MODEL-BASED TRANSRATING OF CODED VIDEO

NAAMA HAIT

MODEL-BASED TRANSRATING OF CODED VIDEO

RESEARCH THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN ELECTRICAL ENGINEERING

NAAMA HAIT

SUBMITTED TO THE SENATE OF THE TECHNION — ISRAEL INSTITUTE OF TECHNOLOGY
KISLEV, 5768 HAIFA NOVEMBER, 2007

THE RESEARCH THESIS WAS DONE UNDER THE SUPERVISION OF PROF. DAVID MALAH IN THE DEPARTMENT OF ELECTRICAL ENGINEERING

ACKNOWLEDGMENT

I would like to express my deep gratitude to Prof. David Malah, for his devoted supervision and professional guidance throughout this research work. I also wish to thank the staff of the Signal and Image Processing Lab (SIPL) for their help and technical support. Finally, special thanks to my family and to Boaz for their support and encouragement.

This research was supported in part by STRIMM consortium under the MAGNET program of the Ministry of Trade and Industry via the Samuel Neaman Institute and by the Signal and Image Processing Lab.

THE GENEROUS FINANCIAL HELP OF THE TECHNION IS GRATEFULLY ACKNOWLEDGED

Contents

List of Tables

List of Figures

Abstract

Video services and multimedia applications use pre-encoded video in different formats for storage and transmission. Usually, servers store a single copy at a high quality, while various user types require different formats and bit rates. Therefore, the high quality pre-encoded video is converted on-line to match user-specific requirements. Bit rate reduction within the same video format is called *transrating*, and can be achieved by a number of methods. In this research work, we examine model-based transrating via requantization of the transform coefficients, in the state of the art H.264 coder.

Many previous works on requantization chose the optimal step sizes via Lagrangian optimization that minimizes the distortion subject to a rate constraint. However, these works did not use analytic models for the relation between the rate and the quantization step. Hence, they required an exhaustive search for the optimal steps, including repetitive quantization and coding.

The new H.264 standard offers advanced coding features, such as intra spatial prediction and context adaptive entropy coding, at the expense of higher complexity. But, these features pose additional algorithmic problems that do not allow the implementation of the methods developed in previous transrating works as is.

The goal of a transrating system is to reduce the bit rate of an encoded video sequence, at low complexity, while preserving a high quality video. Therefore, the

naive solution of re-encoding (cascaded decoder-encoder) is put aside due to its high computational complexity. To reduce the computational complexity, the proposed transrating system reuses as much input coding decisions as possible (e.g. motion vectors). The model-based requantization further reduces the computational burden by alleviating the repetitive quantization and coding required during the search for the optimal step sizes. The models incorporated in this work relate the rate and the distortion to the fraction of zeroed quantized transform coefficients, ρ, rather than to the step size itself.

The intra spatial prediction in H.264 introduces block dependencies that pose two algorithmic problems. First, to avoid a drift error, the intra-coded frame should be fully decoded. Second, estimating the relation between ρ and the step size becomes a challenging task, for which we propose a novel statistical-based model.

For optimal requantization in inter-coded frames, we propose two novel modifications of previous work. First, an extended Lagrangian optimization is proposed, to improve the subjective quality by regulating the changes in the quantization step sizes throughout the frame. Second, the ρ-domain rate-distortion models suggested in previous works are not suitable for macroblock level coding in H.264. The macroblock level rate-distortion models developed in this work are adapted to H.264 requantization and consider its context adaptive entropy coding.

Overall, as compared to re-encoding, the proposed transrating system reduces the computational complexity by a factor of about 4, at a maximal cost of 1.4[dB] in PSNR. In comparison with a simple one-pass requantization, the proposed algorithm achieves better performance both objectively (PSNR gain of up to 1.6[dB]) and subjectively, at the cost of twice the complexity.

List of Symbols

Q_1, Q_2	Input, output quantization step size
BR_{factor}	Average transrating factor
D	Distortion
R	Rate
B	Buffer status
$Th(Q_2)$	Deadzone interval of the second quantizer
I	Decoded picture
X,Y	Residual signal in the pixel domain/transform domain
ε	Transrating error
Z	Quantized transform coefficients
C	Closed-loop correction signal
W	Corrected residual transform coefficients
ρ	Fraction of zero coefficients in the data
$p(x)$,$F(x)$	Probability/Cumulative distribution function
N_B	Number of macroblocks in the frame
d_i	Distortion caused to the i-th macroblock
r_i	Number of bits produced by the coding i-th macroblock
j_i	Rate-distortion cost of the i-th macroblock
V	Value function
m	Prediction mode

List of Abbreviations

CAVLC	Context Adaptive Variable Length Coding
CBP	Coded Block Pattern
CPDT	Cascaded Pixel Domain Transcoder
FPDT	Fast Pixel Domain Transcoder
GOP	Group Of Pictures
HVS	Human Vision System
ICT	Integer Cosine Transform
MAD	Mean Absolute Difference
MB	Macroblock
MC	Motion Compensation
ME	Motion Estimation
MSE	Mean Square Error
MV	Motion Vector
PSNR	Peak Signal to Noise Ratio
SAD	Sum of Absolute Difference
VLC	Variable Length Coding
VLD	Variable Length Decoding
VQM	Video Quality Model

Chapter 1

Introduction

1.1 Transrating of coded video in H.264

Video services and multimedia applications use pre-encoded video in different formats for storage and transmission. As various user types require different formats and bit rates, a single copy of the encoded video cannot satisfy all users. One could store many copies of the video in the server, each encoded at a different format or bit rate, and send the bitstream with the closest requirements to those requested by the user. However, such server has very high storage costs and the chosen bitstream may not meet the exact user requirements. Therefore, servers store a single copy, pre-encoded at a high quality, and convert it on-line to match user-specific requirements. Bit rate reduction within the same video format is called *transrating*, and can be achieved by a number of methods, such as frame rate reduction, spatial resolution reduction and requantization of the transform coefficients. In this research work, we examine model-based transrating via requantization of the transform coefficients, in the state of the art H.264 coder.

The goal of a transrating system is to reduce the bit rate of an encoded video sequence, at low complexity, while preserving a high quality video. The naive solution is to perform re-encoding by cascading a decoder and an encoder. Since the encoder

re-estimates the motion vectors and prediction modes, the re-encoding has a high computational complexity and this solution is put aside. To save computations, the common approach is either to reuse the input coding decisions, or to use them as a prior to restrict the search for new ones, where the target bit rate is achieved via requantization.

Previous works, related to previous standards, chose the optimal step sizes via Lagrangian optimization that minimizes the distortion subject to a rate constraint. The optimal step-sizes search required evaluating the rate and the distortion obtained by requantizing each picture region at multiple step-sizes. As these previous works did not use analytic models for the relation between the rate and the quantization step, the rate assessment involved repetitive quantization and coding. As a result, the optimization procedure became an exhaustive search.

H.264 is currently the state of the art video coding standard. It offers advanced coding features, such as intra spatial prediction, variable block size motion compensation, integer transform, context adaptive entropy coding and an in-loop de-blocking filter. However, due to these features, the rate control becomes computationally expensive, as the choices of quantization step-size and coding modes are dependent. Therefore, previous works on transrating in H.264 focus on adapting the input coding decisions to the lower rate, and the requantization is addressed by a simple one-pass algorithm. The methods suggested in previous transrating works cannot be applied as is to H.264, and new algorithms should be developed.

1.2 Proposed transrating scheme

In this research work, new model-based optimal requantization algorithms were developed and examined, for transrating of H.264 coded video. The models incorporated in this work relate the rate and the distortion to the fraction of zeroed quantized transform coefficients, ρ, rather than to the step size itself.

Frame-level bit allocation is found by minimizing the overall distortion over a group of frames, such that the target average bit rate is achieved. To keep a smooth constant video quality, the frame distortions are equalized.

For intra-coded frames, the spatial prediction introduces dependencies between neighboring residual blocks. Due to these dependencies, the residual coefficients to be requantized are not available in advance, when the requantization step-size should be selected. Therefore, the estimation of the relation between ρ and the requantization step size becomes a challenging task. To this end, we propose a novel closed-loop statistical estimator, that outperforms the simple open-loop estimator.

For inter-coded frames, we propose optimal requantization that improves the subjective quality by regulating the changes in the requantization step sizes throughout the frame. To this end, we suggest extending the Lagrangian optimization by a dynamic programming algorithm. To reduce the computational burden of the optimization, we use rate-distortion models at the macroblock level. As the models suggested in the literature are not suitable for macroblock level coding in H.264, we developed macroblock level rate-distortion adapted to H.264 requantization. Since the recommended encoder eliminates very sparse blocks, we also examine extending the optimal requantization by selective coefficient elimination.

1.3 Thesis outline

This thesis is organized as follows:

Chapter 2 overviews previous work, starting from transrating architectures to bit-rate reduction algorithms, both at the frame level and at the macroblock level. A brief overview on rate-distortion modeling is given, including both models that relate the rate and the distortion to the quantization step size and to the fraction of zero coefficients in the data. Some design guidelines that consider the perceived quality of the transrated video are also given.

As this work focuses on transrating an H.264 bit stream, Chapter 3 briefly overviews the main coding features of the standard. The standard details not given in this chapter and relevant to this work are given in Appendix A.

Chapter 4 defines the proposed transrating scheme. It discusses the transrating architectures chosen for the intra-coded and the inter-coded frames. It defines the GOP-level bit allocation that solves a model-based optimization problem. In addition, it defines a simple requantization algorithm, to which we will compare our proposed algorithm performance.

Chapter 5 presents the suggested model-based uniform requantization for transrating of intra-coded frames. It suggests a novel model for estimating the relation between the fraction of zeros in the data and the requantization step size, while considering the blocks dependency problem introduced by spatial prediction.

Selective modification of the intra prediction modes is suggested in Chapter 6. The algorithm incorporates input prior with human vision considerations to allow selective modification only where the coding gain is expected to increase.

Chapter 7 proposes a new optimal requantization algorithm, adapted for transrating of H.264 inter-coded frames. It regulates the change in the quantization step sizes throughout the frame, in order to achieve a smooth perceived quality. In addition, it suggests to extend the optimal requantization by selective coefficients elimination.

To reduce the computational load of the algorithm suggested in Chapter 7, new rate-distortion models at the macroblock level are suggested in Chapter 8. The proposed models are adapted for macroblock level requantization in H.264.

Chapter 9 summarizes simulation results, tested using a few video sequences. The proposed system's performance is measured by its run-time and its quality at a target bit rate. The quality is assessed using both the PSNR measure and the VQM measure. The trade-off between quality and computational complexity is considered.

Chapter 10 concludes the thesis and suggests future research directions.

Chapter 2

Previous Work

The goal of a transrating system is to reduce the bit rate of an encoded video sequence, at low complexity, while preserving a high quality video. In this work, we focus on transrating via requantization of the transform coefficients, in the H.264 coder. There are three major design issues to consider: bit rate reduction, computational complexity and quality.

Section 2.1 describes transrating architectures that provide different compromises between quality and computational complexity. Incorporation of rate-distortion models further reduces the computational load, and is described in section 2.2. Bit-rate control algorithms via requantization of the transform coefficients are discussed in section 2.3. Finally, section 2.4 briefly reviews design guidance for preserving video quality.

2.1 Transrating architectures

The naive and straightforward transrating architecture is *re-encoding* [43, 4]. In this architecture, a decoder and encoder are cascaded, see Fig. 2.1, and therefore some refer to it as the cascaded transcoder. The input bit stream is fully decoded to obtain the reconstructed sequence and then re-encoded at the target output bit

rate using new coding decisions. This architecture has the highest computational complexity among transrating architectures, as it finds new coding decisions, which involve performing motion estimation (ME). We will now outline various architectures to reduce the transrater's computational complexity.

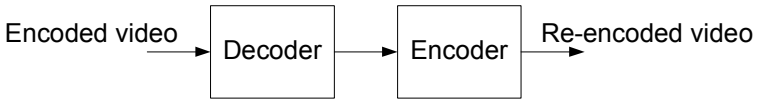

Figure 2.1: Re-encoding architecture.

The architecture with the lowest computational complexity for requantization is the open-loop transrater [23, 43, 49, 4], see Fig. 2.2. The residual's transform coefficients are dequantized and then requantized at a coarser step-size to meet the target bit rate. Following this scheme, expensive operation such as motion estimation (ME) and transforms are avoided and there is no need for a frame-store. However, open loop transraters are subject to *drift error* that degrades the video's quality [49, 4].

In predictive coding, the video frame is predicted, either temporally or spatially, from previously decoded frames or frame parts, and only the residual error is coded. Therefore, the encoder and the decoder must be synchronized, in the sense of using the same reference signal for prediction. In case of a mismatch, the decoder reconstructs the encoded frames with errors, that further accumulate as these erroneous frames are used as references for more spatial and temporal predictors. This error accumulation is called drift error.

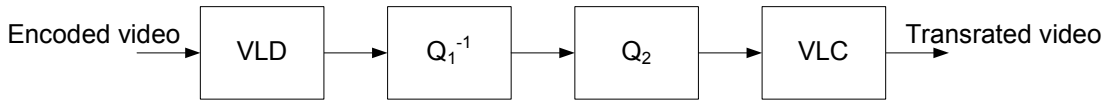

Figure 2.2: Open loop architecture. The encoded video undergoes: variable length decoding (VLD), dequantization, requantization and variable length coding (VLC).

In between these two extremes, there are architectures that reduce the computational complexity as compared to re-encoding, without introducing a drift error

[44, 43, 4, 48]. The Cascaded Pixel Domain Transcoder architecture (CPDT), also known as spatial-domain transcoding architecture, is depicted in Fig. 2.3. The input bit stream is fully decoded and then encoded by reusing the input coding decisions (e.g. the motion vectors (MVs)) to reduce the encoder's complexity. This transcoder does not suffer from drift error as the decoder-loop and the encoder-loop are independent and the motion-compensated residual is recomputed at the encoder.

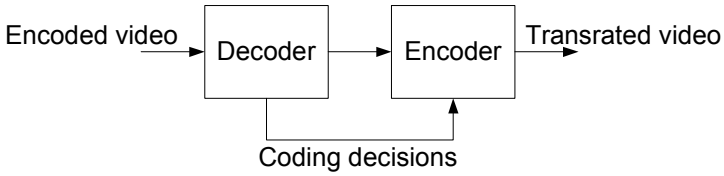

Figure 2.3: Cascaded Pixel Domain Transcoder architecture.

Since the input coding decisions are reused, the architecture can be simplified further [49, 43, 4, 48]. The Fast Pixel Domain Transcoder architecture (FPDT) depicted in Fig. 2.4 performs partial decoding followed by partial encoding. The partial decoding reconstructs just the residual signal in the pixel domain, rather than reconstructing the fully decoded picture. It reuses the input coding decisions and performs a closed-loop correction to compensate for the drift error. Based on the assumption that the predictors are linear, it predicts a single correction signal rather than predicting two fully reconstructed signals, both in the decoder and the encoder.

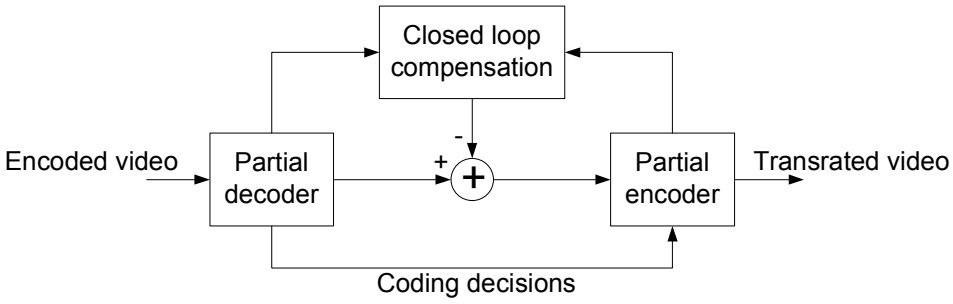

Figure 2.4: Fast Pixel Domain Transcoder architecture.

If the assumption regarding the predictor's linearity is correct, one can exploit the transform's linearity and convert the FPDT to work entirely in the transform domain

[23, 43, 49, 4]. This requires implementing a transform-domain representation of the predictor in the correction loop.

The conclusion from the transrating architectures presented here is that a reasonable tradeoff between quality and computational complexity can be achieved by using the CPDT or the FPDT architectures. It should be noted that the independent decoder-loop and encoder-loop in CPDT allow to modify the input coding decisions, whereas the FPDT architecture requires reusing them all. Now, the question is whether or not to reuse all the input decisions, and in particular the input MVs. On one hand, reusing the input MVs significantly reduces the computational complexity [59] and provides a description of the motion at the high bit rate working point, where the original sequence was used as a reference. On the other hand, [60, 32, 22] argue that the variable block size motion compensation introduced in H.264 (see Chapter 3) changes this assumption as the input MVs that were selected for the high bit rate, are sub-optimal at the lower bit rate, and consume too many overhead bits at the expense of the number of bits for coding the residual.

2.2 Rate-distortion modeling

An encoder, or a transrater, uses a rate control algorithm (see section 2.3) to choose an appropriate quantization step size for an encoding unit (e.g. a frame, a video object, or a macroblock). To this end, it should evaluate the rate and the distortion introduced as a result of quantizing the transform coefficients in the following manner. First, perform the actual quantization, then apply the entropy coding algorithm to calculate the rate. In addition, a comparison between the quantized and the input transform coefficients is required to calculate the distortion. When this process is repeated for multiple combinations of encoding units and step sizes, the system's computational complexity is high. Incorporation of models allows fast evaluation of the rate and the distortion.

It should be noted that using rate-distortion models for transrating algorithms is different than for the first encoding algorithms (encoding the original input video). On one hand, the transrating system has only access to the quantized transform coefficients, not the original ones. Therefore, the transrating distortion can only refer to the requantization distortion, not the total degradation in quality from the original video. Also, estimation of the transform coefficients' statistical distribution is more challenging, if required, as it should be estimated from their quantized version. Nevertheless, the transrating algorithm can use the already encoded information (such as number of bits spent on each block) in order to estimate the rate model parameters. This is not the case for the encoding algorithm, that should either try encoding at a certain quantization step to calculate a rate point, or update the model parameters based on the previously encoded frames, which is not robust to scene cuts.

Over the years, a number of models were reported in the literature. Most of which relate the rate and the distortion to the quantization step size, as described in section 2.2.1. Recently, new models that relate the rate and the distortion to the fraction of quantized coefficients in the data were proposed, as described in section 2.2.2.

2.2.1 Modeling in the quantization step size domain

There are two main approaches to rate-distortion modeling: distribution-based modeling and empirical-based modeling [8]. The distribution-based modeling approach assumes that the transform coefficients statistics was drawn from a probability distribution function. Given the probability distribution, analytical models are derived using either the entropy of the quantized transform coefficients or a closed-form rate-distortion function. The empirical-based modeling gathers statistics from observed operational measured rate-distortion curves. Models are then fitted to best describe the observed trends.

Distribution-based modeling - Laplacian distribution

A Laplacian distribution is a popular model for the AC transform coefficients distribution [9, 50, 55, 38, 33, 57, 51, 15]:

$$p(x) = \frac{\alpha}{2} e^{-\alpha|x|} \tag{2.1}$$

where a lower value of the parameter α corresponds to a wider distribution. In [9, 57], the distortion is measured as $d(x, \hat{x}) = |x - \hat{x}|$ where x and $\hat{x}$ are the original and decoded samples, respectively. The closed-form solution of the rate-distortion function is used: $R(D) = -ln(\alpha D)$ for $0 < D \leq \frac{1}{\alpha}$, where $D = E\{d\}$. First, the analytic $R(D)$ function is approximated by a Taylor series. Then, the distortion is approximated as the quantization step-size itself:

$$D(Q) = Q \tag{2.2}$$

to derive the quadratic $rate - Q$ model:

$$R(Q) = \frac{a}{Q} + \frac{b}{Q^2} \tag{2.3}$$

where a, b are the model parameters. Variations of this model are incorporated in the recommended rate control algorithms of both the MPEG-4 standard [50, 43] and the H.264 standard [57, 40].

In [38, 51, 55, 33], different $rate - Q$ models were derived for different operating points. At first, the empirical entropy of the quantized transform coefficients was calculated. Then, different models were fit to describe the entropy at different bit-rates. In [51, 38], a high bit-rate model is used:

$$R(Q) = \frac{1}{2} log_2(2e^2 \frac{\sigma^2}{Q^2}) \quad , \quad \frac{\sigma^2}{Q^2} > \frac{1}{2e} \tag{2.4}$$

where $\sigma^2 = \frac{2}{\alpha^2}$. The low bit-rate approximation: [51, 33, 38]

$$R(Q) = \frac{e}{ln(2)} \frac{\sigma^2}{Q^2} \quad , \quad \frac{\sigma^2}{Q^2} < \frac{1}{2e} \tag{2.5}$$

takes either a quadratic form of

$$R(Q) = a\frac{\sigma^2}{Q^2} + b\frac{\sigma}{Q} \tag{2.6}$$

in [51], or the form of

$$R(Q) = a\frac{\sigma^2}{Q^2} \tag{2.7}$$

in [38, 33]. In [55], at intermediate bit-rates, a linear model is suggested:

$$R(Q) = b\frac{\sigma}{Q} \tag{2.8}$$

The $distorton - Q$ model used in [38, 33] is a high-resolution approximation (for high bit-rates):

$$D(Q) = \frac{Q^2}{12} \tag{2.9}$$

In [55], the mean squared error (MSE) was calculated according to the Laplacian distribution, to derive another $distorton - Q$ model:

$$D(Q) = a\sigma Q^c + b \tag{2.10}$$

Distribution-based modeling - Cauchy distribution

In [16], a Cauchy probability distribution function is suggested to describe the transform coefficients distribution:

$$p(x) = \frac{1}{\pi}\frac{\mu}{x^2 + \mu^2} \tag{2.11}$$

where a larger value of the parameter μ corresponds to a wider distribution. Since the Cauchy distribution does not have a closed-form analytic rate-distortion function, the entropy of the quantized transform coefficients and the corresponding MSE were calculated. Then, approximated models were derived for the $rate - Q$ relation:

$$R(Q) = aQ^{-\alpha} \tag{2.12}$$

and for the $distortion - Q$ relation:

$$D(Q) = bQ^{\beta} \tag{2.13}$$

where $a, \alpha, b, \beta > 0$ are parameters that depend on μ.

Empirical-based modeling

$Rate - Q$ models were also derived by examining operational measured curves. In [26], the authors suggest the model

$$R(Q) = \frac{K \cdot SAD}{Q} \tag{2.14}$$

where SAD is the sum of absolute differences of the residual. In [41], a linear relation between $log(R)$ and $log(Q)$ is suggested:

$$R(Q) = \delta \frac{1}{Q^{\gamma}} \tag{2.15}$$

and in [42], a quadratic relation between $log(R)$ and $log(Q)$ is suggested for intra-coded frames of H.264.

In light of the various suggested models for the $rate - Q$ relation, there is obviously no unanimous agreement regarding which model is the most suitable.

2.2.2 Modeling in the ρ domain

In [14, 13], the ρ-domain source model is suggested, where ρ is the fraction of zero coefficients among the quantized transformed coefficients in a frame. The model states that there is a strong linear relation between ρ and the actual frame's bit rate: coarser quantization step-sizes generate more zero coefficients (and hence increase ρ) while decreasing the rate. Therefore, the suggested $rate - \rho$ relation is:

$$R(\rho) = \theta \cdot (1 - \rho) \tag{2.16}$$

where the parameter θ is the graph's slope, as depicted in Fig. 2.5. According to this equation, for $\rho = 1$ all the quantized coefficients are zeroed and thus the coding rate should approach zero. The parameter θ [14] is related to the amount of texture in the encoded data. The texture is represented in the transform domain by medium to high frequencies, whereas for smooth data most of the energy is concentrated at

the low frequencies. When the encoded data has more texture, the coding bit rate increases, and so thus θ.

It is also argued that the $rate - \rho$ model is more robust than a $rate - Q$ model: the observed $rate - \rho$ curves for both I and P frames share a very similar pattern, whereas the $rate - Q$ curves change between different frame types.

The distortion too is more conveniently described in the ρ domain than in the quantization step-size domain as it is defined within the finite range of $0 \leq \rho \leq 1$ and follow a more robust and regular behavior. In [15], an exponential model for the MSE distortion in the ρ domain was suggested as

$$D(\rho) = \sigma^2 \cdot e^{-\alpha \cdot (1-\rho)} \tag{2.17}$$

where σ^2 is the variance and $\alpha > 0$ is a model parameter, as depicted in Fig. 2.5. Again, as $\rho \rightarrow 1$ and all the quantized coefficients are zeroed, the distortion approaches the σ^2 bound.

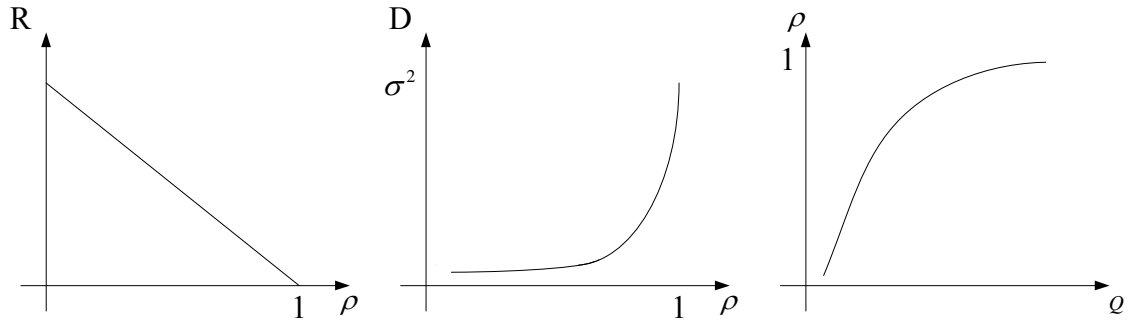

Figure 2.5: Rate-distortion models in ρ domain. Left: Linear $rate - \rho$, middle: $distortion - \rho$, right: $\rho - Q$ relation.

Even though the ρ domain models are empirical based, [14] numerically justifies the models for transform coefficients with a Generalized Gaussian distribution (that includes both the Gaussian distribution and the Laplacian distribution as special cases). It also justifies the models analytically for the case of a Laplacian distribution.

Finally, the $\rho - Q$ relation should be estimated. One approach is to assume the transform coefficients have a certain probability distribution, estimate the distribution

parameter(s) from the data and then analytically derive ρ. A computational simpler and more accurate approach uses a histogram count, where ρ is the fraction of the transform coefficients whose values fall below the quantizer's deadzone.

These models were derived for describing the rate and the distortion at the frame level, and were found quite accurate in [14, 13, 15], when tested for standards such as MPEG-2 and H.263. The ρ-domain models were also used for frame level encoding in the H.264 in [42, 29, 47, 12, 45], but in [45] it is argued that they do not describe well coding at the macroblock (MB) level. We address this issue in Chapter 8.

2.3 Rate control for requantization

There are numerous rate control algorithms in the literature. In this section, we briefly overview algorithms for quantization, with an emphasis on requantization. Rate control algorithms have two main stages. The first determines the bit allocation between the video frames, while the second allocates the bits between the frame's encoding units (e.g. a macroblock) to achieve the frame's target rate.

Frame-level bit allocation

Consider transrating at an average transrating factor $BRfactor > 1$. The simplest frame-level bit allocation [25, 41] reduces the bit rate of each frame by $BRfactor$: $R_{out} = \frac{R_{in}}{BRfactor}$, where R_{in} and R_{out} are the input and output frame's bit rates, respectively. This method does not require any assumptions regrading the Group Of Pictures (GOP) structure or buffer monitoring.

Since intra-coded frames are the reference for the rest of the GOP, [25] also suggests to allocate them more bits by: $R^I_{out} = \frac{R^I_{in}}{\sqrt{BRfactor}}$; reduce the bit rate of P-frames by the average factor: $R^P_{out} = \frac{R^P_{in}}{BRfactor}$ and equally adjust the bits left in the GOP for the B frames.

In [48], the authors suggest to solve an optimization problem, where the overall

GOP distortion is minimized subject to the average GOP bit rate target, which is reduced by $BRfactor$. That work uses the frame level $rate - \rho$ and $distortion - \rho$ models of (2.16) and (2.17), respectively. In [5], an additional constraint is added to the optimization problem, for equal frame distortions. The frame level ρ domain models are also used in [18, 30] to encode the original input video at a smooth quality. First, the frame's target distortion is set as the average distortion of the previously encoded frames, then the frame's target rate is extracted using the models. In [18], an operational $distortion - Q$ curve is used to avoid the estimation of the $distortion - \rho$ model parameters. In [56], another optimization problem is defined. A rate-distortion model based on the Generalized Gaussian distribution is altered to model the inter-frame dependency. However, only sub-optimal allocation is implemented to enable real time transcoding.

Macroblock-level bit allocation

The second type of rate control algorithms adjust the quantization step size at the macroblock level to achieve the frame's target bit rate. The one-pass algorithm processes one macroblock at a time, and sets its quantization step size according to the output buffer fullness. The update rule in [25] is simple, whereas in [48, 23], the frame level linear $rate - \rho$ model is used. In [23], the fraction of zeroed coefficients for every step size choice is evaluated using a histogram count. Based on the buffer fullness, the average available number of bits for the macroblock coding is calculated. Then, the requantization step that yields the closest target bits according to the linear model is chosen. The one-pass of [48] is based on the algorithm suggested in [13]. It sets an initial average step size for the frame according to the linear $rate - \rho$ model. The step size at each macroblock is adjusted relative to that average step, based on the buffer fullness. These one-pass algorithms are not optimal, and active frame regions towards the end of the frame's raster scan may suffer coarse quantization.

Optimal requantization for MPEG-2 encoded video is suggested in [6] by minimizing the frame's distortion subject to its target bit rate. In that work, Lagrangian optimization is applied as the step size choices at different macroblocks are independent. However, that optimization involves a high computational complexity since it evaluates the rates and the distortions at each combination of macroblock and requantization step exhaustively, with no models. In [24], it is suggested to restrict the search to check even and odd multiples of the initial quantization step-size to reduce the complexity of an open-loop Lagrangian optimization. In [19], the optimal requantization step sizes selection was extended by indices modification while minimizing the overall frame level cost.

In [18], the quantization step sizes are optimally selected for encoding the original input video, as suggested in [38], to minimize the frame's overall distortion subject to the target bit rate constraint, by using the $rate-Q$ model of (2.7) and the $distortion-Q$ model of (2.9).

Another optimization algorithm [50] finds the optimal requantization step sizes at a video object resolution (it refers to the MPEG-4 standard). It uses the $rate - Q$ model of (2.3) and evaluates the distortion by $D(Q) = Q^2$. To assure that the requantization steps are not finer than the input step for each video object, the authors suggest extending the Lagrangian optimization with a dynamic programming algorithm.

2.4 Quality

The transrating system should reduce the bit rate of an encoded video sequence while preserving a high quality video. Since the quality rating is judged by a human observer, the quality measure should be defined in view of our visual perception. Yet, there is no absolute solution for this matter.

There are basically two types of quality metrics. The metrics in the first category follow a vision model, whereas those in the second class are oriented to detect specific artifacts in the processed video signal.

Various vision-model-based quality metrics differ in many aspects. These include the phenomena they account for, their complexity and their desired output. In this work, we have used the Video Quality Model (VQM) to evaluate the quality of the transrated video [36]. The VQM measures the degradation of the test sequence, as compared to a reference sequence. At first, a perceptual filter is applied to both decoded video sequences, to enhance some percieved property, such as edge information, motion flow and local contrast. Then, perceptual features are extracted from small spatio-temporal subregions. For each such spatio-temporal region, the test features are compared to the reference features. The next stage performs spatial and temporal collapsing to obtain one quality parameter that detects impairments such as blur, blocking and jerkiness. The final VQM score is a fucntion of all quality parameters, with a value ranging from 0 (no perceived impairment) to 1 (maximum perceived impairment).

The metrics in the second class look for specific artifacts, and try to evaluate the distortions' strength. The perceived quality is considered best where the impairment is minimal. The common artifacts in block-based compression schemes such as H.264 are both spatial and temporal [39, 58]. The spatial impairments include blocking and blur, where blocking is the appearance of blocks' boundaries and is most noticeable in smoothly changing regions, and blur can be defined as a loss of spatial detail. Both of which are the result of a coarse quantization of the transform coefficients. The temporal defects include jerkiness and flickering, where jerkiness is the perception of originally continuous motion as a sequence of distinct snapshots and flickering is the appearance of an unsteady light that is fluctuating with time. Flickering is especially evident when high texture regions are quantized differently over time.

Most of these metrics have a high computational complexity and are not suitable for compressed domain transrating, as they process the decoded picture in the pixel domain.

Even though the MSE is not well correlated to the human perception, an encoding scheme that uses it as the distortion metric can achieve a high perceptual quality if the design is guided by simple perceptual rules [34]. A human viewer will likely rate the overall sequence distortion as more tolerable when all frames suffer similar distortion, and not necessarily according to the average distortion [34]. Therefore, in [34, 18, 30], the authors suggest to minimize the overall distortion while equalizing the frames' distortions rather than minimizing the average distortion. We follow this assumption in Chapter 4.

The perceived distortion of different image regions may vary. For example, generally, a viewer is more interested in moving foreground objects than the background [15]. Psychovisual studies have led to the concept of a perceptual three component image model [46], where the three components are known as: 'edge', 'texture' and 'smooth'. Artifacts are more visible in smooth regions, whereas texture regions can suffer higher distortion. Therefore, [31, 15] suggest to modify the block's distortion value according to its perceptual importance. In [31], the modified distortion is used to select prediction modes, whereas in [15], it is used to select the optimal quantization step sizes. In Chapter 6, we follow [31].

High constant perceived quality is obtained when the quantization step sizes changes throughout the frame are small [33, 31, 30]. In Chapter 5, we requantize an intra-coded frame using a uniform step size, whereas in Chapter 7 we regulate the step size changes for inter-coded frame.

Chapter 3

The H.264 Coder - A Brief Overview

H.264 is currently the most powerful state of the art video coding standard. It is designed to improve the coding efficiency by a factor of about two over MPEG-2 (the same quality at half the encoded bit rate) [40, 54]. In this chapter we will briefly outline the encoder scheme and the main new coding features that enable the improvement in coding efficiency. Since our algorithmic development is based on the baseline profile (it can be extended to the other profiles), we will focus on it. Appendix A fills in the details not given in this chapter.

3.1 Encoding scheme

The basic macroblock level encoding scheme is depicted in Fig. 3.1 and in the following explanation we refer to the numbered points. At first, a prediction block (pt. 2) is subtracted from the input image block (pt. 1) to form a residual block in the pixel domain (pt. 3). This residual is then transformed and quantized to yield quantized indices (pt. 4). The decoder loop inside the encoder (denoted by a blue dotted

rectangle) performs inverse quantization and an inverse transform to generate the residual in the pixel domain (pt. 6) as would be decoded at the decoder side. The decoded output image is formed by adding (pt. 10) the decoded residual (pt. 6) to its prediction (pt. 9) and applying a deblocking filter (pt. 11). This output image is stored in the reference buffer (pt. 7) for future prediction. The coder control (pt. 12) selects the coding modes and the quantization parameters. The quantized transform indices and additional side information (such as coding modes, motion vectors, quantization parameters, etc.) are entropy coded (pt. 5) and stored at the output bitstream.

