Verbeterde compressieperformantie voor gedistribueerde videocodering

Improved Compression Performance for Distributed Video Coding

Jürgen Slowack

Promotoren: prof. dr. ir. R. Van de Walle, prof. dr. ir. A. Munteanu
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 2010 - 2011

UNIVERSITEIT
GENT

ISBN 978-90-8578-389-3
NUR 965, 980
Wettelijk depot: D/2010/10.500/65

Acknowledgements

The start of this book ends a chapter in my life. While I have tried to postpone it as much as possible, in a few weeks I will no longer be officially classified as a student. However, despite the acknowledgment that the youngster from before has become an adult for quite a while now, each change in live brings new directions to explore, new challenges to undertake, and new people to meet. The past few years have been both interesting and joyful, on a personal level as well as on a professional level. As such, the start of this book lends me the opportunity to thank the people that contributed.

First of all, I would like to thank prof. Rik Van de Walle, who gave me the opportunity to pursue a PhD at Multimedia Lab. His guidance and directions were very helpful during the four years that I performed this research. He also gave me the opportunity to perform part of this research abroad, in collaboration with the University of Central Lancashire (UCLAN), in Preston, in the United Kingdom.

The roughly four months that I have stayed there has left me with nothing but pleasant memories. It is also in this context that I would like to thank professor Christos Grecos for his help and hospitality. The many discussions and brain storming sessions we had surely provided me with better insights in the problems that were faced, and the solutions they required. The atmosphere during these meetings was always very open and productive, and discussions often continued in a less formal setting after-hours, even leading to suggestions for new journal papers and book chapters.

The collaboration with professor Adrian Munteanu and Nikos Deligiannis from the Department of Electronics and Informatics (ETRO) at the Vrije Universiteit Brussel (VUB) was also a very learning experience. As experts in the field, Adrian's and Nikos' comments and reviews have surely helped improving the quality of my work, stressing the importance of theory in addition to developing a working system that improves over previous work.

Special thanks goes to Stefaan and Jozef, who started working in Distributed Video Coding at the same time as me, also pursuing their PhD's.

During the first year, the three of us have designed the codec and written the software from scratch. For the remainder of our research work, each of us has specialized in different areas, taking the system to a higher level. I am convinced that the work I present here in this dissertation would not be the same without their input, comments, and support. Also on a personal level, many moments have found their place in a happy corner in my memory. As I am the first to defend my PhD, I wish them all the best for their defense in the near future.

The atmosphere at Multimedia Lab has always been open and friendly, and I would like to thank my colleagues for any suggestions and remarks that have helped to improve the quality of my articles, and finally, of this book.

I also want to thank my parents for their continuous support during the years. They have helped me reach out to things I would not have imagined possible.

Special thanks to Gwenny, who has conquered a special place in my heart, and supported me throughout. Together we became mum and dad of a little baby girl called Lenka, who will be playing the key role in the next chapter of my personal life.

Summary

The amount of digital video information is ever growing. People are watching video at home, on the internet, and even on the train using mobile devices or portable DVD players. Storing, processing, or transmitting these video sequences is not at all a trivial task, as the amount of video data is large. On the one hand, there is a demand for higher resolutions and better user experience. On the other hand there is an ongoing miniaturization, in which devices capable for performing video processing operations are getting smaller and smaller.

Due to practical limits encountered when handling these large data amounts, research on video compression systems and standards has always been important. A typical video compression (or coding) system consists of an encoder that converts the video sequence to a compact representation useful for transmission or storage, while the decoder performs the opposite operation.

Most of the current techniques achieve compression by exploiting correlation in the frame sequence at the encoder. This correlation is exploited by searching for similarities between the current frame to be coded and the previously coded frames. Knowing that video frames are typically displayed at a frame rate of 20 to 30 frames per second to the user, it is easy to understand that neighboring frames often show high resemblance. Hence, instead of sending or storing the current image segments, the encoder searches for similar previously coded segments, and only codes the difference between the current segment and its prediction. As these differences are often small, high compression ratios can be achieved in practice.

Finding the best prediction is a cumbersome process which involves a lot of computations at the encoder. As a result, the complexity distribution between the encoder and the decoder shows an imbalance, with an encoder that is roughly 5 to 10 times more complex than the decoder. This favors applications in which decoding devices need to be simple, cheap, and/or small.

A radically new way for performing video coding was discovered only recently, although the theoretical results were known for 30 years already. This

new topic is called *distributed video coding* (DVC). The term *distributed* can be explained in a theoretical context, but in practice these systems usually consist of one encoder and one decoder, with the main characteristic that the complexity bulk is shifted to the decoder's side. This is realized by generating the prediction at the decoder instead of at the encoder. However, while the encoder could select the best prediction by comparing possible candidates with the current block to be coded, the decoder can only observe what it has decoded so far. Using this information the decoder needs to estimate what the current block or frame at the encoder's side looks like.

As the estimation generated by the decoder is often error-prone, additional information is sent from encoder to decoder. This additional information is calculated by the encoder on the original, and it allows the decoder to correct errors in its estimation.

Given the two extremes of a complex encoder but simple decoder in conventional video coding, and a simple encoder but complex decoder in DVC, in a first contribution we combine these ideas to develop a flexible architecture. This system allows distributing complexity between the encoder and the decoder in a dynamic way, in the sense that complexity can be shifted between the encoder and the decoder on-the-fly, i.e., during the coding process. For example, if the encoder has more computational resources available than the decoder, it can take over some of the workload and vice versa. Such a flexible system can be particularly useful in situations where available complexity is non-static, for example, in the case of multi-tasking, battery-constrained devices or session mobility. Moreover, the system could serve in any situation as conventional coding and DVC are simply two extremes of the spectrum of possible complexity distributions.

One of the problems with this architecture is that the compression performance of the DVC techniques is still quite low compared to the performance of conventional coding solutions. Hence, to make the flexible system more competitive, in the remainder of this dissertation we focus on improving compression performance for DVC.

An important problem in DVC is that the decoder needs to have accurate knowledge about the quality of predicting the missing information. This problem is known as estimating the correlation between the original at the encoder and the prediction at the decoder. If the decoder is better able to estimate this correlation, it can correct errors in its prediction more efficiently (i.e., with less bits). Hence, one way to achieve better compression performance is to improve the accuracy of estimating the correlation. In this dissertation, current techniques for correlation noise estimation are improved by focusing on a number of problems.

A first problem is that the accuracy of current techniques decreases as quantization noise in the reference frames increases. As a solution, a new model is presented that accounts for the quantization noise as well. A second problem is related to the accuracy of generating the prediction at the decoder. While current techniques typically assume motion to be linear, this assumption does not always hold. Therefore, the correlation model is extended so that uncertainties about the assumptions of linear motion are taken into account as well. Both improvements lead to significant performance gains.

