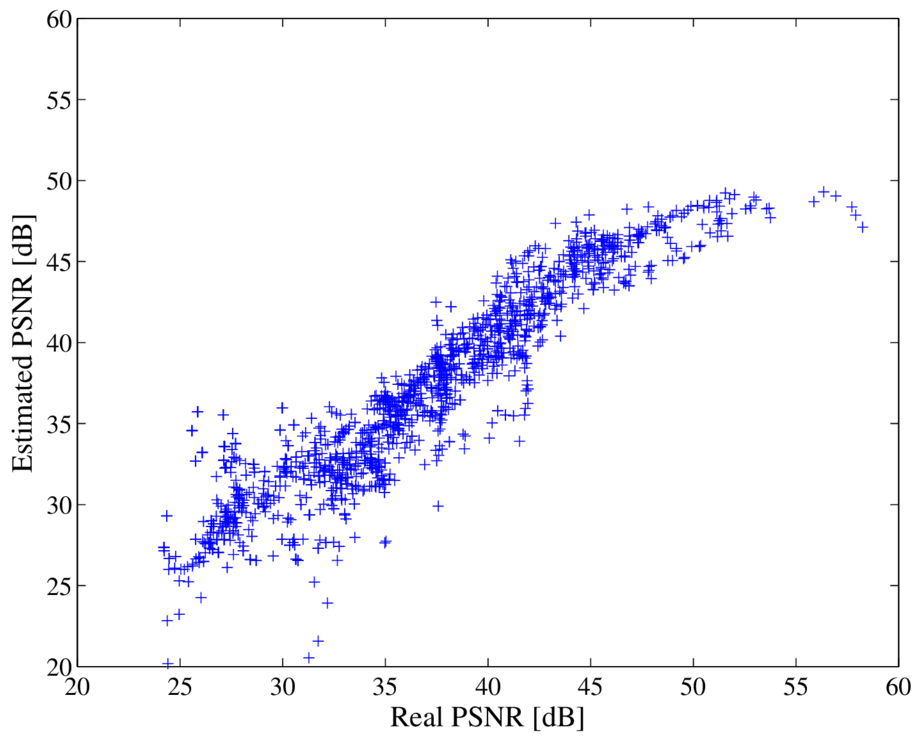

Figure 3.31: Scatter plot diagram: Estimated versus real PSNR values for Inter frames, 30 frames in a GOP. High profile.

train such universal network. Among the networks trained for the High profile (see Tab. 3.11), the network with three hidden units performs best. Using the network for video sequences compressed in the remaining profiles (15-frame GOPs) gives the following results:

Extended and Main profiles give identical results (the reasons were mentioned above). Using the network trained for High profile, the correlation coefficient reaches 0.8605 and the MSE is 9.0312. For the Baseline profile, the correlation coefficient and the MSE are 0.8716 and 11.4092, respectively.

Interesting results can be obtained when using a network trained for the Baseline profile with ten hidden units (the last row in Tab. 3.8). In this case, for the Extended and High profiles the correlation coefiicient and MSE are 0.8824 and 5.8800, respectively. For the High profile, the values are 0.8653 and 6.7384, respectively. Even though network trained for the profile supporting the least features is used at this moment, the results are surprisingly good for the "higher" profiles. The results are comparable to those obtained when both the training and evaluation of the network is done for a single encoder configuration.

A different network configuration

In the previous text, a multi-layer perceptron was considered with `tansig` transfer function units in the input layer and `logsig` transfer function in the hidden layer, assuming this configuration shall give better results as it gives the best results for Intra frames. We experimented with an inverse configuration as well (`logsig` units in the input layer, `tansig` units in the hidden layer), but no significant improvement was achieved.

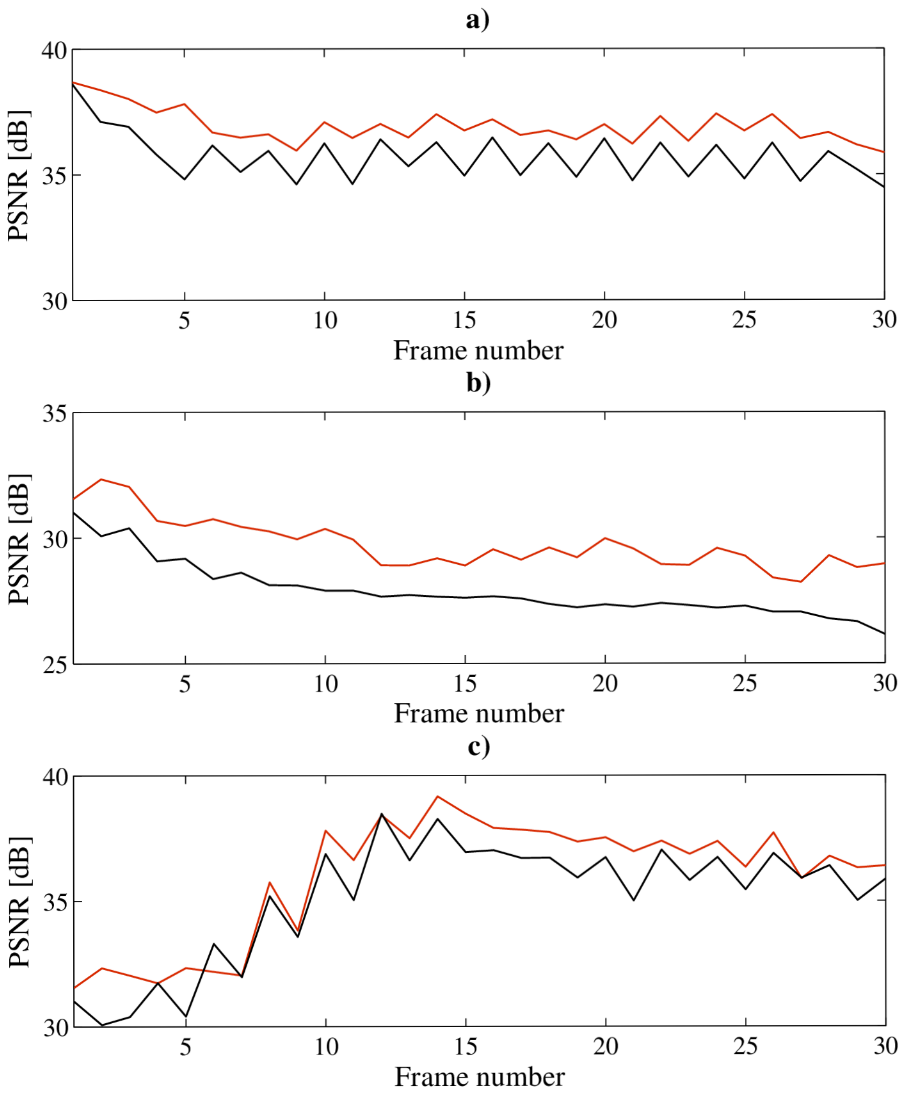

Figure 3.32: Real (black) and estimated (red) PSNR in the first 30 frames in a "coast-guard" sequence (Fig. 4.2b) with different encoder settings.