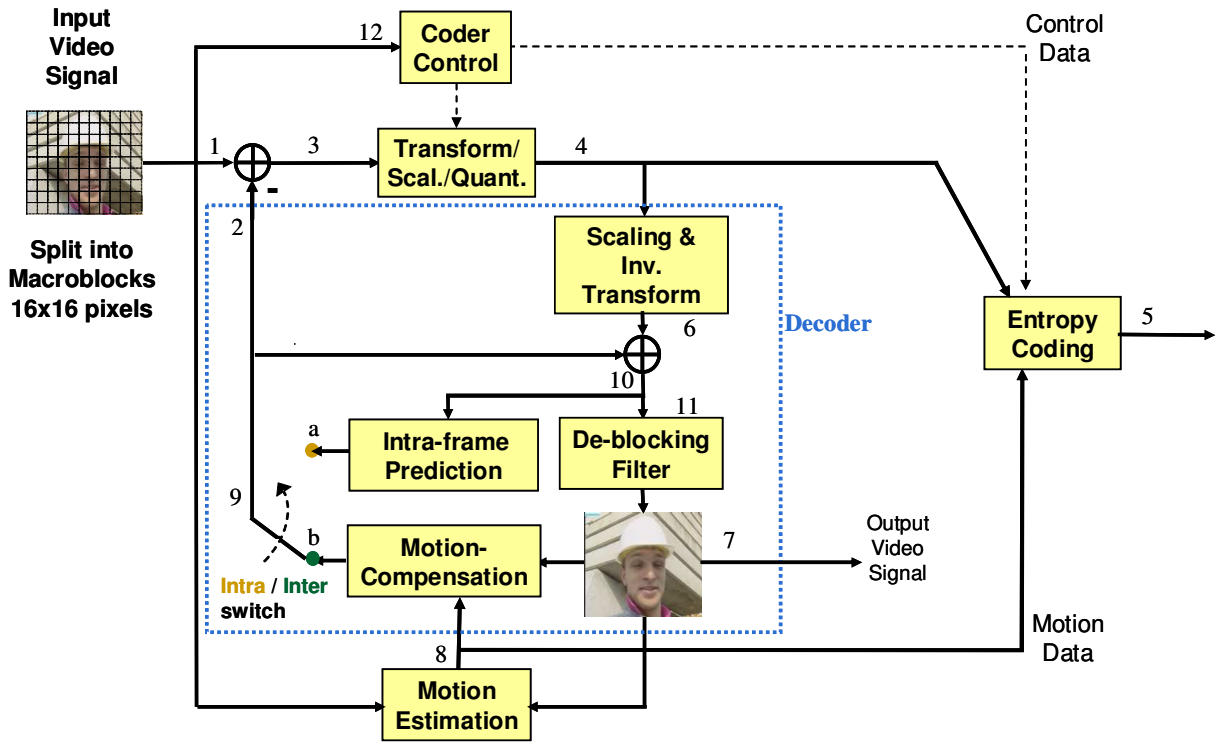

Figure 3.1: Basic macroblock level encoding scheme in H.264.

Prediction can be formed either for intra coded macroblock (MB) or for inter coded MB. Intra spatial prediction (switch at yellow position - pt. 9.a) uses previously decoded neighbor pixels in the same frame to predict the current block pixels [40]. Luminance intra blocks can be predicted either in 16x16 blocks or in 4x4 blocks

and chrominance blocks are predicted using 8x8 blocks. The 16x16 luminance prediction and the 8x8 chrominance prediction are intended for relatively smooth regions. Their prediction modes include {vertical, horizontal, DC and plane}. The 4x4 prediction is intended for coding detailed regions and includes 9 different modes: vertical, horizontal, DC and 6 diagonal patterns, all depicted in Appendix A.

Inter temporal prediction (switch at green position - pt. 9.b) uses previously decoded reference frames to predict the current block pixels [53]. The motion estimation (ME) algorithm (pt. 8) finds the best matching block for each input image block (pt. 1) from the reference frame stored in the memory (pt. 7), where a motion vector (MV) points to the best matching block location in the reference frame. It allows variable block size motion compensation, at different block sizes between 16x16 and 4x4. The motion vectors resolution is 1/4 pixel and the interpolation is performed using a separable 6-tap FIR in each direction. Both predictors output are integer valued pixels in the dynamic range of [0,255]. So, rounding and clipping operations are performed. In addition, when the decoded picture is formed by adding the decoded residual to its prediction (pt. 10), another clipping is performed to ensure the result has a valid dynamic range.

The transform defined in H.264 is carried out on small 4x4 blocks. The core transform is an Integer Cosine Transform (ICT) [28] that can be implemented at low computational cost using just shifts and adds, as it transforms integer pixel values to integer transform coefficients. To approximate the 4x4 DCT, additional scaling is required, and is incorporated in the quantization process. For smooth regions, coded as 16x16 luminance intra prediction blocks or any 8x8 chrominance blocks, some spatial correlation remains between the 4x4 transform blocks. Therefore, a Hadamard transform is carried out on the grouped DC coefficients of that smooth macroblock (where the grouped DC coefficients either form a block of size 4x4 for

16x16 luminance intra prediction blocks or of size 2x2 for a chrominance block).

The standard defines 52 quantization steps that grow logarithmically with the quantization parameter QP, where an increase of 6 in QP corresponds to a step-size factor of 2 (a factor of about $2^{\frac{1}{6}} = 1.12$ between consecutive step sizes). The step sizes defined cover a wide range, starting from 0.625 to 224. In the recommended reference software, the quantizer has a deadzone, which is wider for inter coded blocks. Since the quantization parameter QP is encoded differentially, the common H.264 rate control limits the change between consecutive macroblocks QPs. Specifically, the quantization parameter at macroblock i+1, QP_{i+1}, is typically limited to take the values $QP_{i+1} \in \{QP_i - 2, QP_i - 1, QP_i, QP_i + 1, QP_i + 2\}$.

For modeling purposes it is more convenient to define the scaled coefficients domain, where the scaled transform (no longer integer values): $Y = T(X)$, and the quantization: $Z = Quant(Y)$ are performed separately as explained in Appendix A.

The H.264 context adaptive entropy coding with VLC tables (CAVLC), is designed to take advantage of the sparse (compact energy) characteristics of the quantized transform coefficients [40]. To this end, it uses a set of syntax elements, that includes both the customary run-level representation and additional overhead counts that mainly describe the zero valued coefficients distribution. On top of that, it switches between several VLC tables for each syntax element, in a context adaptive manner.

Though the run and level are encoded separately, their encoding is efficient due to the context based VLC tables switching. The additional overhead counts consist of two symbols. One describes the combination of the number of non-zero coefficients and the high-frequency trailing-ones (± 1 at the end of the block). It is referred to as (TotalCoefficients, TrailingOnes). The other symbol, called TotalZeros, denotes the number of zeroed coefficients from the DC coefficient to the highest frequency non-zero coefficient. Both of which use multiple VLC tables.

To improve the coding gain, the recommended reference software [1] performs expensive coefficient elimination for inter-coded blocks. A sparse block is considered as a candidate for elimination if it is all zeroed, except a few trailing-ones. The decision whether or not to eliminate such a block depends on the number of its trailing-ones and their location inside the block, as further described in Appendix A.

3.2 Main differences between H.264 and MPEG-2

The main differences between the H.264 standard and the common MPEG-2 standard can be summarized in Table 3.1:

Table 3.1: Main differences between H.264 and MPEG-2 standards.

Feature	MPEG-2	H.264
Intra spatial prediction	No	Yes; supporting several block sizes
Motion compensation	Fixed block size - 16x16, $\frac{1}{2}$ pel. resolution	Variable block size, $\frac{1}{4}$ pel. resolution
Transform and quantization	8x8 DCT transform; weighted quantization	Two-stage transform (4x4 ICT followed by Hadamard for DC coefficients); logarithmically growing quantization step size
Entropy coding	Fixed VLC tables	Context adaptive
De-blocking filter	No	Adaptive in-loop de-blocking filter

Chapter 4

Proposed Transrating Scheme

In this chapter, we define the proposed transrating scheme. At first, we discuss the system's architecture, where we distinguish between the intra coded frames and the inter coded frames. Intra coded frames require full decoding and full encoding to avoid a drift error. Inter coded frames can be transrated using a closed-loop residual based architecture, with negligible quality loss. The second section defines a GOP level bit allocation algorithm that sets the target bit rate for each frame in the GOP given an average transrating factor. The last section defines a simple one-pass algorithm, to which we will compare our proposed algorithm performance.

4.1 Transrating architecture

Though the intra coded frame precede the inter coded frames, the architecture of the inter coded frames is simpler to explain and therefore will be discussed first.

4.1.1 Inter frames transrating architecture

Inter coded frames are transrated using FPDT (Fast Pixel Domain Transcoder) architecture that uses closed-loop residual-based corrections [21] (see section 2.1). This efficient architecture saves computations by decoding the frame up to its residual

transform coefficients and performing the motion compensation operation once instead of twice (during both decoding and encoding). To obtain this scheme, the three approximations below are made. In the sequel, the superscripts $(n), (n-1)$ denote the current frame and the previous frame, respectively, and the subscripts in, out denote the input frame and the output frame, respectively. The motion compensation operation is denoted by $MC(I)$, where I is the reference frame.

In addition, we assume that the in-loop deblocking filter, which is applied on the fully decoded pictures in the pixel domain [21], is disabled.

(i) No rounding and clipping take place at the decoder. That is, the decoded pictures in the pixel domain I are formed by adding the motion compensated prediction $MC()$ to the residual signal X:

$$\begin{aligned} I_{in}^{(n)} &\approx MC(I_{in}^{(n-1)}) + X_{in}^{(n)} \\ I_{out}^{(n)} &\approx MC(I_{out}^{(n-1)}) + X_{out}^{(n)} \end{aligned} \tag{4.1}$$

(ii) The prediction is linear (we neglect rounding and clipping operations):

$$MC(I_{out}^{(n-1)}) - MC(I_{in}^{(n-1)}) \approx MC(I_{out}^{(n-1)} - I_{in}^{(n-1)}) \tag{4.2}$$

(iii) We compensate for the drift so that the transrating error, measured between the input and output decoded images, is exactly the requantization error, measured between the input to the second quantizer $X^{(n)}$ and its output $X_{out}^{(n)}$:

$$\varepsilon^{(n)} = X_{out}^{(n)} - X^{(n)} = I_{out}^{(n)} - I_{in}^{(n)} \tag{4.3}$$

Thus, using (4.1)-(4.3), we find that in order to minimize the transrating error,

$$\begin{aligned} I_{out}^{(n)} - I_{in}^{(n)} &\approx MC(I_{out}^{(n-1)}) + X_{out}^{(n)} - MC(I_{in}^{(n-1)}) - X_{in}^{(n)} \\ &\approx MC(I_{out}^{(n-1)} - I_{in}^{(n-1)}) + X_{out}^{(n)} - X_{in}^{(n)} \\ &= MC(I_{out}^{(n-1)} - I_{in}^{(n-1)}) + X^{(n)} + \varepsilon^{(n)} - X_{in}^{(n)} \end{aligned} \tag{4.4}$$

we should define the signal to be requantized as

$$X^{(n)} \triangleq X_{in}^{(n)} - MC(I_{out}^{(n-1)} - I_{in}^{(n-1)}) = X_{in}^{(n)} - MC(\varepsilon^{(n-1)}) \tag{4.5}$$

and that is how we define the closed loop correction signal in Fig. 4.1.

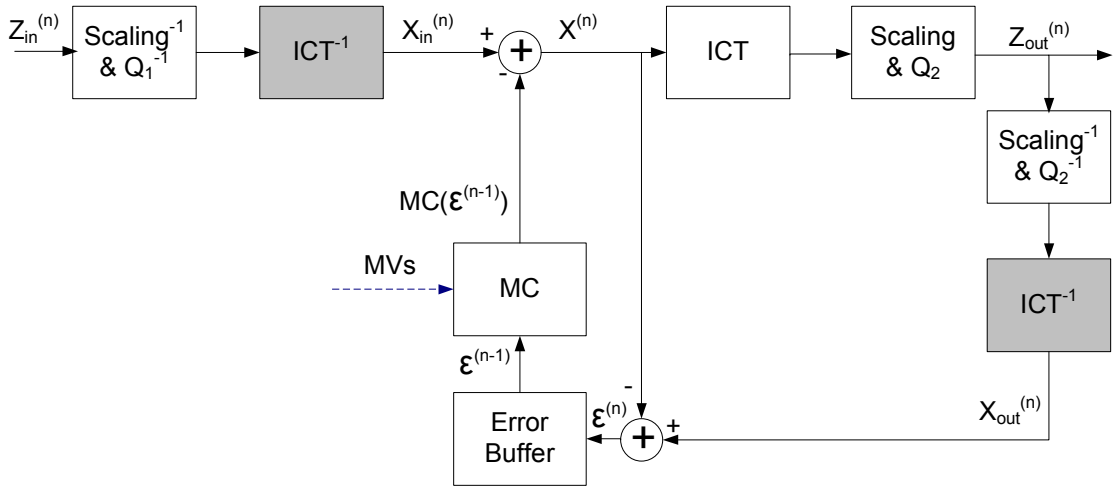

Figure 4.1: Inter frames transrating architecture based on closed-loop residual corrections. The motion compensation reuses the input MVs.

At first, the input quantized transform indices, $Z_{in}^{(n)}$, are dequantized and inverse transformed to yield $X_{in}^{(n)}$, the input residual in the pixel domain. Then, a new residual $X^{(n)}$, is formed by subtracting the correction signal $MC(\varepsilon^{(n-1)})$ from $X_{in}^{(n)}$. This correction signal is formed by feeding the requantization error from the previous frame, $\varepsilon^{(n-1)}$, into the motion compensation module, that reuses the input MVs (denoted as "side information" in Fig. 4.1). The corrected residual $X^{(n)}$ is then transformed and requantized to yield $Z_{out}^{(n)}$, the requantized transform indices. In order to find the requantization error $\varepsilon^{(n)}$ (to be used for the next frame), the output indices are dequantized and inverse transformed to yield $X_{out}^{(n)}$, the output residual in the pixel domain. The requantization error $\varepsilon^{(n)}$ is then found as the difference between the residual signals before and after the requantization: $\varepsilon^{(n)} \triangleq X_{out}^{(n)} - X^{(n)}$.

The FPDT can also work in the transform domain. However, it requires implementing MC-ICT [37], that is, motion compensation operations in the transform domain. Due to the complex sub-pixel interpolation defined in H.264 (6-tap FIR in vertical and horizontal directions), this method was not examined.

The FPDT reduces the run-time of the inter transrating architecture (measured without our rate control unit) by a factor of about 1.7, as compared to CPDT, and as a result reduces the entire inter transrating time (including the rate control unit) by about 15%. It should be noted that we are forced to reuse the input MVs while following this residual based architecture. Our attempt to modify the input MVs (using a CPDT architecture) has shown that a MV refinement search is required to avoid quality degradation. Since such refinement further increases the computational complexity, we chose to reuse the input MVs and therefore the FPDT is the chosen architecture for inter frames.

The drift error for inter coded blocks (using FPDT) is very small and it takes a number of frames before the accumulated error is noticeable. Intra coded blocks inside inter frames are transrated using the FPDT too (with the appropriate changes, e.g. the MC block is replaced by the spatial predictor, etc.) though this is not the recommended architecture for them as will be explained in section 4.1.2. Therefore, transrating inter frames with many intra coded blocks using FPDT architecture do cause some drift, but these cases are rather infrequent.

4.1.2 Intra frames transrating architecture

The spatial prediction in intra frames use previously decoded neighbor pixels in the same frame to predict the current block pixels. Therefore, any mismatch between the transcoder and the encoder/decoder introduces a drift error that propagates throughout the same picture [21]. Since some of the operations are not linear, a FPDT scheme (similar to Fig. 4.1) results in a drift error that looks like Fig. 4.2. Each constellation of prediction modes has its own typical drift error, which propagates throughout

the frame both in the horizontal and the vertical directions and results in large and noticeable luminance changes. In addition, even if one requantized intra frame looks fine by itself, the temporal transition between GOPs that rely on intra frames that contain different constellations of the prediction modes is apparent and results in an unacceptable flicker at the beginning of new GOPs.

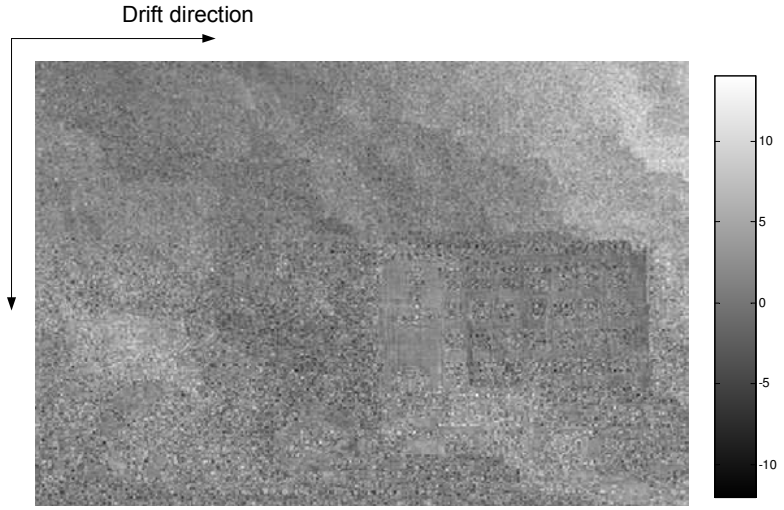

Figure 4.2: An example for the drift error (luminance component) in an intra coded frame from the 'Mobile & calendar' sequence transrated using FPDT.

The CPDT architecture depicted in Fig. 4.3 assures there will be no drift. In the sequel, we will denote the intra prediction from the reference I by $IntraPred^*(I)$. At the decoder side, the input residual X_{in} is decoded according to the input quantized indices Z_{in} using dequantization and inverse transform operations. Then, the prediction $IntraPred^*(I_{in})$ based on the previously decoded pixels at the input I_{in} is formed and added (using the input spatial prediction modes as side information). At the encoder side, the prediction $IntraPred^*(I_{out})$ based on the previously decoded pixels at the output I_{out} is formed (using the output spatial prediction modes as side information) and subtracted to generate a new residual. This residual is transformed and requantized to yield the output quantized indices Z_{out}. The decoder

feedback in the encoder decodes the output residual X_{out} and adds it to the prediction $IntraPred^*(I_{out})$ to get the output decoded image I_{out} that is stored at the frame buffer. The scheme of Fig. 4.3 follows all non linear operations as defined in the standard (see gray blocks in Fig. 4.3); that is, rounding the inverse transform output, rounding and clipping of the predicted pixels and clipping the decoded image. Since we have to decode the input intra frame using this scheme, it is also possible to modify the prediction modes, as will be discussed in Chapter 6. Therefore, the input side information (see decoder in Fig. 4.3) may be different than the output side information (see encoder in Fig. 4.3).

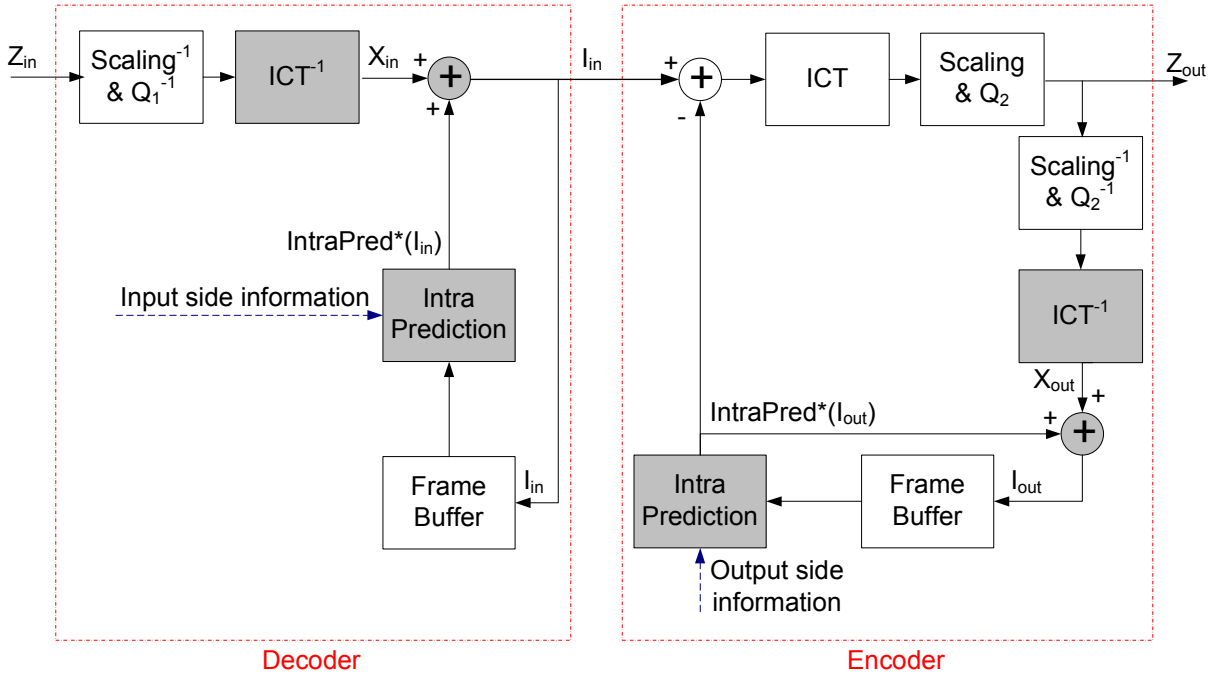

Figure 4.3: Intra frames transrating architecture based on Cascaded Pixel Domain Transcoder. Nonlinear blocks are denoted in gray. The side information contains the prediction modes for the input and the output coded frames.

4.2 Model-based optimal GOP level bit allocation

Consider transrating a GOP of N frames at an average transrating factor $BRfactor$. We use the following definitions to formulate the bit allocation problem:

$R_{in,k}$, $R_{out,k}$	- bits spent on coding the k^{th} frame in the GOP at the input and output, respectively
$R_{in,k}^{texture}$, $R_{out,k}^{texture}$	- bits spent on coding the k^{th} frame texture bits at the input and output, respectively
$R_{in,k}^{overhead}$, $R_{out,k}^{overhead}$	- bits spent on coding the k^{th} frame overhead bits at the input and output, respectively
$R_{target,k}$	- target bit rate for the k^{th} frame in the GOP
$R_{target,k}^{texture}$	- target texture bit rate for the k^{th} frame in the GOP

The overhead bits component for the k^{th} frame is composed of two parts: $R_k^{overhead,fixed}$ and $R_k^{overhead,variable}$. The $R_k^{overhead,fixed}$ bit count is not changed as a result of the transrating and it mainly describes the coding modes, MB types, slices partition, etc. The $R_k^{overhead,variable}$ bit count depends on the residual's texture bits (e.g. coding the quantization parameters and the coded block pattern) and therefore does change during the transrating. Since $R_k^{overhead,fixed}$ is the major part of $R_k^{overhead}$, we assume that the change in the overhead bits due to the transrating is negligible, so that $R_{in,k}^{overhead} = R_{out,k}^{overhead}$, for $1 \leq k \leq N$. It should be noted that in H.264, the overhead bits are not negligible and therefore the simple frame-level bit allocation described in section 2.3 is not suitable as it may leave too little texture bits for coding the residual.

Thus, we would like to find the optimal texture bits allocation between the frames of that GOP; that is $\{R_k^{texture}\}_{k=1}^{N}$. To this end, we use the frame level R-D models

(explained in Chapter 2):

$$R_k^{texture}(\rho_k) = \theta_k(1-\rho_k) \quad (4.6)$$

$$D_k(\rho_k) = \sigma_k^2 \cdot exp(-\alpha_k(1-\rho_k)) \quad (4.7)$$

where $R^{texture}$ is the texture bits, θ is the slope of the $rate-\rho$ model, D corresponds to the MSE distortion, σ^2 is the maximal MSE distortion (if all coefficients are zeroed) and α is the shape parameter of the $distortion-\rho$ model.

4.2.1 Optimization problem formulation

The first optimization problem we have examined was to minimize the overall distortion throughout the GOP, subject to the target bit rate constraint [15, 48]:

$$\min_{\{R_k^{texture}\}} \sum_{k=1}^{N} D_k(\rho_k) \quad (4.8)$$

$$subject \quad to:$$

$$\sum_{k=1}^{N} R_k^{texture}(\rho_k) \leq R_{GOP,target}^{texture}$$

As shown in Appendix B, the solution to this problem is given by:

$$R_{target,k}^{texture} = \xi_k \cdot ln(\frac{\sigma_k^2}{\xi_k}) + \frac{\xi_k}{\sum_{k=1}^{N}\xi_k} \cdot (R_{GOP,target}^{texture} - \sum_{k=1}^{N} \xi_k ln(\frac{\sigma_k^2}{\xi_k})) \quad (4.9)$$

$$D_k = \xi_k \cdot exp(\frac{\sum_{k=1}^{N}\xi_k \cdot ln(\frac{\sigma_k^2}{\xi_k}) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N}\xi_k}) = \xi_k \cdot C_1 \quad (4.10)$$

where C_1 is a constant, and

$$\xi_k = \frac{\theta_k}{\alpha_k} \quad (4.11)$$

As typically $\xi_1 > \xi_k$ for $2 \leq k \leq N$, this allocation introduces a relatively high transrating distortion for the intra frame whereas the inter frames have a low distortion.

Subjectively, the overall sequence distortion is more tolerable when all frames suffer similar distortion [34, 18, 30]. Therefore, following [5], we propose to equalize

the transrating distortion over all the frames of that GOP:
$D_1(\rho_1) = D_2(\rho_2) = ... = D_N(\rho_N)$. That way, the optimization problem formulation becomes:

$$\min_{\{R_k^{texture}\}} \sum_{k=1}^{N} D_k(\rho_k) \tag{4.12}$$
$$subject \quad to:$$
$$\sum_{k=1}^{N} R_k^{texture}(\rho_k) \leq R_{GOP,target}^{texture}$$
$$D_1(\rho_1) = D_2(\rho_2) = ... = D_N(\rho_N)$$

The solution to this problem is (see Appendix B):

$$R_{target,k}^{texture} = \xi_k \cdot [ln(\sigma_k^2) - \frac{\sum_{k=1}^{N} \xi_k \cdot ln(\sigma_k^2) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \xi_k}] \tag{4.13}$$
$$D_k = exp(\frac{\sum_{k=1}^{N} \xi_k \cdot ln(\sigma_k^2) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \xi_k}) = C_2 \tag{4.14}$$

where C_2 is a constant (independent of k). Indeed, this solution allocates more texture bits for the intra coded frame, to keep an equal distortion over all the frames.

4.2.2 Optimization procedure

1. Set $R_{GOP,target}^{texture}$ according to:

$$R_{GOP,target}^{texture} = \frac{\sum_{k=1}^{N} R_{in,k}}{BRfactor} - \sum_{k=1}^{N} R_{in,k}^{overhead} \tag{4.15}$$

2. Extract each frame's model-parameters ($k = 1, 2, ..., N$):

   (a) Evaluate the percentage of zeros at the input $\rho_{in,k}$, and extract the rate model parameter θ_k by

$$\theta_k = \frac{R_{in,k}^{texture}}{1 - \rho_{in,k}} \tag{4.16}$$

   (b) Evaluate σ_k^2 as the mean of squared coefficients.

(c) Perform simplified requantization simulation at one coarser step-size (using our definition of "scaled coefficients"), to evaluate an additional (ρ_k, D_k) point.

(d) Extract α_k from (4.7) by

$$\alpha_k = \frac{1}{1-\rho_k} \cdot ln(\frac{\sigma_k^2}{D_k}) \tag{4.17}$$

3. Find $\{R_{target,k}^{texture}\}_{k=1}^{N}$ according to (4.13) and set $R_{target,k} = R_{target,k}^{texture} + R_{in,k}^{overhead}$.

4. Update at the end of each frame's encoding:

(a) Calculate the total rate deviation from the target

$\Delta R_k = R_{target,k} - R_{out,k}$

(b) Uniformly distribute the deficit ($\Delta R_k < 0$) or surplus ($\Delta R_k > 0$) among the remaining frames in the GOP:

$$\begin{aligned} \{R_{target,j}^{texture} &= R_{target,j}^{texture} + \frac{\Delta R_k}{N-k}\}_{j=k+1}^{N-1} \\ \{R_{target,j} &= R_{target,j} + \frac{\Delta R_k}{N-k}\}_{j=k+1}^{N-1} \end{aligned} \tag{4.18}$$

Fig. 4.4 depicts an example for the rate and distortion allocations of the above optimization problems. By minimizing the overall distortion, the intra-coded frame (first in each GOP) has a lower bit allocation and therefore a higher distortion, as compared to the equal frame distortion problem.

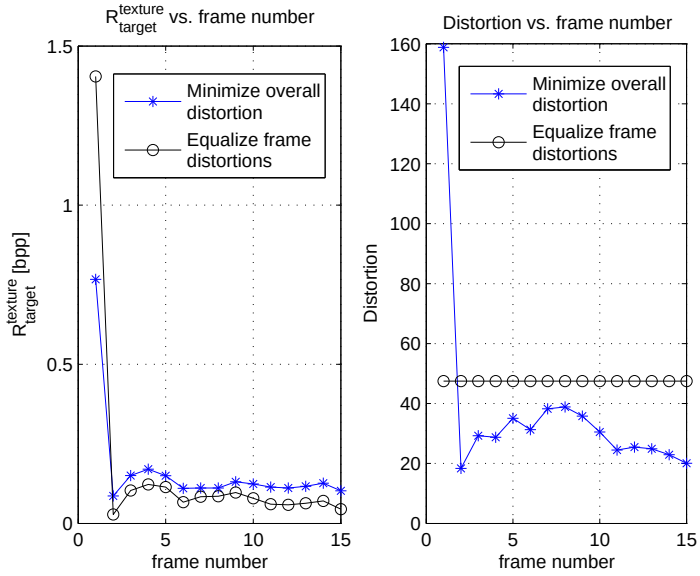

Figure 4.4: Solutions of GOP level optimization problems. Left: Texture bit allocation vs. frame number, right: distortion vs. frame number. Blue asterisk: minimize overall distortion, black circles: equalized frame distortion.

4.3 Simple reference requantization algorithm

Our reference for transrating via requantization that reuses the input prediction modes will be a simple one-pass algorithm.

The one-pass algorithm sets the requantization step at each MB according to the estimated number of bits left. Instead of using a rate− quantization step-size model, it alters the previous MB step based on a simple updating rule using buffer monitoring [25]. If an overflow is expected, we should decrease the step size and vice versa: an expected underflow indicates that the step size should be increased.

Specifically, the complexity of MB #i at the input is defined [2] as the product of its step size and its number of texture bits,

$$Complexity^{(i)} \triangleq Q_1^{(i)} \cdot r_{i,in} \tag{4.19}$$

The input prior is the set of complexities at the MB level calculated according to input encoding decisions $\{Complexity^{(i)}\}_{i=1}^{N_B}$. The buffer status is initialized to the current texture bits target, $B = R_{target,k}^{texture}$.

Before MB #i is encoded, the number of bits required to encode all the remaining macroblocks if the previous requantization step size $Q_2^{(i-1)}$ is chosen, is approximated by:

$$\hat{B}_i = \frac{1}{Q_2^{(i-1)}} \sum_{j=i}^{N_B} Complexity^{(j)} \tag{4.20}$$

Then, the requantization step size $Q_2^{(i)}$ is found via its quantization index $QP_2^{(i)}$ according to:

$$QP_2^{(i)} = \begin{cases} QP_2^{(i-1)} + 1 & \hat{B}_i > B \\ QP_2^{(i-1)} - 1 & \hat{B}_i < B \\ QP_2^{(i-1)} & otherwise \end{cases} \tag{4.21}$$

After coding MB #i, the buffer status is updated using the number of texture bits spent on requantizing that MB, $r_{i,out}$:

$$B = B - r_{i,out} \tag{4.22}$$

Chapter 5

Intra Frames Transrating - Model-based Uniform Requantization

In Chapter 4, we have defined the architecture used for transrating intra-coded frames. We concluded that the spatial prediction introduced requires a full decoding and encoding architecture in order to avoid a drift error. In the following two chapters, we discuss the transrating algorithm for such frames. The main mean for bit rate reduction in this work is via transform coefficients requantization, which is discussed in this chapter. A secondary mean for intra-coded frames is via modification of the prediction modes, that increases the coding efficiency, as discussed in Chapter 6.

For intra-coded frames, we propose using a uniform requantization for two reasons: One is that the typical bit budget for intra-coded frames is sufficiently high (as compared to inter-coded frames) as to allow a frame-level rate control. The other reason is that the spatial prediction introduces block dependencies that extremely increase the computational complexity and memory requirements of solving an optimal non-uniform requantization problem. Specifically, it involves solving a 3D trellis, where each of its states requires about $52 \cdot 7$ rate-distortion evaluations (where a practical

step size change regularization is taken).

The uniform requantization step size is found by using ρ domain models. The $rate - \rho$ model evaluation is fairly simple and is described in section 5.1. Most of the effort in this chapter is therefore aimed at estimating ρ, the expected fraction of zero coefficients, for different requantization step sizes (as described in section 5.2).

5.1 Uniform requantization using a $rate - \rho$ model

The first step in the proposed intra frame transrating scheme is to find the uniform requantization step-size. To this end, we apply a linear $rate - \rho$ model at the frame level (see Chapter 2):

$$R(\rho) = \theta(1 - \rho) \tag{5.1}$$

The model parameter θ is estimated using the input rate-ρ point, $(\rho_{in}, R_{in}^{texture})$ and the anchor point at $(1, 0)$, see Fig. 5.1. Given the texture bits target for that frame, $R_{target}^{texture}$, we extract the expected fraction of zeros ρ_{target} by

$$\rho_{target} = 1 - \frac{R_{target}^{texture}}{\theta} \tag{5.2}$$

The next step is to estimate the $\rho - Q_2$ relation, which is discussed in section 5.2. Once these values are estimated as a $\rho = f(Q_2)$ lookup table, the target requantization step-size $Q_{2,target}$ is found by

$$Q_{2,target} = f^{-1}(\rho_{target}) \tag{5.3}$$

This process is illustrated in Fig. 5.1.

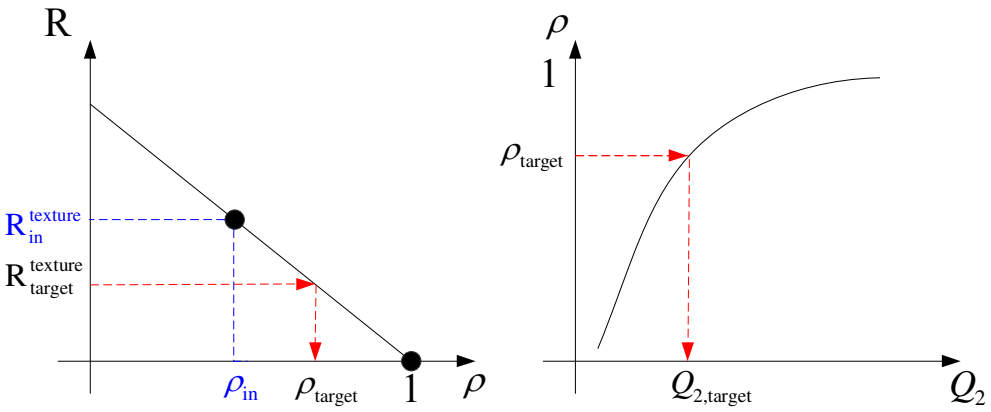

Figure 5.1: Left: $rate - \rho$ relation, the dark circles are at $(\rho_{in}, R_{in}^{texture})$ and $(1, 0)$, from which θ is estimated. Right: $\rho - Q_2$ relation. Given $R_{target}^{texture}$, we extract ρ_{target} and then find the corresponding $Q_{2,target}$.

5.2 Statistical estimation of ρ

The spatial prediction in intra frames introduces a dependency between neighboring residual blocks. Due to this dependency, the changes in the residual, as a result of transrating, propagate throughout that frame. Therefore, the residual coefficients to be requantized are not available in advance. This, of course, affects the evaluation of ρ, the fraction of zeros expected after requantization.

5.2.1 Open loop ρ estimator

The simplest ρ estimator is the open loop estimator, which is estimated from the output of the scheme depicted in Fig. 5.2. At first, let us assume that the input frame was uniformly quantized. The input quantized indices, Z_{in}, are dequantized using the input quantization step size, Q_1, to yield the residual transform coefficients Y (in the scaled transform domain). When Y is requantized using a quantizer with step size Q_2 and deadzone dz, the output indices are derived by:

$$Z_{out} = sign(Y) \cdot \lfloor \frac{|Y|}{Q_2} + dz \rfloor \tag{5.4}$$

Therefore, all transform coefficients that fall in the interval $[-(1-dz)Q_2, (1-dz)Q_2]$ are requantized to zero. The $\rho_{open-loop}$ estimator evaluates

how many transform coefficients fall in that interval. In the sequel, we will denote this interval by $Th(Q_2) = (1 - dz)Q_2$, where for intra frames $dz = \frac{1}{3}$ so $Th(Q_2) = \frac{2}{3}Q_2$.

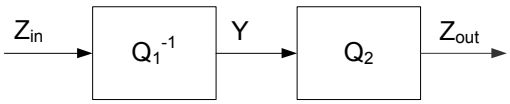

Figure 5.2: Open loop requantization scheme.

This open loop ρ estimator has two disadvantages. One is that it is not accurate enough at moderate to coarse requantization, where the changes in residual intensity are large. The other is its staircase characteristic. Given a target ρ value, the open loop estimator may encounter an uncertainty regarding the matching requantization step size, see Fig. 5.3. That is why we chose to estimate ρ more accurately using a closed-loop residual modeling architecture.

