Bitsnelheidstranscodering van H.264/AVC gebaseerd op bitsnelheidsschaling en herkwantisatie

Bit Rate Transcoding of H.264/AVC Based on Rate Shaping and Requantization

Stijn Notebaert

Promotoren: prof. dr. ir. R. Van de Walle, dr. P. Lambert
Proefschrift ingediend tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen: Computerwetenschappen

Vakgroep Elektronica en Informatiesystemen
Voorzitter: prof. dr. ir. J. Van Campenhout
Faculteit Ingenieurswetenschappen
Academiejaar 2009 - 2010

UNIVERSITEIT
GENT

ISBN 978-90-8578-357-2
NUR 965
Wettelijk depot: D/2010/10.500/33

Dankwoord

Eindelijk, het is zover! De laatste hand aan dit proefschrift is gelegd. Het was een boeiende en leerrijke ervaring. Maar dit was me nooit alleen gelukt. En daarom wil ik de mensen bedanken die hiertoe hebben bijgedragen.

Allereerst bedank ik mijn promotor Rik Van de Walle voor de kans die hij mij gegeven heeft om onderzoek te verrichten in zijn onderzoeksgroep. Op die manier heb ik de fascinerende wereld van signaalcodering en signaalverwerking ontdekt. Het is ook dankzij hem dat ik meermaals mijn werk heb kunnen voorstellen op internationale conferenties en dat ik heb mogen kennismaken met standaardisatie binnen MPEG en VCEG.

Ik bedank ook alle medewerkers van de onderzoeksgroep Multimedia Lab. Jullie hebben steeds gezorgd voor een aangename sfeer, zowel tijdens als na de werkuren. Tenslotte wil ik de collega's bedanken die delen van deze tekst hebben nagelezen. Discussies en gesprekken hebben dit doctoraatswerk sterk verbeterd.

In het bijzonder bedank ik Jan De Cock. Samen hebben we de eerste stappen gezet in de wereld van de transcodering. Het was echt aangenaam om met jou samen te werken. Veel gesprekken zijn gestart achter het bureau of aan het bord, en deze zijn ons blijven achtervolgen tot op de fiets of tussen pint en pita.

Ik bedank ook Ellen Lammens en Rita Breems voor de administratieve hulp. Jullie waren steeds bereid om alle praktische zaken zoals conferenties, vluchten en hotels te regelen. Dat heeft er voor gezorgd dat ik mij meer kon toeleggen op het onderzoek.

Ik bedank ook iedereen waarmee ik heb samengewerkt in projecten. In het bijzonder bedank ik Samie Beheydt en Jan De Lameillieure van Cisco voor de aangename samenwerking tijdens het bilateraal project.

Furthermore I would like to thank the members of the jury, Patrick Bergmans, Jan De Lameillieure, Bart Dhoedt, Peter Lambert, Adrian Munteanu, Aleksandra Pizurica, Rik Van de Walle, Ronny Verhoeven, and Mathias Wien, for the remarks and suggestions which improved the final version of this book.

Een ongelooflijke dankjewel aan Eveline en Geert, Sofie en Daniel, Nele en Bram, Sofie en Bart, en Joris en Kim. Jullie toonden geregeld interesse in het onderzoek hoewel dit niet bepaald jullie domein is. Jullie vroegen steeds weer opnieuw naar de verdediging van het proefschrift.

Een speciaal woord van dank gaat uit naar mijn schoonouders Rita en Yvan. Bedankt voor jullie steun en hulp, jullie vriendschap doet echt deugd. Jullie deur staat altijd voor mij open.

Beste mama en papa, ik bedank jullie voor de ontelbare kansen die jullie mij gegeven hebben. Jullie liefde en toewijding hebben mij gemaakt tot wie ik nu ben. Bedankt voor de bemoedigende woorden en de liefdevolle schouderklopjes maar vooral voor de mooie momenten in het leven.

Liefste Eva, helemaal onverwacht maar aangenaam verrast kwam jij in mijn leven. Jij bent mijn steun en toeverlaat. Ik bedank je voor alles wat je mij gegeven hebt. Zonder jou was dit nooit gelukt. Daarom draag ik dit werk aan jou op. Ik zie je ongelooflijk graag.

Stijn Notebaert
juni 2010

Summary

During the past decades, different multimedia applications have been developed from streaming video over the Internet and digital interactive television to mobile multimedia communication. The video component is very important for these applications: the visual quality for the users, the bit rate for the networks, and the decoding for the devices.
Efficient coding of digital video is therefore required. There has been put a lot of effort in order to improve the coding efficiency. Often different techniques are combined such as predictive and transform coding. This class of algorithms is often called *hybrid video coding*. The purpose of predictive coding is to decorrelate adjacent pixel values which results in a prediction error, while the purpose of transform coding is to represent the prediction error with as few bits as possible.
This coding design is frequently used for video coding standards, such as the video coding standards developed by ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG). The standards which are jointly developed by these standardization bodies are very successful. The H.262/MPEG-2 Video standard is used as coding format for digital television and DVD-Video. The H.264/AVC standard is used for a broader range of applications such as video over Internet (video streaming, download and play), interactive applications (real-time videoconferencing), digital television (cable and satellite broadcast), video storage (Blu-ray Disc, digital cinema), and professional applications (editing, archiving).
The number of networks and terminals has increased exponentially over the last years. Different transport streams are sent over the network and these streams compete with each other for bandwidth. In some cases, the bandwidth is not sufficient for transmitting the video bitstream at full quality. As a result, an adaptation of the bit rate of the video bitstream is required. Also the functionality of the terminals has evolved: sending multimedia messages, taking pictures and making videos, satellite navigation, surfing the Web, e-mail, agenda, contacts, ... The terminals have their own specifications and it is possi-

ble that the properties of video bitstreams are not in line with the specifications. As a result, the properties of the video bitstream need to be adapted in order to allow decoding and displaying.
When an adaptation of the video bitstream is required, a transcoder can be used. A transcoder is a device that changes the properties of video bitstreams in an efficient way. In this work, we focus on the trancoders that only reduce the bit rate of the video bitstreams. This class of transcoders is often called *transraters*.
Transrating is defined as the operation that reduces the bit rate of video bitstreams, while other characteristics of the video bitstreams are left unchanged. Different methods for transrating have been proposed in the literature: *rate shaping* and *requantization*. These methods are based on decreasing the amount of residual data and are extensively used in the past for reducing the bit rate. In this work, rate shaping and requantization are investigated for H.264/AVC.

Rate shaping is a technique that copies transform coefficients from the incoming video bitstream to the outgoing video bitstream. The breakpoint determines what transform coefficients are removed. This way, the amount of residual data is decreased which results in a video bitstream with reduced bit rate. Rate shaping is often used as a technique with low complexity. In Chapter 2, we investigate rate shaping for H.264/AVC.
Removing transform coefficients was an easy operation for H.262/MPEG-2 Video bitstreams. This results from the construction of the entropy coding. This is not the case for H.264/AVC bitstreams due to the increased complexity of the entropy coding. The coefficients are represented by different syntax elements which code the sign, the amplitude, and the position of a transform coefficient. These syntax elements are coded dependently, so it is impossible to operate on the video bitstream directly. Entropy decoding is required at the input of the transrater in order to operate on the transform coefficients. The transform in H.264/AVC is more extensive when compared to the transform in H.262/MPEG-2 Video. Different transform schemes are provided. Some of these transform schemes consist of two successive transforms. The first transform operates on the residual data, while the second transform operates on the low-frequency transform coefficients of the first transform. As a result, the second transform only operates on low-frequency data. Since the human visual system is very sensitive to low-frequency data, we decide not to change these coefficients.
Using the extensions for entropy coding and transform, H.264/AVC rate shaping is feasible. In this context, we evaluate the performance of H.264/AVC rate shaping. The results show that significant drift is found. The spatial drift

is clearly visible in the I pictures. Spatial and temporal drift is found in P and B pictures; however, they have less impact on the distortion in the video. As a result, we decide to apply rate shaping only to P and B pictures. In order to improve the visual quality of these pictures, we compare different rate shaping solutions with and without compensation. The compensation technique was first used for requantization where we compensate for the requantization error in the reference. When compensation is applied, the rate shaper is more complex; however, this results in better visual quality. The rate shaping results show a significant gain when compensation is used. The gains depend on the characteristics of the video bitstreams. The spatial compensation is better than temporal compensation when visual quality and computational resources are considered.
Finally, an overview of bit allocation algorithms is presented in the context of H.264/AVC rate shaping. The rate-distortion optimal solution is far too complex, so a low-complexity alternative is more practical. A suitable solution is clustering which defines one breakpoint for a group of blocks. This solution was also proposed for rate shaping of H.262/MPEG-2 Video.

Requantization is a technique that applies a coarser quantizer in the transrater. This way, the amount of residual data is decreased which results in a video bitstream with reduced bit rate. An important problem for requantization deals with the quantizer design in the transrater. The selection of an appropriate quantizer for the transrater is called the *requantization problem*. Most solutions are based on the requantization error which results from the successive quantization. The requantization error results from the transrater which has only access to the already quantized transform coefficients instead of the original transform coefficients.
A technique called *perfect requantization* was proposed in the literature which eliminates the requantization errors by choosing an appropriate quantizer. The main drawbacks of this approach are the coarse quantization in the transrater and the additional information that is required in the transrater. The coarse quantization restricts the flexibility for transrating which is not advantageous for the performance of the transrater.
We present a different approach in Chapter 3. We derive a transrating heuristic based on theoretical rate-distortion results of the Laplace source which is quantized twice. In order to simplify the calculations for entropy and distortion, we provide a solution based on the characteristics of the Laplace source and the effective quantizer. The effective quantizer is derived as the superposition of the quantizers in encoder and transrater and generates the same output as successively applying these quantizers. The memoryless property from the Laplace source is combined with the periodic property of the effective quan-

tizer in order to derive expressions for entropy and distortion.
The transrating heuristic is derived from the theoretical results for different quantization and requantization. We found that increasing the quantizer step size with fixed quantizer offset is a good solution for fine quantization in the encoder. When the quantization in the encoder is coarse, a better solution consists of decreasing the quantizer offset with fixed quantizer step size. The transrating heuristic was implemented in transrating software for H.264/AVC and is applied to open-loop transrating of B pictures. We found gains in visual quality (more than 1 dB for certain configurations) compared to transrating with fixed quantizer step size.

The selection of an appropriate architecture for requantization transrating may even have more impact on the transrating performance. The performance and the complexity of transrating are mainly determined by the architecture. In the past, many transrating solutions have been presented that apply one transrating method to all blocks in the video bitstream. These solutions often do not meet transrating requirements such as fast and efficient transrating. In Chapter 4, we make an evaluation of transrating methods in the context of H.264/AVC and we propose mixed architectures which select an appropriate transrating method.
We start with an investigation of the basic transrating architectures in the context of H.264/AVC: the open-loop transrater, the fast pixel-domain transrater, and the cascaded pixel-domain transrater. The open-loop transrater has low-complexity, but introduces spatial and temporal drift. The drift degrades the visual quality which makes this transrating architecture not useful. The fast-pixel domain transrater compensates for the requantization errors; however, the prediction loop adds extra complexity and introduces small rounding errors. The cascaded pixel-domain transrater maintains two prediction loops which results in drift-free transrating while the complexity is increased compared to open-loop transrating and fast pixel-domain transrating. We found that none of these architectures meet the transrating requirements.
Finally, we propose mixed transrating architectures which apply different transrating techniques depending on the picture/macroblock type. The cascaded pixel-domain transrater is used for I pictures, so drift-free transrating is applied. The open-loop transrater or the fast-pixel domain transrater are selected for P and B pictures. The spatial compensation shows a significant improvement which in addition requires slightly more resources. The temporal compensation results in minor improvements with more need for processing power and memory buffers.

Samenvatting

De voorbije decennia zijn tal van nieuwe applicaties ontwikkeld die gebruikmaken van digitale video, van videostreaming over het internet en digitale interactieve televisie tot mobiele multimediacommunicatie. Voor deze applicaties is de videocomponent van groot belang: de visuele kwaliteit voor de gebruikers, de bandbreedte voor de netwerken en het decoderen voor de terminals.

Een efficiënte codering van digitale video is onontbeerlijk voor deze applicaties. Er zijn heel wat inspanningen geleverd om de codering van digitale video te verbeteren. Dit heeft geleid tot behoorlijke winsten in codeerefficiëntie. Algoritmen voor de codering van digitale video combineren vaak verschillende technieken, bijvoorbeeld predictieve codering en transformatiecodering. Deze klasse van algoritmen wordt wel eens *hybride videocodering* genoemd. De predictieve codering zorgt voor een decorrelatie van naburig pixelwaarden. Het resulterende verschilsignaal wordt dan efficiënter gecodeerd gebruikmakende van transformatiecodering.

Dit codeerschema is vaak het ontwerp dat gebruikt wordt in standaarden voor videocodering, zoals de standaarden die werden ontwikkeld door ISO/IEC Moving Picture Experts Group (MPEG) en ITU-T Video Coding Experts Group (VCEG). Vooral de standaarden die deze standaardisatiecommissies gezamenlijk hebben ontwikkeld, kennen een groot succes. De H.262/MPEG-2 Video-standaard wordt vaak gebruikt als codeerformaat voor digitale televisie en DVD-Video. De H.264/AVC-standaard kan ingezet worden voor een breder spectrum aan applicaties zoals video over internet (videostreaming, downloaden en afspelen), interactieve applicaties (realtime videoconferentie), digitale televisie (uitzendingen via kabel of satelliet), opslag van video (Blu-ray Disc, digitale cinema) en professionele applicaties (editeren, archiveren).

De afgelopen jaren is de diversiteit aan netwerken en terminals sterk toegenomen. Verschillende videostromen worden over het netwerk verstuurd. Deze videostromen dingen mee naar de optimale bandbreedte. Als er op het netwerk niet voldoende bandbreedte beschikbaar is, dan kan de videostroom

niet verzonden worden aan volle kwaliteit. Een aanpassing van de bitsnelheid van de videostroom dringt zich op. Ook de functionaliteit van terminals is de laatste jaren sterk geëvolueerd: verzenden van multimediaberichten, vastleggen van foto en film, satellietnavigatie, gebruik van internet, elektronische post, agendabeheer, contactbeheer ... De terminals zijn gebonden aan hun specificaties. Het is dus mogelijk dat de beeldsnelheid of de beeldresolutie van de videostromen niet overeenstemmen met de specificaties van de terminals. Ook hier is een aanpassing van de eigenschappen van de videostroom noodzakelijk om het decoderen mogelijk te maken.
Als er een aanpassing aan de videostroom nodig is, dan kan dat gerealiseerd worden door een transcoder. De transcoder is een apparaat dat op efficiënte wijze een videostroom aanpast aan de noden van het netwerk of de terminal. In dit werk beperken we ons tot de klasse van transcoders die enkel de bitsnelheid verlagen. Deze klasse wordt vaak aangeduid als *transraters*.
Transrating is een techniek om de bitsnelheid van videostromen te verlagen, terwijl de andere karakteristieken van de videostromen onveranderd blijven. Verschillende technieken voor transrating werden voorgesteld in de literatuur: *bitsnelheidsschaling* en *herkwantisatie*. Deze methoden verminderen de residuele data in de videostroom en werden in het verleden vaak gebruikt voor het verlagen van de bitsnelheid. In dit proefschrift wordt bitsnelheidsschaling en herkwantisatie voor H.264/AVC onderzocht.

Bitsnelheidsschaling is een techniek die coëfficiënten kopieert van de inkomende videostroom naar de uitgaande videostroom. Het breekpunt bepaalt hoeveel coëfficiënten verwijderd worden. Op deze manier wordt de residuele data gereduceerd en ontstaat er een videostroom met een lagere bitsnelheid. Bitsnelheidsschaling wordt vooral gebruikt als een techniek met lage complexiteit. We onderzoeken bitsnelheidsschaling in de context van H.264/AVC in Hoofdstuk 2.

Het ontwerp van entropiecodering is heel eenvoudig bij H.262/MPEG-2 Video waardoor op eenvoudige wijze coëfficiënten verwijderd kunnen worden. Dit is niet geval bij H.264/AVC door de toegenomen complexiteit van de entropiecodering. De coëfficiënten worden gecodeerd met behulp van codewoorden die het teken, de grootte en de positie van de coëfficiënt voorstellen. Tussen de codewoorden bevinden zich heel wat afhankelijkheden waardoor het niet mogelijk is om direct op de bitstroom in te werken. Entropiedecodering is dus noodzakelijk om in te werken op de residuele data. Ook het schema voor transformatie is ingewikkelder geworden. Er worden verschillende schema's voor transformatie voorzien. Sommige daarvan bestaan zelfs uit twee opeenvolgende transformaties waarbij de tweede transformatie inwerkt op de laagfre-

quente coëfficiënten van de eerste transformatie. Het resultaat na de tweede transformatie bevat enkel laagfrequente informatie. Onze ogen zijn gevoeliger voor deze frequenties waardoor we besluiten om deze coëfficiënten ongewijzigd te laten. We werken enkel in op de hoogfrequente coëfficiënten uit de eerste transformatie.
Met de nodige extensies is het mogelijk om H.264/AVC-bitstromen te herschalen. We evalueren de prestatie van bitsnelheidsschaling van H.264/AVC. De resultaten tonen dat er significante drift aanwezig is. De spatiale drift in I-beelden is nadrukkelijk aanwezig. Spatiale en temporele drift komen voor in P- en B-beelden, maar zijn veel minder opvallend aanwezig. Aangezien de spatiale drift in I-beelden zeer uitdrukkelijk aanwezig is, lijkt het ons geschikt om de I-beelden niet te herschalen. We passen dus enkel schaling toe op de P- en B-beelden. Om de kwaliteit van deze beelden te behouden vergelijken we verschillende algoritmen voor schaling met en zonder compensatie. De compensatie werd voor het eerst gebruikt bij herkwantisatie waar gecompenseerd wordt voor de herkwantisatiefout in het referentieblok. Als compensatie wordt toegepast dan neemt de complexiteit van de bitsnelheidsschaler toe, maar gaat de visuele kwaliteit van de aangepaste videostroom omhoog. De winsten hangen af van de karakteristieken van de videostromen. De spatiale compensatie is beter dan de temporele compensatie indien visuele kwaliteit en complexiteit in acht worden genomen.
We geven tenslotte nog een overzicht van verschillende algoritmen voor bitallocatie. De optimale oplossing is veel te complex waardoor een alternatief met lagere complexiteit meer geschikt is vanuit praktisch standpunt. Een geschikte oplossing is clustering waarbij één breekpunt wordt gebruikt voor een groep van blokken. Deze oplossing werd ook voorgesteld voor bitsnelheidsschaling van H.262/MPEG-2 Video.

Herkwantisatie is een techniek die een grovere kwantisatie in the transrater toepast. Op deze manier wordt de residuele data gereduceerd en ontstaat er een videostroom met lagere bitsnelheid. Een belangrijk probleem bij herkwantisatie handelt over de karakteristiek van de kwantiser in the transrater. De selectie van een geschikte kwantiser in de transrater wordt vaak bestempeld als het *herkwantisatieprobleem*. De meeste oplossingen zijn gebaseerd op de herkwantisatiefout die voortvloeit uit de opeenvolgende kwantisatie in encoder en transrater. De herkwantisatiefout ontstaat omdat de transrater enkel beschikt over de gekwantiseerde coëfficiënten en niet over de originele coëfficiënten.
In de literatuur wordt *perfecte herkwantisatie* voorgesteld als een techniek waarbij geen herkwantisatiefouten meer optreden. Dit wordt gerealiseerd door een aangepaste kwantiser voor de transrater te selecteren gebaseerd op de eigenschappen van de kwantiser in de encoder. Hoewel geen herkwanti-

satiefouten meer optreden, brengt het wel een aantal nadelen met zich mee. Zo zal steeds een grove kwantisatie worden toegepast in vergelijking met de kwantiser in de encoder. Dit beperkt aanzienlijk het aantal mogelijkheden voor de herkwantisatie en het stemt vaak niet overeen met de eisen opgelegd door het netwerk of de terminal. Er moet dan ook nog eens bijkomende informatie over de kwantiser verzonden worden van encoder naar transrater. Zonder deze informatie is het niet mogelijk om perfecte herkwantisatie toe te passen.
In Hoofdstuk 3 stellen we een alternatieve methode voor. We leiden een heuristiek af gebaseerd op theoretische rate-distortion resultaten voor een Laplace-bronmodel die twee keer wordt gekwantiseerd. De berekeningen van entropie en distortie worden vereenvoudigd door gebruik te maken van karakteristieken van het Laplace-bronmodel en de effectieve kwantiser. De effectieve kwantiser wordt gezien als de superpositie van de kwantisers in encoder en transrater en genereert hetzelfde resultaat als de opeenvolgende toepassing van de kwantisers. De geheugenloze eigenschap van het Laplace-bronmodel wordt gecombineerd met de periodieke eigenschap van de effectieve kwantiser om vergelijkingen voor de entropie en de distortie af te leiden.
De heuristiek wordt afgeleid van de theoretische resultaten voor verschillende configuraties van encoder en transrater. We stellen vast dat het verhogen van de stapgrootte met een vaste verschuiving een goede oplossing is voor een fijne kwantiser in de encoder. Als de kwantiser in de encoder grover wordt, dan is het beter om de verschijving te laten afnemen met een vaste stapgrootte. De heuristiek is geïmplementeerd in software voor transrating van H.264/AVC en wordt toegepast op open-loop transrating van B-beelden. Dit geeft aanleiding tot winsten in visuele kwaliteit (tot meer dan 1 dB voor bepaalde configuraties) in vergelijking met transrating met een vaste verschijving.

Minstens even belangrijk is de keuze welke architectuur gebruikt wordt voor transrating. De keuze bepaalt in grote mate de visuele kwaliteit van de aangepaste video en de complexiteit van de transrater. In het verleden werden vaak oplossingen voorgesteld die slechts één architectuur gebruiken voor alle blokken in de videostroom. Deze architectuur komt niet tegemoet aan de eisen van snelle en efficiënte transrating. In Hoofdstuk 4 evalueren we verschillende technieken voor transrating in de context van H.264/AVC en stellen we gemengde architecturen voor die op macroblokniveau een geschikte architectuur selecteren.
We beginnen met een evaluatie van de basisarchitecturen in de context van H.264/AVC: open-loop transrating, snelle transrating in het pixeldomein en gecascadeerde transrating in het pixeldomein. De open-loop transrater heeft lage complexiteit maar introduceert spatiale en/of temporele drift. De drift degradeert de visuele kwaliteit waardoor deze architectuur nagenoeg niet bruik-

baar is. De snelle transrater in het pixeldomein compenseert de herkwantisatiefout. Dit is mogelijk door gebruik te maken van een extra predictielus die aanleiding geeft tot een hogere complexiteit en waardoor afrondingsfouten ontstaan. De gecascadeerde transrater in het pixeldomein heeft twee predictielussen en is per definitie vrij van drift. De complexiteit is toegenomen in vergelijking met de open-loop transrater en de snelle transrater in het pixeldomein. Geen van deze architecturen voldoet aan de eisen van snelle en efficiënte transrating.
Tenslotte stellen we gemengde architecturen voor die een techniek selecteren die afhangt van het beeldtype en het macrobloktype. De gecascadeerde transrater in het pixeldomein wordt geselecteerd voor I-beelden. Op die manier wordt geen drift geïntroduceerd en vertoont het I-beeld enkel verminderde kwaliteit door herkwantisatie. Voor de P- en B-beelden wordt open-loop transrating of snelle transrating in het pixeldomein geselecteerd. De spatiale compensatie heeft een behoorlijke verbetering van de visuele kwaliteit zonder echter de complexiteit aanzienlijk te verhogen. De temporele compensatie kan de visuele kwaliteit nog verbeteren maar vraagt meer rekenkracht en geheugen in vergelijking met spatiale compensatie.

List of abbreviations

AVC	Advanced Video Coding
CABAC	context-adaptive binary arithmetic coding
CAVLC	context-adaptive variable-length coding
CIF	Common Intermediate Format
CPDT	Cascaded Pixel-Domain Transcoder
DCT	discrete cosine transform
DF	Deblocking Filter
DFD	Displaced Frame Difference
DZ+UTQ	dead-zone plus uniform threshold quantization
RS	Rate Shaping
FPDT	Fast Pixel-Domain Transcoder
fps	frames per second
FRExt	Fidelity Range Extensions
GOP	group of pictures
HD	high definition
IBBT	Interdisciplinary Institute for Broadband Technology
IDR	Instantaneous Decoding Refresh
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
ITU	International Telecommunication Union
JM	Joint Model
JPEG	Joint Photographic Experts Group
JVT	Joint Video Team
MBAFF	macroblock adaptive frame/field
Mbps	Megabits per second
MPEG	Moving Picture Experts Group
MVC	Multi-view Video Coding
NAL	network abstraction layer
NURQ	nearly-uniform reconstruction quantizer
OL	Open-loop
PAFF	picture adaptive frame/field
pdf	probability distribution function
pmf	probability mass function
PSNR	peak signal-to-noise ratio

QoS	Quality of Service
QP	Quantization Parameter
RDO	rate-distortion optimization
SAD	sum of absolute differences
SD	standard definition
SNR	Signal-to-Noise Ratio
SSD	sum of squared differences
SVC	Scalable Video Coding
URQ	uniform reconstruction quantizer
VCEG	Video Coding Experts Group
VLC	Variable Length Code
WMV	Windows Media Video

Contents

Chapter 1

Introduction

During the last decades, the number of multimedia applications is growing explosively, from streaming video over the Internet [1] and mobile interactive applications to digital television broadcasting [2]. These applications typically use different kinds of media sources such as text, images, audio, and video. The video component has a major impact on two important aspects of these applications. Firstly, the bit rate of the video bitstream is important and has an impact on the data traffic on the network. Secondly, the visual quality of the video is important and has an impact on the user experience.

Need for compression Uncompressed video produces an enormous amount of data and is extremely inefficient for transmission and storage in most applications. For example, 720p HD video (a resolution of 1280×720 pixels) with a frame rate of 60 frames per second (fps) in YUV 4:2:0 format with 8 bits color depth requires about 660 Mbps. The uncompressed video contains spatial and temporal redundancy which can be exploited by a variety of compression techniques. The compression techniques determine the required bit rate and the visual quality and are therefore important for many applications.
A popular design for video compression combines predictive and transform coding techniques which is often denoted as *hybrid video coding* [3]. The purpose of predictive coding is to decorrelate adjacent pixel values and only code the prediction error, while the purpose of transform coding is to describe the prediction error with as few bits as possible.
A lot of effort has gone to improving this design which has led to significant gains in compression efficiency over the last decades and the standardization of two families of video compression algorithms: the MPEG-x standards developed by ISO/IEC Moving Picture Experts Group (MPEG) and the H.26x standards developed by ITU-T Video Coding Experts Group (VCEG).

Compression standards Two compression standards are jointly developed by MPEG and VCEG in the Joint Video Team (JVT), and both standards have been shown very successful: H.262/MPEG-2 Video[1] and H.264/AVC.
The MPEG-2 Video standard [4] has been developed for broadcasting and digital television for both standard-definition and high-definition content. The main objective for these applications is high quality. Main Profile at Main Level (MP@ML) is widely implemented. This conformance point is adopted in a number of applications such as DVD-Video, digital cable television, and terrestrial broadcasting.
The H.264/AVC standard [5] has been developed for a wider range of applications: streaming applications, interactive applications (e.g., videoconferencing), and entertainment-quality applications (e.g., digital television broadcasting). In a first version, three profiles have been standardized (Baseline, Extended, and Main) [6]. Afterwards, a second version with professional profiles was added. These profiles are often indicated as the Fidelity Range Extensions (FREext) [7, 8]. Finally, a scalable extension of H.264/AVC (SVC) and a multi-view extension of H.264/AVC (MVC) were standardized. In this work, we focus on the H.264/AVC video coding standard.

Network and device constraints The video bitstreams are sent over heterogeneous networks which have different characteristics. The main characteristics are bandwidth, delay, latency, and jitter. Some networks have variable availability of bandwidth and cannot guarantee quality of service (QoS) for real-time applications. Other networks are susceptible to bit errors or packet losses and the received data may be corrupted or incomplete, so correcting actions are necessary in the network or at the receiver. Finally, the video bitstream arrives at the receiver and is decoded and displayed. The devices at receiver side may have different characteristics like display properties, processing power, battery lifetime, network connectivity etc. When the video bitstream does not comply with the constraints imposed by the receiving device, an adaptation of the video bitstream is required.
In many applications, network and device constraints are unknown at encoding time. In these cases, the properties of the video bitstream may not comply with these constraints and an adaptation of the video bitstream is required. The adaptation of the video bitstream will be performed somewhere in the delivery chain; this could be right after encoding (at the sender), somewhere in an intelligent network node, or just before decoding (at the receiver). This adaptation operation should be done in an efficient way. This means that the quality should be as good as possible while the adaptation does not require too many

[1] In the remainder, we refer to the H.262/MPEG-2 Video standard as MPEG-2 Video.

resources. In this context, two approaches may provide a solution: single-layer video coding with video transcoding or scalable video coding with bitstream extraction.

Scalable video coding Scalable video coding refers to the ability to remove parts of a video bitstream in order to adapt it to the constraints imposed by networks and devices. This typically results in a layered approach with a base layer and one or more enhancement layers. Scalable video coding has a number of benefits compared to single-layer coding. The layering is provided during coding and the bitstream extraction is a simple operation. This requires less processing in networks and devices. As a result of the layered approach, the video bitstreams can be transmitted over networks with different characteristics, and decoded and displayed on devices with different characteristics. On top of that, different error protection can be used for different layers. This makes the transmission system more robust. Nevertheless, scalable video coding has also a number of drawbacks which can be problematic for some applications. The bitstream extraction is a simple operation; however, the coding process requires more processing compared to single-layer coding. The layering is provided during the coding process, so the extraction points need to be known before coding. The compression efficiency of layered coding is typically lower compared to single-layer coding.
Recently, the scalable extension of H.264/AVC has been standardized [9]. This extension has shown significant improvement in coding efficiency combined with increased support for scalability relative to scalable profiles of prior video coding standards. This scalable extension has a number of advantages for specific applications; however, single-layer video coding is still widely used for most applications. Scalable video coding will only be used when the benefits outweigh the drawbacks.

Transcoding In the remainder of this work, we only consider single-layer video coding where video transcoding is used for the adaptation of video bitstreams. Transcoding is the operation of converting a video bitstream from one format to another format [10]. A format is defined by such characteristics as the bit rate, the frame rate, the spatial resolution, the coding format etc. In a transcoding scenario, there are two important aspects: the required processing and memory resources for the transcoding operation and the visual quality of the video after the transcoding operation.
There exist different types of transcoding which are often combined in order to meet the constraints imposed by networks and devices. The first solutions for transcoding were focusing on reducing the bit rate [11, 12]. When the avail-

able bandwidth on the network is not sufficient for transmission of a video bitstream, a reduction of the bit rate is required. As a result of the bit rate reduction, the visual quality of the video will decrease accordingly. This type of transcoder is indicated as bit rate reduction transcoder and is the main topic in this work.
Later on, when mobile devices became more common, there was a need for transcoding solutions that change the temporal or spatial properties of the video bitstreams. This results in transcoding solutions for frame rate reduction [13–15] or spatial resolution reduction [16–18].
Another transcoding operation changes the coding format. This operation is often called heterogeneous transcoding and is required when a device is not capable of decoding a video bitstream. The conversion often requires a decoding operation followed by an encoding operation due to differences in the coding tools. When other techniques are used, drift compensation is often incorporated. The conversion from MPEG-2 Video to H.264/AVC was extensively investigated [19, 20].
Other types of transcoding exist which are less common. Some applications require the insertion of information such as a logo or a banner [21]. This type of transcoding is popular in broadcasting solutions. When the network is not reliable, one could add error protection. This type of transcoder is often used for mobile applications. In the remainder of this work, we only consider bit rate reduction transcoding.

Transrating In this work, we focus on bit rate reduction transcoding, which is often called *transrating*. The objective of transrating is to reduce the bit rate while maintaining low complexity and achieving the highest quality possible [22]. Transrating is defined as an operation which, given a video bitstream and a set of bit rate constraints, produces another video bitstream that complies with the set of bit rate constraints by reducing the amount of residual data. Different techniques for transrating have been proposed in the literature: *rate shaping* [23, 24] and *requantization* [11, 12, 25]. These transrating techniques reduce the amount of residual data in a different way. Rate shaping removes high-frequency or less-relevant transform coefficients without changing the quantization parameter, while requantization increases the quantization parameter and introduces more quantization noise.

Rate shaping The first method for reducing the bit rate of video bitstreams is rate shaping. Rate shaping is a technique that reduces the amount of residual data by dropping transform coefficients. The transform coefficients which are copied to the outgoing video bitstream are given by the breakpoint. The

breakpoint should be chosen in such a way that the least-relevant transform coefficients are dropped first.
The first application of transform coefficient removal was found in encoders [26, 27]. The technique is called *thresholding* and is used in order to improve the rate-distortion performance. Thresholding is often combined with quantizer selection. The first video coding standards had a fixed quantizer for all blocks in a picture and the quantizer selection is therefore based on global statistics. The breakpoint can vary for each block and is typically determined using local statistics in the picture.
Afterwards, the thresholding operation was shifted to the network where it was used for reducing the bit rate of video bitstreams. This is the first application of rate shaping and is applied to MPEG-2 Video. The rate shaping solution directly operates on the video bitstream, so the architecture is very simple. The variable-length codewords of the incoming video bitstream are parsed and copied to the outgoing bitstream as indicated by the breakpoint. The main problem is the way the breakpoints are determined [23, 24] which is done for each block individually. There has been shown that the optimal set of breakpoints is somewhat invariant to the accumulated error from past pictures. Additionally, different approaches have been compared, from the optimal algorithm with high complexity to clustering with low complexity.
We provide an investigation of rate shaping for H.264/AVC in Chapter 2. First, we show that extensions are required in order to apply rate shaping to H.264/AVC. Afterwards, we make a study of the quality degradation which results from data dependencies and propose a solution that compensates for the shaping error. Finally, we show that an optimal algorithm will be very complex and that clustering is required in order to control the complexity.

Requantization Another method for reducing the bit rate is requantization. Requantization is a technique that reduces the amount of residual data by applying a coarser quantizer. As a result, more quantization noise is introduced and the output levels can be coded with fewer bits. There are two main issues about requantization which have been studied in the past for various compression schemes: what is the best quantizer design in the transrater and what is the best architecture for transrating.
The best quantizer design in the transrater deals with the characteristics of the quantizer (i.e., quantizer step size and quantizer offset). Generally, the outcome of direct quantization is compared with the outcome of requantization. The outcome of direct quantization is often not equal to the outcome of requantization. This results from the fact that the quantizer in the transrater has only access to the already quantized transform coefficients instead of the original

transform coefficients. An additional error is found which is often indicated as the requantization error. When no requantization error is found, direct quantization and requantization have the same outcome.
In the past, different approaches have been presented in order to reduce or eliminate the requantization error. The main problem is how to find an appropriate quantizer design. One approach, called *perfect requantization* [28], selects a suitable quantizer based on characteristics of the quantizer in the coder in order to eliminate the requantization error. No requantization errors are found; however, there are some important drawbacks. The quantizer in the transrater is coarse compared with the quantizer in the coder, and additional information is required in the requantization transrater which is typically not transmitted in the video bitstream.
We propose a different approach in Chapter 3 where we derive a transrating heuristic. The transrating heuristic is based on theoretical rate-distortion results from a Laplace source which is quantized twice. Furthermore, we evaluate the transrating heuristic using H.264/AVC video bitstreams and show that requantization is improved.

Another important issue for requantization is the architecture. The first solutions for requantization used one architecture for transrating all pictures. In that time, open-loop requantization was compared with the cascaded decoder-encoder. These solutions result in low visual quality or high complexity [25].
Afterwards, solutions have been proposed in order to reduce the drift propagation in open-loop requantization. Only temporal drift is found which results from the motion-compensated prediction. One of the first approaches for reducing the temporal drift compensates for the requantization errors made in the reference block [11]. This approach requires one prediction loop and memory buffers for storing the requantization errors of the reference pictures.
Three solutions for requantization are found in the literature: open-loop requantization (no prediction loop, low complexity, severe drift propagation), fast pixel-domain transrating (one prediction loop, medium complexity, small rounding errors), and the cascade of decoder and encoder (two prediction loops, high complexity, drift-free transrating).
In the H.264/AVC specification, the number of modes is highly increased. This was required in order to meet the performance requirements. It is important to use a suitable transrating technique depending on the impact on rate, distortion, and complexity.
Therefore, a mixed architecture will provide a better solution. This technique was first presented for H.264/AVC [29]. The I pictures are decoded and encoded, while the P and B pictures are compensated. This way, no temporal drift is found. However, nothing was done with the I macroblocks in P and

B pictures, so spatial drift was found. The spatial drift can be problematic, especially when the number of I macroblocks in P and B pictures is high.
We start in Chapter 4 with an evalution of transrating techniques in the context of H.264/AVC. None of the techniques provides a good solution which results in good visual quality and low complexity. We propose mixed architectures which apply different techniques based on the picture/macroblock type.

Transrating software In order to study rate shaping and requantization, an operational framework was developed. This framework can transrate Main Profile and High Profile H.264/AVC video bitstreams. The software is developed in such a way that different techniques can be easily combined in an efficient way. Transrating speed is not the main purpose; however, it gives an indication of how fast the transrating is.

Overview of publications Parts of the work presented in this book has been performed in the context of Interdisciplinary Institute for Broadband Technology (IBBT) projects or in association with the Service Provider Video Technology group of Cisco (formerly Scientific Atlanta) located in Kortrijk, Belgium. The author's research has led to eight publications and two submissions (still under review) in SCI-indexed journals. Furthermore, the presented work has led to more than 20 conference publications from which eight as first author. We refer to Appendix A for the full publication list.

Chapter 2

Rate shaping for H.264/AVC

2.1 Rationale and related work

Video compression is required in order to efficiently store and transmit digital video. Many compression schemes have been presented and standardized. Most of these schemes are based on single-layer coding; however, multi-layer coding has been extensively investigated (e.g., subband coding and hierarchical coding). Although single-layer and multi-layer coding have their strengths and weaknesses, single-layer coding has always been dominating. Single-layer coding results in higher compression efficiency and lower complexity for coding and decoding; however, bitstream adaptation cannot be achieved efficiently.

In some cases, the channel capacity may not be sufficient for sending the video bitstream in real-time. This may result from data traffic generated on the network by other users, or the capacity of the transmission link may not be sufficient. In both cases, fast and efficient techniques for adaptation are necessary. In the literature, two techniques have been presented for reducing the bit rate of single-layer video bitstreams: rate shaping and requantization. In this chapter, we investigate rate shaping for H.264/AVC bitstreams.

Rate shaping is often used in multimedia applications in order to adjust the bit rate of video bitstreams. Various examples can be found where rate shaping can provide a solution for bit rate reduction. A first example is streaming video over the Internet [1]. The video bitstream is transmitted over the network where a best-effort strategy is used. As a result, there is no guarantee that the video bitstream will be received by the client in real-time. Here, rate shaping can dynamically adjust the bit rate in order to meet the real-time constraint. A second example covers the transport of digital television channels over the cable network [2]. The video bitstream is sent over the backbone network

with sufficient quality. However, the access network may not have sufficient capacity to transmit the video bitstream in real-time. Here, the rate shaping solution can adjust the bit rate of the video bitstream at the cable head-end.

The concept of thresholding was first implemented in encoding systems where transform coefficients are dropped in order to increase the coding efficiency. Ramchandran *et al.* presented a thresholding algorithm that drops transform coefficients in rate-distortion optimal sense [30]. The thresholding algorithm is applied to compressed image and video bitstreams and results in compliant bitstreams. Crouse *et al.* extended the thresholding algorithm with an algorithm for optimal selection of the quantizer [26, 27]. The authors combined the optimal quantizer selection and the optimal thresholding, and additionally customize the Huffman coding tables in order to adjust the entropy coding. The selection of the quantizer is applied to all blocks of the picture. As a result, the quantizer selection exploits picture-wide statistics. The thresholding operation is applied to each block individually. As a result, the thresholding operation exploits block-wide statistics. Therefore, the algorithms for optimal quantizer selection and optimal thresholding are nearly orthogonal.

Later on, data partitioning has been investigated as an extension of thresholding. The residual data in the video bitstream is segmented into two partitions. These partitions can be treated differently by the network. Eleftheriadis *et al.* provided an analysis of the problem of optimal data partitioning in an operational rate-distortion context [31, 32]. The optimal algorithm is shown to have significantly high complexity and delay. This comes from the motion-compensated prediction. The causally optimal algorithm optimally solves the problem when additional constraints of causal operation and low-delay are imposed. The memoryless algorithm is shown to perform almost identical; however, the computational complexity is significantly reduced.

Afterwards, the concept of data partitioning was shifted to the network in order to reduce the bit rate of video bitstreams. The operation was called rate shaping which, given the original video bitstream and a bit rate constraint, generates a new video bitstream that complies with the bit rate constraint by discarding transform coefficients from the original video bitstream. Eleftheriadis *et al.* made an evaluation of rate shaping for MPEG-2 bitstreams [23, 24]. They present theoretical results from a statistical and rate-distortion analysis of a first-order autoregressive source for motion-compensated prediction. They give an explanation as to why the optimal set of breakpoints is somewhat invariant to the accumulated error from the reference pictures. They also present experimental results for various bit allocation algorithms (optimal algorithm, causally optimal algorithm, and memoryless algorithm). They show that fast designs perform very close to the optimal solution with a significant reduction

in processing time.

The investigation of rate shaping has shown that the visual quality degrades from picture to picture. The shaping operation results in blocking artifacts and causes temporal drift due to motion-compensated prediction. Some approaches try to preserve the visual quality by applying techniques in the transcoder. Zeng *et al.* applied feature-oriented rate shaping where they preserve transform coefficients in the edge blocks in order to avoid severe blocking artifacts [33]. Ho *et al.* proposed an efficient and practical foveation model for MPEG-1 bitstreams and use this model for rate shaping [34].

The popularity of the H.264/AVC video coding standard in multimedia applications is driven by the strengths of this compression standard. The main objective for the standardization was a significant increase of the compression efficiency. A number of questions about H.264/AVC rate shaping are still unanswered. These questions deal with the design of the solution, the complexity and the performance, the drift control, the bit allocation and so on. We tackle these issues in this chapter.

In this chapter, we provide an investigation of rate shaping for H.264/AVC video bitstreams. An overview of MPEG-2 Video specification is given Section 2.2 where rate shaping for MPEG-2 Video is presented. The rate shaping problem for H.264/AVC is presented in Section 2.3, where we elaborate on the main problems. The extensions for H.264/AVC rate shaping are presented in Section 2.4. We make an analysis of drift in Section 2.5, and present compensation techniques in order to restrain the drift. Bit allocation for H.264/AVC rate shaping is treated in Section 2.6. Performance results are given in Section 2.7 and the conclusion and future work follow in Section 2.8.

2.2 Rate shaping of MPEG-2 Video

This section provides an overview of the basics of MPEG-2 Video [4] and elaborates on MPEG-2 Video rate shaping. In particular, the MPEG-2 Video coding tools are described which have an impact on the rate shaping operation.

2.2.1 MPEG-2 Video coding standard

The MPEG-2 Video coding standard defines hybrid block-based coding where predictive and transform coding techniques are combined in order to compress the video signal [35]. The pictures of the video signal are divided into macroblocks which are the basic units for coding. When 4:2:0 subsampling is used, each macroblock contains one 16×16 block with luminance pixel values and two 8×8 blocks with chrominance pixel values. As a consequence,

a macroblock consists of six 8×8 blocks which are the input for transform coding.

Different picture types are used in the MPEG-2 Video coding standard. The I pictures represent the key pictures in the video bitstream. These pictures are coded independently and the intensity values are transformed, quantized, and entropy coded. For the P pictures and the B pictures, predictive and transform coding are combined. The predictive coding encloses motion-compensated prediction where the best match in the reference picture is indicated by the motion vector and the error values are transformed, quantized, and entropy coded.

In the remainder of this section, we describe MPEG-2 Video transform coding and we focus on the way the transform coefficients are manipulated. Therefore, other data generated during the coding process, not relevant for this analysis, is not considered. The N 8×8 blocks in a picture are the input for transform coding and contain, depending on the picture type, intensity values or error values: $\mathbf{f^i}, i \in \{0, \ldots, N-1\}$. These input samples are transformed using the *discrete cosine transform* (DCT) [36]:

$$F^i_{m,n} = \frac{C(m)}{2}\frac{C(n)}{2}\sum_{k=0}^{7}\sum_{l=0}^{7} f^i_{k,l} \cos\left(\frac{(2k+1)\pi m}{16}\right)\cos\left(\frac{(2l+1)\pi n}{16}\right),$$
$$\text{for } i \in \{0, \ldots, N-1\} \text{ and } m, n \in \{0, \ldots, 7\}, \tag{2.1}$$

where

$$C(x) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } x = 0; \\ 1, & \text{if } x \neq 0. \end{cases} \tag{2.2}$$

The quantization of the transform coefficients is position-dependent. The quantizer step size $Q_{m,n}$ and the quantizer offset $\epsilon_{m,n}$ control the quantization. The quantization can be described as follows:

$$\hat{F}^i_{m,n} = \left\lfloor \frac{F^i_{m,n}}{Q_{m,n}} + \epsilon_{m,n} \right\rfloor Q_{m,n},$$
$$\text{for } i \in \{0, \ldots, N-1\} \text{ and } m, n \in \{0, \ldots, 7\}, \tag{2.3}$$

where $\lfloor a \rfloor$ is the largest integer not larger than a.

After the transformation and the quantization, the mode data, the motion data, and the residual data are entropy coded. The mode data includes the macroblock mode, the prediction mode, and the macroblock partitioning, the

motion data covers the motion vectors and the reference picture indices, and the residual data consists of the quantized transform coefficients. The entropy coding assigns bitstrings to the syntax elements in an efficient way so that more probable symbols are assigned short codewords and less probable symbols are assigned long codewords.

In the following, we provide a description for MPEG-2 Video bitstreams. For simplicity, we omit the mode and motion data and only consider the residual data. Before the entropy coding is applied, the residual data is rearranged. The residual data is converted from a two-dimensional structure to a one-dimensional sequence according to the zig-zag scan order or the alternate scan order[1]. Scanning the transform coefficients in this fashion allows efficient run-level coding in the subsequent entropy coding. The transform coefficients are addressed using the index p instead of the position m, n.

The quantized transform coefficients are coded in a different manner depending on their frequency. A different method is applied for the DC transform coefficient and the AC transform coefficients:

- **DC transform coefficient:** The differential coding assigns a Huffman codeword from a one-dimensional table $\mathcal{H}_{dc}$ to each quantized DC transform coefficient according to the difference between its value and the quantized DC transform coefficient from the previous block.
- **AC transform coefficients:** The run-length coding assigns a Huffman codeword from the two-dimensional table $\mathcal{H}_{ac}$ to each non-zero quantized AC transform coefficient according to the number of zero quantized transform coefficients preceding the AC transform coefficient in zig-zag scan order or the alternate scan order z_p^i and the amplitude of the AC transform coefficient.

As a result, the variable-length codeword corresponding to the quantized transform coefficient $\hat{F}_p^i$ is assigned to the bitstring S_p^i:

$$S_p^i = \begin{cases} \mathcal{H}_{dc}\left(\hat{F}_p^i - \hat{F}_p^{i-1}\right) & \text{if } p = 0; \\ \mathcal{H}_{ac}\left(z_p^i, \hat{F}_p^i\right) & \text{if } p \neq 0, \hat{F}_p^i \neq 0; \\ \phi & \text{if } p \neq 0, \hat{F}_p^i = 0. \end{cases} \tag{2.4}$$

The symbol ϕ represents the null codeword which is used for zero transform coefficients. For coding efficiency, an end-of-block symbol E is introduced. Then, the output bitstream $B_{\text{MPEG-2 Video}}$ can be represented as:

[1]The zig-zag scan order is more suitable for progressive material while the alternate scan order is more suitable for interlaced material.

$$B_{\text{MPEG-2 Video}} \approx \bigvee_{i=0}^{N-1} \left(\left(\bigvee_{p=0}^{63} S_p^i \right) \vee E \right), \tag{2.5}$$

where the operator $\vee$ denotes concatenation of bitstrings. The number of bits $R_{\text{MPEG-2 Video}}$ needed for coding one picture can be expressed as follows:

$$R_{\text{MPEG-2 Video}} \approx \sum_{i=0}^{N-1} \left(\sum_{p=0}^{63} \left| S_p^i \right| + |E| \right), \tag{2.6}$$

where $|a|$ measures the length of the codeword a.