3.6 Conclusion

In Chapter 3, we presented a method to estimate PSNR values of compressed video sequences conforming to the H.264/AVC standard. The algorithm does not make use of the original (uncompressed) video material to compute the PSNR, and thus can be classified as a no-reference metric.

The whole metric framework, along with the design details and its performance tests are results of my own original research. They were published in [28, 29, 30, 31, 32].

Video Sequence Database 4

This section describes the set of video sequences used for the development and testing of the new metric for H.264/AVC. When designing the algorithm (Sec. 3), only sequences in low resolution are considered. For the performance testing (Sec. 5), a different set of video sequences is used, with resolution up to full HD (1920×1080 pixels, progressive). The video sequences will be described in this section and their characteristics will be considered.

4.1 Low Resolution Sequences (Metric Design)

For the design of the new metric and artificial neural network optimization, two sets of video sequences were constructed: the training set and the evaluation set. The training set consists of ten video sequences, while the evaluation set is made up of seven short video sequences. Special care is taken for both sets to cover a variety of characteristics – different content should be present in both, reaching from still scenes to scenes with fast motion, from smooth and low-detailed frames to complex and fine structures within a frame. Both the training and the evaluation sets include sport sequences, talking head sequences, nature views, etc. Fig. 4.1 shows the training sequences. For each sequence, frame 0, frame 10, frame 20 and frame 30 is shown to demonstrate the spatial activity or motion in a sequence besides the level of details within a frame. The evaluation sequences used to verify the network performance in the training process are displayed in Fig. 4.2.

All the low resolution video sequences are freely available on the internet [1]. They are progressive coded in CIF resolution (352×288 pixels). The color coding is 4:2:0. However, we do not use chroma components in our considerations so in Fig. 4.1 and in Fig. 4.2, only grayscale images are displayed.

Each sequence is encoded with variable encoder settings. The aim is to alter the encoder parameters in such manner that the encoder changes its decision on prediction modes and, of course, the sequence frames are coded with different PSNRs. For Intra frames, it was observed that the only parameter influencing the encoder's decision on the prediction modes is the quantization parameter (QP). Tab. 4.1 lists the QP values used in our experiments for Intra frames. The situation is different for Inter frame encoding – another parameter is used to control the encoder behavior, the "Target bitrate" parameter. The list of the seven encoder configurations is given in Tab. 4.2. Please note that these are profile independent configurations. As the networks are trained for different profiles in Sec. 3.5, the configurations listed in Tab. 4.1 and Tab. 4.2 are used for each profile separately.

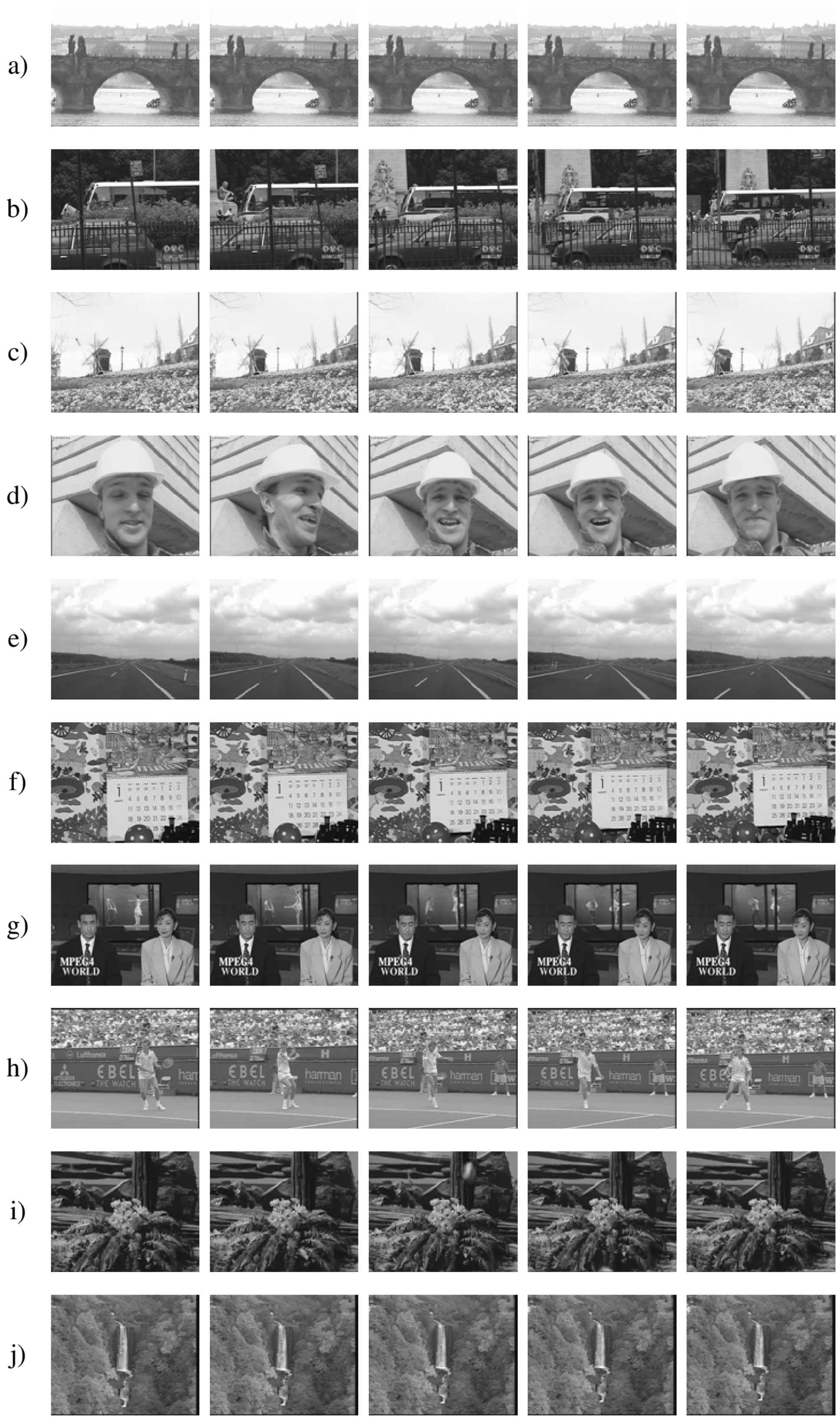

Figure 4.1: CIF resolution sequences used for the training of the artificial neural networks.