In a final contribution, the set of possible coding modes is extended. Having additional coding modes to choose from allows avoiding common DVC inefficiencies, such as inaccuracies in the correlation model. An efficient strategy is defined to decide between the different coding modes using equations for rate and distortion. The latter describe compression performance from a mathematical point of view. As shown by the results, extending the set of possible coding modes leads to large compression gains compared to existing DVC solutions found in the literature.

The improvements presented in this dissertation have contributed significantly to the current research in DVC. Several problems have been identified and new solutions have been presented. This research has enabled us to successfully narrow the performance gap between DVC and conventional video coding.

These improvements can be coupled back to the flexible architecture where we started from, increasing its performance. Also, DVC techniques can be useful in other systems as well, extending the set of coding modes with DVC-like coding modes, for example. As such, the results of this dissertation are not only useful in a DVC context, but also in the context of video coding in general.

Samenvatting

De hoeveelheid aan digitale video informatie groeit in een razend snel tempo. Er wordt naar video gekeken in de huiskamer, op het internet, en zelfs op de trein via mobiele telefoontoestellen en draagbare DVD spelers. Het opslaan, verwerken, en versturen van al deze informatie is niet vanzelfsprekend. Door de hoge eisen van de gebruikers zien we langs de ene kant een groei in beeldresolutie, kwaliteit en gebruikerservaring. Langs de andere kant is er ook een toenemende miniaturisatie, waarbij de toestellen die in staat zijn om video te verwerken steeds kleiner en kleiner worden.

Door praktische beperkingen qua opslagcapaciteit, verwerkings- en transmissiesnelheid, is het nodig om de informatie op een compacte manier voor te stellen. Dit is meteen ook de reden waarom onderzoek naar nieuwe videocompressietechnieken zeer belangrijk is. Nieuwe technieken kunnen bijvoorbeeld toelaten om meer videos op te slaan op harde schijf, of in hogere kwaliteit door te sturen.

Een systeem voor videocompressie bestaat typisch uit een encoder en een decoder. De taak van de encoder is om de video te converteren naar een compacte voorstelling, terwijl de decoder net de omgekeerde operatie uitvoert. In conventionele systemen is het de encoder die de correlatie uitbuit waardoor compressie wordt gerealiseerd. Vermits de illusie van bewegende beelden in feite wordt gecreëerd door stilstaande beelden met een voldoende hoge snelheid aan de gebruiker te tonen (typisch zo'n 30 beelden per seconde), is het zo dat opeenvolgende beelden veelal sterke gelijkenissen vertonen. Deze correlatie kan uitgebuit worden door een voorspelling te genereren voor het huidige beeld, op basis van reeds gecomprimeerde beelden. Vervolgens wordt enkel het verschil tussen het huidige beeld en deze voorspelling verwerkt, en gecomprimeerd door middel van zogenaamde entropiecodering. Hierbij worden grote compressieratios gerealiseerd.

Het zoeken naar een goede voorspelling voor het huidige beeld bepaalt de graad van compressie. De nieuwste technieken evalueren dan ook een zeer groot aantal mogelijke voorspellingen, vooraleer een definitieve beslissing te

nemen. Dit heeft ervoor gezorgd dat de encoder typisch veel complexer is dan de decoder. Een dergelijk systeem heeft bijvoorbeeld voordelen wanneer de grootte, het energieverbruik of de kost van de decoder belangrijk is.

Recentelijk zijn er echter ook nieuwe technieken voorgesteld waarbij de voorspelling niet wordt gegenereerd door de encoder, maar door de decoder. Deze systemen worden gedistribueerde videocoderingssystemen genoemd, of in het Engels *Distributed Video Coding (DVC) systems*. Het "gedistribueerde" aspect kan verklaard worden in theoretische context, maar in de praktijk gaat het meestal over een compressiesysteem met 1 encoder en 1 decoder, met als specifieke eigenschap dat de complexiteit zich nu aan decoderzijde bevindt in plaats van aan encoderzijde zoals bij de conventionele systemen.

Bij DVC wordt het huidige beeld voorspeld aan decoderzijde, op basis van reeds gedecodeerde beelden. Vermits deze voorspelling meestal nog redelijk veel fouten bevat, wordt er extra informatie gestuurd door de encoder. Deze extra informatie – berekend op het originele beeld – maakt het voor de decoder mogelijk om de fouten in zijn voorspelling te verbeteren.

Merk op dat de decoder enkel kan gebruik maken van reeds gedecodeerde beelden. Bij de conventionele systemen kon de encoder de beste voorspelling kiezen door deze te vergelijken met het originele beeld. Dit is bij DVC uiteraard niet mogelijk, wat als resultaat heeft dat het genereren van de voorspelling in DVC een stuk uitdagender is dan bij conventionele systemen. Niettemin is het theoretisch bewezen dat de compressieperformantie van beide alternatieven eenzelfde niveau kan bereiken.

Het is niet altijd voor alle toepassingen duidelijk of het nu beter is om het meeste rekenwerk uit te voeren aan encoderzijde (zoals in de conventionele systemen) of aan decoderzijde (zoals in DVC). Bij sommige systemen is de beschikbare rekenkracht namelijk variabel, bijvoorbeeld, omdat er verschillende taken tegelijkertijd kunnen uitgevoerd worden (zogenaamde *multitasking* systemen). Andere systemen worden dan weer gekenmerkt door een niet-constante energievoorziening (bv. via batterijen). Dergelijke dynamische factoren hebben me ertoe aangezet om ideëen uit beide technieken te combineren, zodat de verdeling van het rekenwerk dynamisch kan worden aangepast aan de beschikbare middelen. Als de encoder meer rekenkracht beschikbaar heeft dan de decoder, dan kan deze meer op zich nemen en vice versa. Dit resulteert in een adaptief systeem dat zich aanpast aan de beschikbare rekenkracht. DVC en conventionele videocompressie zijn hierbij slechts twee uitersten van het spectrum.

Het grootste probleem met de architectuur die we hiervoor hebben ontwikkeld, is dat de performantie van DVC vrij laag is in vergelijking met de performantie van de conventionele technieken. Daarom wordt er in de rest

van dit doctoraat aandacht besteed aan het verbeteren van de performantie van DVC.

Een eerste bijdrage in deze context is het verbeteren van de nauwkeurigheid van de correlatieruisschatting. Voor het corrigeren van de schatting aan decoderzijde is het nodig om informatie te hebben omtrent de kwaliteit van deze schatting. Deze kwaliteit kan niet worden gemeten, vermits het origineel zich aan encoderzijde bevindt, en de predictie aan decoderzijde. Dus moet de kwaliteit op zijn beurt geschat worden. In wiskundige termen wordt dit uitgedrukt als het schatten van de correlatie tussen het origineel en de predictie. Indien de nauwkeurigheid van deze schatting hoger is, dan kunnen de fouten in de predictie eenvoudiger (d.w.z. met minder bits) gecorrigeerd worden.