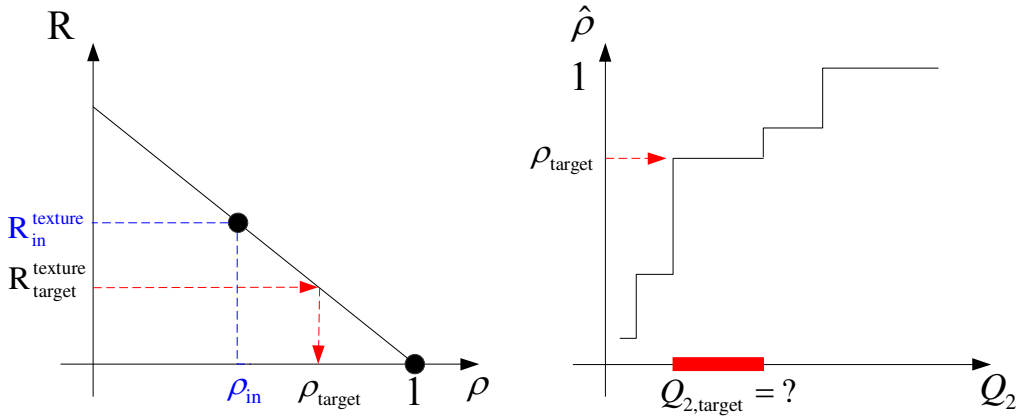

Figure 5.3: Left: $rate - \rho$ relation, right: $\rho - Q_2$ open loop estimator. There is an uncertainty regarding $Q_{2,target}$ choice.

5.2.2 Closed-loop residual modeling architecture

As explained in section 4.1.2, the proposed architecture for intra-frame transrating is CPDT, that performs full decoding and encoding. In order to estimate ρ, we use a closed-loop residual modeling architecture, as depicted in Fig. 5.4. Again, the input quantized indices, Z_{in}, are dequantized using the input quantization step size, Q_1, to yield the residual transform coefficients, Y.

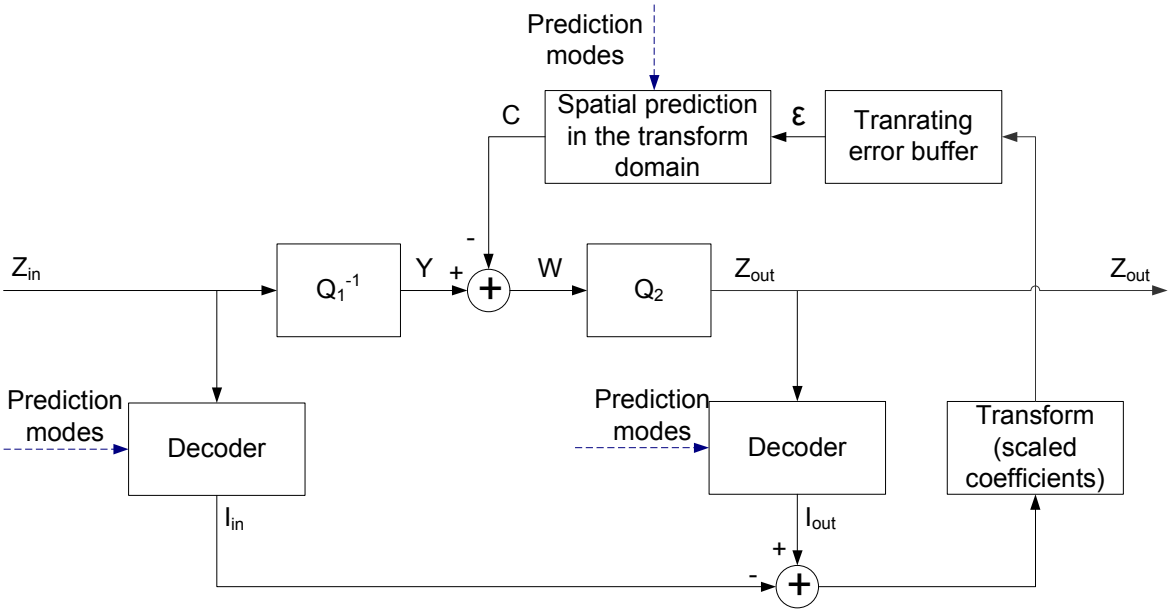

Figure 5.4: A closed-loop modeling scheme for estimating ρ. The transrating error ε is fed into the predictor to yield the correction signal C. Then, ρ is estimated using the corrected signal $W \triangleq Y - C$.

However, instead of evaluating ρ at this point, our closed loop ρ estimator evaluates how many corrected transform coefficients W fall in that deadzone interval. The corrected signal is defined as $W \triangleq Y - C$, where C is the closed-loop correction signal. This correction signal is formed by feeding the transrating error in the transform domain, ε, into the transform-domain spatial-predictor. As explained in section 4.1.2, the transrating error ε cannot be defined simply as the requantization error due to some nonlinearities (rounding and clipping operations). Rather, we define it as $\varepsilon \triangleq T(I_{out} - I_{in})$, the transform of the difference between the decoded input and output images, where the output image is decoded using the requantized indices $Z_{out} = Q_2(W)$.

It should be noted that the scheme of Fig. 5.4 is merely used in order to model the distribution of the corrected signal W, from which ρ is estimated. During actual transrating, we do not follow this scheme that calculates exactly the output Z_{out} for each step Q_2.

In order to evaluate ρ from W, we first characterize the distributions of Y and C, and then find how W is distributed. Using the estimated distribution of W, we evaluate ρ as the probability that the corrected signal W falls in the requantization deadzone:

$$\rho(Q_2) = F_W(Th(Q_2)) - F_W(-Th(Q_2)) \tag{5.5}$$

where $F_W(w)$ is the cumulative distribution of W.

Since the input transform coefficients Y have values that are multiples of the input quantization step size Q_1, their distribution is discrete, and given as:

$$p_Y(y) = \sum_{m=-M}^{M} p_m \cdot \delta(y - mQ_1) \tag{5.6}$$

where $\{p_m\}_{m=-M}^{M}$ are extracted from the input coefficients.

The correction signal C is modeled as a continuous distribution. Since this signal can not be explicitly extracted from the input stream, most of the effort in this chapter is aimed at its characterization (section 5.2.3) and its statistical modeling (section 5.2.4).

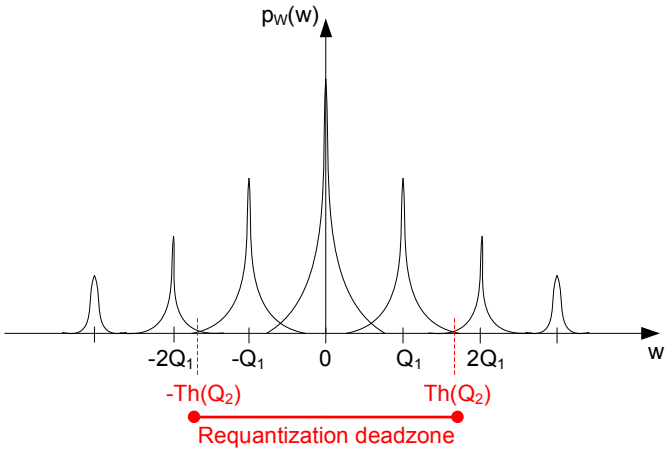

Figure 5.5: Schematic illustration of the probability distribution of $W = Y - C$. Red thick line: second quantizer deadzone

Once the distribution of C is obtained, the next step is to find the distribution of $W = Y - C = Y + (-C)$. A schematic illustration for its distribution is depicted

in Fig. 5.5. As we cannot assume that C is independent of Y, we use the joint probability of $(Y, -C)$:

$$p_{Y,-C}(y,c) = p_{-C|Y}(c|y) \cdot p_Y(y) \tag{5.7}$$

to calculate the cumulative distribution of W:

$$\begin{aligned} F_W(w_0) &= \int_{-\infty}^{\infty}\int_{-\infty}^{w_0-y} p_{Y,-C}(y,c)dcdy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{w_0-y} p_{-C|Y}(c|y) \cdot \sum_{m=-M}^{M} p_m \cdot \delta(y - mQ_1)dcdy \\ &= \int_{-\infty}^{\infty} \sum_{m=-M}^{M} p_m \cdot g(w_0 - y|y) \cdot \delta(y - mQ_1)dy \\ &= \sum_{m=-M}^{M} p_m \cdot g(w_0 - mQ_1|mQ_1), \end{aligned} \tag{5.8}$$

where $g(t|mQ_1) = \int_{-\infty}^{t} p_{-C|Y}(c|y = mQ_1)dc$.

Therefore, the closed-loop ρ evaluation is given by:

$$\rho(Q_2) = \sum_{m=-M}^{M} p_m \cdot (g(Th(Q_2) - mQ_1|Y = mQ_1) - g(-Th(Q_2) - mQ_1|Y = mQ_1)) \tag{5.9}$$

Lacking a known model for the correlation between Y and C, we are left with the unfeasible task of modeling $p_{-C|Y}(c|y = mQ_1)$, for every possible value of Y (that is $-M \leq m \leq M$). From statistical observations, we found that a reasonable approximation would be to distinguish between zero and non-zero inputs, that is, to model $p_{-C|Y}(c|Y = 0)$ and $p_{-C|Y}(c|Y \neq 0)$ separately. In that case, the model for ρ is simpler (as there are two possible input dependencies instead of $2M + 1$):

$$\begin{aligned} \rho(Q_2) = \sum_{m=-M, m\neq 0}^{M} p_m \cdot (g(Th(Q_2) - mQ_1|Y \neq 0) - g(-Th(Q_2) - mQ_1|Y \neq 0)) + \\ p_0 \cdot (g(Th(Q_2)|Y = 0) - g(-Th(Q_2)|Y = 0)) \end{aligned} \tag{5.10}$$

To complete the evaluation of $\rho(Q_2)$, we should find the conditional distributions $p_{-C|Y}(c|Y=0)$ and $p_{-C|Y}(c|Y\neq 0)$ (or their equivalents $g(t|Y=0)$ and $g(t|Y\neq 0)$), respectively, which we address in the following two subsections.

5.2.3 Correction signal characterization

In this subsection, we characterize the correction signal C, to ease its statistical modeling. To this end, the correction signal is segmented into homogenous data groups that share the same characteristics. The first partition of the data is according to its spatial prediction modes, as can be deduced from Fig. 5.4.

The second partition distinguishes the affected coefficients from the unaffected coefficients. Affected coefficients are the coefficients that are changed as a result of the spatial prediction, whereas for unaffected coefficients, the correction signal is zeroed. For example, DC prediction affects just one transform coefficient out of the 4x4 ICT block. This classification is predefined for each prediction mode by an "affected mask" whose shape is characterized by the mode's pattern in the transform domain, see Fig. 5.6.

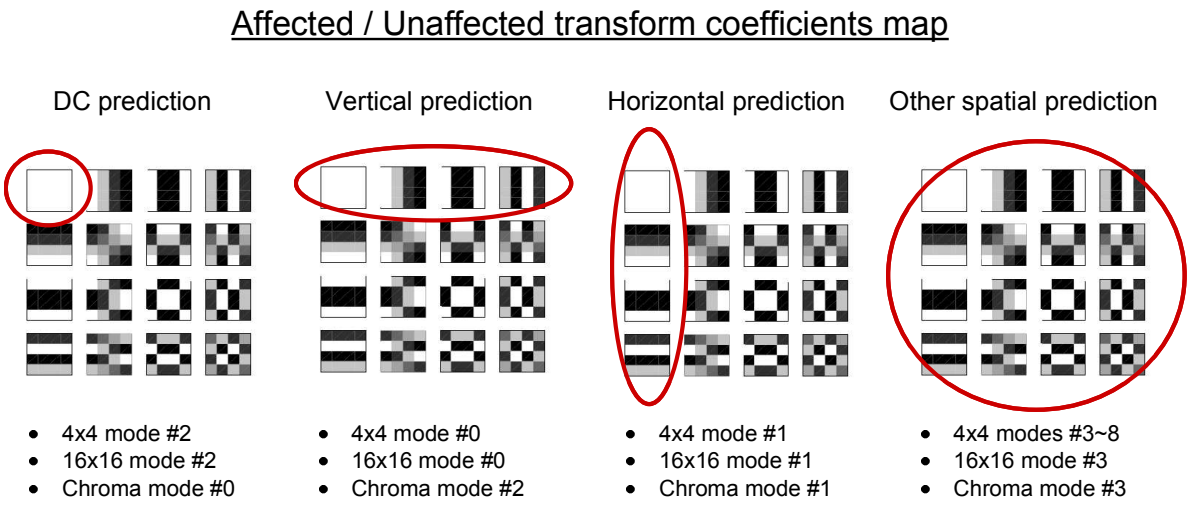

Figure 5.6: Affected / unaffected transform coefficients map denoted over the ICT basis images. The classification is done according to the prediction modes. Affected coefficients are encircled in red.

This allows two simplifications:

(i) Increased precision - the affected coefficients have a distribution which is less sparse than the distribution that describes all the coefficients together (affected and unaffected).

(ii) Complexity reduction - ρ is decomposed into the weighted average of ρ_A and ρ_U, for the affected and unaffected coefficients, respectively. ρ_U is evaluated as in the case of an open loop estimation, which is simpler.

The overall complexity reduction achieved depends on the proportions of the different prediction modes in the frame. Table 5.1 denotes the unaffected coefficients fraction for different modes.

Table 5.1: Unaffected Coefficients fraction for different prediction modes

Prediction mode	U fraction
DC (4x4, 16x16 and chrominance)	15/16
horizontal, vertical (4x4, 16x16 and chrominance)	12/16
diagonal 4x4, plane (16x16 and chrominance)	0

The third partition involves the approximation we made in section 5.2.2, that is, to distinguish between the corrections applied to zero/non-zero input coefficients.

Latter, we will note that for the case of non-uniform input quantization, a further distinction is required, based on Q_1 values.

5.2.4 Correction signal modeling using a Γ distribution

Given an offline evaluation of the correction signal C (following the scheme of Fig. 5.4), we found that the Γ distribution is a good descriptor of this signal. The probability density function for the two-sided Γ distribution is defined as [35]:

$$p_X(x) = \frac{1}{2\sqrt{\pi}}\sqrt{\frac{\beta}{|x|}} \cdot exp\{-\beta|x|\} \tag{5.11}$$

where $\beta > 0$ is the scale parameter, whose decrease results in a wider distribution. The Γ cumulative distribution function is defined by:

$$F_X(x) = \begin{cases} \frac{1}{2} + \frac{1}{2\sqrt{\pi}}\Gamma(\beta x, 0.5) & x \geq 0 \\ \frac{1}{2} - \frac{1}{2\sqrt{\pi}}\Gamma(-\beta x, 0.5) & x < 0 \end{cases} = \frac{1}{2} + sgn(x)\frac{1}{2\sqrt{\pi}}\Gamma(\beta|x|, 0.5) \qquad (5.12)$$

where the Γ function and the incomplete Γ function are given in (5.13) and (5.14), respectively.

$$\Gamma(x) = \int_0^\infty t^{x-1}exp(-t)dt \qquad (5.13)$$

$$\Gamma(a, x) = \int_0^a t^{x-1}exp(-t)dt \qquad (5.14)$$

For more details about the Γ distribution and its Maximum Likelihood (ML) estimator, see appendix C.

For each prediction mode, a ML estimator was applied to find the scale parameter β for the affected correction coefficients, while distinguishing $\beta^{C|Y=0}$ from $\beta^{C|Y\neq 0}$ for zero/non-zero input coefficients, respectively. Using these parameters, the functions $g(t|Y = 0)$ and $g(t|Y \neq 0)$ from (5.10) take the form:

$$g(t|Y = 0) = \frac{1}{2} + sgn(t)\frac{1}{2\sqrt{\pi}}\Gamma(\beta^{C|Y=0}|t|, 0.5) \qquad (5.15)$$

$$g(t|Y \neq 0) = \frac{1}{2} + sgn(t)\frac{1}{2\sqrt{\pi}}\Gamma(\beta^{C|Y\neq 0}|t|, 0.5)$$

By substituting (5.15) into (5.10), ρ was estimated for each prediction mode (only the affected coefficients group) according to:

$$\rho(Q_2) = \sum_{m=-M,m\neq 0}^{M} p_m \frac{1}{2\sqrt{\pi}} \cdot \{sgn(Th(Q_2) - mQ_1)\Gamma(\beta^{C|Y\neq 0}|Th(Q_2) - mQ_1|, 0.5) -$$
$$sgn(-Th(Q_2) - mQ_1)\Gamma(\beta^{C|Y\neq 0}| - Th(Q_2) - mQ_1|, 0.5)\} +$$
$$p_0 \cdot \frac{1}{\sqrt{\pi}}\Gamma(\beta^{C|Y=0} \cdot Th(Q_2), 0.5)$$
$$= \sum_{m=-M,m\neq 0}^{M} p_m \frac{1}{2\sqrt{\pi}} \cdot \{sgn(Th(Q_2) - mQ_1)\Gamma(\beta^{C|Y\neq 0}|Th(Q_2) - mQ_1|, 0.5) +$$
$$sgn(Th(Q_2) + mQ_1)\Gamma(\beta^{C|Y\neq 0}|Th(Q_2) + mQ_1|, 0.5)\} +$$
$$p_0 \cdot \frac{1}{\sqrt{\pi}}\Gamma(\beta^{C|Y=0} \cdot Th(Q_2), 0.5) \tag{5.16}$$

where $Th(Q_2) = \frac{2Q_2}{3}$ accounts for the second quantizer deadzone. Fig. 5.7 depicts the frame level $\rho - Q_2$ relation obtained by combining all data groups. The Γ distribution fit is derived using the ML estimated β values found during the offline evaluation.

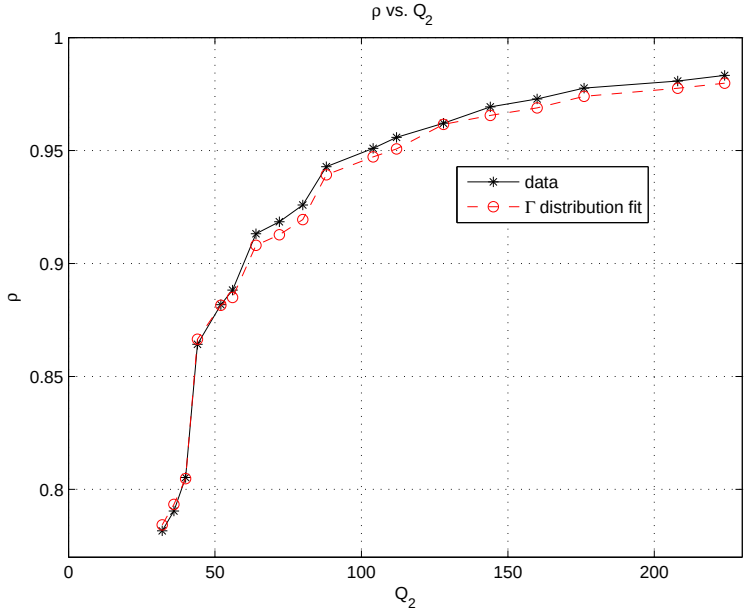

Figure 5.7: Frame level $\rho - Q_2$ relation. Black asterisk: data. Red circle: Γ-distribution fit.

In order to evaluate β in a realtime scenario, where we do not have the correction signal at hand, we need to model a β vs. Q_2 relation. An example for such curves (actually $1/\beta$ vs. Q_2) is depicted in Fig. 5.8.

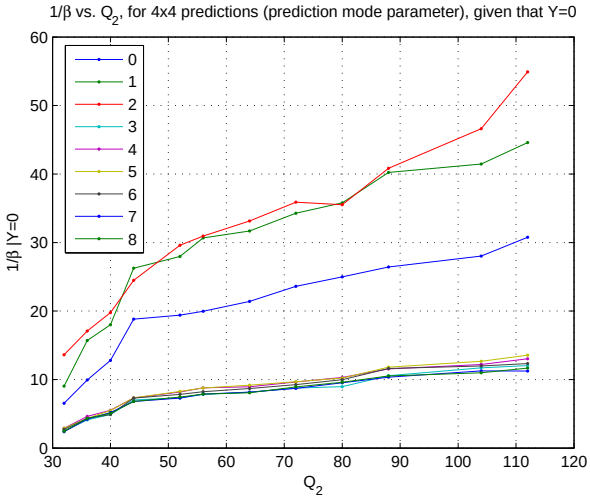

Figure 5.8: $1/\beta$ vs. Q_2 for 4x4 prediction modes (parameters), for the case of correcting a zeroed input residual ($Y = 0$).

We notice that as we increase the requantization step size Q_2, the coarser requantization generates transrating error with a wider dynamic range. When we feed this error back to the predictor, we get a correction signal with a wider dynamic range, that corresponds to a bigger $\frac{1}{\beta}$ value. Although these curves increase monotonically with Q_2, the variability of the different characteristics complicates the modeling, as this relation depends on a number of factors, such as the examined prediction mode, the value of Q_2 and some "initial conditions", such as Q_1, or $||Y||_2$.

Therefore, we suggest to decompose the β vs. Q_2 relation into two separate models: β vs. the transrating error characteristic and the transrating error characteristic vs. Q_2.

5.2.4.1 Modeling β vs. $||\varepsilon||_1$ relation

As stated earlier, a transrating error with a wide dynamic range (here, measured in terms of its $||\cdot||_1$) yields a correction with a wide dynamic range, which means a

smaller β value. Specifically, when the transrating error is zeroed, so will the closed loop correction, for all prediction modes. This observation gives us an anchor point at $\beta = \infty$.

For each combination of prediction mode (9 modes for 4x4, 4 modes for 16x16 and 4 modes for chrominance) and the zero/non-zero input, we drew graphs of $1/\beta$ vs. $||\varepsilon||_1$, where the latter is at the frame level. We then found the best slope for the main ray and modeled β as:

$$\beta = \frac{\beta_0}{||\varepsilon||_1} \tag{5.17}$$

An example for such a fit for the 4x4 diagonal down-left prediction is depicted in Fig. 5.9. The initial values β_0 for all combination of prediction modes and zero/non-zero input are given in Tables 5.2, 5.3 and 5.4.

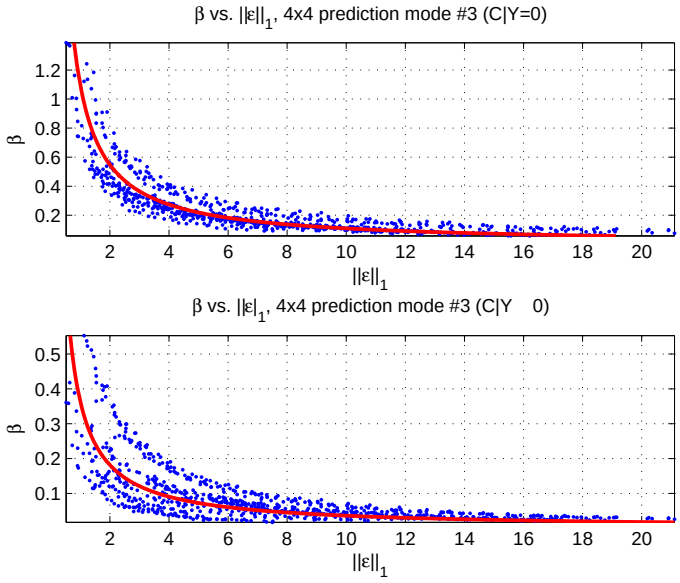

Figure 5.9: β vs. $||\varepsilon||_1$ for 4x4 prediction mode #3 (diagonal down-left). Blue: data. Red: fit. Top: $C|Y = 0$ case; bottom: $C|Y \neq 0$ case.

Table 5.2: β_0 for 4x4 prediction modes

4x4 prediction mode	$C\|Y=0$	$C\|Y\neq 0$
0 (vertical)	0.346	0.149
1 (horizontal)	0.277	0.158
2 (DC)	0.191	0.151
3 (diagonal down-left)	1.096	0.364
4 (diagonal down-right)	0.952	0.394
5 (diagonal vertical-right)	0.954	0.388
6 (diagonal horizontal-down)	0.974	0.377
7 (diagonal vertical-left)	1.066	0.365
8 (diagonal horizontal-up)	1.159	0.337

Table 5.3: β_0 for 16x16 prediction modes

16x16 prediction mode	$C\|Y=0$	$C\|Y\neq 0$
0 (vertical)	0.483	0.181
1 (horizontal)	0.420	0.247
2 (DC)	0.255	0.250
3 (plane)	4.606	0.682

Table 5.4: β_0 for Chrominance prediction modes

Chrominance prediction mode	$C\|Y=0$	$C\|Y\neq 0$
0 (DC)	0.064	0.036
1 (horizontal)	0.276	0.062
2 (vertical)	0.316	0.059
3 (plane)	0.983	0.086

5.2.4.2 Modeling $||\varepsilon||_1$ vs. Q_2 relation

As stated earlier, as we increase the requantization step size Q_2, the coarser requantization generates transrating error with a wider dynamic range. Therefore, we drew the $||\varepsilon||_1$ vs. Q_2 relation at the frame level. In this case, different "initial conditions", such as Q_1, or $||Y||_2$ generate different characteristics. Moreover, it should be noted that since our system is only approximately linear, requantization using even the input step size ($Q_2 = Q_1$) may still introduce some small transrating error. At first, we found that the following parametric model is adequate:

$$||\varepsilon||_1 = a_1 \cdot (ln(Q_2))^2 + a_2 \tag{5.18}$$

Then, we modeled a_1, a_2 as functions of the input.

$$a_1 = a_{1,1} \cdot ||Y||_2 \tag{5.19}$$

$$a_2 = Q_1 \cdot (a_{2,1}||Y||_2 + a_{2,2}||Y||_2^2) \tag{5.20}$$

where $a_{1,1}, a_{2,1}, a_{2,2}$ are given in Table 5.5 for both luminance and chrominance components.

Table 5.5: $||\varepsilon||_1$ vs. Q_2 parameters

$||\varepsilon||_1$ vs. Q_2 parameters	Luminance	Chrominance
$a_{1,1}$	0.02	0.01
$a_{2,1}$	0	-0.003
$a_{2,2}$	-0.0002	0

5.3 Summary and experimental results

In section 5.2, we evaluated the $\rho - Q_2$ relation using a closed-loop residual modeling architecture. The modeling steps are as follows:

1. Segment the transform coefficients into data groups as explained in subsection 5.2.3.

2. For each data group, evaluate the β distribution parameter from the input data at two stages:

   (a) Model the $||\varepsilon||_1$ vs. Q_2 relation using (5.18).

   (b) Model the β vs. $||\varepsilon||_1$ relation using (5.17).

   Use (5.16) to evaluate the $\rho(Q_2)$ relation for that data group.

3. Linearly weight the obtained $\rho(Q_2)$ relations for the different data parts according to their size to get the frame level $\rho(Q_2)$ relation.

If the input frame is not uniformly quantized during the first encoding, an additional data partition according to the initial quantization step is added to the data groups segmentation of subsection 5.2.3.

Fig. 5.10 depicts an example for a $\rho - Q_2$ relation at the frame level. The open loop estimator (blue asterisk) is biased as compared to the data relation (black asterisk) and has a staircase characteristic. Both Γ estimators (red and green circles) are not biased and follow the same trend as the data. The proposed model for predicting the β parameters results in an accurate estimator (green circles) with an average relative error of less than 1.7%. We examined the average rate deviation from the target, where the uniform requantization step-size was selected using different $\rho - Q_2$ estimators, as listed in Table 5.6. The true data $\rho - Q_2$ relation was used as a yardstick for the performance, as it cannot be evaluated in a real-time scenario. It shows some

small rate estimation error, mainly because of the $rate - \rho$ model's inaccuracy. Due to the inherent bias of the open-loop estimator, it tends to choose finer steps than required, at the cost of an increased rate. Therefore, it has a large rate estimation error. The proposed $\rho - Q_2$ estimator outperforms the open-loop estimator, providing a smaller rate estimation error, close to the estimation from the true data.

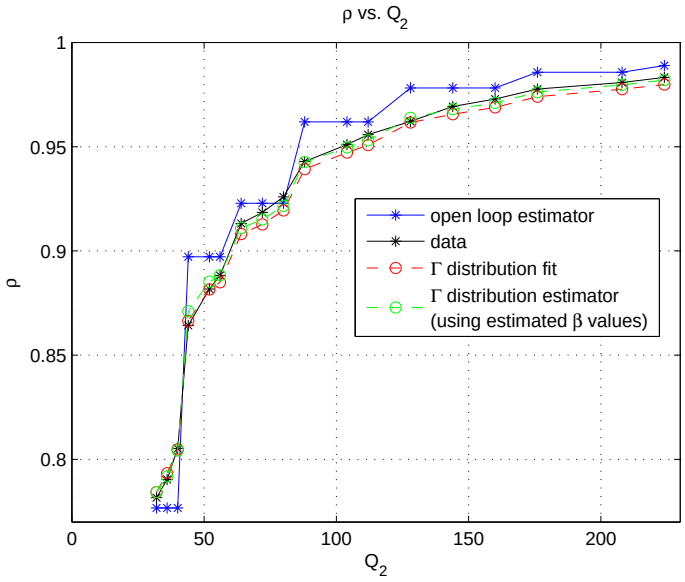

Figure 5.10: Frame level $\rho - Q_2$ relation. Blue asterisk: open loop estimator. Black asterisk: data. Red circles: Γ distribution fit. Green circles: Γ distribution estimator (using estimated β values).

Table 5.6: Mean relative rate deviation from the target.

$\rho - Q_2$ estimator	Mean relative rate deviation [%]
Data	2.5
Open-loop	10.8
Proposed	3.0

Chapter 6

Intra Frames Transrating - Modification of Prediction Modes

The proposed architecture used for transrating intra-coded frames (see section 4.1.2) requires full decoding and encoding in order to avoid a drift error. Although we have to fully decode the frame, we need *not* fully encode it by means of a computationally expensive full prediction modes search. Rather, we perform a *guided encoding*, which uses already encoded information from the input bitstream. One option is to reuse the input prediction modes. The other option is to selectively modify the input prediction modes where the coding efficiency is expected to improve. In this chapter, we propose an algorithm for selective prediction modes modification, and compare its performance with a scheme that reuses the input prediction modes and with a full re-encoding scheme.

6.1 Introduction

Spatial prediction in intra-coded frames significantly increases the coding efficiency when the coding modes are appropriately selected. As the bit rate is reduced, the quality is degraded and fine details are less likely to be preserved. The observed trend

regarding the encoder's intra coding decisions shows that as the bit rate is reduced, larger prediction blocks are chosen (more 16x16 partitions) and the frequency of "simple" modes (horizontal, vertical and DC prediction) increases at the expense of the more complex "diagonal" modes for the remaining 4x4 partitions. However, at some blocks, "complex" modes usage significantly improves the coding efficiency, so these modes cannot be completely discarded from the search.

Therefore, when the bit rate reduction is substantial (e.g., by 50 % or more), prediction modes modification is required. For a transrating application, we would like to use as much information possible from the first encoder decisions to reduce the computational complexity (as compared to full search for the new coding modes). On the other hand, a full modes search is expected to yield better results as it finds the best modes in terms of the overall rate and distortion.

In that trade-off between quality and computational complexity, we chose to limit the new modes search to the blocks whose probability to increase the coding efficiency is high. The prior is deduced from the first encoder coding decisions, such as partition size and level of bits consumption. The choice between the limited set of modes is guided by HVS considerations to improve the perceptual quality.

6.2 Low complexity mode modification using input prior

Observations from low bit rate encoding show that most of the coding gain expected is due to the modification of 4x4 modes. The blocks that were originally encoded using large partitions are usually smooth, and remain as such with the bit rate reduction. So, with high probability, large partitions remain large partitions.

Previous work [20] considered the modification of prediction modes originally coded as 4x4. The number of bits spent on coding the original MBs was used to

discern the smooth from the highly detailed MBs. MB tagged as smooth was examined to be coded using 16x16 prediction. For MB tagged as highly detailed, each 4x4 block was examined to be coded using the most probable mode, which saves overhead bits spent on coding the modes but is not necessarily the most suitable mode. The decision whether or not to change the mode was based on the distortion solely, which can yield large rate deviations, as the best mode definition is correlated with its rate-distortion cost at the current bit rate working point.

We suggest to define the prior as follows. The N_B macroblocks in the frame are first sorted in ascending order according to their input bit consumption. The lowest 30% are denoted as the G^L group, the highest 30% as the G^H group, and the rest as the G^M group, as depicted in Fig. 6.1.

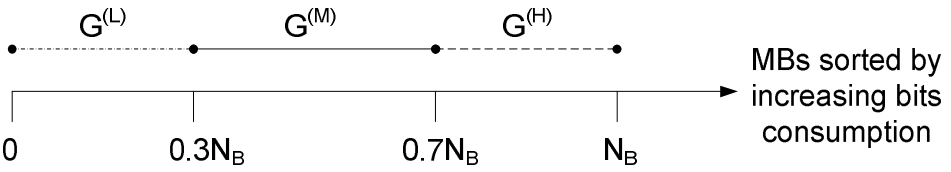

Figure 6.1: Macroblocks classification to G^L, G^M, G^H groups according to the input bits consumption.

The blocks in the G^L group, which are "non-active" at the input in terms of their bit cost, are assumed to be relatively smooth, and are therefore candidates for a 16x16 prediction. The "active" blocks at the input, that is, blocks from the G^H group, are considered to be more "problematic", and therefore require further modes examination. Since these constitute only 30% out the macroblocks, but expected to increase the coding efficiency if the best matched modes are chosen, we examine all 4x4 modes for this group. For the blocks in the G^M group, we examine just the 4x4 DC mode, as we observed that in many cases it is more suitable than the most probable mode. These decisions are illustrated in the flow chart of Fig. 6.2.

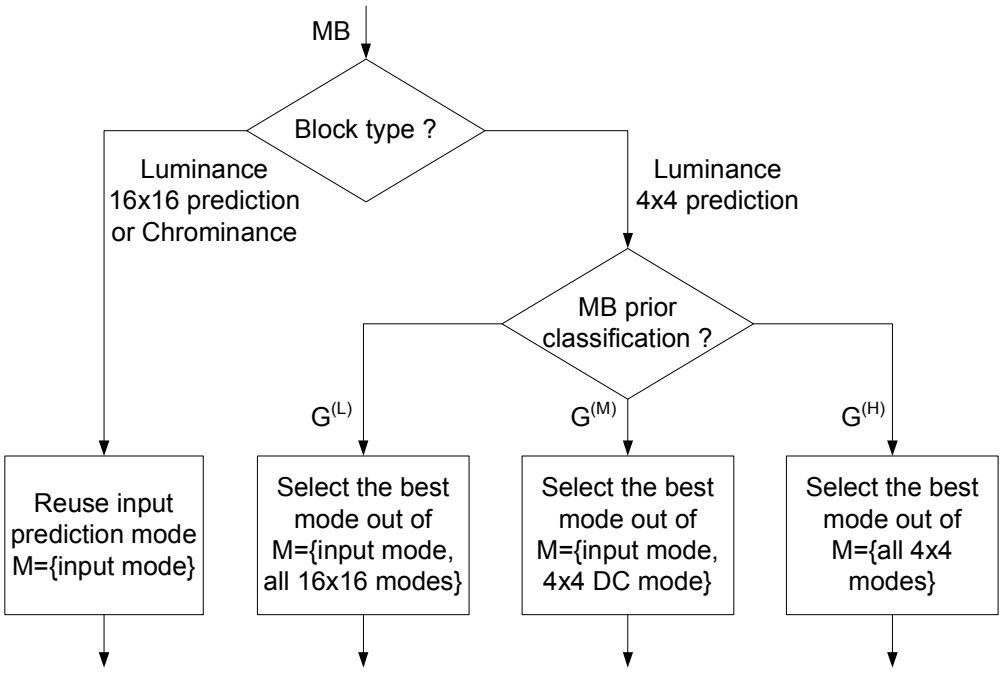

Figure 6.2: Modes examined for modification, guided by the input prior.

6.3 HVS considerations

Our visual perception of an image is not based on the collection of individual pixel intensity values. Rather, it is the interactions between them that form shapes and objects, which are the main descriptors of an image. Psychovisual studies have led to the concept of perceptual three component image model [46]. Each component has a distinct characteristic and plays a different role in the visual perception. We will now briefly review these three components, known as: 'edge', 'texture' and 'smooth'.

The most important information content is in edges. Edges are locations in an image where the intensities change abruptly or rapidly [27]. These abrupt intensity changes tend to occur at object boundaries, and are thus helpful in the representation of separate objects. The edges' high perceptual importance is also supported by neurophysiology. Most of the visual neurons in the primary areas of the visual cortex react to specific orientation intensity jumps. It was therefore concluded that the low level visual cortex performs orientation selective edge detection while processing the visual information.

However, image segmentation requires more than just edge detection. The reason is that not all of the intensity jumps correspond to boundaries of objects [27]. Thus, we should distinct between strong edges and weak edges, where the latter are referred to as textures [46]. The perceived strength of the edge is related to three properties: its intensity variation, its width and its neighboring edges. Since neighboring edges interact in an inhibitive way, the closer two edges stand, the more severely their strengths are weakened. Consequently, strong sharp edges are characterized by high intensity variation, narrow width and relatively isolated locations, whereas the weak edges have lower intensity variation, wide width and crowded edge neighbors. The phenomenon of inhibitive interaction between close weak edges is known as 'texture masking' and explains the texture's relatively low importance role in perception.