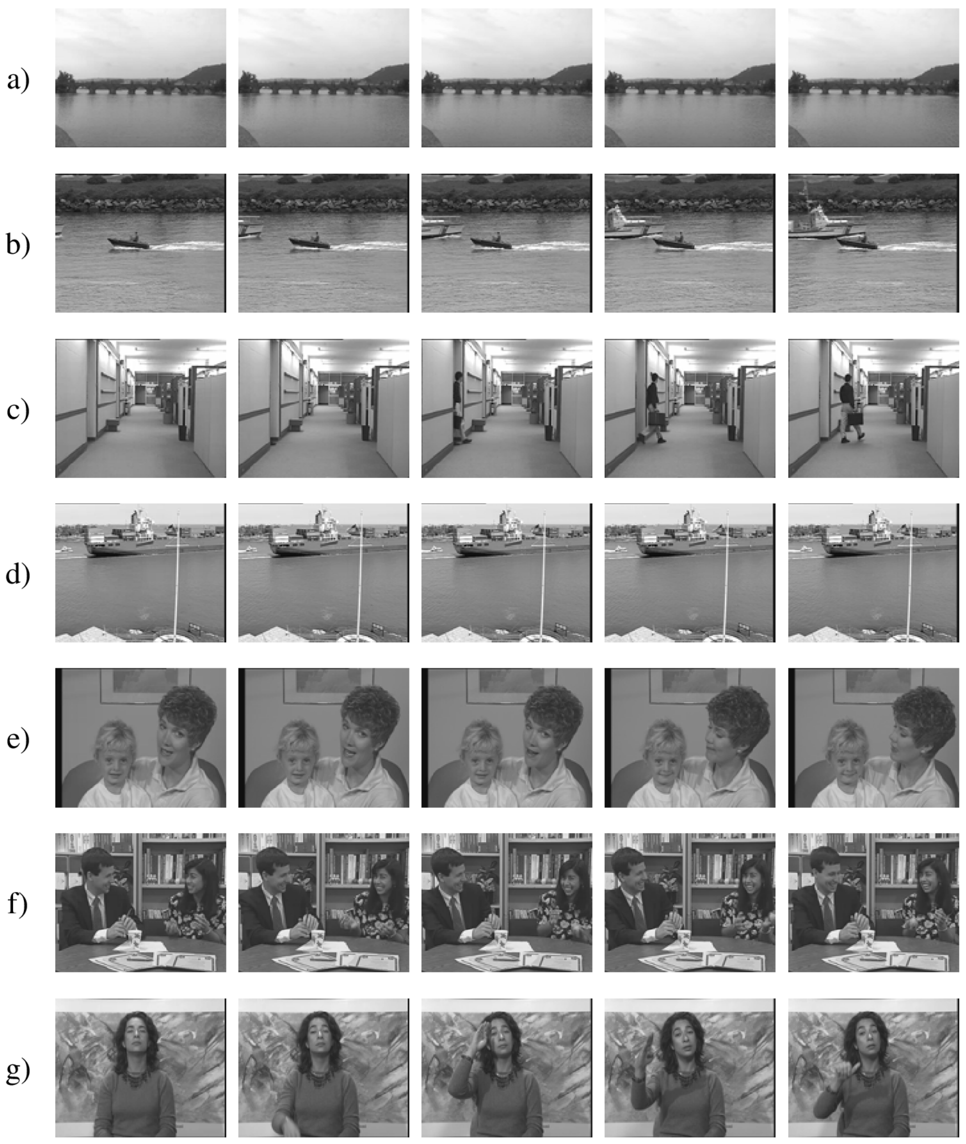

Figure 4.2: CIF resolution sequences used for the evaluation when designing the artificial neural networks.

Table 4.1: Variable encoder parameters used for training an artificial neural network for Inter frame PSNR estimation.

Config no.	Initial QP
1	20
2	25
3	30
4	35

Table 4.2: Variable encoder parameters used for training an artificial neural network for Intra frame PSNR estimation.

Config no.	Initial QP	Target bitrate [kbps]
1	25	100
2	25	1000
3	35	100
4	35	1000
5	45	100
6	45	1000
7	45	5000

4.2 Sequences for Performance Analysis

Another set of video sequences was constructed for performance tests of the metric. To be able to test the metric performance in different conditions including varying video resolution, the test set consists of five sequences with resolution up to full HD (1920×1080 pixels, progressive - 1080p). The five video sequences are displayed in Fig. 4.3, with the first frame in the sequence on the left, frame 5 in the center and frame 10 on the right. The sequences are publicly available [8].

A list of the available format is given in Tab. 4.3. A note 'original' with a certain format says that the sequence was taken as is from [8]. Some formats are derived from the 1080p (the 576p format is downsampled from 1920×1080 to 1024×576 pixels and then cropped to 4:3 aspect ratio at 720×576). Regarding the color coding – all the sequences are in 4:2:0 chroma sampled, but again as we are only considering luma pixel values, the chroma format is not important and not mentioned in Tab. 4.3.

Table 4.3: Performance analysis video sequences: available formats.

Format ID	Aspect	Resolution	format	fps	note
1080p	16:9	1920×1080	progressive	50	original
720p	16:9	1280×720	progressive	50	original
576p	4:3	720×576	progressive	50	scaled, cropped from 1080p
288p	4:3	352×288	progressive	50	scaled, cropped from 576p

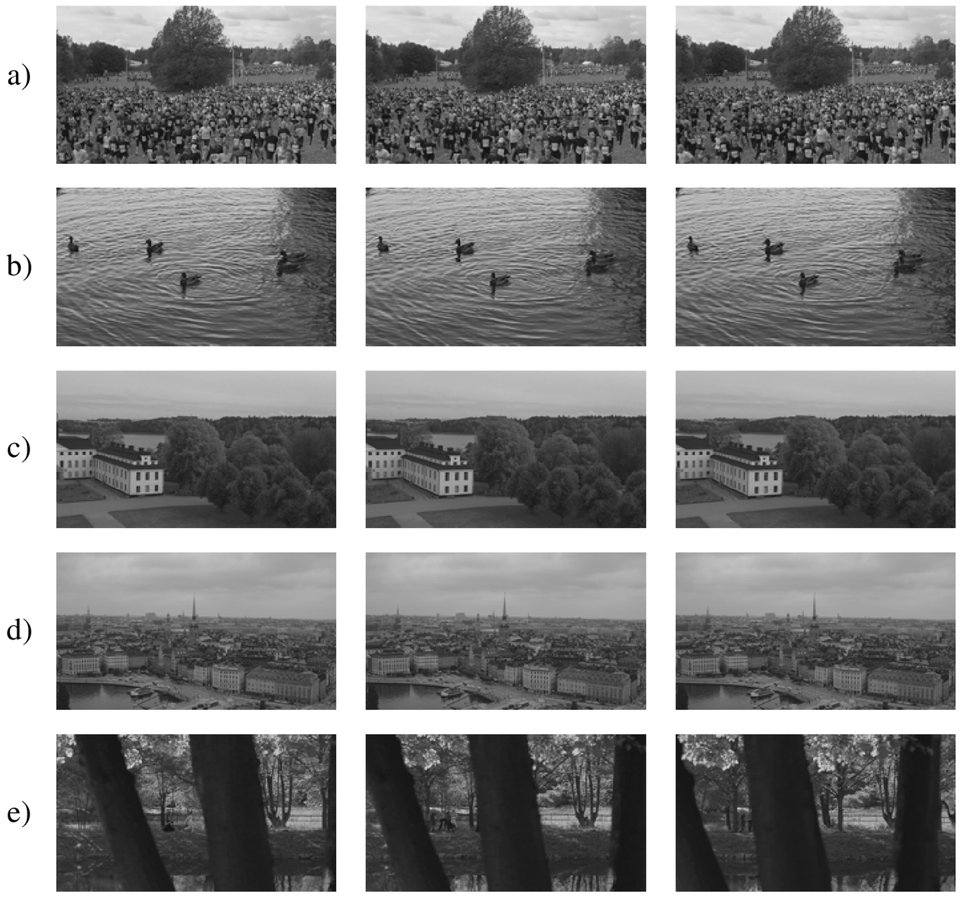

Figure 4.3: High definition video sequences.