In dit doctoraat worden de bestaande technieken voor correlatieruisschatting verbeterd. In een eerste uitbreiding onderzoeken we het effect van quantizatieruis, en stellen we technieken voor die hiermee beter rekening houden. In een tweede uitbreiding onderzoeken we het effect van niet-lineare beweging, en verfijnen we het model verder. Beide uitbreidingen resulteren in significante compressiewinsten.

In een laatste bijdrage worden er verschillende codeermodes toegevoegd aan het systeem. Door de decoder te laten kiezen tussen deze codeermodes kunnen een aantal inefficiënties eigen aan DVC beter aangepakt worden. De keuze tussen de modes wordt bepaald via formules voor bitsnelheid en distortie, hetgeen een wiskundige beschrijving is van de compressieprestatie. Deze uitbreiding levert grote winsten in compressie, vergeleken met de huidige DVC systemen die in de literatuur kunnen worden teruggevonden.

De technieken die in dit doctoraat zijn beschreven hebben de compressieperformantie van DVC sterk verbeterd. Hierdoor is het verschil met de conventionele aanpak een heel stuk kleiner geworden. De uitbreidingen voor DVC kunnen worden teruggekoppeld naar de flexible architectuur vanwaar we vertrokken waren. Ook in conventionele systemen kunnen deze technieken interessant zijn, bijvoorbeeld door extra DVC-gerelateerde modes toe te voegen. Hierdoor zijn de resultaten in dit doctoraat niet enkel interessant voor DVC, maar voor videocompressie in het algemeen.

List of abbreviations

AVC	Advanced Video Coding
CIF	Common Intermediate Format
CRC	Cyclic Redundancy Check
DC	Direct Current
DCT	Discrete Cosine Transform
DISCOVER	Distributed Coding for Video Services
DISCUS	Distributed Source Coding using Syndromes
DSC	Distributed Source Coding
DVD	Digital Versatile Disc
DVC	Distributed Video Coding
FEC	Forward Error Correction
fps	Frames per Second
GOP	Group Of Pictures
IDCT	Inverse Discrete Cosine Transform
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
ITU	International Telecommunication Union
IQP	Intra Quantization Parameter
JSCC	Joint Source-Channel Coding
kbps	Kilobits per Second
LAPP	Logarithm of the a Posteriori Probability
LDPC	Low-Density Parity-Check
LDPCA	LDPC Accumulate
LP	Low-Pass
LR	low Resolution
MAD	Mean Absolute Difference
Mbps	Megabits per second
MPEG	Moving Picture Experts Group
MSE	Mean Squared Error
MV	Motion Vector
NRWZ	Non-reference Wyner-Ziv
PDA	Personal Digital Assistant
PRISM	Power-Efficient, Robust, Highcompression, Syndrome-Based Multimedia Coding
PSNR	Peak Signal-to-Noise Ratio

QCIF	Quarter CIF
QP	Quantization Parameter
RD	Rate-Distortion
SAD	Sum of Absolute Differences
SISO	Soft-Input Soft-Output
SNR	Signal-to-Noise Ratio
SSE	Sum of Squared Errors
WZ	Wyner-Ziv
XOR	Exclusive Or

Contents

Chapter 1

Introduction

Several trends can be observed in the context of digital video processing, storage, transmission and display. A first trend is that devices capable for processing video are getting smaller and smaller. A mobile phone can hardly be called just a phone these days because it has become more than that. Most of these devices support – apart from the conventional calling functions associated with a phone – taking pictures and recording videos, connecting to the internet, route planning, etc. As such, they are evolving from a phone to a mini-computer as they support a subset of applications found on conventional personal computers. As the devices are getting smaller and smaller, with the number of supported features getting larger and larger, the need for efficient algorithms for processing and storing information becomes evident. This is even more true due to limited power-supply of devices running on batteries.

A different trend can be observed towards higher resolutions and improved user experience. Television broadcasters are switching from analog to digital, trying to convince their customers by offering higher quality video, extra channels, and additional services such as video-on-demand. Research towards improving the user experience is ongoing, for example, in the context of free viewpoint video [1, 2] and 3-D television [3, 4]. These developments result in a large increase in the amount of video information that needs to be processed, transmitted, or stored. Hence, also in this case, efficient algorithms are needed to handle this explosion of information, representing the video information in a compact way.

As a result, video compression is more than ever an important field of research, and several systems and standards have been developed through the years.

A video compression system typically consists of an encoder communicating information to a decoder. The encoder converts the uncompressed video

stream into a compressed stream suitable for storage or transmission, while the decoder performs the inverse operation. If the output of the decoder is identical to the input of the encoder then the system is operating in a lossless fashion. In this case, the compression performance of the system can be evaluated through the bit rate (i.e., the number of bits per second) of the coded stream. The smaller the bit rate, the better the compression. However, an even smaller bit rate can be obtained by allowing some loss of quality between the video sequence at the encoder's input, and the sequence at the decoder's output. While this might seem undesirable from a user's point of view, in most cases introducing loss to improve compression is a necessary step given practical limits and/or costs. As such, lossy video compression is used frequently in practice. The performance of a lossy video coding system is characterized by two parameters: the (coded) bit rate and the amount of quality loss (also called the distortion).

The main idea behind video compression is to exploit redundancies. Nearby frames typically show strong resemblance, as well as nearby pixels and blocks in the same frame. These temporal and spatial correlations are exploited to achieve compression, by sending/storing only what is different compared to previously coded information. The current values to be coded are predicted by evaluating a (usually large) number of candidate predictors taken from previously coded information. Next, only the residual between the current values and the best predictor is coded using entropy coding techniques.

Finding a good prediction of the current block of information is important, as better predictors lead to better compression ratios. Consequently, one of the strategies in video compression has been to significantly extend the set of candidate predictors. As a result, the task of finding the best predictor requires a lot of computations.

Different video compression[1] solutions have been proposed through the years. Opposed to the current evolution of these techniques, a new paradigm has been introduced only recently. This new way for performing video compression is called *distributed video coding* (DVC), and it forms the main subject in this dissertation.

Before describing the general idea behind DVC, we first explain how video compression is performed the "conventional" way.

[1] In related literature, *video compression* and *video coding* are often used as synonyms, despite the fact that the term *coding* can be found in other contexts as well (e.g., referring to computer programming, or classification schemes). However, unless explicitly stated, in this dissertation, *coding* refers to *compression*.