The transform coefficients are transformed to single bitstrings which make up the main part of the video bitstream. Each bitstring represents both run and level information of the transform coefficient. This way, rate shaping is an easy operation since each transform coefficient can be uniquely identified in the bitstream. When a transform coefficient is removed only an update of the variable-length codeword associated with the next transform coefficient in zig-zag scan order is required in order to correct the run information.

2.2.2 MPEG-2 Video rate shaping

Rate shaping is an operation which, given a bitstream B and a bit rate constraint R_T, produces another bitstream $\hat{B}$ that complies with the bit rate constraint by discarding transform coefficients from bitstream B. This is shown in Figure 2.1 where the bit allocation operation determines a breakpoint which is used in the rate shaping operation.

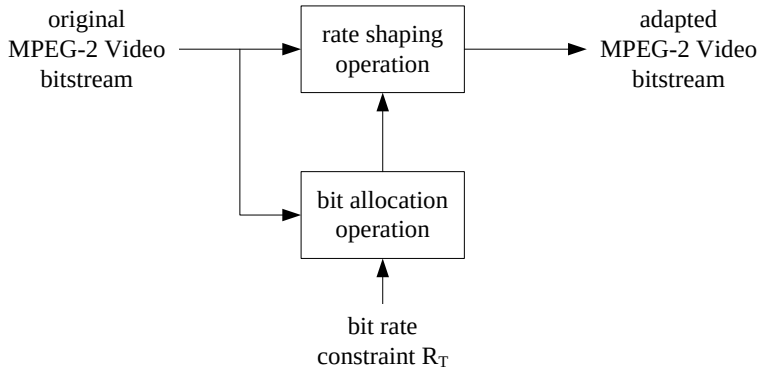


Figure 2.1: Rate shaping operation and bit allocation operation.

There is no communication path between the rate shaping operation and the source of the video. As a result, no access to the encoder is required and the coding process is detached from the rate shaping operation. Of particular interest is the source of the bit rate constraint R_T. The bit rate constraint may be constant and known a priori (e.g., defined as the bandwidth of a circuit-switched network) or the bit rate constraint may be variable[2] (e.g., provided by the network management layer after end-to-end bandwidth availability estimates).

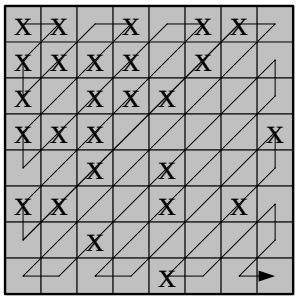
Generally, two rate shaping methods are distinguished [24]: constrained rate shaping and general rate shaping. Constrained rate shaping discards a range of non-zero transform coefficients at the end of each block. The number of non-zero transform coefficients that will be kept is determined by the breakpoint b where $b \in \{1, \ldots, 64\}$. General rate shaping removes an arbitrary set of transform coefficients in each block. The set of transform coefficients is indicated by zeros and ones in the breakpoint vector $\mathbf{b} = \{\Theta_0, \ldots, \Theta_{63}\}$ where $\Theta_0 = 1$[3] and $\Theta_i \in \{0, 1\}$ for $i > 0$. The difference between constrained and general rate shaping is depicted in Figure 2.2 where X represents a non-zero transform coefficient.

General rate shaping outperforms constrained rate shaping, but it does so only marginally [24]. The experimental results for MPEG-2 Video rate shaping from 4 Mbps to 3.2 Mbps have shown a small gap less than 1 dB between constrained and general rate shaping. The effectiveness of constrained rate shaping comes from the zig-zag scan order which is used to convert the residual data from a two-dimensional structure to a one-dimensional sequence and the optimized design of the tables with variable-length codewords of the MPEG-2 Video entropy coding for residual data. General rate shaping may drop any subset of transform coefficients for which the zig-zag scan order and the entropy coding are not optimized. On top of that, the number of breakpoints for general rate shaping highly exceeds the number of breakpoints for constrained rate shaping which results in a significant increase in complexity. Since general rate shaping is far more complex compared to constrained rate shaping and the latter one performs close to the former one, we concentrate on constrained rate shaping in the remainder of this chapter.

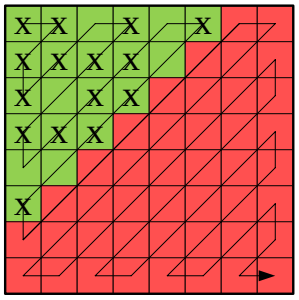
A rate shaping solution for MPEG-2 Video bitstreams discards high-frequency transform coefficients from the bitstream without changing other syntax elements. As a result of the simple design of the entropy coding, re-

[2] A variable bit rate constraint most often comes from statistical multiplexes where the bit rate is shared between video sources with dynamic complexity.

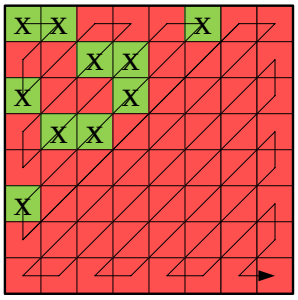
[3] This additional condition avoids the recoding of the coded block patterns and the re-execution of the DC predictions.



(a) Original residual 8×8 block.



(b) Constrained rate shaping: $b = 15$.



(c) General rate shaping: $\mathbf{b} = \{1, 1, 0, 1, 0, 0, 0, 1, \ldots\}$.

Figure 2.2: Illustration of constrained and general rate shaping.

moving a transform coefficient is an easy operation. We only need to parse the bitstream and remove variable-length codewords according to the breakpoint given by the bit allocation operation. The rate shaping operation for MPEG-2 Video bitstreams is shown in Figure 2.3.

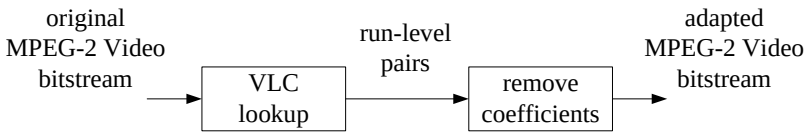


Figure 2.3: Rate shaping operation for MPEG-2 Video bitstreams: VLC lookup and transform coefficient removal.

Let $\hat{\mathbf{x}}$ and $\mathbf{x}$ be the decoded video with and without rate shaping, and let $\hat{R}$ and R be the corresponding bit rates of the video bitstreams. Then, the objective of rate shaping is to minimize the rate shaping distortion $\|\mathbf{x} - \hat{\mathbf{x}}\|$ subject to the bit rate constraint R_T [24]: $\min_{\hat{R} \leq R_T} \|\mathbf{x} - \hat{\mathbf{x}}\|$. No assumption

is made about the bit rate of the original bitstream R, which can indeed be arbitrary. The minimum rate variation $\hat{R}/R$ depends on the amount of residual and non-residual data[4] in the original bitstream.

Rate shaping is often considered a low-complexity technique; however, the bit allocation can be very complicated. In the literature, different algorithms for bit allocation of MPEG-2 bitstreams were presented with varying performance and complexity [24]:

- The *optimal algorithm* for the minimization problem would have to examine a complete group of pictures since breakpoint decisions at the initial I picture may even affect the last P or B picture. Not only the computational overhead would be extremely high, but the delay would be unacceptable as well. This algorithm is illustrated in Figure 2.4(a).

- An attractive alternative is one that solves the minimization problem on a picture basis, where only the error accumulated from past pictures is taken into account. This algorithm will be referred to as *causally optimal algorithm*. This algorithm is illustrated in Figure 2.4(b).

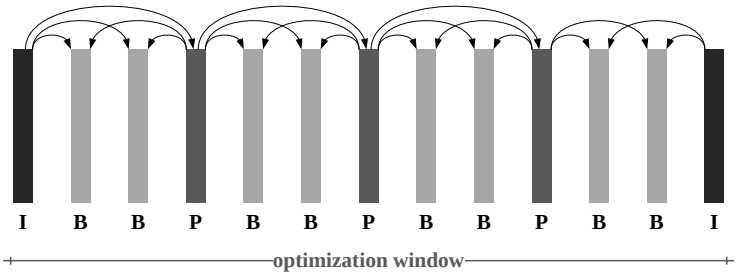
- Starting from the causally optimal algorithm, it is interesting to examine the benefit of error accumulation tracking. This can be evaluated by applying the algorithm for intra-only to P and B pictures. Surprisingly, the results of this *memoryless algorithm* are almost identical to the causally optimal algorithm. This algorithm is illustrated in Figure 2.4(c).

No indication is given on the length of the optimization window. Complexity and delay considerations make it desirable that it is kept small. Therefore, our interests will focus on the case where the algorithm is optimized for one picture (refer to Figure 2.4(b) and Figure 2.4(c)).

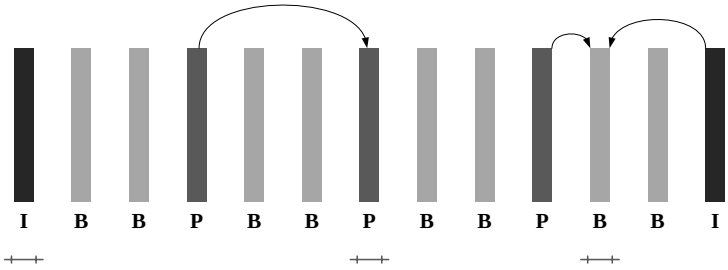
2.3 Impact of coding tools on rate shaping

One of the goals of the H.264/AVC video coding standard is to enhance the coding efficiency compared to previous video coding standards. In order to enhance the coding efficiency, the coding tools have been improved [5, 6], in particular the predictive coding, the transform coding, and the entropy coding. As a consequence, these coding tools are more complex compared to their counterparts in previous video coding standards. These changes also have an impact on the rate shaping and the bit allocation. We give an overview of the

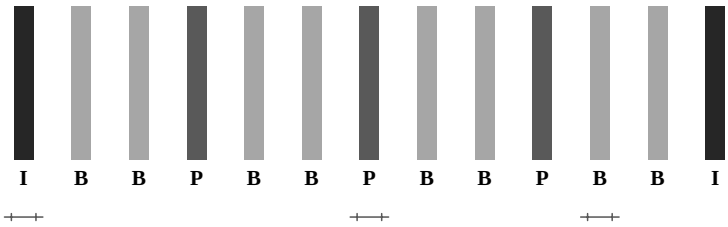
[4]The residual data only consists of the quantized transform coefficients (both DC and AC coefficients), while the non-residual data mainly consists of the mode and motion data.



(a) Optimal algorithm: high complexity, high delay



(b) Causally optimal algorithm: high complexity, low delay



(c) Memoryless algorithm: low complexity, low delay

Figure 2.4: Different algorithms for bit allocation (arrows indicate error accumulation tracking, intervals indicate optimization window).

H.264/AVC coding tools and the bit allocation and discuss the impact on the rate shaping operation. The block diagram of the H.264/AVC video coder is shown in Figure 2.5.

Entropy coding The entropy coding of previous video coding standards combines zig-zag scan, run-level coding, and variable-length coding in order to generate the bitstream. The bitstream consists of variable-length codewords which represent the mode data, the motion data, and the residual data. When applying rate shaping, variable-length codewords belonging to residual data

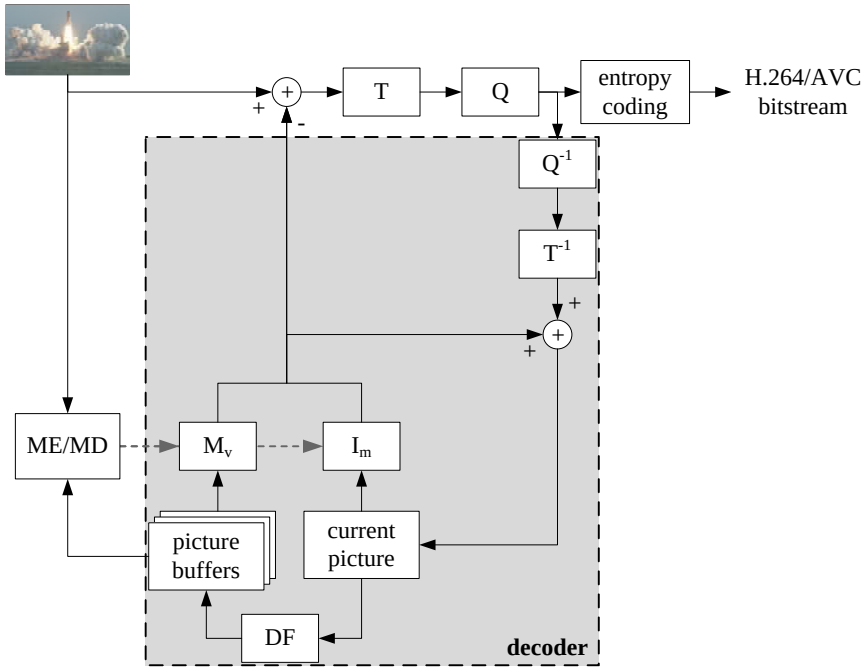


Figure 2.5: Block diagram of the H.264/AVC video coder: transform T, quantization Q, motion estimation ME, mode decision MD, motion compensation M_v, intra coding I_m, and deblocking filter DF.

are removed. This is a simple operation which requires little processing power. The entropy coding of the H.264/AVC video coding standard is more complicated. The entropy coding is context-adaptive which means that the coding of blocks or transform coefficients is based on the coding of previous blocks or transform coefficients. On top of that, the run and level information are coded separately. Thus, a transform coefficient is not represented by a single bitstring in the bitstream. We elaborate shortly on the design of H.264/AVC entropy coding schemes in Section 2.4.1 and show that simply discarding variable-length codewords is not possible.

Transform coding Previous video coding standards define a two-dimensional transform that operates on 8×8 blocks with luminance or chrominance data. As a result, one single breakpoint can be used for all types of residual data. The bit allocation selects a single breakpoint for each 8×8 block which results in 65 possible truncation points. The H.264/AVC video coding standard defines integer and Hadamard transforms which can be com-

bined depending on the coding mode [37, 38]. As a result, the residual data is dispersed over different blocks with different sizes. Therefore, one single breakpoint cannot be used. We give a brief overview of the H.264/AVC transform schemes in Section 2.4.2 and propose a solution based on a virtual breakpoint.

Predictive coding Predictive coding in previous video coding standards is restricted to motion-compensated prediction. When applying rate shaping to bitstreams using motion-compensated prediction, temporal drift arises. The temporal drift results from changes in the reference pictures as a result of rate shaping and is acceptable for bitstreams with short GOP length. In H.264/AVC video coding, intra prediction and motion-compensated prediction are combined. Both spatial and temporal redundancies are exploited in order to reduce the amount of residual data. When applying rate shaping to H.264/AVC bitstreams, both spatial and temporal drift are found. We need to know what the impact is of the spatial and temporal drift on the visual quality. We propose to use compensation techniques in order to restrain the drift in Section 2.5.

Bit allocation The objective of bit allocation is to find the optimal set of breakpoints. Even for previous video coding standards, this was a very complex issue. The rate-distortion optimal solution is very complex and introduces an unacceptable delay. Therefore, some interesting alternatives were presented which are more suitable for practical systems. The bit allocation for H.264/AVC is even more complicated due to the increased number of dependencies. These dependencies result from the predictive coding. We elaborate on bit allocation issues in Section 2.6.

2.4 Proposed extensions

In this section, we tackle the problems of entropy and transform coding. We describe the more complex design of H.264/AVC entropy coding and explain why full entropy decoding and encoding are required. One single breakpoint is not sufficient for different transforms and block sizes. Therefore, we propose to use a virtual breakpoint that is used for deriving the actual breakpoints. The proposed extensions are required in order to apply rate shaping to H.264/AVC video bitstreams.

2.4.1 Entropy coding

The simple approach for MPEG-2 Video bitstreams cannot be extended for H.264/AVC bitstreams since the complexity of the entropy coding is significantly increased. The H.264/AVC video coding standard provides two entropy coding schemes:

- *context-based adaptive variable length coding* (CAVLC);
- *context-based adaptive binary arithmetic coding* (CABAC).

We discuss how these entropy coding schemes work and what the impact is on the rate shaping operation.

CAVLC entropy coding The CAVLC entropy coding encodes the i^{th} 4×4 block using five syntax elements. The following data is used for the entropy coding: $\#C^i$ is the number of non-zero transform coefficients, $\#T1^i$ is the number of trailing ones (i.e., $+1$ or -1 quantized transform coefficients located at the end of the block), $\#C^A$ is the number of non-zero transform coefficients in the block to the left, and $\#C^B$ is the number of non-zero transform coefficients in the block above. The run-level pairs (z_j^i, c_j^i) where $j \in \left\{1, \ldots, \#C^i\right\}$ are arranged in reverse zig-zag scan order. The run-level pairs can be organized in two vectors $\mathbf{z^i} = \left\{z_1^i, \ldots, z_{\#C^i}^i\right\}$ and $\mathbf{c^i} = \left\{c_1^i, \ldots, c_{\#C^i}^i\right\}$ which represent the runs of zeros and the non-zero transform coefficients. The total number of zero transform coefficients before the last non-zero transform coefficients is $\#z^i = \sum_{j=1}^{\#C^i} z_j^i$.

We shortly explain the syntax elements and make clear when they change due to rate shaping:

- The syntax element *coeff_token* signals the number of non-zero transform coefficients $\#C^i$ and the number of trailing ones $\#T1^i$. A table with variable-length codewords is selected based on the number of non-zero transform coefficients in spatially adjacent blocks. An update of the syntax element is required when transform coefficients of the current block or spatially adjacent blocks are dropped. The variable-length codeword associated with this syntax element is assigned to the bitstring S_1^i:
$$S_1^i = \mathcal{H}_1(\#C^i, \#T1^i, \#C^A, \#C^B). \tag{2.7}$$
- The syntax element *trailing_ones_sign_flag* defines the signs of the trailing ones. A sign flag is dropped when a trailing one is removed. The

variable-length codeword associated with this syntax element is assigned to the bitstring S_2^i:

$$S_2^i = \mathcal{H}_2(\#T1^i, \mathbf{c^i}). \tag{2.8}$$

- The syntax element *level* codes the levels of the non-zero transform coefficients, not including the trailing ones. The levels are coded in reverse zig-zag scan order and the coding process is matched with already coded transform coefficients. All non-zero transform coefficients need to be recoded when a high-frequency transform coefficient is removed. The variable-length codeword associated with this syntax element is assigned to the bitstring S_3^i:

$$S_3^i = \mathcal{H}_3(\#C^i, \#T1^i, \mathbf{c^i}). \tag{2.9}$$

- The syntax element *total_zeros* denotes the total number of zero transform coefficients before the last non-zero transform coefficient. A bitstream update is required when transform coefficients are dropped. The variable-length codeword associated with this syntax element is assigned to the bitstring S_4^i:

$$S_4^i = \mathcal{H}_4(\#z^i). \tag{2.10}$$

- The syntax element *run_before* represents the number of zero transform coefficients preceding each non-zero transform coefficient. This syntax element needs to be updated when transform coefficients are dropped. The variable-length codeword associated with this syntax element is assigned to the bitstring S_5^i:

$$S_5^i = \mathcal{H}_5(\#z^i, \mathbf{z^i}). \tag{2.11}$$

The output bitstream B_{CAVLC} can be represented as:

$$B_{\text{CAVLC}} \approx \bigvee_{i=0}^{N-1} \left(\bigvee_{j=1}^{5} S_j^i \right). \tag{2.12}$$

The number of bits R_{CAVLC} needed for coding one picture can be expressed as follows:

$$R_{\text{CAVLC}} \approx \sum_{i=0}^{N-1} \left(\sum_{j=1}^{5} |S_j^i| \right). \tag{2.13}$$

We would like to know how the video bitstream is affected when constrained rate shaping is applied. When a trailing one is removed, all syntax

elements change except the level information of the non-zero transform coefficients. When other non-zero transform coefficients are dropped, all syntax elements change. As a result, nearly all syntax elements are affected when rate shaping is applied. So, full entropy decoding and encoding is required since an efficient solution that directly operates on the video bitstream and changes the syntax elements does not exist.

CABAC entropy coding The CABAC entropy coding encodes the i^{th} 4×4 block according to the following steps [39]: 1) the binarization, 2) the context modeling, and 3) the arithmetic coding. The block diagram of the CABAC coder is shown in Figure 2.6.

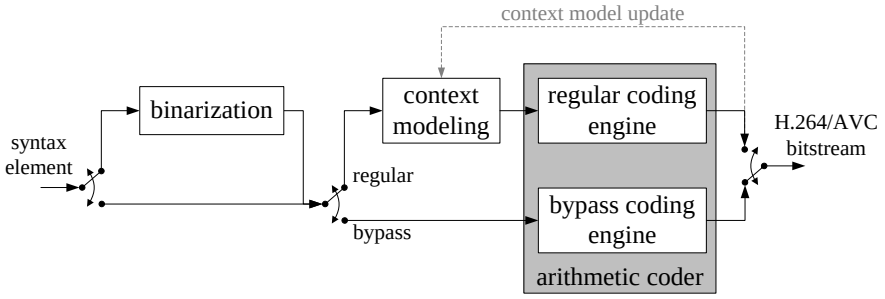


Figure 2.6: Block diagram of the CABAC coder.

The binarization maps non-binary valued syntax elements onto bin strings. Different binarization schemes are defined: unary binarization, truncated unary binarization, k^{th} order Exp-Golomb binarization, unary/k^{th} order Exp-Golomb binarization, and fixed length binarization. A context model is assigned to each bin of the bin string according to the context index γ. The context index γ depends on the semantics of the bin, the position of the bin, and the spatial context. The context model indicates the probability state index σ_γ (represented by 6-bit value) and the most probable symbol ω_γ (represented by a binary value). The binary arithmetic coder performs arithmetic coding of each bin using the selected context model. The principle of arithmetic coding is based on recursive subdivision of an interval which is defined by its lower bound and its interval length. This range is indicated by the symbol ρ. There are two coding modes: the regular coding which uses the context models and the bypass coding which is used for bins for which one expects a constant probability of $1/2$ for both values 0 and 1.

The locations of the non-zero transform coefficients are represented by the *significance map*. For each transform coefficient in zig-zag scan order, a one-

bit symbol *significant_coeff_flag* is transmitted. For each *significant_coeff_flag* symbol which is one, a one-bit symbol *last_significant_coeff_flag* is transmitted. This symbol indicates if the current transform coefficient is the last one inside the 4×4 block or if further transform coefficients follow.

The number of transform coefficients considered for building the significance map is denoted as $\#P^i$. The context model for coding the *significant_coeff_flag* symbol is indicated by $(\sigma_\chi, \omega_\chi)$, while the context model for coding the *last_significant_coeff_flag* symbol is indicated by (σ_ξ, ω_ξ). Both indexes χ and ξ are determined by the position in the significance map. The output of the arithmetic coder after coding the significance map is assigned to the bitstring S^i_{SM}, and the selected context models $(\sigma_\chi, \omega_\chi)$ and (σ_ξ, ω_ξ) and the probability state ρ of the arithmetic coder are updated:

$$S^i_{SM} = \bigvee_{j=1}^{\#P^i} \big(\mathcal{A}\left(\textit{significant\_coeff\_flag}_j|\sigma_\chi, \omega_\chi, \rho\right) \vee \mathcal{A}(\textit{last\_significant\_coeff\_flag}_j|\sigma_\xi, \omega_\xi, \rho)\big)\,. \tag{2.14}$$

The values of the non-zero transform coefficients are coded using two coding symbols: *coeff_abs_level_minus1* (representing the absolute value of a non-zero transform coefficient, minus one) and *coeff_sign_flag* (representing the sign of a non-zero transform coefficient).
The number of non-zero transform coefficients is indicated by $\#C^i$. The *coeff_abs_level_minus1* symbol is binarized using the unary/k^{th} order Exp-Golomb binarization and results in the bin string *bin_string* with length $\#bins$. The *coeff_sign_flag* symbol is coded using the bypass mode where a constant probability of $1/2$ indicates a probability state index $\sigma = 0$ and a most probable symbol $\omega = \times$, where the symbol $\times$ represents a don't care.
The context model for coding the bin $bin\_string_j[k]$ is indicated by $(\sigma_\psi, \omega_\psi)$ where the index ψ is determined by the coding process for transform coefficients. The output of the arithmetic coder is assigned to the bitstring S^i_{LV}. The selected context models $(\sigma_\psi, \omega_\psi)$ and the probability state ρ of the arithmetic coder are updated:

$$S^i_{LV} = \bigvee_{j=1}^{\#C} \left(\left(\bigvee_{k=1}^{\#bins} \mathcal{A}\left(bin\_string_j[k]|\sigma_\gamma, \omega_\gamma, \rho\right) \right) \vee \mathcal{A}\left(\textit{coeff\_sign\_flag}_j|\sigma = 0, \omega = \times, \rho\right) \right). \tag{2.15}$$

The output bitstream B_{CABAC} can be represented as:

$$B_{\text{CABAC}} \approx \bigvee_{i=0}^{N-1} \left(S_{SM}^{i} \vee S_{LV}^{i} \right). \tag{2.16}$$

The number of bits R_{CABAC} needed for coding one picture can be expressed as follows:

$$R_{\text{CABAC}} \approx \sum_{i=0}^{N-1} \left(\left| S_{SM}^{i} \right| + \left| S_{LV}^{i} \right| \right). \tag{2.17}$$

We would like to know how the video bitstream is affected when constrained rate shaping is applied. The main problems come from the arithmetic coding. Firstly, the output bits of the arithmetic coder cannot be uniquely assigned to syntax elements. This way it is impossible to remove a transform coefficient and correct the significance map. Secondly, if one could assign output bits to syntax elements, there would still be a problem with the synchronization of the context models and the range of the arithmetic coder. In order to guarantee correct decoding, the entropy coding of all subsequent 4×4 blocks should be updated as well. Such an update operation is quite complex and requires that all subsequent 4×4 blocks are entropy decoded and encoded.

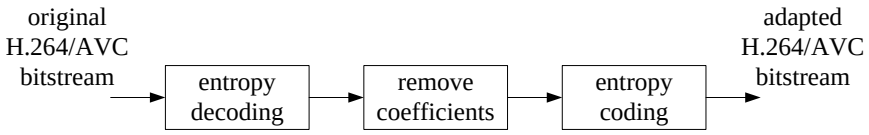


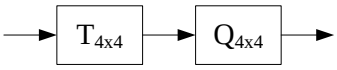
Figure 2.7: Rate shaping process for H.264/AVC video bitstreams: entropy decoding, transform coefficient removal, and entropy coding.

Because of the design of CAVLC and CABAC entropy coding schemes, directly removing transform coefficients from H.264/AVC video bitstreams is not possible. As a result, full entropy decoding and encoding are required. This, however, complicates the rate shaping process for H.264/AVC. The rate shaping process for H.264/AVC video bitstreams is presented in Figure 2.7.

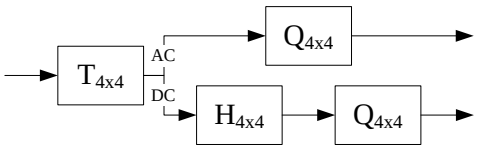
2.4.2 Transform coding

The H.264/AVC video coding standard provides four different transforms: 4×4 integer transform, 4×4 Hadamard transform, 2×2 Hadamard transform, and 8×8 integer transform. These transforms are used in four transform schemes which are illustrated in Figure 2.8:

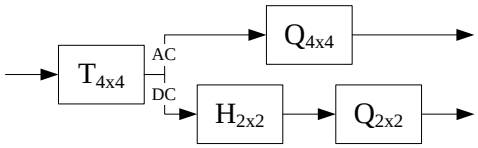
- 4×4 integer transform;
- 4×4 integer transform followed by 4×4 Hadamard transform;
- 4×4 integer transform followed by 2×2 Hadamard transform;
- 8×8 integer transform.



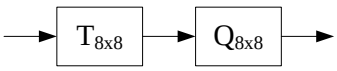
(a) 4×4 integer transform



(b) 4×4 integer transform followed by 4×4 Hadamard transform



(c) 4×4 integer transform followed by 2×2 Hadamard transform



(d) 8×8 integer transform

Figure 2.8: Illustration of four transform schemes ($T_{B\times B}$, $H_{B\times B}$, and $Q_{B\times B}$ denote the integer transform, the Hadamard transform, and the quantization for $B\times B$ blocks, respectively).

The one-stage schemes apply the 4×4 or 8×8 integer transform to the residual data before applying quantization (see Figure 2.8(a) and Figure 2.8(d)). The two-stage schemes apply the 4×4 integer transform to the residual data and subsequently apply the 2×2 or 4×4 Hadamard transform to the DC values from the integer transform (see Figure 2.8(b) and Figure 2.8(c)).

The different transforms and block sizes complicate the rate shaping operation. When all blocks are treated the same way (i.e., the same breakpoint is used regardless of the origin of the transform coefficients), the rate shaping

process removes transform coefficients with different importance. This will have an impact on the visual quality. As a consequence, different breakpoints are required for rate shaping.

Therefore, we propose to use a *virtual breakpoint* b in order to control the rate shaping process. This breakpoint spans 64 frequency components and determines the actual breakpoints which are used during rate shaping. We define a mapping between the virtual breakpoint and the actual breakpoints: $b_{8\times8}^{int} = b$ and $b_{4\times4}^{int} = \lfloor \frac{b+2}{4} \rfloor$. These values are chosen so that the number of coefficients retained is proportional to the total number of coefficient bands (16 resp. 64).

These breakpoints are used for rate shaping of blocks which result from the 4×4 and 8×8 integer transforms. Other transform coefficients are the output of the 2×2 or 4×4 Hadamard transforms. Since these transform coefficients represent low-frequency information (i.e., DC transform coefficients from the integer transforms), no rate shaping is applied to these blocks. In the remainder of this chapter, we always indicate the virtual breakpoint of the rate shaping process.

2.5 Rate shaping with drift compensation

In this section, we make an analysis of the rate shaping operation for motion-compensated prediction and intra prediction. In order to solve the minimization problem, we need to calculate the accumulated error. We propose to compensate the accumulated error in order to restrain drift propagation. We present mixed architectures for rate shaping that combine spatial and temporal compensation. Finally, we discuss the impact of deblocking on the adapted video bitstreams.

2.5.1 Drift propagation evaluation

In order to evaluate H.264/AVC rate shaping, we perform some experiments for motion-compensated and intra-predicted pictures.

Rate shaping of motion-compensated pictures In a first experiment, we apply rate shaping to motion-compensated pictures and leave intra-predicted pictures unchanged. We show some screenshots of motion-compensated pictures after rate shaping in Figure 2.9. The visual quality of the pictures mainly depends on the temporal drift. The size of the GOP, the breakpoint values, and the motion activity in the video scene have an impact on the temporal drift, while the number of intra-predicted blocks have an impact on the spatial drift.

(a) $b = 16$, 36.89 dB, 0.778 Mbps, 2.5% reduction

(b) $b = 8$, 34.75 dB, 0.700 Mbps, 12.2% reduction

(c) $b = 4$, 33.23 dB, 0.577 Mbps, 27.7% reduction

Figure 2.9: Rate shaping of motion-compensated pictures (Main Profile, CABAC, IPPP, GOP length $= 16$, $QP = 26$, *Foreman*, CIF, 30 fps, 96^{th} picture, motion-compensated).

For breakpoints $b = 16$ and $b = 8$, almost no visual artifacts can be seen for bit rate reductions of 2.5% and 12.2%, respectively. For breakpoint $b = 4$, some smaller errors are found; however, the visual quality is still acceptable. This results in a bit rate reduction of 27.7%. From this, it follows that rate shaping can be applied to motion-compensated pictures. However, when the size of the GOP increases, the breakpoint values increase, or the motion activity of the video scene is high, the visual quality will degrade more severely due to temporal drift.

(a) $b = 24$, 21.93 dB, 0.778 Mbps, 2.6% reduction

(b) $b = 12$, 17.59 dB, 0.702 Mbps, 12.1% reduction

(c) $b = 8$, 16.63 dB, 0.627 Mbps, 21.5% reduction

Figure 2.10: Rate shaping of intra-predicted pictures (Main Profile, CABAC, IPPP, GOP length $= 16$, $QP = 26$, *Foreman*, CIF, 30 fps, 97^{th} picture, intra-predicted).

Rate shaping of intra-predicted pictures In a second experiment, we apply rate shaping to intra-predicted pictures and leave motion-compensated pictures unchanged. We show some screenshots of intra-predicted pictures after rate shaping in Figure 2.10. The visual quality of the pictures mainly depends on the spatial drift. For breakpoint $b = 24$, small artifacts are already visible for a bit rate reduction of barely 2.5%. For breakpoints $b = 12$ and $b = 8$, the loss in visual quality becomes unacceptable. This results in bit rate reductions of 12.1% and 21.5%, respectively. From this, it follows that rate shaping cannot be applied to intra-predicted pictures without taking precautions [29]. This was not the case for rate shaping of MPEG-2 bitstreams where the blocks in intra-coded pictures are coded independently.

2.5.2 Drift analysis for motion-compensated blocks

Most video coding standards provide motion-compensated prediction where the blocks are predicted based on already coded pictures and only the prediction error and the motion data are coded in the bitstream. The motion-compensated prediction is improved in the H.264/AVC video coding standard with features such as variable block sizes, multiple reference pictures, and $1/4$-pixel motion vector accuracy. The motion-compensated pictures may also contain intra-predicted blocks; however, we omit these blocks in the following analysis. The concept of motion-compensated prediction in the H.264/AVC video coding standard is illustrated in Figure 2.11.

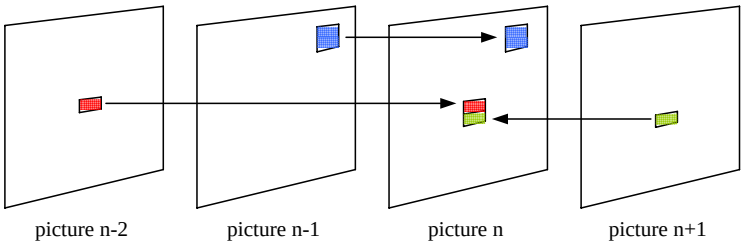


Figure 2.11: Illustration of motion-compensated prediction in H.264/AVC.

The pixel values of motion-compensated blocks are reconstructed at decoder side according to the following formula:

$$\mathbf{x_i} = M_v(\mathbf{p_i}) + \mathbf{e_i}, \tag{2.18}$$

where $\mathbf{x_i}$, $\mathbf{p_i}$, and $\mathbf{e_i}$ represent the decoded pixel values, the reference pixel values, and the residual values, respectively, and M_v denotes motion-compensated prediction with motion data v (i.e., motion vectors and reference picture indices). The motion-compensated prediction only uses decoded pixel values from reference pictures. When rate shaping is applied, the pixel values are reconstructed at decoder side according to the following formula:

$$\hat{\mathbf{x}}_\mathbf{i} = M_v(\hat{\mathbf{p}}_\mathbf{i}) + \hat{\mathbf{e}}_\mathbf{i}, \tag{2.19}$$

where $\hat{\mathbf{x}}_\mathbf{i}$, $\hat{\mathbf{p}}_\mathbf{i}$, and $\hat{\mathbf{e}}_\mathbf{i}$ represent the decoded pixel values, the reference pixel values, and the residual values after rate shaping.

In order to solve the minimization problem, we need to compare the reconstructed pixel values with and without rate shaping. We define an S-dimensional breakpoint vector $\mathbf{B} = \{b_1, \ldots, b_S\}$ where breakpoint b_i only affects the residual data of block $\mathbf{x_i}$ and the number S represents the size of

the optimization window. The minimization problem has the following formulation:

$$\begin{aligned}\mathbf{B}^* &= \arg\min_{\mathbf{B}} \sum_{i=1}^{S} \|M_v(\mathbf{p_i}) + \mathbf{e_i} - M_v(\hat{\mathbf{p}}_\mathbf{i}) - \hat{\mathbf{e}}_\mathbf{i}\| \\ &= \arg\min_{\mathbf{B}} \sum_{i=1}^{S} \|\mathbf{a_i} + \mathbf{e_i} - \hat{\mathbf{e}}_\mathbf{i}\| \\ &\text{Subject to: } R(\mathbf{B}^*) = \sum_{i=1}^{S} R_i(b_i) \leq R_T, \end{aligned} \tag{2.20}$$

where $\mathbf{a_i}$ is the accumulated error due to motion-compensated prediction based on truncation of transform coefficients in reference pictures, $\mathbf{e_i} - \hat{\mathbf{e}}_\mathbf{i}$ is the current-block only distortion due to rate shaping of the prediction error, and $R_i(b_i)$ is the rate required for coding residual block i using breakpoint b_i.

In order to solve the minimization problem, we need to calculate the accumulated error due to motion-compensated prediction. The motion-compensated prediction is a non-linear operator: $M_v(\mathbf{p_i}) - M_v(\hat{\mathbf{p}}_\mathbf{i}) \neq M_v(\mathbf{p_i} - \hat{\mathbf{p}}_\mathbf{i})$. This way, two loops for motion-compensated prediction are required, one that receives the original signal and one that receives the shaped signal. With some lack in arithmetic accuracy, the prediction loops can be collapsed together and the complexity for calculating the accumulated error significantly reduces.

When the distortion metric is the squared error, the minimization problem has the following formulation in the transform domain:

$$\begin{aligned}\mathbf{B}^* &= \arg\min_{\mathbf{B}} \sum_{i=1}^{S} \left(\sum_{j} A_i(\xi(j))^2 + \sum_{j \geq b_i} 2A_i(\xi(j))E_i(j) + \sum_{j \geq b_i} E_i(j)^2 \right) \\ &\text{Subject to: } R(\mathbf{B}^*) = \sum_{i=1}^{S} R_i(b_i) \leq R_T, \end{aligned} \tag{2.21}$$

where $E_i(j)$ is the j^{th} transform coefficient of the i^{th} block of the prediction error in zig-zag scan order, $A_i(j)$ is the j^{th} transform coefficient of the i^{th} block of the accumulated error in zig-zag scan order, and $\xi(\cdot)$ maps run-level positions from the prediction error to actual zig-zag scan positions. The distortion not only involves the shaping error and the accumulated error, but some crossterms as well.

The accumulated error causes temporal drift which typically grows from picture to picture and results in severe degradation of the visual quality near

the end of the GOP. Three factors have an impact on the temporal drift: the size of the GOP, the breakpoint values, and the motion activity in the video scene. Since the visual quality is an important aspect for bit rate reduction, techniques are necessary for reducing the accumulated error.

2.5.3 Drift analysis for intra-predicted blocks

Most video coding standards provide intra coding where the blocks are coded independently using transform, quantization, and entropy coding. The intra coding in the H.264/AVC video coding standard is improved with spatial prediction which provides different prediction directions that use reconstructed pixel values of spatially adjacent blocks. The spatial prediction selects the optimal prediction direction for estimating the current block and only the prediction error and the prediction direction are coded in the bitstream. The concept of intra 4×4 prediction in the H.264/AVC video coding standard is illustrated in Figure 2.12.

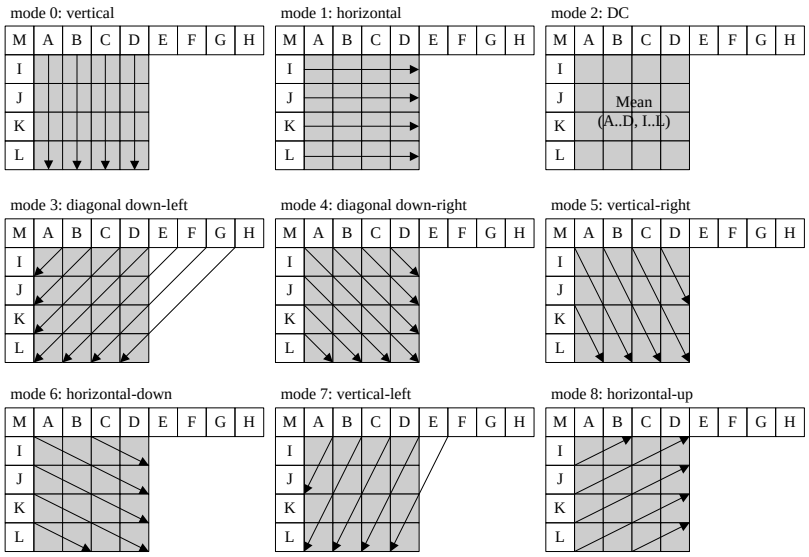


Figure 2.12: Illustration of intra 4×4 prediction in H.264/AVC: surrounding reference pixel A-M are used for spatial prediction according to one of the nine prediction modes.

The pixel values of intra-predicted blocks are reconstructed at decoder side according to the following formula:

$$\mathbf{x_i} = I_m(\mathbf{p_i}) + \mathbf{e_i}, \tag{2.22}$$

where I_m denotes intra prediction with prediction mode m. The intra prediction only uses decoded pixel values from spatially adjacent blocks. When rate shaping is applied, the pixel values are reconstructed at decoder side according to the following formula:

$$\hat{\mathbf{x}}_\mathbf{i} = I_m(\hat{\mathbf{p}}_\mathbf{i}) + \hat{\mathbf{e}}_\mathbf{i}, \tag{2.23}$$

where $\hat{\mathbf{x}}_\mathbf{i}$, $\hat{\mathbf{p}}_\mathbf{i}$, and $\hat{\mathbf{e}}_\mathbf{i}$ represent the decoded pixel values, the reference pixel values, and the residual values after rate shaping.

In order to solve the minimization problem, we need to compare the reconstructed pixel values with and without rate shaping. The minimization problem for intra prediction has an equivalent formulation as for motion-compensated prediction:

$$\begin{aligned}\mathbf{B}^* &= \arg\min_{\mathbf{B}} \sum_{i=1}^{S} \|I_m(\mathbf{p}_\mathbf{i}) + \mathbf{e}_\mathbf{i} - I_m(\hat{\mathbf{p}}_\mathbf{i}) - \hat{\mathbf{e}}_\mathbf{i}\| \\ &= \arg\min_{\mathbf{B}} \sum_{i=1}^{S} \|\mathbf{a}_\mathbf{i} + \mathbf{e}_\mathbf{i} - \hat{\mathbf{e}}_\mathbf{i}\| \\ &\text{Subject to: } R(\mathbf{B}^*) = \sum_{i=1}^{S} R_i(b_i) \leq R_T,\end{aligned} \tag{2.24}$$

where $\mathbf{a}_\mathbf{i}$ is the accumulated error due to intra prediction based on truncation of transform coefficients in spatially adjacent blocks and $\mathbf{e}_\mathbf{i} - \hat{\mathbf{e}}_\mathbf{i}$ is the current-block only distortion due to rate shaping of the prediction error.

In order to solve the minimization problem, we need to calculate the accumulated error due to intra prediction. The intra prediction is also non-linear: $I_m(\mathbf{p}_\mathbf{i}) - I_m(\hat{\mathbf{p}}_\mathbf{i}) \neq I_m(\mathbf{p}_\mathbf{i} - \hat{\mathbf{p}}_\mathbf{i})$. This way, two loops for intra prediction are required, one that receives the original signal and one that receives the shaped signal. With some lack in arithmetic accuracy, the prediction loops can be merged which results in a complexity reduction.

The accumulated error results in spatial drift which is visible for rate shaping. The drift accumulates and propagates from the top-left corner to the bottom-right corner. This propagating effect results from the raster scan order and the prediction directions. The spatial drift is mainly determined by the prediction directions, the breakpoint values, and the amount of texture in the picture. Since the visual quality is very important, techniques are required for reducing or removing the spatial drift.

2.5.4 Drift compensation

The question remains how we are going to restrain the drift propagation. The problem was solved for requantization transcoding where the requantization errors are used for drift compensation. The approach was first demonstrated for motion-compensated blocks [11] and afterwards applied to intra-predicted blocks [40]. The same principle can be applied to rate shaping where we store the shaping error and use these error values in order to compensate in dependent blocks.

Compensation technique The starting point is the cascaded decoder-encoder with thresholding. The thresholding operation serves as the operation which reduces the amount of residual data and results in a video bitstream with reduced bit rate. The cascaded decoder-encoder with thresholding is depicted in Figure 2.13. Since two prediction loops are maintained in the cascaded decoder-encoder with thresholding, this solution is by definition drift-free. The signals in the decoder of the transcoder have superscript 1 while the signals in the encoder of the transcoder have superscript 2. The decoder and the encoder in the transcoder have the same quantizer step size.

The mode and motion data are copied from the incoming to the outgoing video bitstream. This way, a significant reduction in complexity can be achieved. The complexity reduction consists of two things. Firstly, the picture reordering is canceled out which results in a significant reduction of the memory buffers; also the delay which results from the picture reordering is eliminated. Secondly, the coding decisions of the incoming video can be used for the outgoing video bitstream which results in a significant reduction of the computational complexity. As a result, an encoded I picture is again coded as an I picture, a P picture is again coded as a P picture, and a B picture is again coded as a B picture.

There is no need for decoding the pictures since no pixel-domain operations are required. The reconstruction in the decoder can be written as follows: $r_n^1 = p_n^1 + e_n^1$ where p_n^1 represents the prediction and e_n^1 denotes the residuals. The residuals in the encoder can be written as follows: $e_n^2 = r_n^1 - p_n^2$ where r_n^1 is the reconstruction of the decoder and p_n^2 is the prediction in the encoder. The rate shaping operation introduces an error signal d_n^2. Thus, the reconstruction in the encoder can be written as follows: $r_n^2 = e_n^2 + d_n^2 + p_n^2$. From this, it follows that $r_n^2 = r_n^1 + d_n^2$.

The rate shaping solution with compensation is presented in Figure 2.14. We start by taking the difference in the pixel domain between the original signal and the shaped signal. The difference values are stored in memory buffers

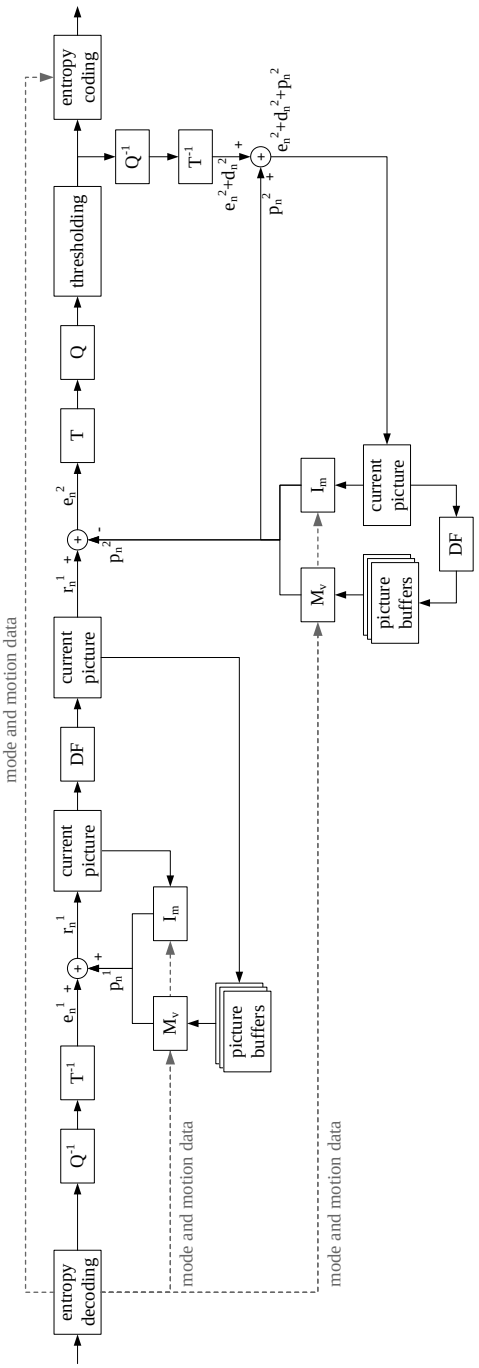


Figure 2.13: Cascaded decoder-encoder with thresholding.

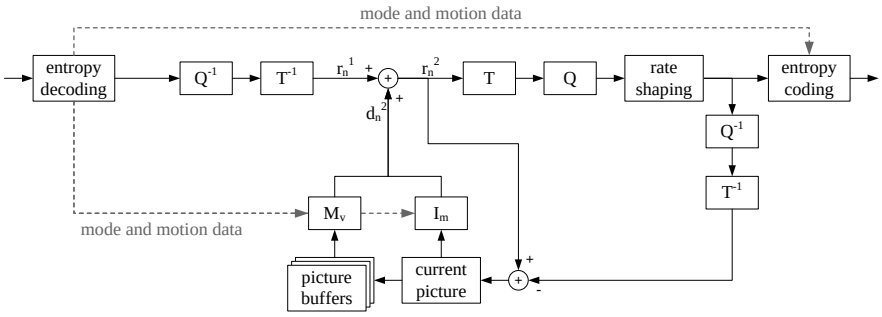


Figure 2.14: Transcoder with reduced complexity.