Performance Analysis 5

In this section, the proposed PSNR estimation algorithm will be tested for different encoder settings and for different resolution of the processed video. The artificial neural network configurations are fixed for a given encoder profile as derived in Sec. 3.

5.1 Impact of Resolution

All the experiments described in Chapter 3 were done on CIF format encoded video sequences - their resolution is 352×288 pixels. Such video sequences are likely to be used in mobile video transmission, such as DVB-H. Anyway, as the area where the H.264/AVC codec is used is very broad, let us try to extend our considerations for higher resolution videos.

To verify the results for video sequences with higher resolution, video sequences encoded in 288p, 576p, 720p and 1080p formats (see Tab. 4.3) are used. It is clear that the sequences can hardly use the bitrate constraints defined for CIF (288p) sequences in Tab. 4.2. Our solution is in raising the target bitrate for each format in a defined manner – let us give an example for 576p sequences. Their resolution is 720×576 pixels, which means each frame consists of roughly four times as many pixels as a 288p frame. The target bitrate is thus quadrupled for such sequences.

288p. To verify the performance for higher resolution video sequences, the sequences used in Chapter 3 can not be used as they are only available in CIF (288p) format. Instead, HDTV sequences shall be used, as described in Tab. 4.3 and Fig. 4.3. Performance of 288p sequences should thus be evaluated as well for comparison purposes. Using the same settings as those used in Tab. 4.2, the sequences are encoded using the High profile H.264/AVC encoder.

576p,720p,1080p. The higher resolution sequences were also encoded with High profile H.264/AVC encoder. Again, the settings in Tab. 4.2 are used. However, as the video frames include more pixels than the 288p frames, the target bitrate has to be changed accordingly in order to achieve comparable bit per pixel values. For example, the 576p format has $720 \times 576 = 414720$ pixels in a frame, which is approximately 4.09 times more than in the case of 288p format with $352 \times 288 = 101376$ pixels. This means four times higher target bitrates are used for the 576p compared to those listed in Tab. 4.2 for 288p.

Tab 5.1 shows the correlation coefficient and the mean squared error of the real and the estimated PSNRs for video sequences with different resolutions encoded with High profile H.264/AVC encoder. For the PSNR estimation, the artificial neural network with three hidden units trained on High profile sequences is used (see Tab. 3.11). The results for high resolution sequences are comparable (in fact, slightly better) to those obtained for 288p sequences, which is the format the network was trained on. The scatter plot diagrams for the 288p sequences and 1080p sequences are shown in Fig. 5.1 and Fig. 5.2,

Table 5.1: PSNR Estimation results for video sequences with different resolution.

Format ID	MSE	Corr coef.
288p	8.165	0.8884
576p	5.910	0.9113
720p	4.291	0.9202
1080p	5.711	0.8593

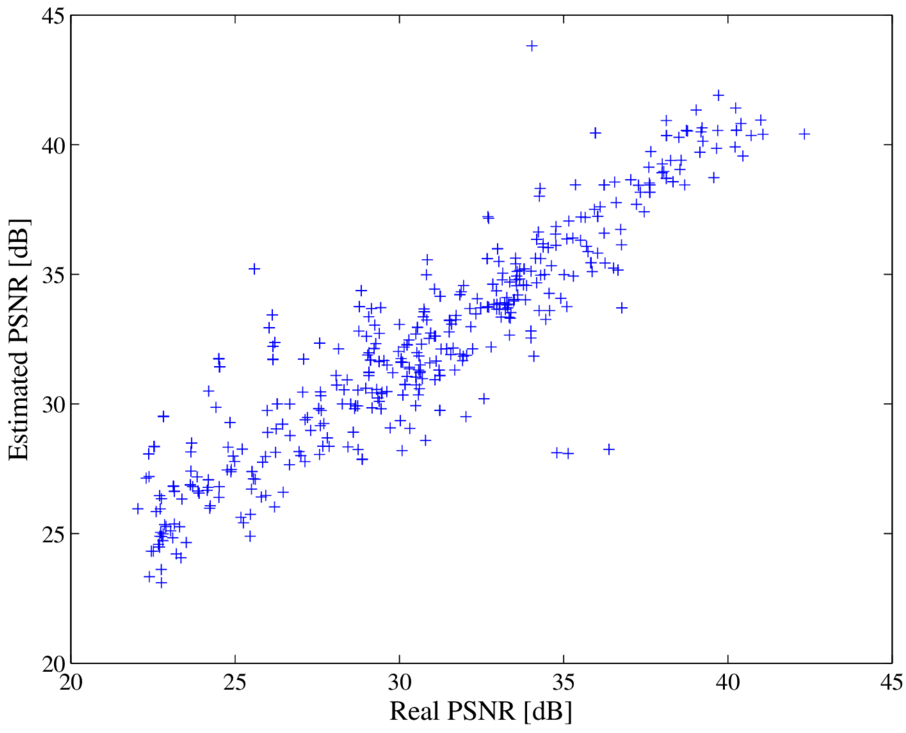

Figure 5.1: Scatter plot diagram: Estimated versus real PSNR values for 288p video sequences.

respectively, illustrating the difference in the two extreme cases of low and high resolution. The scatter plot diagram for 720p sequences, which reached the best results, is shown in Fig. 5.3. The network trained on Baseline profile sequences was also tested (10 hidden units, Tab. 3.8), but the results were significantly worse (MSE in the range 6.830 – 12.291, correlation 0.7528 – 0.8402). Some results for the HDTV video sequences were published in [29]. As fewer encoder configurations were used for evaluation in the paper, the results are slightly different.