1.1 Conventional video coding

Numerous video coding systems and standards have been developed through the years. The standards have always been more widespread than the proprietary systems, due to global compatibility and interoperability. The most popular standards have been published by the International Telecommunications Union (ITU), or by the International Organization for Standardization (ISO) in conjunction with the International Electrotechnical Commission (IEC). Experts from different standardization bodies have been collaborating as well, particularly for the more recent standards.

Some examples of video coding standards are H.261 [5] and its successor H.263 [6], which have been widely used in video conferencing applications. MPEG-2 [7] has been used for digital broadcast and as a storage format on DVD. This standard also included the MPEG-2 audio layer 3 specifications, which is known as the popular MP3 audio format for distributing audio on the internet. The more recent standards include MPEG-4 Visual [8], and H.264/AVC [9]. The latter can be considered the current state-of-the-art, providing bit rate reductions of about 50% over its predecessors.

As more intelligent techniques were used, the encoder's task to search for the best predictor has become highly-complex. As a result, complexity is out of balance, with an encoder that is significantly more complex than the decoder. In practice, this translates to relatively simple, small, and cheap decoders, at the expense of more sophisticated encoders. Such properties are particularly useful in cases where sequences are coded only once but decoded many times, or in cases where the size or cost of the decoding device is an issue. This scenario more or less fits the current multimedia landscape, where a large number of users receive content from different providers (such as television stations, movie producers, and online media providers) using relatively cheap or simple decoding devices.

1.2 Distributed video coding

A new way for performing video coding has been introduced in the last decade. This new paradigm, called distributed video coding (DVC), shifts the complexity from the encoder to the decoder. This shift is realized by making the decoder responsible for generating the prediction signal, hereby relieving the encoder from this complex task. However, as the encoder could select the best predictor based on a comparison with the original to be coded, the decoder can not perform this comparison as it has only previously decoded information at its disposal, and not the original. This implies that the decoder has a more

challenging task than the encoder in producing the same performance results, since it only has access to reconstructed signals.

The task of the decoder in DVC is to estimate the missing information based on what has been decoded so far. This estimation is called the side information. Since the side information is often inaccurate, the encoder sends error correcting information calculated on the original. This information enables the decoder to correct the side information, through a channel decoding procedure.

As the complexity burden now resides at the decoder instead of at the encoder, DVC translates to cheap, small and power-friendly encoding devices, possibly at the expense of decoders with opposite characteristics. This setup facilitates a different range of applications, where the main focus and constraints are on the video (capturing and) coding devices, instead of on the decoding (and displaying) devices.

Some examples of these applications include [10]:

Video conferencing with mobile devices

Video conferencing adds an extra dimension to the standard phone call by allowing people in distinct places to communicate in both a visual and auditive way. While in some cases it could be more effective to have a face-to-face meeting, video conferencing can be preferred in situations where traveling time and costs are an issue.

To offer more freedom, a flexible setup can be used in which users communicate through mobile devices. Given the restrictions of these devices, transcoding services can be provided by components in the network, hereby alleviating workload on the mobile devices. An example scenario is depicted in Figure 1.1.

Wireless sensor networks

Sensor networks typically consist of a large number of small, inexpensive sensors monitoring certain parameters in their environment, and communicating these events in a wireless fashion. Different types of sensing devices may be employed, optical (e.g., cheap cameras such as Cyclops [11]), or non-optical (e.g., thermal, electrical, or biological sensors). Typically, sensors are locally powered as providing wiring is unpractical or even impossible.

Applications for sensor networks have been identified in various domains, such as environmental monitoring, healthcare, and defense. Some examples include monitoring parameters of an active volcano [12], monitoring human body parameters [13], and sniper detection [14].

Given the low-complexity and low-power constraints of the sensors, distributed video coding can be considered an ideal candidate in the context of visual sensor networks.

Wireless capsule endoscopy

Endoscopy has been widely used to analyze anomalies in the gastrointestinal tract of the human body, for example, in the context of coeliac disease, Crohn's disease and the detection of tumors [15]. As the common wired approach provides pain and discomfort for the patient a new emerging solution is to use a wireless technique, where an endoscopic capsule – the size of a large pill – is swallowed by the patient [16]. The endoscopic capsule contains a battery, a camera, and a small transmitter. The capsule transmits images as it passes through the patient's digestive system, which allows abnormalities to be detected by medical staff.

Obviously, in this case, having small endoscopic capsules is a considerable advantage for the comfort of the patient. This clearly favors the use of DVC over conventional approaches.

Multi-view video entertainment

Several cameras can be used to capture one particular scene, instead of using a single one. Such a multi-view approach can offer the viewer the choice between several views on the scene, or in extension, it could allow the user to freely control the viewpoint position of any dynamic real-world scene.

Although very interesting, one of the problems with multi-view entertainment is that it increases the amount of video information significantly. Luckily, due to correlation between the views, sequences captured by different cameras can be compressed jointly. In a conventional approach, the encoders for each view need to communicate among each other, in an attempt to obtain the best

> predictor. Using a DVC approach, however, communication between the encoders is avoided, since it is the decoder that generates the predictions for the views, and the task of each encoder is to send information for the corresponding view only [17].

Another advantage of DVC is that it is robust in error-prone environments. If the compressed video sequence is sent across an unreliable network, information can get lost in transmission or get corrupted for example due to network congestion or channel fading. To allow error detection and information recovery at the receiver side, typical techniques add forward error correction (FEC) bits as redundancy to the bits representing the compressed data. This separation of source coding (i.e., compression) and channel coding (i.e., protection against errors) has been proved to be optimal [20] under a number of assumptions. However, systems based on this separation principle do not cope well when the channel quality is variable, and they tend to break down completely in cases where the channel quality falls under a certain threshold, and the channel code is no longer capable of correcting the errors.

As a reaction to this problem, systems have been designed in which source coding and channel coding is performed jointly, showing a more graceful degradation of quality as the channel conditions worsen [21–23]. In this context, one of the advantages of DVC is that joint source-channel coding (JSCC) is inherently supported, showing increased performance compared to conventional solutions applying separate source and channel coding [24, 25].

1.3 Outline of this dissertation

DVC is an interesting topic, forming the main subject in this dissertation. To provide more context, Chapter 2 briefly introduces the basic concepts and the most important systems described in DVC literature. The remaining chapters describe the contributions that have led to this dissertation.