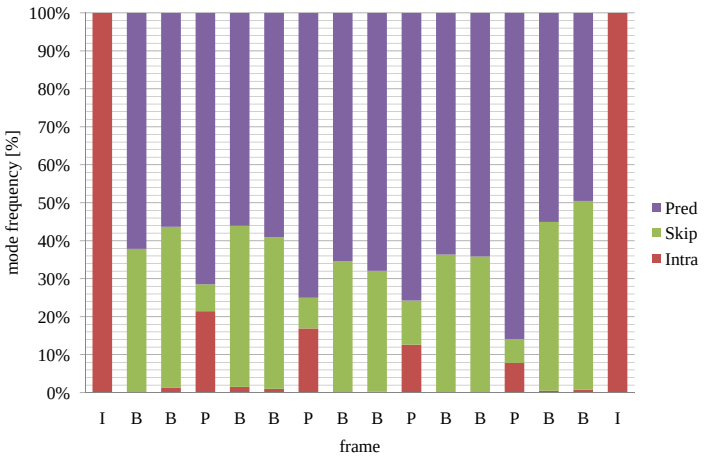
and are used to generate the spatial and temporal compensation in the pixel domain. Since we work on rate shaping errors instead of reconstructed pixel values, no deblocking filter is required. This reduces the computational complexity of the rate shaping architecture.

Independence of compensation techniques The motion-compensated pictures contain both intra-predicted and motion-compensated blocks. For these pictures, spatial and temporal compensation can be used in order to control the drift propagation. The spatial and temporal compensation can be applied independently. This means that spatial compensation is not required in order to apply temporal compensation and vice versa. The only requirement for the compensation is that the error values are available for compensation. The error values of spatially adjacent blocks are necessary for spatial compensation while temporal compensation requires the error values of the reference pictures.

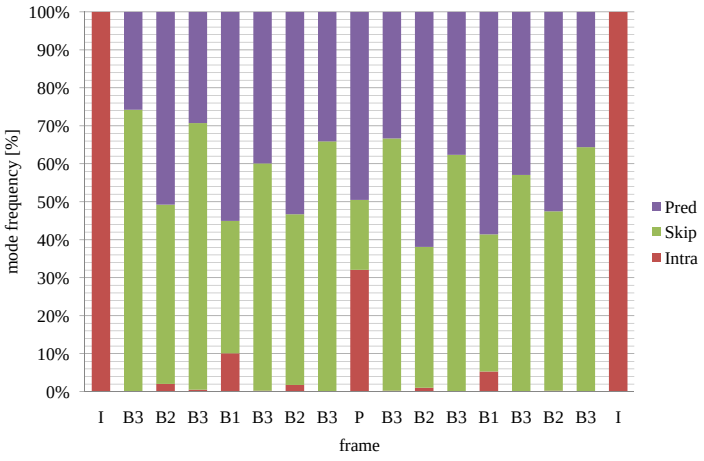
The number of intra-predicted blocks in motion-compensated pictures determines the amount of spatial drift, and the spatial drift has a major impact on the quality degradation of these pictures. Often less than 10% of the blocks are intra-predicted; however, when the temporal prediction fails (e.g., scenes with high motion ...), complete intra-predicted regions occur. The mode frequency is presented for IBBP coding in Figure 2.15(a) and for hierarchical coding in Figure 2.15(b).

For IBBP coding, the temporal prediction fails more frequently for P pictures than for B pictures. This follows from the increased distance to the reference pictures. As a result, the P pictures contain more I macroblocks and less skip macroblocks, while almost no I macroblocks are found in B pictures. For hierarchical coding, similar observations can be made. The distance to the refer-

ence pictures again determines how many I macroblocks are found. Since the distance to the reference pictures is typically higher for the pictures in lower temporal layers, more I macroblocks will be found in these pictures.



(a) IBBP coding



(b) hierarchical coding

Figure 2.15: Mode frequency (Main Profile, $QP_I = 27$, *Foreman* sequence, CIF resolution, 30 fps).

Rate-distortion behavior We would like to know what the impact is on rate and distortion when adding compensation to rate shaping. For rate shaping without compensation, both rate and distortion will be affected by the rate

shaping operation on previous blocks: $R_i(b_1, \ldots, b_i)$ and $D_i(b_1, \ldots, b_i)$. The rate is affected as a result of the context-adaptive property of the entropy coding. The distortion is affected due to the predictive coding for intra-predicted and motion-compensated blocks. This is not the case when we apply compensation, the impact on rate and distortion will be different: $R_i(b_1, \ldots, b_i)$ and $D_i(b_i)$. The rate will change due to the compensation which adds a signal that represents the mismatch in prediction. The compensation rectifies the mismatch in prediction and the breakpoint operations on previous blocks will have no impact on the distortion of the current block.

Bit allocation In the context of MPEG-2 Video rate shaping, different algorithms for bit allocation have been presented. These algorithms are shortly discussed in Section 2.2.2. The dependencies result from motion-compensated prediction. The rate shaping solution does not compensate for the shaping errors which results in temporal drift. Only low-complexity bit allocation is used. So, the causally optimal algorithm and the memoryless algorithm are considered. These algorithms optimize the rate shaping operation picture by picture using rate and distortion. As a result, the rate shaping solution affects all blocks in a picture in a similar way.
Bit allocation is even more important for H.264/AVC. This results from the increased number of dependencies in the video bitstream. Again, only low-complexity bit allocation is used. We would like to know what the outcome is when open-loop rate shaping is applied. For the causally optimal algorithm, both the shaping error and the accumulated error are involved. The spatial drift tends to increase according to the prediction direction, so the blocks in the lower right corner will be affected more compared to the blocks in the upper left corner. For the memoryless algorithm, only the shaping error is considered. As a result of that, the rate shaping will have an equal impact on all parts of the picture. What is the impact of the compensation on the bit allocation? The compensation eliminates the spatial and temporal drift. Only spatial drift is important since the optimization window covers one single picture for the causally optimal algorithm and the memoryless algorithm. The rate shaping will affect all blocks in the picture in a similar way (i.e., according to the rate-distortion metric). The problem corresponds to the case where open-loop rate shaping is combined with the memoryless algorithm.

2.5.5 Rate shaping architecture

The evaluation in Section 2.5.1 has shown that drift propagation is found after rate shaping. The spatial drift is already visible for small bit rate reductions.

Therefore, intra-predicted blocks should be handled carefully. The temporal drift is less dominant compared to the spatial drift. The temporal drift becomes visible when the size of the GOP increases, the breakpoint values increase, or motion activity of the video scene is high. In order to improve the visual quality, we propose to combine rate shaping and compensation techniques. The compensation techniques restrain the drift propagation. These techniques can be applied independently in the spatial and temporal direction. The only requirement is the availability of the shaping errors of the reference blocks.

We combine different rate shaping techniques in order to enable a trade-off between visual quality and complexity. The compensation techniques result in better visual quality but these techniques also require more computational resources. On top of that, the compensation removes the accumulated error but the compensation is not accurate when the prediction loops are collapsed together. As a result, small rounding errors are found in the decoded video. The intra-predicted pictures are the key pictures and have a major impact on the quality of all pictures in the GOP. As a consequence, we only apply rate shaping to the motion-compensated pictures and leave the intra-predicted pictures unchanged.

Since the spatial and temporal compensation can be applied independently in motion-compensated pictures, we can decide what blocks should be compensated. Since spatial drift is more severe than temporal drift and intra prediction is easier than motion-compensated prediction, we prefer to activate spatial compensation in order to suppress the spatial drift. Additionally, we could add temporal compensation in order to further improve the visual quality. This, however, will increase the complexity significantly. This results in the following mixed architectures for rate shaping:

- **RS-OL architecture**
  - No rate shaping for I pictures.
  - Open-loop rate shaping for P pictures and B pictures.
- **RS-SC architecture**
  - No rate shaping for I pictures.
  - Rate shaping with spatial compensation for intra-predicted blocks in P pictures and B pictures.
  - Open-loop rate shaping for motion-compensated blocks in P pictures and B pictures.
- **RS-STC architecture**

- No rate shaping for I pictures.
- Rate shaping with spatial and temporal compensation for P pictures and B pictures.

These architectures will be evaluated in Section 2.7.

2.6 Bit allocation

The problem of bit allocation has been extensively investigated for video coding. The goal of bit allocation for video compression is to minimize the distortion by properly allocating quantizers to the various macroblocks. Bit allocation for rate shaping is similar to bit allocation for video coding. The main differences are the selection of breakpoints instead of quantizers and the relation with rate and distortion. The goal of bit allocation for rate shaping is to minimize the transcoding distortion by properly allocating breakpoints to the various blocks.

In the remainder of this section, we discuss bit allocation for rate shaping in the context of independent coding (e.g., MPEG-2 Video intra coding) and dependent coding (e.g., H.264/AVC intra coding). We elaborate in more detail on issues related to rate shaping and show that rate-distortion optimality cannot be achieved for dependent coding due to high complexity. In order to find a fast solution, an alternative sub-optimal method is presented that allocates one breakpoint for a set of blocks.

Bit allocation for independent coding The minimization problem for independent coding has the following formulation:

$$\begin{aligned} \mathbf{B}^* &= \arg\min_{\mathbf{B}} \sum_{i=1}^{S} D_i(b_i) \\ \text{subject to: } & R(\mathbf{B}^*) = \sum_{i=1}^{S} R_i(b_i^*) \leq R_T. \end{aligned} \tag{2.25}$$

The breakpoint only affects the rate and distortion of one block. The complexity of the minimization problem is exponential in the dependency-tree depth S: 65^S. In order to reduce the complexity, the constrained problem can be converted to an unconstrained problem using Lagrange multipliers [41]:

$$\mathbf{B}^* = \arg\min_{\mathbf{B}} \left\{ \sum_{i=1}^{S} D_i(b_i) + \lambda \sum_{i=1}^{S} R_i(b_i) \right\}. \tag{2.26}$$

The two minimization problems (Equation (2.25) and Equation (2.26)) are not equivalent; however, for some value of λ, their solutions become identical [42]. In order to find the optimal solution $\mathbf{B}^* = \{b_1^*, \ldots, b_S^*\}$, we need to find the optimal value for λ. This can be done using a bisection algorithm [41] or a descent algorithm [43]. The complexity of the minimization problem is significantly reduced: $65S$.

Bit allocation for dependent coding The minimization problem for open-loop rate shaping for dependent coding has the following formulation:

$$\mathbf{B}^* = \arg\min_{\mathbf{B}} \left\{ \sum_{i=1}^{S} D_i(b_1, \ldots, b_i) + \lambda \sum_{i=1}^{S} R_i(b_1, \ldots, b_i) \right\}$$
$$\text{subject to: } R(\mathbf{B}^*) = \sum_{i=1}^{S} R_i(b_1^*, \ldots, b_i^*) \leq R_T, \tag{2.27}$$

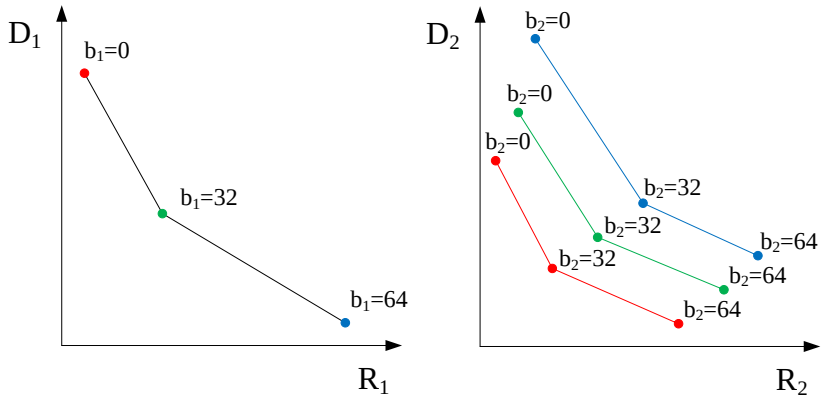
where both rate and distortion depend on the set of breakpoints $\{b_1, \ldots, b_i\}$. The minimization problem for rate shaping with compensation for dependent coding has the following formulation:

$$\mathbf{B}^* = \arg\min_{\mathbf{B}} \left\{ \sum_{i=1}^{S} D_i(b_i) + \lambda \sum_{i=1}^{S} R_i(b_1, \ldots, b_i) \right\}$$
$$\text{subject to: } R(\mathbf{B}^*) = \sum_{i=1}^{S} R_i(b_1^*, \ldots, b_i^*) \leq R_T, \tag{2.28}$$

where the distortion only depends on the breakpoint b_i, while the rate depends on the set of breakpoints $\{b_1, \ldots, b_i\}$.

Both rate shaping with and without compensation are exponentially complex in the dependency-tree depth S: 65^S. The dependency-tree depth can be very large in state-of-the-art video coding. This results from the combination of intra prediction and motion-compensated prediction. In this context, we illustrate the complexity of the bit allocation for rate shaping of one I picture. We consider a video sequence with QCIF resolution that contains 99 macroblocks per picture. Each macroblock contains 16 4×4 blocks which results in 1584 blocks per picture. Since these blocks are coded dependently, the maximum dependency-tree depth is very high. As a consequence, the bit allocation is a too difficult problem when we exhaustively evaluate all rate shaping possibilities. Thus, other techniques are required in order to speed up the rate shaping operation.

Pruning conditions In order to obtain a fast solution for rate shaping, we need to find pruning conditions. Ramchandran *et al.* verified *the monotonicity property* for MPEG-1 bitstreams [44]. This property states that better coding of the independent block leads to more efficient coding of the dependent block due to better prediction.



(a) Rate-distortion curve for independent block

(b) Rate-distortion curve for dependent block

Figure 2.16: The monotonicity property for dependent coding with two blocks and three breakpoints.

In Figure 2.16, the monotonicity property is illustrated for dependent coding with two blocks and three breakpoints. The rate-distortion curve on the left shows results for rate shaping with the breakpoints $b_1 = \{0, 32, 64\}$ for the independent block. The rate-distortion curve on the right shows results for rate shaping with the breakpoints $b_2 = \{0, 32, 64\}$ for the dependent block. The fast solution for rate shaping is based on Trellis coding where sub-optimal rate shaping paths are removed.

Ferguson *et al.* have shown that the monotonicity property does not hold for strongly-dependent coding [45]. The authors show that significant discontinuities are found on the rate-distortion curves. When the monotonicity property is used, rate-distortion optimality cannot be guaranteed. So, this property cannot be used for reducing the complexity of the algorithm for finding the rate-distortion optimal solution.

Clustering A popular technique is clustering where a common breakpoint is used for a set of blocks. This technique was also used for rate shaping of MPEG-2 bitstreams [24]. For rate shaping of MPEG-2 bitstreams, the cluster-

ing does not decrease the complexity of rate shaping by much for any of the rate shaping algorithms.

This is different when we look at clustering for rate shaping of H.264/AVC bitstreams. Here, intra-predicted blocks are found which are strongly connected with the neighbouring blocks due to spatial prediction. As a consequence, the bit allocation is hard to solve due to the increased number of dependencies. So, clustering will significantly decrease the complexity since intra-predicted blocks, which have many spatial dependencies with neighbouring blocks, are treated as a single unit for bit allocation.

In the remainder of the chapter, the rate-distortion results are generated with fixed breakpoints for all blocks in the picture. As a result, no bit allocation is necessary and the power of the rate shaping architecture is measured.

2.7 Performance results

2.7.1 Experimental setup

In order to evaluate the performance of rate shaping, we selected sequences with varying characteristics: *Foreman*, *Paris*, and *Stefan* sequences (CIF resolution, 30 fps). The video sequences are coded using the H.264/AVC reference software (Joint Model version 14.2). The default coding tools are used: five reference pictures, CABAC entropy coding [39], and rate-distortion optimization (RDO) [46].

The sequences are coded using Main Profile (only 4×4 blocks). An IBBP coding structure and a GOP length of 15 pictures is used. Alternatively, a hierarchical coding structure with three temporal layers (periods of seven consecutive B pictures) is used with an intra period of 16 pictures. The coding structures have one thing in common, they all have an instantaneous decoding refresh (IDR) access unit approximately every 500 ms [5]. The periodic insertion of an IDR access unit enables fast random access which is often required in multimedia applications [46]. The sequences are coded with QP_I values (for I slices) 22, 27, and 32, $QP_P = QP_I + 1$ values (for P slices), and $QP_B = QP_I + 2$ values (for B slices). These correspond to the values used in the VCEG common test conditions [47]. In the remainder of this section, we only mention the QP value for the I slices.

The rate shaping architectures are implemented in software which is used

[5] An IDR access unit contains an I picture which is a picture that can be decoded without decoding any previous pictures in the video bitstream. The presence of an IDR access unit indicates that no subsequent pictures in the video bitstream will require reference to pictures prior to the I picture in the IDR access unit.

to generate the shaped video bitstreams. The shaped video bitstreams are generated using increasing breakpoints. By using fixed breakpoints for all blocks, we eliminate the impact of bit allocation in the performance of rate shaping. In order to cover realistic bit rate constraints, we focus on bit rate reductions up to 40%.

2.7.2 Hierarchical coding

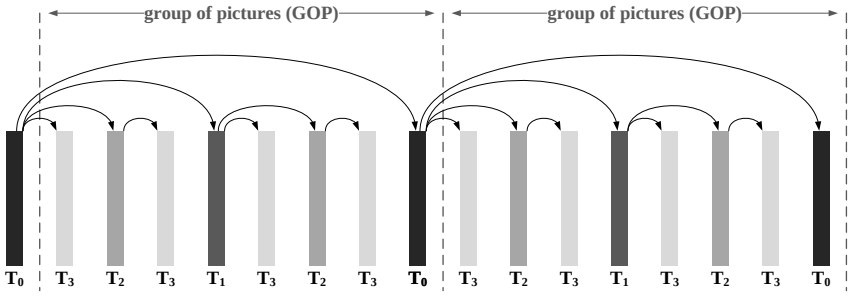
Although IBBP coding is widespread, hierarchical coding is frequently encountered in some scenarios. These coding structures not only improve the coding performance [9] but also permit temporal scalability. In this case, the GOP[6] consists of a sequence of P or B pictures which are organized in different temporal layers T_k with k representing the corresponding temporal layer identifier. The P or B pictures which belong to a certain temporal layer only make use of pictures from lower temporal layers. This way the highest temporal layer can always be removed without impairing the visual quality of the decoded video sequence at reduced frame rate. This is also the technique for temporal scalability in SVC, the scalable extension of the H.264/AVC specification [9, 48]. Hierarchical coding with P and B pictures are presented in Figure 2.17(a) and Figure 2.17(b), respectively.

2.7.3 Rate-distortion results

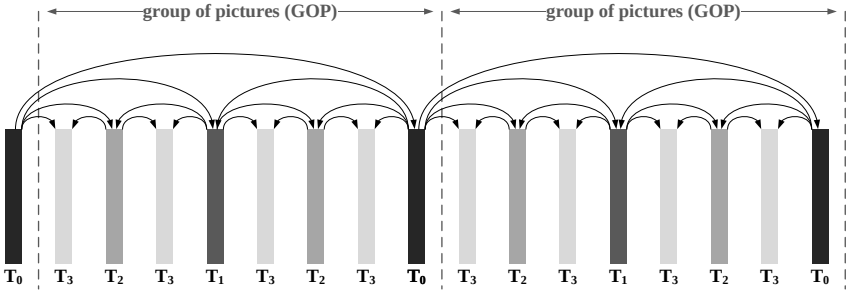
IBBP coding, Main Profile, CIF resolution, 30 fps The rate-distortion results for IBBP coding are shown in Figure 2.18 ($QP_I = 22$) and Figure 2.19 ($QP_I = 32$). Results for recoding are provided as reference. Recoding consists of decoding followed by encoding where the mode and motion data are retained.

The open-loop rate shaping (RS-OL architecture) points out that the visual quality is severely degraded. The quality loss results from both spatial and temporal drift. The spatial compensation (RS-SC architecture) already improves the visual quality for the *Stefan* sequence. Less improvement is found for the *Paris* sequence. The degree of improvement mainly depends on the amount of intra-predicted blocks in motion-compensated pictures. For sequences with a considerable amount of intra-predicted blocks, the compensation may improve the visual quality with 2 to 3 dB. For sequences with almost no intra-predicted

[6] The H.264/AVC video coding standard provides more flexibility for defining reference pictures for motion-compensated prediction. Temporal scalability with dyadic temporal enhancement layers can be very efficiently provided with the concept of hierarchical P or B pictures. In this context, a GOP is defined as the set of pictures between two successive pictures of the temporal base layer together with the succeeding base layer picture [9].



(a) Hierarchical prediction structure with P pictures (structural encoder/decoder delay of zero)



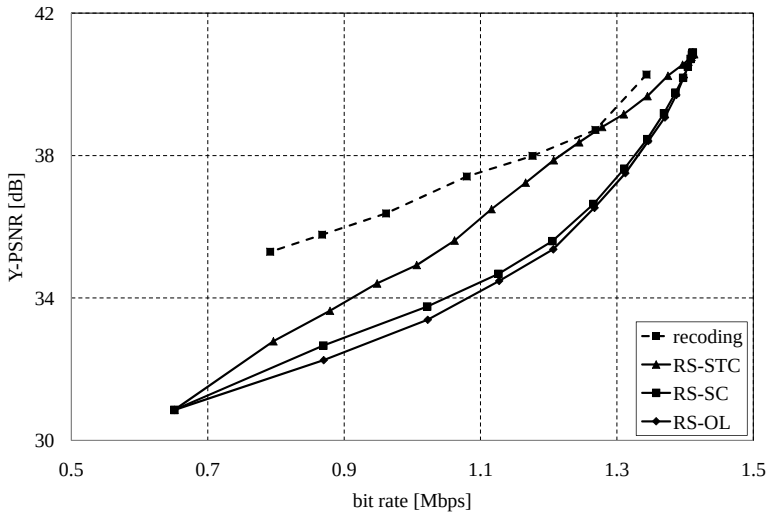
(b) Hierarchical prediction structure with B pictures

Figure 2.17: Hierarchical prediction structures.

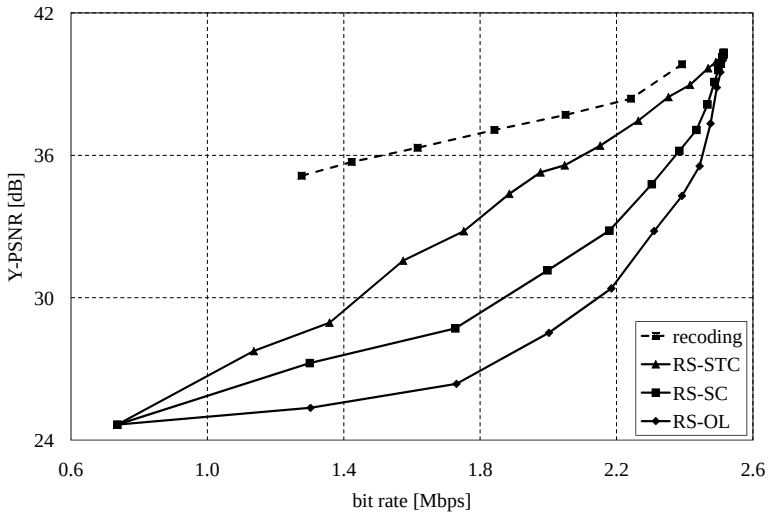
blocks, the compensation will have almost no impact on the visual quality and the drift propagation mainly results from temporal dependencies. We can further improve the rate shaping by adding temporal compensation (RS-STC architecture). We measured an improvement of 2 to 3 dB for sequences with a strong temporal dependence between successive pictures.

The rate-distortion results can be analyzed in a different way. What happens when compensation is added when the same breakpoint is used? When spatial compensation is added, the bit rate stays approximately the same while the visual quality improves. When temporal compensation is added, the visual quality is approximately the same while the bit rate reduces. When all transform coefficients are removed, the compensation is completely canceled out. The three solutions result in one single rate-distortion point.

The PSNR values of the 45 first pictures of the *Stefan* sequence are depicted in Figure 2.20. The results are generated for the sequence *Stefan*

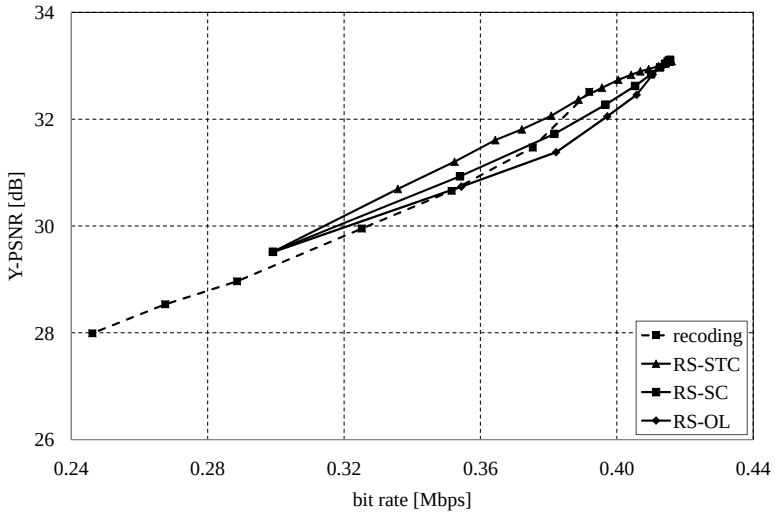


(a) *Paris* sequence

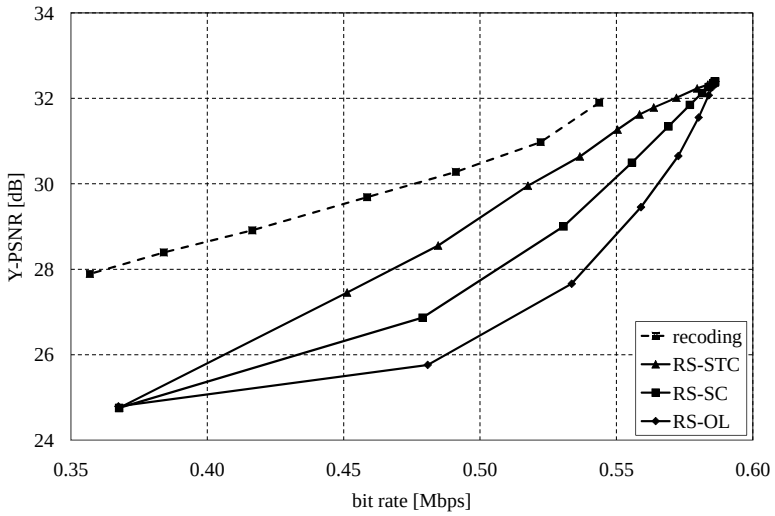


(b) *Stefan* sequence

Figure 2.18: Rate-distortion results for IBBP coding, Main Profile, $QP_I = 22$, CIF resolution, 30 fps.



(a) *Paris* sequence



(b) *Stefan* sequence

Figure 2.19: Rate-distortion results for IBBP coding, Main Profile, $QP_I = 32$, CIF resolution, 30 fps.

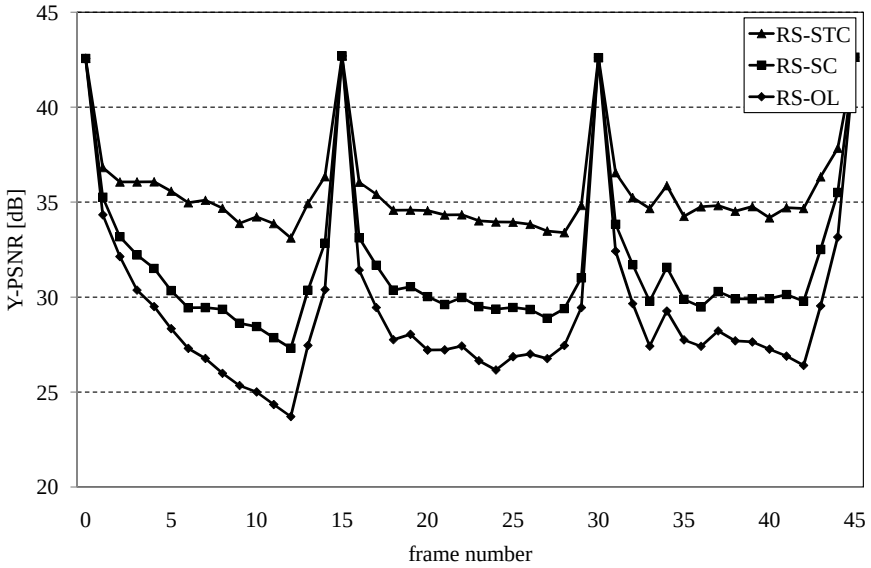
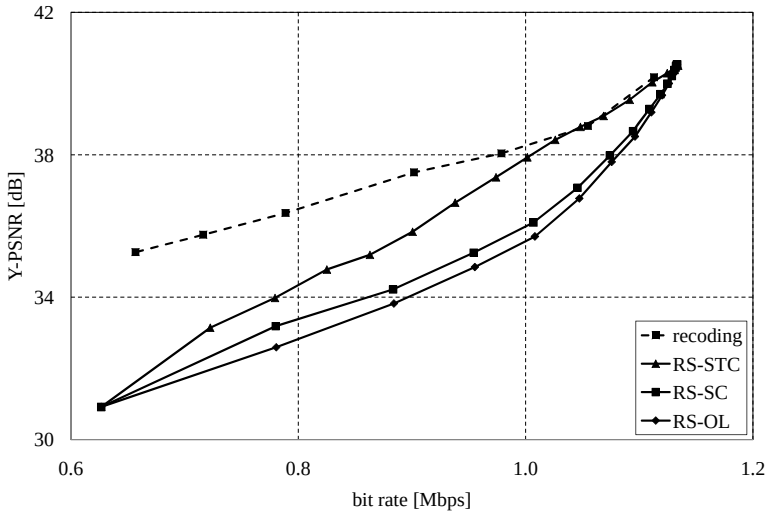


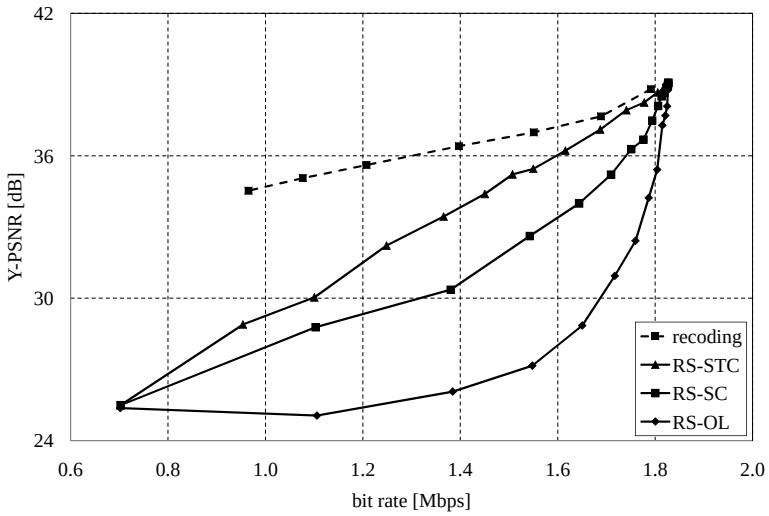
Figure 2.20: PSNR results for IBBP coding, Main Profile, $QP_I = 22$, *Stefan* sequence, CIF resolution, 30 fps: RS-OL ($b = 12$, $28,53$ dB, $2,00$ Mbps), RS-SC ($b = 12$, $31,15$ dB, $2,00$ Mbps), and RS-STC ($b = 24$, $35,28$ dB, $1,98$ Mbps).

with $QP_I = 22$. We selected breakpoint $b = 12$ for the RS-OL architecture ($28,53$ dB, $2,00$ Mbps), breakpoint $b = 12$ for the RS-SC architecture ($31,15$ dB, $2,00$ Mbps), and breakpoint $b = 24$ for the RS-STC architecture ($35,28$ dB, $1,98$ Mbps). The figure shows the evolution of the temporal drift from picture to picture. We can see that the maximum quality loss is up to 10 dB near the end of the first GOP. The PSNR values fluctuate a lot with the period of the GOP which results in the "pumping" or "breathing" artifact commonly known in industry.

Hierarchical coding, Main Profile, CIF resolution, 30 fps The rate-distortion results for hierarchical coding are shown in Figure 2.20 ($QP_I = 22$) and Figure 2.21 ($QP_I = 32$). The main difference between IBBP coding and hierarchical coding is the distance between the reference picture and the picture to be coded. This is so for the B picture in the lowest temporal layer where the distance to the reference pictures is high. As a result, the temporal prediction may fail and more spatial prediction may be expected. This results in more spatial drift when rate shaping is applied. The spatial drift propagates to other temporal layers due to the prediction mechanism in the hierarchical prediction structure. The spatial compensation will be more important which

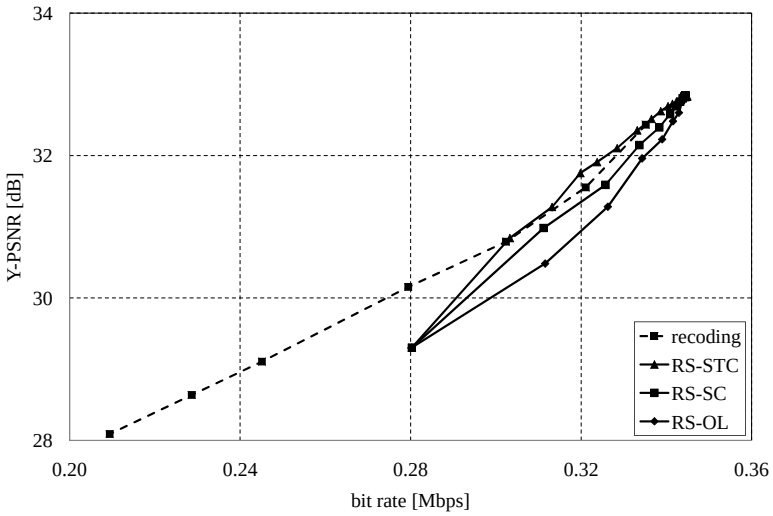


(a) *Paris* sequence

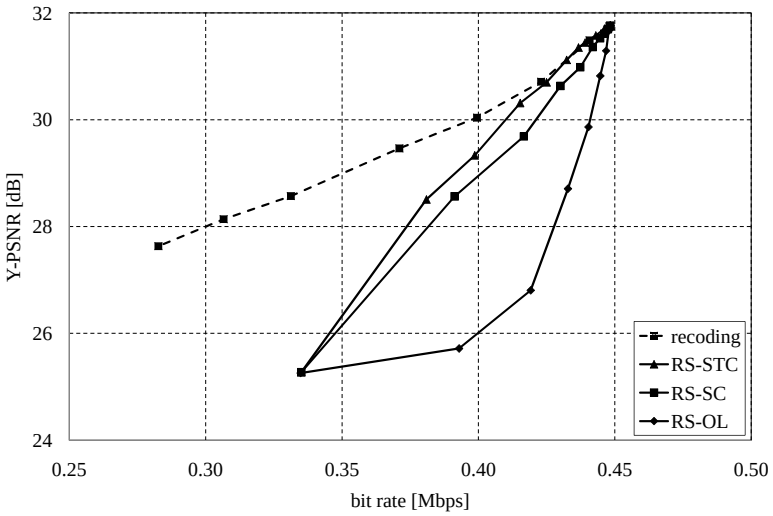


(b) *Stefan* sequence

Figure 2.21: Rate-distortion results for hierarchical coding, Main Profile, $QP_I = 22$, CIF resolution, 30 fps.



(a) *Paris* sequence



(b) *Stefan* sequence

Figure 2.22: Rate-distortion results for hierarchical coding, Main Profile, $QP_I = 32$, CIF resolution, 30 fps.

is clearly visible in the figures.

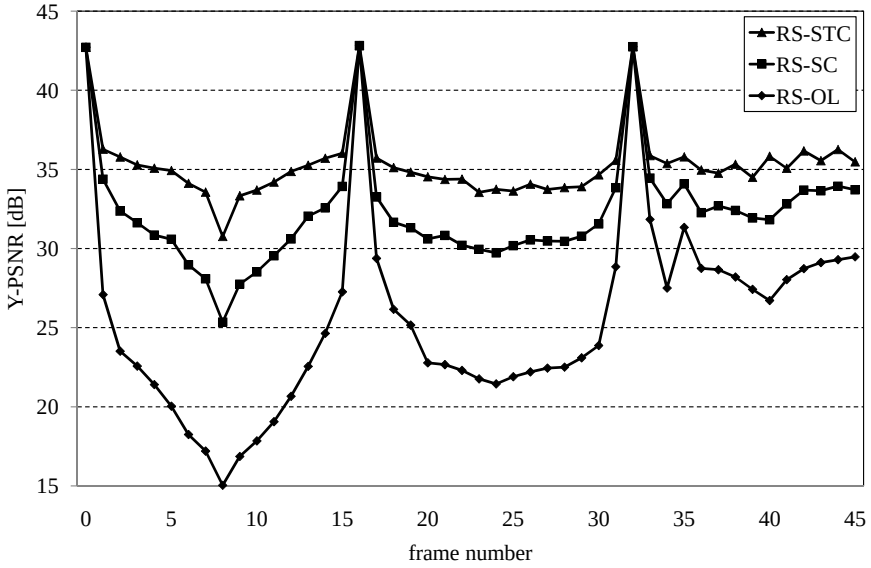
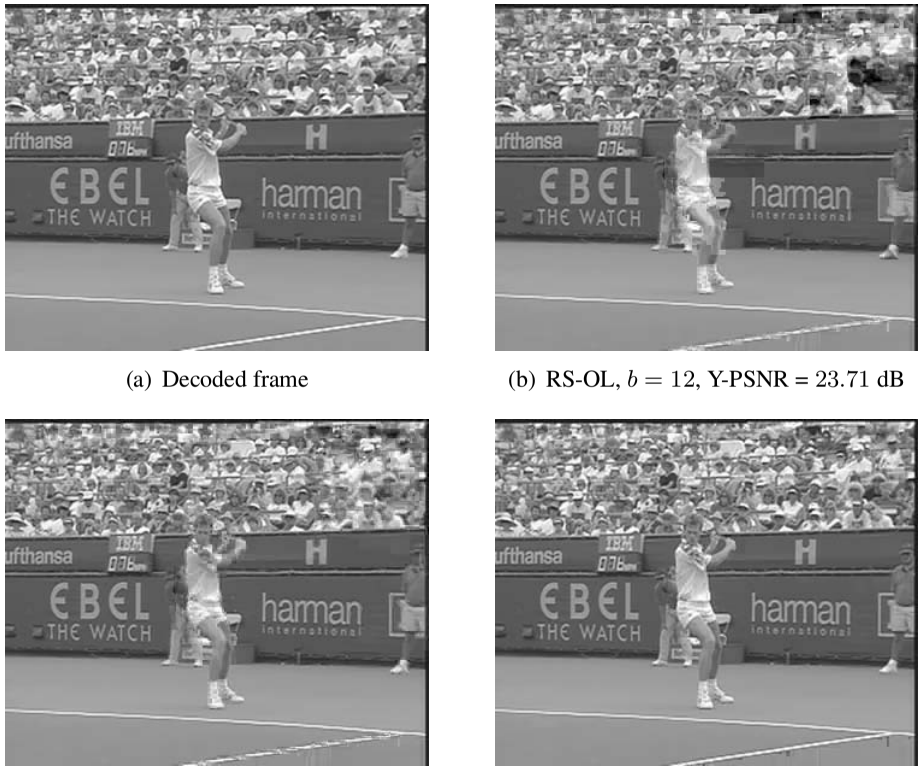


Figure 2.23: PSNR results for hierarchical coding, Main Profile, $QP_I = 22$, *Stefan* sequence, CIF resolution, 30 fps: RS-OL ($b = 12$, $27,16$ dB, $1,55$ Mbps), RS-SC ($b = 12$, $32,62$ dB, $1,54$ Mbps), and RS-STC ($b = 28$, $35,45$ dB, $1,55$ Mbps).

The PSNR values of the 45 first pictures of the *Stefan* sequence are depicted in Figure 2.23. The results are generated for the sequence *Stefan* with $QP_I = 22$. We selected breakpoint $b = 12$ for the RS-OL architecture ($27,16$ dB, $1,55$ Mbps), breakpoint $b = 12$ for the RS-SC architecture ($32,62$ dB, $1,54$ Mbps), and breakpoint $b = 28$ for the RS-STC architecture ($35,45$ dB, $1,55$ Mbps). We can see that the maximum quality loss is more than 15 dB in the middle of the first GOP.

2.7.4 Visual quality

Besides rate-distortion results, visual results clearly show the impact of the drift compensation on the transrated video bitstreams. Figure 2.24 shows visual results for the 13^{th} picture of the *Stefan* sequence for IBBP coding, while Figure 2.25 shows visual results for the 9^{th} picture of the *Stefan* sequence for hierarchical coding.



(a) Decoded frame

(b) RS-OL, $b = 12$, Y-PSNR = 23.71 dB

(c) RS-SC, $b = 12$, Y-PSNR = 27.31 dB

(d) RS-STC, $b = 24$, Y-PSNR = 33.12 dB

Figure 2.24: Visual quality for IBBP coding, Main Profile, $QP_I = 22$, *Stefan* sequence, CIF resolution, 30 fps, 13^{th} picture.

(a) Decoded frame

(b) RS-OL, $b = 12$, Y-PSNR = 15.04 dB

(c) RS-SC, $b = 12$, Y-PSNR = 25.35 dB

(d) RS-STC, $b = 28$, Y-PSNR = 30.76 dB

Figure 2.25: Visual quality for hierarchical coding, Main Profile, $QP_I = 22$, *Stefan* sequence, CIF resolution, 30 fps, 9^{th} picture.

2.7.5 Rate shaping speed

The average processing speed is presented for different architectures in Table 2.1 (rate shaping for IBBP coding) and in Table 2.2 (rate shaping for hierarchical coding). These results are obtained from non-optimized software and serve as an indication of the complexity of the different architectures. Many optimizations are possible in order to speed up the rate shaping architectures; however, it does not belong to the scope of this work. These timing results were generated on a platform with an Intel Xeon X5355 processor and 16 GB RAM in a Microsoft Windows XP environment.

Table 2.1: Average transrating speed [fps] for rate shaping over all possible breakpoints for IBBP coding, Main Profile, *Foreman* sequence, CIF resolution, 30 fps.

	QP_I		
	22	27	32
RS-OL	25.34	27.08	29.80
RS-SC	26.62	28.33	29.75
RS-STC	6.72	6.89	6.99

Table 2.2: Average transrating speed [fps] for rate shaping over all possible breakpoints for hierarchical coding, Main Profile, *Foreman* sequence, CIF resolution, 30 fps.

	QP		
	22	27	32
RS-OL	25.42	27.91	30.32
RS-SC	24.32	27.10	29.59
RS-STC	6.77	7.08	7.18

The breakpoint value only has impact on the number of transform coefficients that needs to be processed by the entropy coding engine. When more transform coefficients are discarded from the video bitstream, less transform coefficients are packed in the video bitstream and the rate shaping operation becomes faster. Therefore, the presented results correspond to averages taken over the rate shaping operations for different breakpoints. The average processing speed is shown in frames per second (fps). The spatial compensation has almost no impact on the rate shaping speed. The temporal compensation severely increases the complexity of the rate shaping solution.

2.7.6 Memory requirements

Another important issue for rate shaping systems are the memory requirements. The memory capacity consists of two parts: 1) memory for input and output buffers and 2) memory for the rate shaping process. Memory buffers are required at the input and output of the rate shaping system. The size of the buffers depends on the level and profile of the video bitstreams. Each video processing system needs these buffers. Besides the input/output memory buffers, memory is required for the rate shaping process. The memory size is mainly determined by the picture buffers, as described in [49]. Open-loop rate shaping requires no picture buffers while spatial compensation needs storage for one picture and temporal compensation needs buffers for the number of reference pictures.

2.8 Conclusions and original contributions

In this chapter, we identified problems for H.264/AVC rate shaping. We showed that the rate shaping solution for MPEG-2 bitstreams requires extensions in order to tackle the improved H.264/AVC coding tools. We cannot simply discard run-level pairs due to the increased complexity of the entropy coding. As a consequence, full entropy decoding and encoding are required. Since different transforms are defined, one single breakpoint is not sufficient. We proposed a solution that derives the actual breakpoint based on the virtual breakpoint. Spatial and temporal drift are introduced as a result of the improved predictive coding. We made an analysis of the drift problem and proposed to use compensation techniques in order to restrain the drift propagation. Afterwards, we discussed architectures that combine compensation with open-loop rate shaping. We also investigated the bit allocation problem and showed that the problem is very complex. We presented clustering as an alternative with an acceptable complexity. The rate-distortion results showed that spatial and temporal compensation result in significant gains. These gains depend on the characteristics of the video bitstreams. We found that spatial compensation is better than temporal compensation when considering complexity and visual quality.

The work that was presented in this chapter can also be found in the following publications:

- Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Rate shaping for H.264/AVC coded video. *Submitted to IEEE Transactions on Multimedia*.

- Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Improved dynamic rate shaping for H.264/AVC video streams. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 1616–1619, San Diego, CA, USA, October 2008.

Chapter 3

Quantizer offset selection for improved requantization

3.1 Rationale and related work

Requantization transrating is often used for reducing the bit rate of video bitstreams according to constraints imposed by client or network. An important question deals with the design of the quantizer in the transrater. The quantizers typically used in image and video coding are controlled by the quantizer step size and the quantizer offset. The quantizer step size defines the width of the quantizer bins, while the quantizer offset controls the size of the dead-zone. The size of the dead-zone is important in image and video coding since typical coding algorithms address most of the effort to efficient coding of zeros [50].

In the past, many efforts have been done in order to optimally change the quantizer in the transrater [28, 51–54]. Only a few contributions are concerned with both the quantizer step size and the quantizer offset. These contributions try to eliminate the requantization errors which follow from the successive quantization in encoder and transrater. This leads to coarse quantization in the transrater compared to the quantization in the encoder. Therefore, we investigate requantization transrating where both the quantizer step size and the quantizer offset are involved.

Werner presented a theoretical analysis of the requantization problem for MPEG-2 Video intra coding [51]. The aim of the work is twofold: 1) present a theoretical analysis of requantization transrating and 2) derive quantization methods for efficient transrating. Two cost functions are evaluated and compared in this paper. The first cost function is optimal in distortion and results in gains up to 1.3 dB compared to the quantizer in the TM-5 reference software encoder for higher bit rates. The second cost function is optimal in rate-

distortion sense and results in gains up to 0.4 dB compared to the quantizer in the TM-5 reference software encoder for the same bit rate. These results are not limited to MPEG-2 Video coding and thus can be adopted by other non-predictive image and video coding algorithms such as JPEG image coding and H.263 intra-only video coding.

Bauschke *et al.* present a heuristic for requantization of JPEG compressed images [52, 53]. They provide a mathematical analysis of the requantization problem based on a Laplace source for the DCT transform coefficients. They derive a heuristic for requantization which determines the optimal quantizer step size. The resulting images are often smaller and have better visual quality compared to blind requantization where no properties of the quantizer in the encoder are taken into account. The approach can be applied to other image and video coding algorithms which are based on transform coding and quantization. The main difference with our approach is that their approach does not consider the quantizer offset.

Bialkowski *et al.* proposed an adapted reconstruction based on the quantizers in encoder and transrater [54]. After requantization, the reconstruction level is centered in each effective quantizer bin. This method results in a substantial gain in visual quality for most pairs of quantizers. The presented technique, however, is not compliant with a specification since both quantizer step sizes need to be signaled in the adapted video bitstream.

Shen investigated the requantization problem and derived conditions for *perfect requantization* [28]. Perfect requantization is achieved when the outcome of requantization is equivalent to the outcome of direct quantization for all possible transform coefficients. This implies conditions on the quantizer design of the transrater. These conditions usually result in coarse quantization which leads to constraints that are too restrictive for practical transrating.

Gendler *et al.* investigated the rounding problem for requantization transrating [55]. They provide a theoretical analysis of the requantization problem by calculating entropy and distortion for different rounding methods. They derive a heuristic for the quantizer design based on the observations from the analysis. The main difference with our approach is that their approach is based on a fixed quantizer offset.

In this chapter, we present a different approach for the requantization problem based on rate-distortion observations. Our approach is based on the rate-distortion behavior of a residual signal which is quantized twice. The outcome of the theoretical analysis is used for deriving a transrating heuristic. This transrating heuristic is used in a transrating solution for H.264/AVC.