5.2 Impact of Cut in a Sequence

Let us now test the performance of the algorithm when processing a video sequence with a cut. To form such sequence, two short parts of sequences were merged, 25 frames each, resulting in a 50-frame video sequence with a cut right in the middle. The sequence in the beginning is the "paris" sequence (Fig. 4.2f), the second half of the sequence with cut is the "hall" sequence (Fig. 4.2c). These particular sequences are selected in order to be coded with significantly different resulting PSNR even when the encoder settings are the

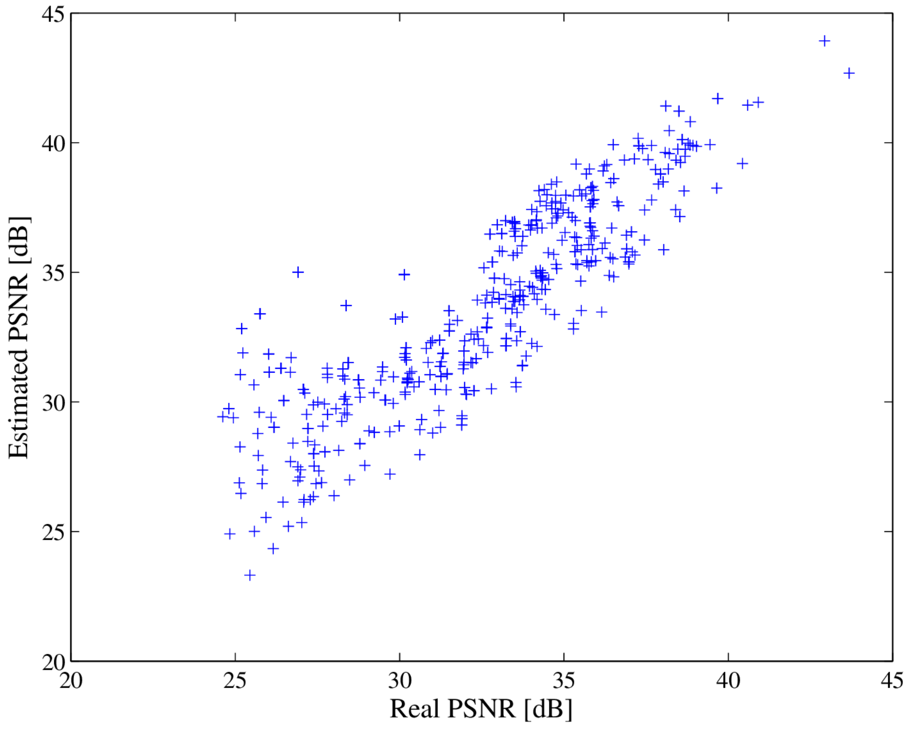

Figure 5.2: Scatter plot diagram: Estimated versus real PSNR values for 1080p video sequences.

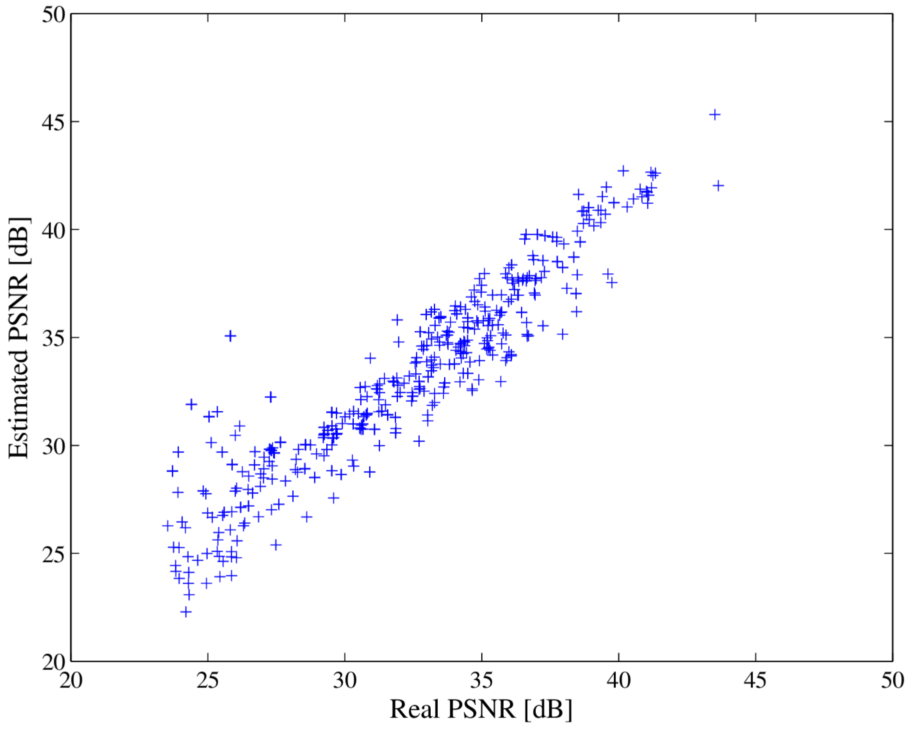

Figure 5.3: Scatter plot diagram: Estimated versus real PSNR values for 720p video sequences.

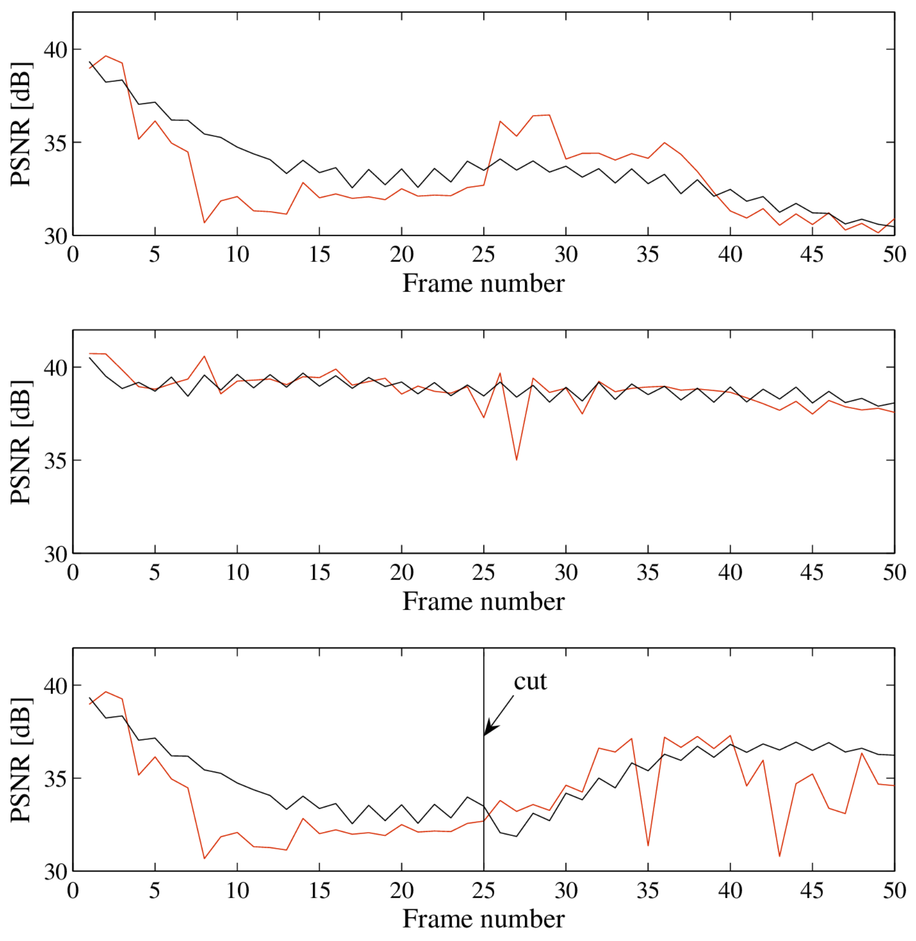

Figure 5.4: Real (black) and estimated (red) PSNR in a sequence with a cut.

same. In order to keep an approximately constant PSNR through the whole 50-frame sequences, the encoder is set to have an initial QP 25 and the target bitrate is 250 kbps.