As discussed in this introduction, conventional video coding approaches are characterized by a highly-complex encoder, while an opposite complexity distribution is observed in the case of DVC. Both approaches can be combined to create hybrid systems. However, we can take this one step further and generalize these ideas to a system where the distribution of complexity between the encoder and the decoder is dynamic. For example, based on the amount of computational resources at either end, the encoder can take over some of the workload from the decoder or vice versa. Such a flexible system allows on-the-fly redistribution of complexity, having conventional video coding (complex encoding, simple decoding) and DVC (simple encoding, complex decoding)

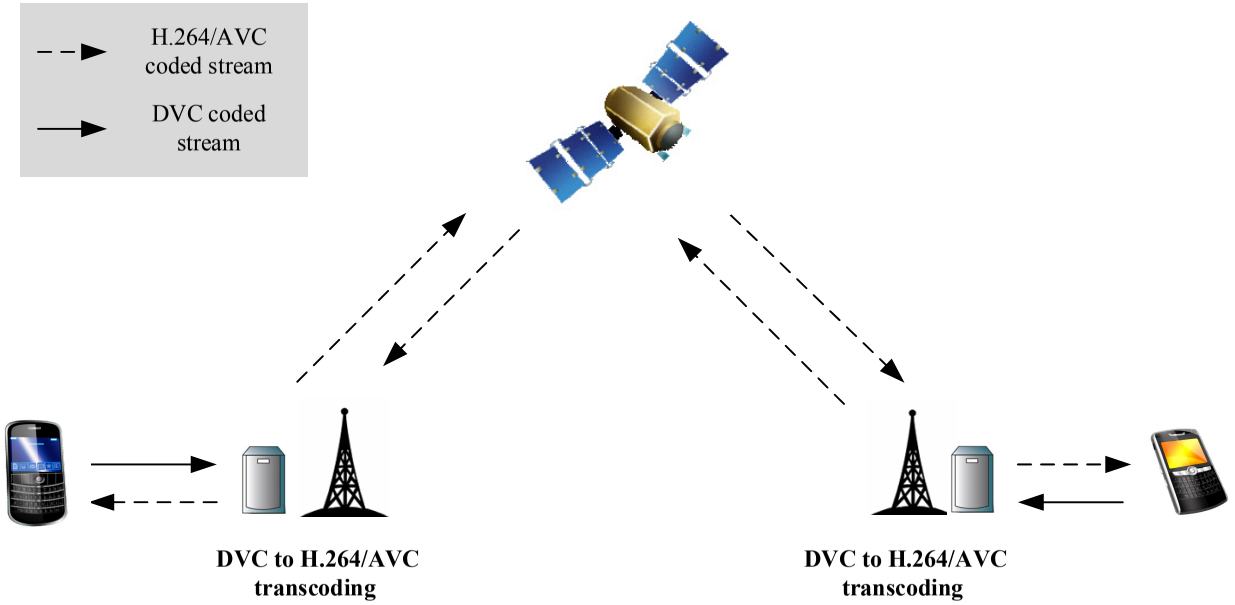

Figure 1.1: Example of a transcoding architecture for video conferencing with mobile devices. Video is captured and coded on the mobile devices using a low-complexity DVC encoder. In the network, the coded stream is converted to a different format, for example, H.264/AVC. This computationally expensive operation involves DVC decoding followed by H.264/AVC encoding, possibly exploiting overlap between both steps [18, 19]. At the other end of the communication, low-complexity H.264/AVC decoding operations can be used.

as the two extremes. This new idea is described as a first contribution in Chapter 3 of this dissertation.

One of the problems with this new system is that current DVC techniques do not provide the same compression performance as conventional techniques. Therefore, the remaining chapters focus on improving DVC compression performance. These results can then be fed back to the flexible architecture described in Chapter 3, making the system even more interesting.

To improve compression performance in DVC, several problems are identified and solutions are presented in the following chapters. One of the major problems in DVC is estimating the quality of the side information at the decoder. This problem is known in mathematical terms as estimating the correlation between the original and the side information. As the original is obviously not available at the decoder, estimating the correlation is a difficult task, especially because of spatial and temporal non-stationarity. Having accurate correlation information at the decoder-side is nonetheless of great importance, due to its impact on coding performance.

Several techniques for correlation estimation are proposed in Chapter 4. In a first improvement, we refine current techniques by analyzing the effect of quantization noise in the side information. A second improvement compensates for inaccurate assumptions made when generating the side information. Both improvements result in significant compression gains.

One of the reasons behind the superior performance of conventional solutions such as H.264/AVC, is the use of a large set of coding modes out of which only one is selected to perform the actual coding operation. In DVC, it is less straightforward to define a large number of modes, but even adding some of the most essential modes such as skip and intra can already give high benefits. This is explored in Chapter 5, where a rate-distortion model is developed to decide which coding mode to use.

The conclusions of this dissertation can be found in Chapter 6.

The work described in this dissertation has led to three book chapters (one as a first author), five journal publications (two as a first author), and eleven publications presented on international conferences (of which four as a first author).

Chapter 2

Introduction to distributed video coding

This chapter provides a brief overview of some of the most important results in distributed video coding (DVC), both theoretical and practical. The main purpose of this chapter is to provide sufficient background for the contributions presented in the following chapters of this dissertation.

The theoretical foundations of DVC do not apply to video in particular. Instead, they have been developed in the context of *distributed source coding* (DSC). In such a setup, correlated sequences are coded using independent encoders (but a joint decoder). Bounds on the compression performance of such DSC systems have been derived in the case of lossless source coding by David Slepian and Jack K. Wolf, as early as 1973. Their work – discussed in Section 2.1.1 – can be regarded as a fundamental mathematical first step for DVC. A second fundamental theoretical contribution is due to Aaron D. Wyner and Jacob Ziv, as described in Section 2.1.2. Their work extends the work of Slepian and Wolf in the case of lossy source coding with side information available at the decoder.

These theoretical contributions also apply to the special case of video coding, but practical DVC systems were only developed almost thirty years later. The two pioneers in the field are the so-called PRISM system, described in Section 2.2.1, and the DVC architecture developed at Stanford University, discussed in Section 2.2.2. Both systems are compared in Section 2.2.3.

These fundamental architectures have been extended by numerous researchers throughout the years. One of those important extensions has been developed in the context of the DISCOVER project. The resulting architecture – commented on in Section 2.2.4 – can be regarded as the current state-of-the-art in DVC. For this reason, the DISCOVER architecture has been used as a

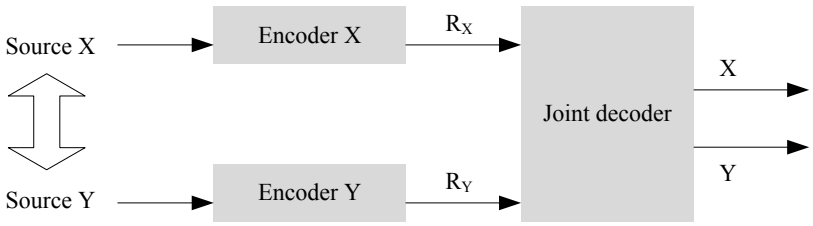

Figure 2.1: Distributed source coding of two correlated sources X and Y using separate encoders and a joint decoder.

starting point for the new techniques presented in this dissertation.