The chapter begins with a discussion of the design of a scalar quantizer in Section 3.2. The requantization problem is explained in Section 3.3. We

compare direct quantization and requantization and define different types of requantization errors. We also elaborate on the conditions for perfect requantization. In Section 3.4, the analysis of the effective quantizer, which results from the superposition of the quantizers in encoder and transrater, shows that the effective quantizer has a *periodic property*. Furthermore, it is well-known that a Laplace source satisfies the *memoryless property* for exponentially decreasing functions. Using these properties, we derive expressions for entropy and distortion. These can be used for further investigation of the behavior of the requantization process. We examine the requantization problem for both fine and coarse first-step quantization and derive a heuristic for improving the requantization process. By applying the proposed requantization theory to H.264/AVC in Section 3.5, we show that H.264/AVC requantization can be improved by adapting both quantizer step size and quantizer offset in the transrater. We observe gains of about 1 dB compared to requantization transrating with a fixed quantizer offset. Finally, conclusions are given in Section 3.6.

3.2 Quantizer design

The quantizer is an important coding tool when optimizing the coding performance of image and video coding systems. The quantizer consists of two mapping operations: the classification in the encoder and the reconstruction in the decoder. The optimization of the quantizer is a difficult problem and has been extensively investigated in the literature [56–59]. We provide a short overview of some important characteristics of quantizers in this section.
A quantizer maps a signal with a range of values to a quantized signal with a reduced range of values, allowing for a more efficient representation. A scalar quantizer maps one sample of the input signal to one quantized output value, while a vector quantizer maps a group of samples to a group of quantized output values. In image and video coding, scalar quantization is frequently used.

3.2.1 Scalar quantization

The classification maps a transform coefficient x_i to a quantization index l_i while the reconstruction generates a reconstruction level r_i based on the quantization index l_i. The quantization in a system typically involves application of the classification at the encoder, transmission of the information through the communication channel, and application of the reconstruction at the decoder. In the remainder of this section, we first discuss the reconstruction and the classification. Finally, we elaborate on a popular combination of classification and reconstruction which is frequently found in image and video coding.

Reconstruction A well-known reconstruction is the *nearly-uniform reconstruction quantizer* (NURQ) which uses two control parameters: the quantizer step size Q_1 and the reconstruction offset Δ_1. The NURQ reconstruction can be represented as follows:

$$r_i = sgn(l_i) \cdot Q_1 \cdot (|l_i| + \Delta_1), \tag{3.1}$$

where $sgn(a)$ is the sign of a and $|a|$ is the absolute value of a. A NURQ reconstruction is very popular and is often used in image and video coding standards such as the JPEG standards for image coding and the MPEG-x and H.26x standards for video coding.

An important special case is the *uniform reconstruction quantizer* (URQ), which is defined as a NURQ reconstruction with reconstruction offset $\Delta_1 = 0$:

$$r_i = sgn(l_i) \cdot Q_1 \cdot |l_i| \,. \tag{3.2}$$

Two well-known examples of standards that use the URQ reconstruction are the JPEG image coding standard and the H.264/AVC video coding standard.

Another special case of a NURQ reconstruction is the case where $\Delta_1 = 1/2$, as found in most older standards for image and video coding such as H.261, MPEG-1/2 Video, H.263, MPEG-4 Visual, and JPEG2000.

Classification One classification for the NURQ reconstruction is the *dead-zone plus uniform threshold quantization* (DZ+UTQ). A DZ+UTQ classification can be expressed using two control parameters: the quantizer step size Q_1 and the quantizer offset ϵ_1. The quantizer step size Q_1 defines the width of all quantizer bins except the width of the dead-zone. The width of the dead-zone is controlled by both the quantizer step size Q_1 and the quantizer offset ϵ_1. The DZ+UTQ classification can be represented as follows:

$$l_i = sgn(x_i) \cdot \left\lfloor \frac{|x_i|}{Q_1} + \epsilon_1 \right\rfloor, \tag{3.3}$$

where $\lfloor a \rfloor$ is the largest integer not larger than a.

DZ+UTQ classification and URQ reconstruction It has been shown that a DZ+UTQ classification with a URQ reconstruction provides the optimal solution under a rate-distortion constraint for sources with a Laplace probability distribution function [58, 59].

Until now, the quantizer has been fully determined by the quantizer step size Q_1 and the quantizer offset ϵ_1. Alternatively, the quantizer can be described by the set of decision levels $\{d_i\}$ and the set of reconstruction levels $\{r_i\}$, for

$i \in \mathbb{Z}$. In the figures, the decision levels are denoted by the | symbol and the reconstruction levels are indicated with the $\times$ symbol.
The dead-zone is determined by the decision levels $d_{-1} = (-1 + \epsilon_1)Q_1$ and $d_1 = (1-\epsilon_1)Q_1$. The size of the dead-zone is therefore $2(1-\epsilon_1)Q_1$ for $\epsilon_1 < 1$. Each transform coefficient $x \in (d_{-1}, d_1)$ is mapped onto the reconstruction level $r_0 = 0$. The positive transform coefficients outside the dead-zone are partitioned into intervals $[d_i, d_{i+1})$ where $d_i = (i - \epsilon_1)Q_1$, $i \in \mathbb{Z}_0^+$. Each transform coefficient $x \in [d_i, d_{i+1})$ is mapped onto the same reconstruction level $r_i = iQ_1$. The negative transform coefficients outside the dead-zone are partitioned into intervals $[d_{i-1}, d_i)$ where $d_i = (i + \epsilon_1)Q_1$, $i \in \mathbb{Z}_0^-$. Each transform coefficient $x \in [d_{i-1}, d_i)$ is mapped onto the same reconstruction level $r_i = iQ_1$.
The quantizer characteristic $q(x)$ is shown in Figure 3.1 for the combination of DZ+UTQ classification and URQ reconstruction.

3.2.2 Transform coefficient distribution

A widespread model for transform coefficients found in the literature [60–63] is the generalized Gaussian distribution given by

$$p_{gg}(x) = \frac{\beta}{2\alpha\Gamma(1/\beta)} e^{-(|x|/\alpha)^\beta}, x \in \mathbb{R}, \tag{3.4}$$

where Γ is the gamma function, α is the scale factor, and β is the shape factor. The generalized Gaussian distribution turns into the Laplace distribution or the Gaussian distribution when appropriate values for the parameters are chosen. For $\beta = 1$, the generalized Gaussian distribution becomes the Laplace distribution with parameter $\lambda = 1/\alpha$:

$$p_l(x) = \frac{\lambda}{2} e^{-\lambda|x|}, x \in \mathbb{R}. \tag{3.5}$$

For $\beta = 2$, the generalized Gaussian distribution becomes the Gaussian distribution with variance $\sigma^2 = \alpha^2/2$:

$$p_g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-x^2/2\sigma^2}, x \in \mathbb{R}. \tag{3.6}$$

Reininger *et al.* [60] and Lam *et al.* [63] have shown that the DCT transform coefficients in image and video coding systems can be very well approximated by the Laplace distribution.

Another model which is often used for transform coefficients is the Cauchy-Lorentz distribution with parameter μ given by

$$p_c(x) = \frac{1}{\pi} \frac{\mu}{\mu^2 + x^2}, x \in \mathbb{R}. \tag{3.7}$$

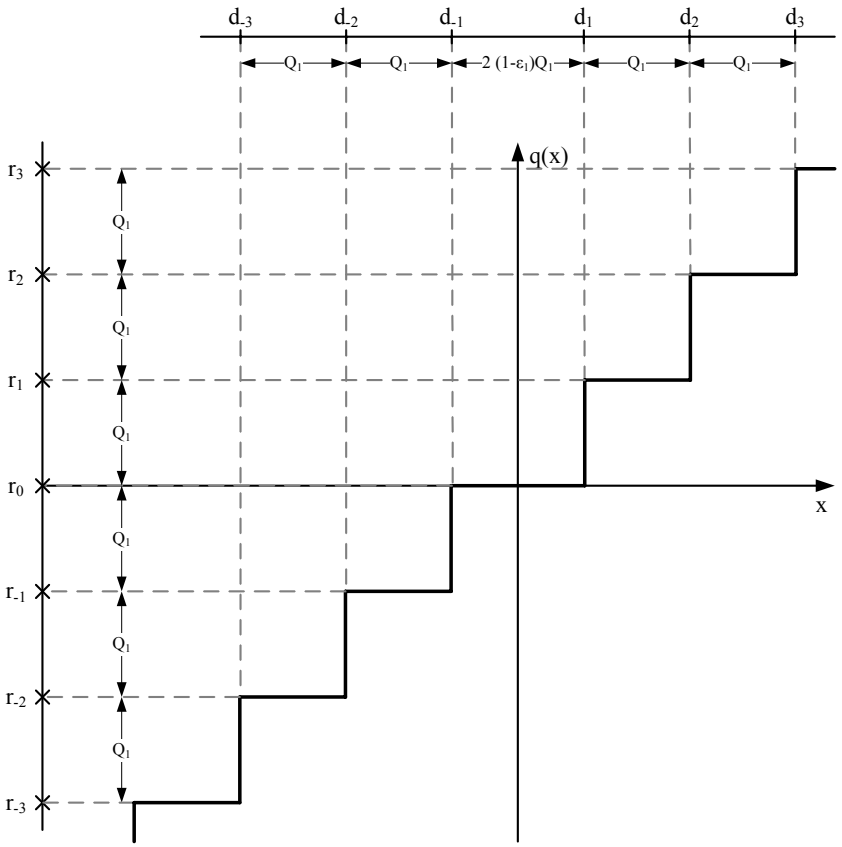


Figure 3.1: Quantizer characteristic for the combination of DZ+UTQ classification and URQ reconstruction.

Kamaci *et al.* [64] used this model to better represent the distribution of the transform coefficients. The distribution of the transform coefficients has fatter tails which are better represented by the Cauchy-Lorentz distribution.

Since the Laplace distribution is a good choice regarding simplicity and accuracy, we found it appropriate to select this distribution. In the following, we elaborate on the quantization of a Laplace source before we tackle the problem of requantization.

3.2.3 Quantization of Laplace source

The transform coefficients are typically uniformly quantized with quantizer step size Q_1 and quantizer offset ϵ_1. Let $P_1(iQ_1)$ be the probability that a

transform coefficient is quantized to the value iQ_1, where $i \in \mathbb{Z}$:

$$P_1(iQ_1) = \begin{cases} \int\limits_{(i-1+\epsilon_1)Q_1}^{(i+\epsilon_1)Q_1} p_l(x)dx = \left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) e^{\lambda i Q_1}, & \text{if } i < 0; \\ \int\limits_{(-1+\epsilon_1)Q_1}^{(1-\epsilon_1)Q_1} p_l(x)dx = 1 - e^{-\lambda(1-\epsilon_1)Q_1}, & \text{if } i = 0; \\ \int\limits_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} p_l(x)dx = \left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) e^{-\lambda i Q_1}, & \text{if } i > 0. \end{cases} \tag{3.8}$$

Note that the probabilities $P_1(iQ_1)$ decrease exponentially when moving away from the value zero. The probability distribution function (pdf) of a Laplace source and the probability mass function (pmf) of a quantized Laplace source are further used for deriving entropy and distortion.
The entropy of the quantized transform coefficients is defined as:

$$H_1 = -\sum_{i=-\infty}^{+\infty} P_1(iQ_1) \log_2 P_1(iQ_1). \tag{3.9}$$

We refer to Appendix C for the derivation of the closed-form expression for entropy:

$$\begin{aligned} H_1 = &- \left(1 - e^{-\lambda(1-\epsilon_1)Q_1}\right) \log_2 \left(1 - e^{-\lambda(1-\epsilon_1)Q_1}\right) \\ &- \left(e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}\right) \frac{e^{-\lambda Q_1}}{1 - e^{-\lambda Q_1}} \\ &\times \left[\log_2 \left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) - \frac{\lambda Q_1}{(1 - e^{-\lambda Q_1}) \ln 2}\right]. \end{aligned} \tag{3.10}$$

Using the squared-error criterion $(x - \hat{x})^2$, the distortion of the quantized transform coefficients is defined as:

$$D_1 = \sum_{i=-\infty}^{+\infty} \int_{L_i}^{U_i} p_l(x) \left(x - iQ_1\right)^2 dx, \tag{3.11}$$

where L_i and U_i are the upper and lower bounds of the quantizer bins.
We refer to Appendix C for the derivation of the closed-form expression for

distortion:

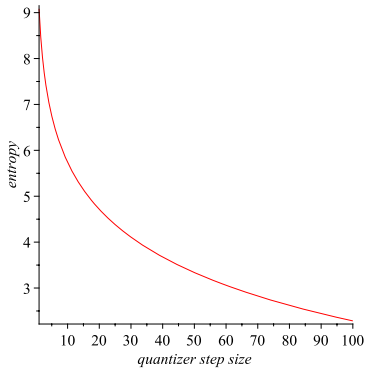
$$\begin{aligned} D_1 =& \frac{2}{\lambda^2} - e^{-\lambda(1-\epsilon_1)Q_1}\left((1-\epsilon_1)^2Q_1^2 + \frac{2(1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2}\right) \\ &+ \frac{2e^{-\lambda Q_1}}{1-e^{-\lambda Q_1}} \times \left[e^{\lambda\epsilon_1 Q_1}\left(\frac{\epsilon_1^2 Q_1^2}{2} - \frac{\epsilon_1 Q_1}{\lambda} + \frac{1}{\lambda^2}\right)\right. \\ &\left. - e^{-\lambda(1-\epsilon_1)Q_1}\left(\frac{(1-\epsilon_1)^2Q_1^2}{2} + \frac{(1-\epsilon_1)Q_1}{\lambda} + \frac{1}{\lambda^2}\right)\right]. \end{aligned} \tag{3.12}$$

The entropy in function of the quantizer step size is shown in Figure 3.2(a) and the distortion in function of the quantizer step size is shown in Figure 3.2(b). When a fine quantizer is used, the data rate is high while there is almost no degradation. When the quantizer becomes coarser, the amount of data is reduced while the quality is decreased. The entropy in function of the quantizer offset is shown in Figure 3.2(c) and the distortion in function of the quantizer offset is shown in Figure 3.2(d). When the quantizer offset increases, the dead-zone decreases which results in a reduced number of zero transform coefficients. The pdf diffuses which results in higher entropy. This effect changes when the dead-zone becomes very small. From this point, the entropy starts to decrease. The distortion is rather high for low and high values of the quantizer offset ϵ_1. In this case, the reconstruction level is located near the boundaries of the quantizer bin and the average quantization error is high. For ϵ_1 values around 0.5, the distortion is significantly lower. The distortion in function of the entropy is shown in Figure 3.2(e). The same results for entropy and distortion are reported by Rajpoot [65].

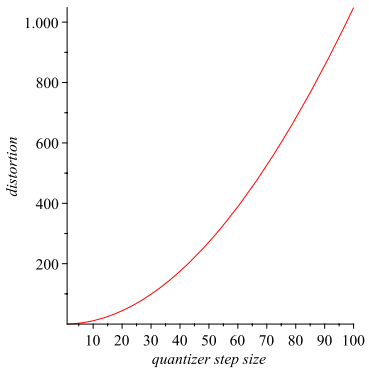
Typically, the optimal quantizer offset is in the range $[0, 0.5]$ for a unimodal pdf. An algorithm for calculating the optimal quantizer parameters is proposed for several generalized Gaussian distributions in [66]. Each quantizer offset results in a unique rate-distortion point. This results from the fact that the transform coefficients are modeled using a continuous source.

3.3 Requantization problem

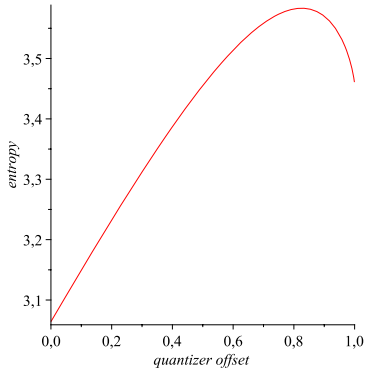
From this point on, we only use a Laplace source, the DZ+UTQ classification, and the URQ reconstruction. We also assume $Q_2 > Q_1$ which is typical for transrating. We start with a comparison of direct quantization and requantization. We define different requantization errors and show for which cases the outcome of direct quantization is equal to the outcome of requantization. Afterwards, we elaborate on perfect requantization where no requantization errors occur. Finally, we point out the typical problems of perfect requantization.



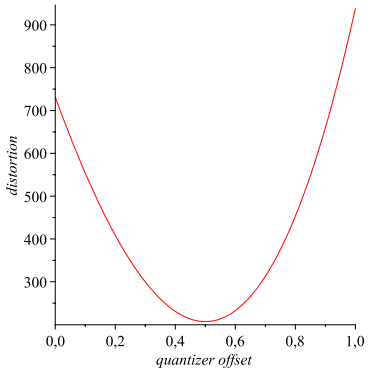
(a) Entropy in function of quantizer step size ($\epsilon_1 = 1/3$).



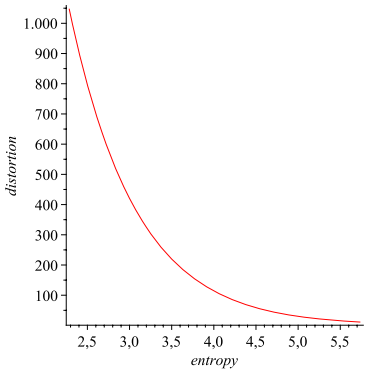
(b) Distortion in function of quantizer step size ($\epsilon_1 = 1/3$).



(c) Entropy in function of quantizer offset ($Q_1 = 50$).



(d) Distortion in function of quantizer offset ($Q_1 = 50$).

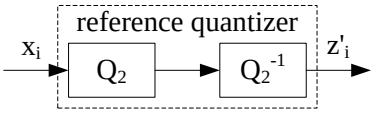


(e) Distortion in function of entropy.

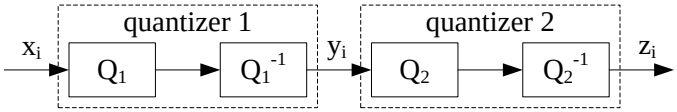
Figure 3.2: Quantizer behavior for Laplace source with parameter $\lambda = 0.01$.

3.3.1 Problem formulation

The requantization problem investigates the difference between direct quantization and requantization. The reference in the comparison is direct quantization which corresponds to the ideal situation where the transrater knows the original signal perfectly. The quantizer in the transrater for requantization uses the same quantizer step size as the quantizer for direct quantization. Direct quantization and requantization are presented in Figure 3.3.



(a) Direct quantization.



(b) Requantization.

Figure 3.3: Direct quantization with the quantizer (Q_2, ϵ_2) versus requantization with the first-step quantizer (Q_1, ϵ_1) and the second-step quantizer (Q_2, ϵ_2).

Direct quantization applies a coarse quantizer to the original transform coefficients. The quantizer characteristic is determined by the quantizer step size Q_2 and the quantizer offset ϵ_2. Let x_i be the transform coefficient and let $q_2'(.)$ be the reference quantizer. Then the outcome of the reference quantizer is denoted as z_i':

$$z_i' = q_2'(x_i) = sgn(x_i) \cdot \left\lfloor \frac{|x_i|}{Q_2} + \epsilon_2 \right\rfloor \cdot Q_2. \tag{3.13}$$

Requantization applies a first-step quantizer in the encoder followed by a second-step quantizer in the transrater [51]. The first-step quantizer characteristic is determined by the quantizer step size Q_1 and the quantizer offset ϵ_1 while the second-step quantizer characteristic is determined by the quantizer step size Q_2 and the quantizer offset ϵ_2. Let x_i be the transform coefficient, and let $q_1(.)$ and $q_2(.)$ be the first-step and second-step quantizers. Then the outcome of both quantizer processes are denoted as y_i and z_i:

$$y_i = q_1(x_i) = sgn(x_i) \cdot \left\lfloor \frac{|x_i|}{Q_1} + \epsilon_1 \right\rfloor \cdot Q_1 \tag{3.14}$$

and

$$z_i = q_2(y_i) = sgn(y_i) \cdot \left\lfloor \frac{|y_i|}{Q_2} + \epsilon_2 \right\rfloor \cdot Q_2. \tag{3.15}$$

Ideally, the outcome of requantization should be identical to the outcome of direct quantization: $z_i = z'_i$. However, this is often not the case. This results from the requantization errors which are introduced by requantization, because requantization has only access to the already quantized transform coefficients instead of the original transform coefficients.

3.3.2 Requantization errors

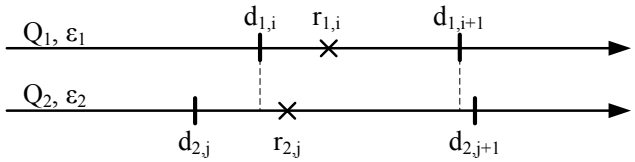
In this chapter, we identify different types of requantization errors. The discussion is restricted to positive transform coefficients and the extension for negative transform coefficients is straightforward due to symmetry of the quantizers. Direct quantization maps transform coefficients in the range $[d_{2,j}, d_{2,j+1})$ to the second-step reconstruction level $r_{2,j}$. Requantization maps transform coefficients in the range $[d_{1,i}, d_{1,i+1})$ to the first-step reconstruction level $r_{1,i}$ in the encoder. Afterwards, the first-step reconstruction level $r_{1,i}$ is mapped to the second-step reconstruction level $r_{2,j}$ in the transrater. Finally, we compare the range of transform coefficients which are mapped to the second-step reconstruction level $r_{2,j}$ and show that one of the following cases will be found as illustrated in Figure 3.4:

- **Case 1:** $d_{2,j} \leq d_{1,i}$ and $d_{1,i+1} \leq d_{2,j+1}$. The quantizer bin of the first-step quantizer completely fits in a quantizer bin of the second-step quantizer. In this case, no requantization error is found for the transform coefficients in the range $[d_{1,i}, d_{1,i+1})$.

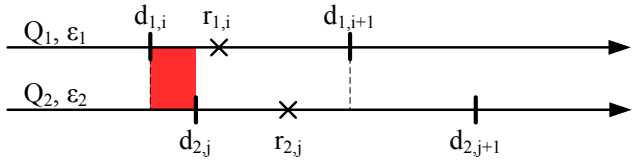
- **Case 2:** $d_{1,i} < d_{2,j}$ and $d_{1,i+1} \leq d_{2,j+1}$. In this case, requantization for the transform coefficients in the range $[d_{1,i}, d_{2,j})$ results in a reconstruction level which is higher (positive error, indicated in red) than the reconstruction level obtained by direct quantization.

- **Case 3:** $d_{2,j} \leq d_{1,i}$ and $d_{2,j+1} < d_{1,i+1}$. In this case, requantization for the transform coefficients in the range $[d_{2,j+1}, d_{1,i+1})$ results in a reconstruction level which is lower (negative error, indicated in blue) than the reconstruction level obtained by direct quantization.

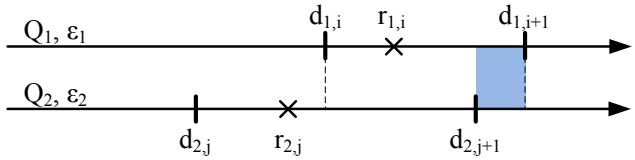
We illustrate this with a simple example shown in Figure 3.5. This example is based on the first-step quantizer ($Q_1 = 40$, $\epsilon_1 = 1/3$) and the second-step quantizer ($Q_2 = 56$, $\epsilon_2 = 1/3$). The figure shows both quantizers and what the difference is between direct quantization and requantization. The red zones



(a) Case 1: no requantization error.



(b) Case 2: positive requantization error.



(c) Case 3: negative requantization error.

Figure 3.4: Identification of requantization errors.

indicate the range of transform coefficients where a positive requantization error is found while the blue zones indicate the range of transform coefficients where a negative requantization error is found. The figure also shows the periodic behavior of the superposition of the quantizers as will be explained in Section 3.4.1.

3.3.3 Perfect requantization

Transrating without requantization errors can be accomplished if the set of decision levels of the second-step quantizer $\{d_{2,j}\}$ forms a subset of the set of the decision levels of the first-step quantizer $\{d_{1,i}\}$ [51]: $\{d_{2,j}\} \subseteq \{d_{1,i}\}$. This is only possible when the quantizers in encoder and transrater are geared to one another.

Previous research has shown that careful selection of the quantizer step size and the quantizer offset in the second-step quantizer results in highly improved transrating performance [28]. The selection of the control parameters for the second-step quantizer is based on the properties of the first-step quantizer.

The cases in which no requantization errors occur are identified as *perfect*

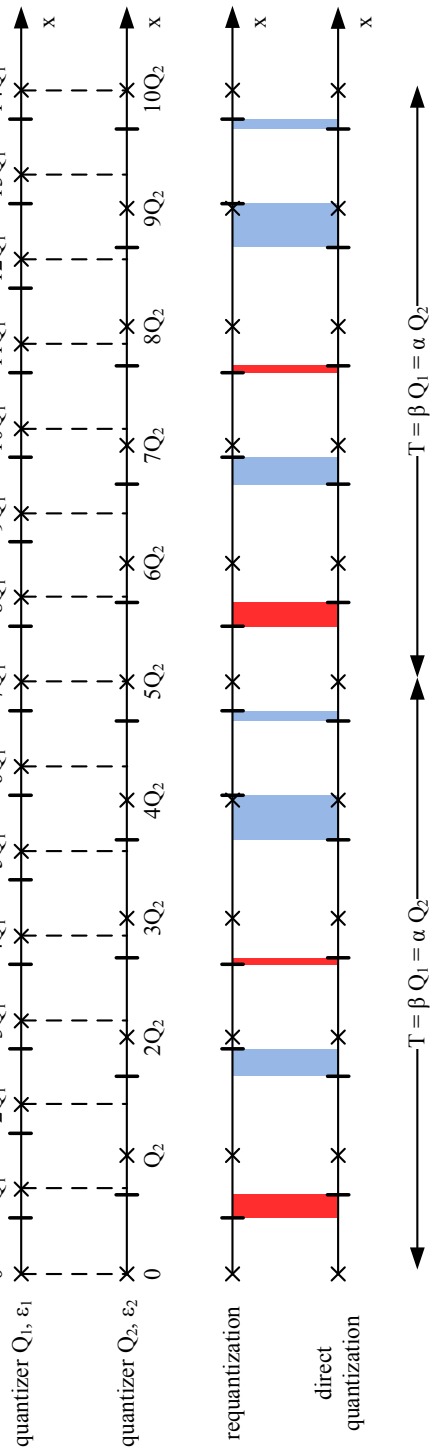


Figure 3.5: Requantization errors for the first-step quantizer ($Q_1 = 40$, $\epsilon_1 = 1/3$) and the second-step quantizer ($Q_2 = 56$, $\epsilon_2 = 1/3$).

requantization. All boundaries of quantizer bins of the second-step quantizer must align with the boundaries of quantizer bins of the first-step quantizer. The condition for perfect requantization can be decomposed in two parts (illustrated in Figure 3.6):

- The boundary of the dead-zone of the second-step quantizer should be aligned with a boundary of a quantizer bin of the first-step quantizer. This can be expressed as follows: $(1 - \epsilon_2)Q_2 = (1 - \epsilon_1)Q_1 + kQ_1$ where $k \in \mathbb{N}$.

- The quantizer step size of the second-step quantizer must be an integer multiple of the quantizer step size of the first-step quantizer. This is represented by the quantizer step size ratio $R_Q = Q_2/Q_1 \in \mathbb{N}_0$.

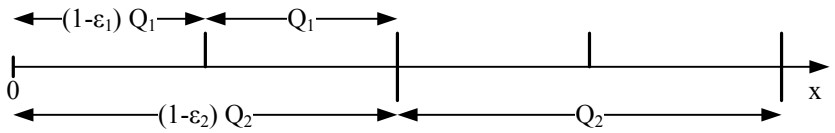


Figure 3.6: Perfect requantization: perfect alignment of quantizer bins.

This is only feasible for quantizer offset combinations (ϵ_1, ϵ_2) for quantizer step size ratio $R_Q = Q_2/Q_1$ as presented in Table 3.1, where $n \in \mathbb{N}$ [28].

Table 3.1: The quantizer step size ratio R_Q for different combinations (ϵ_1, ϵ_2).

ϵ_1 \ ϵ_2	0	1/6	1/4	1/3	1/2
0	n	6n	4n	3n	2n
1/6	-	6n+1	-	-	-
1/4	-	-	4n+1	-	-
1/3	-	6n+2	-	3n+1	-
1/2	-	6n+3	4n+2	-	2n+1

For example, when both quantizer offsets are $1/3$, the quantizer step size ratio should satisfy the condition $R_Q = 3n + 1$, where $n \in \mathbb{N}$. As a result, the quantizer step size ratio should have one of the following values $R_Q \in \{1, 4, 7 \ldots\}$. A quantizer step size ratio $R_Q = 1$ means that the quantizers in encoder and transrater are equivalent (both quantizer step size and quantizer offset). In this case, the requantization does not change the residual data. A

quantizer step size ratio $R_Q > 1$ corresponds to a coarser requantization in the transrater compared to the quantization in the encoder.

Although perfect requantization allows to avoid requantization errors, the outcome is often not suitable for practical implementations due to the following reasons:

- **Quantizer information** The quantizer step size Q_1 and the quantizer offset ϵ_1 are both required for encoding, while only the quantizer step size Q_1 is required for decoding. Typically no extra bits are spent for sending information that is not required at the decoder side. For perfect requantization, both control parameters are necessary in the transrater. As a result, we need to transmit this information in order to derive the control parameters for perfect requantization.

- **Coarse requantization** When mode and motion data are reused from the incoming video bitstream, a significant reduction in complexity is achieved. This approach yields good results when the target bit rate is close to the original bit rate. When the target bit rate strongly deviates from the original bit rate, the quality drops due to sub-optimal use of mode and motion data. These observations were found by Lefol *et al.* [67, 68] when they evaluated mode refinement in the context of intra and inter coding. The conditions for perfect requantization typically result in coarse requantization and coarse requantization results in a high reduction of the bit rate. As a consequence, mode and motion refinement are necessary in order to improve the coding performance when perfect requantization is applied.

3.3.4 Requantization architectures

The requantization problem, as discussed in Section 3.3.1, focuses on open-loop transrating. The open-loop transrater is a good solution for non-predictive coding schemes such as JPEG image coding and MPEG-2 Video intra-only video coding. No drift is introduced by transrating and the visual quality is good. The open-loop transrater, however, does not perform as good for predictive coding schemes. The requantization errors will propagate according to the prediction directions. This can be solved by choosing another architecture for transrating. In the remainder of this section, we give a short overview of three basic architectures for transrating which are most commonly found in literature [69]. These architectures are evaluated in the context of predictive coding. We are interested in the statistics of the signal before requantization and explain which architectures can benefit from an adapted quantizer offset.

Open-loop transrater (OL transrater) The OL transrater applies inverse quantization with quantization parameter QP_1 and requantization with quantization parameter QP_2. This transrater has low complexity; however, severe quality degradation is found. The OL transrater is presented in Figure 3.7.

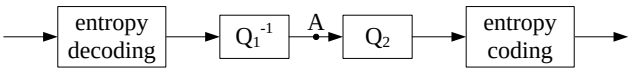


Figure 3.7: The OL transrater.

We are interested in the signal statistics before requantization. The performance of the requantization is also determined by the characteristics of the signal before requantization. This signal consists of the reconstruction levels $r_i = iQ_1$ which are shown in Figure 3.8. This transrating architecture will be further investigated in this chapter in order to know what the impact of quantizer step size and quantizer offset is on the signal statistics.

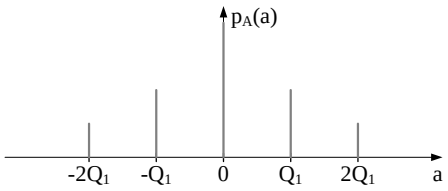


Figure 3.8: Signal statistics in the OL transrater before requantization.

Fast pixel-domain transrater (FP transrater) The FP transrater reconstructs the residual signal in the pixel domain. A closed-loop correction is performed in order to compensate for the requantization drift. This requires intra prediction and motion-compensated prediction. The complexity is increased compared to OL transrater; however, the drift propagation is restrained. Besides the requantization noise, small errors are found which come from nonlinear operations in the coding tools. The FP transrater is depicted in Figure 3.9.

Again, the statistics of the signal just before requantization are important in order to improve the quantization in the transrater. The signal consists of two components: 1) the residual signal and the correction signal. The residual signal is equivalent to the signal found for OL transrating. According to Hait *et al.* [70], the correction signal for intra coding can be modeled using a γ distribution. The combined signal is shown in Figure 3.10. This transrating

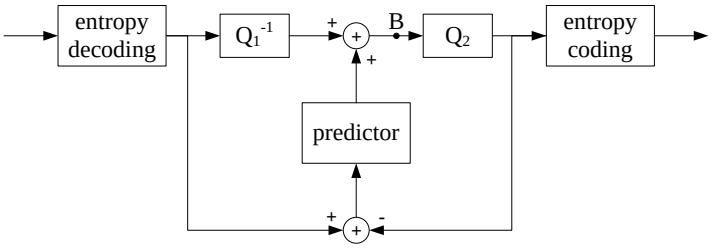


Figure 3.9: The FP transrater.

architecture will not be studied in this chapter. An extension of the approach presented in this chapter could be further investigated.

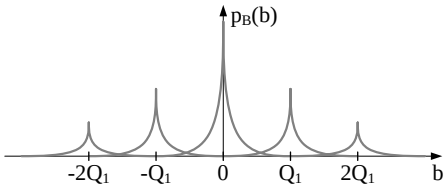


Figure 3.10: Signal statistics in the FP transrater before requantization.

Cascaded pixel-domain transrater (CP transrater) The CP transrater consists of a decoder followed by an encoder where the encoding process uses the mode and motion data from the decoding process. The transrating architecture is by definition drift-free; however, it requires more resources compared to OL and FP transrating. Since the video bitstream is decoded and subsequently encoded, the input signal to the quantizer of the encoder in the transrater will have approximately a Laplace distribution. So, a quantizer offset as found in encoding systems will be appropriate. Therefore, this transrating architecture is not further investigated in this chapter.

3.4 Requantization analysis

The relatively simple signal statistics for OL transrating allow us to find closed-form expressions for entropy and distortion. We make a rate-distortion analysis of the requantization problem and investigate the behavior of the combination of quantizers in encoder and transrater. The combined quantizer is called the effective quantizer. Based on properties of the effective quantizer and the pdf

of the transform coefficients, we derive an efficient method for calculating entropy and distortion of a signal which is quantized twice. Using the theoretical results, we derive a transrating heuristic which can be used during requantization transrating.

3.4.1 Effective quantizer characteristic

The requantization process is a mapping operation which assigns first-step reconstruction levels to second-step quantizer bins. This operation is determined by characteristics of both quantizers.

In order to understand the combined effect of both quantizers, we apply the superposition principle to the first-step and second-step quantizer characteristics. The superposition of both quantizer characteristics results in the *effective quantizer characteristic* [54]. This corresponds to the quantizer which has to be applied to the original transform coefficients in order to obtain the same outcome as the requantization process. The effective quantizer characteristic $q_{\text{eff}}(x)$ is depicted in Figure 3.11 for the first-step quantizer ($Q_1 = 40$, $\epsilon_1 = 1/3$) and the second-step quantizer ($Q_2 = 56$, $\epsilon_2 = 1/3$). The effective quantizer has the following properties:

- The quantizer characteristic has a *non-uniform property* since the length of the quantizer bins correspond to multiples of the quantizer bin of the first-step quantizer. Apart from exceptional cases, the superposition of uniform quantizers typically does not result in a uniform quantizer. This results from the mapping operation in the requantization where one or more first-step reconstruction levels are mapped to one second-step quantizer bin.

- The quantizer characteristic has a *periodic property* with period T. The period is determined by the quantizer step sizes. The length of the period can be described as $T = \beta Q_1 = \alpha Q_2$, where the values α and β are defined as Q_1/gcd and Q_2/gcd and gcd is the greatest common divisor of the two quantizer step sizes.

These properties are illustrated in Figure 3.11. The values $\alpha = 5$ and $\beta = 7$ are computed using $gcd = 8$ for the first-step quantizer ($Q_1 = 40$, $\epsilon_1 = 1/3$) and the second-step quantizer ($Q_2 = 56$, $\epsilon_2 = 1/3$). The non-uniform and periodic properties can be observed in the figure.

The periodic property of the effective quantizer characteristic is an interesting property when analyzing the requantization process. When combining

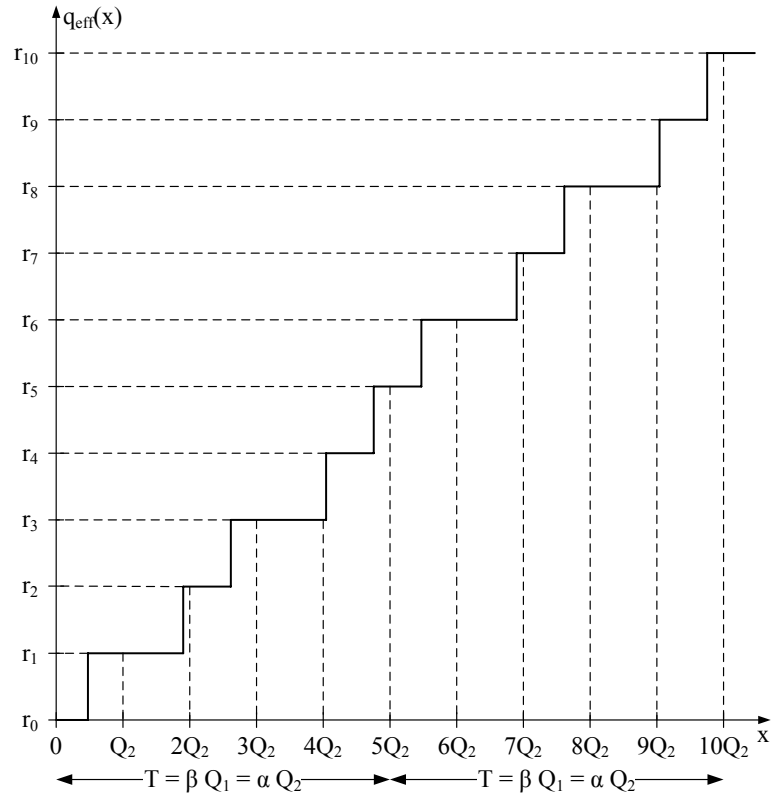


Figure 3.11: Effective quantizer characteristic for the superposition of the first-step quantizer ($Q_1 = 40$, $\epsilon_1 = 1/3$) and the second-step quantizer ($Q_2 = 56$, $\epsilon_2 = 1/3$). The length of the period of the effective quantizer is indicated at the bottom of the figure.

this quantizer property with the *memoryless property* of the exponentially decreasing pdf of the transform coefficients, the requantization analysis can be simplified.

The memoryless property of a Laplace source can be described as follows: $p_E(x) = \lambda e^{-\lambda x}$ for $x \geq 0$. Given the random variable X with exponential distribution, the pdf of $X - t$ is equivalent to the pdf of X for any non-negative value t. The memoryless property can be expressed as $p_E(x+t) = p_E(x)e^{-\lambda t}$. This property is valid for both sides of the Laplace distribution.

The memoryless property and the periodic property will be used in the remainder of this section in order to find expressions for entropy H and distortion D of a double-quantized signal. As a result, only a small part of the effective quantizer characteristic needs to be investigated.

3.4.2 Entropy calculation

After the second-step quantization, the entropy of the transform coefficients is calculated as follows:

$$H_2 = -\sum_{i=-\infty}^{+\infty} P_2(iQ_2)\log_2 P_2(iQ_2). \tag{3.16}$$

The requantization process assigns first-step reconstruction levels to second-step quantizer bins. This mapping process results in the following probabilities:

$$P_2(iQ_2) = \sum_{l=\sigma_i}^{\tau_i} P_1(lQ_1). \tag{3.17}$$

The group of first-step reconstruction levels that are mapped to the same second-step quantizer bin are determined by the values τ_i and σ_i. For coarser requantization, i.e., requantization with $Q_1 < Q_2$, at least one reconstruction level falls in each second-step quantizer bin.
Using the symmetry of the quantizers in encoder and transrater (see Equation (3.14) and Equation (3.15)), the expression for the entropy can be split in different components as follows:

$$H_2 = H_{2,0} + 2\sum_{i=1}^{+\infty} H_{2,i}. \tag{3.18}$$

The entropy component $H_{2,0}$ is determined by the coefficients which fall in the dead-zone after second-step quantization:

$$H_{2,0} = -P_2(0)\log_2 P_2(0) \tag{3.19}$$

The probability of these coefficients is described as

$$P_2(0) = P_1(0) + 2\sum_{l=1}^{\eta} P_1(lQ_1). \tag{3.20}$$

The value η is determined by the number of reconstruction levels of the first-step quantizer which fall in the dead-zone of the second-step quantizer. The same condition can be expressed mathematically as a Diophantine inequality: $\eta Q_1 < (1-\epsilon_2)Q_2 \leq (\eta+1)Q_1$. The expression for entropy $H_{2,0}$ is given in Equation (3.23) using Equation (3.8).

The entropy components $H_{2,i}$ are determined by the probability of the non-zero quantized transform coefficients after second-step quantization. Due

to the memoryless property of the Laplace source and the periodic property of the effective quantizer, the same mapping is found for quantizer intervals $jQ_2, (j+\alpha)Q_2, (j+2\alpha)Q_2, \ldots$ for a fixed value j. As a result, the following substitutions are applied: $i = j + k\alpha$ for $j \in \{0, \ldots, \alpha - 1\}$ and $m = n + k\beta$ for $n \in \{0, \ldots, \beta - 1\}$ with $k \in \{\delta(j), \ldots, +\infty\}$. The function $\delta(j)$ represents the Dirac function which has value 1 for $j = 0$ and value 0 for $j \neq 0$. The expression for the entropy (Equation (3.18)) can be rewritten as

$$H_2 = H_{2,0} + 2\sum_{j=0}^{\alpha-1} H'_{2,j}. \tag{3.21}$$

The entropy components $H'_{2,j}$ describe the entropy as a result of mapping μ first-step reconstruction levels to one second-step quantizer bin. In our work, the value μ is restricted to the set $\{1, 2\}$. This results from the condition on the requantization process: $Q_1 < Q_2 < 2Q_1$. This is an acceptable condition on the quantizers of encoder and transrater since transrating applications often demand for small reductions of the bit rate (as described in Section 3.3.3). The entropy component $H'_{2,j}$ can be formulated as follows:

$$H'_{2,j} = -\sum_{k=\delta(j)}^{+\infty} P_2((j+k\alpha)Q_2)\log_2 P_2((j+k\alpha)Q_2). \tag{3.22}$$

The evaluation of the previous expression for $j \neq 0$ and $j = 0$ results in two closed-form expressions which are given in Equation (3.24) and Equation (3.25), respectively.

The proposed rate-distortion model has a limitation on the range of values for the quantizer offset ϵ_2. The model is only valid for the range $[0, \epsilon_{max})$, where $\epsilon_{max} = Q_1/Q_2$. When the quantizer offset ϵ_2 exceeds the value ϵ_{max}, the calculation is wrong and a correction is required. This is not a serious problem since the value ϵ_{max} falls in the range $[0.5, 1]$ when $Q_2 < 2Q_1$. The problem results from the fact that a reconstruction level of the first-step quantizer is excluded from the calculations.

This is illustrated in Figure 3.12 for the first-step quantizer $Q_1 = 40$ with $\epsilon_1 = 1/3$ and the second-step quantizer $Q_2 = 56$ with $\epsilon_2 = 2/3$ or $\epsilon_2 = 4/5$. In this example, we consider what happens with the first-step reconstruction level $r_{1,6}$ when increasing the quantizer offset ϵ_2. For $\epsilon_2 < \epsilon_{max}$ the reconstruction level is assigned to entropy term H_4, while for $\epsilon_2 > \epsilon_{max}$ the reconstruction level is assigned to entropy term H_5. This is an assignment over the period boundary, so this requires corrections to Equation (3.24) and Equation (3.25). Because this situation is not encountered in practical solutions, we only need the expressions for $\epsilon_2 < \epsilon_{max}$.

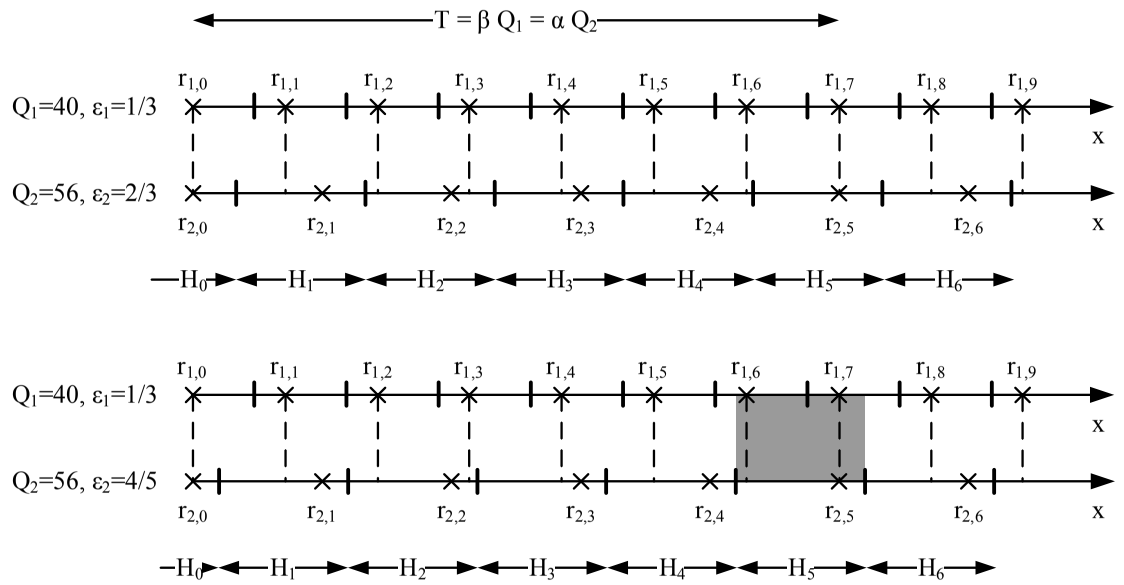


Figure 3.12: Quantizer offset limitation for the first-step quantizer ($Q_1 = 40$, $\epsilon_1 = 1/3$) and the second-step quantizer ($Q_2 = 56$ with $\epsilon_2 = 2/3$) and ($Q_2 = 56$ with $\epsilon_2 = 4/5$). When the quantizer offset ϵ_2 exceeds the value ϵ_{max}, the representation level $r_{1,6}$ from the first period of the first-step quantizer falls in the first bin of the second period of the second-step quantizer.

$$
\begin{aligned}
H_{2,0} =& - P_2(0)\log_2 P_2(0) \\
=& -\left(\left(1-e^{-\lambda(1-\epsilon_1)Q_1}\right)+\left(e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}\right)\sum_{l=1}^{\eta} e^{-\lambda l Q_1}\right) \\
& \log_2\left(\left(1-e^{-\lambda(1-\epsilon_1)Q_1}\right)+\left(e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}\right)\sum_{l=1}^{\eta} e^{-\lambda l Q_1}\right)
\end{aligned} \tag{3.23}
$$

$$
\begin{aligned}
H'_{2,0} =& -\sum_{k=1}^{+\infty} P_2(k\alpha Q_2)\log_2 P_2(k\alpha Q_2) \\
=& -\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\frac{e^{-\lambda\beta Q_1}}{1-e^{-\lambda\beta Q_1}} \\
& \times\left[\log_2\left(\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\right)\right. \\
& \left. -\frac{\lambda\beta Q_1}{(1-e^{-\lambda\beta Q_1})\ln 2}\right]
\end{aligned} \tag{3.24}
$$

$$
\begin{aligned}
H'_{2,j} =& -\sum_{k=0}^{+\infty} P_2((j+k\alpha)Q_2)\log_2 P_2((j+k\alpha)Q_2) \\
=& -\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\frac{1}{1-e^{-\lambda\beta Q_1}} \\
& \times\left[\log_2\left(\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\right)\right. \\
& \left. -\frac{\lambda\beta Q_1 e^{-\lambda\beta Q_1}}{(1-e^{-\lambda\beta Q_1})\ln 2}\right]
\end{aligned} \tag{3.25}
$$

3.4.3 Distortion calculation

After the second-step quantization, the distortion of the transform coefficients is calculated as follows:

$$D_2 = \sum_{i=-\infty}^{+\infty} \int_{L_i}^{U_i} p_l(x) \left(x - mQ_2\right)^2 dx, \tag{3.26}$$

where the values L_i and U_i are the lower and upper bounds of a particular first-step quantizer bin and the value mQ_2 is the second-step reconstruction level. The magnitude-error criterion $|x - \hat{x}|$ is used where x is the original symbol and $\hat{x}$ is the output symbol after requantization.
Using the symmetry of the quantizers in encoder and transrater (see Equation (3.14) and Equation (3.15)), the expression for the distortion can be split in different components as follows:

$$D_2 = D_{2,0} + 2\sum_{i=1}^{+\infty} D_{2,i}. \tag{3.27}$$

The distortion component $D_{2,0}$ represents the distortion of the transform coefficients which are quantized to the value zero by the first-step quantizer. This distortion component is defined as

$$\begin{aligned} D_{2,0} &= \int_{(-1+\epsilon_1)Q_1}^{(1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda|x|} x^2 dx = 2\int_{0}^{(1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} x^2 dx \\ &= \frac{2}{\lambda^2} - e^{-\lambda(1-\epsilon_1)Q_1}\left((1-\epsilon_1)^2 Q_1^2 + \frac{2(1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2}\right). \end{aligned} \tag{3.28}$$

The distortion components $D_{2,i}$ represent the distortion of the non-zero transform coefficients after first-step quantization. Due to the memoryless property of the Laplace source and the periodic property of the effective quantizer, we apply the substitutions $i = j + k\beta$ for $j \in \{0, \ldots, \beta - 1\}$ and $m = n + k\alpha$ for $n \in \{0, \ldots, \alpha - 1\}$. The expression for distortion (Equation (3.27)) can be rewritten as

$$D_2 = D_{2,0} + 2\sum_{j=0}^{\beta-1} D'_{2,j}. \tag{3.29}$$

The distortion component $D'_{2,j}$ can be formulated as follows:

$$D'_{2,j} = \sum_{k=\delta(j)}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - (n+k\alpha)Q_2\right)^2 dx. \tag{3.30}$$

Evaluation of this expression, for both $j \neq 0$ and $j = 0$, are given in Equation (3.31) and Equation (3.32), respectively.

$$
\begin{aligned}
D'_{2,0} =& \sum_{k=1}^{+\infty} \int_{(k\beta-\epsilon_1)Q_1}^{(k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - (n+k\alpha)Q_2\right)^2 dx \\
=& \frac{e^{-\lambda\beta Q_1}}{1-e^{-\lambda\beta Q_1}} \left[e^{\lambda\epsilon_1 Q_1} \left(\frac{(-\epsilon_1 Q_1 - nQ_2)^2}{2} \right.\right. \\
& \left. + \frac{(-\epsilon_1 Q_1 - nQ_2)}{\lambda} + \frac{1}{\lambda^2} \right) \\
& - e^{-\lambda(1-\epsilon_1)Q_1} \left(\frac{((1-\epsilon_1)Q_1 - nQ_2)^2}{2} \right. \\
& \left.\left. + \frac{((1-\epsilon_1)Q_1 - nQ_2)}{\lambda} + \frac{1}{\lambda^2} \right) \right]
\end{aligned} \tag{3.31}
$$

$$
\begin{aligned}
D'_{2,j} =& \sum_{k=0}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - (n+k\alpha)Q_2\right)^2 dx \\
=& \frac{1}{1-e^{-\lambda\beta Q_1}} \left[e^{-\lambda(j-\epsilon_1)Q_1} \left(\frac{((j-\epsilon_1)Q_1 - nQ_2)^2}{2} \right.\right. \\
& \left. + \frac{((j-\epsilon_1)Q_1 - nQ_2)}{\lambda} + \frac{1}{\lambda^2} \right) \\
& - e^{-\lambda(j+1-\epsilon_1)Q_1} \left(\frac{((j+1-\epsilon_1)Q_1 - nQ_2)^2}{2} \right. \\
& \left.\left. + \frac{((j+1-\epsilon_1)Q_1 - nQ_2)}{\lambda} + \frac{1}{\lambda^2} \right) \right]
\end{aligned} \tag{3.32}
$$

3.4.4 Rate-distortion analysis

The objective of the rate-distortion analysis is to allow us to choose optimal parameters Q_2 and ϵ_2 in rate-distortion sense. Using the expressions for entropy and distortion, we plot several rate-distortion graphs for different configurations. The transform coefficients are modeled with Laplace parameter $\lambda = 0.01$. We refer to Section 3.5.2 for more information about how to determine an appropriate value for this parameter.