The third perceptual component corresponds to the smooth areas of the image. These areas have low spatial content, since they characterize a gradual slow variation in intensity. Nevertheless, they influence our perception together with the edge information [46] and play a more important role than textures.

In [31], the authors suggest to modify the block's distortion value according to its perceptual importance. To this end, they classify the picture macroblocks into 6 groups: {Textured, Dark Contrast, Smooth, Edge, Detailed and Normal} and define f factors for each of these. Then, the distortion measured by the MSE is weighted by the $1/f$ factors and is plugged into the rate distortion cost function. Image regions with higher perceptual importance are assigned with $f < 1$, so their distortion will weigh more, and vice versa.

We decided to distinguish just between the three main groups of: {Edge, Texture regions and Smooth regions}. Since artifacts are most apparent at smooth regions and less noticeable at textured regions, we found that
$f_{texture} = 1.2, f_{smooth} = 0.8, f_{edge} = 1$ are suitable.

We chose to perform the segmentation at the 8x8 block resolution level, as a compromise between the too fine 4x4 level and the too coarse 16x16 macroblock level. We follow [19] and calculate the variance of the block coefficients, where the DC term and the first two AC coefficients are not taken into account to avoid slow intensity changes detection. The variances map is translated into low and high activity blocks using an adaptive threshold. Morphological operations are then used to detect the edges and smooth regions and form the segmented picture at the 8x8 block level. An example for such classification for a frame from the "football" sequence is depicted in Fig. 6.3.

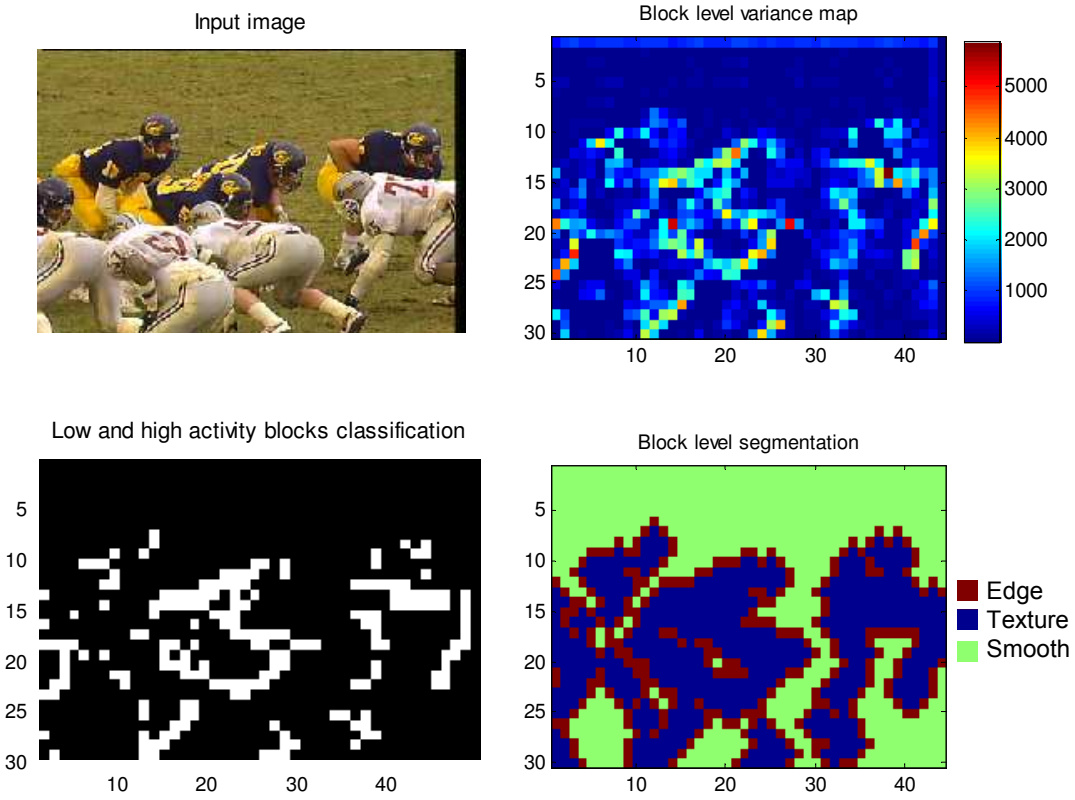

Figure 6.3: An example for picture segmentation into {edges, texture regions, smooth regions}. Upper left: Input picture; upper right: Block level variances map; bottom left: Classification into low and high activity blocks; bottom right: Block level segmentation (red: edge; blue: texture region; green: smooth region).

6.4 Suggested mode selection algorithm

Let us denote by d_i and r_i the transrating distortion and the number of bits spent for block i. The basic mode selection algorithm specified by the H.264 rate-distortion defines the Lagrangian cost function $j = d+\lambda(QP)r$, where the Lagrangian parameter λ is a function of the quantization parameter QP: $\lambda(QP) = 0.85 \cdot 2^{\frac{QP-12}{3}}$.

The prior discussed in section 6.2 can be formalized as an additive term in the cost function of block b_i:

$$cost(m|prior(b_i)) = \begin{cases} 0 & m \in M \\ \infty & else \end{cases} \tag{6.1}$$

where m is the examined mode for block b_i and M is the subset of modes defined by the prior macroblocks classification (see Fig. 6.2).

By substituting d_i by $d_i/f_{HVS}(b_i)$ in the Lagrangian cost function and adding the prior term we find m_i^*, the best mode for block b_i is given by:

$$m_i^* = \underset{m}{\operatorname{argmin}}\{d_i(m, QP) + \lambda(QP) \cdot f_{HVS}(b_i) \cdot r_i(m, QP) + cost(m|prior(b_i))\} \tag{6.2}$$

6.5 Experimental results

We compare the following three intra transrating schemes:

- Re-encoding - Performs a full search for the prediction modes.
- Proposed selective modes modification
- Reuse of input modes.

The comparison is made in terms of computational complexity and quality. All three schemes use a *uniform requantization step size* found as explained in Chapter 5.

Fig. 6.4 depicts the average run-time measured during the transrating of intra-coded frames from different video sequences. As expected, the re-encoding scheme

has the highest computational complexity among the three schemes, whereas reusing the input prediction modes has the lowest complexity. Reusing the input prediction modes reduces the run-time by a factor of about 4.5, on average, as compared to re-encoding, whereas for the selective modes modification scheme, the run-time reduction factor is only about 1.6, on average.

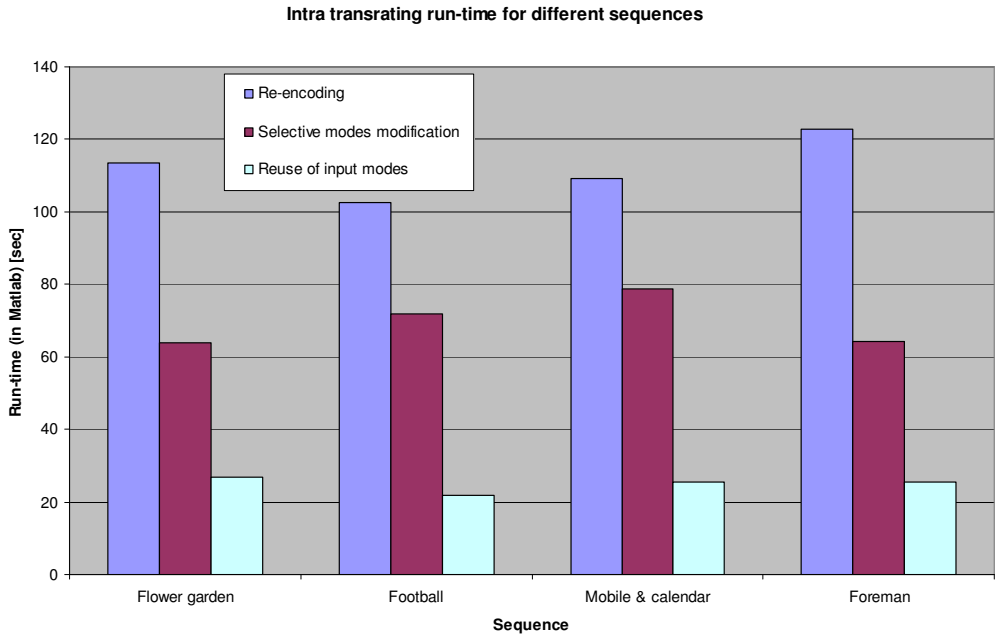

Figure 6.4: Run-time comparison for different intra transrating algorithms: re-encoding, selective modes modification, and reuse of input modes.

Fig. 6.5 depicts a typical quality comparison in terms of PSNR vs. bit rate. It reveals two interesting results:

- The proposed selective modes modification scheme consistently outperforms the scheme that reuses the input prediction modes. The PSNR gain is up to 1[dB].
- The proposed selective modes modification scheme practically reaches the re-encoding performance bound.

Since the PSNR does not reflect the perceived quality, we bring the decoded pictures obtained at the lower bit rate end of Fig. 6.5 as an example. Fig. 6.6(a), Fig.

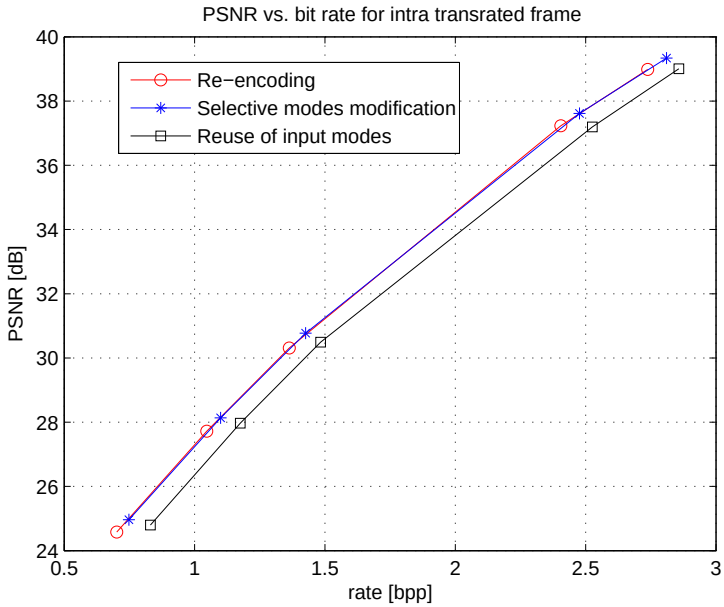

Figure 6.5: PSNR vs. bit rate comparison. Red circle: re-encoding. Blue asterisk: selective modes modification. Black square: reuse of input modes.

6.6(b) and Fig. 6.6(c) show the decoded pictures obtained by reuse of the input modes, selective modes modification and re-encoding, respectively. As we compare the schemes at a uniform requantization step-size, neither the bit rates obtained nor the PSNRs are equal. By comparing Fig. 6.6(b) to Fig. 6.6(a) , we notice that the perceived quality of Fig. 6.6(b) is higher at a lower bit rate. It is most apparent at the sky region, where the block artifacts are less noticeable. Fig. 6.6(c) has a lower PSNR at a lower bit rate, as compared both to Fig. 6.6(a) and to Fig. 6.6(b). However, from Fig. 6.5 we note that the re-encoding scheme and the selective modes modification scheme have practically the same performance, so it can be regarded as a different working point of the selective modes modification curve.

The proposed selective modes modification scheme has practically the same performance as the re-encoding scheme in terms of PSNR vs. bit rate at about 37.5% less computations. Therefore, our general recommendation would be to choose it over reusing the input modes. The reuse of input modes is faster and more suitable for small transrating factors, where the transrated frame prediction modes are expected to be similar to the input modes.

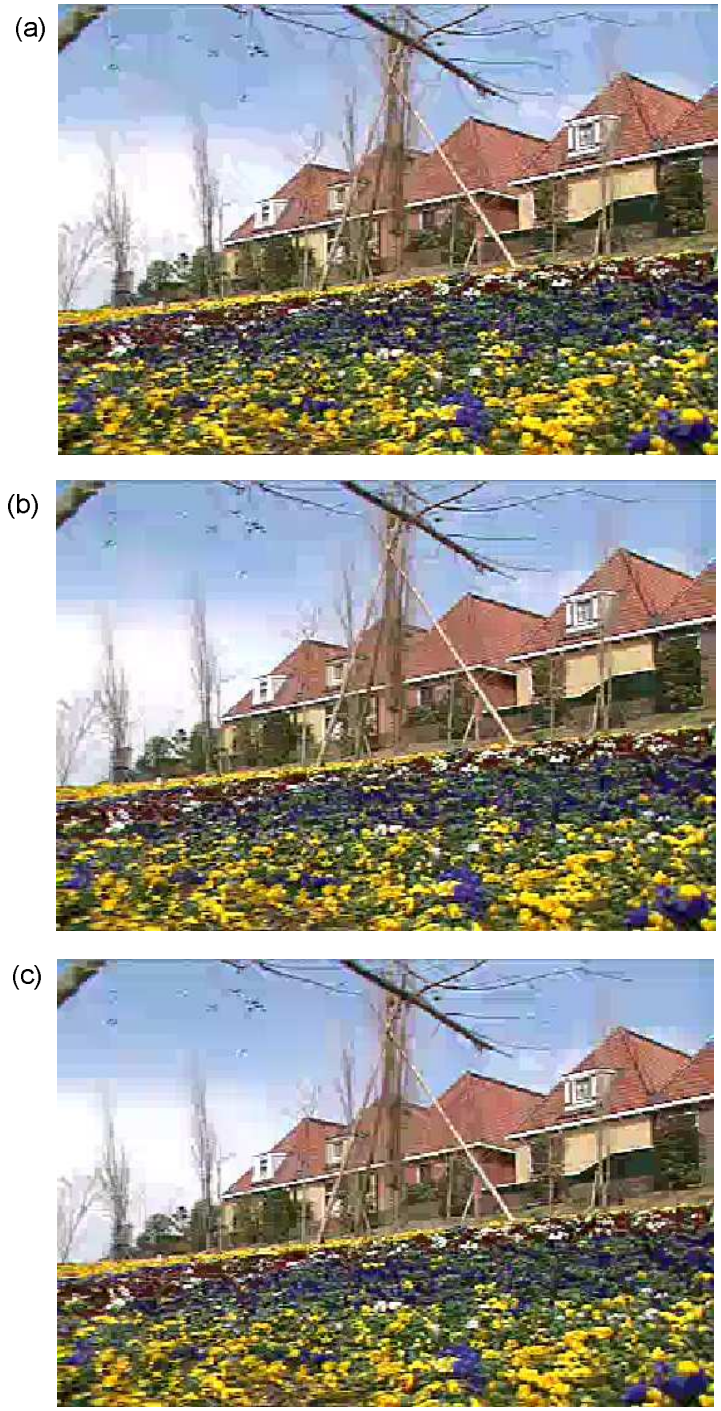

Figure 6.6: (a) Reuse of input modes: bit rate= 0.83[bpp], PSNR= 24.8[dB]. (b) Selective modes modification: bit rate= 0.75[bpp], PSNR= 24.9[dB]. (c) Re-encoding: bit rate= 0.7[bpp], PSNR= 24.6[dB].

Chapter 7

Inter Frames Transrating - Optimal Requantization

In Chapter 4, we have defined the architecture used for transrating inter-coded frames. This architecture uses a closed-loop residual corrections scheme, which also reuses the input motion decisions. In this chapter, we discuss the bit rate reduction of inter-coded frames via transform coefficients requantization and optional coefficients elimination. Since the typical bit budget for inter-coded frames is low (as compared to intra-coded frames), the rate control should be accurate in order to meet the target bit rate. Therefore, we propose an optimal *non-uniform* requantization. In section 7.1, we discuss the definition and the solution of the optimal requantization problem. In section 7.2, we examine the incorporation of selective coefficients elimination into the optimal requantization.

7.1 Optimal requantization

7.1.1 Introduction

In previous standards, like MPEG-2, the optimal requantization problem is defined as finding a set of optimal new step-sizes, where optimality is in the sense of minimizing

the total distortion, subject to a given bit-rate constraint:

$$\min_{\{QP_i\}} \quad D, \quad subject \quad to \quad R \leq R_{target} \tag{7.1}$$

where

$$D = \sum_{i=1}^{N_B} d_i(QP_i)\,,\; R = \sum_{i=1}^{N_B} r_i(QP_i) \tag{7.2}$$

and

N_B - number of macroblocks in the frame

QP_i - quantization parameter for the i-th macroblock

d_i - distortion caused to the i-th macroblock

r_i - number of bits produced by the i-th requantized macroblock

A common approach [6] is to convert the constrained optimization problem to an unconstrained one:

$$\min_{\{QP_i\}} \quad J, \quad J = D + \lambda(R - R_{target}) \tag{7.3}$$

where λ is the Lagrangian parameter. The main advantage of solving the unconstrained problem is that the cost J can be broken into a sum of independent costs for each macroblock. Given a λ value, the set of quantization steps $\{QP_i^*\}_{i=1}^{N_B}$ that minimizes the set of independent costs is found and the corresponding average rate is calculated by $\sum_{i=1}^{N_B} r_i(QP_i^*)$. Then, the λ parameter is altered, using for instance, bisection iterations, until an average rate that is close enough to the target is obtained.

In [33, 31, 30], it is argued that small fluctuations in the quantization step size throughout the frame yield better subjective quality, as the overall perceived frame's quality appears constant and blocking artifacts are reduced. In addition, the H.264 standard encodes the quantization parameter differentially, that is, it encodes $\Delta QP = QP - QP_{Prev}$, where QP_{Prev}, QP are the quantization parameters of consecutive macroblocks. The cost in bits of the ΔQP transition increases with its absolute value, such that:

$$cost(QP_{Prev}, QP) = cost(\Delta QP) = \begin{cases} 1 & \Delta QP = 0 \\ 3 & |\Delta QP| = 1 \\ 5 & 2 \leq |\Delta QP| \leq 3 \\ 7 & 4 \leq |\Delta QP| \leq 7 \\ 9 & 8 \leq |\Delta QP| \leq 15 \\ 11 & 16 \leq |\Delta QP| \leq 31 \\ etc. & \end{cases} \tag{7.4}$$

As a result, many rate control algorithms for H.264 limit $|\Delta QP|$ to take small values (up to 2). In subsection 7.1.2 we examine an optimal requantization problem where $|\Delta QP|$ is not explicitly limited but is regulated according to its bits cost. Based on these results, in subsection 7.1.3, we explicitly limit $|\Delta QP|$ to reduce the computational complexity.

7.1.2 Full solution

Following the assumption that the change in the overhead bits due to the transrating is negligible (see section 4.2), we define the optimization problem in terms of the texture bits:

$$\min_{\{QP_i\}} D \quad subject \quad to \quad R^{texture} \leq R^{texture}_{target} \tag{7.5}$$

and convert this constrained problem to an unconstrained problem by introducing the Lagrangian parameter λ:

$$\min_{\{QP_i\}} J, \quad J = D + \lambda(R^{texture} - R^{texture}_{target}) \tag{7.6}$$

In addition, we propose to regulate the changes in QP to achieve better subjective quality by adding a regularization term $\mu \cdot \sum_{i=2}^{N_B} cost(\Delta QP_i)$ that accounts for the cost of coding ΔQP:

$$\min_{\{QP_i\}} J, \quad J = D + \lambda(R^{texture} - R^{texture}_{target}) + \mu \cdot \sum_{i=2}^{N_B} cost(\Delta QP_i) \tag{7.7}$$

where $\Delta QP_i = QP_i - QP_{i-1}$, the transition cost is according to (7.4), and μ is its relative weight in the joint cost function. Specifically, the weight parameter μ translates the regularization term measured in bits to distortion units. As we do not try to achieve an exact bit target for coding ΔQP, we do not know how to set (and refine) the value of μ. Therefore, we choose to set $\mu = \lambda$, that has the same units to simplify the solution:

$$\min_{\{QP_i\}} \quad J, \quad J = D + \lambda(R^{texture} - R^{texture}_{target}) + \lambda \sum_{i=2}^{N_B} cost(\Delta QP_i) \tag{7.8}$$

Since the choices of quantization step sizes for different macroblocks are no longer independent, the whole set of quantization step-sizes $\{QP_i^*\}$ should be found at once. Therefore, we propose to extend each Lagrangian iteration with a dynamic programming stage. The external Lagrangian iterations change the Lagrangian parameter λ to improve the rate guess. At each examined value of λ, the dynamic programming algorithm finds an optimal QP path by solving (7.8).

The dynamic programming algorithm is defined over the set of states $\{(QP, i)\}$, where i is the macroblock index and QP is the quantization index, see Fig. 7.1. Each state (QP, i) has its cost-value $j_i(QP) = d_i(QP) + \lambda r_i(QP)$, where $r_i(QP)$ includes only the texture bits, and the total frame's cost along a path is $J = \sum_{i=1}^{N_B} j_i(QP)$.

The optimal path up to the state (QP, i) is the path that has the minimal accumulated cost, $V_i(QP^*)$, over all possible paths that end at that state [34]. There are multiple possible paths that end at the previous macroblock (#i-1) and that can be continued to the current state (QP, i). We choose among these by minimizing the value function of the current state:

$$V_i(QP) = V_{i-1}(QP_{Prev}) + j_i(QP) + \lambda \cdot cost(QP_{Prev}, QP) \tag{7.9}$$

where QP_{Prev} can take each of the 52 quantization parameters defined by the standard ($0 \leq QP_{Prev} \leq 51$). It is the sum of the cost of the path until the previous macroblock $V_{i-1}(QP_{Prev})$, plus the cost of the current state $j_i(QP)$, plus the cost of moving from

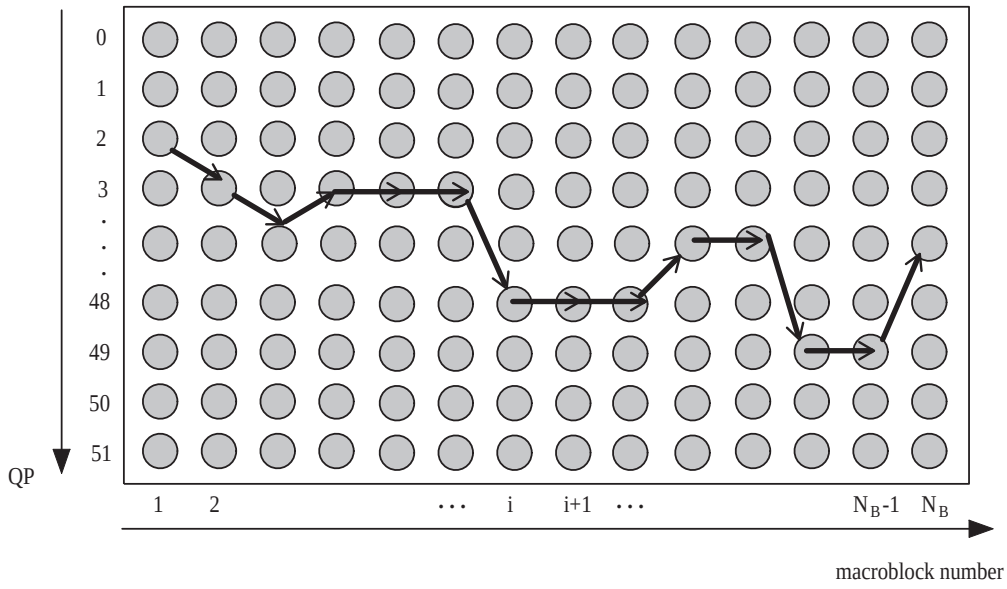

Figure 7.1: Dynamic programming path illustration. Horizontal axis: macroblock number, vertical axis: the quantization parameter QP. Each state is denoted by a circle, and each stage corresponds to one macroblock (denoted by a column of circles). The arrows show a path example, with a typical small change in QP from one macroblock to the next.

state $(QP_{Prev}, i-1)$ to (QP, i), which is defined in (7.4). Or, in other words, the best path up to state (QP, i) is continued from state $(QP^*_{Prev}, i-1)$, where

$$QP^*_{Prev} = \arg\min_{QP_{Prev}} \{V_{i-1}(QP_{Prev}) + \lambda \cdot cost(QP_{Prev}, QP)\} \tag{7.10}$$

The corresponding value function update is then:

$$V_i(QP) = V_{i-1}(QP^*_{Prev}) + j_i(QP) + \lambda \cdot cost(QP^*_{Prev}, QP) \tag{7.11}$$

At each stage i of the dynamic programming algorithm (from the first to the last macroblock), the best paths for all (QP, i) states are found and kept as lists of pointers, along with their values. When the algorithm reaches the last stage ($i = N_B$), the optimal path is the optimal path over the entire frame:

$$BestPathEnd = \operatorname{argmin}_{QP} V_{N_B}(QP) \tag{7.12}$$

The algorithm then traces back the optimal frame path using the chosen list of pointers, to obtain the optimal path: $\{QP^*_i\}_{i=1}^{N_B}$.

In order to update the Lagrangian parameter λ before the next Lagrangian iteration, we evaluate the frame's overall texture bit rate obtained by requantization using the optimal path:

$$R_{\lambda}^{texture} = \sum_{i=1}^{N_B} r_i(QP_i^*) \tag{7.13}$$

If $R_{\lambda}^{texture} > R_{target}^{texture}$, λ is increased, and vice versa.

7.1.3 Practical constrained optimization problem

Examination of the optimal solution shows that the algorithm rarely chooses $|\Delta QP|$ values bigger than 3, see red circles curve in Fig. 7.2. Most rate control algorithms for H.264 actually limit $|\Delta QP|$ to take values up to 2. As the cost of the transition by 2 units is equal to that of a transition by 3 units (7.4) and there is no practical need for larger $|\Delta QP|$, we limit the allowed transition to $|\Delta QP| \leq 3$.

The optimization problem is then defined by:

$$\min_{\{QP_i\}} D \quad subject \quad to \quad R^{texture} \leq R_{target}^{texture} \quad and \quad |\Delta QP| \leq 3 \tag{7.14}$$

and is solved as explained in subsection 7.1.2, where at each Lagrangian iteration, the dynamic programming algorithm solves:

$$\min_{\{QP_i\}} J \quad subject \quad to \quad |\Delta QP| \leq 3 \tag{7.15}$$

considering the cost of changing QP from one macroblock to the next. This way, the dynamic programming examines only 7 possible QP_{Prev} states (for $QP_{Prev} \in \{QP-3, QP-2, QP-1, QP, QP+1, QP+2, QP+3\}$), rather than all 52 options.

The average $|\Delta QP|$ distribution obtained for this sub-optimal algorithm is depicted by blue asterisk in Fig. 7.2 and is practically the same as the distribution chosen by the optimal algorithm. The PSNR loss as a result of solving the sub-optimal algorithm is negligible and the system's overall computational complexity is reduced by at most 8%. Therefore, we propose to use the sub-optimal algorithm.

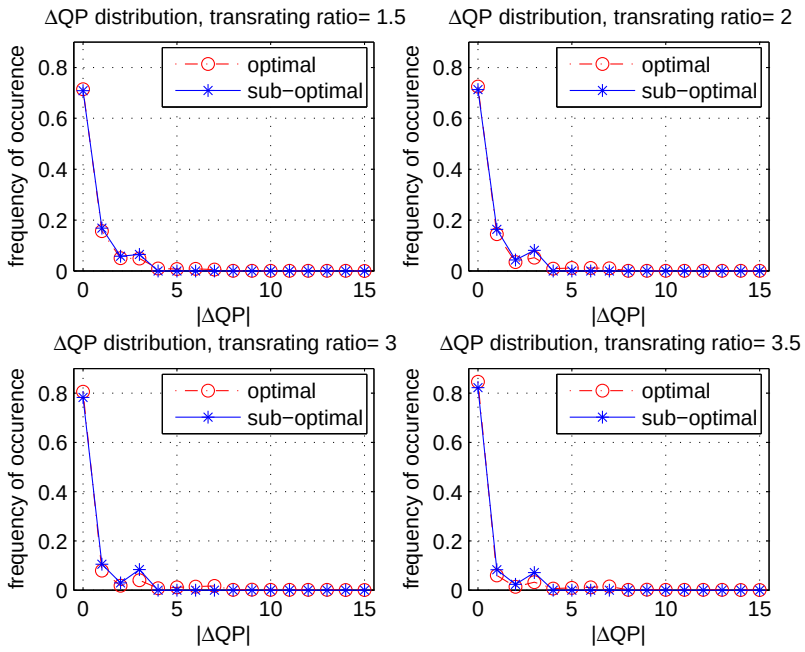

Figure 7.2: Average $|\Delta QP|$ distribution at different transrating ratios (Upper left: 1.5. Upper right: 2. Bottom left: 3. Bottom right: 3.5). Red circles: Optimal algorithm. Blue asterisk: sub-optimal algorithm that limits ΔQP so that $|\Delta QP| \leq 3$.

7.2 Selective Coefficient Elimination

After applying the transform and quantization, the quantized indices blocks are typically sparse, see example in Fig. 7.3. At the encoder, or the transcoder for that matter, it is possible to modify the obtained indices levels to achieve a lower cost, in terms of rate-distortion.

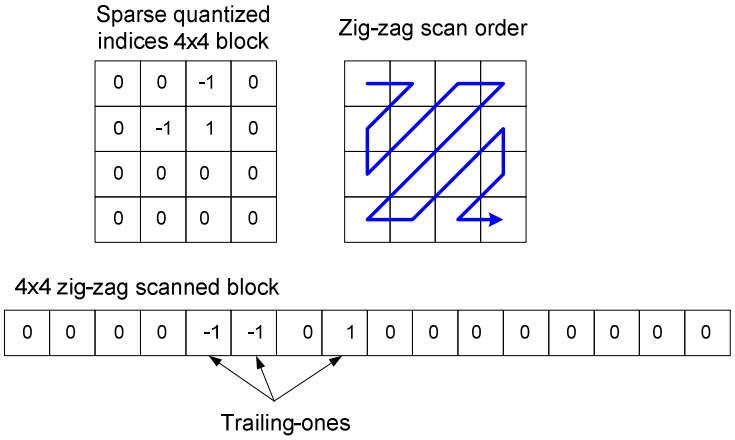

Figure 7.3: An example for a sparse 4x4 quantized indices block, which is all-zeroed except for three trailing-ones.

Previous works examined indices modification by evaluating the modified costs exhaustively, that is, evaluate a few optional rates directly from the entropy coding tables *without using models*. In [19], optimal requantization step sizes selection was extended by indices modification while minimizing the overall frame level cost. That work was done under the MPEG-2 standard, using its VLC tables. In [52], quantized indices modification was examined for a given step size for H.264 encoding. That work examined the optimal modification at the 4x4 block level, using H.264 CAVLC tables.

A simpler case of indices modification is coefficient elimination, or thresholding. In [17], the authors examined excluding AC coefficients in MPEG-2 intra coded frames only, as the only mean for bit rate reduction in a transcoding scheme. In [10], the elimination of the last coefficients in the zigzag scan was examined for frame level optimization. Finally, [7] considers coefficients elimination for the H.264 encoder. It examines the elimination of inter-coded blocks using the reference software elimination rule (see appendix A). It zeroes sparse blocks that are almost zeroed except for a few trailing-ones that correspond to transform coefficients at high frequencies.

We examined incorporating selective coefficient elimination into the proposed rate-distortion optimization algorithm. To reduce the computational load regarding which coefficient to eliminate, we follow the simple elimination rule used in the recommended reference software. We optimally choose for each quantized MB whether to encode it as is or to perform coefficient elimination first.

7.2.1 Optimal selective elimination algorithm

To allow the selective elimination, we evaluate two rate-distortion pairs for each combination of quantization parameter QP and macroblock index i. These are denoted by $\{d_i^0(QP), r_i^0(QP)\}$ and $\{d_i^1(QP), r_i^1(QP)\}$, for the case of no elimination and the

case of elimination according to the reference software rule, respectively. As a result, a 3D array for the rate and the distortion is generated over the set of states $\{(QP, i, elim)\}$, where $elim \in \{0, 1\}$ is a binary flag that denotes whether or not elimination is performed, see Fig. 7.4.

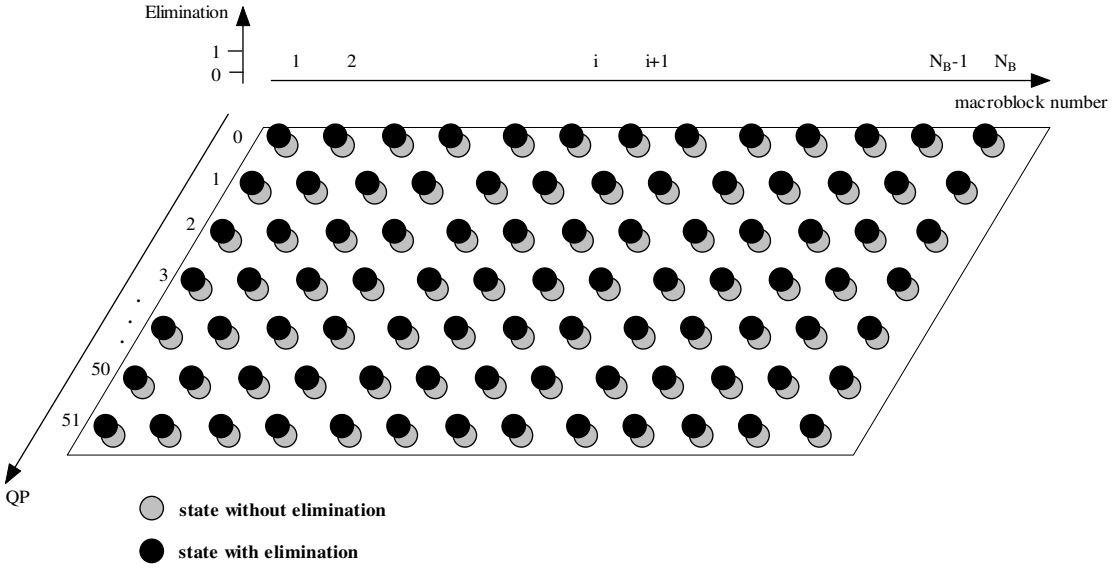

Figure 7.4: 3D trellis illustration. Horizontal axis: macroblock number, vertical axis: the quantization parameter QP. Each disc denotes a state, where the black and gray colors correspond to states with and without elimination, respectively.

Now, the extended trellis algorithm can optimally choose at which states to perform the elimination. The optimal path up to state $(QP, i, elim)$ is the path that has the minimal accumulated cost, $V_i^{elim}(QP^*)$, over all possible paths that end at that state. There are at most 14 possible sub-paths that end at the previous macroblock (#i-1) and that can be continued to the current state $(QP, i, elim)$, as there are no more than 7 allowable QP_{Prev} due to the ΔQP limitation, and each has two optional values: $V_{i-1}^0(QP_{Prev})$ in case no elimination was performed and $V_{i-1}^1(QP_{Prev})$ in case the elimination was performed at the sub-path end.

The best path up to state $(QP, i, elim)$ is defined by continuing from state $(QP_{Prev}^*, i-1, elim_{Prev}^*)$, where

$$(elim^*_{Prev}, QP^*_{Prev}) = \tag{7.16}$$
$$\arg\min_{elim_{Prev}} \arg\min_{QP_{Prev}} \{V_{i-1}^{elim_{Prev}}(QP_{Prev}) + j_i^{elim}(QP) + \lambda \cdot cost(QP_{Prev}, QP)\}$$

From (7.16), we observe that both state $(QP, i, 0)$ and state $(QP, i, 1)$ will be continued from the same subpath that ends at $(QP^*_{Prev}, i-1, elim^*_{Prev})$. Therefore,

$$(elim^*_{Prev}, QP^*_{Prev}) = \arg\min_{elim_{Prev}} \arg\min_{QP_{Prev}} \{V_{i-1}^{elim_{Prev}}(QP_{Prev}) + \lambda \cdot cost(QP_{Prev}, QP)\} \tag{7.17}$$

The value functions at the current state are then updated according to:

$$\begin{aligned} V_i^0(QP) &= V_{i-1}^{elim^*_{Prev}}(QP^*_{Prev}) + j_i^0(QP) + \lambda \cdot cost(QP^*_{Prev}, QP) \\ V_i^1(QP) &= V_{i-1}^{elim^*_{Prev}}(QP^*_{Prev}) + j_i^1(QP) + \lambda \cdot cost(QP^*_{Prev}, QP) \end{aligned} \tag{7.18}$$

At each stage i, the best paths for all $(QP, i, elim)$ states are found and kept as list of pointers, along with their values. When the algorithm reaches the last stage $(i = N_B)$, the optimal path is the optimal path over the entire frame:

$$(BestQPEnd, BestElimEnd) = \operatorname{argmin}_{QP} \operatorname{argmin}_{elim} V_{N_B}^{elim}(QP) \tag{7.19}$$

The algorithm then traces back the optimal path using the chosen list of pointers to obtain the optimal QP path $\{QP_i^*\}_{i=1}^{N_B}$ and the optimal elimination decisions $\{elim_i^*\}_{i=1}^{N_B}$. Since the elimination is performed only at the transrater (encoder) side, the $\{elim_i^*\}_{i=1}^{N_B}$ side information is passed just to the partial encoder (that performs the requantization and entropy coding). No extra bits need to be transmitted as side information.