2.1 The theoretical foundations

The information-theoretic results from Slepian and Wolf, and Wyner and Ziv provide bounds for compression performance. These results will be discussed in the following sections, starting with the results from Slepian and Wolf.

2.1.1 Lossless distributed source coding – Slepian-Wolf

The configuration considered by Slepian and Wolf [26] is depicted in Fig. 2.1. In this setup, two sources X and Y generate correlated sequences of information symbols. Each of these sequences is compressed by a separate encoder, i.e., one for X and one for Y. The encoder of each source is constrained to operate without knowledge of the other source, hence the term *distributed* source coding. The decoder, on the other hand, receives both encoded streams as input, which allows exploiting the statistics between both sequences for reconstructing the output of X and Y.

Surprisingly, Slepian and Wolf proved that the compression bound is the same as in the case where both encoders are allowed to communicate. More specifically, they prove that the rates R_X and R_Y satisfy the following set of equations:

$$\begin{aligned} R_X + R_Y &\geq H(X,Y) \ , \\ R_X &\geq H(X \mid Y) \ , \\ R_Y &\geq H(Y \mid X) \ , \end{aligned} \tag{2.1}$$

where $H(\cdot)$ denotes the entropy.

These conditions can be presented graphically as well, as a so-called *admissible* or *achievable rate region*, depicted in Figure 2.2. While any point on

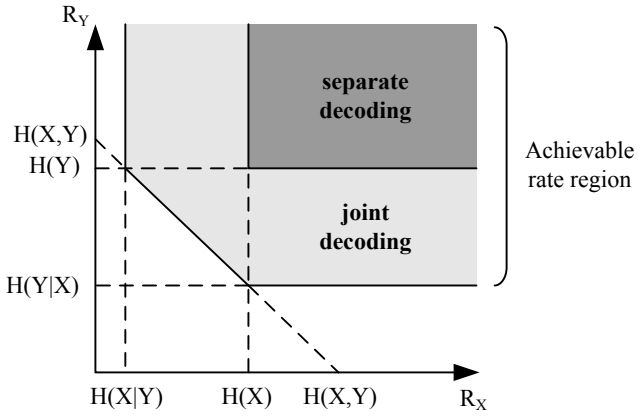

Figure 2.2: Achievable rate region for the coding of two correlated sources X and Y using independent encoders and a joint decoder.

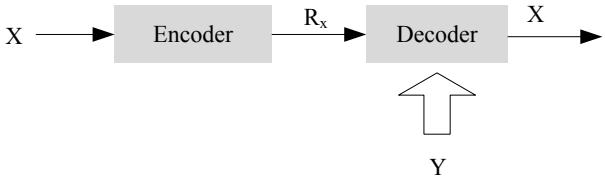

Figure 2.3: Compression with side information available at the decoder, a special case of distributed source coding.

the line defined by $H(X,Y)$ is equivalent from a compression point of view, special attention goes to the corner points of the achievable rate region (e.g., the point $(H(X|Y), H(Y))$). These corner points correspond to the special case of compression with side information available at the decoder (Figure 2.3). This special case is of particular interest in the context of distributed video coding. In Figure 2.3, side information Y can be coded at a rate R_Y close to the entropy $H(Y)$, and be used at the decoder to decode X. According to the Slepian-Wolf theorem, the rate R_X that is needed to reconstruct X reliably can be brought close to the conditional entropy $H(X \mid Y)$.

2.1.2 Lossy compression with receiver side information – Wyner-Ziv

The work of Slepian and Wolf involves lossless compression. This was extended to lossy compression by Aaron D. Wyner and Jacob Ziv [27–29].

Denote the acceptable distortion between the original X and the decoded signal $\hat{X}$ as $D = E[d(X, \hat{X})]$, where d is a specific distortion metric such as the mean squared error (MSE). Two cases are considered for compression with side information available at the decoder (Figure 2.3). In the first case, the side information is not available at the encoder, resulting in a compression rate denoted $R^{WZ}_{X|Y}(D)$. In the second case, the side information is available at the encoder as well, resulting in a rate denoted $R_{X|Y}(D)$.

Using these notations, Wyner and Ziv proved that

$$R^{WZ}_{X|Y}(D) - R_{X|Y}(D) \geq 0. \tag{2.2}$$

In other words, not having the side information available at the encoder results in a rate loss greater than or equal to zero, for the same distortion D. Interestingly, the rate loss was proved to be zero in the case of Gaussian memoryless sources and a mean squared error distortion metric [28, 29].

These results were extended later on by other researchers, for example, proving that the equality also holds for source sequences that are the sum of arbitrarily distributed side information and independent Gaussian noise [30]. Zamir [31] showed that the rate loss for sources with general statistics and a mean squared error distortion metric is less than 0.5 bits per sample.

2.2 Practical systems for DVC

The theoretical contributions of Slepian and Wolf, and Wyner and Ziv, provide bounds for the compression performance of a DSC system. These bounds also apply to the specific case of distributed video coding, in which the sources generate video data. However, the proofs do not provide insights in how to build a practical system able to achieve those bounds. As a result, practical systems had not been developed at the time. Instead, it took almost thirty years for the first systems to be proposed in the literature, in the late nineties and the beginning of this century.

Most solutions focus on a mono-view scenario, in which only one video sequence is compressed. To obtain the correlated sources X and Y mentioned by the Slepian-Wolf and Wyner-Ziv theorems, the video sequence is typically split into different parts. A first part is coded using conventional techniques (e.g., intra coding or skip). This information is decoded by the decoder, and used to generate a prediction Y of the missing information X. Next, the encoder sends information about X enabling the decoder to correct its prediction.

Two architectures can be considered as the pioneers in the field. A first system is the PRISM codec developed at Berkeley (University of California).

This system, discussed in Section 2.2.1, applies a block-based approach to split the video sequence into different parts. A second pioneering architecture is the codec developed at Stanford University, discussed in Section 2.2.2. The latter uses a frame-based approach, i.e., entire frames are predicted using already decoded frames.

A major difference between both systems is that the Stanford architecture adopts a feedback channel, where error correcting bits are sent in portions to the decoder upon request. While this is certainly less practical, it allows the Stanford architecture to better adapt to changing characteristics in the video sequence, which attributes to the higher performance observed for the Stanford codec when comparing to PRISM. A short discussion of the main differences between both systems is provided in Section 2.2.3.