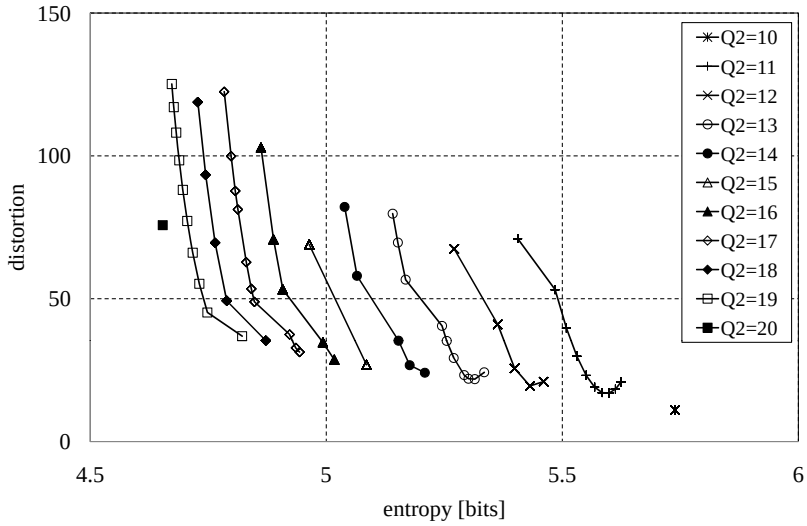
The first-step quantization is performed with quantizer step sizes $Q_1 \in \{10, 20, 50, 100, 200, 500\}$ and a fixed quantizer offset $\epsilon_1 = 1/3$. The quan-

tizer step sizes are in accordance with the range of quantizers available in the H.264/AVC coding standard. We refer to Appendix B for more information about the range of quantizers in H.264/AVC and the mapping between the quantization parameter and the quantization step size. Most multimedia applications use quantization parameters in the range from 28 to 40, or equivalently, quantizer step sizes in the range from 16 to 64. The optimal quantizer offset for a Laplace source depends on the Laplace parameter λ and the design of the quantizer and the entropy coder. An algorithm for calculating the optimal control parameters for the quantizer is proposed for several generalized Gaussian distributions in [66]. In this work, the quantizer offset has a fixed value $\epsilon_1 = 1/3$. The H.264/AVC reference software provides flexibility regarding the quantizer offset. The quantizer offsets change based on the prediction mechanism of the macroblock and the position of the transform coefficient in the residual block. The default values for the quantizer offsets are approximately $1/3$ or $1/6$.

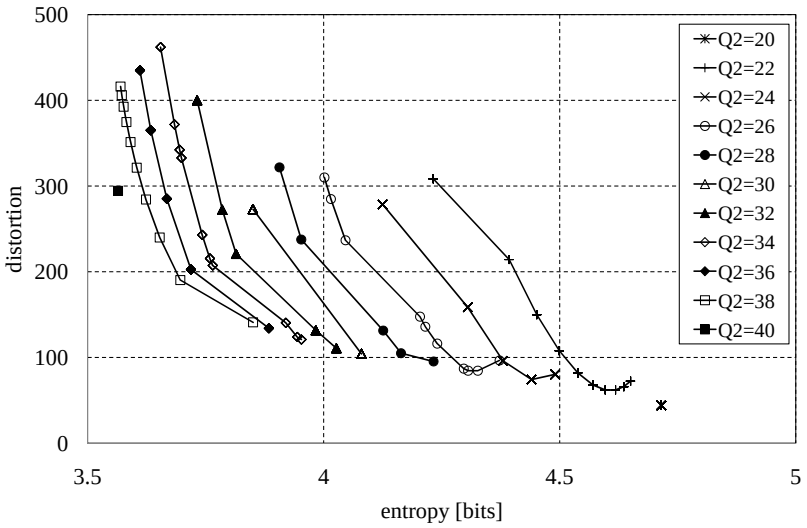
The second-step quantization is performed with quantizer step size $Q_2 = Q_1 + g\Delta Q$ where $\Delta Q = Q_1/10$ and $g \in \mathbb{N}_0$. The quantizer offset ϵ_2 can take values from the range $[0, \epsilon_{max})$, where the value ϵ_{max} depends on the quantizers in encoder and transrater. We refer to the last part of Section 3.4.2 for more information about the range of values for the quantizer offset ϵ_2.

Rate-distortion results The rate-distortion results are presented in Figure 3.13 for fine quantization in the encoder, Figure 3.14 for medium quantization in the encoder, and Figure 3.15 for coarse quantization in the encoder. Each curve in the rate-distortion graph represents requantization with one second-step quantizer step size. The rate-distortion points are obtained by changing the quantizer offset with a fixed quantizer step size. We connect the rate-distortion points for clarity. The number of points depends on the quantizer step sizes used for encoding and transrating. In contrast to encoding, a range of quantizer offsets may generate one rate-distortion point. This results from the fact that the distribution of the input signal of the transrater is not continuous. The distribution of the input signal is modeled using a pmf instead of a pdf as elaborated on in Section 3.3.4. Therefore, each transrating operation which leads to the same mapping operation in the transrater has the same entropy H and distortion D.

Model verification We validate the proposed rate-distortion model in Matlab. We generate an array with samples which result from a Laplace source. We apply the first-step quantizer with quantizer steps size Q_1 and quantizer offset ϵ_1 and successively apply the second-step quantizer with

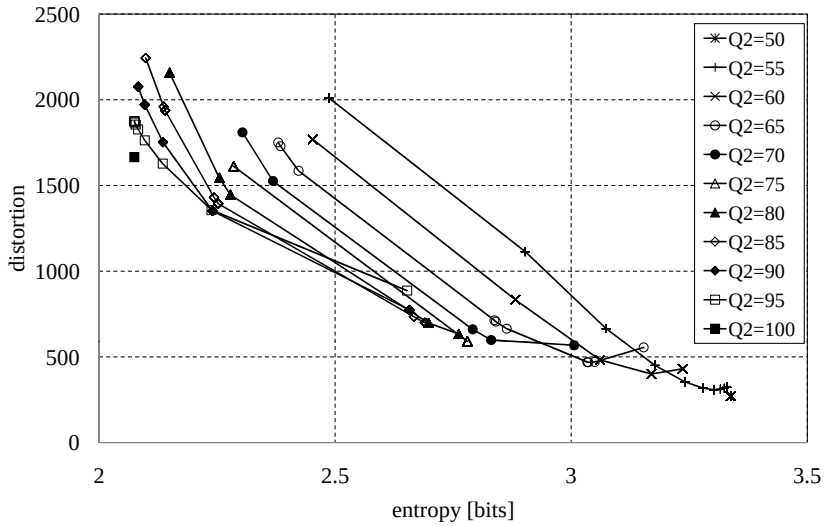


(a) $\lambda = 0.01, Q_1 = 10, \epsilon_1 = 1/3$

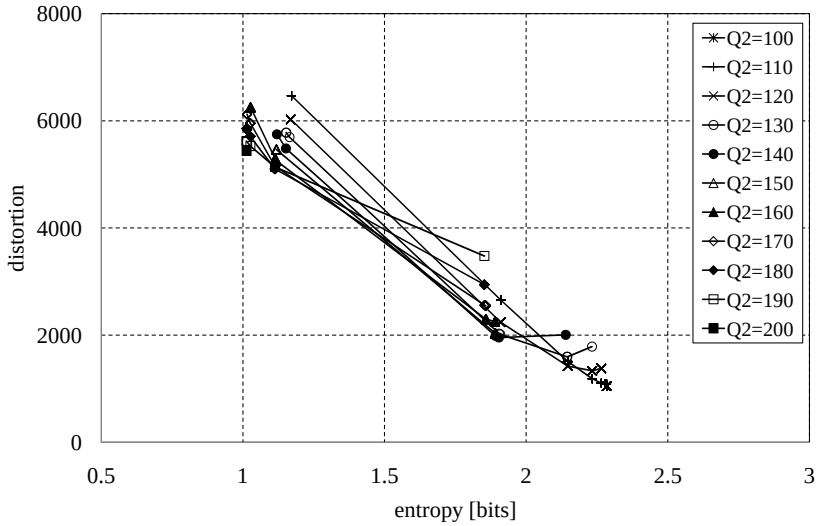


(b) $\lambda = 0.01, Q_1 = 20, \epsilon_1 = 1/3$

Figure 3.13: Theoretical rate-distortion results for fine quantization in the encoder.

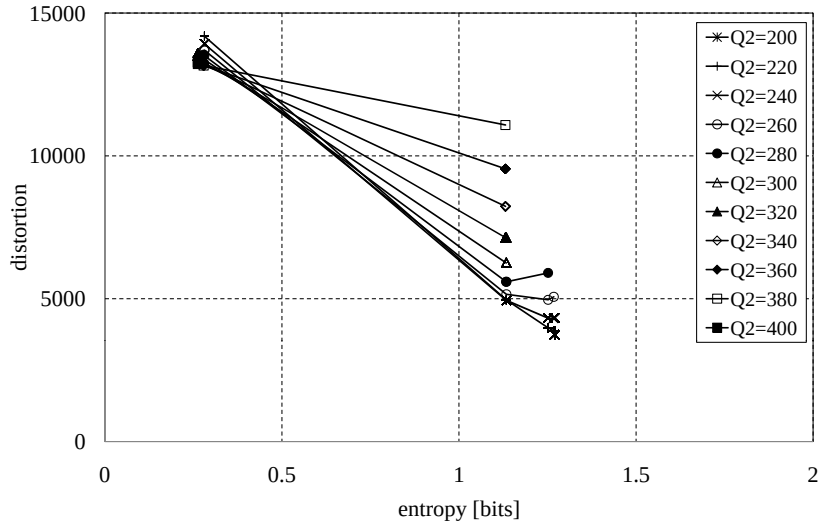


(a) $\lambda = 0.01$, $Q_1 = 50$, $\epsilon_1 = 1/3$

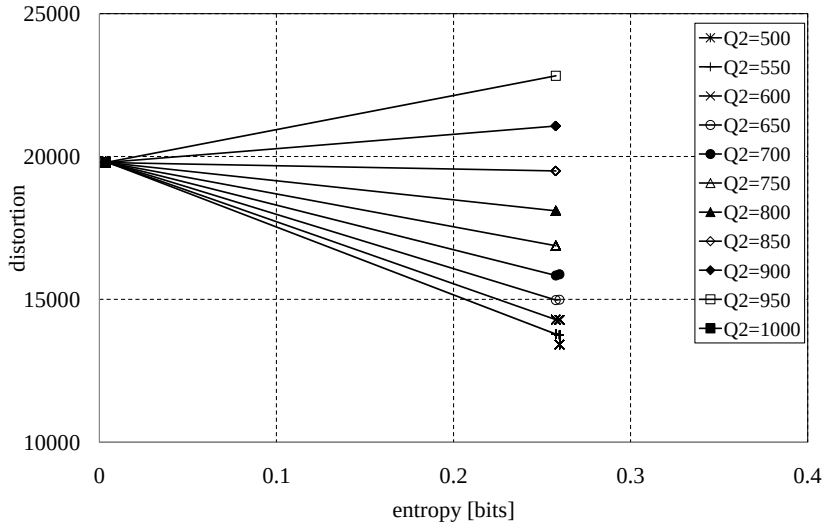


(b) $\lambda = 0.01$, $Q_1 = 100$, $\epsilon_1 = 1/3$

Figure 3.14: Theoretical rate-distortion results for medium quantization in the encoder.



(a) $\lambda = 0.01$, $Q_1 = 200$, $\epsilon_1 = 1/3$



(b) $\lambda = 0.01$, $Q_1 = 500$, $\epsilon_1 = 1/3$

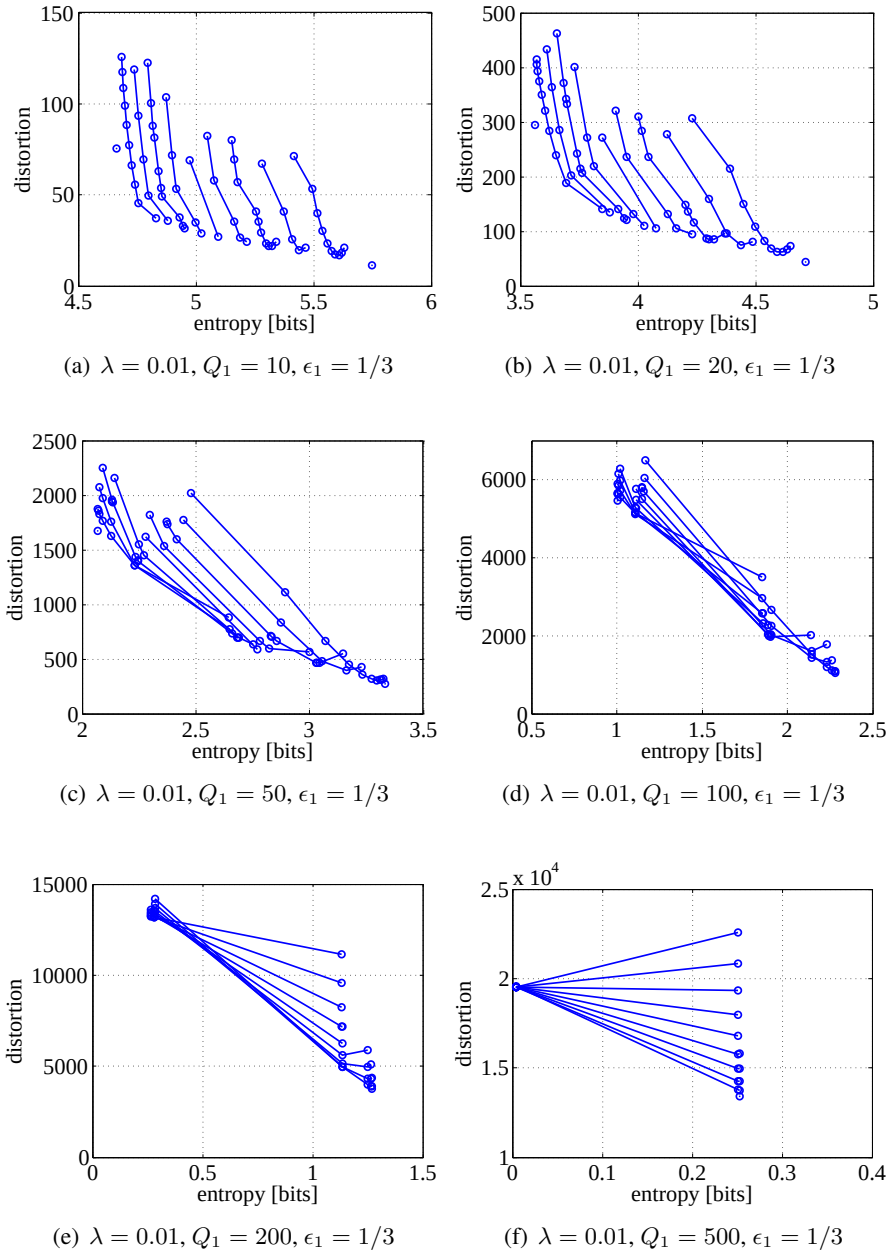
Figure 3.15: Theoretical rate-distortion results for coarse quantization in the encoder.

quantizer step size Q_2 and quantizer offset ϵ_2. We show the rate-distortion results in Figure 3.16.

Convex hull The theoretical model allows us to improve the requantization. In order to find a good solution for requantization, we need to operate on the *convex hull*. The convex hull corresponds with the set of optimal rate-distortion points [31] [24]. The rate-distortion points which lie above the convex hull, and hence are not optimal, should not be considered for requantization. As we can see from the theoretical model, the convex hull covers other points for different configurations. Since the Laplace parameter λ and the first-step quantizer have a major impact on the rate-distortion results, we distinguish two cases: low λQ_1 product and high λQ_1 product. These cases are discussed in the remainder of this section.

Low λQ_1 product For fine quantization in the encoder, the pmf of the signal consists of many small probability peaks which decrease slowly for higher reconstruction levels. This signal has high entropy and low distortion and the pmf is presented in Figure 3.17(a). A big number of probability peaks and the freedom in the design of the second-step quantizer allow for a broad range of target bit rates to be achieved. Results are shown in Figure 3.13 for the first-step quantizers $Q_1 = 10$ and $Q_1 = 20$, respectively. This typically corresponds with fine quantization in the encoder (i.e., high-quality video coding). The convex hull consists of the rate-distortion points generated with increasing second-step quantizer step size Q_2 and a fixed second-step quantizer offset.

High λQ_1 product For coarse quantization in the encoder, the pmf of the signal consists of a small number of peaks of significant probability. This results from the fast decay of the probability for higher reconstruction levels. This signal has low entropy and high distortion and the pmf is presented in Figure 3.17(b). The number of mapping operations in the transrater is limited. Only a few reconstruction levels will have a noticeable impact on the requantization. The probability peaks associated with higher reconstruction levels have almost no impact on the entropy and distortion of the requantized signal. When we ignore the higher reconstruction levels, only a small number of mapping operations is possible. Results are shown in Figure 3.15 for the first-step quantizers $Q_1 = 200$ and $Q_1 = 500$, respectively. The convex hull consists of the rate-distortion points generated with second-step quantizer step size $Q_2 = Q_1 + 1$ and decreasing second-step quantizer offset ϵ_2. Further increasing the quantization can be realized by increasing ΔQ with a fixed



(a) $\lambda = 0.01$, $Q_1 = 10$, $\epsilon_1 = 1/3$

(b) $\lambda = 0.01$, $Q_1 = 20$, $\epsilon_1 = 1/3$

(c) $\lambda = 0.01$, $Q_1 = 50$, $\epsilon_1 = 1/3$

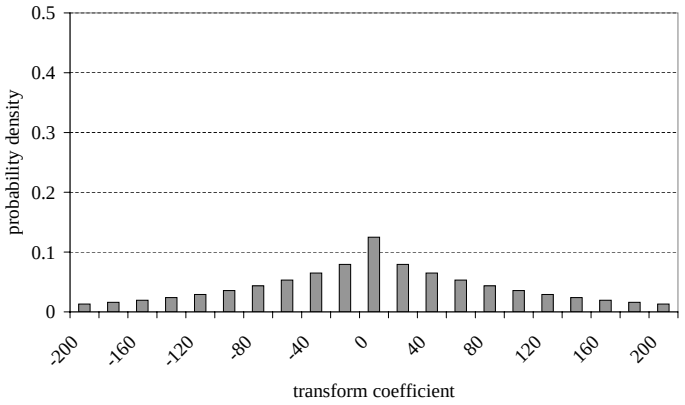
(d) $\lambda = 0.01$, $Q_1 = 100$, $\epsilon_1 = 1/3$

(e) $\lambda = 0.01$, $Q_1 = 200$, $\epsilon_1 = 1/3$

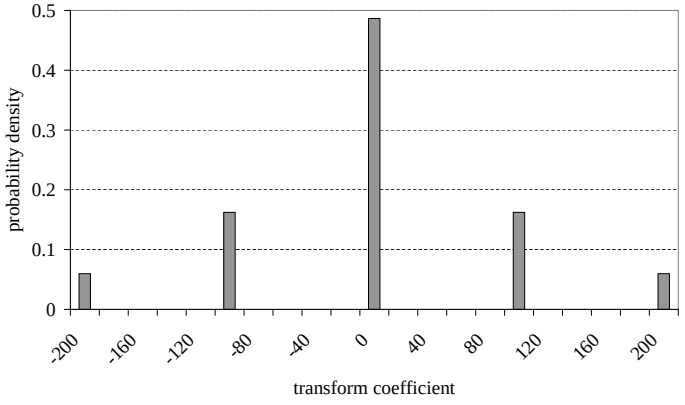
(f) $\lambda = 0.01$, $Q_1 = 500$, $\epsilon_1 = 1/3$

Figure 3.16: Verification of theoretical results with Matlab.

quantizer offset $\epsilon_2 = 0$.



(a) $Q_1 = 20$, $\epsilon_1 = 1/3$



(b) $Q_1 = 100$, $\epsilon_1 = 1/3$

Figure 3.17: Probability mass function of quantized Laplace source ($\lambda = 0.01$).

Transrating heuristic Based on the rate-distortion analysis we propose the following heuristic for transrating. For coarse quantization in the encoder, a good solution consists of first decreasing the quantizer offset ϵ_2 with fixed quantizer step size Q_2. When we reach $\epsilon_2 = 0$ and higher bit rate reductions are required, the quantizer step size Q_2 can be increased with a fixed quantizer offset $\epsilon_2 = 0$. The classical approach with variable quantizer step size Q_2 and fixed quantizer offset $\epsilon_2 = 1/3$ is used as reference.

3.5 H.264/AVC requantization transrating

In order to verify the heuristic derived from the theoretical rate-distortion model, the heuristic is implemented in an H.264/AVC transrater. The remainder of this section describes the experimental setup and the rate-distortion results for H.264/AVC requantization.

3.5.1 Experimental setup

In order to evaluate the performance of requantization, we selected sequences with varying characteristics: *Foreman*, *Paris*, and *Stefan* sequences (CIF resolution, 30 fps). The video sequences are coded using the H.264/AVC reference software (Joint Model version 15.1). The default coding tools are used: five reference pictures, CABAC entropy coding [39], and rate-distortion optimization (RDO) [46].

The encoder was configured to use Main Profile, with IBBP coding structure with GOP length of 15 pictures. Each picture corresponds to a single slice which is coded with a fixed QP_1 and ϵ_1 value. The sequences are coded with QP_I values (for I slices) 22, 27, and 32, $QP_P = QP_I + 1$ values (for P slices), and $QP_B = QP_I + 2$ values (for B slices). These correspond to the values used in the VCEG common test conditions [47]. In the remainder of this section, we only mention the QP value for the I slices.

The encoder enables explicit quantizer offset support [71–73]. Both the quantizer step size and the quantizer offset can vary. The quantizer offsets change based on the prediction mechanism of the macroblock and the position of the transform coefficient in the residual block. Both 4×4 or 8×8 blocks are supported in the H.264/AVC specification. When no quantizer offsets are given, default values are selected.

Based on our work [40, 74], a mixed architecture was chosen for H.264/AVC requantization transrating. The mixed architecture has good visual results and acceptable complexity. This transrating architecture applies decoding and re-encoding to the I pictures, spatial and temporal compensation to the P pictures, and open-loop transrating to the B pictures. In the architecture, we employ our heuristic in order to improve the transrating of B pictures.

3.5.2 Transform coefficient distribution

In order to quantify typical residual data for B pictures in H.264/AVC, we performed some statistical measurements. In order to measure the statistics of the transform coefficients, we need to correct the outcome of the transform function in the encoder. The transform function only incorporates the kernel

transform while the post-scaling operation is postponed to the quantization. In order to have the transform coefficients at our disposal, a position-dependent multiplication is required with factors $1/4$, $1/10$, and $1/2\sqrt{10}$. These factors are used for the 4×4 transform, the factors for the 8×8 transform are derived the same way. We refer the reader to Appendix B for more information about the multiplication factors and the intertwined transform and quantization.

The Laplace parameter can be computed using different methods. A first method computes the variance σ^2 of the transform coefficients [75]. The Laplace parameter is found using the well-known relation: $\lambda = \sqrt{2}/\sigma$. A second method could be the maximum likelihood estimation for the Laplace parameter [76]. We select the variance-based method in order to find a typical value for the Laplace parameter. We excluded data from *Intra* and *Skip* macroblocks from the results. From these measurements it is clear that a Laplace parameter $\lambda = 0.01$ is an appropriate choice.

3.5.3 Transrating results for variable quantizer offset

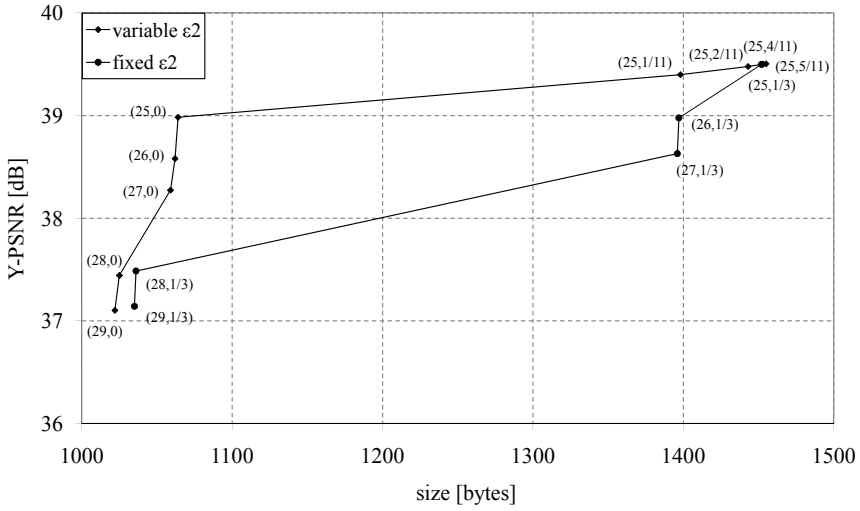
The rate-distortion results for requantization transrating with fixed and variable quantizer offset are shown in Figure 3.18 (*Foreman* sequence), Figure 3.19 (*Paris* sequence), and Figure 3.20 (*Stefan* sequence). These results present the achievable rate-distortion points for requantization transrating of the first B picture for different first-step quantizers.

The rate-distortion graphs show two curves: one curve represents requantization with a fixed quantizer offset ($\epsilon_2 = 1/3$) and one curve represents requantization with a variable quantizer offset. The ΔQP value ranges from 1 to 5 and the quantizer offset ϵ_2 is in the range $[0, \epsilon_{max})$. The results for the variable quantization offset are generated using the transrating heuristic described in the previous section.

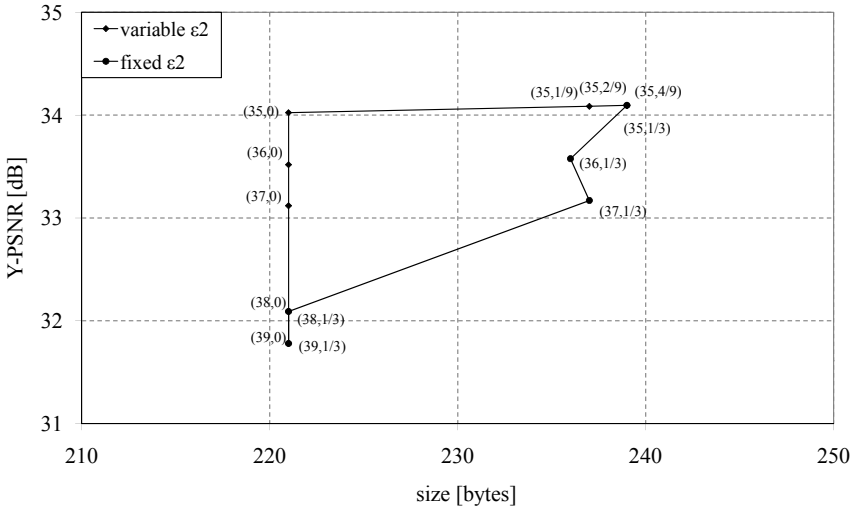
For fixed quantizer offset ϵ_2, a gap in the bit budget is found when ΔQP increases from 3 to 4. For variable quantizer offset ϵ_2, a gap in the bit budget is found when the quantizer offset ϵ_2 is decreased to the value 0. This results from the small transform coefficients in the video bitstream. These small transform coefficients have value -1 or $+1$ and disappear due to requantization.

From the rate-distortion results, it is clear that both approaches can achieve the same bit budget. However, the visual quality is improved when we allow some flexibility regarding the quantizer offset ϵ_2. When the transrating heuristic is used, gains are found for all experiments. The gains are smaller when the quantizer in the encoder becomes finer. This follows from the discussion given in Section 3.4.4.

The maximum gain is found when the following two rate-distortion points

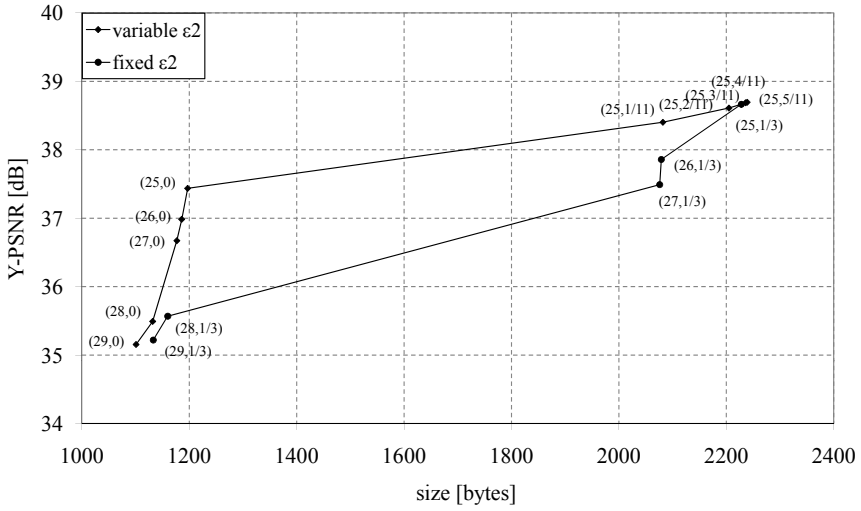


(a) first B picture, $QP_I = 22$

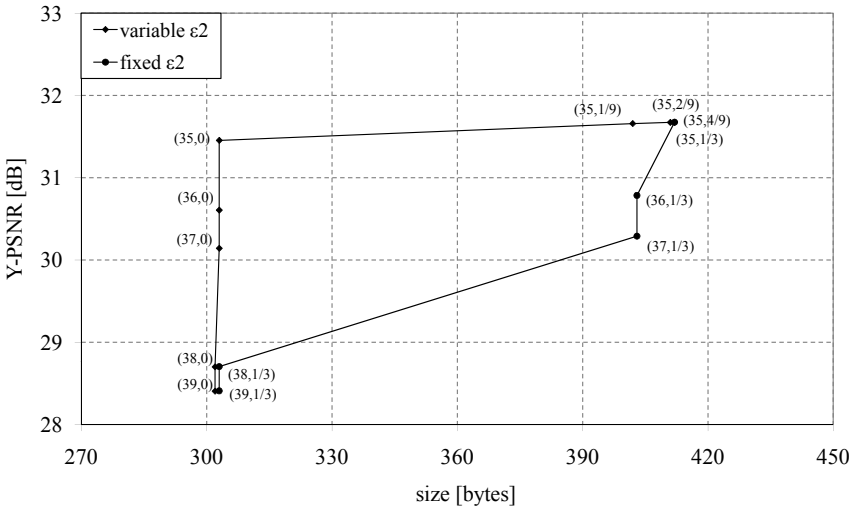


(b) first B picture, $QP_I = 32$

Figure 3.18: Rate-distortion results for open-loop requantization of B pictures for the *Foreman* sequence. The notation for the labels in the figures is (Q_2, ϵ_2).

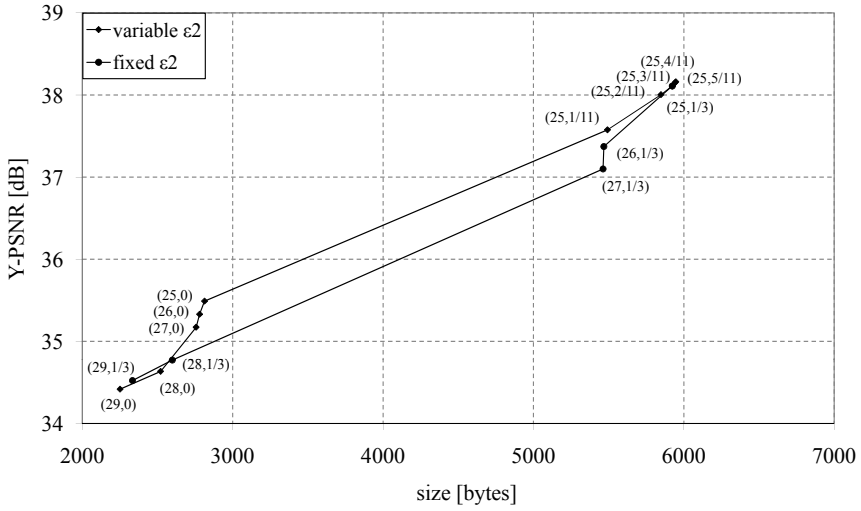


(a) first B picture, $QP_I = 22$

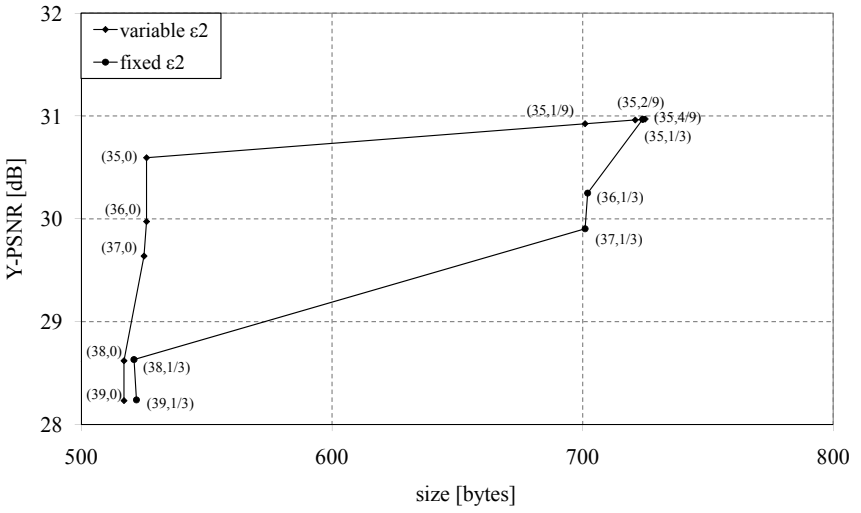


(b) first B picture, $QP_I = 32$

Figure 3.19: Rate-distortion results for open-loop requantization of B pictures for the *Paris* sequence. The notation for the labels in the figures is (Q_2, ϵ_2).



(a) first B picture, $QP_I = 22$



(b) first B picture, $QP_I = 32$

Figure 3.20: Rate-distortion results for open-loop requantization of B pictures for the *Stefan* sequence. The notation for the labels in the figures is (Q_2, ϵ_2).

are compared: 1) fixed quantization offset with $\Delta QP = 4$ and $\epsilon_2 = 1/3$ and 2) variable quantization offset with $\Delta QP = 1$ and $\epsilon_2 = 0$. The maximum gain is about 1 to 2 dB for all experiments. The maximum gain increases for low-quality video coding (coarser quantization, higher QP values) and decreases for high-quality video coding (finer quantization, lower QP values). This also corresponds with the findings from Section 3.4.4.

3.6 Conclusions and original contributions

In this chapter, we derive a transrating heuristic based on theoretical rate-distortion results of the Laplace source which is quantized twice. In order to simplify the calculations for entropy and distortion, we provide a solution based on the characteristics of the Laplace source and the effective quantizer. The effective quantizer is derived as the superposition of the quantizers in encoder and transrater and generates the same output as successively applying these quantizers. The memoryless property from the Laplace source is combined with the periodic property of the effective quantizer in order to derive expressions for entropy and distortion. The transrating heuristic is derived from the theoretical results for different quantization and requantization. We found that increasing the quantizer step size with fixed quantizer offset is a good solution for fine quantization in the encoder. When the quantization in the encoder is coarse, a better solution consists of decreasing the quantizer offset with fixed quantizer step size. The transrating heuristic was implemented in transrating software for H.264/AVC and is applied to open-loop transrating of B pictures. We found gains in visual quality (more than 1 dB for certain configurations) compared to transrating with fixed quantizer step size.

The work that was presented in this chapter can also be found in the following publications:

- Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Quantizer offset selection for improved requantization transcoding. *Submitted to Elsevier, Signal Processing: Image Communication*.

- Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Leveraging the quantization offset for improved requantization transcoding of H.264/AVC video. In *Proceedings of the Picture Coding Symposium (PCS)*, Chicago, IL, USA, May 2009.

Chapter 4

Mixed architectures for H.264/AVC requantization

4.1 Rationale and related work

Requantization transrating is often used in order to meet the bit rate constraints imposed by network or client. The requantization transrater applies a coarser quantizer in order to reduce the amount of residual data. One of the main issues for requantization transrating is what transrating architecture should be used. The transrating architecture has a major impact on the complexity of the transrating solution and the quality of the transrated video bitstreams. The selection of a transrating architecture should be done consciously. In this chapter, different transrating architectures are compared and evaluated in the context of H.264/AVC.

In the past, various transrating architectures have been presented which apply one transrating technique to all pictures in the video bitstream. Two transrating techniques were considered: the open-loop transrater and the cascaded decoder-encoder. Sun *et al.* evaluated transrating architectures that reduce the bit rate of MPEG video bitstreams [25]. They investigated two open-loop architectures and two cascaded decoder-encoder architectures. The authors presented two open-loop transrating architectures, one architecture discards high-frequency transform coefficients, the other architecture requantizes the residual data with a coarser quantizer. These low-complexity architectures do not have a decoding and encoding loop nor a reference picture buffer. The visual quality tends to degrade from picture to picture until an I picture is reached. The degradation results from temporal drift which is introduced by the motion-compensated prediction. The authors also investigated two cascaded decoder-encoder architectures, one architecture keeps the coding de-

cisions of the incoming video bitstream, the other architecture evaluates all coding modes again. These architectures maintain two prediction loops (i.e., a decoding loop and an encoding loop) and require more computational and memory resources; however, the transcoded video bitstreams have good visual quality.

The first approach towards a low-complexity technique that avoids the temporal drift was proposed by Morisson *et al.* [77]. The solution is developed for video bitstreams that are generated with a predictive algorithm, such as motion-compensated prediction. The distortion introduced by the requantization is calculated and a correction signal is included in the video bitstream. The correction signal restrains the temporal drift which results from the reference mismatch due to the predictive coding. However, the calculation of the requantization distortion and the generation of the correction signal requires more processing.

Assunção *et al.* provided an in-depth description of the transrating technique that restrain the temporal drift [78]. The authors focused on transrating of MPEG-2 Video bitstreams and started from the cascade of decoder and encoder and merged the prediction loops. They provided a description of the transrating architecture which shows that the low-complexity technique adds the correction signal required in order to restrain the temporal drift. The visual quality and the bit rate are similar to single coding at the reduced bit rate, the maximum overhead in bit rate is about 5% and the difference in PSNR is less than 1 dB.

Keesman *et al.* analyzed two important problems involved in the use of a transrater in the transmission chain [11]: the transrater complexity and the transrater performance. When the incoming and outgoing picture types are equal, the prediction information can be copied from incoming to outgoing video bitstream which results in a reduction of the computational complexity. Further reducing the complexity of the transrater can be achieved by considering motion compensation as a linear operator. As a result, the decoding and encoding loop in the transrater can be replaced by only one prediction loop which operates on requantization differences instead of reconstructed pixel values.

A drift-free transrater for reducing the bit rate of MPEG-2 Video bitstreams was proposed by Assunção *et al.* [12]. The transrating architecture fully operates in the transform domain. As a result, motion-compensated prediction needs to be implemented in the transform domain. The computational complexity of the approach can be reduced by 81% when approximations are used. In addition, Lagrange optimization is used for the bit allocation in the transrater. This results in better visual quality compared to direct encoding at the target bit rate with non-optimized MPEG-2 Video software.

Since the market makes the shift from MPEG-2 Video to H.264/AVC as coding format for various multimedia applications, the producers of equipment for generation, transmission, and consumption have to focus more on H.264/AVC technology. One of the main reasons of the switch is the improved coding efficiency attainable by the H.264/AVC video coding standard [5,6,46]. As a consequence, intelligent solutions are required in order to efficiently process H.264/AVC bitstreams, in particular transrating.

Lefol *et al.* presented an approach for transrating of H.264/AVC video bitstreams [29]. The authors combine different techniques for requantization in a mixed architecture which selects a requantization technique based on the picture type. The cascade of decoder and encoder is used for the I pictures, while transrating with temporal compensation is used for the P and B pictures. Although the proposed architecture results in acceptable visual quality, a PSNR gap of about 3 to 4 dB was found compared to the cascaded decoder-encoder. They found that the properties of the video bitstream, in particular the number of I macroblocks in P and B pictures, have a major impact on the visual quality, but they do not propose any solution for this problem.

In this chapter, the transrating problem of H.264/AVC video bitstreams is studied. Three basic architectures with their strengths and weaknesses are presented in the context of H.264/AVC. These basic transrating architectures retain the coding decisions from the incoming video bitstream in order to restrict the computational complexity of the transrating solution. These architectures are the open-loop transrater, the transrater with spatial/temporal compensation, and the cascaded decoder-encoder.

Applications require both fast transrating and good visual quality. We found that none of the basic architectures can meet these requirements. The open-loop transrater has low computational complexity and memory requirements. However, severe drift is introduced due to the spatial and temporal dependencies in the video bitstream. The transrater with compensation has one prediction loop. This transrater improves the visual quality of the transrated video but requires more memory capacity and processing power. The cascaded decoder-encoder is the only drift-free architecture in this comparison. This transrating architecture, however, requires significantly more resources (both processing power and memory capacity) due to two prediction loops. Since all three basic architectures lack either fast transrating and good visual quality, a mixture of basic architectures is more appropriate.

Mixed transrating architectures are presented in this chapter which intelligently combine different transrating techniques based on the picture/macroblock type. The mixed transrating architectures decode and encode the I pictures, while the P and B pictures are transrated using open-loop requanti-

zation or compensation techniques. Applying spatial compensation in motion-compensated pictures highly reduces the visual artifacts and therefore results in acceptable video bitstreams with reduced bit rate. Adding temporal compensation in motion-compensated pictures further improves the visual quality, albeit to a smaller extent, but also increases the computational complexity and memory requirements of the transrating architecture.

The main focus of this chapter is on mixed architectures for transrating. The mixed architectures were developed in the context of a project with the Service Provider Video Technology group of Cisco (formerly Scientific Atlanta). An extensive discussion of compensation techniques is found in the dissertation of Jan De Cock [79].

The chapter is organized as follows. The basic architectures are discussed in context of H.264/AVC transrating in Section 4.2. In Section 4.3, we present four mixed architectures for H.264/AVC requantization transrating. We discuss the transrating methods for different picture types, in particular the compensation methods for reducing drift propagation. Finally, Section 4.4 provides a performance analysis and Section 4.5 concludes this chapter.

4.2 Basic transrating architectures

In the literature, different architectures for transrating have been presented which can be classified according to their computational complexity. We discuss three basic transrating architectures which retain the coding decisions of the incoming video bitstream. The classification of the basic transrating architectures is given in Figure 4.1.

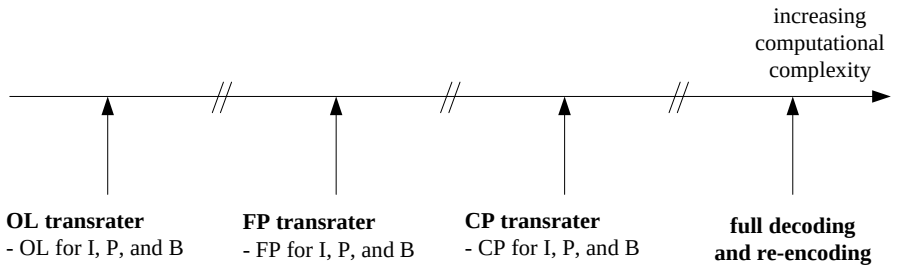


Figure 4.1: Transrating architecture classification: basic transrating architectures.

The open-loop (OL) transrater is the transrating solution with the least computational and memory requirements. This type of transrater parses the original video bitstream, changes the characteristics without taking into ac-

count potential dependencies, and finally writes out the modified video bitstream.

In order to improve the visual quality of the transrated video bitstreams, spatial/temporal compensation techniques can be applied. These techniques compensate for the requantization error made in the reference blocks. This type of transrater is referred to as the fast pixel-domain (FP) transrater.

The cascaded pixel-domain (CP) transrater consists of the cascade of decoder and encoder. The coding decisions of the incoming video bitstream can be copied to the outgoing video bitstream. This results in a significant reduction of the computational complexity. The transrating process is drift-free due to the double-loop character of the transrating architecture.

In this section, the three basic architectures for transrating are discussed in the context of H.264/AVC.

4.2.1 Open-loop transrater (OL transrater)

The OL transrating architecture is the requantization transrating architecture with the least computational complexity and memory requirements. The coding decisions remain unchanged for both intra-predicted and motion-compensated macroblocks, since coding parameters from the incoming video bitstream are passed on to the outgoing video bitstream. The different steps in the OL transrating architecture are as follows. First, CAVLC or CABAC entropy decoding is applied to the residual data. Using the original quantization parameter QP_1 the residual data is inverse quantized and inverse transformed. Next, the difference values are transformed and requantized using the new quantization parameter QP_2. Finally, the residual coefficients are CAVLC or CABAC entropy coded in order to obtain the new video bitstream at the target bit rate. The pixel-domain OL transrater is shown in Figure 4.2.