7.2.2 Sub optimal elimination algorithm

The algorithm described in section 7.2.1 is defined over a 3D state array. However, according to (7.17), both state $(QP, i, 0)$ and state $(QP, i, 1)$ are continued from the

same state $(QP^*_{Prev}, i-1, elim^*_{Prev})$. Therefore, a sub-optimal algorithm was developed.

At the end of stage i, the 3D state column is collapsed into a 2D state column by keeping the better state among $(QP, i, 0)$ and $(QP, i, 1)$, which is the state associated with the lower value function:

$$\begin{aligned}(QP, i) &= \underset{elim}{\operatorname{argmin}} V_i^{elim}(QP) \\ V_i(QP) &= \min_{elim} V_i^{elim}(QP)\end{aligned} \tag{7.20}$$

Now, at state $(QP, i, elim)$, the question is which is the best combination of $(QP^*_{Prev}, i-1)$ state and $elim^*$ as the current state elimination choice. Again, the best $(elim^*, QP^*_{Prev})$ combination is the one that minimizes the value function at the current state:

$$\begin{aligned}&(elim^*, QP^*_{Prev}) = \\ &\underset{elim}{\operatorname{argmin}} \arg\min_{QP_{Prev}} \{V_{i-1}(QP_{Prev}) + j_i^{elim}(QP) + \lambda \cdot cost(QP_{Prev}, QP)\}\end{aligned} \tag{7.21}$$

From (7.21), we note that both $(QP, i, 0)$ and $(QP, i, 1)$ states choose the same sub-path that ends at the previously collapsed state $(QP^*_{Prev}, i-1)$. As we move on to the next stage, only the better $(QP, i, elim)$ state is saved. Therefore, this minimization can be broken into two independent minimization problems:

$$QP^*_{Prev} = \arg\min_{QP_{Prev}} \{V_{i-1}(QP_{Prev}) + \lambda \cdot cost(QP_{Prev}, QP)\} \tag{7.22}$$

$$elim^* = \underset{elim}{\operatorname{argmin}} \{j_i^{elim}(QP)\} \tag{7.23}$$

Moreover, as the Lagrangian parameter λ is updated for each trellis iteration, during which it remains fixed, the choice of (7.23) at each (QP, i) state can be predetermined before the trellis iteration starts. This way, the dynamic programming algorithm is defined over a 2D state array, as illustrated in Fig. 7.5, rather than a 3D state array.

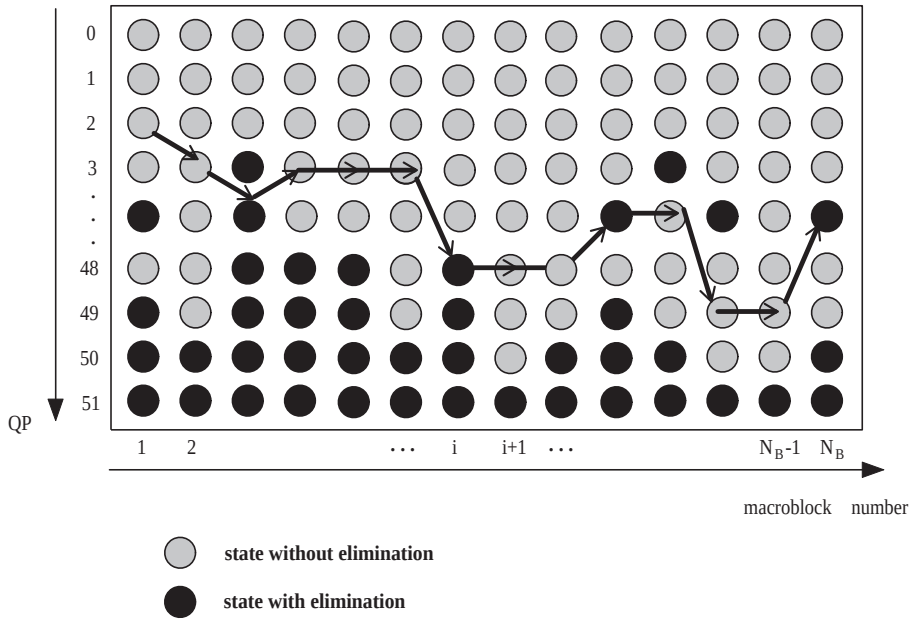

Figure 7.5: Illustration of one dynamic programming iteration using the sub-optimal selective elimination algorithm. Horizontal axis: macroblock number, vertical axis: the quantization parameter QP. At each Lagrangian iteration, λ determines which is the lower cost at each state. The black and gray colors correspond to states for which elimination took or did not take place, respectively.

7.3 Experimental results

We compare here the following proposed requantization algorithms for inter-coded frames:

- Optimal requantization (practical constrained optimization problem).
- Optimal requantization with optimal selective coefficient elimination.
- Optimal requantization with sub-optimal selective coefficient elimination.

The comparison is made in terms of computational complexity and quality. All three *model-based* algorithms evaluate the macroblock level rates $\{r_i(QP)\}$ and the distortions $\{d_i(QP)\}$ using the proposed $rate - \rho$ and $distortion - \rho$ models, as described in Chapter 8. For fair comparison, the three compared algorithms are incorporated into the same transrating scheme, whose intra-coded frames undergo the proposed selective modes modification described in Chapter 6.

Fig. 7.6 depicts the average run-time measured during the transrating of inter-coded frames from different video sequences. As expected, transrating using the optimal selective coefficient elimination has the highest computational complexity among the three algorithms, whereas the transrating that does not perform elimination has the lowest complexity. Incorporating optimal selective elimination increases the run-time of transrating an inter-coded frame by 24%, on average, as compared to no elimination. The sub-optimal selective elimination increases the run-time by 14%, on average. The run-time increase is due to two reasons. One is that more rates and distortions should be evaluated (both $\{d_i^0(QP), r_i^0(QP)\}$ and $\{d_i^1(QP), r_i^1(QP)\}$ rather than just $\{d_i^0(QP), r_i^0(QP)\}$), as explained in section 7.2.1. The other is that the optimization procedure is more complicated with the extension of the selective elimination.

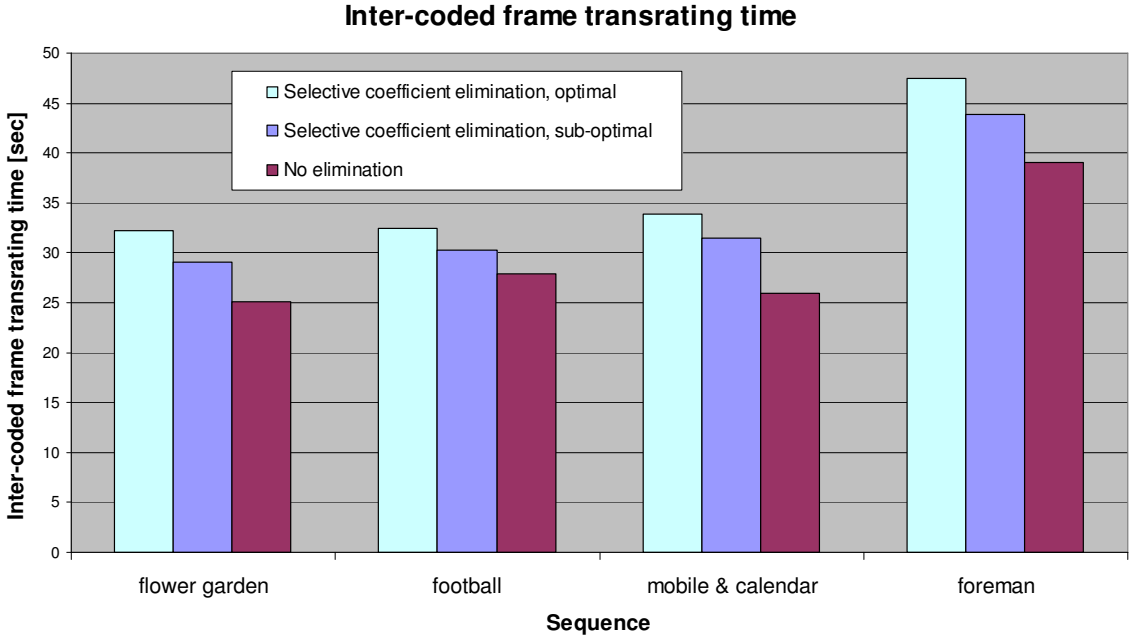

Figure 7.6: Run-time comparison for different optimal requantization inter transrating algorithms: optimal selective coefficient elimination, sub-optimal selective coefficient elimination, and no elimination.

The proposed selective elimination follows the simple elimination rule used in the recommended reference software. As the bit rate decreases, the probability of appearance of a sparse block increases and therefore potentially more blocks can be eliminated after requantization. However, as depicted in Fig. 7.7, only a small part of the frame blocks are actually eliminated after requantization as a result of applying the algorithm. Therefore, the PSNR gain of the selective elimination is easier to discern on a GOP-level. Fig. 7.8 depicts an example for the GOP-level PSNR vs. bit rate, where the PSNR gain achieved by the selective elimination is up to 0.07[dB] at the low bit rates as marked in red for each transrated bit rate. The overall sequence PSNR is practically the same as that of a sequence transrated without the elimination, due to averaging.

Even though, we believe that this algorithm can potentially achieve a higher gain. A possible future direction is to examine other elimination rules rather than the one used in the recommended reference software.

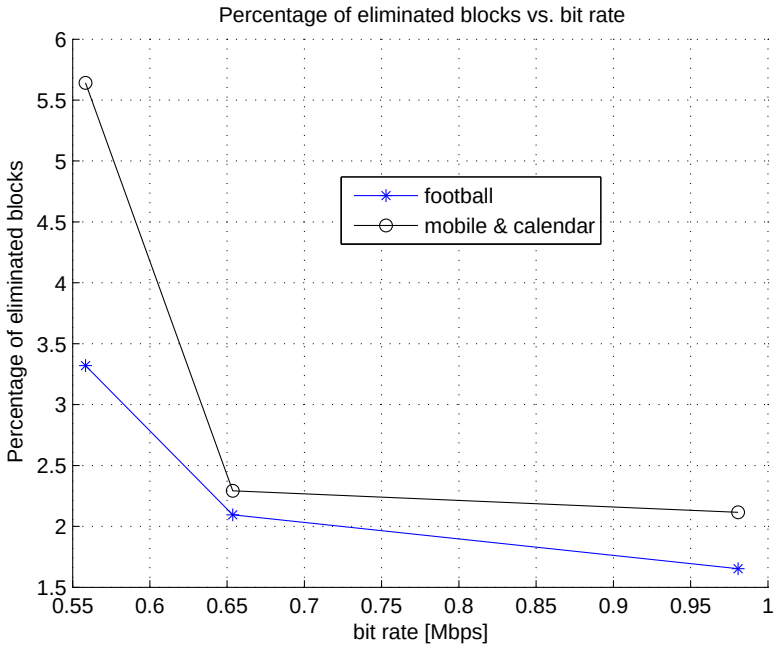

Figure 7.7: Percentage of eliminated blocks vs. bit rate. Blue asterisk: football sequence, black circles: mobile & calendar sequence.

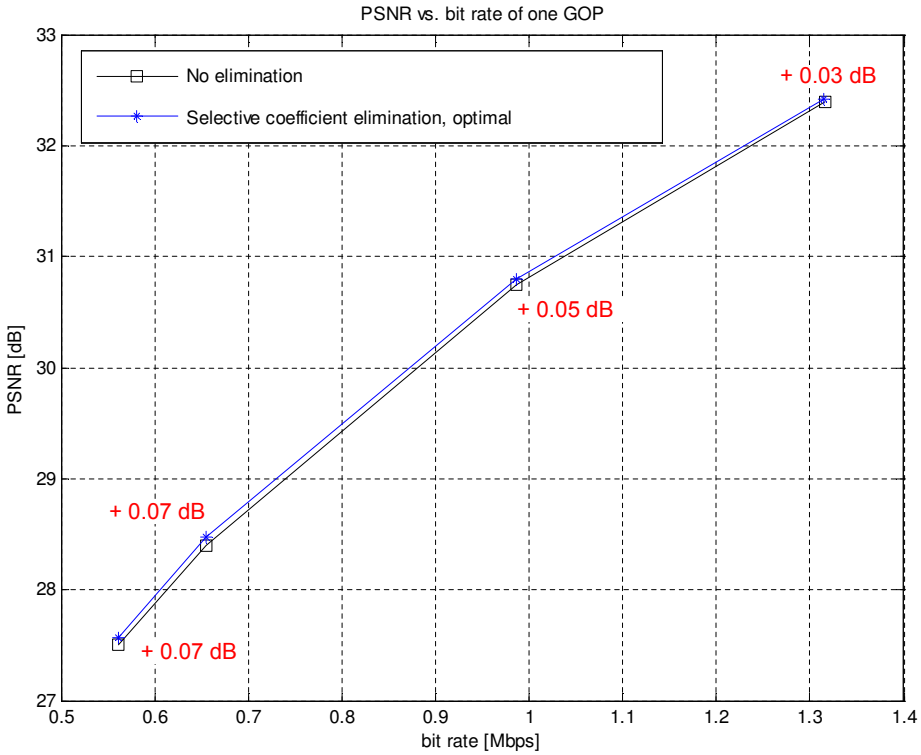

Figure 7.8: PSNR vs. bit rate comparison with and without coefficient elimination (for one GOP). Black squares: without elimination, blue asterisk: selective coefficient elimination. The measured PSNR gain at each bit rate is denoted in red.

Chapter 8

Inter Frames Transrating - Rate-Distortion Modeling

The optimization algorithm described in Chapter 7 requires the evaluation of the rate and distortion obtained by requantizing each macroblock at multiple step-sizes. If no prior knowledge is used, such rate assessment involves the simulation of the actual requantization followed by entropy coding. As this procedure must be repeated multiple times, the optimization becomes computationally expensive. The computational complexity can be greatly reduced by using an analytic model for the relation between rate and quantization step-size, *for each macroblock*. In this chapter, we will elaborate on the model-based evaluation of the rate and the distortion.

In order to incorporate the ρ domain models (see section 2.2.2) into the optimization described in Chapter 7, we suggest modified models for H.264 at the *macroblock level*. Section 8.1 describes the $rate - \rho$ model, section 8.2 the $distortion - \rho$ model and section 8.3 the evaluation of $\rho - Q_2$ relation. The last section analyzes the models performance.

8.1 MB-level $rate - \rho$ model for H.264 requantization

Examination of the $rate - \rho$ relation at the macroblock level has shown that a linear relation is not a good descriptor of the empirical data. Therefore, and in light of the new entropy coding features of H.264, we suggest a different $rate - \rho$ model at the macroblock level. We decompose the rate into "*data*" and "*overhead*" components, where the "*data*" stands for the bits spent on coding the run-level, and the "*overhead*" designates the bits spent on coding the new syntax elements (see Chapter 3 and appendix A). The total $rate - \rho$ relation is evaluated by:

$$r(\rho) = r^{data}(\rho) + r^{overhead}(\rho) \tag{8.1}$$

where $r^{data}(\rho)$ and $r^{overhead}(\rho)$ are evaluated from (8.2) and (8.6) in the sequel, respectively. For the model parameters estimation we use prior information, such as the original input quantized transform coefficients and their encoded rate.

8.1.1 "Data" Component

The "data" texture bits component is composed of coding the run-level syntax elements that form the majority of the texture bits at moderate to high bitrates. This component $rate - \rho$ relation is a monotonically decreasing convex function.

Therefore, for the "data" component $rate - \rho$ relation, we suggest the following closed-form model:

$$r^{data}(\rho) = \theta \cdot ln(1 + (1 - \rho)^{\eta}) \tag{8.2}$$

where $\theta \geq 0, \eta \geq 1$. The parameter θ controls the scale of the graph, whereas the parameter η changes its shape. Now, given this component's original input encoded rate of a macroblock, $r_{in}^{data}(\rho_{in})$, we can fit one of the parameters. Since this model requires fitting two parameters, we apply a two-dimensional search to fit its shape parameter η and an average scale parameter $\bar{\theta}$ using the input ensemble $\{r_{in,i}^{data}(\rho_{in,i})\}_{i=1}^{N_B}$ of all

the frame macroblocks. The estimated shape parameter η is used for all the frame macroblocks. The scale parameter θ_i is then matched to each macroblock separately by:

$$\theta_i = \frac{r_{in,i}^{data}}{ln(1+(1-\rho_{in,i})^{\eta})} \tag{8.3}$$

The luminance and the chrominance components are modeled separately.

Since the frame macroblocks share the same parameter η, but each has a different parameter θ_i, we cannot depict their model-based fits on one graph. However, we can scale all macroblock level relations using the average frame level parameter $\bar{\theta}$, by drawing $r_i^{data}(\rho_i) \cdot \frac{\bar{\theta}}{\theta_i}$ and then draw their common fit $r^{data}(\rho_i) = \bar{\theta} \cdot ln(1+(1-\rho_i)^{\eta})$. Fig. 8.1 depicts for each macroblock its scaled $rate - \rho$ relation by blue dots and the common fit by a red line.

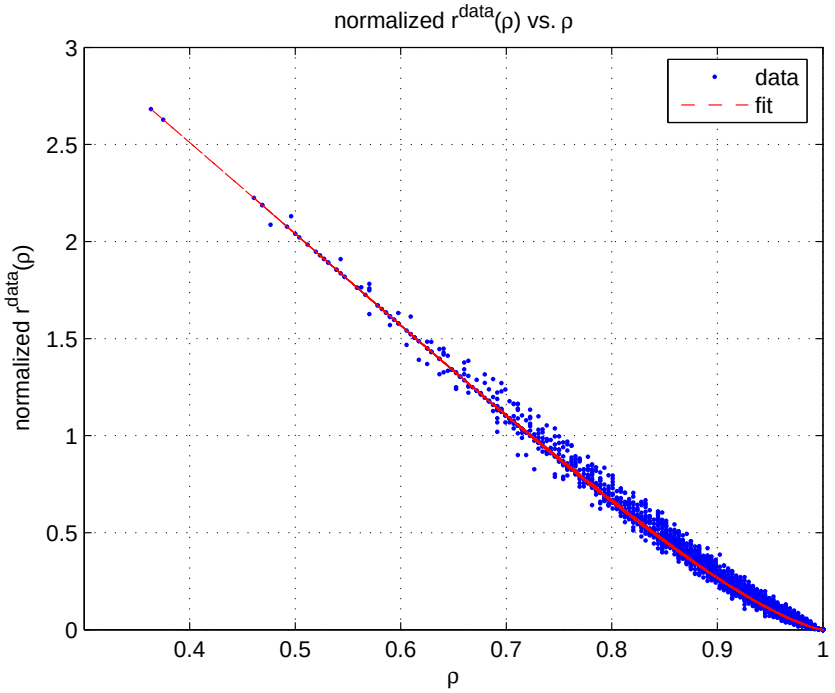

Figure 8.1: Blue dots: normalized $r^{data}(\rho)$ relation of one frame's macroblocks; red line: the fit with the common shape parameter η. Here, $\eta = 1.36$ and $\bar{\theta} = 6.2$.

A typical distribution of the parameters θ and η for luminance and chrominance components is depicted in Fig. 8.2.

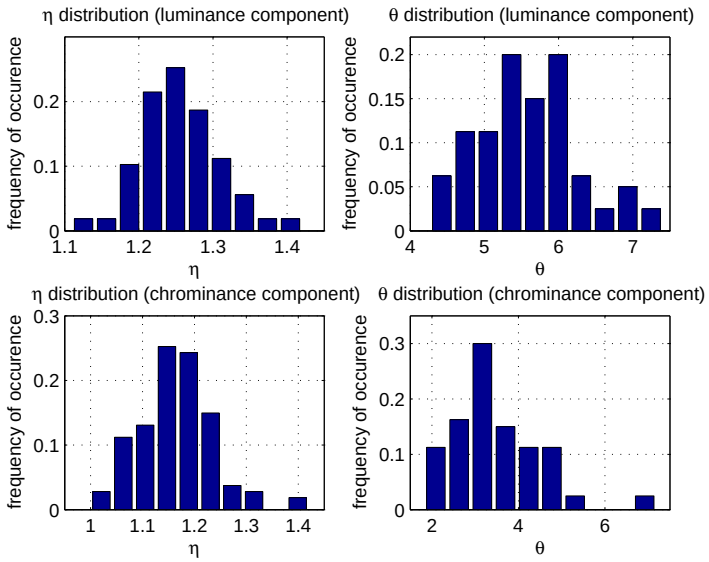

Figure 8.2: Distribution of $rate - \rho$ model's parameters. Top: luminance component, bottom: chrominance component. Left: shape parameter η distribution, right: scale parameter θ distribution.

8.1.2 "Overhead" Component

The "overhead" texture bits component describes two additional syntax elements:

- (TotalCoefficients, TrailingOnes) - the combination of the number of non-zero coefficients and the high-frequency trailing-ones (± 1 at the end of the block).
- TotalZeros - the number of zero coefficients from the DC coefficient to the highest frequency non-zero coefficient.

Fig. 8.3 shows an example for a 4x4 zig-zag scanned block, with 6 non-zero coefficients, 2 trailing-ones, and 2 TotalZeros (that are marked in gray).

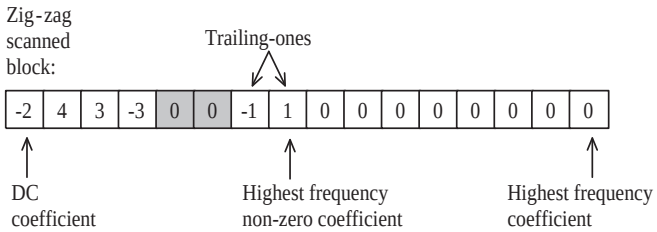

Figure 8.3: An example of the additional overhead syntax elements in H.264.

The "overhead" component $rate - \rho$ relation is very noisy due to two reasons. One is that the overhead syntax elements values are not uniquely defined by the local block's ρ. For example, in Fig. 8.3 the 6 non-zero coefficients correspond to (TotalCoefficients, TrailingOnes)=(6,2) and TotalZeros=2, but for other blocks these 6 non-zero coefficients can be spread differently throughout the scanned block and correspond to many other combinations of these syntax elements. The other is the use of multiple VLC tables for each syntax element, which means that the number of bits spent on coding the same syntax element value changes with the context. As a result, fitting a closed-form model for it becomes practically impossible. However, due to the partial dependency in the local ρ, we chose to use a statistical model to characterize the average code length at the 4x4 block level, and then average over the 16 blocks in the macroblock.

Each 4x4 block has a local percentage of zeroed coefficients, ρ_b, which is related to the local total non-zero coefficients count TC_b, by

$$\rho_b = 1 - \frac{TC_b}{16} \tag{8.4}$$

The macroblock-level ρ is simply the average of these local ρ_b's:

$$\rho = \frac{1}{16}\sum_{b=1}^{16}\rho_b \tag{8.5}$$

Using the statistical model that follows, we calculate once the average code lengths $\overline{c}_{(TC,Tr)}(\rho_b|context - prior)$ and $\overline{c}_{TZ}(\rho_b|input - prior)$ of the
(TotalCoefficients, TrailingOnes) and TotalZeros syntax elements, respectively. These average lengths are kept in look-up tables and the rate "overhead" component is obtained by averaging over all the blocks in the macroblock:

$$\begin{aligned} r^{overhead}(\rho) = \frac{1}{16}\sum_{b=1}^{16}\overline{c}_{(TC,Tr)}(\rho_b|context - prior) \\ + \frac{1}{16}\sum_{b=1}^{16}\overline{c}_{TZ}(\rho_b|input - prior) \end{aligned} \tag{8.6}$$

Statistical model

We assume that the quantized transform coefficients are not correlated and follow a Laplacian distribution [15]. Another assumption is that all ± 1 quantized coefficients appearances occur at the highest nonzero frequencies, and are thus considered as high-frequency trailing-ones. Using the Laplacian distribution, the probability that the magnitude of a quantized transform coefficient will take the value k is:

$$Pr.(|l| = k) = \begin{cases} \rho & k = 0 \\ \frac{(1-\rho)^{2k}\rho(2-\rho)}{1-\rho} & k > 0 \end{cases} \tag{8.7}$$

and therefore the probability of a trailing-one coefficient, given that it is non-zero is:

$$Pr.(TR) = Pr.(|l| = 1 | l \neq 0) = \frac{Pr.(|l| = 1)}{1 - Pr.(l = 0)} = \rho(2 - \rho) \tag{8.8}$$

(TotalCoefficients, TrailingOnes) average code table

We define a binomial random variable that denotes the number of trailing-ones appearances given ρ_b and sum over the joint
(TotalCoefficients, TrailingOnes) code length tables (there are 4 different tables) to obtain the average VLC tables $\overline{c}_{(TC,Tr)}(\rho_b | context - prior)$. We switch between these four average VLC tables by predicting the number of non-zero coefficients from the neighboring blocks, in accordance with the standard's context-based encoding, see appendix A.

TotalZeros average code table

Since the quantized blocks are typically sparse and most of the energy is concentrated at low frequencies, there is usually a tail of zeros at the end of the scanned block (see example in Fig. 8.4). So, instead of counting the TotalZeros syntax element, TZ, as the number of zeroed coefficients from the DC coefficient to the highest frequency non-zero coefficient, we can count its complement, the tail, since $TC + TZ + Ztail = 16$. As we increase the requantization step, the number of non-zero

coefficients, TC, decreases, and the tail length monotonically increases. Therefore, $TC + TZ$ monotonically decreases.

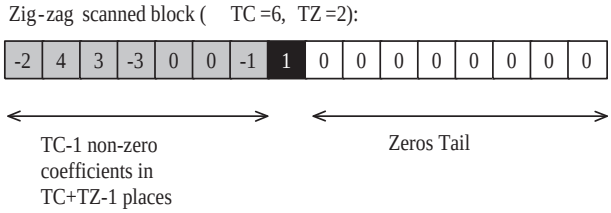

Figure 8.4: The example of Fig. 8.3 with TC, TZ and the zeros tail. There are TC=6 non-zero coefficients and TZ=2 zeros counted from the DC coefficient to the highest frequency non-zero coefficient (which is denoted in black).

We find the probability of having TZ TotalZeros given ρ_b using the statistical model and the input prior information (TC_{in}, TZ_{in}). The input prior defines the possible (TC, TZ) pairs that can be obtained as a result of the coarser requantization, as depicted in Fig. 8.5. The average code length for each of the 15 (TC_{in}, TZ_{in}) input priors, $\bar{c}_{TZ}(\rho_b|input - prior)$, is evaluated by summing over the joint (TotalCoefficients,TotalZeros) code length tables.

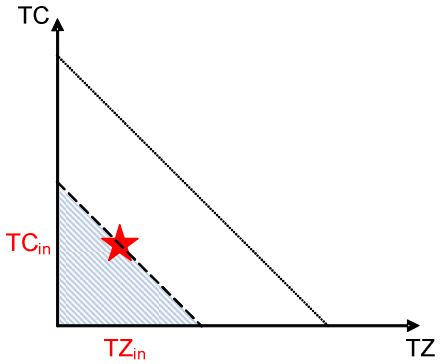

Figure 8.5: The TC-TZ plane. The initial (TC_{in}, TZ_{in}) point is marked with a red star. The (TC, TZ) pair obtained by coarser requantization will be in the marked triangle.

8.2 MB-level distortion$-\rho$ model

The PSNR is a widely used objective quality metric that is related to the MSE distortion. That is why we examined the validity of the exponential $distortion-\rho$ model suggested in [15] in describing the MSE. According to this model, $ln(\overline{d}(\rho))$ should be linearly proportional to $1-\rho$, where $\overline{d}(\rho)=\frac{d(\rho)}{\sigma^2}$ is the normalized distortion. Examining this relation at the macroblock level, we found that a linear model does not describe it with sufficient accuracy. We therefore suggest to extend the model to an exponential-quadratic relation:

$$d(\rho)=\sigma^2\cdot e^{\alpha_1\cdot(1-\rho)^2+\alpha_2\cdot(1-\rho)} \tag{8.9}$$

that better matches the empirical data, see Fig. 8.6.

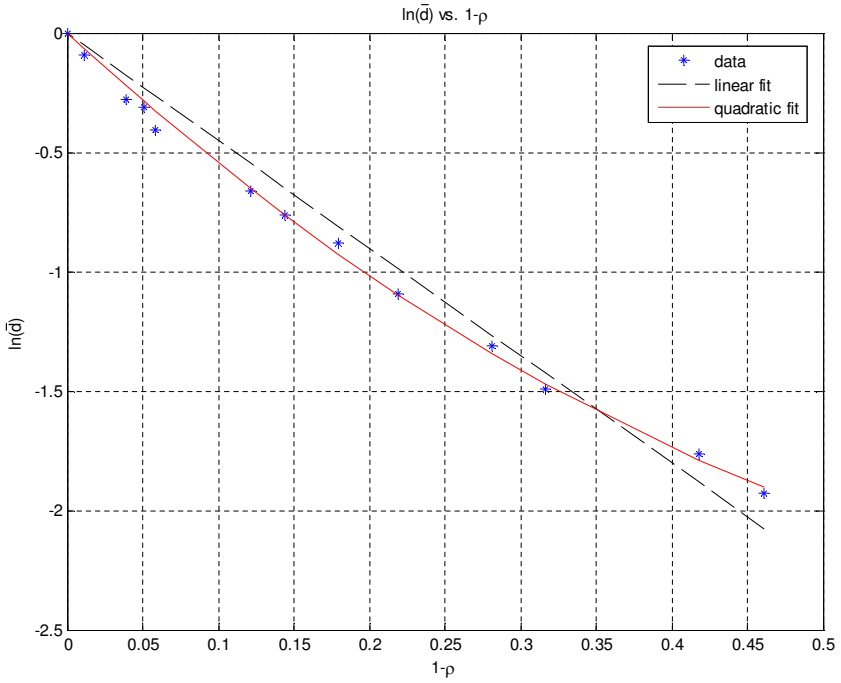

Figure 8.6: Distortion$-\rho$ model. Blue points: $ln(\overline{d}(\rho))$; black dashed line: linear fit; red solid line: quadratic fit.

The modified $disortion-\rho$ model has three parameters that should be estimated. Since we can only measure the requantization distortion, and not the total degradation from the reference, we do not have any prior information from the first encoder.

The scale parameter σ^2 is calculated once as the sum of squares of the input transform coefficients, as this would be the MSE if the block is zeroed. Given the scale parameter, we evaluate the normalized distortion $\overline{d}(\rho)$, that has two parameters to be estimated: α_1, α_2. To this end, we should get two different (ρ, d) points. The suggestion is to first evaluate the $\rho - Q_2$ relation for each macroblock, see section 8.3. Then estimate the distortion at the finest requantization step size (that is bigger than the original step) that corresponds to a fraction ρ_1 of zeroed coefficients. Based on ρ_1, we would like to find ρ_2, such that $1 - \rho_2 \simeq \frac{1}{2} \cdot (1 - \rho_1)$. Since we can only find ρ_2 at the resolution of the available quantization step sizes, we choose the closest available ρ_2 (using the $\rho - Q_2$ table we already have at hand). Based on these two points, $(1-\rho_1, ln(\overline{d_1}))$ and $(1-\rho_2, ln(\overline{d_2}))$, we can estimate the quadratic fit for $ln(\overline{d})$ vs. $1 - \rho$ curve (see illustration in Fig. 8.7) and extract the α_1, α_2 parameters. The luminance and chrominance components are modeled separately.

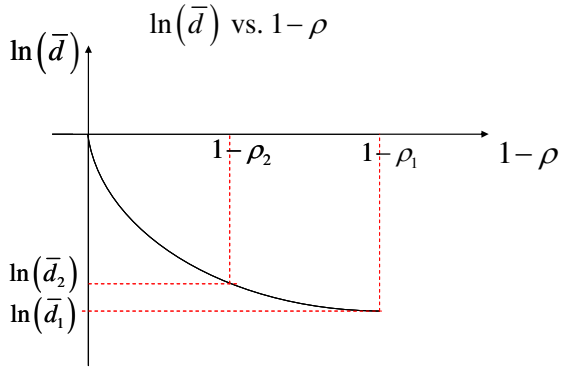

Figure 8.7: Parameters estimation for the $distortion - \rho$ model.

A typical distribution of the parameters σ^2, α_1 and α_2 for luminance and chrominance components is depicted in Fig. 8.8.

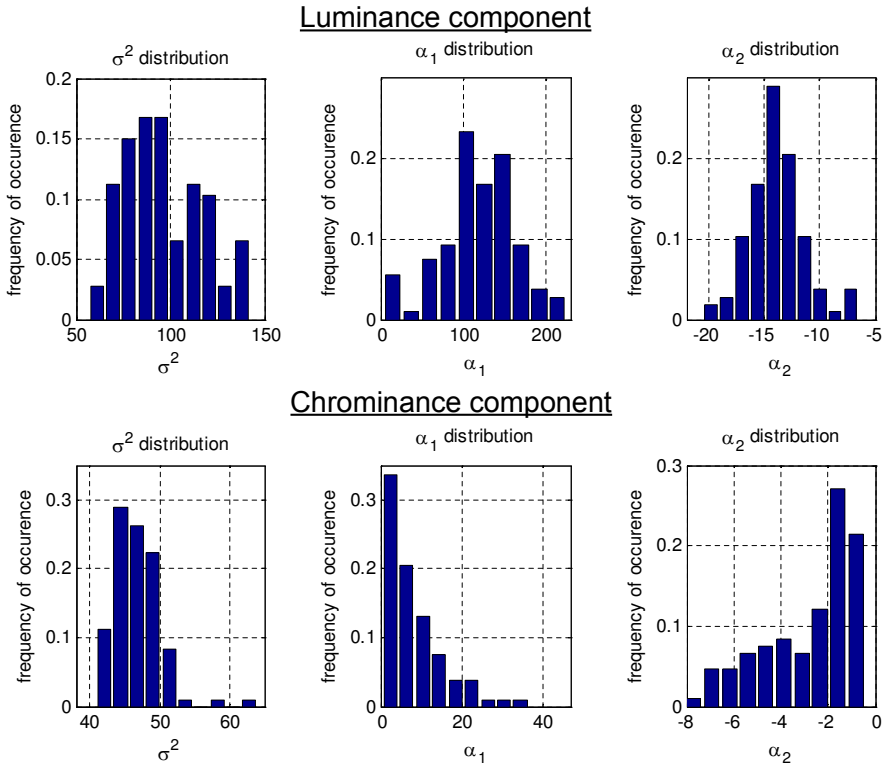

Figure 8.8: Distribution of *distortion* $-\ \rho$ model's parameters. Top: luminance component, bottom: chrominance component. Left: scale parameter σ^2 distribution, middle: shape parameter α_1 distribution, right: shape parameter α_2 distribution.

8.3 $\rho - Q_2$ relation

Contrary to intra-coded frames, the estimation of ρ for inter-coded frames is fairly simple and has a low computational complexity. Since the inter-coded blocks are predicted using previously decoded frames, their closed loop correction signal is available and the models evaluation is performed based on the corrected transform coefficients to be requantized, see Fig. 4.1.

Therefore, we count the number of coefficients that fall in the second quantizer deadzone, $[-Th(Q_2), Th(Q_2)]$, where $Th(Q_2) = (1 - dz)Q_2$. The $\rho - Q_2$ relation is evaluated using this histogram count by normalizing the expected number of zeros at the quantizer output to the data size (either 256 coefficients or 128 coefficients for the luminance and chrominance MB components, respectively). It is evaluated for

each macroblock for all the step sizes that are coarser than the input step size, prior to the rate and the distortion evaluation.

In case the selective elimination algorithm is applied, ρ is evaluated by applying the same histogram count on the quantized coefficients after elimination.

8.4 Performance analysis of proposed models

The motivation of using the rate-distortion models is to provide a low computational accurate evaluation. In this section, we will evaluate the performance of the proposed MB-level models in terms of accuracy and computational complexity.

8.4.1 Rate models accuracy

The proposed $rate - \rho$ model is incorporated as part of the optimal requantization. Therefore, its accuracy should be evaluated using two metrics. The first metric measures the deviation of the model-based rate estimation from the actual encoded number of bits. The second metric measures the deviation of the actual encoded number of bits from the target rate for that frame.