Still, one of the problems with these pioneering architectures is their limited compression performance, especially when comparing to the state-of-the-art in conventional video coding. Hence, many researchers have focused on extending these DVC architectures to improve their compression performance. One of these extensions is called the DISCOVER codec. This codec can be regarded as the current state-of-the-art in DVC. It is largely based on the Stanford architecture, with some important improvements regarding side information generation and correlation noise estimation. More information about this codec is provided in Section 2.2.4.

Many others have proposed alternatives or improvements to existing DVC solutions, but providing a complete overview falls outside the scope of this chapter. Instead, references to other systems or techniques described in the literature will be provided in the following chapters – wherever relevant – when presenting the contributions that have led to this dissertation.

2.2.1 PRISM

In 1999, Pradhan and Ramchandran proposed a technique called Distributed Source Coding Using Syndromes (DISCUS) [32, 33] and (together with additional authors) they created a framework for video coding which they called Power-efficient, Robust, hIghcompression, Syndrome-based Multimedia coding (PRISM) [34–36].

In PRISM, each frame is divided into non-overlapping blocks of size 8×8 or 16×16. Each of these blocks B is classified as skip, intra or Wyner-Ziv (WZ), based on thresholds obtained through an offline training stage. Only the blocks in the WZ class are decoded using side information at the decoder. The classification scheme is based on the squared error difference between B and the colocated block B^p in the previous frame. If B and B^p show strong corre-

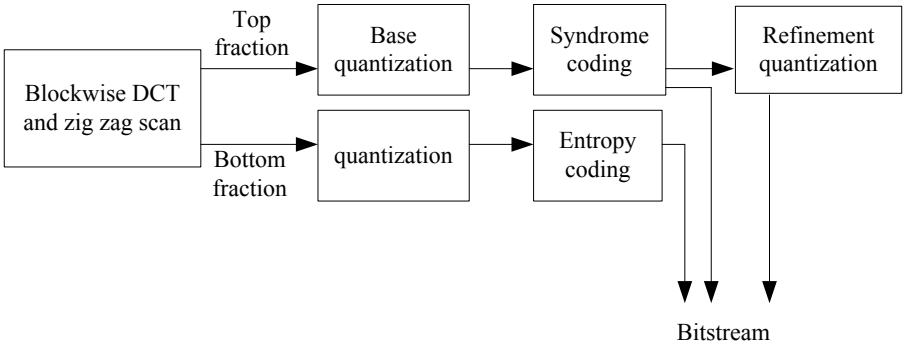

Figure 2.4: The PRISM encoder.

lation, B is signaled to the decoder as SKIP, and the decoder simply takes the colocated macroblock in the previous frame as the result. On the other hand, if there is very little correlation, then B is intra coded as the side information at the decoder will probably not be very accurate.

In all other cases, B is WZ coded as illustrated by Figure 2.4. Each block is first transformed using a discrete cosine transformation (DCT) and the coefficients are scanned in zig-zag order. The higher frequency coefficients (denoted *bottom fraction*) are intra coded, i.e., they are quantized and run-length Huffman (entropy) coded. Only the low frequency coefficients (denoted *top fraction*) are actually WZ coded. First, the low frequency coefficients are *base quantized*, i.e., they are quantized with a quantizer step size that is proportional to the expected difference between the original (at the encoder) and the side information (at the decoder). The base quantized low frequency coefficients are then channel coded using syndrome codes. To achieve the target distortion, the quantization is further refined, and the index of the refinement interval inside the base interval is transmitted to the decoder. In addition, a CRC check is calculated which serves as a signature of the quantized codeword sequence.

The decoder (Figure 2.5) performs motion search in already decoded frames, obtaining blocks that can be used as side information for the WZ coded blocks. From the set of candidate side information blocks, the best predictor is selected using the Viterbi algorithm and the received syndrome bits. If the decoded result matches the CRC, syndrome decoding terminates. Otherwise, the next best predictor is chosen and so on.

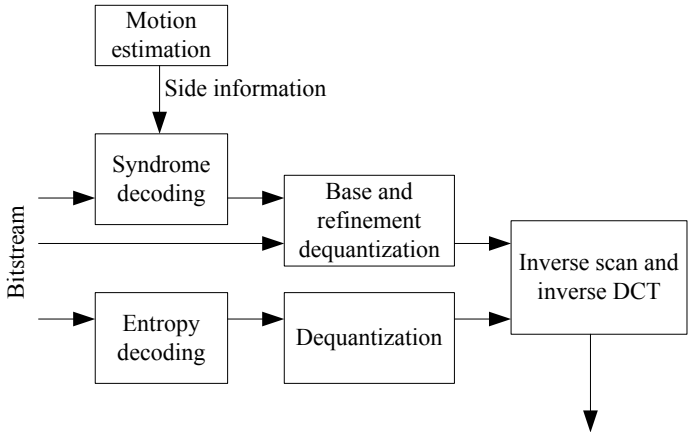

Figure 2.5: The PRISM decoder.

2.2.2 The Stanford codec

Instead of using a block-based approach as in PRISM, Aaron et al. adopt a frame-based approach and they partition the video sequence into I frames (*key* frames) and WZ frames. First, a pixel-domain codec has been developed [37–39], which was later on extended to the transform domain [40–42].

In the transform-domain architecture [40] (Figure 2.6)[1], at the encoder, key frames are intra coded using conventional intra coding techniques such as H.263+ intra coding. WZ frames are partitioned into non-overlapping blocks of size 4×4 or 8×8. Each of these blocks is transformed using a DCT, and the coefficients at the same position k in every block are grouped together, forming coefficient bands $\mathbf{X_k}$. For example, all DC-coefficients form coefficient band $\mathbf{X_0}$. Next, each coefficient band $\mathbf{X_k}$ is uniformly quantized into $\mathbf{q_k}$ by a 2^{M_k}-level quantizer. The output of the quantizer is written as a binary string, and the bits at corresponding positions are grouped into so-called bitplanes. For example, each first bit of each DC-coefficient will form one bitplane, each second bit will form another bitplane, and so on. Finally, each bitplane is coded by a turbo coder [43] and the result is stored in a buffer. A feedback channel is used where portions of bits from the buffer are sent to the decoder upon request.

At the decoder, side information is generated for each WZ frame, using already decoded frames as references. A hierarchical GOP structure is used,

[1] Many modules in this system are described in detail in the remainder of this dissertation.

meaning that the sequence $I_1 - WZ_1 - WZ_2 - WZ_3 - I_2$ is coded and decoded in the following order: $I_1 - I_2 - WZ_2 - WZ_1 - WZ_3$. For example, the side information for WZ_1 will be generated using I'_1 as a past reference frame, and WZ'_2 as a future reference frame [39]. Remark that $'$ has been used to indicate decoded frames. Several techniques for generating the side information have been proposed, such as pixel-by-pixel interpolation of the reference frames [37], and motion compensated interpolation and extrapolation [40]. From these techniques, the best performance is achieved when using motion compensated interpolation.