Fig. 5.4 shows PSNR of three encoded video sequences developing over time. The top part of the plot displays 50 frames of the “paris” sequence, the center plot displays 50 frames of the “hall” sequence and the bottom plot displays a sequence with a cut, with the first 25 frames taken from the “paris” sequence and the following 25 frames taken from the “hall” sequence. For the “paris” sequence, the real PSNR begins at about 40 dB and decreases to some 34 dB at frame 25. At this point, the estimation is quite close and the estimation error is about 1 dB (even though larger errors are present in frames 5-15). A more precise PSNR estimation can be observed for the “hall” sequence, where the real PSNR stays about 40 dB with no big fluctuations. Now let’s have a look at the sequence with a cut (bottom plot in Fig. 5.4). The first half of the plot is identical to the top plot, belonging to the “paris sequence”. At the point of the cut, the real PSNR decreases a bit to consequently raise again. The errors in PSNR estimation, especially for particular frames in the sequence, have grown rapidly. Anyway, the estimation error at the end of the sequence is about 2 dB

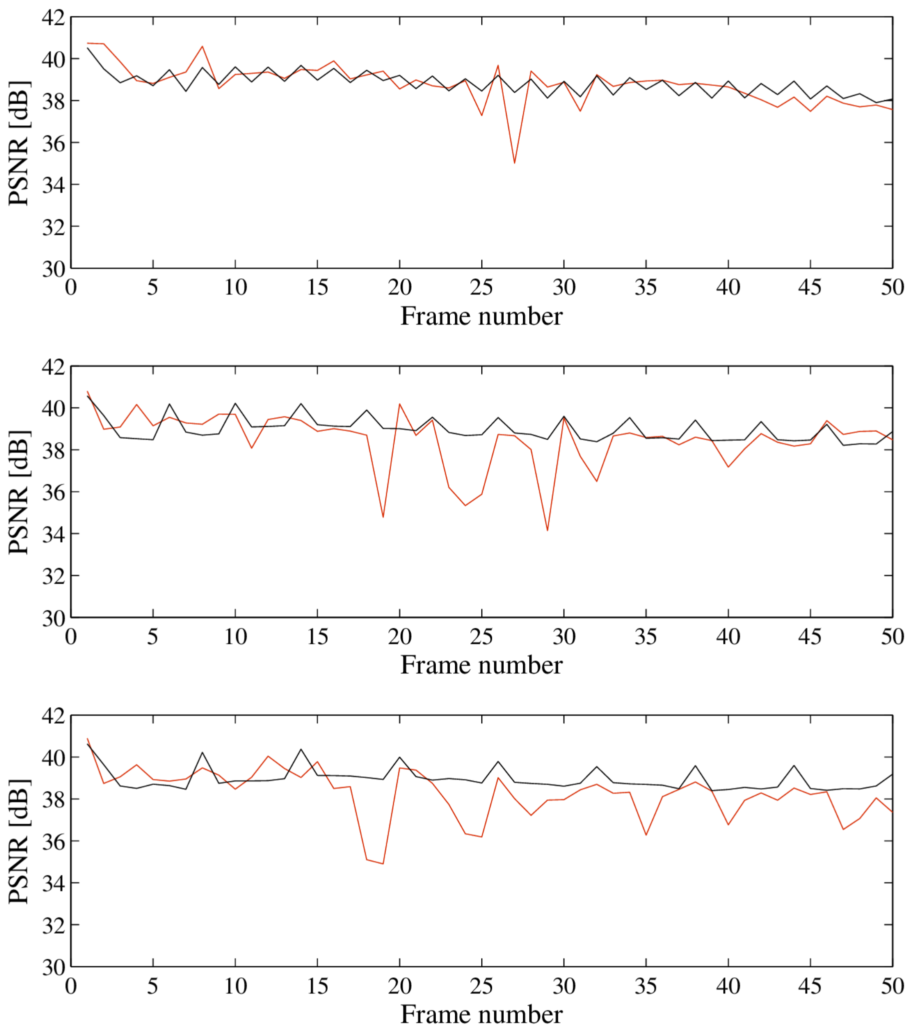

Figure 5.5: Real (black) and estimated (red) PSNR in sequences with different GOP formats.

5.3 Impact of GOP format

All High profile video sequence encoding in Chapter 3 were considered in only one GOP format, with one Intra (I) frame at the beginning of the sequence and P and B frames taking turns in the rest – the configuration was IBPBPBP. . . . In the following, different GOP formats, increasing the number of B frames inserted between I and P frames will be tested.

Fig. 5.5 shows real (black) and estimated (red) PSNR for three different GOP configurations in the first 50 frames of the sequence "hall". The plot in the top represents a sequence coded with one B slice inserted between P (I) frames (IBPBPBP. . .), the plot in the middle is for a sequence with three B slices inserted (IBBBPBBBP. . .), and the plot in the bottom is for a sequence with five B slices inserted (IBBBBBPBBBBBP. . .).

In these 50-frame sequences, the mean squared errors of the real and estimated PSNR values are 0.488, 1.683 and 1.633 for sequences with 1, 3 and 5 B frames inserted, respectively. The PSNR estimating algorithm had troubles in frames 20 – 30 for all the sequences, where the estimated PSNR for B frames, especially, is much lower than the real PSNR. In the sequences with increased number of B frames inserted, there are more consequent frames for which the estimated PSNR is too low – an error is PSNR estimation is thus propagated until a new P frame is inserted. Generally, there is quite few information for the B frames in the bitstream, which results in less accurate PSNR estimation.

5.4 Conclusion

The proposed PSNR estimation algorithm has been tested for different properties of the input video sequences in terms of resolution and content (cut) and for different encoder configurations in terms of GOP structure. The performance in the respective cases has been discussed.

The tests for varying video sequence resolution were published in [29].

Conclusions 6

At the beginning of the doctoral thesis, an introduction to the problem of video quality assessment was done and the present-day methods available for video quality assessment and evaluation were presented. Digital video quality assessment itself is a very broad area and a big deal of research has been done in it in the worldwide scientific community. However numerous techniques for objective video quality assessment exist, there is still much space further where the limits can be pushed.

The main contribution of the doctoral thesis is in designing a new no-reference metric for evaluating quality of video sequences compressed in the H.264/AVC standard. The idea is to extract the parameters, which may carry information about the quality of the encoded video. The designed metric reads the quantization parameter and the prediction modes used within a video frame. These parameters are then fed into an artificial neural network. The network is first trained on a set of examples – training sequences. Its performance is then verified using a different set of compressed video sequences – evaluation sequences.

For the network training, peak signal-to-noise ratios of the respective frames in the compressed video sequences are used as the network target output values. Even though the PSNR is no perfect quality measure, the benefit of our algorithm is in removing the necessity of having an original (uncompressed) video material available for PSNR calculations. We are estimating the PSNR solely from the encoded bit stream, which is easily applicable in any situation H.264/AVC encoded video is received. For example, the algorithm may be applied for detecting weak spots of a statistically multiplexed broadcast channel or to verify quality of a compressed video at the video content provider side before delivering it to the customer.