To conduct this evaluation, we transrated video sequences at different working points (transrated bit rates). For each transrated inter-coded frame, two relative errors were calculated: model-based optimization as compared to the actual frame bit rate, and actual frame bit rate as compared to the target rate.

To put these accuracy measurements into proportion, we repeated this procedure where the linear $rate - \rho$ model (2.16) is applied at the macroblock level, for the luminance and chrominance components, separately.

Fig. 8.9 depicts the mean relative model errors. The linear $rate - \rho$ model errors (shown by dotted textured bars) range from -13.3% to -22.6%, on average. The proposed $rate - \rho$ model errors (shown by solid color bars) range from -5.8% to -1.8%, on average.

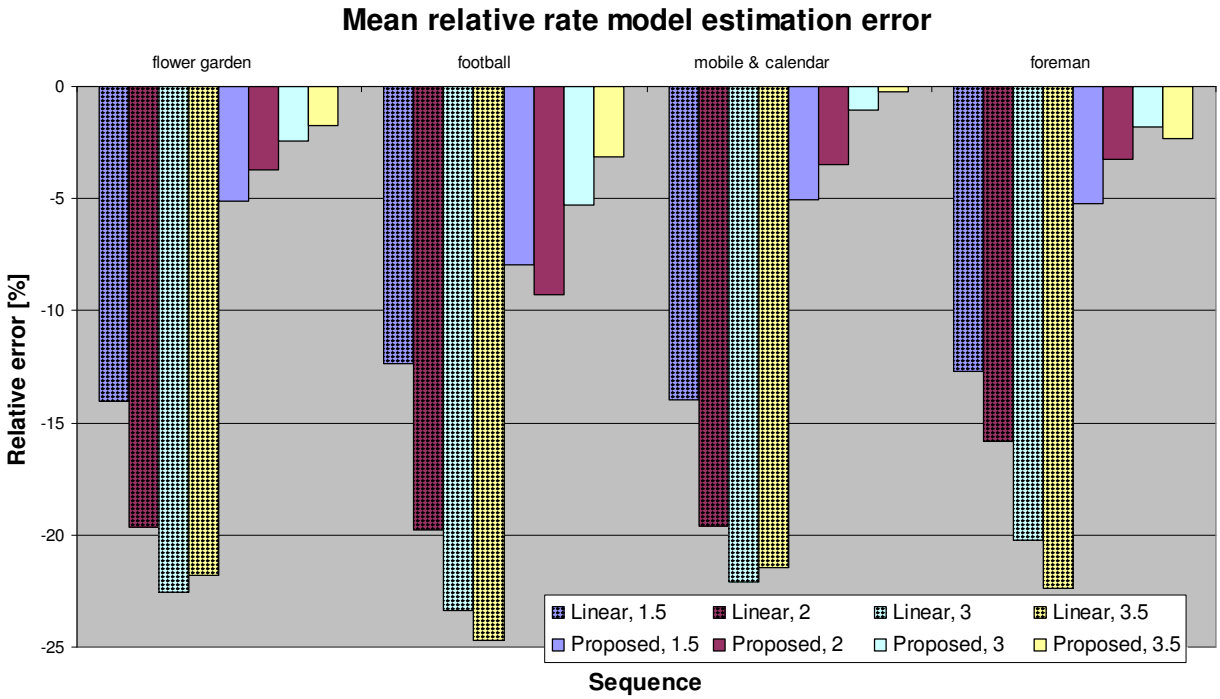

Figure 8.9: Mean relative rate estimation error at different transrating ratios. Horizontal axis: sequence, vertical axis: relative error [%]. From left to right: blue: transrating ratio of 1.5, purple: transrating ratio of 2, cyan: transrating ratio of 3, and yellow: transrating ratio of 3.5. The dotted textured bars correspond to the linear $rate-\rho$ model whereas the solid color bars correspond to the proposed $rate-\rho$ model.

Fig. 8.10 depicts the mean deviation from the target bit rate, using the model-based optimization, where the optimization stopping criterion is a tolerance of 4% deviation from the target. The linear $rate-\rho$ model deviations (shown by dotted textured bars) range from 15.6% to 30.2%, on average. The proposed $rate-\rho$ model deviations shown by solid color bars) range from 2.9% to 6.3%, on average.

Modeling the $rate-\rho$ relation for H.264 requantization is a challenging task. The proposed $rate-\rho$ model outperforms the linear model suggested in the literature and its incorporation in the optimal requantization algorithm allows better convergence to the target rate.

As pointed in section 4.2.2, at the end of each frame encoding, the deficit/overplus of bits is uniformly distributed among the remaining frames in the GOP, such that the average GOP rate will be closer to its target.

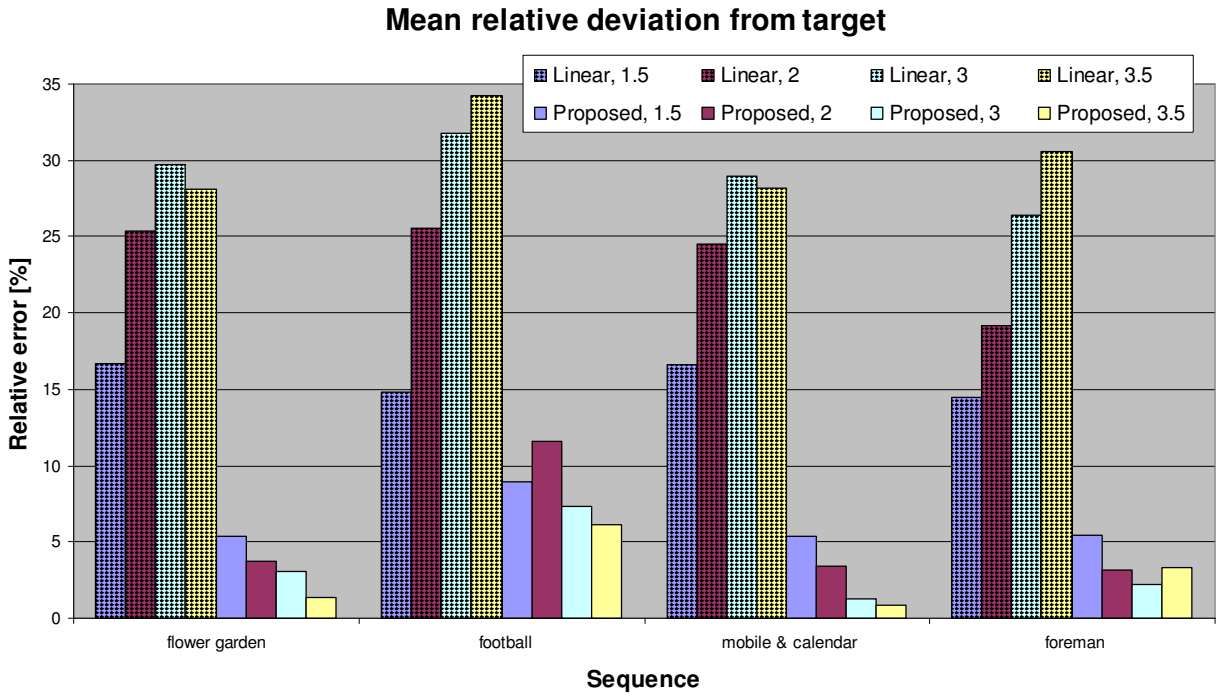

Figure 8.10: Mean relative deviation from the target bit rate at different transrating ratios. Horizontal axis: sequence, vertical axis: relative error [%]. From left to right: blue: transrating ratio of 1.5, purple: transrating ratio of 2, cyan: transrating ratio of 3, and yellow: transrating ratio of 3.5. The dotted textured bars correspond to the linear $rate - \rho$ model whereas the solid color bars correspond to the proposed $rate - \rho$ model.

8.4.2 Distortion models accuracy

We repeated a test similar to that described in subsection 8.4.1 in order to evaluate the distortion model performance, see Fig. 8.11. The mean relative error does not exceed 4%.

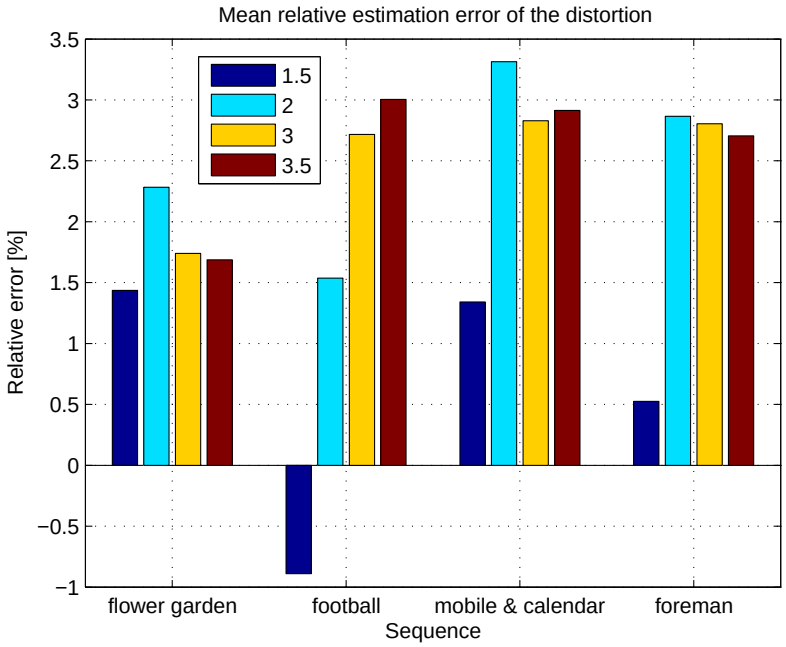

Figure 8.11: Mean relative distortion estimation error at different transrating ratios. Horizontal axis: sequence, vertical axis: relative error [%]. Blue: transrating ratio of 1.5, cyan: transrating ratio of 2, yellow: transrating ratio of 3, and red: transrating ratio of 3.5.

8.4.3 Computational complexity

We now discuss the computational complexity savings by comparing the evaluation time of the rates and distortions using the proposed models with the full exhaustive evaluation time (without the models aid). The evaluation time was measured using our Matlab simulation. Fig. 8.12 depicts the average rate-distortion evaluation time per one MB. The proposed model-based approach reduces the run-time by a factor of about 3.5, on average. Fig. 8.13 depicts the average transrating time of an inter-coded frame, where an optimal requantization is performed (see section 7.1), with and without models. Here, the run-time reduction factor is about 2.3, on average.

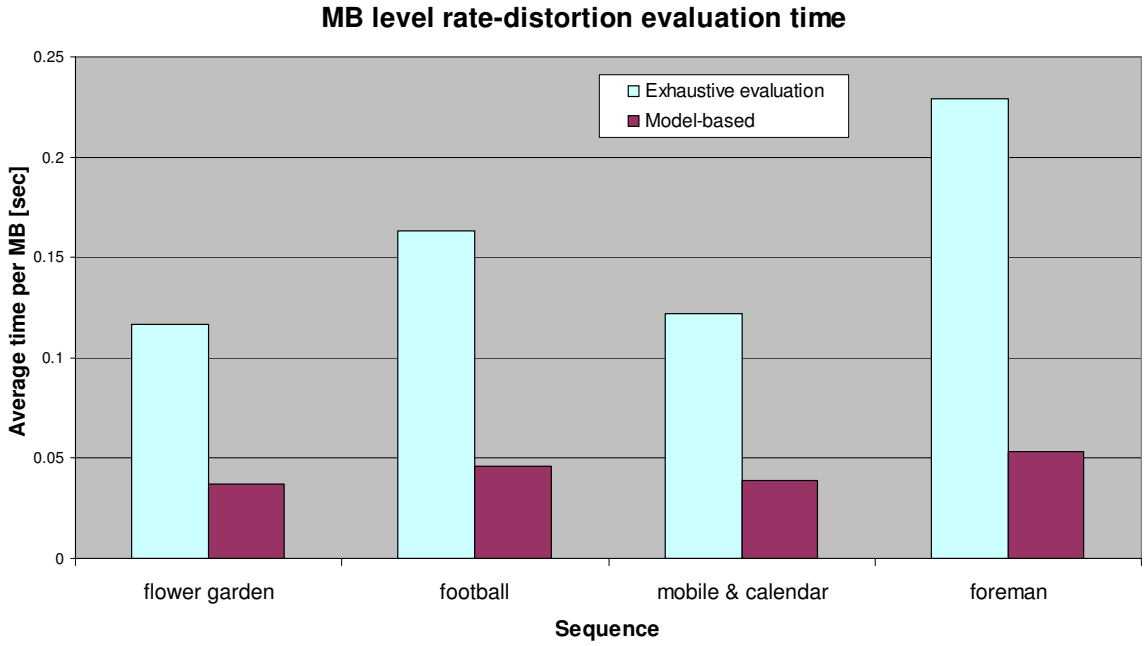

Figure 8.12: Rate-distortion evaluation time per MB (in seconds, as measured by Matlab). Horizontal axis: sequence, vertical axis: time. Cyan: Exhaustive evaluation (no models), purple: model-based evaluation.

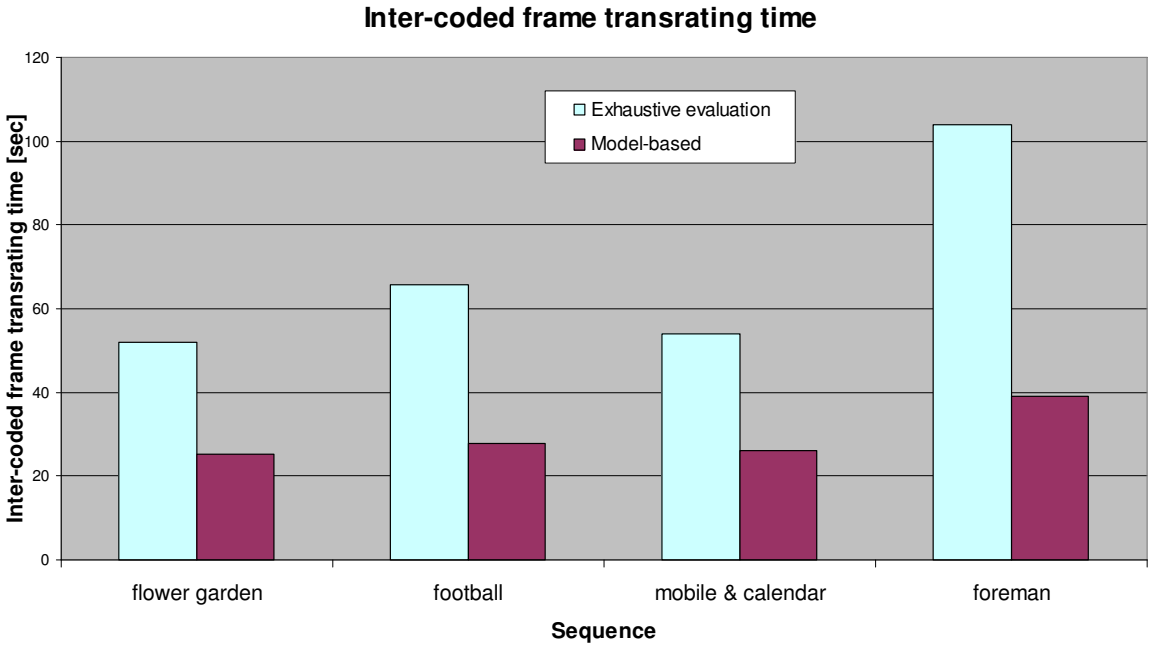

Figure 8.13: Inter-coded frame transrating time (in seconds, as measured by Matlab). Horizontal axis: sequence, vertical axis: time. Cyan: Exhaustive evaluation (no models), purple: model-based evaluation.

Chapter 9

Simulation Results

In this chapter we summarize and compare by simulations the following transrating algorithms:

- Re-encoding.
- Developed algorithm, reuse of intra modes, MB-level r-d models.
- Developed algorithm, selective intra modes modification, MB-level r-d models.
- Developed algorithm, selective intra modes modification, exhaustive r-d evaluation (without the MB-level rate-distortion models).
- One-pass requantization (see section 4.3).

The comparison is made in terms of computational complexity and quality. The *developed algorithm* performs optimal requantization, as suggested in Chapter 5 for intra-coded frames and in section 7.1 for inter-coded frames. All schemes but the re-encoding follow the optimal GOP level bit allocation as discussed in section 4.2. The original video sequences were first encoded at 2[Mbps] using H.264 baseline profile and then transrated at four transrating ratios.

The standard video sequences used for the analysis are described in Table 9.1. The first frame of each sequence is depicted in Fig. 9.1.

Table 9.1: Description of the examined video sequences.

Sequence	Format	Description
Flower garden	SIF(352x240) 115 frames	Panning, scene depth
Football	SIF(352x240) 209 frames	Panning, fast motion
Mobile & calendar	SIF(352x240) 299 frames	Panning, synthetic scene, medium motion
Foreman	CIF(352x288) 299 frames	Moving face, slow motion

Figure 9.1: First frame from each of the examined sequences. Upper left: flower garden, upper right: football, bottom left: mobile & calendar, bottom right: foreman.

Fig. 9.2 depicts the average transrating run-time for the different transrating algorithms. As expected, re-encoding has the highest computational complexity, whereas the one-pass requantization has the lowest computational complexity. The proposed model-based transrating system (developed algorithm, MB-level r-d models) can either reuse the intra prediction modes or selectively modify them. The average run-time increase, as compared to the simpler proposed system (developed algorithm, reuse of intra modes, MB-level r-d models) is given in Table 9.2.

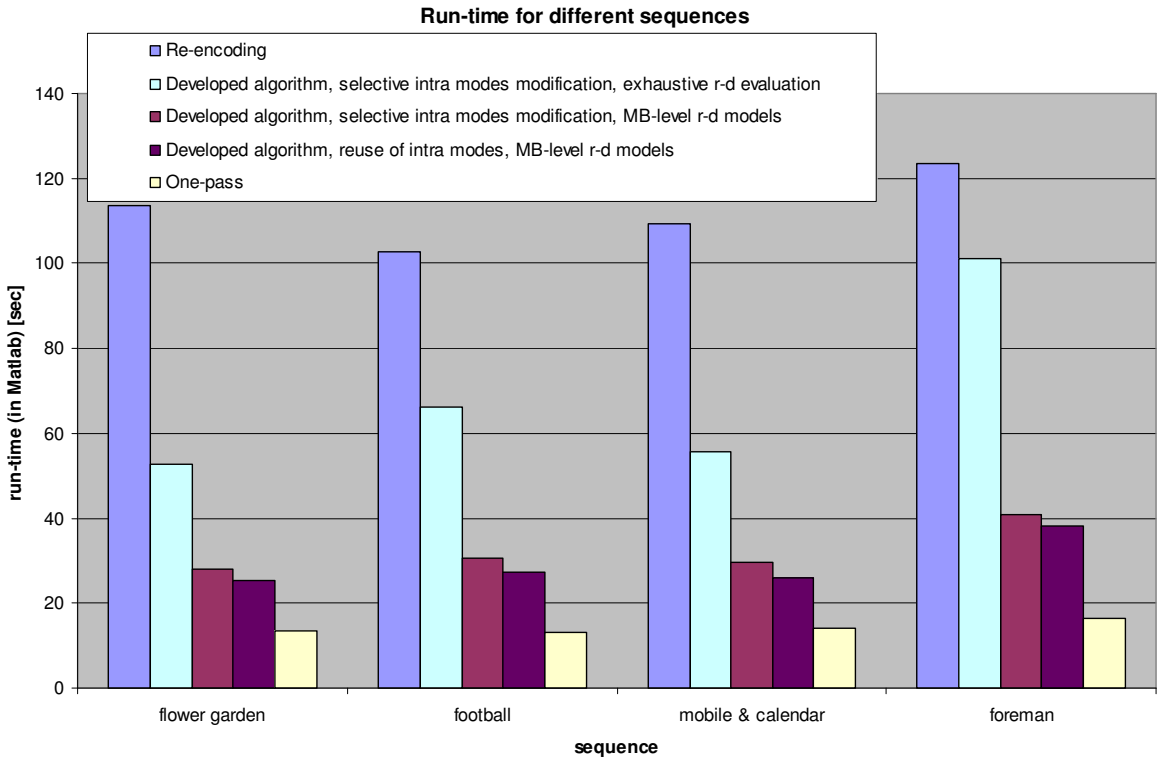

Figure 9.2: Run-time comparison for different transrating algorithms: re-encoding; developed algorithm with selective intra modes modification and exhaustive r-d evaluation; developed algorithm with selective intra modes modification and MB-level r-d models; developed algorithm with reuse of intra modes and MB-level r-d models; and one-pass requantization.

Table 9.2: Overall transrating methods run-time comparison.

Transrating method	Run-time increase [%], as compared to developed algorithm, reuse of intra modes, MB-level r-d models
Re-encoding	+290 %
Developed algorithm, selective intra modes modification, exhaustive r-d evaluation	+130 %
Developed algorithm, selective intra modes modification, MB-level r-d models	+ 10 %
Developed algorithm, reuse of intra modes, MB-level r-d models	0 %
One-pass	- 50 %

The quality comparison was performed using two measures: PSNR and VQM. The PSNR vs. bit rate graphs are depicted in Fig. 9.3 to 9.6. As expected, it rates the performance of the transrating algorithms at the following order, from the best to the worst quality: re-encoding; developed algorithm with selective intra modes modification and exhaustive r-d evaluation; developed algorithm with selective intra modes modification and MB-level r-d models; developed algorithm with reuse of intra modes and MB-level r-d models and one-pass requantization. The proposed transrating scheme outperforms the one-pass requantization, at up to 1.6[dB] gain in PSNR. It should be noted that for fair comparison, the model-based optimal GOP level bit allocation was applied for the one-pass method too, but it is more likely that such a simple requantization would use a simpler GOP allocation as well, which is expected to further decrease its performance. At high bit rates, the proposed scheme can achieve the re-encoding bound (see Fig. 9.3 and Fig. 9.5) and the PSNR gap between them increases as the bit rate is reduced, up to 1.4[dB], since the proposed schemes reuse the input motion vectors.

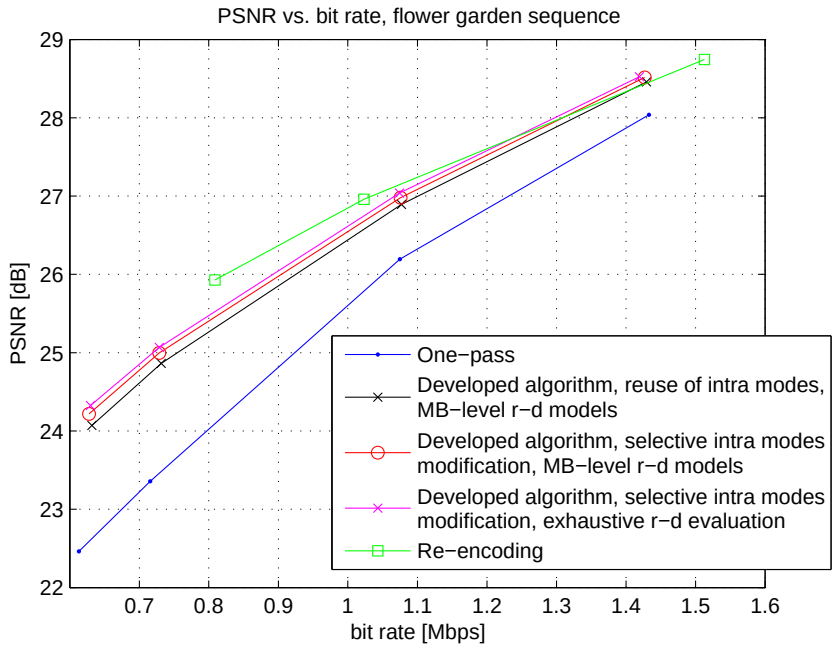

Figure 9.3: PSNR vs. bit rate, for the flower garden sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

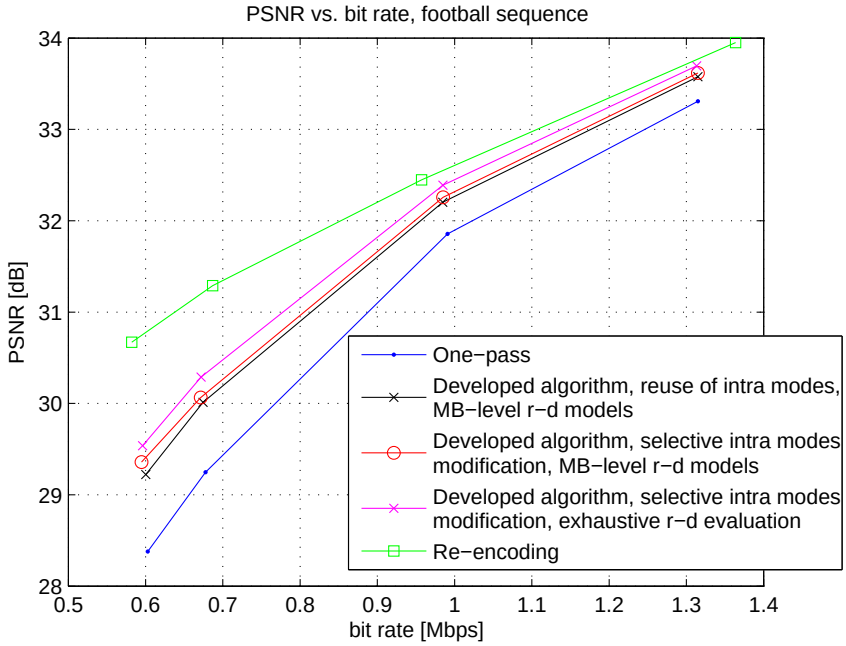

Figure 9.4: PSNR vs. bit rate, for the football sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

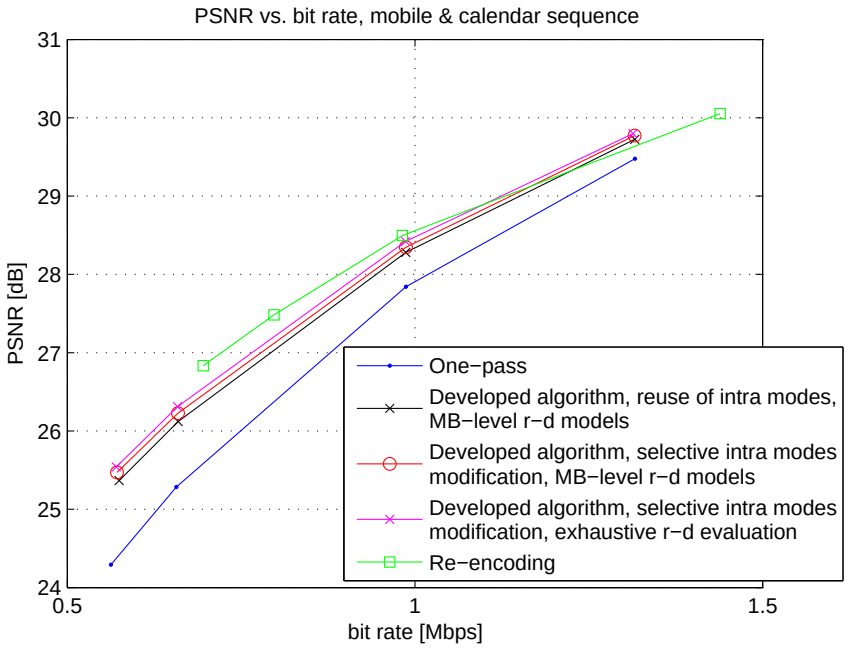

Figure 9.5: PSNR vs. bit rate, for the mobile & calendar sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

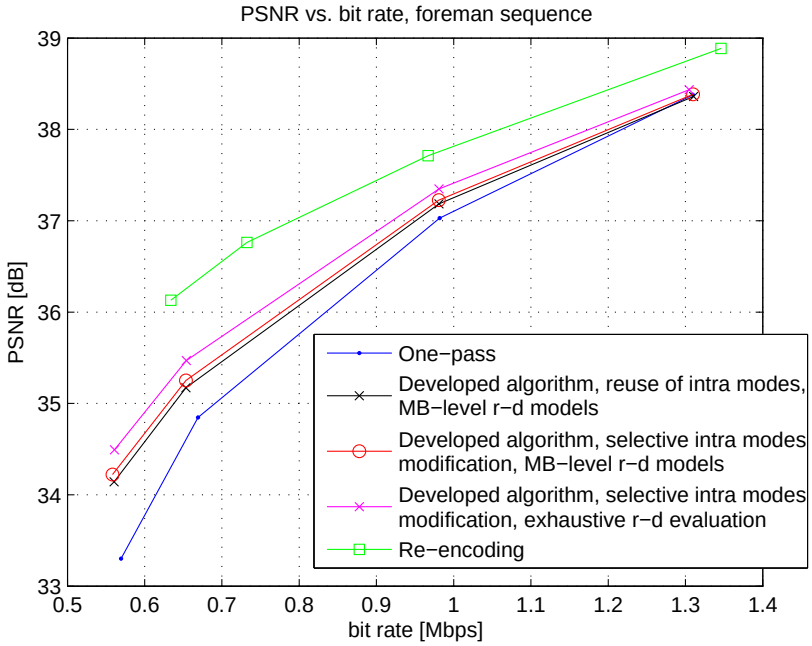

Figure 9.6: PSNR vs. bit rate, for the foreman sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

The VQM score attempts to measure the subjective quality degradation (see section 2.4) at a scale ranging from 0% (no perceived impairment) to 100% (maximum perceived impairment). Fig. 9.7 to 9.9 depict its value vs. the bit rate. It has the same general trend as the PSNR vs. bit rate graphs, and it consistently shows that selective intra modes modification is subjectively better than their reuse.

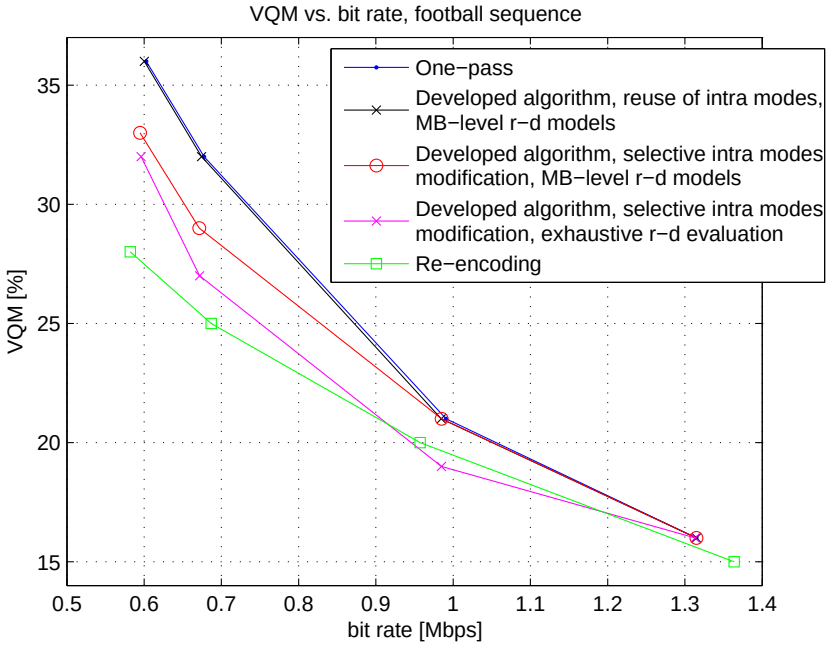

Figure 9.7: VQM vs. bit rate, for the football sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

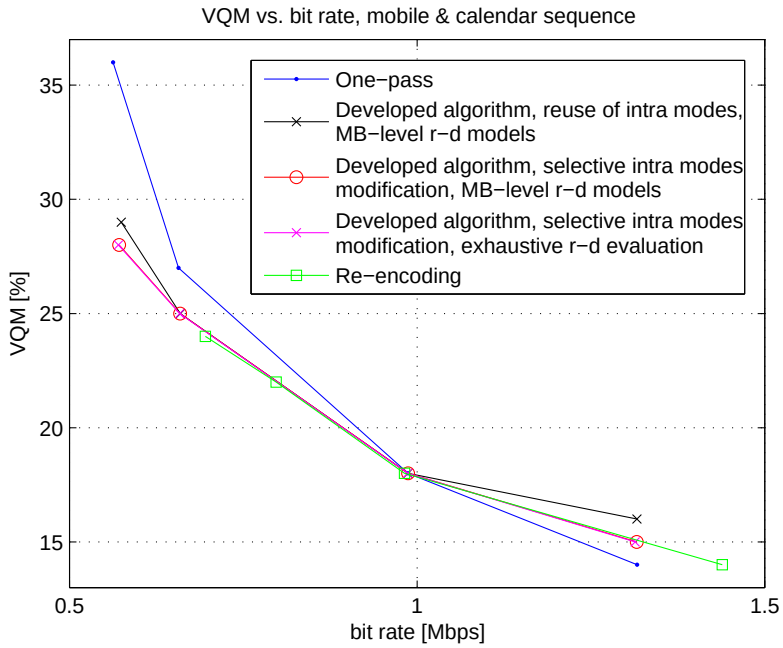

Figure 9.8: VQM vs. bit rate, for the mobile & calendar sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

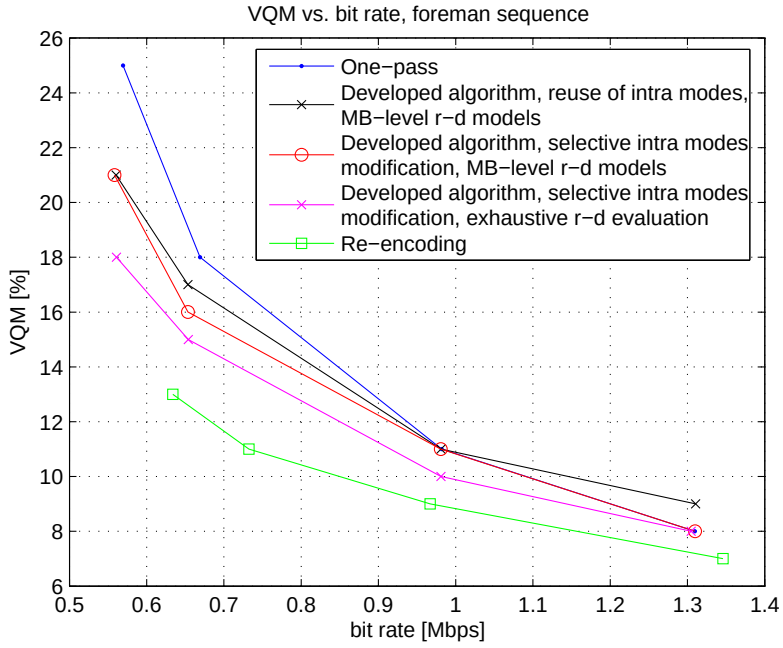

Figure 9.9: VQM vs. bit rate, for the foreman sequence. Blue dots: one-pass requantization. Black x: developed algorithm, reuse of intra modes, MB-level r-d models. Red circles: developed algorithm, selective intra modes modification, MB-level r-d models. Magenta x: developed algorithm, selective intra modes modification, exhaustive r-d evaluation. Green squares: re-encoding.

Overall, we conclude that the proposed model-based transrating scheme provides a good trade-off between quality and computational complexity, as summarized in Fig. 9.10. As compared to re-encoding, it saves the run-time by a factor of about 4, on average, with small PSNR loss at high to medium bit rates. By examining the graph slopes in Fig. 9.10, we conclude that the the proposed system's gain as compared to the one-pass requantization is higher than the re-encoding gain as compared to the proposed system.