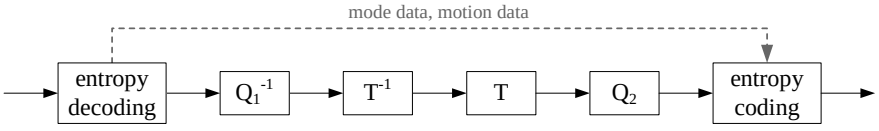


Figure 4.2: Pixel-domain open-loop transrater.

One can see that the inverse and forward transforms can be combined as a result of orthogonality. This way, the pixel-domain OL transrater is converted to a transform-domain OL transrater. This transrating solution further reduces the required amount of processing power. The transform used in the H.264/AVC specification is non-unitary and the normalization and the quan-

tization are intertwined. As a result, special care has to be taken regarding the multiplication factors in the requantization. This results in adjusted multiplication factors in the requantization [29], which is indicated by an accent in Figure 4.3. More information about the transform and the quantization in H.264/AVC is given in Appendix B.

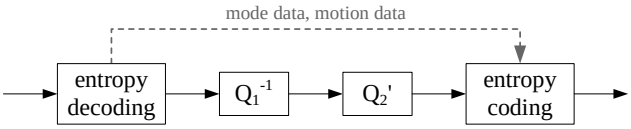


Figure 4.3: Transform-domain open-loop transrater.

The H.264/AVC specification provides two entropy coding schemes: CAVLC and CABAC. CAVLC entropy coding has low complexity and is supported by all profiles of the specification. CABAC entropy coding results in additional coding gain and is only supported in some profiles of the specification. According to Marpe *et al.* [39], average bit rate savings of 9% to 14% can be achieved for a set of video sequences representing typical material in broadcast applications and for a range of acceptable video quality of about 30 to 38 dB. The entropy coding can be different for the incoming and the outgoing video bitstreams. The selection of another entropy coding scheme has only impact on the coding efficiency and the entropy coding complexity. Other components of the transrating solution will not be affected by the entropy coding process.

The requantization process will reduce the amount of residual data in order to decrease the number of bits needed for entropy coding. The requantization process may convert blocks with less residual data to blocks with only zero coefficients. When all 4×4 blocks of a macroblock contain zeros and the motion vector difference is zero, the macroblock will be efficiently signaled as *P_Skip* or *B_Skip* macroblock [80]. In this case, the transrating process converts the macroblock type.

In the video bitstream, besides the requantized residual data, two syntax elements should be updated before entropy coding. The change in QP needs to be signaled at the slice and/or macroblock level by changing the syntax elements *slice_qp_delta* and *mb_qp_delta*. All other syntax elements remain unchanged. When the $\Delta QP = QP_2 - QP_1$ is kept fixed throughout the slice, changing *mb_qp_delta* is only necessary for avoiding overflow and underflow of the QP value. The QP values must be in the range $[0, 51]$ according to the H.264/AVC specification [5].

An important disadvantage of the OL transrater is encoder-decoder mis-

match, resulting in degraded visual quality of the transrated video bitstreams. The reference pixel values are changed due to the requantization process; however, no action is taken in order to compensate for the changes in the reference. The requantization process introduces errors in the pictures, denoted as *requantization drift*, which propagate both spatially and temporally according to the dependencies in the video bitstream.

The OL transrater severely reduces the visual quality, primarily as a consequence of spatial error propagation, caused by intra prediction. This can be seen from the transrated I picture shown in Figure 4.4. In most cases, the visual quality of transrated I pictures is unacceptable due to spatial drift propagation and accumulation. Since these pictures are used as reference for subsequent P and B pictures, the visual quality of I pictures should be as good as possible. From this observation, we can conclude that OL transrating for I pictures should be avoided.

The OL transrating solution is the *de facto* approach for MPEG-2 Video transrating. This transrating approach has low complexity and results in good visual quality for typical coding settings. Spatial drift is not found since the intra-coded blocks does not apply spatial prediction. Temporal drift is found due to motion-compensated prediction; however, the temporal drift is canceled out when the next GOP starts.

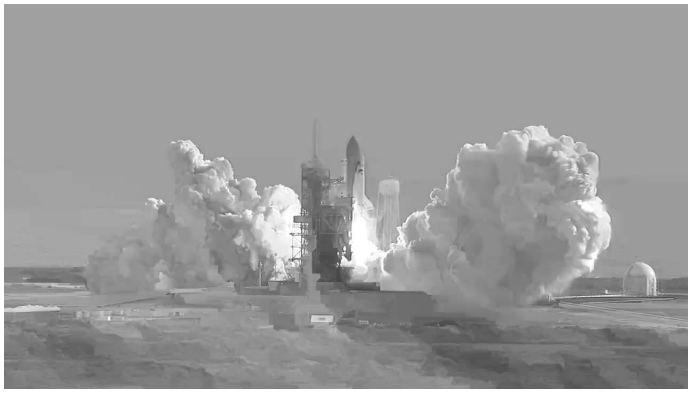
4.2.2 Fast pixel-domain transrater (FP transrater)

The OL transrater has low complexity; however, spatial and temporal drift is found. In order to restrain the drift, we calculate the requantization error and use this error for compensation. This, however, introduces one prediction loop which increases the complexity. The same technique was presented for rate shaping in Chapter 2.

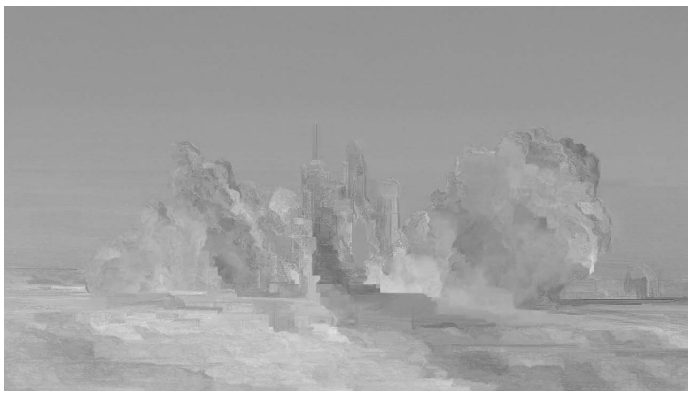
We start from the cascade of decoder and encoder. This architecture is drift-free due to the double-loop character. The mode and motion data are copied from the incoming to the outgoing bitstream. This leads to a significant complexity reduction. Firstly, the memory requirements are reduced since the picture reordering is canceled out. Secondly, reuse of mode and motion data leads to a significant reduction of the complexity since no resources are spent for calculating the new mode and motion data. We collapse the prediction loops together. This way, we calculate and store the requantization error. Afterwards, the requantization error is used to compensate for the requantization error in depending blocks. This leads to the FP transrater with pixel-domain compensation shown in Figure 4.5.

A first simplification can be realized by moving the forward and inverse

(a) Before OL transrating.



(b) After OL transrating ($\Delta QP = 4$, Y-PSNR = 24.09 dB).



(c) Difference values (mid-level gray = no difference).

Figure 4.4: Visual results for the 197^{th} picture of the *Shuttlestart* sequence (HD 720p, 60 fps, $QP_I = 27$, I picture).

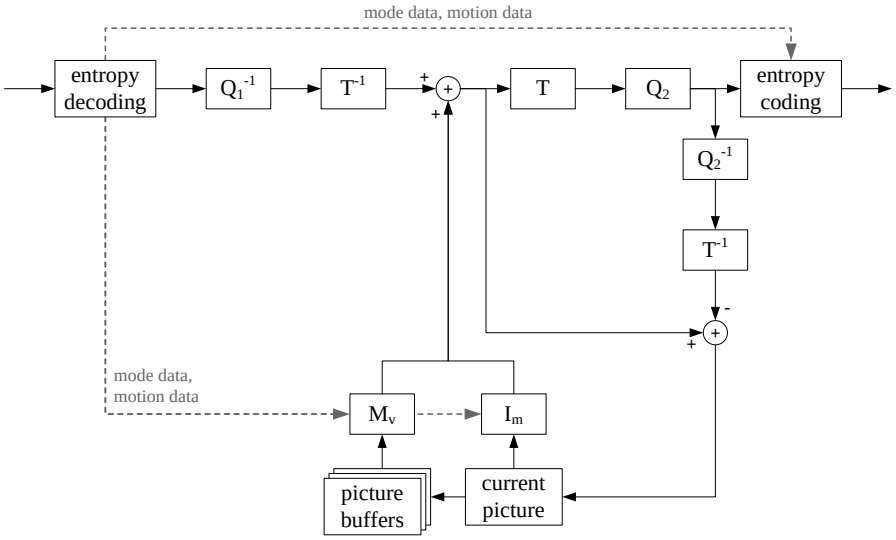


Figure 4.5: Fast pixel-domain transrater (calculation of the requantization error and compensation in the pixel-domain).

transform operations [11]. This way, the compensation is added in the transform domain. A disadvantage of this architecture is that an error is introduced due to less accuracy in the difference value. This yields the FP transrater with transform-domain compensation shown in Figure 4.6.

A second simplification can be realized by removing the forward and inverse transform operations [12]. This way, the compensation is generated and added in the transform domain. This requires a transform-domain equivalent of intra prediction and motion-compensated prediction. We already mentioned that adding the compensation in the transform domain results in small rounding errors since the difference values are calculated in the transform domain. Additionally, there will be an error due to intra prediction and motion-compensated prediction in the transform domain. This results from the fact that an equivalent operation in the transform domain will results in small rounding errors. This yields the FP transrater with transform-domain compensation shown in Figure 4.7.

There are still a number of complex operations in this transrating architecture which have an important impact on the transrating speed of the architecture:

- **Intra prediction:** The intra prediction in the H.264/AVC video coding standard provides directional prediction (nine prediction modes for *In-*

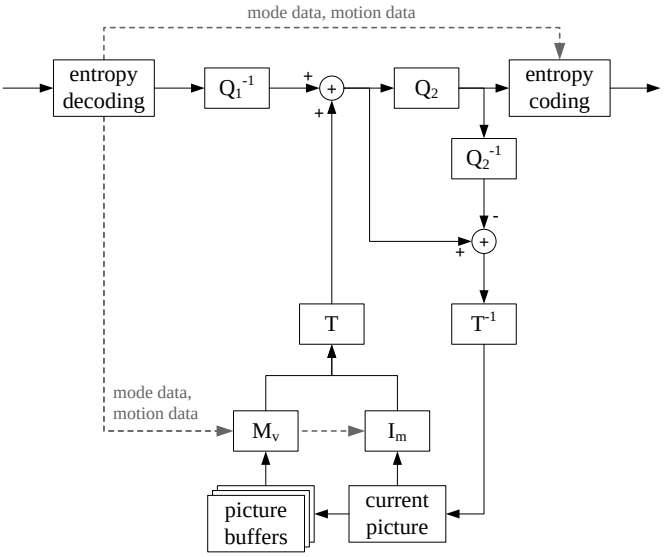


Figure 4.6: Fast pixel-domain transrater (calculation of the requantization error in the pixel domain and compensation in the transform domain).

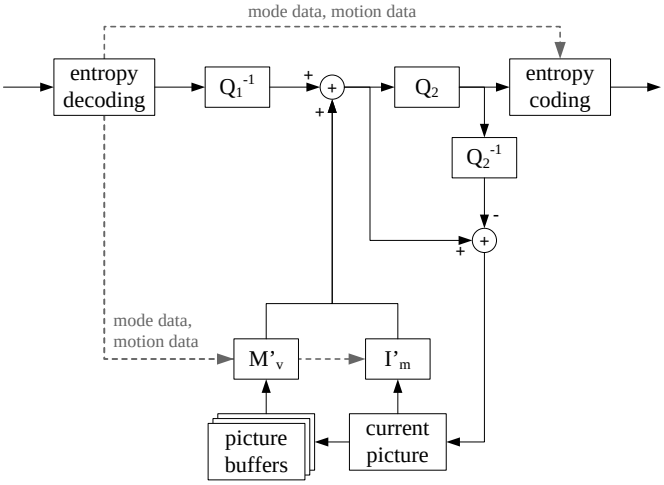
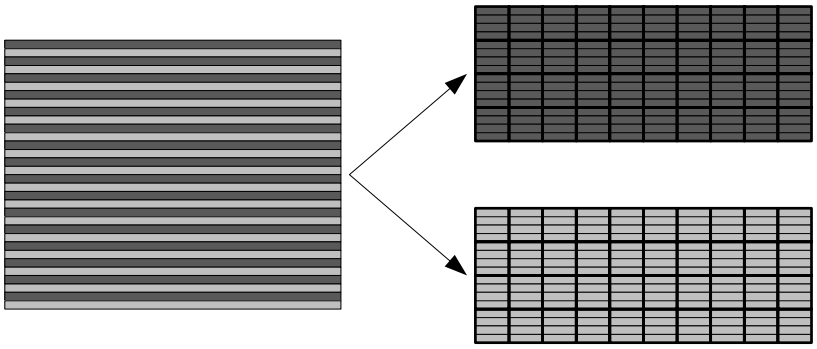
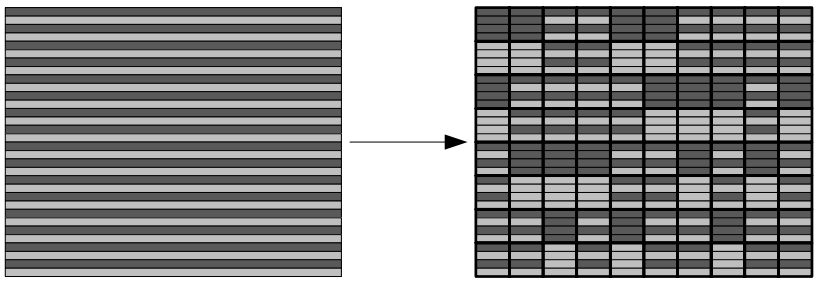


Figure 4.7: Fast pixel-domain transrater (calculation of the requantization error and compensation in the transform domain).

tra_4x4, nine prediction modes for *Intra_8x8*, and four prediction modes for *Intra_16x16*). The intra prediction is performed using the intra prediction mode of the incoming video bitstream. Besides the intra pre-



(a) Field mode: an interlaced frame coded as two separate coded fields.



(b) Frame mode: an interlaced frame coded as field and frame macroblock pairs.

Figure 4.8: Support for interlaced coding in the H.264/AVC video coding standard.

diction itself, the selection of the appropriate reconstructed pixel values needs to be done.

- **Motion-compensated prediction:** The motion-compensated prediction in the H.264/AVC video coding standard defines variable block-size motion compensation with multiple reference pictures and quarter-pixel motion vector accuracy [81]. The motion-compensated prediction is performed using the motion vectors and the reference picture indices of the incoming video bitstream. Besides the motion-compensated prediction itself, the reference pixel values should be fetched from memory.
- **Support for interlaced coding**: The H.264/AVC video coding standard allows different coding modes for interlaced content. A frame can be coded either as two separate coded fields (field mode, see Figure 4.8(a)) belonging to two different NAL units or the two fields may be coded together as a frame (frame mode, see Figure 4.8(b)) inside a single network abstraction layer (NAL) unit. In addition, in frame mode, pairs

of vertical macroblocks can be either coded as field macroblocks (macroblocks containing samples from a single field) or frame macroblocks (macroblocks containing interleaved samples from the two fields). It is possible to switch between these different modes, at picture level (picture adaptive frame/field - PAFF) and, when frame mode is used, at macroblock level (macroblock adaptive frame/field - MBAFF).

- **Motion vector derivation**: The H.264/AVC video coding standard provides an efficient compression technique for motion vectors. The median of motion vectors of neighboring (sub)macroblock partitions is calculated and the motion vector difference is entropy coded. The skip modes (*P_Skip* and *B_Skip*) and the direct modes (*B_Direct_16x16* and *B_Direct_8x8*) further increase the computational complexity. In order to find the neighboring (sub)macroblock partitions, some additional processes are necessary. These processes are even more complex in the context of interlaced coding.

- **Sub-pixel interpolation:** The interpolation process may be required to perform half-pixel or quarter-pixel interpolation. The interpolation process, defined in the H.264/AVC video coding standard [82], uses a 6-tap filter for half-pixel interpolation and a 2-tap filter for quarter-pixel interpolation. This process is required in both the decoder and encoder loop.

- **Reference picture management**: The reference picture management comprises the reference picture list construction and the reference picture marking process. The reference picture marking process can be sliding window or adaptive reference picture marking. In the former case, a first-in first-out (FIFO) mechanism is used for the short-term reference pictures. In the latter case, memory management control operations (MMCO) are provided in the video bitstream for marking pictures as long-term reference, as unused for reference etc. The required memory buffers depend on the number of prediction loops. An architecture with two prediction loops requires reconstruction at decoder and encoder side, and therefore this architecture demands the memory capacity of both a decoder and an encoder.

Advantages of the single-loop nature of the FP transrating architecture are the limitation of the required buffer sizes and the need for only one intra prediction or motion compensation step, hereby also avoiding the complex deblocking process. However, motion vector derivation and quarter-pixel motion

compensation are still necessary. An important disadvantage is the introduction of small errors which will result in *compensation drift*. The compensation drift is typically smaller compared to the requantization drift. This also results in small visual artifacts as can be seen from the visual results. The errors are introduced due to non-linear operations in intra prediction and motion compensation, and due to the absence of clipping operations (elaborated on in Section 4.3.8).

This architecture allows spatial and temporal compensation to be applied independently (i.e., spatial compensation is not required in order to apply temporal compensation and vice versa). The only requirement is that the requantization errors should be stored in memory buffers. For spatial compensation only the requantization errors of the current picture are used, while for temporal compensation the requantization errors of all reference pictures should be available. These requantization errors are generally small and tend to increase when higher ΔQP values are used. This results from the increased distortion when the QP value grows.

The spatial compensation compensates for the requantization errors in neighboring blocks since these blocks are used as reference for the intra prediction. The requantization error, which will propagate spatially according to the intra prediction direction, is rectified by the spatial compensation. Since intra prediction is a low-complexity method, this compensation technique will hardly have impact on the total processing time.

The temporal compensation compensates for the requantization errors which result from requantization of reference pictures. This requires motion vector derivation (at most two motion vectors for every (sub)macroblock partition) and half-pixel or quarter-pixel interpolation. Therefore temporal compensation will have more impact on the processing time and the memory buffer requirements.

The FP transrating architecture compensates for requantization errors in the reference pixel values. No requantization drift is found, however, compensation drift exists due to a number of inaccuracies. This results in small errors which can propagate according to the dependencies in the video bitstream. This can be seen from the transrated I picture shown in Figure 4.9.

4.2.3 Cascaded pixel-domain transrater (CP transrater)

The CP transrating architecture consists of the cascade of decoder and encoder. The transrating process is drift-free as a result of two prediction loops. The coding decisions of the incoming video bitstream are copied to the outgoing video bitstream. This results in a significant reduction of the computational

(a) Before FP transrating.

(b) After FP transrating ($\Delta QP = 4$, Y-PSNR = 34.65 dB).

(c) Difference values (mid-level gray = no difference).

Figure 4.9: Visual results for the 197^{th} picture of the *Shuttlestart* sequence (HD 720p, 60 fps, $QP_I = 27$, I picture).

complexity.

4.2.3.1 Rate-distortion optimal coding

The coder control decides what part of the picture should be coded using what method. This is an encoder issue which is typically a complex operation.
The main goal of video coding is to minimize the coding distortion D subject to a rate constraint R_C: $\min\{D\}$ subject to $R \leq R_C$. In order to simplify the problem, Lagrange optimization is often used. This way, the constrained problem is converted to an unconstrained problem: $\min\{D + \lambda R\}$.
Coder control consists of two operations [46, 83, 84]: *rate-constrained motion estimation* and *rate-constrained mode decision*. The coder control is discussed in the context of H.264/AVC.

Rate-constrained motion estimation In the first step, motion estimation is performed by minimizing the following Lagrange functional:

$$J_{ME}(\mathbf{m}, \lambda_{ME}) = D_{DFD}(s, c, \mathbf{m}) + \lambda_{ME} \cdot R_{MOT}(\mathbf{m} - \mathbf{p}), \tag{4.1}$$

with $\mathbf{m} = (m_x, m_y)$ being the motion vector and $\mathbf{p} = (p_x, p_y)$ being the motion vector prediction, s being the original block and c being the motion-compensated block, λ_{ME} being the Lagrange multiplier for the motion estimation, D_{DFD} being the distortion of the *displaced frame difference* (DFD) and R_{MOT} being the number of bits for coding the motion data.
The distortion cost is defined as follows, with $p = 1$ for the *sum of absolute differences* (SAD) and $p = 2$ for the *sum of squared differences* (SSD):

$$D_{DFD}(s, c, \mathbf{m}) = \sum_{y=1}^{B_y} \sum_{x=1}^{B_x} |s[x, y] - c[x - m_x, y - m_y]|^p, \tag{4.2}$$

with values B_x and B_y being the vertical and horizontal dimensions of the partition.
The rate cost only represents the motion data and can easily be computed using a table lookup.

Rate-constrained mode decision In the second step, the mode decision is performed by minimizing the following Lagrange functional:

$$J_{MD}(s, r, v|QP, \lambda_{MD}) = D_{REC}(s, r, v|QP) + \lambda_{MD} \cdot R_{MD}(s, r, v|QP), \tag{4.3}$$

with r being the reconstructed macroblock, v being the macroblock mode, QP being the quantizer parameter, and λ_{MD} being the Lagrange multiplier for the mode decision. The macroblock mode is chosen from a defined set which depends on the slice type.
The distortion cost is defined as the SSD calculation between the original block s and the reconstructed block r:

$$D_{REC}(s,r,v|QP) = \sum_{y=1}^{16}\sum_{x=1}^{16} |s_Y[x,y] - r_Y[x,y,v|QP]|^2 + \sum_{C\in\{U,V\}} \left(\sum_{y=1}^{8}\sum_{x=1}^{8} |s_C[x,y] - r_C[x,y,v|QP]|^2 \right), \tag{4.4}$$

with s_Y and r_Y being the original and reconstructed luminance values, while s_U, s_V and r_U, r_V being the original and reconstructed chrominance values.
The rate cost is the number of bits associated with choosing macroblock mode v and quantization parameter QP. This also includes the bits for the macroblock header, the motion data, and the residual data.

Lagrange parameters Some experiments have been conducted in order to determine the optimal values for the λ parameters [46]. From these experiments, the following parameter follows for mode decision:

$$\lambda_{MD} = 0.85 \cdot 2^{(QP-12)/3}. \tag{4.5}$$

Additionally, λ parameters for motion estimation are determined as follows:

$$\lambda_{ME} = \sqrt{\lambda_{MD}} \text{ or } \lambda_{ME} = \lambda_{MD}, \tag{4.6}$$

when using SAD or SSD distortion, respectively.

The rate-constrained motion estimation and the rate-constrained mode decision demonstrate that rate-distortion optimal coding is a complex operation. As a consequence, this operation should be avoided during transrating. In this case, the mode and motion data of the incoming video bitstream is used which results in a significant reduction of the complexity. However, this may result in decreased coding efficiency.
For small bit rate reductions, the mode and motion data of the incoming video bitstream are still representative. Again calculating the mode and motion data in the transrater will slightly improve the transrating in rate-distortion sense. For higher bit rate reductions, new mode and motion data is necessary in order

to preserve coding efficiency. Many applications only require small reductions of the bit rate, the mode and motion data can be used for the outgoing video bitstream without severely affecting the coding efficiency.

4.2.3.2 Deblocking filter

The deblocking filter has to be performed in both decoder and encoder loop [85]. Since the deblocking filter itself requires much processing power, this operation will have a noticeable impact on the transrating speed of the architecture. When it comes to visual quality, the position of the deblocking filter has a significant impact. Deblocking has to be applied in two places in the architecture, both in the decoder as well as in the encoder loop. The position of the deblocking filter at the decoder side can be an issue for the CP transrating architecture. If we want to simulate the behavior of the cascade of decoder and re-encoder, the position of the deblocking filter should be as indicated in Figure 4.10. In the cascade, the pictures are deblocked before being output. Then, these pixels are used as a starting point for the encoder. In this way, however, differences are introduced in the images, even when requantizing with $\Delta QP = 0$. Another strategy is to position the deblocking filter as in Figure 4.11, where no changes are introduced when $\Delta QP = 0$ is applied.

For OL and FP transrating, a macroblock-based data flow model can be used. Due to the introduction of the deblocking filter, however, this is no longer possible for the CP transrating architecture in Figure 4.10. Deblocking of a macroblock affects all the pixels in one macroblock, as well as the right two columns of pixels of the macroblock to the left (when available), and the bottom two rows of pixels of the top macroblock (when available). A dataflow model could be used that transrates macroblocks with a delay of one macroblock row. For the sake of elegance of implementation and symmetry, however, a picture-based dataflow model was used. When the CP transrating architecture in Figure 4.11 is used, this restriction no longer applies, and a macroblock-based data flow model is again possible. Since non-deblocked pixels are used as a reference for encoding, it is no longer necessary to wait for deblocking of the macroblock to the right or to the bottom to allow encoding.

4.3 Mixed transrating architectures

In practical applications, the OL and FP transrating architectures will result in degraded quality when applied to H.264/AVC video bitstreams, even though for prior video coding standards (such as MPEG-2 Video) the output quality remained close to that of drift-free transrating with the CP transrating architec-

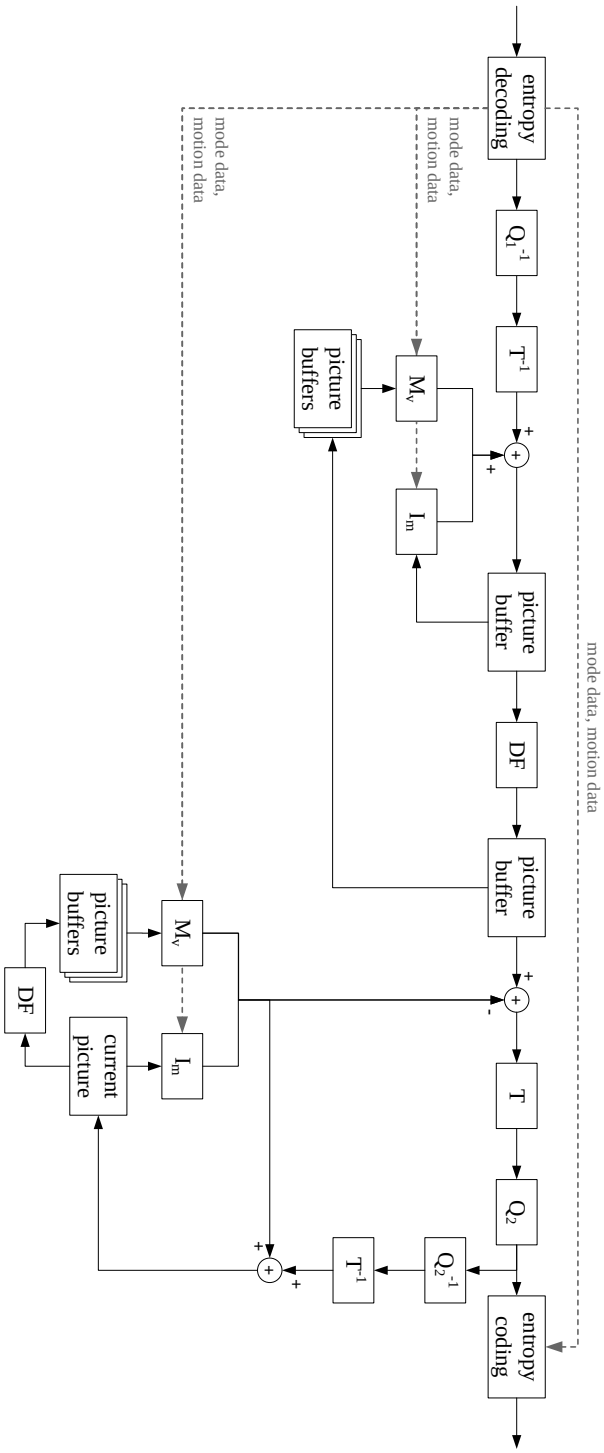


Figure 4.10: Cascaded pixel-domain transrater (DF inline).

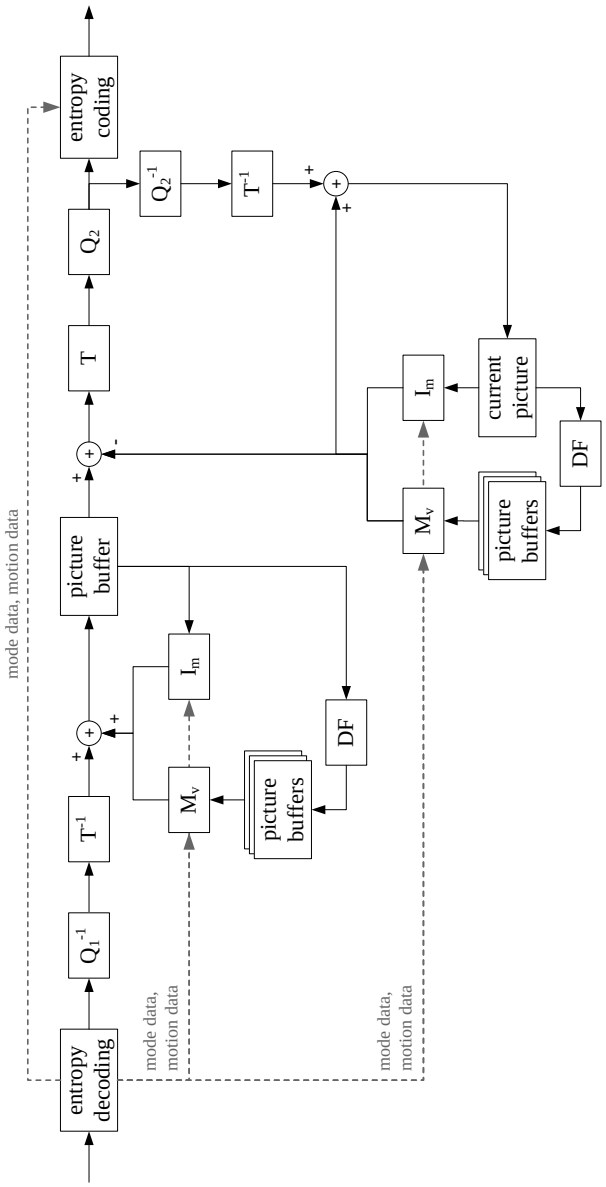


Figure 4.11: Cascaded pixel-domain transrater (DF offline).

ture [11]. Still, this finding does not eliminate the applicability of the underlying techniques used in these architectures. While for intra prediction, both OL and FP transrating result in spatial drift propagation, requantization and compensation errors will only slowly propagate throughout the video sequence for motion-compensated prediction.

For this reason, it becomes clear that mixed architectures, which apply different transrating techniques depending on the picture/macroblock type, can pose a low-complexity alternative to the CP transrating architecture, while keeping quality loss limited. In this way, the benefits of the different transrater techniques can be exploited to the fullest.

The technique of combining different methods is often used in heterogeneous transcoding in order to enable a compensation technique. This results in a transcoding solution which is complexity scalable and has an adaptive method for drift error control. Shen *et al.* proposed an architecture for transcoding from MPEG-2 Video to Windows Media Video (WMV) [86]. On top of the format conversion, a reduction of the bit rate or a reduction of the spatial resolution is performed. Both Qian *et al.* [19] and Tang *et al.* [20] proposed an architecture for transcoding from MPEG-2 Video to H.264/AVC. They consider the drift error due to requantization and the mismatch in motion compensation and propose a transform-domain solution for transcoding.

4.3.1 Transrating rules

Different architectures can be constructed as a combination of the three basic techniques. A number of rules can be formulated in order to derive mixed architectures that reduce computational complexity and memory requirements but nonetheless result in high-quality transrated video bitstreams:

Rule 1: In order to obtain reliable reference pictures, I pictures should preferably be transrated using the CP transrating architecture. In this way no spatial drift is introduced and all dependent pictures can use the drift-free I picture as reference.

Rule 2: For intra-predicted macroblocks, a choice can be made between the OL or FP transrating architectures. For example, pictures that are used as reference should use spatial compensation, while non-reference pictures can be transrated using the OL architecture.

Rule 3: For motion-compensated macroblocks, a choice can be made between the OL or FP transrating architectures. For example, pictures that are used as reference should use temporal compensation, while non-reference pictures can be transrated using the OL architecture.

We already mentioned that spatial and temporal compensation can be ap-

plied independently. This means that spatial compensation is not required in order to apply temporal compensation and vice versa. The only requirement is that the requantization errors are available for compensation. As a result of that, the number of combinations is also limited since certain combinations cannot be realized. For example, when P pictures are OL transrated and the B pictures are compensated, the compensation values will not be available in memory buffers and thus compensation cannot be applied. This combination is not sensible since these transrating techniques cannot be combined in this way. So, the feasible solutions for H.264/AVC transrating take into account the typical dependencies in video bitstreams.

Decoding and re-encoding the intra-predicted macroblocks in P and B pictures without fully decoding these pictures is not possible. The reference pixel values for intra-predicted macroblocks may belong to motion-compensated macroblocks, so these reference pixel values also need to be decoded. However, when *constrained_intra_pred_flag* is enabled, the intra-predicted macroblocks are not allowed to use pixel values from motion-compensated macroblocks. In this case, the intra-predicted regions in P and B pictures can be decoded and re-encoded without the need for decoding and re-encoding motion-compensated macroblocks.

4.3.2 Transrating architectures

We make a distinction between reference pictures and non-reference pictures. The transrating errors, denoted as requantization drift (OL transrating) or compensation drift (FP transrating), may propagate depending on the *nal_ref_idc* syntax element which indicates if the slice belongs to a reference or non-reference picture. In order to make a clear distinction between the presented transrating architectures, every architecture will be indicated as a quintuplet of transrater techniques (α, β, γ, δ, ϵ). The elements of this quintuplet correspond to the choice of techniques used for transrating each of the five following picture/macroblock pairs:

- α: I pictures (all intra-predicted macroblocks);
- β: intra-predicted macroblocks in P and B reference pictures;
- γ: motion-compensated macroblocks in P and B reference pictures;
- δ: intra-predicted macroblocks in P and B non-reference pictures;
- ϵ: motion-compensated macroblocks in P and B non-reference pictures.

In the remainder of the chapter, the reference pictures are the P pictures while the non-reference pictures correspond to the B pictures. However, the mixed transrating architectures can easily be applied to other video bitstream configurations such as hierarchical coding with B pictures.

The classification of the basic and mixed transrating architectures is given in Figure 4.12. The computational complexity increases from left to right in the figure. The basic architectures are the OL transrater, the FP transrater, and the CP transrater. These transraters apply the same algorithm to all macroblocks with no regard to the picture/macroblock type. Four additional mixed architectures are presented. These architectures apply different techniques depending on the picture/macroblock type.

The global architecture for the mixed transrating solutions is depicted in Figure 4.13. The I pictures are decoded and subsequently re-encoded using the CP transrating architecture. For these pictures, only intra prediction is used, so computational complexity remains fairly low for this part of the architecture. The P and B pictures are transrated using the OL or FP transrating architectures. Three switches (S_{acc}: storage of requantization errors, S_{spat}: controls the spatial compensation, and S_{temp}: controls the temporal compensation) are able to activate the compensation techniques. In this combined architecture, no DF is used since all operations for transrating P and B pictures are performed on difference values instead of pixel values.

Taking into account the rules above, four sensible additional architectures can be derived which are discussed in Section 4.3.3 to Section 4.3.6. This results in a complexity-scalable solution as elaborated on in Section 4.3.7. The discussion is based on regular IBBP coding structures, however, the mixed architectures can also be applied on hierarchical B pictures. The compensation techniques (both spatial and temporal) are not perfect due to a number of inaccuracies which are elaborated on in Section 4.3.8.

4.3.3 Architecture 1: no spatial or temporal compensation in P and B pictures (CP,OL,OL,OL,OL)

This architecture applies OL transrating to all macroblocks in P and B pictures. Accumulation of requantization errors is not required (S_{acc} is open) and both spatial and temporal compensation are disabled (S_{spat} and S_{temp} are open). As a consequence of the absence of compensation, spatial and temporal drift appear in P and B pictures and requantization errors propagate according to the dependencies in the video bitstream. However, the drift problem is less problematic for video bitstreams with small resolution and short IDR period. Besides the resolution and the IDR period, other characteristics are

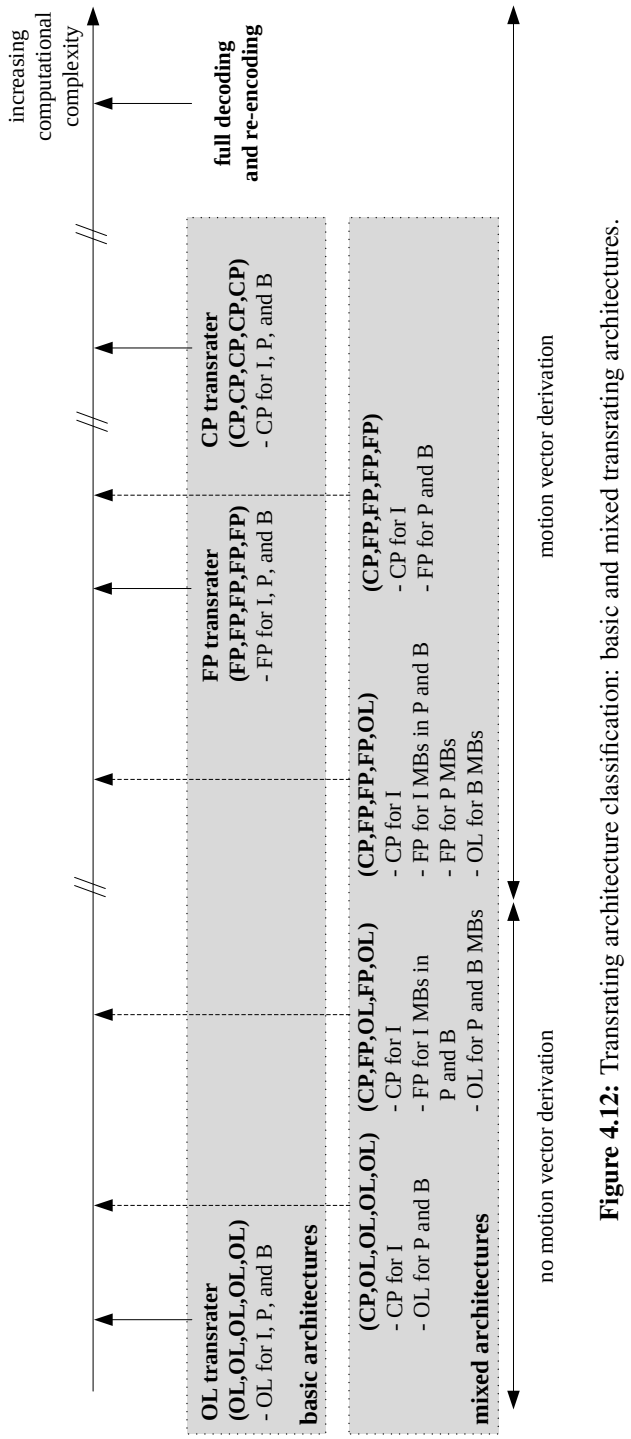


Figure 4.12: Transrating architecture classification: basic and mixed transrating architectures.

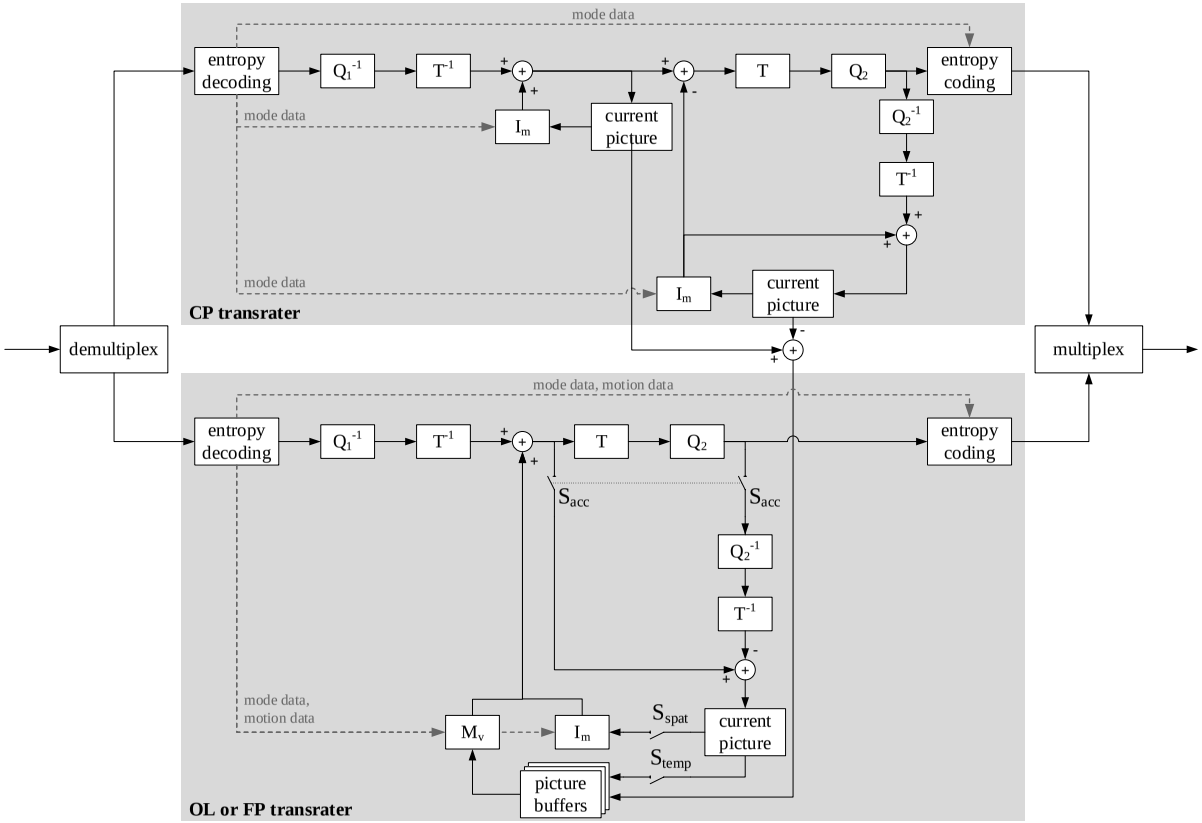


Figure 4.13: Global architecture with switches for the construction of four mixed transrating architectures.

found which have an important impact on the visual quality of the transrated video bitstreams. A significant factor is the number of intra-predicted macroblocks in P and B pictures which determines the amount of spatial drift that is introduced.

Since P and B pictures are transrated using the OL transrating architecture, the computational complexity of the overall transrating architecture is very low. No temporal compensation is applied and therefore motion vector derivation and quarter-pixel motion-compensated prediction are not necessary. This results in a transrating architecture which approximates the transrating speed of the OL transrating architecture. On top of that, no memory buffers for reference pictures need to be provided since the requantization errors of the reference pictures do not have to be stored for future compensation of dependent pictures.

4.3.4 Architecture 2: only spatial compensation in P and B pictures (CP,FP,OL,FP,OL)

This architecture spatially compensates the intra-predicted macroblocks and applies OL transrating to motion-compensated macroblocks. The requantization errors are stored in frame memory (S_{acc} is closed) as they will be used for future compensation. Applying only spatial compensation (S_{spat} is closed) eliminates the spatial drift, however, temporal drift propagation still exists (S_{temp} is open). The spatial drift has a major impact on visual quality; spatial compensation highly improves the visual quality of the transrated video bitstream.

The computational complexity of this architecture is also very low since only spatial compensation is applied. Spatial compensation only requires intra prediction techniques, without the more complex tools from the motion-compensated prediction (such as the motion vector derivation and the quarter-pixel interpolation). The spatial compensation also has a minor impact on the memory requirements. No requantization errors need to be stored for reference pictures, only the requantization error of the current picture is required.

4.3.5 Architecture 3: spatial compensation in P and B pictures and temporal compensation in P pictures (CP,FP,FP,FP,OL)

P pictures are transrated using spatial and temporal compensation (S_{acc}, S_{spat}, and S_{temp} are closed), while for B pictures only spatial compensation is performed and thus the motion-compensated macroblocks are OL transrated (S_{acc} and S_{spat} are closed, S_{temp} is open). In order to obtain a reference for temporal compensation, the difference between the decoded and re-encoded intra

picture is passed on to the decoded picture buffer for temporal compensation of P pictures.

In order to apply the temporal compensation, the motion vectors have to be derived for the P pictures. Together with the quarter-pixel interpolation process, this process is the determining factor in the computational complexity of the overall architecture. This way, temporal compensation has much more impact on processing time compared to spatial compensation. Besides more processing power, more frame memory is required. The frame memory will be used for storing the requantization errors of reference frames in order to apply temporal compensation.

4.3.6 Architecture 4: spatial and temporal compensation in P and B pictures (CP,FP,FP,FP,FP)

This transrating architecture applies both spatial and temporal compensation regardless of the picture type. The requantization errors are stored in the frame memory (S_{acc} is closed). Both spatial and temporal compensation is performed (S_{spat} and S_{temp} are closed). No requantization drift is found, however, the compensation is not perfect due to a number of inaccuracies (elaborated on in Section 4.3.8).

This architecture only requires half the memory requirements of the CP transrating architecture and needs approximately half the processing power of the CP transrating architecture. However, the transrating speed of this architecture is mainly determined by the characteristics of the incoming video bitstream. More intra-predicted macroblocks in P and B pictures will increase the transrating speed of the overall architecture as temporal compensation is more complex compared to spatial compensation.

4.3.7 Dynamic architectures

The global architecture, shown in Figure 4.13, embodies four mixed transrating architectures which are obtained by using the three switches. It is possible to change these switches during operation, resulting in a complexity-scalable solution. This can be advantageous in situations where the load on the transrating system needs to be controlled or the visual quality of the transrated video bitstreams needs to be retained.

Most transrating systems typically process multiple video bitstreams simultaneously. In this case, the transrating system could take advantage of using a complexity-scalable design by applying reconfigurable or reprogrammable hardware. When, for example, the number of video bitstreams in-

creases, temporal compensation could be disabled in order to reduce the load on the transrating system.

Complexity-scalable solutions could also be beneficial in case the visual quality of the transcoded video bitstreams should be retained. The amount of drift depends on the dependencies in the video bitstream, in particular the amount of spatial (smooth or textured) and temporal (slow motion or varying content) information found in the video bitstream.

4.3.8 Inaccuracies of compensation technique

The spatial and temporal compensation techniques are not accurate due to non-linear operations in H.264/AVC coding tools. These non-linear operations introduce rounding errors which are small and therefore do not result in disturbing visual artifacts. Some of these coding tools are:

- **Intra prediction:** Both encoder and decoder use the reconstructed pixel values of neighboring blocks for spatial extrapolation. The spatial extrapolation, more precisely in the directional prediction modes not including horizontal and vertical prediction modes, calculates prediction values using division operations. Single-loop architectures apply these extrapolation formulas to requantization errors instead of pixel values resulting in small errors.

- **Sub-pixel interpolation:** Both the encoder and decoder generate half-pixel and quarter-pixel values using a 6-tap and 2-tap FIR filter, respectively, in order to generate the appropriate reference values. These filters average a number of pixel values in order to generate a sub-pixel value. These filters are normalized using divisions which results in small rounding errors when these filters are applied to requantization errors instead of pixel values.

Other sources of inaccuracies are found which result from the fact that operations cannot be applied to the residual values:

- **Deblocking filter:** The reconstruction process for reference frames in the H.264/AVC specification uses an in-loop deblocking filter for reducing blockiness by changing pixel values at block boundaries. Single-loop architectures apply intra prediction or motion-compensated prediction to requantization errors and therefore the deblocking process is not applicable for removing block boundaries.

- **Clipping operations:** Encoding and decoding processes incorporate saturating operations in order to restrict pixel values to the range of $[0, 255]$ (for 8-bit samples). Single-loop transrating architectures operate on the requantization errors and use these error values in order to compensate. Mismatches can be introduced due to pixel values which are clipped at decoder side where as no action is taken in the single-loop transrating architecture.

4.4 Performance results

4.4.1 Experimental setup

In order to evaluate the performance of the architectures for requantization, we selected a number of sequences with varying characteristics: *Foreman*, *Paris*, and *Stefan* sequences (CIF resolution, 30 fps) and *Crew*, *Sailormen*, and *Shuttlestart* sequences ($720p$ HD resolution, 60 fps). The video sequences are coded using the H.264/AVC reference software (Joint Model version 13.2). The default coding tools are used: five reference pictures, CABAC entropy coding [39], and rate-distortion optimization (RDO) [46].