Experiments were performed with several artificial neural network configurations. An important point is that Intra coded frames (constrained in choice of prediction modes) are treated different from Inter coded frames (having all prediction modes available). However, in the best performing configuration among the tested network structures, the networks have the well-known and very common topology of a multi-layer perceptron. For Intra coded frames, the correlation of the results with the real peak signal-to-noise ratios reached as high as 0.9835. The situation is something worse for the Inter coded frames. In this case, the performance strongly depends on the choice of H.264/AVC profile when encoding the video sequences, as different profiles have different prediction modes available, not to mention that there are other tools enabled only in selected profiles, which may influence the PSNR. However, the reached correlation never dropped below 0.85 for the evaluation set sequence Inter frames. It is important to note that the algorithm was trained and verified only for one specific encoder. It shall be expected that for video encoded using different encoder implementations, different artificial neural networks would have to be trained. Algorithms to estimate PSNR of only the luma component of the encoded video sequences have been designed. The approach can be easily extended for chroma components as well, as chroma samples are predicted similarly to luma samples.

In Chapter 5, the behavior of the algorithm was analyzed when alternating some parameters of the encoded video or the encoder, always changing only one parameter. It has been verified that the algorithm is working with high-definition video and video sequences with a cut. It has difficulties in estimating PSNR for sequences where more subsequent B pictures are inserted.

The designed algorithm can easily be implemented in different video applications, where H.264/AVC compressed (but error-free) video is received. One such example might be content verification, when a video content provider needs an information whether or not the compressed video has satisfactory quality to be delivered to the customer.

The contribution of the doctoral thesis can be summarized in the following points:

- design of a new no-reference quality metric for H.264/AVC compressed video sequences,
- optimization of the metric's classification algorithm for a selected set of video sequences,
- performance tests of the metric, verifying its universality for different video material and encoder configurations.

Finally, constraints of the proposed solution should be considered and direction of consequent research suggested:

- The proposed algorithm has only been tested for progressive video sequences. It should be verified whether it works for interlaced video or a modified approach should be used.
- All the tests have been done with one particular encoder. It can be expected that for a different encoder implementation, the classification algorithm will have to be re-trained, as the encoder behavior is likely to be different.
- Interested results may be obtained when changing the estimation target from PSNR to a different metric (better correlating with subjective scores) or to subjective scores.

References

[1] Arizona State University. Video Traces Research Group: *CIF Sequences*. [online]. URL <http://trace.eas.asu.edu/yuv/cif.html>

[2] BISHOP, C. M.: *Pattern Recognition and Machine Learning*. New York: Springer, 2006, ISBN 0-387-31073-8.

[3] DALY, S. J.: The visible difference predictor: an algorithm for the assessment of image fidelity. In *Proc. SPIE: Human Vision, Visual Processing, and Digital Display III*, volume 1666, 1992, pp. 2–15.

[4] DEMUTH, H.; BEALE, M.: *Neural Network Toolbox for Use with MATLAB. User's Guide*. Natick: Mathworks, 2000, version 4.

[5] EDEN, A.: No-Reference Estimation of the Coding PSNR for H.264-Coded Sequences. *IEEE Transactions on Consumer Electronics*, volume 53, no. 2, May 2007: pp. 667–674.

[6] FISCHER, W.: *Digital Television: A Practical Guide for Engineers*. Berlin: Springer, 2004, ISBN 3-540-01155-2.

[7] GASTALDO, P.; et al.: Objective assessment of MPEG-2 video quality. *Journal of Electronic Imaging*, volume 11, no. 3, July 2002: pp. 365–374.

[8] HAGLUND, L.: *The SVT high definition multi format test set.* [online], 2005. URL <http://www.ebu.ch/en/technical/hdtv/test_sequences.php>

[9] HORITA, Y.; ARATA, S.; MURAI, T.: No-Reference Image Quality Assessment for JPEG/JPEG2000 Coding. In *Proceedings of XII. European Signal Processing Conference EUSIPCO-2004*, volume 2, 2004, pp. 1301–1304, ISBN 3-200-00148-8.

[10] ITU-R Recommendation BT.1439-1: *Measurement methods applicable in the analogue television studio and the overall analogue television system*. Geneva: The International Telecommunication Union, 2006.

[11] ITU-R Recommendation BT.500-11: *Methodology for the subjective assessment of the quality of television pictures*. Geneva: The International Telecommunication Union, 2002.

[12] ITU-T Recommendation H.264: *Advanced video coding for generic audiovisual services*. Geneva: The International Telecommunication Union, 2005.

[13] ITU-T Recommendation P.910: *Subjective video quality assessment methods for multimedia applications*. Geneva: The International Telecommunication Union, 1999.

[14] MARZILIANO, P.; et al.: A No-Reference Perceptual Blur Metric. In *Proceedings of the International Conference on Image Processing*, volume 3, September 2002, pp. 57–60.

[15] MICHIE, D.; SPIEGELHALTER, D. J.; TAYLOR, C. C.: *Machine Learning, Neural and Statistical Classification*. Prentice Hall, 1994, ISBN 0-1310-6360-X.

[16] MITCHELL, T. M.: *Machine Learning*. McGraw-Hill, 1997, ISBN 0-07-042807-7.

[17] ONG, E. P.; et al.: A No-Reference Quality Metric for Measuring Image Blur. In *Proceedings of the Seventh International Symposium on Signal Processing and Applications*, volume 1, July 2003, pp. 469–472, ISBN 0-7803-7946-2.

[18] PAN, F.; et al.: A Locally-Adaptive Algorithm for Measuring Blocking Artifacts in Images and Videos. In *Proceedings of the 2004 International Symposium on Circuits and Systems ISCAC '04*, volume 3, 2004, pp. 925–928, ISBN 0-7803-8251-X.

[19] PINSON, M. H.; WOLF, S.: Comparing subjective video quality testing methodologies. *Visual Communications and Image Processing*, volume 5150, 2003: pp. 573–582.

[20] RICHARDSON, I. E. G.: *H.264 and MPEG-4 Video Compression*. Chichester (England): Wiley, 2003, ISBN 0-470-84837-5.

[21] SAFRANEK, R. J.; JOHNSTON, J. D.: A perceptually tuned sub-band image coder with image dependent quantization and post-quantization data compression. In *Proceedings of IEEE International Conference on Acoustics Speech, and Signal Processing*, May 1989, pp. 1945–1948.

[22] SHEIKH, H. R.; BOVIK, A. C.: Image Information and Visual Quality. *IEEE Transactions on Image Processing*, volume 15, no. 2, February 2006: pp. 430–444, ISSN 1057-7149.

[23] SHEIKH, H. R.; et al.: No-Reference Quality Assessment Using Natural Scene Statistics: JPEG2000. *IEEE Transactions on Image Processing*, volume 14, no. 11, november 2005.

[24] SLANINA, M.: Evaluation of Full-Reference Image Quality Assessment Methods. In *Proceedings of the* 12^th *Conference Student EEICT 2006*, volume 4, Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií a Fakulta informačních technologií, 2006, pp. 341–345, ISBN 80-214-3163-6.