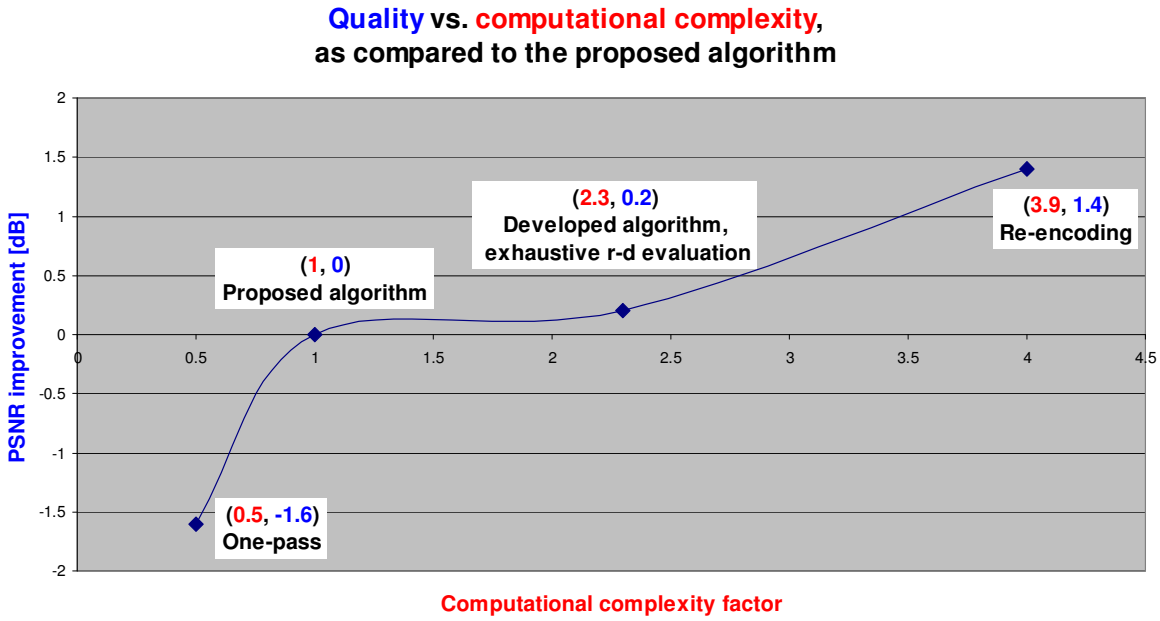

Figure 9.10: Quality vs. computational complexity, as compared to the proposed algorithm. The quality is measured by the PSNR improvement, and the computational complexity is measured by the run-time factor.

Chapter 10

Conclusions and Future Directions

10.1 Conclusion

This research work concerns model-based transrating of H.264 coded video. New requantization algorithms that incorporate models in ρ domain for H.264 were developed for two purposes. The first is to reduce the computational load of the optimal requantization algorithms. The second is to provide a model for the requantized coefficients in the presence of block dependencies.

To reduce the average bit rate of an encoded sequence, we should allocate the bits between the sequence frames. The simplest approach is to reduce the rate of each frame at the same factor. However, due to the advanced coding features in H.264, the overhead bits are not negligible, and such solution may leave too few bits for coding the residual. Therefore, in this work, we implemented an optimal bit allocation within a group of pictures, such that the average target bit rate is achieved, at minimal overall distortion. To keep a smooth constant video quality, the frame distortions were equalized. The first frame of each group of pictures is an intra-coded frame, followed by inter-coded frames, which are temporally predicted from previously encoded frame. Due to the different characteristic of intra-coded frames and

inter-coded frames, different requantization algorithms were developed.

The spatial prediction in H.264 intra-coded frames uses previously decoded neighbor pixels in the same frame to predict the current block pixels. It introduces dependencies between neighboring casual residual blocks. To avoid a noticeable drift error, the frame is fully decoded and then encoded. We perform a guided encoding, which uses already encoded information from the input bitstream. One option is to reuse the input prediction modes, and thereby reduce the run-time of intra transrating by a factor of 4, as compared to full modes search, at a cost of up to 1[dB] in PSNR (for an intra-coded frame). The other option is to selectively modify the input prediction modes where the coding efficiency is expected to improve (better quality at the same rate). The selective mode modification achieves practically the same quality as the full modes search, at a factor of about 1.6 less time.

The main bit rate reduction in intra-coded frames is achieved via uniform requantization. That is, all frame macroblocks share the same requantization step-size. Due to the residual block dependencies, the residual coefficients to be requantized are not available in advance, when the requantization step-size should be selected. The step-size selection uses the $rate - \rho$ model suggested in the literature, but the estimation of the relation between ρ and the requantization step size becomes a challenging task. To this end, we propose a novel closed-loop statistical estimator, that models the correction signal required for the residual coefficients. The correction signal is first segmented into homogenous data groups that share the same characteristic. Then, each group is fitted using a probability function, whose parameter is estimated from input encoded information. Its incorporation yields a 3% average rate deviation from the target, as compared to 10.8% average deviation, obtained using an open-loop estimator for ρ.

For inter-coded frames, we reuse the input motion decisions. The input frame is partially decoded, up to its residual transform coefficients. These coefficients undergo closed-loop correction and are then partially encoded. The non-uniform requantization step-sizes are optimally selected, using rate-distortion models in the ρ domain. To improve the subjective quality, we regulate the changes in the requantization step sizes throughout the frame. Therefore, we suggest extending the Lagrangian optimization in the following manner. At each Lagrangian iteration, where the relative rate-distortion weight is set, we apply a dynamic programming algorithm that minimizes the overall frame cost, subject to the step-size change regulation. At the end of each iteration, the achieved rate is compared to the target rate, and the relative weight is updated correspondingly.

To reduce the computational burden of the optimization, we use rate-distortion models at the macroblock level. As the models suggested in the literature are not suitable for macroblock level coding in H.264, we developed macroblock level rate-distortion models adapted to H.264 requantization. The models parameters are estimated based on the input encoded information. By incorporating the proposed models, the average rate deviation from the target is 4.5%, rather than 24.7%, using the models suggested in the literature. Also, the incorporation of macroblock level models reduces the optimization run-time by a factor of about 3.5, on average, as compared to an exhaustive optimization.

Overall, as compared to re-encoding (cascaded decoder-encoder), the proposed system reduces the computational complexity by a factor of about 4, at a maximal cost of 1.4[dB] in PSNR. In comparison with a simple one-pass requantization, the proposed algorithm achieves better performance both objectively (PSNR gain of up to 1.6[dB]) and subjectively, at the cost of twice the complexity.

10.2 Main contributions

The main contributions in this work are summarized below:

- **Closed-loop statistical $\rho - Q_2$ estimator for intra-coded frames**. A novel closed-loop statistical estimator for the $\rho - Q_2$ relation is proposed. It overcomes the block dependency problem by modeling the correction signal of the requantized residual.

- **Non-uniform optimal requantization for inter-coded frames**. We suggest to select optimal requantization step-sizes, while regulating the step-size changes throughout the frame, to improve the subjective quality. To this end, we extend each Lagrangian iteration by a constrained dynamic programming algorithm. To reduce the computational burden, rate-distortion models are used.

- **Macroblock level rate-distortion models in the ρ domain**. Novel rate-distortion models at the macroblock level, adapted to requantization in H.264, are proposed. The unique $rate - \rho$ model decomposition accurately describes the context adaptive entropy coding used in H.264. The models parameters are estimated based on the input encoded information.

10.3 Future directions

We see three main issues for future directions. The first and the second issues are inspired by the re-encoding approach, whereas the third is related directly to the proposed optimization.

The first of which involves the motion decisions. In this work, the input motion decisions are kept, as part of the low computational complexity FPDT architecture for inter-coded frames. As H.264 uses variable block size motion compensation, new motion decisions at lower bit rates can improve the quality. This extension involves

both motion partition modification and re-estimation of the motion vectors. Its incorporation will increase the system's computational complexity due to two reasons. First, to change the motion vectors, the input sequence should be fully decoded, which means performing the motion compensation operation twice (both at the decoder and at the encoder) instead of once (as done now). The motion decisions update itself requires finding what new block partition (among many possible combinations) is most suitable and re-estimating the corresponding motion vector(s).

The second extension is to enable the de-blocking filter. The de-blocking filter was disabled in this work as we use the FPDT architecture for inter-coded frames, whereas the filter is applied on the fully decoded sequence. Its incorporation can further reduce block artifacts and improve the quality, at the cost of an additional computational load, due to the full decoding of the input sequence and the adaptive filtering operations.

The recommended encoder eliminates very sparse blocks to reduce the rate (at increased distortion) using a simple elimination rule. When this rule was incorporated as part of the proposed selective elimination, it gave minor improvement over the optimal requantization. Incorporating a more sophisticated coefficients elimination rule should improve its gain.

Appendix A

The H.264 Standard

H.264 is currently the most powerful state of the art video coding standard. It is designed to improve the coding efficiency by a factor of about two over MPEG-2 (the same quality at half the encoded bit rate) [40, 54]. In this appendix we will briefly outline the encoder scheme and the main new coding features that enable the improvement in coding efficiency. Our algorithmic development is based on the baseline profile, therefore the proposed system does not support interlaced video and coding of B frames.

The video sequence is composed of groups of pictures (see Fig. A.1). Each Group is called a GOP and starts with an intra-coded frame. The rest of the frames are predicted temporally (inter-coded frames). Each coded picture, i.e. a video frame, contains slices, where a slice is a unit which is encoded and decoded independently of the other slices. A slice includes an integer number of macroblocks. Each macroblock describes a picture region of size 16x16 pixels. Coding decisions such as quantization step size and intra/inter prediction are controlled at the macroblock level.

In the sequel, we fill in details not given in Chapter 3, that are relevant to this work. As this work concerns requantization of the transform coefficients, more focus is given to the transform, quantization and entropy coding.

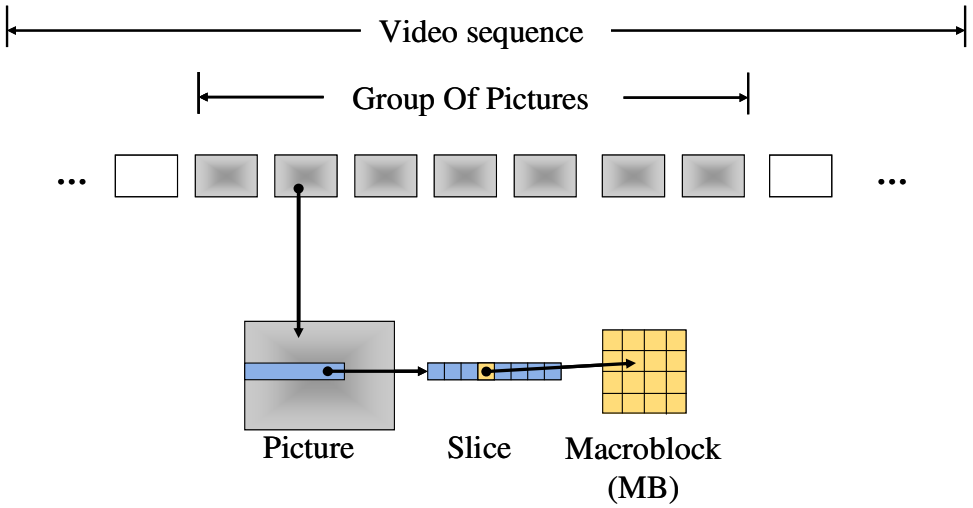

Figure A.1: Coded video structure.

A.1 Intra prediction

Neighboring image blocks in intra coded frames often have a high spatial correlation. To improve the coding efficiency, image blocks are predicted from previously decoded neighbor pixels in the same image [40]. The prediction is performed in the pixel domain, and is based on decoded pixel values of blocks at the left or above the current predicted block. There are a number of prediction modes types: 4x4 luminance block prediction, 16x16 luminance block prediction and 8x8 chrominance block prediction. In addition, there is a possibility to encode the pixel values of the block directly (without prediction and transform).

4x4 luminance block prediction

Small blocks prediction suits high detailed image regions. There are nine such prediction modes, as illustrated in Fig. A.2. These include vertical and horizontal prediction, averaging pixels from neighboring blocks and diagonal predictions that match different textures, as explained in Table A.1 [40]. The best prediction mode for each block is the one that minimizes the rate-distortion cost function.

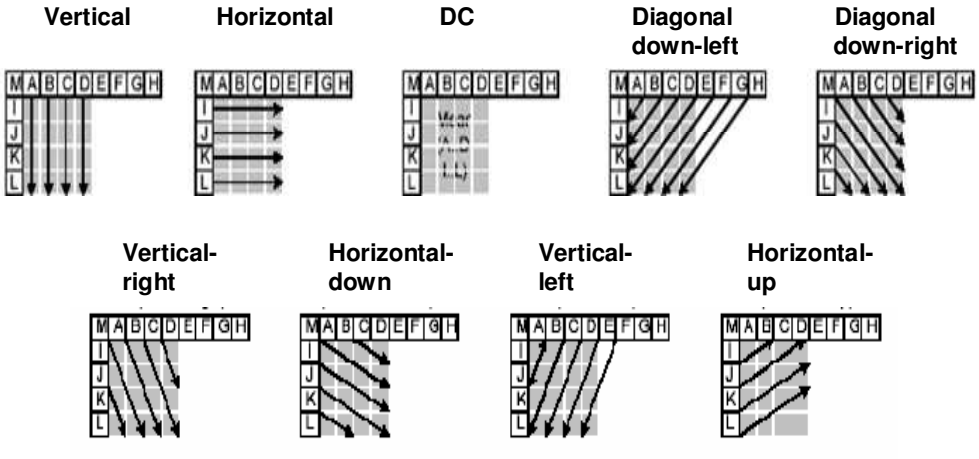

Figure A.2: Luminance 4x4 intra prediction modes.

Table A.1: Description of intra 4x4 prediction modes.

Mode name	Description
Vertical	The upper samples A, B, C, D are extrapolated vertically.
Horizontal	The left samples I, J, K, L are extrapolated horizontally.
DC	All samples are predicted by the mean of samples A...D and I...L.
Diagonal Down-Left	The samples are interpolated at a 45° angle between lower -left and upper-right.
Diagonal Down-Right	The samples are extrapolated at a 45° angle down and to the right.
Vertical-Right	Extrapolation at an angle of approximately 26.6° to the left of vertical (width/height = 1/2).
Horizontal-Down	Extrapolation at an angle of approximately 26.6° below horizontal.
Vertical-Left	Extrapolation (or interpolation) at an angle of approximately 26.6° to the right of vertical.
Horizontal-Up	Interpolation at an angle of approximately 26.6° above horizontal.

16x16 luminance block prediction

For relatively smooth image regions, the intensity changes within a 16x16 block are small, and therefore it is better to predict the whole block with no further division. There are four possibilities for such a prediction, as illustrated in Fig. A.3. The vertical, horizontal and DC predictions are obtained similarly to the corresponding 4x4 predictions. In the plane mode, a linear plane function is fitted to the upper and left samples (H and V in Fig. A.3) to predict regions of smoothly changing luminance.

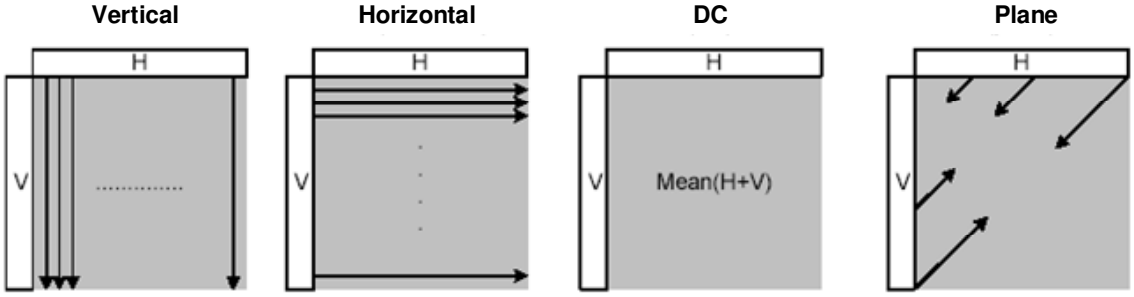

Figure A.3: Luminance 16x16 and chrominance 8x8 intra prediction modes.

8x8 chrominance block prediction

Chrominance 8x8 blocks are relatively smooth (using a 4:2:0 color format). These are predicted from their neighboring blocks using the same modes defined for 16x16 luminance block prediction. The same mode is chosen for both chrominance components (U & V) of the same block.

Coding without prediction and transform (Intra PCM)

This coding mode allows to bypass the prediction and transform for an intra coded block and to encode the pixel values directly [54]. Such block is encoded losslessly but with poor compression. This coding type allows to accurately represent the content of an anomaly block and poses an upper bound to the number of bits spent on coding a block. Practically, this mode is rarely used.

A.2 Motion compensation

Similar to previous standards, H.264 performs motion compensation to improve the coding efficiency. The main new features enabled regarding motion compensation are variable block size support and $\frac{1}{4}$ pixel MV resolution [40].

Variable block size motion compensation allows assigning different motion to different macroblock parts. Homogenous image parts can be assigned with the same motion, that is a large partition, whereas high detailed regions can have small partitions. For the luminance component, these parts can vary from 16x16 pixel blocks to 4x4 pixel blocks, with many intermediate options in between. Fig. A.4 depicts the allowed partitions for a luminance component of one macroblock (16x16 pixels). It can be partitioned using one of the four possibilities: $\{16x16, 16x8, 8x16, 8x8\}$ shown in the top row. If the chosen option is 8x8, then each of the four 8x8 blocks can be further partitioned using one of the four possibilities :$\{8x8, 8x4, 4x8, 4x4\}$ shown in the low row of Fig. A.4. The chrominance components share the same motion vectors as the luminance component, up to a scale factor due to their decimation.

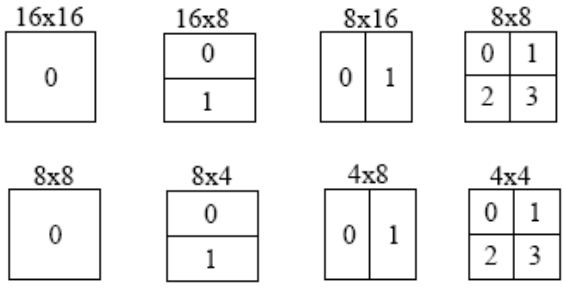

Figure A.4: Variable block size motion compensation. Top row: optional macroblock partitions (16x16, 8x16, 16x8 and 8x8), low row: optional 8x8 block partitions (8x8, 4x8, 8x4 and 4x4).

The motion vectors have a $\frac{1}{4}$ pixel resolution (in the luminance component), where the interpolation is composed of two stages. First, a $\frac{1}{2}$ pixel interpolation is performed using the 6-tap FIR $[1, -5, 20, 20, -5, 1]$ separably in the vertical and horizontal dimensions. Then, the $\frac{1}{2}$ pixel interpolated samples are averaged to obtain the $\frac{1}{4}$ pixel samples.

A.3 Transform

The transform defined in H.264 is carried out on small 4x4 blocks. The core transform is an Integer Cosine Transform (ICT) [28] that can be implemented at low computational cost using just shifts and adds, as it transforms integer pixel values to integer transform coefficients. It is defined by CXC^T, where X is an 4x4 pixels block and C is the transform matrix of (A.1).

$$C = \begin{pmatrix} 2 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix} \tag{A.1}$$

To approximate the 4x4 DCT, additional scaling is required, and is incorporated in the quantization process. The standard defines scaling factors that are slightly altered as a function of the quantization step size. The average scaling factors are given in (A.2). The scaled transform coefficients are then obtained using (A.3), where the operator $.*$ denotes pointwise multiplication.

$$E = \begin{pmatrix} 0.25 & 0.15811 & 0.25 & 0.15811 \\ 0.15811 & 0.1 & 0.15811 & 0.1 \\ 0.25 & 0.15811 & 0.25 & 0.15811 \\ 0.15811 & 0.1 & 0.15811 & 0.1 \end{pmatrix} \tag{A.2}$$

$$Y = CXC^T . * E \tag{A.3}$$

For smooth regions, coded as 16x16 luminance intra prediction blocks or any 8x8 chrominance blocks, some spatial correlation remains between the 4x4 transform blocks. Therefore, a Hadamard transform is carried out on the grouped DC coefficients of that smooth macroblock. The grouped DC coefficients can either form a block of size 4x4 for 16x16 luminance intra prediction blocks (where the matrix of

(A.4) is used) or of size 2x2 for a chrominance block (where the matrix of (A.5) is used). The overall two-phase transform is illustrated in Fig. A.5 for a macroblock whose luminance component is predicted using a 16x16 prediction.

$$H_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \tag{A.4}$$

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{A.5}$$

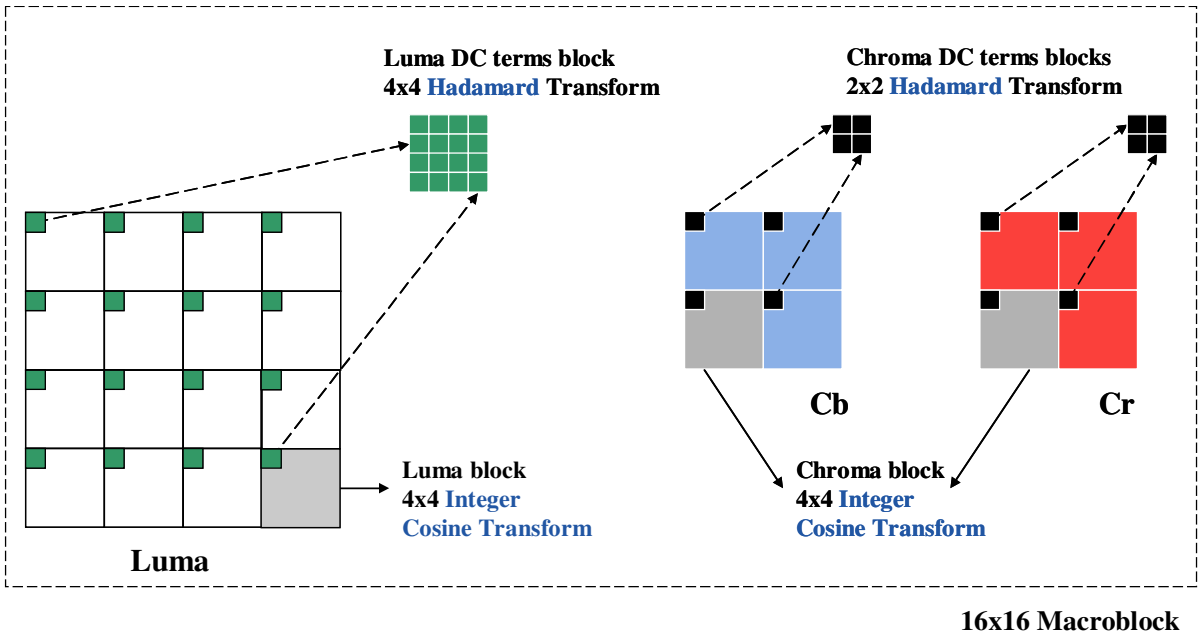

Figure A.5: Two-phase transform illustrated for a macroblock whose luminance component is predicted as intra 16x16.

A.4 Quantization

The standard defines 52 quantization steps that grow logarithmically with the quantization parameter QP, where an increase of 6 in QP corresponds to a step-size factor of 2 (a factor of about $2^{\frac{1}{6}} = 1.12$ between consecutive step sizes). The step sizes

Table A.2: Quantization steps. QP is the quantization parameter (index), and the step size grows logarithmically with QP.

QP	0	1	2	3	4	5	6	7	8
step size	**0.625**	0.6875	0.8125	0.875	1	1.125	**1.25**	1.375	1.625
QP	9	10	11	12	13	14	15	16	17
step size	1.75	2	2.25	**2.5**	2.75	3.25	3.5	4	4.5
QP	18	19	20	21	22	23	24	25	26
step size	**5**	5.5	6.5	7	8	9	**10**	11	13
QP	27	28	29	30	31	32	33	34	35
step size	14	16	18	**20**	22	26	28	32	36
QP	36	37	38	39	40	41	42	43	44
step size	**40**	44	52	56	64	72	**80**	88	104
QP	45	46	47	48	49	50	51		
step size	112	128	144	**160**	176	208	224		

defined cover a wide range, starting from 0.625 to 224, see Table A.2. The step sizes for the chrominance components are finer than those of the luminance component, and defined using a mapping of the quantization parameter. In the recommended reference software, the quantizer has a deadzone, dz, set according to the block type:

$$dz = \begin{cases} \frac{1}{3} & Intra\ coded\ block \\ \frac{1}{6} & Inter\ coded\ block \end{cases} \tag{A.6}$$

The quantization process for a scaled coefficient Y with step size Q and deadzone dz is defined by:

$$Z = sgn(Y) \cdot \lfloor \frac{|Y|}{Q} + dz \rfloor \tag{A.7}$$

A scaled coefficient that underwent the Hadamard transform has a different effective quantization step size, according to:

- $4Q(QP)$ - DC coefficients of 16x16 prediction modes
- $2Q(QP)$ - DC coefficients of chroma prediction modes
- $Q(QP)$ - 4x4 prediction modes and AC coefficients of the others

A.5 Entropy coding

There are two main types of data that is entropy coded. The texture bits describe the residual coding whereas the overhead bits are spent on coding side information.

A.5.1 Texture bits coding

Introduction

The entropy coding used in H.264 baseline profile for coding the quantized transform coefficients is called Context Adaptive Variable Length Coding, or simply CAVLC [40]. It translates a 4x4 quantized indices block to a sequence of code words. The CAVLC was designed in order to take advantage of the quantized blocks characteristics, as listed in Table A.3.

Table A.3: Characteristics of quantized blocks and their usage in the CAVLC.

Characteristic	CAVLC usage
Blocks are usually sparse	Uses run-level coding
The number of non-zero coefficients in neighboring blocks is correlated	Predict the number of non-zero coefficients in the current block and switch between the VLC tables accordingly
The highest frequency non-zero coefficients are usually ± 1	Treat these ± 1 as special levels, where only the sign should be encoded
The magnitude of non-zero coefficients tends to be larger at the lower frequencies	Switch the level's code table choice based on the previous coded levels.

Definitions

- TotalCoeffs - The number of non-zero coefficients in the block
- TrailingOne - Coefficient at the end of the scanned block, whose value is ± 1. There can only be maximum three trailings.
- Level - The value of a non-zero coefficient which is not a trailing (including the sign)

- TotalZeros - The number of zeroed coefficients from the DC coefficient (including) to the highest frequency non-zero coefficient
- RunBefore - The number of zeroed coefficients preceding each non-zero coefficient
- ZerosLeft - The number of zeros left to encode

After a zig-zag scan of the block, it is translated to a set of syntax elements by scanning its values from the highest frequency to the lowest frequency.
For example, a 4x4 quantization indices block:

$$\begin{pmatrix} 0 & 3 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

After zig-zag scan, the reordered block is: 0,3,0,1,-1,-1,0,1,0. . .
This set of values is translated into the following syntax elements, as described in Table A.4.

Table A.4: Example for the CAVLC syntax elements decomposition of one block.

Syntax Element	Value
(TotalCoeffs, TrailingOnes)	(5, 3)
TrailingOne sign (4) +	0
TrailingOne sign (3) -	1
TrailingOne sign (2) -	1
Level (1)	1
Level (0)	3
TotalZeros	3
RunBefore (4)	ZerosLeft = 3, RunBefore = 1
RunBefore (3)	ZerosLeft = 2, RunBefore = 0
RunBefore (2)	ZerosLeft = 2, RunBefore = 0
RunBefore (1)	ZerosLeft = 2, RunBefore = 1
RunBefore (0)	ZerosLeft = 1, RunBefore = 1

Syntax elements and tables

(TotalCoeffs, TrailingOnes)

This syntax element describes the combination of the number of non-zero coefficients in the block (between 0 and 16 for a 4x4 block) and the number of trailing ones (between 0 and 3). If there are more than three ±1 at the end of the block, only the last three are considered as trailings, and the rest are coded as regular coefficients. There are 4 different tables for this element (see Fig. A.6).

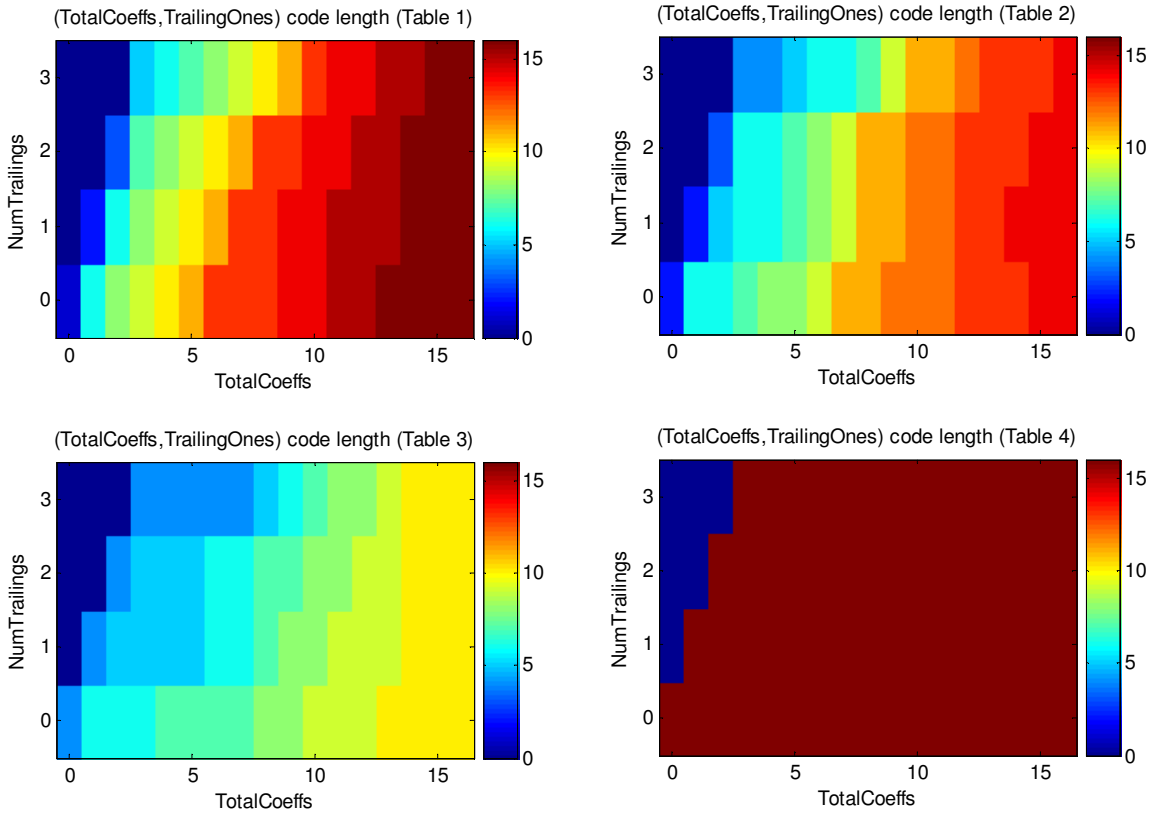

Figure A.6: The 4 tables for coding the (TotalCoeffs, TrailingOnes) syntax element. Horizontal axis: TotalCoeffs, vertical axis: TrailingOnes. The color describes the code word length (warmer colors for longer codes).

First, we should note that TotalCoeffs≥TrailingOnes and thus non-feasible combinations of (TotalCoeffs, TrailingOnes) do not have a code word (darkest blue color in the tables). All variable length tables assign a short code to the common combination of (TotalCoeffs, TrailingOnes)=(0,0), that is an all-zero block. In case such block is encoded (see coded block pattern description), its texture bits include only

the (TotalCoeffs, TrailingOnes) syntax element. Table 1 is biased towards small numbers of coefficients such that high values of TotalCoeff are assigned with particularly long codes. Table 2 is biased towards medium numbers of coefficients (TotalCoeff values around $2 \sim 4$ are assigned relatively short codes). Table 3 is biased towards higher numbers of coefficients and Table 4 assigns a fixed six-bit code to every feasible combination of (TotalCoeff, TrailingOnes).

The choice of the table depends on the number of non-zero coefficients at the left and upper neighbors of the block, denoted by n_{left}, n_{up}, respectively. The expected number of non-zero coefficients in the current block is predicted by:

$$n = \begin{cases} round(\frac{n_{left}+n_{up}}{2}) & \text{if both are available} \\ n_{left} & \text{if only the left is available} \\ n_{up} & \text{if only the upper is available} \\ 0 & \text{if nither are available} \end{cases} \tag{A.8}$$

Then, the VLC table to be used is chosen according to:

$$TN = \begin{cases} 1 & n = 0, 1 \\ 2 & n = 2, 3 \\ 3 & n = 4, 5, 6, 7 \\ 4 & else \end{cases} \tag{A.9}$$

TrailingOne sign

The sign of each trailing is coded with one bit (1 for - and 0 for +). The trailings are coded in reverse order from the highest frequency towards the lowest frequency.

Level of the remaining non-zero coefficients

The level (magnitude and sign) is encoded in reverse order from the highest frequency towards the lowest frequency. There are 7 different tables for the level coding, see Fig. A.7. The code word length depends on the magnitude of the level and includes its sign. As the table number increases, it assigns shorter codes to bigger coefficients. The choice of the table is updated after each level is coded !! (context adaptive) in the following manner. If the coefficient's magnitude exceeded a certain threshold, the table number is increased. The highest frequency coefficient is coded with table #0 and the first threshold is zero, so that the next coefficient is encoded by table #1. These thresholds are given in Table A.5.

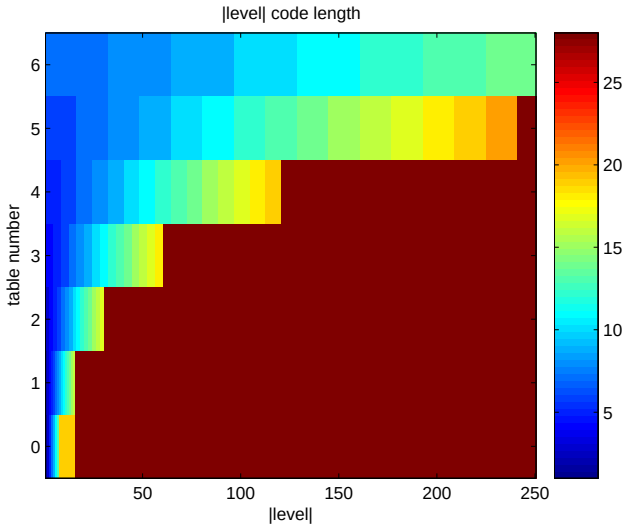

Figure A.7: The 7 tables for coding the Level values. Horizontal axis: Level magnitude, vertical axis: table number. The color describes the code word length (warmer colors for longer codes).

Table A.5: Level tables updating.

Current table number	Threshold to increment the next table number
0	0
1	3
2	6
3	12
4	24
5	48
6	Last table

TotalZeros

The number of zeroed coefficients from the DC coefficient to the highest frequency non-zero coefficient is encoded separately. The reason is that most blocks contain a number of non-zero coefficients at the start of the block. Given the TotalZeros, the zero-run for the lowest frequency coefficient need not be encoded.

TotalZeros can take values between 0 and 15 (TotalZeros=16 is an all zeroed block). Since TotalZeros+TotalCoeffs$\leq$ 16 , the code table for TotalZeros is chosen according to the TotalCoeffs value (there are 15 different tables, see Fig. A.8).

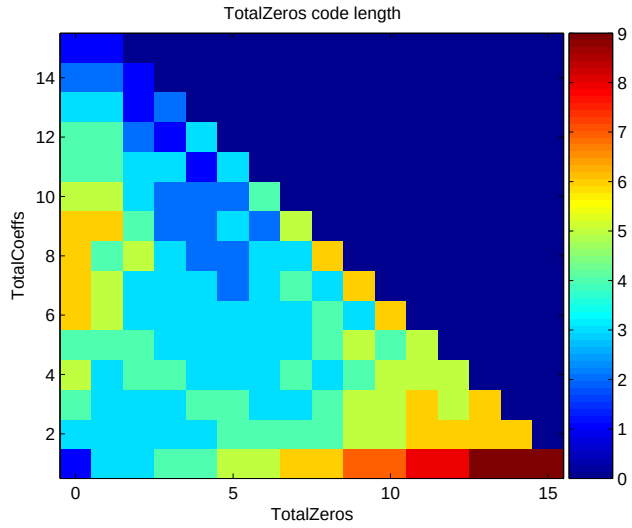

Figure A.8: The 15 tables for TotalZeros. Horizontal axis: TotalZeros, vertical axis: TotalCoeffs. The color describes the code word length (warmer colors for longer codes).

RunBefore

The number of zeroed coefficients preceding each non-zero coefficient is encoded in reverse order from the highest frequency towards the lowest frequency. Since the zero-run for the lowest frequency coefficient need not be coded, the run can take values between 0 and 14. There are 7 different tables for the run code (see Fig. A.9). The choice of table depends on the number of zeros left to encode. For example, if ZerosLeft=2, then run can be 0, 1 or 2 and a two-bit code word is selected.

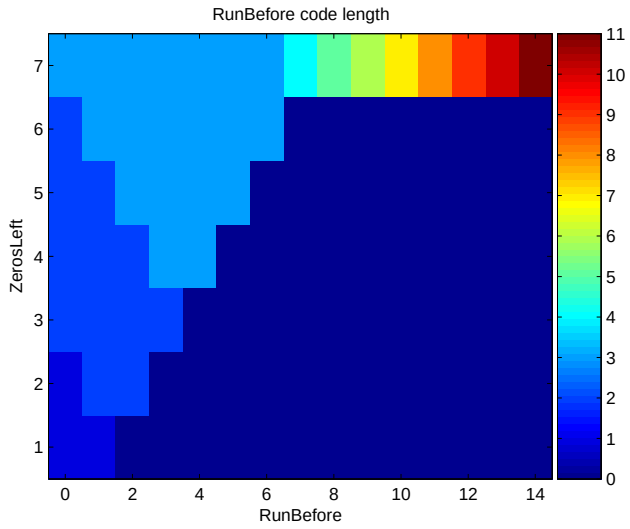

Figure A.9: The 7 tables for RunBefore. Horizontal axis: RunBefore, vertical axis: ZerosLeft. The color describes the code word length (warmer colors for longer codes).