Main Profile, where only 4×4 integer transform is allowed, is used for coding the CIF sequences while High Profile, where both 4×4 and 8×8 integer transforms are allowed, is used for the $720p$ HD sequences. An IBBP coding structure with a GOP length of 15 pictures is used for the CIF sequences while an IBBBP structure with a GOP length of 28 pictures is used for the $720p$ HD sequences. Alternatively, a hierarchical prediction structure with three temporal layers (periods of seven consecutive B pictures) is used with an intra period of 16 pictures for CIF sequences and an intra period of 32 pictures for $720p$ HD sequences. All coding structures defined for CIF and $720p$ HD content insert an IDR access unit approximately every 500 ms. The periodic insertion of an IDR access unit enables fast random access which is often required in multimedia applications [46].

The sequences are coded with QP_I values (for I slices) 22, 27, and 32, $QP_P = QP_I + 1$ values (for P slices), and $QP_B = QP_I + 2$ values (for B slices). These correspond to the values used in the VCEG common test conditions [47]. In the remainder of this section, we only mention the QP value for the I pictures.

The requantization architectures are implemented in software which is used to generate the transrated video bitstreams. The transrated video bitstreams are generated using increasing QP. The ΔQP ranges from 1 to 6 for all slices in the coded H.264/AVC video bitstream. Since no rate control

mechanism is used, it is possible to measure the intrinsic power of the requantization architectures without having an impact on the transrating speed. In order to cover realistic bit rate constraints, we focus on bit rate reductions up to 40%.

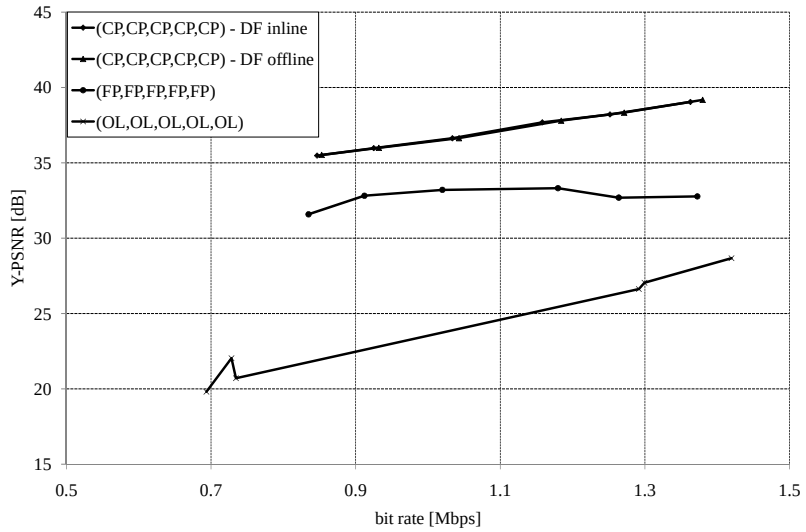
4.4.2 Rate-distortion results for basic transrating architectures

The rate-distortion results for the basic transrating architectures are shown in Figure 4.14 for CIF sequences and Figure 4.15 for $720p$ HD sequences.

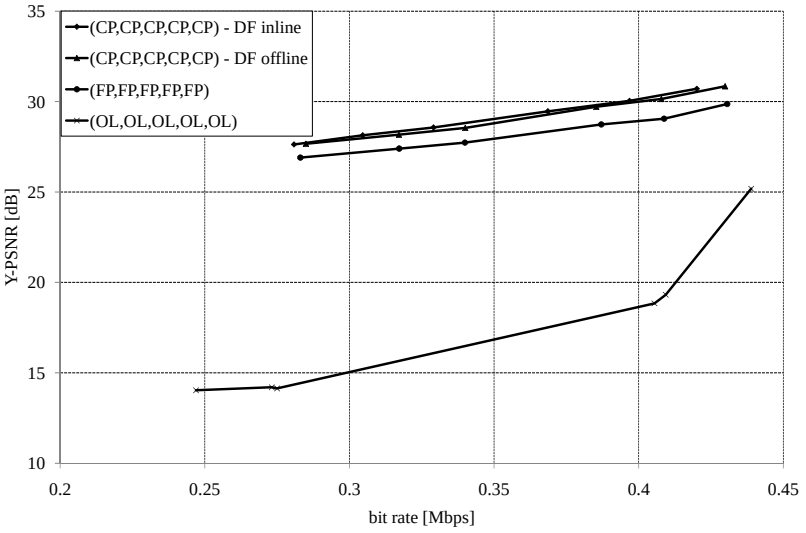
The OL transrater applies requantization without considering the dependencies in the video bitstream. The requantization results in errors which result in unpredictable drift. The PSNR values are below 35 dB for small bit rate reductions and decrease further to 20 dB for higher bit rate reductions. This results in severe quality degradation, rendering this transrating architecture useless. However, this architecture only applies entropy decoding, requantization and entropy coding and therefore has the lowest computational complexity; in this way, it can serve as a reference for processing speed. The OL transrater does not always result in monotonic behavior due to randomness of drift propagation. This can be seen in Figure 4.14(a) for OL transrating with $\Delta QP = 4$ to $\Delta QP = 6$. Another observation shows a significant gap in bit rate for OL transrating with $\Delta QP = 3$ and $\Delta QP = 4$. This effect results from the removal of small transform coefficients. Similar observations can be made for transrating of $720p$ HD material.

The CP transrating consists of decoding followed by encoding. The mode and motion data of the decoding process is used for the encoding process. The transrater consists of two prediction loops, so drift propagation is completely avoided. This solution has the best rate-distortion results. Minimal quality degradation is found for a given transrating ratio. The rate-distortion graphs for CP transrating are monotonic decreasing. The PSNR values decrease with about 5 dB for a reduction of the bit rate of approximately 50% when CP transrating with $\Delta QP = 6$ is applied. This is found for both CIF and $720p$ HD material. There is almost no difference between the CP transrater with DF inline and the CP transrater with DF offline. As a result, the best solution can be chosen in function of system requirements.

The FP transrating applies requantization where the requantization error from the reference block is compensated. Due to intra prediction and motion-compensated prediction, both spatial and temporal compensation is provided which results in one prediction loop. The compensation is not perfect and results in rounding errors. The rate-distortion curves are not always monotonic. This comes from the rounding errors which propagate and accumulate. For

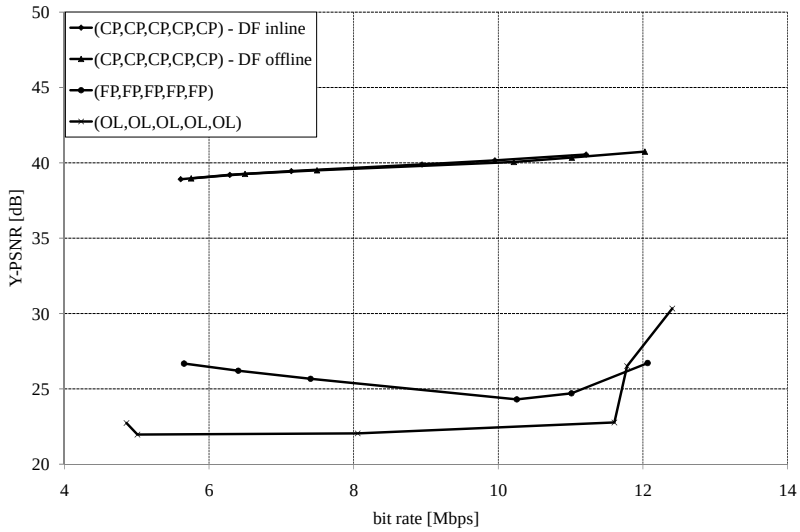


(a) *Paris* sequence, IBBP coding, $QP_I = 22$

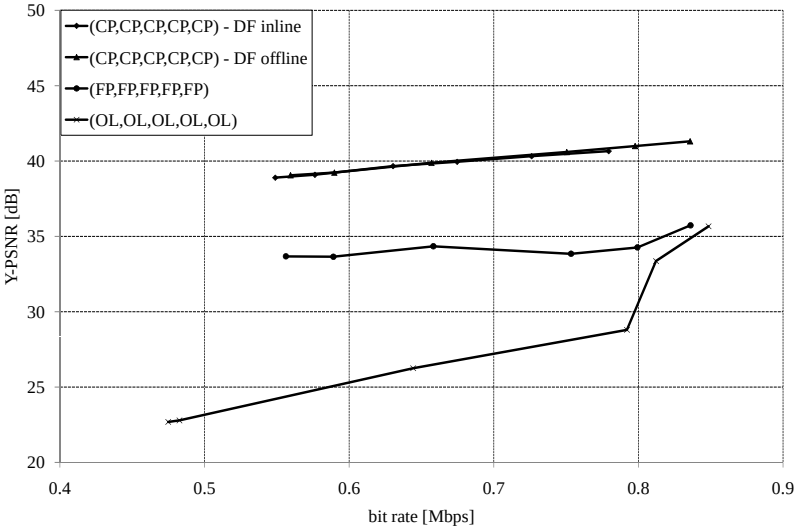


(b) *Stefan* sequence, hierarchical coding, $QP_I = 32$

Figure 4.14: Rate-distortion results for the basic transrating architectures for Main Profile, CIF resolution, 30 fps.



(a) *Crew* sequence, IBBBP coding, $QP_I = 22$



(b) *Shuttlestart* sequence, hierarchical coding, $QP_I = 27$

Figure 4.15: Rate-distortion results for the basic transrating architectures for IBBBP coding, High Profile, $720p$ HD resolution, 60 fps, $QP_I = 22$.

CIF sequences, the PSNR values are between 2 and 10 dB below the results for CP transrating. For $720p$ HD sequences, the PSNR gap further increases. Transrating results for I pictures (see Section 4.2.2) have shown that FP transrating introduces many artifacts. This way, FP transrating should be used for spatial and temporal compensation in P and B pictures.

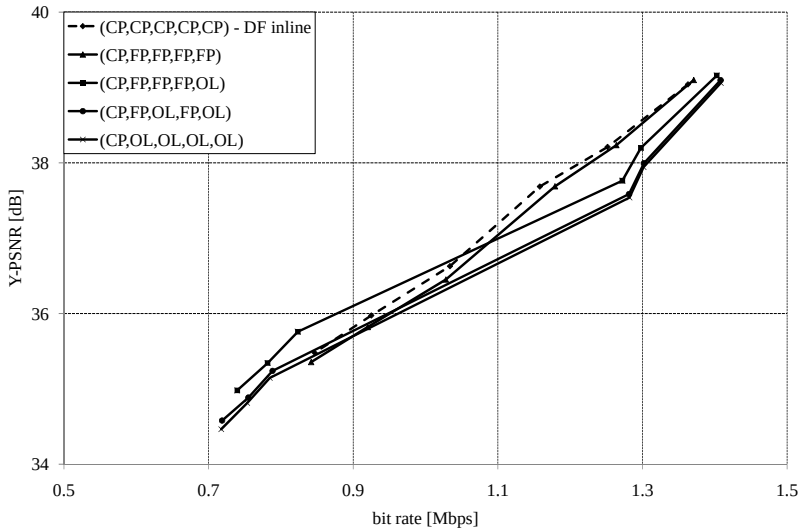
4.4.3 Rate-distortion results for mixed transrating architectures

The rate-distortion results for the mixed transrating architectures are found in Figure 4.16 for the CIF sequences and Figure 4.18 for the HD sequences.

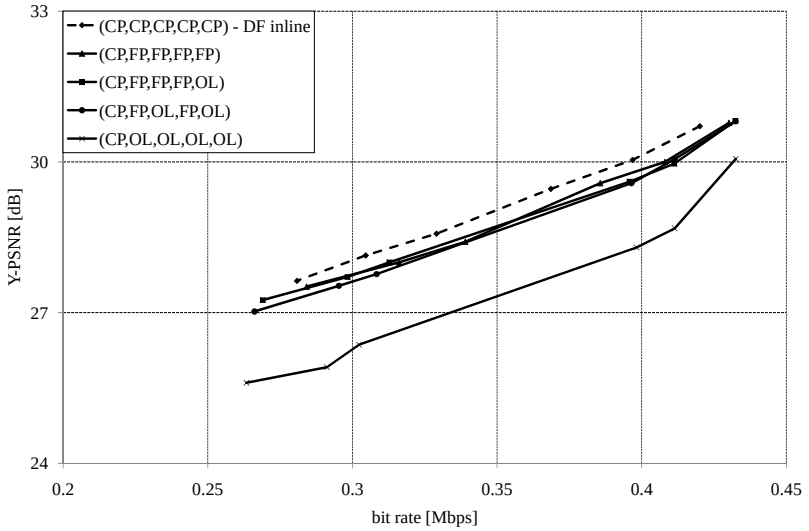
The mixed architectures provide rate-distortion results in between. The mixed architecture which applies CP transrating to I pictures and OL transrating to P and B pictures (CP, OL, OL, OL, OL), already approaches the rate-distortion performance of the CP architecture within 4 dB. From this, the importance of re-encoding the I pictures in the video bitstream becomes clear. This architecture is comparable with the OL transrater in terms of computational complexity. This results from the fact that the computational complexity of the intra prediction is marginal, when compared to the motion vector derivation and quarter-pixel motion-compensated prediction. The use of OL transrating for P and B pictures, however, will lead to a decline in visual quality over a longer GOP. In particular, spatially propagating drift can result in artifacts in the pictures and a drop in visual quality. For this reason, the (CP, OL, OL, OL, OL) architecture is restricted in its applicability for transrating of H.264/AVC bitstreams.

Spatial compensation performs particularly well, while not applying spatial compensation leads to losses up to 4 dB for larger transrating ratios. These compensation techniques require slightly more processing power compared with the OL transrating and thus result in a transrater solution which is somewhat slower than the OL transrater. The (CP, FP, OL, FP, OL) architecture is constructed in this way, with a speed loss of up to 30%.

Temporal compensation for P pictures further improves the visual quality, in particular for low to medium transrating ratios. Using temporal compensation for B pictures leads to little or no additional gain when compared to OL requantization of B macroblocks. Temporal compensation demands motion vector derivation and quarter-pixel motion-compensated prediction; as can be expected, these coding tools have a major impact on the total processing time. However, applying both spatial and temporal compensation result in a transrating architecture (CP, FP, FP, FP, FP) which almost doubles the transrating speed and halves the memory requirements compared to the CP architecture, while having a loss of 1 to 2 dB in visual quality. For nearly identical quality

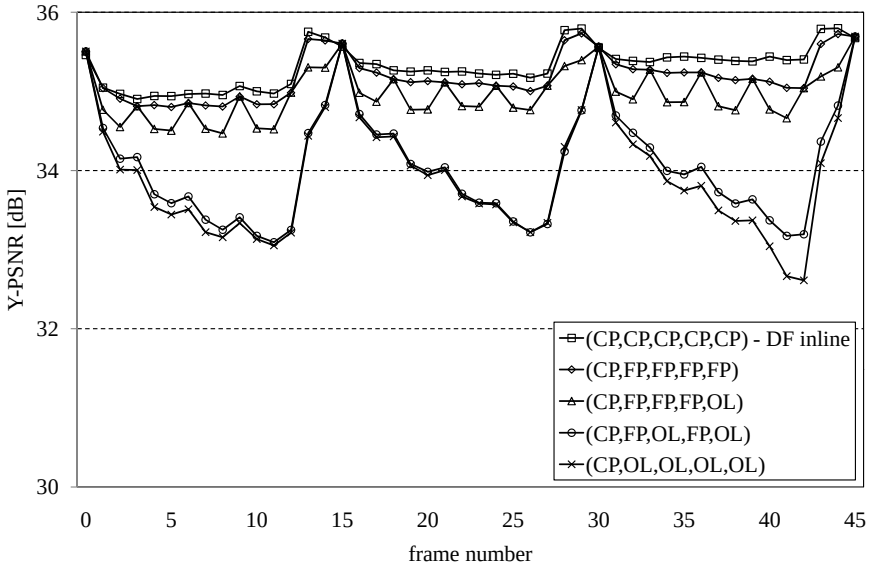


(a) *Paris* sequence, IBBP coding, $QP_I = 22$

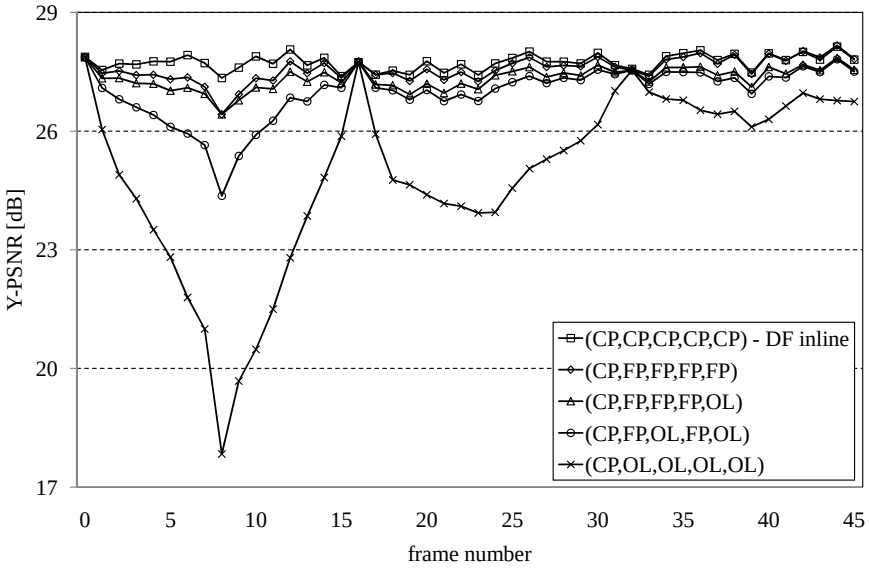


(b) *Stefan* sequence, hierarchical coding, $QP_I = 32$

Figure 4.16: Rate-distortion results for the mixed transrating architectures for Main Profile, CIF resolution, 30 fps.

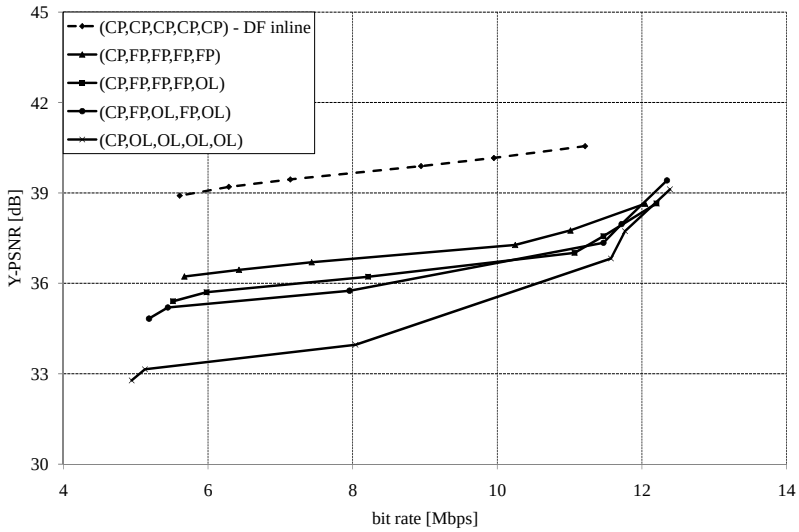


(a) *Paris* sequence, IBBP coding, $QP_I = 22$

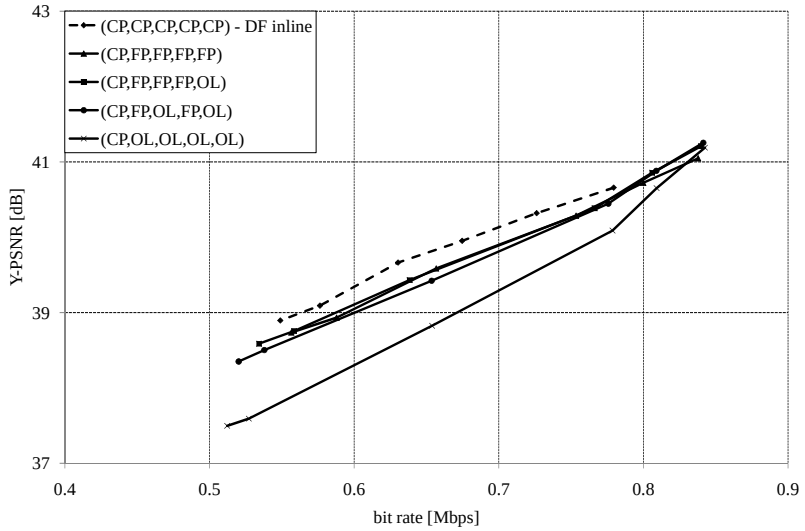


(b) *Stefan* sequence, hierarchical coding, $QP_I = 32$

Figure 4.17: PSNR results for the mixed architectures for Main Profile, CIF resolution, 30 fps: $\Delta QP = 6$ for all transrating architectures.



(a) *Crew* sequence, IBBBP coding, $QP_I = 22$



(b) *Shuttlestart* sequence, hierarchical coding, $QP_I = 27$

Figure 4.18: Rate-distortion results for the mixed transrating architectures for High Profile, $720p$ HD resolution, 60 fps.

results, the (CP, FP, FP, FP, OL) architecture leads to a speed-up by a factor of four when compared to CP transrating.

The PSNR values of the 45 first pictures of the *Paris* and *Stefan* sequences are shown in Figure 4.17(a) and Figure 4.17(b), respectively. The figures show the evolution of the temporal drift from picture to picture. For IBBP coding, we selected $\Delta QP = 6$ for all transrating architectures and we can see that the maximum quality loss is up to 3 dB near the end of the third GOP. For hierarchical coding, we selected $\Delta QP = 6$ for all transrating architectures and we can see that the maximum quality loss is up to 10 dB in the middle of the first GOP. The PSNR values fluctuate a lot with the period of the GOP which results in the "pumping" or "breathing" artifact commonly known in industry.

4.4.4 Visual quality

Besides rate-distortion results, visual results clearly show the impact of drift propagation on the transrated video bitstreams. Figure 4.19 and Figure 4.20 show visual results for the 260^{th} frame of the *Crew* sequence (B picture in the middle of a GOP). The decoded frame is shown in Figure 4.19(a), while the transrated frames for OL transrating, FP transrating, and CP transrating are shown in Figure 4.19(b), Figure 4.19(c), and Figure 4.19(d). Figure 4.20 shows the impact of spatial and temporal compensation in the mixed architecture: no compensation (see Figure 4.20(a)), only spatial compensation (see Figure 4.20(b)), spatial compensation for both P and B pictures and temporal compensation for P pictures (see Figure 4.20(c)), and spatial compensation and temporal compensation for both P and B pictures (see Figure 4.20(d)).

4.4.5 Transrating speed

We measured the processing time for transrating of H.264/AVC bitstreams according to eight transrating modes (four basic transrating architectures and four mixed transrating architectures) and we calculated the number of pictures which are processed every second. The average transrating speed for the *Foreman* sequence is presented for IBBP coding and hierarchical coding in Table 4.1 and Table 4.2, respectively. The average transrating speed for the *Crew* sequence is presented for IBBP coding and hierarchical coding in Table 4.3 and Table 4.4, respectively. These results are obtained from non-optimized software and serve as an indication of the complexity of the different architectures. Many optimizations are possible in order to speed up the rate shaping architectures; however, it does not belong to the scope of this work. These timing results were generated on a platform with an Intel Xeon X5355 processor

(a) Decoded frame

(b) (OL,OL,OL,OL,OL) for $\Delta QP = 4$, Y-PSNR = 19.35 dB

(c) (FP,FP,FP,FP,FP) for $\Delta QP = 4$, Y-PSNR = 28.04 dB

(d) (CP,CP,CP,CP,CP) for $\Delta QP = 4$, Y-PSNR = 34.99 dB

Figure 4.19: Visual results for basic architectures for the 260^{th} picture of the *Crew* sequence (HD 720p, 60 fps, $QP_I = 27$, IBBBP).

(a) (CP,OL,OL,OL,OL) for $\Delta QP = 4$, Y-PSNR = 22.64 dB

(b) (CP,FP,OL,FP,OL) for $\Delta QP = 4$, Y-PSNR = 30.65 dB

(c) (CP,FP,FP,FP,OL) for $\Delta QP = 4$, Y-PSNR = 30.74 dB

(d) (CP,FP,FP,FP,FP) for $\Delta QP = 4$, Y-PSNR = 32.48 dB

Figure 4.20: Visual results for mixed architectures for the 260^{th} picture of the *Crew* sequence (HD 720p, 60 fps, $QP_I = 27$, IBBBP).

and 16 GB RAM in a Microsoft Windows XP environment.

Table 4.1: Average transrating speed [fps] for requantization transcoding over all ΔQP values for the *Foreman* sequence, IBBP coding, Main Profile.

	QP_I		
	22	27	32
(OL, OL, OL, OL, OL)	57.36	62.51	66.91
(FP, FP, FP, FP, FP)	8.10	8.32	8.47
(CP, CP, CP, CP, CP) - DF inline	4.83	4.94	5.07
(CP, CP, CP, CP, CP) - DF offline	4.89	5.04	5.13
(CP, OL, OL, OL, OL)	42.73	47.60	52.03
(CP, FP, OL, FP, OL)	40.84	46.46	52.22
(CP, FP, FP, FP, OL)	20.11	20.59	20.91
(CP, FP, FP, FP, FP)	7.94	8.16	8.33

Table 4.2: Average transrating speed [fps] for requantization transcoding over all ΔQP values for the *Foreman* sequence, hierarchical coding, Main Profile.

	QP_I		
	22	27	32
(OL, OL, OL, OL, OL)	58.30	65.69	67.44
(FP, FP, FP, FP, FP)	7.68	7.70	7.52
(CP, CP, CP, CP, CP) - DF inline	4.60	4.63	4.54
(CP, CP, CP, CP, CP) - DF offline	4.69	4.70	4.59
(CP, OL, OL, OL, OL)	44.06	49.54	56.29
(CP, FP, OL, FP, OL)	42.94	48.70	51.90
(CP, FP, FP, FP, OL)	27.83	28.72	29.58
(CP, FP, FP, FP, FP)	7.52	7.57	7.39

The OL transrater (used as reference in the comparison) clearly has the lowest complexity, while the CP transrater requires most processing power (factor of about 10 to 15 more). The mixed architectures with and without spatial compensation result in low-complexity solutions which approach the transrating speed of the OL transrater, while the mixed architecture with both spatial and temporal compensation substantially increases computational complexity. The temporal compensation, which requires motion vector derivation and quarter-pixel motion-compensated prediction, has a major impact on the total processing time.

Table 4.3: Average transrating speed [fps] for requantization transcoding over all ΔQP values for the *Crew* sequence, IBBBP coding, High Profile.

	QP_I		
	22	27	32
(OL, OL, OL, OL, OL)	4.06	4.60	5.12
(FP, FP, FP, FP, FP)	0.84	0.83	0.83
(CP, CP, CP, CP, CP) - DF inline	0.51	0.51	0.52
(CP, CP, CP, CP, CP) - DF offline	0.51	0.51	0.52
(CP, OL, OL, OL, OL)	4.91	5.99	6.89
(CP, FP, OL, FP, OL)	3.98	5.00	5.57
(CP, FP, FP, FP, OL)	2.31	2.37	2.48
(CP, FP, FP, FP, FP)	0.82	0.83	0.81

Table 4.4: Average transrating speed [fps] for requantization transcoding over all ΔQP values for the *Crew* sequence, hierarchical coding, High Profile.

	QP_I		
	22	27	32
(OL, OL, OL, OL, OL)	3.86	4.57	4.64
(FP, FP, FP, FP, FP)	0.78	0.77	0.75
(CP, CP, CP, CP, CP) - DF inline	0.46	0.47	0.47
(CP, CP, CP, CP, CP) - DF offline	0.46	0.48	0.47
(CP, OL, OL, OL, OL)	5.90	6.58	6.92
(CP, FP, OL, FP, OL)	4.52	5.55	5.74
(CP, FP, FP, FP, OL)	2.82	2.94	2.89
(CP, FP, FP, FP, FP)	0.77	0.76	0.74

4.4.6 Memory requirements

Horowitz *et al.* made a complexity analysis of an H.264/AVC decoder [49] and found that the storage requirements can be divided into different classes: 1) frame memory (i.e., reconstructed frame memory and reference frame memory), 2) macroblock memory (i.e., memory for values from neighboring macroblocks which are used during intra prediction and deblocking and memory for storing transform coefficients, prediction values, and pixel values), and 3) memory for constant data (i.e., tables for entropy coding, etc.). They found that frame memory dominates the storage requirements. Besides memory for the transrating architecture, a processing system needs bit buffers which store the receiving and sending bits. The memory size of these buffers does not

depend on the transrating architecture.

In this section, we examine in more detail the frame memory requirements for different transrating architectures. No frame memory is required for OL transrating. The FP transrater needs memory for storing requantization differences (9-bit memory for storing difference values in the range [-255, +255]) while the CP transrater requires twice the amount of memory for storing reconstructed pixel values for decoder and encoder (8-bit memory for storing decoded pixel values). Let n be the number of reference frames and w and h the width and the height of the pictures, respectively. We summarize the memory requirements for different transrating architectures in Table 4.5.

Table 4.5: Memory requirements for basic transrating architectures.

Architecture	α	β	γ	δ	ϵ
OL	-	-	-	-	-
FP (9-bit)	$h \cdot w$	$h \cdot w$	$n \cdot h \cdot w$	$h \cdot w$	$n \cdot h \cdot w$
CP (8-bit)	$2 \cdot h \cdot w$	$2 \cdot h \cdot w$	$2 \cdot n \cdot h \cdot w$	$2 \cdot h \cdot w$	$2 \cdot n \cdot h \cdot w$

When the pictures are processed successively, the allocated memory can be reused. As a result, we obtain the memory requirements for the basic and mixed architectures as shown in Table 4.6. These results show that the memory requirements are mainly determined by the transrating architecture for the motion-compensated blocks.

Table 4.6: Memory requirements for basic and mixed transrating architectures.

Architecture	Total memory
(OL, OL, OL, OL, OL)	-
(FP, FP, FP, FP, FP)	$h \cdot w$ (9-bit) + $n \cdot h \cdot w$ (9-bit)
(CP, CP, CP, CP, CP)	$2 \cdot h \cdot w$ (8-bit) + $2 \cdot n \cdot h \cdot w$ (8-bit)
(CP, OL, OL, OL, OL)	$2 \cdot h \cdot w$ (8-bit)
(CP, FP, OL, FP, OL)	$2 \cdot h \cdot w$ (8-bit) + $h \cdot w$ (9-bit)
(CP, FP, FP, FP, OL)	$2 \cdot h \cdot w$ (8-bit) + $h \cdot w$ (9-bit) + $n \cdot h \cdot w$ (9-bit)
(CP, FP, FP, FP, FP)	$2 \cdot h \cdot w$ (8-bit) + $h \cdot w$ (9-bit) + $n \cdot h \cdot w$ (9-bit)

4.5 Conclusions and original contributions

This chapter presents a comparison of different transrating architectures for H.264/AVC video bitstreams. Firstly, basic transrating architectures are dis-

cussed and their strengths and weaknesses are presented. The open-loop transrater is a low-complexity solution, however, this transrater results in severe quality degradation when transrating H.264/AVC video bitstreams. The cascaded pixel-domain transrater preserves the visual quality at the expense of high computational complexity (due to the advanced coding tools of the H.264/AVC specification). Secondly, mixed transrating architectures are presented which combine different transrating techniques for P and B pictures. Spatial compensation seems to be necessary in order to reduce drift propagation and preserve sufficient visual quality. Temporal compensation highly increases the computational complexity of the transrating architecture, but the improvement in visual quality is rather limited for P pictures and negligible for B pictures. The trade-off between visual quality and complexity points out that spatial compensation is required and that temporal compensation can be considered. However, temporal compensation is overkill when transrating system resources are limited.

The work that was presented in this chapter can also be found in the following publications:

- Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Requantization transcoding for H.264/AVC video coding. *Elsevier, Signal Processing: Image Communication*, Volume 25, Issue 4, pages 235–254. April 2010.

- Stijn Notebaert, Jan De Cock, Samie Beheydt, Jan De Lameillieure, and Rik Van de Walle. Mixed architectures for H.264/AVC digital video transcoding. *Springer, Multimedia Tools and Applications*, Volume 44, Issue 1, pages 39–64. August 2009.

- Stijn Notebaert, Jan De Cock, Peter Lambert, and Rik Van de Walle. Rate-controlled requantization transcoding for H.264/AVC video streams. In *Proceedings of SPIE, Applications of Digital Image Processing XXXI*, Volume 7073, San Diego, CA, USA, August 2008.

- Stijn Notebaert, Jan De Cock, Peter Lambert, and Rik Van de Walle. Requantization transcoding for reduced-complexity H.264/AVC video coding applications. In *Proceedings of the IASTED International Conference on Signal and Image Processing (SIP)*, Honolulu, HI, USA, August 2007.

- Stijn Notebaert, Jan De Cock, and Rik Van de Walle. Improved H.264/AVC requantization transcoding using low-complexity interpolation filters for 1/4-pixel motion compensation. In *Proceedings of the*

IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP), pages 307–312, Honolulu, HI, USA, April 2007.

- Jan De Cock, Stijn Notebaert, and Rik Van de Walle. Combined SNR and temporal scalability for H.264/AVC using requantization transcoding and hierarchical B pictures. In *Proceedings of the IEEE International Conference on Multimedia & Expo (ICME)*, pages 448–451, Beijing, China, July 2007.

- Jan De Cock, Stijn Notebaert, and Rik Van de Walle. A novel hybrid requantization transcoding scheme for H.264/AVC. In *Proceedings of the International Symposium on Signal Processing and its Applications (ISSPA)*, Sharjah, UAE, February 2007.

- Stijn Notebaert, Jan De Cock, Koen De Wolf, and Rik Van de Walle. Requantization transcoding of H.264/AVC bitstreams for intra 4×4 prediction modes. *Pacific-Rim Conference on Multimedia (PCM)*, Springer, Lecture Notes in Computer Science, Volume 4261, Hangzhou, China, November 2006.

- Jan De Cock, Stijn Notebaert, Peter Lambert, Davy De Schrijver, and Rik Van de Walle. Requantization transcoding in pixel and frequency domain for intra 16×16 in H.264/AVC. *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, Springer, Lecture Notes in Computer Science, Volume 4179, Antwerp, Belgium, September 2006.

- Stijn Notebaert, Jan De Cock, Davy De Schrijver, Koen De Wolf, and Rik Van de Walle. Quality analysis of requantization transcoding architectures for H.264/AVC. In *Proceedings of SPIE, Applications of Digital Image Processing XXIX*, Volume 6312, San Diego, CA, USA, August 2006.

Chapter 5

Conclusions

In many applications, there is a need for adaptation of video bitstreams. One important adaptation is the reduction of the bit rate. This constraint results from limited bandwidth on the network or limited resources of a device. The bit rate reduction can be achieved by different methods: *rate shaping* and *requantization*. Both methods reduce the bit rate by decreasing the amount of residual data. Rate shaping drops transform coefficients while requantization applies a coarser quantizer. These transrating methods have been investigated in the context of H.264/AVC in this work.

The first method for reducing the bit rate, called rate shaping, is studied in Chapter 2. This method was first presented in the 90s as a simple operation for reducing the bit rate of MPEG-2 Video. Rate shaping for H.264/AVC is complicated due to the improved entropy, transform, and predictive coding.
We show that extensions are required in order to apply rate shaping to H.264/AVC. Firstly, the entropy coding is more complex compared to the counterpart in MPEG-2 Video. As a result, it is impossible to directly operate on the video bitstream. Entropy decoding and encoding is necessary at the input and output of the rate shaper. This makes the solution more complex; however, it is necessary in order to change the residual data. Secondly, different transforms and different block sizes are combined in H.264/AVC. When a two-pass transform is used, the low-frequency coefficients from the first transform are the input to the second transform. The output of the second transform only contains low-frequency data. Since the human visual system is sensitive for low-frequency data, we cannot remove these coefficients. Therefore we only operate on the high-frequency data from the output of the first transform. We also introduce a virtual breakpoint in order to remove an equivalent part of data from the frequency spectrum for different block sizes. This way the effective breakpoint is based on the virtual breakpoint and the block size.

These extensions allow rate shaping of H.264/AVC. The first simulation results showed severe degradation of the visual quality. For the same reduction in bit rate, rate shaping of I pictures has more impact on the visual quality than rate shaping of P and B pictures. The spatial drift is significant in I pictures and the P and B pictures are affected by motion-compensated prediction. Therefore, rate shaping of I pictures should be avoided. When rate shaping is applied to P and B pictures, both spatial and temporal drift are found. However, the visual quality is better compared to rate shaping of I pictures. We also provided an analysis of the drift and showed that the drift can be compensated. This technique has been proposed for requantization transrating.
Afterwards, we investigate the performance of rate shaping the P and B pictures of H.264/AVC video bitstreams. In order to preserve the visual quality, we propose to use compensation which was earlier presented for requantization. The rate shaping results show that significant gain is found when compensation is used. The gains depend on the characteristics of the video bitstreams. The spatial compensation is better than temporal compensation when visual quality and computational resources are considered.
Additionally, we investigate the complexity of the rate shaping solution. The rate-distortion optimal solution is far too complex, so a low-complexity alternative which defines one breakpoint for a set of blocks is more appropriate. This was also proposed for rate shaping of MPEG-2 Video as a low-complexity alternative.

The second method for reducing the bit rate is requantization. An important problem for requantization deals with the characteristics of the quantizer. The selection of an appropriate quantizer in the transrater is called the requantization problem. A technique called perfect requantization has been presented in the literature. This technique eliminates the requantization errors by imposing conditions on the quantizer in the transrater. This technique is not suitable for transrating of H.264/AVC.
In Chapter 3, we present a different approach which derives a requantization heuristic based on theoretical rate-distortion results of a Laplace source which is quantized twice. In order to simplify the calculations for entropy and distortion, we provide a solution based on the characteristics of a Laplace source and the effective quantizer. The effective quantizer is derived as the superposition of the first-step and second-step quantizers and generates the same output as successively applying the first-step and second-step quantizers. The memoryless property from a Laplace source is combined with the periodic property of the effective quantizer in order to derive expressions for entropy and distortion. The requantization heuristic is derived from the theoretical results for different quantization and requantization. We found that increasing the quantizer

step size with fixed quantizer offset is a good solution when fine quantization is applied in the encoder. When the quantization in the encoder is coarse, a better solution consists of decreasing the quantizer offset with fixed quantizer step size. The requantization heuristic was implemented in transrating software. We found gains up to 1 dB for open-loop requantization of B pictures compared to requantization with fixed quantizer step size.

The selection of an appropriate architecture for requantization transrating may have even more impact on the transrating performance. In the past, many transrating solutions have been presented that apply one transrating method to all blocks in the video bitstream. These solutions often do not meet transrating requirements such as fast and efficient transrating. In Chapter 4, we make an evaluation of transrating methods in the context of H.264/AVC and propose mixed architectures which select an appropriate transrating method adaptively. We start with an investigation of the basic transrating architectures in the context of H.264/AVC: the open-loop transrater, the fast pixel-domain transrater, and the cascaded pixel-domain transrater. The open-loop transrater has low-complexity, but introduces spatial and temporal drift. The drift degrades the visual quality which makes this transrating architecture not useful. The fast-pixel domain transrater compensates for the requantization errors; however, the prediction loop adds extra complexity and still introduces small rounding errors. The cascaded pixel-domain transrater maintains two prediction loops which results in drift-free transrating while the complexity is increased compared to open-loop transrating and fast pixel-domain transrating. We found that none of these architectures meet the transrating requirements.
Afterwards, we proposed mixed transrating architectures which apply different transrating techniques depending on the picture/macroblock type. The cascaded pixel-domain transrater is used for I pictures, so drift-free transrating is applied and the P and B pictures use an I picture with reduced quality without spatial drift. The open-loop transrater or the fast-pixel domain transrater is selected for P and B pictures. The spatial compensation shows a significant improvement which in addition required hardly more computational resources. The temporal compensation only improves slightly with more need for processing power.

Efforts are ongoing in MPEG and VCEG in order to develop and standardize new video coding developments. One of the requirements will be higher compression efficiency compared to prior video coding standards. These efforts are targeting on applications where higher resolutions are used.
Even more intelligent solutions will be necessary for transrating these video bitstreams. The coding tools will be more complex, additional tools will be provided; the number of modes and partitions will be higher. This compli-

cates the search for efficient transrating solutions and as a result, new research efforts are required in order to find an appropriate solution.

Appendix A

Publication list

A.1 Journal papers

1. Kenneth Vermeirsch, Jan De Cock, Stijn Notebaert, Peter Lambert, Joeri Barbarien, Adrian Munteanu, and Rik Van de Walle. Efficient adaptive-shape partitioning of video. *Springer, Multimedia Tools and Applications*. Conditionally accepted.

2. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Motion-refined rewriting of H.264/AVC-coded video to SVC streams. *Elsevier, Visual Communication and Image Representation*. Conditionally accepted.

3. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Requantization transcoding for H.264/AVC video coding. *Elsevier, Signal Processing: Image Communication*, Volume 25, Issue 4, pages 235–254. April 2010.

4. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Dyadic spatial resolution reduction transcoding for H.264/AVC. *Springer, Multimedia Systems*, Volume 16, Number 2, pages 139–149. March 2010.

5. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Architectures for fast transcoding of H.264/AVC to quality-scalable SVC streams. *IEEE Transactions on Multimedia*, Volume 11, Number 7, pages 1209–1224. November 2009.

6. Stijn Notebaert, Jan De Cock, Samie Beheydt, Jan De Lameillieure, and Rik Van de Walle. Mixed architectures for H.264/AVC digital video

transcoding. *Springer, Multimedia Tools and Applications*, Volume 44, Issue 1, pages 39–64. August 2009.

7. Jan De Cock, Stijn Notebaert, Peter Lambert, Davy De Schrijver, and Rik Van de Walle. Requantization transcoding in pixel and frequency domain for intra 16×16 in H.264/AVC. *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, Springer, Lecture Notes in Computer Science, Volume 4179, Antwerp, Belgium, September 2006.

8. Robbie De Sutter, Stijn Notebaert, and Rik Van de Walle. Evaluation of metadata standards in the context of digital audio-visual libraries. *European Conference on Research and Advanced Technology for Digital Libraries (ECDL)*, Springer, Lecture Notes in Computer Science, Volume 4172, Alicante, Spain, September 2006.

A.2 Submitted journal papers

1. Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Rate shaping for H.264/AVC coded video. *Submitted to IEEE Transactions on Multimedia*.

2. Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Quantizer offset selection for improved requantization transcoding. *Submitted to Elsevier, Signal Processing: Image Communication*.

A.3 Conference papers

1. Jan De Cock, Stijn Notebaert, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Transcoding of H.264/AVC to SVC with motion data refinement. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 3673–3676, Caïro, Egypt, November 2009.

2. Kenneth Vermeirsch, Stijn Notebaert, Jan De Cock, Peter Lambert, and Rik Van de Walle. Transformation techniques for future video coding. In *Proceedings of the International Mobile Multimedia Communications Conference (MobiMedia)*, London, UK, September 2009.

3. Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Leveraging the quantization offset for improved re-

quantization transcoding of H.264/AVC video. In *Proceedings of the Picture Coding Symposium (PCS)*, Chicago, IL, USA, May 2009.

4. Kenneth Vermeirsch, Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Evaluation of transform performance when using shape-adaptive partitioning in video coding. In *Proceedings of the Picture Coding Symposium (PCS)*, Chicago, IL, USA, May 2009.

5. Kenneth Vermeirsch, Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Extended macroblock bipartitioning modes for H.264/AVC inter coding. In *Proceedings of the IEEE International Symposium on Multimedia (ISM)*, pages 142–147, Berkeley, CA, USA, December 2008.

6. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Advanced bitstream rewriting from H.264/AVC to SVC. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 2472–2475, San Diego, CA, USA, October 2008.

7. Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Improved dynamic rate shaping for H.264/AVC video streams. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 1616–1619, San Diego, CA, USA, October 2008.

8. Jan De Cock, Stijn Notebaert, Kenneth Vermeirsch, Peter Lambert, and Rik Van de Walle. Efficient spatial resolution reduction transcoding for H.264/AVC. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 1208–1211, San Diego, CA, USA, October 2008.

9. Kenneth Vermeirsch, Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Increased flexibility in inter picture partitioning. In *Proceedings of the International Workshop on Multimedia Signal Processing (MMSP)*, pages 284–288, Cairns, Queensland, Australia, October 2008.

10. Stijn Notebaert, Jan De Cock, Peter Lambert, and Rik Van de Walle. Rate-controlled requantization transcoding for H.264/AVC video streams. In *Proceedings of SPIE, Applications of Digital Image Processing XXXI*, Volume 7073, San Diego, CA, USA, August 2008.

11. Maarten Wijnants, Wim Lamotte, Bart De Vleeschauwer, Filip De Turck, Bart Dhoedt, Piet Demeester, Peter Lambert, Dieter Van de

Walle, Jan De Cock, Stijn Notebaert, and Rik Van de Walle. Optimizing user QoE through overlay routing, bandwidth management and dynamic transcoding. In *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM)*, Newport Beach, CA, USA, June 2008.

12. Jan De Cock, Stijn Notebaert, and Rik Van de Walle. Transcoding from H.264/AVC to SVC with CGS layers. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 1769–1772, San Antonio, TX, USA, November 2007.

13. Stijn Notebaert, Jan De Cock, Peter Lambert, and Rik Van de Walle. Requantization transcoding for reduced-complexity H.264/AVC video coding applications. In *Proceedings of the IASTED International Conference on Signal and Image Processing (SIP)*, Honolulu, HI, USA, August 2007.

14. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Bridging the gap: transcoding from single-layer H.264/AVC to scalable SVC video streams. *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, Springer, Lecture Notes in Computer Science, Volume 4678, Delft, the Netherlands, August 2007.

15. Jan De Cock, Stijn Notebaert, and Rik Van de Walle. Combined SNR and temporal scalability for H.264/AVC using requantization transcoding and hierarchical B pictures. In *Proceedings of the IEEE International Conference on Multimedia & Expo (ICME)*, pages 448–451, Beijing, China, July 2007.

16. Stijn Notebaert, Jan De Cock, and Rik Van de Walle. Improved H.264/AVC requantization transcoding using low-complexity interpolation filters for 1/4-pixel motion compensation. In *Proceedings of the IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP)*, pages 307–312, Honolulu, HI, USA, April 2007.

17. Stijn Notebaert, Jan De Cock, Kenneth Vermeirsch, and Rik Van de Walle. Complexity and quality assessment of MPEG-2 to H.264/AVC intra transcoding architectures. In *Proceedings of the International Symposium on Signal Processing and its Applications (ISSPA)*, Sharjah, UAE, February 2007.

18. Jan De Cock, Stijn Notebaert, and Rik Van de Walle. A novel hybrid requantization transcoding scheme for H.264/AVC. In *Proceedings of*

the International Symposium on Signal Processing and its Applications (ISSPA), Sharjah, UAE, February 2007.

19. Stijn Notebaert, Jan De Cock, Koen De Wolf, and Rik Van de Walle. Requantization transcoding of H.264/AVC bitstreams for intra 4×4 prediction modes. *Pacific-Rim Conference on Multimedia (PCM)*, Springer, Lecture Notes in Computer Science, Volume 4261, Hangzhou, China, November 2006.

20. Jan De Cock, Stijn Notebaert, Peter Lambert, Koen De Wolf, and Rik Van de Walle. Low-complexity SNR transcoding for H.264/AVC. In *Proceedings of the IASTED International Conference on Communications, Internet and Information Technology (CIIT)*, St. Thomas, US Virgin Islands, November 2006.

21. Stijn Notebaert, Jan De Cock, Davy De Schrijver, Koen De Wolf, and Rik Van de Walle. Quality analysis of requantization transcoding architectures for H.264/AVC. In *Proceedings of SPIE, Applications of Digital Image Processing XXIX*, Volume 6312, San Diego, CA, USA, August 2006.

22. Davy De Schrijver, Wesley De Neve, Koen De Wolf, Stijn Notebaert, and Rik Van de Walle. XML-based customization along the scalability axes of H.264/AVC scalable video coding. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 465-468, Island of Kos, Greece, May 2006.

23. Robbie De Sutter, Stijn Notebaert, Laurence Hauttekeete, and Rik Van de Walle. IPEA: The digital archives use case. In *Proceedings of the IS&T Archiving 2006*, pages 182–186, Ottawa, Canada, May 2006.

24. Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle. Hardware/software co-design for H.264/AVC intra frame encoding. In *Proceedings of the Euromedia*, pages 56–60, Athens, Greece, May 2006.

25. Wesley De Neve, Dieter Van Rijsselbergen, Charles Hollemeersch, Jan De Cock, Stijn Notebaert, and Rik Van de Walle. GPU-assisted decoding of video samples represented in the YCoCg-R color space. In *Proceedings of the ACM International Conference on Multimedia*, pages 447–450, Singapore, November 2005.

26. Yves Dhondt, Peter Lambert, Stijn Notebaert, and Rik Van de Walle. Flexible macroblock ordering as a content adaptation tool in

H.264/AVC. In *Proceedings of SPIE, Multimedia Systems and Applications VIII*, Volume 6015, Boston, MA, USA, October 2005.

Appendix B

Transform and quantization in H.264/AVC

B.1 Forward transform

The forward integer transform is implemented as:

$$\mathbf{W} = \mathbf{C}_f \mathbf{X} \mathbf{C}_f^T, \tag{B.1}$$

where

$$\mathbf{C}_f = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix}. \tag{B.2}$$

Since the integer transform is not unitary, $\mathbf{W}$ must be normalized with the post-scaling matrix $\mathbf{E}_f$:

$$\mathbf{Y} = \mathbf{W} \otimes \mathbf{E}_f, \tag{B.3}$$

where the operator $\otimes$ indicates the Hadamard product[1] and the post-scaling matrix $\mathbf{E}_f$ is given by

$$\mathbf{E}_f = \begin{pmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{pmatrix} \tag{B.4}$$

with the values $a = 1/2$ and $b = 1/\sqrt{10}$.

[1]The Hadamard product, also known as the entrywise product or the Schur product, for two matrices of the same dimensions $A, B \in \mathbb{R}^{m \times n}$ is given by $(A \otimes B) \in \mathbb{R}^{m \times n}$ and $(A \otimes B)_{i,j} = A_{i,j} \cdot B_{i,j}$.

B.2 Quantization

The quantization is derived as follows:

$$Z_{ij} = \text{Round}\left(\frac{Y_{ij}}{Q_{step}}\right) = \text{Round}\left(\frac{W_{ij}E_{ij}}{Q_{step}}\right) = \text{Round}\left(\frac{W_{ij}M_{ij}}{2^{qbits}}\right), \quad \text{(B.5)}$$

where the post-scaling operation is incorporated. M_{ij} is the multiplication factor and $qbits$ is defined as $15 + \lfloor QP/6 \rfloor$ where QP represents the quantization parameter.
This results in the following implementation:

$$|Z_{ij}| = (|Y_{ij}|\, M_{ij} + \epsilon) \gg qbits, \quad \text{(B.6)}$$

$$sign(Z_{ij}) = sign(Y_{ij}). \quad \text{(B.7)}$$

The multiplication factors M_{ij} are given in Table B.1 according to the following positions in the 4×4 block:

$$r = \begin{cases} 0, & \text{for } (i,j) = \{(0,0),(2,0),(0,2),(2,2)\} \\ 1, & \text{for } (i,j) = \{(1,1),(3,1),(1,3),(3,3)\} \\ 2, & \text{for others.} \end{cases} \quad \text{(B.8)}$$

The relation between the quantization parameter QP and the quantizer step

$QP\%6$	$r=0$	$r=1$	$r=2$
0	13107	5243	8066
1	11916	4660	7490
2	10082	4194	6554
3	9362	3647	5825
4	8192	3355	5243
5	7282	2893	4559

Table B.1: Multiplication factors M_{ij}.

size Q_{step} in H.264/AVC is that $Q_{step} = 2^{(QP-4)/6}$. This relation is shown in Figure B.1.

B.3 Rescaling

The rescaling is derived as follows:

$$Y'_{ij} = Z_{ij} Q_{step}. \quad \text{(B.9)}$$

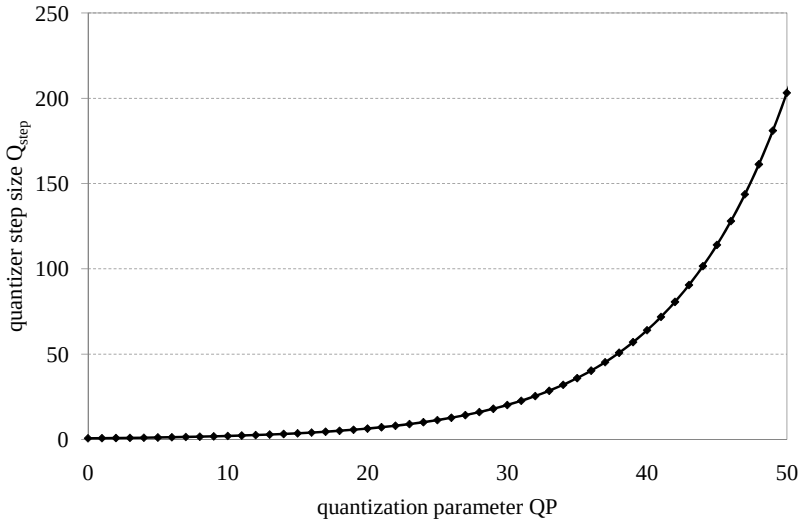


Figure B.1: Relation between the quantization parameter QP and the quantizer step size Q_{step} in H.264/AVC.

The pre-scaling factors of the inverse transform are incorporated in the rescaling operation together with a factor 64 which is used to avoid rounding errors:

$$W'_{ij} = Z_{ij} Q_{step} E_{ij} 64. \tag{B.10}$$

In this equation, W'_{ij} is a scaled coefficient which only needs to be transformed by the inverse core transform. In order to simplify the rescaling operation, multiplication factors V_{ij} are introduced together with a shift operation:

$$Y'_{ij} = Z_{ij} V_{ij} 2^{\lfloor QP/6 \rfloor}. \tag{B.11}$$

The multiplication factors V_{ij} are given in Table B.2 according to the positions in the 4×4 block.

B.4 Inverse transform

The inverse transform is implemented as:

$$\mathbf{X}' = \mathbf{C}_i^T \mathbf{W}' \mathbf{C}_i = \mathbf{C}_i^T \left(\mathbf{Y}' \odot \mathbf{E}_i \right) \mathbf{C}_i, \tag{B.12}$$

where

$$\mathbf{C}_i = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{pmatrix}. \tag{B.13}$$

$QP\%6$	$r=0$	$r=1$	$r=2$
0	10	16	13
1	11	18	14
2	13	20	16
3	14	23	18
4	16	25	20
5	18	29	23

Table B.2: Multiplication factors V_{ij}.

Before applying the inverse integer transform, the input values $\mathbf{Y}'$ need to be normalized with the post-scaling matrix $\mathbf{E}_i$:

$$\mathbf{W}' = \mathbf{Y}' \otimes \mathbf{E}_i, \tag{B.14}$$

where the post-scaling matrix $\mathbf{E}_i$ is given by

$$\mathbf{E}_i = \begin{pmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{pmatrix}. \tag{B.15}$$

Appendix C

Single quantization of Laplace source

C.1 Probability distribution of a quantized Laplace source

The transform coefficient distribution is most commonly approximated by a Laplace pdf with parameter λ:

$$p_l(x) = \frac{\lambda}{2} e^{-\lambda|x|}, x \in \mathbb{R}. \tag{C.1}$$

Assume that the transform coefficients are quantized with a uniform quantizer with quantizer step size Q_1 and quantizer offset ϵ_1. Let $P_1(iQ_1)$ be the probability that a transform coefficient is quantized to the value iQ_1, where $i \in \mathbb{Z}$:

$$P_1(iQ_1) = \begin{cases} \int\limits_{(i-1+\epsilon_1)Q_1}^{(i+\epsilon_1)Q_1} p_l(x)dx = \left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) e^{\lambda i Q_1}, & \text{if } i < 0; \\ \int\limits_{(-1+\epsilon_1)Q_1}^{(1-\epsilon_1)Q_1} p_l(x)dx = 1 - e^{-\lambda(1-\epsilon_1)Q_1}, & \text{if } i = 0; \\ \int\limits_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} p_l(x)dx = \left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) e^{-\lambda i Q_1}, & \text{if } i > 0. \end{cases} \tag{C.2}$$

C.2 Entropy of a quantized Laplace source

The entropy of the quantized transform coefficients is defined as:

$$H_1 = -\sum_{i=-\infty}^{+\infty} P_1(iQ_1)\log_2 P_1(iQ_1). \tag{C.3}$$

Using the symmetry of the transform coefficient distribution and the quantizer characteristic, the formula for entropy can be split into an entropy component for the zero transform coefficients $H_{1,0}$ and entropy components for the non-zero transform coefficients $H_{1,i}$:

$$H_1 = H_{1,0} + 2\sum_{i=1}^{+\infty} H_{1,i}. \tag{C.4}$$

The entropy of the zero transform coefficients is defined as

$$\begin{aligned} H_{1,0} &= -P_1(0)\log_2 P_1(0) \\ &= -\left(1 - e^{-\lambda(1-\epsilon_1)Q_1}\right)\log_2\left(1 - e^{-\lambda(1-\epsilon_1)Q_1}\right). \end{aligned} \tag{C.5}$$

The entropy of the non-zero transform coefficients is defined as:

$$\begin{aligned} \sum_{i=1}^{+\infty} H_{1,i} &= -\sum_{i=1}^{+\infty} P_1(iQ_1)\log_2 P_1(iQ_1) \\ &= -\sum_{i=1}^{+\infty}\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)e^{-\lambda iQ_1} \\ &\quad\times\log_2\left(\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)e^{-\lambda iQ_1}\right). \end{aligned} \tag{C.6}$$

Note that $\log_2(a\cdot b) = \log_2(a) + \log_2(b)$. This results in the following expression:

$$\begin{aligned} \sum_{i=1}^{+\infty} H_{1,i} &= -\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{i=1}^{+\infty} e^{-\lambda iQ_1} \\ &\quad\times\left[\log_2\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) + \log_2\left(e^{-\lambda iQ_1}\right)\right]. \end{aligned} \tag{C.7}$$

Note that $\log_a(x) = \log_a(b) \cdot \log_b(x)$ and $\log_a(b) = \frac{1}{\log_b(a)}$. This results in the following expression:

$$\sum_{i=1}^{+\infty} H_{1,i} = -\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{i=1}^{+\infty} e^{-\lambda i Q_1} \times \left[\log_2\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) - \frac{\lambda i Q_1}{\ln 2}\right]. \tag{C.8}$$

Note that

$$\sum_{i=1}^{+\infty} a^i = \frac{a}{1-a} \text{ and } \sum_{i=1}^{+\infty} i a^i = \frac{a}{(1-a)^2}, \tag{C.9}$$

where $|a| < 1$.
This results in the following expression:

$$\sum_{i=1}^{+\infty} H_{1,i} = -\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\frac{e^{-\lambda Q_1}}{1-e^{-\lambda Q_1}} \times \left[\log_2\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) - \frac{\lambda Q_1}{(1-e^{-\lambda Q_1})\ln 2}\right]. \tag{C.10}$$

Finally, we end up with an expression for the entropy:

$$\begin{aligned} H_1 = &-\left(1-e^{-\lambda(1-\epsilon_1)Q_1}\right)\log_2\left(1-e^{-\lambda(1-\epsilon_1)Q_1}\right) \\ &-\left(e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}\right)\frac{e^{-\lambda Q_1}}{1-e^{-\lambda Q_1}} \\ &\times\left[\log_2\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) - \frac{\lambda Q_1}{(1-e^{-\lambda Q_1})\ln 2}\right]. \end{aligned} \tag{C.11}$$

C.3 Distortion of a quantized Laplace source

Using the squared-error criterion $(x-\hat{x})^2$, the distortion of the quantized transform coefficients is defined as:

$$D_1 = \sum_{i=-\infty}^{+\infty}\int_{L_i}^{U_i} p_l(x)\,(x - iQ_1)^2\,dx, \tag{C.12}$$

where L_i and U_i are the upper and lower bounds of the quantizer bins. Using the symmetry of the transform coefficient distribution and the quantizer

characteristic, the formula for distortion can be split into a distortion component for the zero transform coefficients $D_{1,0}$ and distortion components for the non-zero transform coefficients $D_{1,i}$:

$$D_1 = D_{1,0} + 2\sum_{i=1}^{+\infty} D_{1,i}. \tag{C.13}$$

In the remainder, the following integrals are used:

$$\int_s^t \lambda e^{-\lambda x} dx = e^{-\lambda s} - e^{-\lambda t}, \tag{C.14}$$

$$\int_s^t \lambda e^{-\lambda x} x dx = e^{-\lambda s}\left(s + \frac{1}{\lambda}\right) - e^{-\lambda t}\left(t + \frac{1}{\lambda}\right), \tag{C.15}$$

and

$$\int_s^t \lambda e^{-\lambda x} x^2 dx = e^{-\lambda s}\left(s^2 + \frac{2s}{\lambda} + \frac{2}{\lambda^2}\right) - e^{-\lambda t}\left(t^2 + \frac{2t}{\lambda} + \frac{2}{\lambda^2}\right). \tag{C.16}$$

The distortion of the zero transform coefficients is defined as

$$\begin{aligned} D_{1,0} &= \int_{(-1+\epsilon_1)Q_1}^{(1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda|x|} x^2 dx = 2\int_0^{(1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} x^2 dx \\ &= \frac{2}{\lambda^2} - e^{-\lambda(1-\epsilon_1)Q_1}\left((1-\epsilon_1)^2 Q_1^2 + \frac{2(1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2}\right). \end{aligned} \tag{C.17}$$

The distortion of the non-zero transform coefficients is defined as

$$\begin{aligned} \sum_{i=1}^{+\infty} D_{1,i} =& \sum_{i=1}^{+\infty} \int_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} (x - iQ_1)^2 dx \\ =& \sum_{i=1}^{+\infty} \left[\frac{1}{2}\int_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} \lambda e^{-\lambda x} x^2 dx \right. \\ & - iQ_1 \int_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} \lambda e^{-\lambda x} x dx \\ & \left. + \frac{i^2 Q_1^2}{2} \int_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} \lambda e^{-\lambda x} dx \right]. \end{aligned} \tag{C.18}$$

Using the standard integrals, we get the following expression:

$$\begin{aligned}\sum_{i=1}^{+\infty} D_{1,i} = \sum_{i=1}^{+\infty} \Bigg[& \frac{1}{2} \Bigg(e^{-\lambda(i-\epsilon_1)Q_1} \left[(i-\epsilon_1)^2 Q_1^2 + \frac{2(i-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2} \right] \\ & - e^{-\lambda(i+1-\epsilon_1)Q_1} \left[(i+1-\epsilon_1)^2 Q_1^2 + \frac{2(i+1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2} \right] \Bigg) \\ & - iQ_1 \Bigg(e^{-\lambda(i-\epsilon_1)Q_1} \left[(i-\epsilon_1)Q_1 + \frac{1}{\lambda} \right] \\ & - e^{-\lambda(i+1-\epsilon_1)Q_1} \left[(i+1-\epsilon_1)Q_1 + \frac{1}{\lambda} \right] \Bigg) \\ & + \frac{i^2 Q_1^2}{2} \left(e^{-\lambda(i-\epsilon_1)Q_1} - e^{-\lambda(i+1-\epsilon_1)Q_1} \right) \Bigg]. \end{aligned} \tag{C.19}$$

After rearrangement, we get the following expression:

$$\begin{aligned}\sum_{i=1}^{+\infty} D_{1,i} = \sum_{i=1}^{+\infty} \Bigg[& e^{-\lambda(i-\epsilon_1)Q_1} \left(\frac{(i-\epsilon_1)^2 Q_1^2}{2} + \frac{(i-\epsilon_1)Q_1}{\lambda} + \frac{1}{\lambda^2} \right. \\ & \left. - i(i-\epsilon_1)Q_1^2 - \frac{iQ_1}{\lambda} + \frac{i^2 Q_1^2}{2} \right) \\ & - e^{-\lambda(i+1-\epsilon_1)Q_1} \left(\frac{(i+1-\epsilon_1)^2 Q_1^2}{2} + \frac{(i+1-\epsilon_1)Q_1}{\lambda} + \frac{1}{\lambda^2} \right. \\ & \left. - i(i+1-\epsilon_1)Q_1^2 - \frac{iQ_1}{\lambda} + \frac{i^2 Q_1^2}{2} \right) \Bigg]. \end{aligned} \tag{C.20}$$

After simplification, we get the following expression:

$$\begin{aligned}\sum_{i=1}^{+\infty} D_{1,i} = \sum_{i=1}^{+\infty} e^{-\lambda i Q 1} \times \Bigg[& e^{\lambda\epsilon_1 Q_1} \left(\frac{\epsilon_1^2 Q_1^2}{2} - \frac{\epsilon_1 Q_1}{\lambda} + \frac{1}{\lambda^2} \right) \\ & - e^{-\lambda(1-\epsilon_1)Q_1} \left(\frac{(1-\epsilon_1)^2 Q_1^2}{2} + \frac{(1-\epsilon_1)Q_1}{\lambda} + \frac{1}{\lambda^2} \right) \Bigg]. \end{aligned} \tag{C.21}$$

Note that

$$\sum_{i=1}^{+\infty} a^i = \frac{a}{1-a}, \tag{C.22}$$

where $|a| < 1$.
This results in the following expression:

$$\sum_{i=1}^{+\infty} D_{1,i} = \frac{e^{-\lambda Q_1}}{1-e^{-\lambda Q_1}} \times \left[e^{\lambda \epsilon_1 Q_1} \left(\frac{\epsilon_1^2 Q_1^2}{2} - \frac{\epsilon_1 Q_1}{\lambda} + \frac{1}{\lambda^2} \right) - e^{-\lambda(1-\epsilon_1)Q_1} \left(\frac{(1-\epsilon_1)^2 Q_1^2}{2} + \frac{(1-\epsilon_1)Q_1}{\lambda} + \frac{1}{\lambda^2} \right) \right]. \quad \text{(C.23)}$$

Finally, we end up with an expression for the distortion:

$$D_1 = \frac{2}{\lambda^2} - e^{-\lambda(1-\epsilon_1)Q_1} \left((1-\epsilon_1)^2 Q_1^2 + \frac{2(1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2} \right) + \frac{2e^{-\lambda Q_1}}{1-e^{-\lambda Q_1}} \times \left[e^{\lambda \epsilon_1 Q_1} \left(\frac{\epsilon_1^2 Q_1^2}{2} - \frac{\epsilon_1 Q_1}{\lambda} + \frac{1}{\lambda^2} \right) - e^{-\lambda(1-\epsilon_1)Q_1} \left(\frac{(1-\epsilon_1)^2 Q_1^2}{2} + \frac{(1-\epsilon_1)Q_1}{\lambda} + \frac{1}{\lambda^2} \right) \right]. \quad \text{(C.24)}$$

Appendix D

Entropy calculation for double quantization

When a signal with Laplace source is quantized twice, we get the following expression for the entropy:

$$H_2 = -\sum_{i=-\infty}^{+\infty} P_2(iQ_2)\log_2 P_2(iQ_2), \tag{D.1}$$

with

$$P_2(iQ_2) = \sum_{l=\sigma_i}^{\tau_i} P_1(lQ_1) \tag{D.2}$$

where the values σ_i and τ_i depend on the quantizer bin i. Using the symmetry of the transform coefficient distribution and the first-step and second-step quantizer characteristics, the formula for entropy can be split into an entropy component for the zero transform coefficients after second-step quantization $H_{2,0}$ and entropy components for the non-zero transform coefficients after second-step quantization $H_{2,i}$:

$$H_2 = H_{2,0} + 2\sum_{i=1}^{+\infty} H_{2,i}. \tag{D.3}$$

The entropy component $H_{2,0}$ is determined by the probability of the transform coefficients which fall in the dead-zone after second-step quantization:

$$H_{2,0} = -P_2(0)\log_2 P_2(0), \tag{D.4}$$

with

$$P_2(0) = P_1(0) + 2\sum_{l=1}^{\eta} P_1(lQ_1). \tag{D.5}$$

The value η is determined by the number of reconstruction levels of the first-step quantizer which fall in the dead-zone of the second-step quantizer. This can be expressed using the following Diophantine inequality: $\eta Q_1 < (1 - \epsilon_2)Q_2 < (\eta + 1)Q_1$. The entropy component $H_{2,0}$ is therefore:

$$\begin{aligned} H_{2,0} =& - P_2(0) \log_2 P_2(0) \\ =& - \left(\left(1 - e^{-\lambda(1-\epsilon_1)Q_1}\right) + \left(e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}\right) \sum_{l=1}^{\eta} e^{-\lambda l Q_1} \right) \\ & \log_2 \left(\left(1 - e^{-\lambda(1-\epsilon_1)Q_1}\right) + \left(e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}\right) \sum_{l=1}^{\eta} e^{-\lambda l Q_1} \right). \end{aligned} \tag{D.6}$$

The entropy components $H_{2,i}$ are determined by the probability of the non-zero transform coefficients after second-step quantization:

$$\sum_{i=1}^{+\infty} H_{2,i} = - \sum_{i=1}^{+\infty} P_2(iQ_2) \log_2 P_2(iQ_2), \tag{D.7}$$

with

$$P_2(iQ_2) = \frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2} \sum_{l=0}^{\mu} e^{-\lambda(m+l)Q_1} \tag{D.8}$$

where the values m and μ are defined by the characteristics of the second-step quantizer.
Due to the memoryless property of the Laplace model and the periodic property of the effective quantizer, we apply the substitutions $i = j + k\alpha$ for $j \in \{0, \ldots, \alpha - 1\}$ and $m = n + k\beta$ for $n \in \{0, \ldots, \beta - 1\}$:

$$\begin{aligned} \sum_{i=1}^{+\infty} H_{2,i} &= \sum_{j=0}^{\alpha-1} H'_{2,j} \\ &= - \sum_{j=0}^{\alpha-1} \sum_{k=\delta(j)}^{+\infty} P_2((j + k\alpha)Q_2) \log_2 P_2((j + k\alpha)Q_2) \end{aligned} \tag{D.9}$$

with

$$P_2((j + k\alpha)Q_2) = \frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2} \sum_{l=0}^{\mu} e^{-\lambda((n+k\beta)+l)Q_1}. \tag{D.10}$$

This way we exploit the periodicity in units of the second-step quantizer. We derive a closed-form expression for $H'_{2,j}$:

$$\begin{aligned} H'_{2,j} &= -\sum_{k=\delta(j)}^{+\infty} P_2((j+k\alpha)Q_2)\log_2 P_2((j+k\alpha)Q_2) \\ &= -\sum_{k=\delta(j)}^{+\infty}\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+k\beta+l)Q_1} \\ &\quad\times\log_2\left(\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+k\beta+l)Q_1}\right). \end{aligned} \tag{D.11}$$

Note that $\log_2(a\cdot b)=\log_2(a)+\log_2(b)$. This results in the following expression:

$$\begin{aligned} H'_{2,j} &= -\sum_{k=\delta(j)}^{+\infty} P_2((j+k\alpha)Q_2)\log_2 P_2((j+k\alpha)Q_2) \\ &= -\sum_{k=\delta(j)}^{+\infty}\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+k\beta+l)Q_1} \\ &\quad\times\left[\log_2\left(\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\right)\right. \\ &\quad\left.+\log_2\left(e^{-\lambda k\beta Q_1}\right)\right]. \end{aligned} \tag{D.12}$$

Note that $\log_a(x)=\log_a(b)\cdot\log_b(x)$ and $\log_a(b)=\frac{1}{\log_b(a)}$. This results in the following expression:

$$\begin{aligned} H'_{2,j} &= -\sum_{k=\delta(j)}^{+\infty} P_2((j+k\alpha)Q_2)\log_2 P_2((j+k\alpha)Q_2) \\ &= -\sum_{k=\delta(j)}^{+\infty}\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+k\beta+l)Q_1} \\ &\quad\times\left[\log_2\left(\left(\frac{e^{\lambda\epsilon_1 Q_1}-e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right)\sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\right)\right. \\ &\quad\left.-\frac{\lambda k\beta Q_1}{\ln 2}\right]. \end{aligned} \tag{D.13}$$

Note that

$$\sum_{k=0}^{+\infty} a^k = \frac{1}{1-a}, \sum_{k=1}^{+\infty} a^k = \frac{a}{1-a}, \text{and } \sum_{k=0}^{+\infty} k a^k = \frac{a}{(1-a)^2}, \tag{D.14}$$

where $|a| < 1$.
Finally, we get a closed-form expression for $j = 0$:

$$\begin{aligned} H'_{2,0} &= -\sum_{k=1}^{+\infty} P_2(k\alpha Q_2) \log_2 P_2(k\alpha Q_2) \\ &= -\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) \sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1} \frac{e^{-\lambda\beta Q_1}}{1-e^{-\lambda\beta Q_1}} \\ &\quad \times \left[\log_2 \left(\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) \sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\right) \right. \\ &\quad \left. - \frac{\lambda\beta Q_1}{(1-e^{-\lambda\beta Q_1})\ln 2}\right], \end{aligned} \tag{D.15}$$

and a closed-form expression for $j \neq 0$:

$$\begin{aligned} H'_{2,j} &= -\sum_{k=0}^{+\infty} P_2((j+k\alpha)Q_2) \log_2 P_2((j+k\alpha)Q_2) \\ &= -\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) \sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1} \frac{1}{1-e^{-\lambda\beta Q_1}} \\ &\quad \times \left[\log_2 \left(\left(\frac{e^{\lambda\epsilon_1 Q_1} - e^{-\lambda(1-\epsilon_1)Q_1}}{2}\right) \sum_{l=0}^{\mu} e^{-\lambda(n+l)Q_1}\right) \right. \\ &\quad \left. - \frac{\lambda\beta Q_1 e^{-\lambda\beta Q_1}}{(1-e^{-\lambda\beta Q_1})\ln 2}\right]. \end{aligned} \tag{D.16}$$

Appendix E

Distortion calculation for double quantization

When a signal with Laplace source is quantized twice, we get the following expression for distortion:

$$D_2 = \sum_{i=-\infty}^{+\infty} \int_{L_i}^{U_i} p_l(x) \left(x - mQ_2\right)^2 dx, \tag{E.1}$$

where L_i and U_i are the upper and lower bounds of the quantizer bins of the first-step quantizer. Using the symmetry of the transform coefficient distribution and the first-step and second-step quantizer characteristics, the formula for distortion can be split into a distortion component for the zero transform coefficients after first-step quantization $D_{2,0}$ and distortion components for the non-zero transform coefficients after first-step quantization $D_{2,i}$:

$$D_2 = D_{2,0} + 2\sum_{i=1}^{+\infty} D_{2,i}. \tag{E.2}$$

In the remainder, the following integrals are used:

$$\int_s^t \lambda e^{-\lambda x} dx = e^{-\lambda s} - e^{-\lambda t}, \tag{E.3}$$

$$\int_s^t \lambda e^{-\lambda x} x dx = e^{-\lambda s}\left(s + \frac{1}{\lambda}\right) - e^{-\lambda t}\left(t + \frac{1}{\lambda}\right), \tag{E.4}$$

and

$$\int_s^t \lambda e^{-\lambda x} x^2 dx = e^{-\lambda s}\left(s^2 + \frac{2s}{\lambda} + \frac{2}{\lambda^2}\right) - e^{-\lambda t}\left(t^2 + \frac{2t}{\lambda} + \frac{2}{\lambda^2}\right). \tag{E.5}$$

The distortion component $D_{2,0}$ is determined by the zero transform coefficients after first-step quantization:

$$\begin{aligned} D_{2,0} &= \int_{(-1+\epsilon_1)Q_1}^{(1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda|x|} x^2 dx = 2\int_{0}^{(1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} x^2 dx \\ &= \frac{2}{\lambda^2} - e^{-\lambda(1-\epsilon_1)Q_1}\left((1-\epsilon_1)^2 Q_1^2 + \frac{2(1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2}\right). \end{aligned} \tag{E.6}$$

The distortion components $D_{2,i}$ are determined by the non-zero transform coefficients of the first-step quantizer:

$$\sum_{i=1}^{+\infty} D_{2,i} = \sum_{i=1}^{+\infty} \int_{(i-\epsilon_1)Q_1}^{(i+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - mQ_2\right)^2 dx. \tag{E.7}$$

Due to the memoryless property of the Laplace source and the periodic property of the effective quantizer, we apply the substitutions $i = j + k\beta$ for $j \in \{0, \ldots, \beta-1\}$ and $m = n + k\alpha$ for $n \in \{0, \ldots, \alpha-1\}$:

$$\begin{aligned} \sum_{i=1}^{+\infty} D_{2,i} &= \sum_{j=0}^{\beta-1} D'_{2,j} \\ &= \sum_{j=0}^{\beta-1} \sum_{k=\delta(j)}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - (n+k\alpha)Q_2\right)^2 dx. \end{aligned} \tag{E.8}$$

This way we exploit the periodicity in units of the first-step quantizer. We derive a closed-form expression for $D'_{2,j}$:

$$\begin{aligned} D'_{2,j} &= \sum_{k=\delta(j)}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - (n+k\alpha)Q_2\right)^2 dx \\ &= \sum_{k=\delta(j)}^{+\infty} \left[\frac{1}{2}\int_{(j+k\beta-\epsilon_1)Q_1}^{(n+k\alpha)Q_2} \lambda e^{-\lambda x} x^2 dx \right. \\ &\quad - (n+k\alpha)\, Q_2 \int_{(j+k\beta-\epsilon_1)Q_1}^{(n+k\alpha)Q_2} \lambda e^{-\lambda x} x dx \\ &\quad \left. + \frac{(n+k\alpha)^2 Q_2^2}{2} \int_{(j+k\beta-\epsilon_1)Q_1}^{(n+k\alpha)Q_2} \lambda e^{-\lambda x} dx \right]. \end{aligned} \tag{E.9}$$

Using the standard integrals, we get the following expression:

$$
\begin{aligned}
D'_{2,j} &= \sum_{k=\delta(j)}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x-(n+k\alpha)Q_2\right)^2 dx \\
&= \sum_{k=\delta(j)}^{+\infty} \Bigg[\frac{1}{2} \Big(e^{-\lambda(j+k\beta-\epsilon_1)Q_1} \left[(j+k\beta-\epsilon_1)^2 Q_1^2 \right. \\
&\quad \left. + \frac{2(j+k\beta-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2} \right] \\
&\quad - e^{-\lambda(j+k\beta+1-\epsilon_1)Q_1} \left[(j+k\beta+1-\epsilon_1)^2 Q_1^2 \right. \\
&\quad \left. + \frac{2(j+k\beta+1-\epsilon_1)Q_1}{\lambda} + \frac{2}{\lambda^2} \right] \Big) \\
&\quad - (n+k\alpha) Q_2 \left(e^{-\lambda(j+k\beta-\epsilon_1)Q_1} \left[(j+k\beta-\epsilon_1)Q_1 + \frac{1}{\lambda} \right] \right. \\
&\quad \left. - e^{-\lambda(j+k\beta+1-\epsilon_1)Q_1} \left[(j+k\beta+1-\epsilon_1)Q_1 + \frac{1}{\lambda} \right] \right) \\
&\quad + \frac{(n+k\alpha)^2 Q_2^2}{2} \left(e^{-\lambda(j+k\beta-\epsilon_1)Q_1} - e^{-\lambda(j+k\beta+1-\epsilon_1)Q_1} \right) \Bigg] . \qquad \text{(E.10)}
\end{aligned}
$$

After rearrangement, we get the following expression:

$$
\begin{aligned}
D'_{2,j} &= \sum_{k=\delta(j)}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x-(n+k\alpha)Q_2\right)^2 dx \\
&= \sum_{k=\delta(j)}^{+\infty} \Bigg[e^{-\lambda(j+k\beta-\epsilon_1)Q_1} \left(\frac{(j+k\beta-\epsilon_1)^2 Q_1^2}{2} \right. \\
&\quad + \frac{(j+k\beta-\epsilon_1) Q_1}{\lambda} + \frac{1}{\lambda^2} - (n+k\alpha) Q_2 (j+k\beta-\epsilon_1) Q_1 \\
&\quad \left. - \frac{(n+k\alpha) Q_2}{\lambda} + \frac{(n+k\alpha)^2 Q_2^2}{2} \right) \\
&\quad - e^{-\lambda(j+k\beta+1-\epsilon_1)Q_1} \left(\frac{(j+k\beta+1-\epsilon_1)^2 Q_1^2}{2} \right. \\
&\quad + \frac{(j+k\beta+1-\epsilon_1) Q_1}{\lambda} + \frac{1}{\lambda^2} - (n+k\alpha) Q_2 (j+k\beta+1-\epsilon_1) Q_1 \\
&\quad \left. - \frac{(n+k\alpha) Q_2}{\lambda} + \frac{(n+k\alpha)^2 Q_2^2}{2} \right) \Bigg] . \qquad \text{(E.11)}
\end{aligned}
$$

After simplification, we get the following expression:

$$\begin{aligned}
D'_{2,j} &= \sum_{k=\delta(j)}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x-(n+k\alpha)Q_2\right)^2 dx \\
&= \sum_{k=\delta(j)}^{+\infty} e^{-\lambda k\beta Q_1} \left[e^{-\lambda(j-\epsilon_1)Q_1} \left(\frac{\left((j-\epsilon_1)Q_1 - nQ_2\right)^2}{2} \right.\right. \\
&\quad \left. + \frac{\left((j-\epsilon_1)Q_1 - nQ_2\right)}{\lambda} + \frac{1}{\lambda^2} \right) \\
&\quad - e^{-\lambda(j+1-\epsilon_1)Q_1} \left(\frac{\left((j+1-\epsilon_1)Q_1 - nQ_2\right)^2}{2} \right. \\
&\quad \left.\left. + \frac{\left((j+1-\epsilon_1)Q_1 - nQ_2\right)}{\lambda} + \frac{1}{\lambda^2} \right) \right].
\end{aligned} \tag{E.12}$$

Note that

$$\sum_{k=0}^{+\infty} a^k = \frac{1}{1-a} \text{ and } \sum_{k=1}^{+\infty} a^k = \frac{a}{1-a}, \tag{E.13}$$

where $|a| < 1$.
Finally, we get a closed-form expression for $j = 0$:

$$\begin{aligned}
D'_{2,0} =& \sum_{k=1}^{+\infty} \int_{(k\beta-\epsilon_1)Q_1}^{(k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x-(n+k\alpha)Q_2\right)^2 dx \\
=& \frac{e^{-\lambda\beta Q_1}}{1-e^{-\lambda\beta Q_1}} \left[e^{\lambda\epsilon_1 Q_1} \left(\frac{\left(-\epsilon_1 Q_1 - nQ_2\right)^2}{2} \right.\right. \\
&\left. + \frac{\left(-\epsilon_1 Q_1 - nQ_2\right)}{\lambda} + \frac{1}{\lambda^2} \right) \\
&- e^{-\lambda(1-\epsilon_1)Q_1} \left(\frac{\left((1-\epsilon_1)Q_1 - nQ_2\right)^2}{2} \right. \\
&\left.\left. + \frac{\left((1-\epsilon_1)Q_1 - nQ_2\right)}{\lambda} + \frac{1}{\lambda^2} \right) \right],
\end{aligned} \tag{E.14}$$

and a closed-form expression for $j \neq 0$:

$$
\begin{aligned}
D'_{2,j} =& \sum_{k=0}^{+\infty} \int_{(j+k\beta-\epsilon_1)Q_1}^{(j+k\beta+1-\epsilon_1)Q_1} \frac{\lambda}{2} e^{-\lambda x} \left(x - (n+k\alpha)Q_2\right)^2 dx \\
=& \frac{1}{1-e^{-\lambda\beta Q_1}} \left[e^{-\lambda(j-\epsilon_1)Q_1} \left(\frac{((j-\epsilon_1)Q_1 - nQ_2)^2}{2} \right.\right. \\
& \left. + \frac{((j-\epsilon_1)Q_1 - nQ_2)}{\lambda} + \frac{1}{\lambda^2} \right) \\
& - e^{-\lambda(j+1-\epsilon_1)Q_1} \left(\frac{((j+1-\epsilon_1)Q_1 - nQ_2)^2}{2} \right. \\
& \left.\left. + \frac{((j+1-\epsilon_1)Q_1 - nQ_2)}{\lambda} + \frac{1}{\lambda^2} \right) \right].
\end{aligned} \tag{E.15}
$$

References

[1] D. Wu, Y. T. Hou, W. Zhu, Y.-Q. Zhang, and J. M. Peha. Streaming video over the Internet: approaches and directions. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(3):282–300, March 2001.

[2] Y. Wu, S. Hirakawa, U. H. Reimers, and J. Whitaker. Overview of digital television development worldwide. *Proceedings of the IEEE*, 94(1):8–21, January 2006.

[3] T. Sikora. Trends and perspectives in image and video coding. *Proceedings of the IEEE*, 93(1):6–17, January 2005.

[4] ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG-2 Video), ITU-T and ISO/IEC JTC 1. *Information technology - Generic coding of moving pictures and associated audio information: Video*, February 2002.

[5] ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), ITU-T and ISO/IEC JTC 1. *Advanced Video Coding for Generic Audiovisual Services*, Version 1: May 2003, Version 2: May 2004, Version 3: Mar. 2005, Version 4: Sept. 2005, Version 5 and Version 6: June 2006, Version 7: Apr. 2007, Version 8 (including SVC extension): November 2007.

[6] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, July 2003.

[7] D. Marpe, T. Wiegand, and S. Gordon. H.264/MPEG4-AVC fidelity range extensions: tools, profiles, performance, and application areas. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 593–596, 2005.

[8] D. Marpe, T. Wiegand, and G. J. Sullivan. The H.264/MPEG4 advanced video coding standard and its applications. *IEEE Communications Magazine*, 44(8):134–143, August 2006.

[9] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, September 2007.

[10] J. Xin, C.-W. Lin, and M.-T. Sun. Digital video transcoding. *Proceedings of the IEEE*, 93(1):84–97, January 2005.

[11] G. Keesman, R. Hellinghuizen, F. Hoeksema, and G. Heideman. Transcoding of MPEG bitstreams. *Signal Processing: Image Communication*, 8(6):481–500, September 1996.

[12] P. A. A. Assunção 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.

[13] J. Youn, M.-T. Sun, and C.-W. Lin. Motion vector refinement for high-performance transcoding. *IEEE Transactions on Multimedia*, 1(1):30–40, March 1999.

[14] M.-J. Chen, M.-C. Chu, and C.-W. Pan. Efficient motion-estimation algorithm for reduced frame-rate video transcoder. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(4):269–275, April 2002.

[15] K.-T. Fung, Y.-L. Chan, and W.-C. Siu. New architecture for dynamic frame-skipping transcoder. *IEEE Transactions on Image Processing*, 11(8):886–900, August 2002.

[16] B. Shen, I. K. Sethi, and B. Vasudev. Adaptive motion-vector resampling for compressed video downscaling. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(6):929–936, September 1999.

[17] P. Yin, A. Vetro, B. Liu, and H. Sun. Drift compensation for reduced spatial resolution transcoding. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(11):1009–1020, November 2002.

[18] V. Patil, R. Kumar, and J. Mukherjee. A fast arbitrary factor video resizing algorithm. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(9):1164–1171, September 2006.

[19] T. Qian, J. Sun, D. Li, X. Yang, and J. Wang. 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.

[20] Q. Tang, P. Nasiopoulos, and R. K. Ward. Compensation of requantization and interpolation errors in MPEG-2 to H.264 transcoding. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(3):314–325, March 2008.

[21] D. Xu and P. Nasiopoulos. Logo insertion transcoding for H.264/AVC compressed video. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 3693–3696, 2009.

[22] A. Vetro, C. Christopoulos, and H. Sun. Video transcoding architectures and techniques: an overview. *IEEE Signal Processing Magazine*, 20(2):18–29, March 2003.

[23] A. Eleftheriadis and D. Anastassiou. Constrained and general dynamic rate shaping of compressed digital video. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 396–399, 1995.

[24] A. Eleftheriadis and P. Batra. Dynamic rate shaping of compressed digital video. *IEEE Transactions on Multimedia*, 8(2):297–314, April 2006.

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

[26] M. M. Crouse and K. Ramchandran. Joint thresholding and quantizer selection for decoder-compatible baseline JPEG. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2331–2334, 1995.

[27] M. M. Crouse and K. Ramchandran. Joint thresholding and quantizer selection for transform image coding: entropy-constrained analysis and applications to baseline JPEG. *IEEE Transactions on Image Processing*, 6(2):285–297, February 1997.

[28] B. Shen. Perfect requantization for video transcoding. *Multimedia Tools and Applications*, 35(2):163–173, November 2007.

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

[30] K. Ramchandran and M. Vetterli. Syntax-constrained encoder optimization using adaptive quantization thresholding for JPEG/MPEG coders. In *Proceedings of the Data Compression Conference*, pages 146–155, 1994.

[31] A. Eleftheriadis and D. Anastassiou. Optimal data partitioning of MPEG-2 coded video. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 273–277, 1994.

[32] A. Eleftheriadis and P. Batra. Optimal data partitioning of MPEG-2 coded video. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(10):1195–1209, October 2004.

[33] W. Zeng, B. Guo, and B. Liu. Feature-oriented rate shaping of pre-compressed image/video. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 772–775, 1997.

[34] C.-C. Ho, J.-L. Wu, and W.-H. Cheng. A practical foveation-based rate-shaping mechanism for MPEG videos. *IEEE Transactions on Circuits Systems for Video Technology*, 15(11):1365–1372, November 2005.

[35] D. Le Gall. MPEG: a video compression standard for multimedia applications. *Communications of the ACM*, 34(4):46–58, April 1991.

[36] K. R. Rao and P. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic, 1990.

[37] M. Wien. Variable block-size transforms for H.264/AVC. *IEEE Transactions on Circuits Systems for Video Technology*, 13(7):604–613, July 2003.

[38] H. S. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky. Low-complexity transform and quantization in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):598–603, July 2003.

[39] D. Marpe, H. Schwarz, and T. Wiegand. Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):620–636, July 2003.

[40] J. De Cock, S. Notebaert, and R. Van de Walle. A novel hybrid requantization transcoding scheme for H.264/AVC. In *Proceedings of the IEEE International Symposium on Signal Processing and Its Applications (ISSPA)*, 2007.

[41] H. Everett III. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11(3):399–417, May-June 1963.

[42] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(9):1445–1453, September 1988.

[43] B. L. Fox and D. M. Landi. Searching for the multiplier in one-constraint optimization problems. *Operations Research*, 18(2):253–262, March-April 1970.

[44] K. Ramchandran, A. Ortega, and M. Vetterli. Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders. *IEEE Transactions on Image Processing*, 3(5):533–545, September 1994.

[45] K. L. Ferguson and N. M. Allison. Modified steepest-descent for bit-allocation in strongly-dependent video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7):1057–1062, July 2009.

[46] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan. Rate-constrained coder control and comparison of video coding standards. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):688–703, July 2003.

[47] T. K. Tan, G. J. Sullivan, and T. Wedi. *Recommended Simulation Common Conditions for Coding Efficiency Experiments Revision 1*. Video Coding Experts Group, Doc. VCEG-AE10, Marrakech, Morocco, January 2007.

[48] H. Schwarz, D. Marpe, and T. Wiegand. Analysis of hierarchical B pictures and MCTF. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1929–1932, 2006.

[49] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro. H.264/AVC baseline profile decoder complexity analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):704–716, July 2003.

[50] Z. He and S. K. Mitra. Optimal 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):840–849, October 2002.

[51] O. Werner. Requantization for transcoding of MPEG-2 intraframes. *IEEE Transactions on Image Processing*, 8(2):179–191, February 1999.

[52] H. H. Bauschke, C. H. Hamilton, M. S. Macklem, J. S. McMichael, and N. R. Swart. A requantization-based method for recompressing JPEG images. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2489–2492, 2002.

[53] H. H. Bauschke, C. H. Hamilton, M. S. Macklem, J. S. McMichael, and N. R. Swart. Recompression of JPEG images by requantization. *IEEE Transactions on Image Processing*, 12(7):843–849, July 2003.

[54] J. Bialkowski, M. Barkowsky, and A. Kaup. A new algorithm for reducing the requantization loss in video transcoding. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, 2006.

[55] O. Gendler and M. Porat. Toward optimal real-time transcoding using requantization in the DCT domain. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 3677–3680, 2009.

[56] S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982. (reprint of work originally presented in July 1957).

[57] J. Max. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, March 1960.

[58] G. J. Sullivan. Efficient scalar quantization of Exponential and Laplacian random variables. *IEEE Transactions on Information Theory*, 42(5):1365–1374, 1996.

[59] G. J. Sullivan. Optimal entropy constrained scalar quantization for Exponential and Laplacian random variables. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 265–268, 1994.

[60] R. C. Reininger and J. D. Gibson. Distributions of the two-dimensional DCT coefficients for images. *IEEE Transactions on Communications*, 31(6):835–839, June 1983.

[61] N. Farvardin and J. W. Modestino. Optimum quantizer performance for a class of non-Gaussian memoryless sources. *IEEE Transactions on Information Theory*, 30(3):485–496, May 1984.

[62] R. L. Joshi and T. R. Fischer. Comparison of generalized Gaussian and Laplacian modeling in DCT image coding. *IEEE Signal Processing Letters*, 2(5):81–82, May 1995.

[63] E. Y. Lam and J. W. Goodman. A mathematical analysis of the DCT coefficient distributions for images. *IEEE Transactions on Image Processing*, 9(10):1661–1666, October 2000.

[64] 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.

[65] N. M. Rajpoot. Simulation of the rate-distortion behavior of a memoryless Laplacian source. In *Proceedings of the Middle Eastern Symposium on Simulation and Modelling (MESM)*, 2002.

[66] B. Tao. On optimal entropy-constrained deadzone quantization. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(4):560–563, April 2001.

[67] D. Lefol, D. R. Bull, and C. N. Canagarajah. Mode refinement algorithm for H.264 intra frame requantization. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 4459–4462, 2006.

[68] D. Lefol and D. R. Bull. Mode refinement algorithm for H.264 inter frame requantization. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 845–848, 2006.

[69] N. Hait and D. Malah. Model-based transrating of H.264 coded video. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(8):1129–1142, August 2009.

[70] N. Hait and D. Malah. Model-based transrating of H.264 intra-coded frames. In *Proceedings of the Picture Coding Symposium (PCS)*, 2007.

[71] J. Lu, T. Chen, Y. Kashiwagi, and S. Kadono. *Proposal of quantization weighting for H.264/MPEG-4 AVC Professional Profiles*. Joint Video Team, Doc. JVT-K029, Munich, Germany, March 2004.

[72] A. M. Tourapis and J. Boyce. *Performance evaluation of the 8x8 transform vs. coefficient adaptive deadzone consideration*. Joint Video Team, Doc. JVT-K035, Munich, Germany, March 2004.

[73] A. M. Tourapis, K. Sühring, and G. J. Sullivan. *Proposed amended H.264/MPEG-4 AVC reference software manual*. Joint Video Team, Doc. JVT-N008, Hong Kong, China, January 2005.

[74] J. De Cock, S. Notebaert, and R. Van de Walle. Combined SNR and temporal scalability for H.264/AVC using requantization transcoding and hierarchical B pictures. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 448–451, 2007.

[75] R. L. de Queiroz. Processing JPEG-compressed images and documents. *IEEE Transactions on Image Processing*, 7(12):1661–1672, December 1998.

[76] J. R. Price and M. Rabbani. Biased reconstruction for JPEG decoding. *IEEE Signal Processing Letters*, 6(12):297–299, December 1999.

[77] D. G. Morrison, M. E. Nilsson, and M. Ghanbari. Reduction of the bit-rate of compressed video while in its coded form. In *Proceedings of the International Workshop on Packet Video*, 1994.

[78] P. A. A. Assunção and M. Ghanbari. Post-processing of MPEG2 coded video for transmission at lower bit rates. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1998–2001, 1996.

[79] J. De Cock. *Compressed-domain transcoding of H.264/AVC and SVC video streams*. PhD thesis, Ghent University, October 2009.

[80] A. M. Tourapis, F. Wu, and S. Li. Direct mode coding for bi-predictive pictures in the JVT standard. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 700–703, 2003.

[81] M. Flierl and B. Girod. Generalized B pictures and the draft H.264/AVC video-compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):587–597, July 2003.

[82] 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.

[83] G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, November 1998.

[84] T. Wiegand and B. Girod. Lagrange multiplier selection in hybrid video coder control. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 542–545, 2001.

[85] P. List, A. Joch, J. Lainema, G. Bjøntegaard, and M. Karczewicz. Adaptive deblocking filter. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):614–619, July 2003.

[86] G. Shen, Y. He, W. Cao, and S. Li. MPEG-2 to WMV transcoder with adaptive error compensation and dynamic switches. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(11):1460–1476, November 2006.