[25] SLANINA, M.; PROKOPEC, J.; JANÍČEK, M.: Application for Objective Image Quality Metric Performance Comparison and Demonstration. In *Proceedings of the Sixteenth International Electrotechnical and Computer Science Conference ERK 2007, Volume B*, Portoroz: Slovenia Section IEEE, 2007, pp. 203–206, ISSN 1581-4572.

[26] SLANINA, M.; ŘÍČNÝ, V.: A Comparison of Full-Reference Image Quality Assessment Methods. In *Radioelektronika 2006 Conference Proceedings*, Bratislava: Slovak Technical University Bratislava, 2006, pp. 165–168, ISBN 80-227-2388-6.

[27] SLANINA, M.; ŘÍČNÝ, V.: Hodnocení kvality videosekvencí na základě detekce blokových artefaktů. In *Sborník příspěvků konference VRŠOV*, Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2006, pp. 165–168, ISBN 80-214-3247-0.

[28] SLANINA, M.; ŘÍČNÝ, V.: Estimating H.264/AVC Video PSNR Without Reference Using the Artificial Neural Network Approach. In *Sigmap 2008 Conference Proceedings*, Porto: INSTICC, 2008, pp. 1–7.

[29] SLANINA, M.; ŘÍČNÝ, V.: Estimating PSNR in High Definition H.264/AVC Video Sequences Using Artificial Neural Networks. To appear in *Radioengineering*, 2008.

[30] SLANINA, M.; ŘÍČNÝ, V.: Estimating PSNR Without Reference for Real H.264/AVC Sequence Intra Frames. In *Proceedings of the 18th International Conference Radioelektronika 2008*, Prague: Czechoslovakia Section IEEE, 2008, pp. 55–58, ISBN 978-1-4244-2087-2.

[31] SLANINA, M.; ŘÍČNÝ, V.; FORCHHEIMER, R.: MPEG-4 AVC No-Reference PSNR Estimator: The Framework. In *Proceedings of 17th International Conference Radioelektronika 2007*, Brno: Brno University of Technology, 2007, pp. 133–136, ISBN 1-4244-0821-0.

[32] SLANINA, M.; ŘÍČNÝ, V.; FORCHHEIMER, R.: A Novel Metric for H.264/AVC No-Reference Quality Assessment. In *Proceeding of 2007 14th International Workshop on Systems Signals & Image Processing (IWSSIP) & 6th EURASIP Conference Focused on Speech & Image Processing, Multimedia Communications & Services (EC-SIPMCS)*, Maribor: University in Maribor, 2007, pp. 119–122, ISBN 978-961-248-029-5.

[33] SÜHRING, K.: The H.264/MPEG-4 AVC Reference Software – JM11. [online]. URL `<http://iphome.hhi.de/suehring/tml/download/>`

[34] TEO, P. C.; HEEGER, D. J.: Perceptual image distortion. In *Proceedings SPIE*, volume 2179, 1994, pp. 127–141.

[35] TONG, H.; et al.: Learning No-Reference Quality Metric by Examples. In *Proceedings of the 11th International Multimedia Modelling Conference MMM'05*, Melbourne: IEEE Computer Society, 2005, pp. 247–254, ISSN 1550-5502.

[36] TOURAPIS, A. M.; LEONTARIS, A.: H.264/MPEG-4 AVC Reference Software Manual. [online].
URL `<http://iphome.hhi.de/suehring/tml/JM%20Reference%20Software%20Manual%20(JVT-X072).pdf>`

[37] Video Quality Experts Group: *Final Report from the Video Quality Experts Group on the Validation of Objective Models of Video Quality* Assessment. [online], 2000.
URL `<http://www.its.bldrdoc.gov/vqeg/projects/frtv_phaseI/COM-80E_final_report.pdf>`

[38] Video Quality Experts Group: *Final Report from the Video Quality Experts Group on the Validation of Objective Models of Video Quality Assessment, Phase II.* [online], 2003.
URL <http://www.its.bldrdoc.gov/vqeg/projects/frtv_phaseII/downloads/VQEGII_Final_Report.pdf>

[39] Video Quality Experts Group: *RRNR-TV Group Test Plan.* [online], 2007.
URL <ftp://vqeg.its.bldrdoc.gov/Documents/Projects/rrnr-tv/RRNR-tv_draft_2.1_changes_highlighted.doc>

[40] WANG, Z.; BOVIK, A. C.; EVANS, B. L.: Blind Measurement of Blocking Artifacts in Images. In *Proceedings of 2000 International Conference on Image Processing*, volume 3, Vancouver, 2000, pp. 981–984.

[41] WANG, Z.; LU, L.; BOVIK, A. C.: Video Quality Assessment Based on Structural Distortion Measurement. *Signal Processing: Image Communication*, volume 19, no. 2, February 2004: pp. 121–132.

[42] WATSON, A. B.: Toward a Perceptual Video-Quality Metric. *Proceedings SPIE, Human Vision and Electronic Imaging*, volume 3299, 1998: pp. 139–147.

[43] WATSON, A. B.; HU, J.; McGOWAN, J. F.: DVQ: A Digital Video Quality Metric Based on Human Vision. *Journal of Electronic Imaging*, volume 10, no. 1, 2001: pp. 20–29.

[44] WIEGARD, T.; et al.: Overview of the H.264/AVC Video Coding Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, volume 13, no. 7, July 2003: pp. 469–472, ISSN 1051-8215.

[45] Wikipedia: *MPEG-4*. [online], 2007.
URL <http://en.wikipedia.org/wiki/MPEG-4>

[46] WINKLER, S.: A Perceptual Distortion Metric for Digital Color Images. In *Proceedings of 1998 International Conference on Image Processing*, volume 3, 1998, pp. 399–403, ISBN 0-8166-8821-1.

[47] WINKLER, S.: *Digital Video Quality: Vision Models and Metrics.* Chichester (England): Wiley, 2005, ISBN 0-470-02404-6.

[48] WOLF, S.; PINSON, M.: *Video Quality Measurement Techniques.* NTIA Report 02-392, U.S. Department of Commerce, 2002.

[49] WU, H. R.; RAO, K. R.: *Digital Video Image Quality and Perceptual Coding.* Boca Raton: Taylor & Francis, 2006, ISBN 0-8247-2777-0.

Used Symbols and Abbreviations

ANN	Artificial neural network
AVC	Advanced video coding
CIF	Common intermediate format
CSF	Contrast sensitivity function
DCT	Discrete cosine transform
DVB-T	Digital video broadcasting: Terrestrial
FR	Full reference
GOP	Group of pictures
HDTV	High definition television
HVS	Human visual system
ITU	International telecommunication union
JVT	Joint video team
LMS	least mean squares
MPEG	Motion picture experts group
MSE	Mean squared error
NAL	Network abstraction layer
NR	No reference
PSNR	Peak signal-to-noise ratio
QP	Quantization parameter
RR	Reduced reference
SSIM	Structural similarity index
VCL	Video coding layer
VQEG	Video quality experts group

List of Figures

List of Tables