A.5.2 Overhead bits coding

The overhead bits describe coding the side information, that includes intra prediction modes, motion vectors and partitions, quantization parameters, coded block patterns, etc. Basically, all side information is encoded using the Exp-Golomb code, as listed in Table A.6.

Table A.6: Exp-Golomb code-words.

Code number	Binary code word	Code word length
0	1	1
1	010	3
2	011	3
3	00100	5
4	00101	5
5	00110	5
6	00111	5
7	0001000	7
8	0001001	7
...	...	...

Intra prediction modes coding

For each intra coded MB, the encoder signals the relevant prediction modes as side information [1]:

- Luminance prediction modes: Either one 16x16 mode or 16 4x4 modes.
- Chrominance prediction mode

For intra 4x4 prediction, the encoder uses the correlation between neighboring prediction modes. A "most probable mode" is defined as the minimum between the upper block mode and the left block mode. If either is unavailable for the prediction (out of slice / out of picture / coded as 16x16 prediction), the "most probable mode" is set to the DC prediction mode. In case the chosen mode is indeed the expected "most probable mode", it takes one bit to encode it. Else, 4 bits are required.

The encoder signals the 16x16 mode as part of the macroblock type, at a cost of approximately 9 bits. The chrominance prediction modes cost are $\{1, 3, 3, 5\}$ for the {DC, horizontal, vertical and plane} modes, respectively.

Motion vectors and block partition coding

For each non-skipped MB, both the inter mode(s) and the MV(s) are encoded. The inter modes bit-cost changes with the block partition according to [1]:

$$\begin{cases} 1 & \text{16x16 partition} \\ 3 & \text{16x8 or 8x16 partition} \\ 5 + \sum_{k=1}^{4} Cost(subpartition_k) & \text{8x8 and finer partitions} \end{cases} \tag{A.10}$$

where

$$Cost(subpartition_k) = \begin{cases} 1 & \text{8x8 sub partition} \\ 3 & \text{8x4 or 4x8 sub partition} \\ 5 & \text{4x4 sub partitions} \end{cases} \tag{A.11}$$

Due to the high correlation between neighboring MVs, each MV is first predicted from its neighbors using a median predictor (at each coordinate). The two components of the differential MV obtained are then encoded using an Exp-Golomb table. Obviously, a finer partition requires coding more MVs which costs more bits.

Quantization parameter

The quantization parameter QP is encoded differentially, that is,
$\Delta QP = QP - QP_{Prev}$ is encoded, where QP_{Prev}, QP are the quantization parameters of consecutive macroblocks. The cost in bits of the ΔQP transition increases with its absolute value, such that [1]:

$$cost(QP_{Prev}, QP) = cost(\Delta QP) = \begin{cases} 1 & \Delta QP = 0 \\ 3 & |\Delta QP| = 1 \\ 5 & 2 \leq |\Delta QP| \leq 3 \\ 7 & 4 \leq |\Delta QP| \leq 7 \\ 9 & 8 \leq |\Delta QP| \leq 15 \\ 11 & 16 \leq |\Delta QP| \leq 31 \\ etc. & \end{cases} \tag{A.12}$$

As a result, many rate control algorithms for H.264 limit $|\Delta QP|$ to take small values (up to 2).

Coded Block Pattern

The Coded Block Pattern (CBP) indicates what blocks within a macroblock contain coded coefficients. If a 8x8 indices block is all-zeroed after the quantization, it is signalled as zero in the coded block pattern and no texture bits are spent on coding it.

A.6 Selective coefficients elimination

To improve the coding gain, the recommended reference software [1] performs expensive coefficient elimination for inter-coded blocks. A sparse block is considered as a candidate for elimination if it is all zeroed, except a few trailing-ones. The decision whether or not to eliminate such a block depends on the number of its trailing-ones and their location inside the block. A cost function is defined at the 4x4 block level for quantized indices whose magnitude is 1, according to their location in the block:

$$\begin{pmatrix} 3 & 2 & 1 & 0 \\ 2 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Quantized indices whose magnitude is 0 have zero cost, and those with a magnitude bigger than 1 have an infinite (or any other number higher than the thresholds described next) cost.

- For each luma 8x8 block, the sum of costs of its internal 4x4 blocks is calculated. If it is below a threshold (which is 4), the block is zeroed, otherwise it is retained.
- If a luma macroblock has a total cost below a threshold (which is 5) it is zeroed.
- The same rule goes for 8x8 chroma blocks, but only AC coefficients are candidates for zeroing (elimination may result in a 8x8 block that contains DC coefficients only) with a different threshold (which is 3).

Appendix B

Optimal GOP Level Bit Allocation

In this section we describe the solution of the optimization problems discussed in section 4.2.1 for model-based GOP level bit allocation.

Using the frame level $rate - \rho$ and $distortion - \rho$ models:

$$R_k^{texture}(\rho_k) = \theta_k(1-\rho_k) \quad \text{(B.1)}$$

$$D_k(\rho_k) = \sigma_k^2 \cdot exp(-\alpha_k(1-\rho_k)) \quad \text{(B.2)}$$

we eliminate the ρ_k dependency, and express the rate $R_k^{texture}$ in terms of the re-quantization distortion D_k (B.3), and vice versa (B.4).

$$R_k^{texture}(D_k) = \frac{\theta_k}{\alpha_k} \cdot ln(\frac{\sigma_k^2}{D_k}) \quad \text{(B.3)}$$

$$D_k(R_k^{texture}) = \sigma_k^2 \cdot exp(-\alpha_k \frac{R_k^{texture}}{\theta_k}) \quad \text{(B.4)}$$

B.1 Overall distortion minimization

The first optimization problem discussed in section 4.2.1 is (4.8):

$$\min_{\{R_k^{texture}\}} \sum_{k=1}^{N} D_k(\rho_k) \tag{B.5}$$

$$subject \quad to:$$

$$\sum_{k=1}^{N} R_k^{texture}(\rho_k) \leq R_{GOP,target}^{texture}$$

We convert the constrained problem into an unconstrained problem using the Lagrangian cost function and the distortion-rate relation (B.4):

$$\min_{\{R_k^{texture}\}} L = \min_{\{R_k^{texture}\}} \sum_{k=1}^{N} \sigma_k^2 \cdot exp(-\alpha_k \frac{R_k^{texture}}{\theta_k}) + \lambda \cdot [\sum_{k=1}^{N} R_k^{texture} - R_{GOP,target}^{texture}] \tag{B.6}$$

Differentiating according to the optimization variables $\{R_k\}_{k=1}^{N}$ (B.7) and according to the Lagrangian parameter (B.8):

$$\frac{\partial L}{\partial R_k} = \sigma_k^2 \cdot exp(-\alpha_k \frac{R_k^{texture}}{\theta_k}) \cdot (-\frac{\alpha_k}{\theta_k}) + \lambda = 0 \tag{B.7}$$

$$\Rightarrow R_k^{texture} = \xi_k \cdot ln(\frac{\sigma_k^2}{\xi_k \lambda})$$

$$\frac{\partial L}{\partial \lambda} = \sum_{k=1}^{N} R_k^{texture} - R_{GOP,target}^{texture} = 0 \tag{B.8}$$

where $\xi_k = \frac{\theta_k}{\alpha_k}$.

Substituting (B.7) into (B.8), we get:

$$\sum_{k=1}^{N} \xi_k \cdot ln(\frac{\sigma_k^2}{\xi_k \lambda}) = R_{GOP,target}^{texture} \tag{B.9}$$

$$\Rightarrow ln(\lambda) = \frac{\sum_{k=1}^{N} \xi_k ln(\frac{\sigma_k^2}{\xi_k}) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \xi_k}$$

And the frame level bit allocation is (4.9):

$$R_{target,k}^{texture} = \xi_k \cdot ln(\frac{\sigma_k^2}{\xi_k}) + \frac{\xi_k}{\sum_{k=1}^{N} \xi_k} \cdot (R_{GOP,target}^{texture} - \sum_{k=1}^{N} \xi_k ln(\frac{\sigma_k^2}{\xi_k})) \tag{B.10}$$

Using (4.9) and (B.4), the distortion allocation is:

$$D_k = \xi_k \cdot exp(\frac{\sum_{k=1}^{N} \xi_k \cdot ln(\frac{\sigma_k^2}{\xi_k}) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \xi_k}) \tag{B.11}$$

B.2 Equalizing frame distortions

The proposed equal-distortion optimization problem discussed in section 4.2.1 is (4.12):

$$\min_{\{R_k^{texture}\}} \sum_{k=1}^{N} D_k(\rho_k) \tag{B.12}$$

$$subject \quad to:$$

$$\sum_{k=1}^{N} R_k^{texture}(\rho_k) \leq R_{GOP,target}^{texture}$$

$$D_1(\rho_1) = D_2(\rho_2) = ... = D_N(\rho_N)$$

We use the equi-distortion constraint, and denote by γ these distortions, $\gamma = D_k$ for $1 \leq k \leq N$. As a result, minimizing the total distortion $\sum_{k=1}^{N} D_k$ is equivalent to minimizing $N\gamma$ or simply minimizing γ. Substituting γ and (B.3) into the target rate constraint, we get

$$\begin{aligned} R_{GOP,target}^{texture} &= \sum_{k=1}^{N} R_k^{texture} \\ &= \sum_{k=1}^{N} \frac{\theta_k}{\alpha_k} \cdot ln(\frac{\sigma_k^2}{\gamma}) = \sum_{k=1}^{N} \frac{\theta_k}{\alpha_k} \cdot ln(\sigma_k^2) - ln(\gamma) \sum_{k=1}^{N} \frac{\theta_k}{\alpha_k} \end{aligned} \tag{B.13}$$

Therefore, the distortion is:

$$\begin{aligned} ln(\gamma) = ln(D_k) &= \frac{\sum_{k=1}^{N} \frac{\theta_k}{\alpha_k} \cdot ln(\sigma_k^2) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \frac{\theta_k}{\alpha_k}} \\ D_k &= exp(\frac{\sum_{k=1}^{N} \xi_k \cdot ln(\sigma_k^2) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \xi_k}) \end{aligned} \tag{B.14}$$

where $\xi_k = \frac{\theta_k}{\alpha_k}$. And the target texture bit allocation for each frame is:

$$R_{target,k}^{texture} = \xi_k \cdot [ln(\sigma_k^2) - \frac{\sum_{k=1}^{N} \xi_k \cdot ln(\sigma_k^2) - R_{GOP,target}^{texture}}{\sum_{k=1}^{N} \xi_k}] \tag{B.15}$$

Appendix C

Γ Probability Distribution

In this section we describe the Γ probability distribution, used in section 5.2.4 for modeling the closed-loop correction signal distribution.

C.1 Definition

The probability density function for the two-sided Γ distribution is defined as [35]:

$$p_X(x) = \frac{1}{2\sqrt{\pi}}\sqrt{\frac{\beta}{|x|}} \cdot exp\{-\beta|x|\} \tag{C.1}$$

where $\beta > 0$ is the scale parameter. As we decrease β, the distribution gets wider (see Fig.C.1).

The CDF (Cumulative Distribution Function) is defined by:

$$F_X(x) = \begin{cases} \frac{1}{2} + \frac{1}{2\sqrt{\pi}}\Gamma(\beta x, 0.5) & x \geq 0 \\ \frac{1}{2} - \frac{1}{2\sqrt{\pi}}\Gamma(-\beta x, 0.5) & x < 0 \end{cases} = \frac{1}{2} + sgn(x)\frac{1}{2\sqrt{\pi}}\Gamma(\beta|x|, 0.5) \tag{C.2}$$

where the Γ function is defined as:

$$\Gamma(x) = \int_0^\infty t^{x-1}exp(-t)dt \tag{C.3}$$

and the incomplete Γ function is defined as:

$$\Gamma(a, x) = \int_0^a t^{x-1}exp(-t)dt \tag{C.4}$$

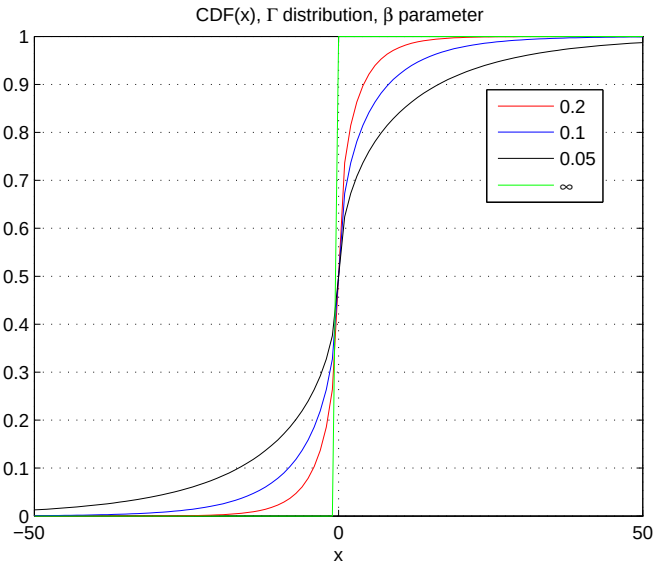

Figure C.1: CDF for the Γ distribution, where β is a parameter.

Evaluating $\Gamma(a, 0.5)$ requires a computational procedure that is based on Taylor's series and continued fractions [11, 3]. During our estimation of the $\rho - Q_2$ relation, $\Gamma(a, 0.5)$ needs to be evaluated multiple times, which makes this task computationally expensive. To reduce that computational load, we evaluated once $\Gamma(a, 0.5)$ at a high resolution set of a values and saved the results in a look up table (LUT).

C.2 Maximum likelihood parameter estimation

We now show how to find the maximum likelihood (ML) estimator for the parameter β. The log-likelihood function for N data points $x_1, x_2, ..., x_N$ (assuming all data points are mutually independent) is:

$$log(p_X(x)) = Nlog(\frac{1}{2}\sqrt{\frac{\beta}{\pi}})) - 0.5\sum_{i=1}^{N} log|x_i| - \beta\sum_{i=1}^{N}|x_i| \tag{C.5}$$

By differentiating with respect to β:

$$\frac{\partial}{\partial\beta}log(p_X(x)) = \frac{N}{2\beta} - \sum_{i=1}^{N}|x_i| = 0 \tag{C.6}$$

we get that the ML estimator is inversely proportional to the mean of absolute data

values:

$$\hat{\beta} = \frac{0.5N}{\sum_{i=1}^{N} |x_i|} \tag{C.7}$$

In case the data is all zeroed, we get $\widehat{\beta} = \infty$, that matches

$\lim_{\beta \to \infty} p_X(x) = \delta(x)$.

References

[1] H.264 reference software. http://bs.hhi.de/∼suehring/tml/download/.

[2] MPEG-2 test model 5, rate control. http://www.mpeg.org/MSSG/tm5.

[3] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions, National Bureau of Standards, Applied Math*, chapter 6.5. 55. Dover Publications, 1965.

[4] I. Ahmad et al. Video transcoding: An overview of various techniques and research issues. *IEEE transactions on multimedia*, 7(5):793–804, October 2005.

[5] Y. Altunbasak and N. Kamaci. ρ domain rate-distortion optimal rate control for DCT-based video coders. In *International Conference on Acoustics, Speech, and Signal Processing*, May 2004.

[6] P.A.A. Assuncao and M. Ghanbari. A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams. *IEEE transactions on Circuits and Systems for Video Technology*, 8(8):953–967, December 1998.

[7] P. Carlsson, F. Pan, and L. T. Chia. Coefficient thresholding and optimized selection of the lagrangian multiplier for non-reference frames in H.264 video coding. In *International Conference on Image Processing*, pages 773–776, 2004.

[8] Z. Chen and K.N. Ngan. Recent advances in rate control for video coding. *Signal Processing: Image Communications*, 22:19–38, 2007.

[9] T. Chiang and Y.Q. Zhang. A new rate control scheme using quadratic rate distortion model. *IEEE transactions on Circuits and Systems for Video Technology*, 7(1):246–250, February 1997.

[10] A. Eleftheriadis and D. Anastassioiu. Constrained and general dynamic rate shaping of compressed digital video. In *International Conference on Image Processing*, pages 396–399, 1995.

[11] A. W. Gautschi. A computational procedure for incomplete gamma functions. *ACM Transactions on Mathematical Software*, 5(4):466–481, December 1979.

[12] Z. He and T. Chen. Linear rate control for JVT video coding. In *IEEE International Conference on Information Technology: Research and Education, ITRE2003*, pages 65–68, 2003.

[13] Z. He, Y.K. Kim, and S.K. Mitra. Low-delay rate control for DCT video coding via ρ-domain source modeling. *IEEE transactions on Circuits and Systems for Video Technology*, 11(8):928–940, August 2001.

[14] Z. He and S.K. Mitra. A linear source model and a unified rate control algorithm for DCT video coding. *IEEE transactions on Circuits and Systems for Video Technology*, 12(11):970–982, November 2002.

[15] Z. He and S.K. Mitra. Optimum bit allocation and accurate rate control for video coding via ρ-domain source modeling. *IEEE transactions on Circuits and Systems for Video Technology*, 12(10):840–894, October 2002.

[16] N. Kamaci, Y. Altunbasak, and R.M. Mersereau. Frame bit allocation for the H.264/AVC video coder via Cauchy-density-based rate and distortion models. *IEEE transactions on Circuits and Systems for Video Technology*, 15(8):994–1006, August 2005.

[17] R.L. Lagendjik, E.D. Frimout, and J. Biemond. Low-complexity rate-distortion optimal transcoding of MPEG I-frames. *Signal Processing: Image Communication*, 15:531–544, 2000.

[18] J. Lan, W. Zeng, and X. Zhuang. Operational distortion-quantization curve-based bit allocation for smooth video quality. *Signal Processing: Image Communications*, 16:527–543, 2005.

[19] M. Lavrentiev. Transrating of Coded Video Signals via Optimized Requantization. M.Sc. thesis, TECHNION, 2004.

[20] D. Lefol, D. Bull, and N. Canagarajah. Mode refinement algorithm for H.264 intra frame requantization. In *International Symposium on Circuits and Systems*, pages 4459–4462, 2006.

[21] D. Lefol, D. Bull, and N. Canagarajah. Performance evaluation of transcoding algorithms for H.264. *IEEE Transactions on Consumer Electronics*, 52(1):215–222, February 2006.

[22] D. Lefol, D. Bull, and N. Canagarajah. An efficient complexity-scalable video transcoder with mode refinement. *Signal Processing: Image Communications*, 22:421–433, April 2007.

[23] Z. Lei and N.D. Georganas. Rate adaptation transcoding for precoded video streams. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 127–136, Juan-les-Pins, France, December 2002.

[24] A. Leventer and M. Porat. On bit allocation in video coding and transcoding. In *IEE International Conference on Visual Information Engineering*, 2003.

[25] W.N. Lie and Y.H. Chen. Dynamic rate control for MPEG-2 bit stream transcoding. In *International Conference on Image Processing*, pages 477–480, 2001.

[26] S. Ma, W. Gao, and Y. Lu. Rate-distrotion analysis for H.264/AVC video coding and its application to rate control. *IEEE transactions on Circuits and Systems for Video Technology*, 15(12):1533–1544, December 2005.

[27] H.A. Mallot. *Computational Vision: Information Processing in Perception and Visual Behavior*, chapter 4, Edge Detection. MIT Press, 2000.

[28] A. Malvar et al. Low-complexity transform and quantization in H.264/AVC. *IEEE transactions on Circuits and Systems for Video Technology*, 13(7):598–603, July 2003.

[29] S. Milani, L. Celetto, and G.A. Mian. A rate control algorithm for the H.264 encoder. In *Sixth Baiona workshop on Signal Processing in Communications*, Spain, September 2003.

[30] M. Militzer, M. Suchomski, and K. Meyer-Wegener. Improved ρ-domain rate control and perceived quality optimizations for MPEG-4 real-time video applications. In *International Conference on Multimedia*, pages 402–411, 2003.

[31] K. Minoo and T.Q. Nguyen. Perceptual video coding with H.264. In *IEEE conference on Signals, Systems and Computers*, 2005.

[32] H.M. Nam et al. Low complexity H.264 transcoder for bitrate reduction. In *International Symposium on Communications and Information Technologies, ISCIT*, pages 679–682, Bangkok, Thailand, October 2006.

[33] A.G. Nguyen and J.N. Hwang. A novel hybrid HVPC/mathematical model rate control for low bit-rate streaming video. *Signal Processing: Image Communication*, 17:423–440, 2002.

[34] A. Ortega and K. Ramchandran. Rate-distortion methods for image and video compression. *IEEE Signal Processing Magazine*, 15:23–50, November 1998.

[35] A. Papoulis. *Probability, random variables, and stochastic processes*. McGraw-Hill, 2nd edition, 1986.

[36] M.H. Pinson and S. Wolf. A new standardized method for objectively measuring video quality. *IEEE Transactions on broadcasting*, 50(3):312–322, September 2004.

[37] T. Qian et al. Transform domain transcoding from MPEG-2 to H.264 with interpolation drift-error compensation. *IEEE transactions on Circuits and Systems for Video Technology*, 16(4):523–534, April 2006.

[38] J. Ribas-Corbera and S. Lei. Rate control in DCT video coding for low-delay communications. *IEEE transactions on Circuits and Systems for Video Technology*, 9(1):172–185, February 1999.

[39] I.E.G. Richardson. *Video codec design - developing image and video compression systems*. John Wiley, 2002.

[40] I.E.G. Richardson. *H.264 and MPEG-4 Video Compression*. John Wiley, 2003.

[41] K. Seo, S. Heo, and J. Kim. Adaptive rate control algorithm based on logarithmic R-Q model for MPEG-1 to MPEG-4 transcoding. *Signal Processing: Image Communication*, 17:857–875, 2002.

[42] I.H. Shin, Y.L. Lee, and H. Park. Rate control using linear rate-ρ model for H.264. *Signal Processing: Image Communications*, 19(4):341–352, April 2004.

[43] H. Sun, X. Chen, and T. Chiang. *Digital video transcoding for transmission and storage*. CRC press, 2005.

[44] H. Sun, W. Kwok, and J.W. Zdepski. Architectures for MPEG compressed bistream scaling. *IEEE transactions on Circuits and Systems for Video Technology*, 6(2):191–199, April 1996.

[45] W.T. Tan and B. Shen. Accurate distortion-driven macroblock level rate control via ρ-domain analysis. In *International Conference on Image Processing*, 2005.

[46] L. Torres and M. Kunt. *Video Coding: The second generation approach*, chapter 6, Segmentation-based motion estimation for second generation video coding techniques. Kluwer Academic Publishers, 1996.

[47] Y.K. Tu, J.F. Yang, and M.T. Sun. Efficient rate-distortion estimation for H.264/AVC coders. *IEEE transactions on Circuits and Systems for Video Technology*, 16(5):600–611, May 2006.

[48] A. Vetro, J. Cai, and C.W. Chen. Rate-reduction transcoding design for wireless video streaming. *Wireless Communications and Mobile Computing*, 2(6):625–641, October 2002.

[49] A. Vetro, C. Christopoulos, and H. Sun. Video transcoding architectures and techniques: an overview. *IEEE signal processing magazine*, 20(2):18–29, March 2003.

[50] A. Vetro, H. Sun, and Y. Wang. Object-based transcoding for adaptable video content delivery. *IEEE transactions on Circuits and Systems for Video Technology*, 11(3):387–401, March 2001.

[51] N. Wang and Y. He. A new bit rate control strategy for H.264. In *IEEE International Conference on Information, Communications and Signal Processing - PCM 2003*, pages 1370–1374, 2003.

[52] W. Wang, H. Cui, and K. Tang. Rate distortion optimized quantization for H.264/AVC based on dynamic programming. *Visual Communications and Image Processing, Proceedings of the SPIE*, 5960:2100–2111, July 2005.

[53] T. Wedi and H.G. Musmann. Motion and aliasing-compensated prediction for hybrid video coding. *IEEE transactions on Circuits and Systems for Video Technology*, 13(7):577–586, July 2003.

[54] T. Wiegand et al. Overview of the H.264/AVC video coding standard. *IEEE transactions on Circuits and Systems for Video Technology*, 13(7):560–576, July 2003.

[55] C.W. Wong et al. Novel H.26X optimal rate control for low-delay communications. In *IEEE International Conference on Information, Communications and Signal Processing - PCM 2003*, pages 90–94, 2003.

[56] M. Xia et al. Rate-distortion optimized bit allocation for error resilient video transcoding. In *International Symposium on Circuits and Systems, ISCAS*, 2004.

[57] W. Yuan, S. Lin, Y. Zhang, W. Yuan, and H. Luo. Optimum bit allocation and rate control for H.264/AVC. *IEEE transactions on Circuits and Systems for Video Technology*, 16(6):705–715, June 2006.

[58] M. Yuen and H.R. Wu. A survey of hybrid MC/DPCM/DCT video coding distortions. *Signal Processing*, 70:247–278, 1998.

[59] J. Zhang, A. Perkis, and N. D. Georganas. H.264/AVC and transcoding for multimedia adaptation. In *Proc. of the 6th workshop on information and knowledge management for integrated media communication*, Greece, May 2004.

[60] P. Zhang, Q.M. Huang, and W. Gao. Key techniques of bit rate reduction for H.264 streams. In *Lecture Notes in Computer Science, Book Advances in Multimedia Information Processing - PCM 2004*, pages 985–992. Springer, October 2004.

הורדת קצב מבוססת-מודל של וידאו מקודד

נעמה הייט

הורדת קצב מבוססת-מודל של וידאו מקודד

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר

מגיסטר למדעים

בהנדסת חשמל

נעמה הייט

הוגש לסנט הטכניון — מכון טכנולוגי לישראל

כסלו תשס"ח חיפה נובמבר 2007

המחקר נעשה בהנחיית פרופ' דוד מלאך

בפקולטה להנדסת חשמל

הכרת תודה

תודתי העמוקה לפרופ' דוד מלאך על הנחייתו והדרכתו המסורה לאורך כל תקופת המחקר. אני רוצה להודות גם לצוות המעבדה לעיבוד אותות ותמונות (SIPL) על העזרה והתמיכה הטכנית. תודה מיוחדת למשפחתי ולבעז על התמיכה והעידוד.

המחקר מומן חלקית ע"י קונסורציום STRIMM במסגרת תוכנית מגנ"ט של לשכת המדען הראשי בתמ"ת דרך מוסד שמואל נאמן למחקר מתקדם במדע וטכנולוגיה וע"י המעבדה לעיבוד אותות ותמונות.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי

תוכן עניינים

רשימת טבלאות

רשימת איורים

תקציר

שרתי וידאו ויישומי מולטימדיה משתמשים בוידאו מקודד בפורמטים שונים לאחסון ושידור. בדרך כלל, בשרת מאוחסן עותק יחיד של הוידאו המקודד באיכות גבוהה, בעוד שלמשתמשי קצה שונים יש דרישות פורמט וקצב שונות. לכן, אות הוידאו המקודד באיכות הגבוהה מומר בזמן אמת כדי לעמוד בדרישות השונות של משתמשי הקצה. הורדת קצב (Transrating) של אות וידאו מקודד באותו פורמט יכולה להתבצע במספר גישות, כגון הורדת הרזולוציה המרחבית, הורדת הרזולוציה הזמנית ורה-קוונטיזציה של מקדמי ההתמרה. בעבודה זו, בחנו הורדת קצב מבוססת-מודל בגישה של רה-קוונטיזציה, בתקן H.264 החדש.

מטרת הורדת הקצב הינה להשיג את הקצב הרצוי, בסיבוכיות חישוב נמוכה, תוך שמירה על איכות וידאו גבוהה באות המוצא. הפתרון הנאיבי הוא לבצע פענוח מלא וקידוד מחדש של רצף התמונות המפוענח. החסרון של פתרון זה הוא הסיבוכיות החישובית הגבוהה הכרוכה בקידוד מח-דש, למשל בשערוך מחדש של וקטורי התנועה. לכן, הגישה המקובלת היא לבצע רה-קוונטיזציה תוך חזרה על החלטות הקידוד הקודמות או ביצוע חיפוש מצומצם של החלטות חדשות המתבסס על ההחלטות הקודמות.

עבודות קודמות על רה-קוונטיזציה, בתקני קידוד וידאו קודמים, מציעות למצוא את סט הצעדים החדשים האופטימליים, כלומר אלו שישיגו את הקצב הנדרש תוך יצירת עיוות מינימלי, בעזרת אופטימיזציה לגרנג'יאנית. על מנת להחליט מהם הצעדים האופטימליים, יש לבחון מספר רב של צעדים באיזורי תמונה שונים. בהעדר ידע מוקדם, בחינת הקצב המתקבל לאיזור תמונה כתוצאה מקוונטיזציה עם צעד נתון, כרוכה בסימולציה של הקוונטיזציה מחדש וקידוד אנטרופיה של המקדמים המתקבלים ממנה. חזרה על התהליך עבור צירופים רבים עלולה להאריך מאוד את זמן החישוב. ניתן לייעל את החיפוש על ידי שימוש במודלים אנליטיים הקושרים בין הקצב לבין צעד הקוונטיזציה. קיימים בספרות מספר מודלים לקצב עבור קידוד תמונות ואותות וידאו.

בעבודה זו, נבחן מודל פרמטרי הקושר בין הקצב לבין אחוז האפסים במקדמי ההתמרה לאחר הקוונטיזציה. למציאת הקשר המלא בין הקצב לצעד, יש לאפיין גם את הקשר בין אחוז האפסים לבין צעד הקוונטיזציה.

מקודד ה- H.264 תומך באפשרויות קידוד מתקדמות, כגון חיזוי מרחבי תוך תמונתי וקידוד אנטרופיה אדפטיבי ותלוי תוכן, במחיר של סיבוכיות חישוב גבוהה. אפשרויות קידוד אלו יוצרות הבדלים מהותיים לעומת התקנים הקודמים, ולכן יש צורך להתאים את האלגוריתמים הקיימים להורדת קצב למקודד החדש. עבודות קודמות על הורדת קצב ב- H.264 מתמקדות בהתאמת החלטות הקידוד לקצב הנמוך והרה-קוונטיזציה בהן מתבצעת במעבר יחיד על הבלוקים בתמונה, שאינו אופטימלי.

בעבודה זו, פותחו ונבחנו אלגוריתמי רה-קוונטיזציה אופטימלית מבוססת מודל, המותאמים לתקן H.264. שילוב המודלים הקושרים בין הקצב לבין אחוז האפסים ובין אחוז האפסים לצעד, כחלק מבקרת הקצב נועד לשרת שתי מטרות. ראשית, להקל על העומס החישובי הכרוך בתהליך חיפוש הצעדים האופטימליים. שנית, לאפיין את הקשר בין אחוז האפסים לבין צעד הקוונטיזציה בחוג סגור.

על מנת להוריד את הקצב הממוצע של אות הוידאו המקודד, יש למצוא את הקצאת הסיביות לכל מסגרת. הגישה הפשוטה ביותר הינה להוריד את הקצב של כל המסגרות באותו היחס, וכך לעמוד באילוץ הקצב הממוצע. אך, בשל אפשרויות הקידוד המתקדמות בתקן H.264, התקורה בקידוד יכולה להגיע לעשרות אחוזים מתוך סך הסיביות המוקצות לקידוד המסגרת, וזאת על חשבון הסיביות המוקצות לקידוד מקדמי ההתמרה. בעבודה זו, ממשנו הקצאת סיביות אופטימלית בתוך קבוצת מסגרות, המשיגה את הקצב הנדרש במוצא, תוך מזעור סך העוות במסגרות. לקבלת איכות וידאו חלקה ונעימה לעין, נוסף אילוץ של השוואת העוות בין המסגרות השונות. המסגרת הראשונה בכל קבוצה היא מסגרת מסוג Intra, אחריה מקודדות מסגרות מסוג Inter, הנחזות זמנית ממסגרות מקודדות קודמות באותה הקבוצה. בשל האפיון השונה של מסגרות מסוג Intra ומסוג Inter, פותחו עבורן אלגוריתמי רה-קוונטיזציה שונים.

במסגרות מסוג Intra, נוסף בתקן H.264 חיזוי מרחבי תוך תמונתי כלומר, בלוק תמונה נחזה מתוך ערכי פיקסלים בבלוקים שכנים שכבר קודדו. החיזוי המרחבי גורם לתלות בין אותות השאריות המקודדים בבלוקים סמוכים. כדי למנוע שגיאה נגררת נראית לעין, יש לפענח את המסגרת

בצורה מלאה ואז לקודד אותה מחדש. הקידוד מחדש הינו מודרך, כלומר נעזר באופני החיזוי של המסגרת בכניסה. אפשרות אחת היא לחזור על כל אופני החיזוי המקוריים ולהשיג חסכון של פי 4 בחישובים. אפשרות שנייה היא לשנות את האופנים בצורה סלקטיבית, רק באותם מקומות בהם צפוי רווח ביעילות הקידוד, כלומר איכות טובה יותר עבור אותו קצב. שינוי האופנים הסלקטיבי משיג את אותה האיכות של חיפוש אופנים מלא בקצב נתון, תוך חסכון ביחס של פי 1.6 בחישובים.

הורדת הקצב העיקרית במסגרות מסוג Intra מתבצעת על ידי רה-קוונטיזציה אחידה של מקדמי ההתמרה, כלומר בחירת אותו צעד רה-קוונטיזציה עבור כל הבלוקים במסגרת. בחירת הצעד נעזרת במודל המוצע בספרות לקשר בין הקצב לבין אחוז האפסים לאחר הקוונטיזציה. על מנת לשערך את הקשר בין אחוז האפסים לבין צעד הרה-קוונטיזציה, פותח מודל חדש בחוג סגור, המאפיין את אות התיקון הנדרש למקדמי ההתמרה של אות השארית. ראשית, מחולק אות התיקון לקבוצות הומוגניות זרות, הנוחות יותר למידול. לאחר מכן, מאופיינת כל קבוצה בעזרת פילוג סטטיסטי, כאשר פרמטר הפילוג מותאם על פי מידע מתוך קידוד המסגרת בכניסה. המודל המוצע מאפשר דיוק של 3% בקצב הרצוי במסגרת, לעומת סטייה של 10.8% בקצב כתוצאה משערוך אחוז האפסים בחוג פתוח.

במסגרות מסוג Inter, אנו מציעים לחזור ולהשתמש בוקטורי התנועה שבכניסה. המסגרת בכניסה מפוענחת בצורה חלקית, עד למקדמי ההתמרה של אות השארית, ואז מקודדת בצורה חלקית, לאחר תיקון בחוג סגור של המקדמים. צעדי הרה-קוונטיזציה נבחרים בצורה אופטימלית, תוך שימוש במודלים לקצב ולעוות כפונקציה של אחוז האפסים. כדי לשמור על איכות תמונה חלקה ונעימה לעין, מוצע בספרות להגביל את מידת השינוי של צעד הקוונטיזציה בתוך המסגרת. אנו מציעים להרחיב את האופטימיזציה הלגרנג'יאנית באופן הבא. בכל איטרציה, בה נקבע המשקל היחסי של הקצב לעומת העיוות בפונקציית המחיר, מופעל אלגוריתם של תכנות דינמי למציאת סט הצעדים שממזער את פונקציית המחיר, תוך הגבלת מידת שינוי הצעד בין בלוק לבלוק. בסיום כל איטרציה, נבדק הקצב המושג על ידי צירוף הצעדים הנבחר לעומת הקצב הרצוי למסגרת, והמשקל היחסי מעודכן בהתאם.

מתוך בחינה של המודלים המוצעים בספרות לקצב ולעוות כפונקציה של אחוז האפסים, מצאנו שהם אינם מתאימים ברזולוציה של בלוק ב- H.264. לכן, אנו מציעים מודלים חדשים, המותאמים לרה-קוונטיזציה ברמת הבלוק ב- H.264. פרמטרי המודלים משוערכים מתוך המידע המקודד במסגרת בכניסה. שילוב המודלים המוצעים מביא לדיוק של 4.5% בקצב הנדרש לעומת 24.7%, כשמשתמשים במודלים המוצעים בספרות.

סך הכל, המערכת המוצעת מקצרת את זמן החישוב פי 4, בהשוואה לגישה הנאיבית של פענוח-קידוד מלא, במחיר של 1.4[dB] ב- PSNR. לעומת רה-קוונטיזציה פשוטה במעבר יחיד על הבלוקים בתמונה, המערכת המוצעת משיגה ביצועים טובים יותר הן אובייקטיבית (רווח של עד 1.6[dB] ב-PSNR) והן סובייקטיבית, במחיר של הכפלת סיבוכיות החישוב.