The side information Y is transformed by a DCT, and used by the turbo decoder in a Viterbi-like decoding procedure. The turbo decoder requests parity bits (also called WZ bits) from the encoder's buffer via the feedback channel, until reliable decoding is achieved. Usually, a residual bit error probability of at most 10^{-3} is tolerated.

The turbo decoder returns for each coefficient the quantization bin with very high probability. The following step – called *reconstruction* – is to select one particular value in this bin as the decoded coefficient. The reconstruction strategy used is to select the value in the quantization bin that is closest to the side information. The result is inverse transformed.

2.2.3 Comparing PRISM and the Stanford codec

It is important to realize that the use of a feedback channel for requesting WZ bits in the Stanford codec is a significant problem in practice. First of all, the support for a feedback channel can only be provided in streaming scenarios, which excludes storage applications. Also, even in streaming scenarios channel delay might be too large to grant the decoder multiple requests for WZ bits per bitplane.

Some researchers have proposed extensions to the Stanford codec in order to develop a feedback-free architecture as in PRISM. One of the disadvantages of these solutions [44–47] is that the elimination of the feedback channel adds complexity to the encoder, and often there is a significant performance penalty as well. This performance penalty is caused by the fact that the feedback channel enabled adapting well to the temporal variability in the number of WZ bits needed by the decoder.

Most researchers have continued working on Stanford-based extensions that still make use of the feedback channel, in an attempt to get DVC compression first at a level that is acceptable and interesting for real-world applications. This motivation also applies to the work described in this dissertation. Once we are able to lift compression performance to an acceptable level, we can

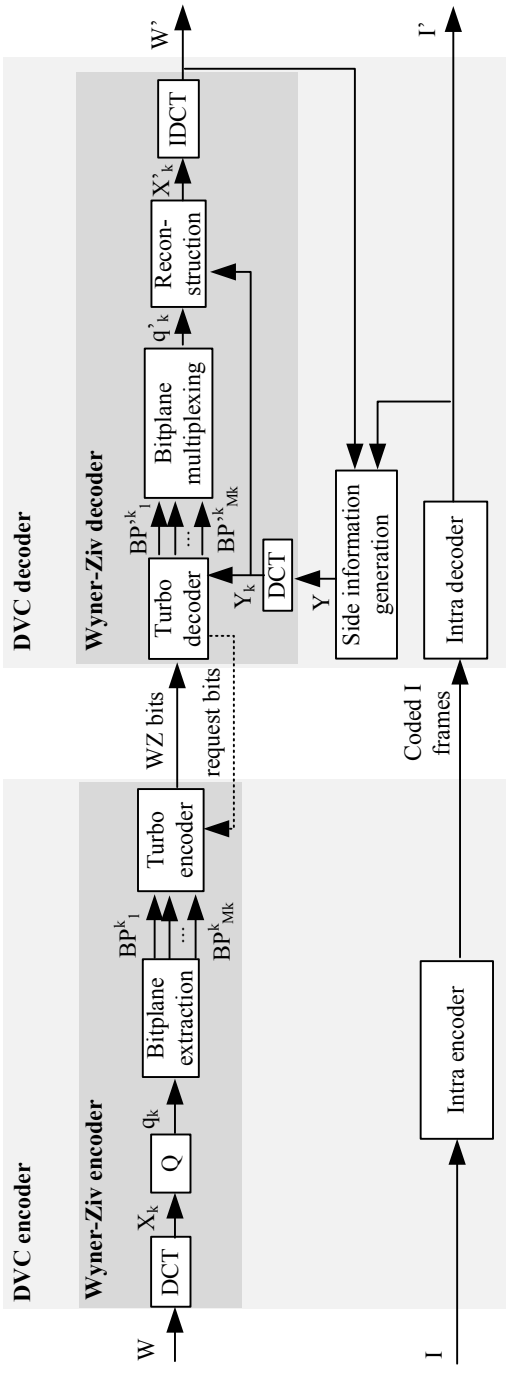

Figure 2.6: Architecture of the DVC codec by Aaron et al. [40].

investigate how to constrain or remove the feedback channel to fit real-world applications.

Another important difference between both systems is that PRISM operates on blocks, while the Stanford codec operates on bitplanes. Both approaches have their advantages. Using a block coding approach has the advantage that one can adapt to spatial differences within one frame. On the other hand, bitplane coding allows neglecting high frequency information easily. Both strategies have been combined in one system, for example, by Ascenso et al. [48], in an attempt to benefit from both approaches.

2.2.4 DISCOVER

PRISM and the Stanford codec can be considered the pioneers in the field, and significant research effort has been spent to improve these architectures. Most authors have been focusing on the Stanford architecture. One of these extensions has been developed in the context of a European-funded project called DISCOVER[2]. This 27-months project started in September 2005, and six partners from different universities in Europe were involved.

Due to its compression performance, the DISCOVER codec can be considered among the current state-of-the-art in DVC. It outperforms the Stanford codec as well as PRISM for the majority of sequences [49]. The codec also provides an excellent benchmark, as its executables have been made available online [50].

The DISCOVER codec extends the Stanford codec in several ways [51,52], but it remains bitplane-based. The feedback channel is also still present. One of the most important improvements concerns the generation of the side information. In DISCOVER, more advanced techniques are used [53,54], featuring unidirectional motion search between the key frames, subpixel refinement [55] and spatial smoothing. More information about this method will be provided in one of the following chapters (Section 3.3.4), since this method has been implemented in the system developed in the context of this dissertation.

Another important extension concerns the modeling of the correlation between the original frame (at the encoder) and the side information (at the decoder). This information is needed for efficient Viterbi-like decoding. Additional details can be found in Chapter 4, where more advanced techniques for correlation noise estimation are proposed.

Opposed to using turbo codes as in the Stanford codec, DISCOVER employs a particular set of low-density parity-check (LDPC) codes [56], called

[2] DIstributed COding for Video sERvices.

rate-compatible LDPC Accumulate (LDPCA) codes [57]. These codes perform slightly better than turbo codes [58].

Other improvements include a coarse estimation of the WZ rate for reducing the number of requests through the feedback channel (similar to [59]), optimal centroid reconstruction of the decoded coefficients [60], and the use of a cyclic redundancy check (CRC) as an LDPCA stopping criterion. Similar or alternative techniques are discussed in this dissertation, and details will be provided where applicable.

Chapter 3

Flexible distribution of complexity

3.1 Introduction

One of the challenges of video streaming scenarios is coping with the bandwidth requirements of different networks, and the heterogeneity of devices. Devices with different characteristics are performing video compression and streaming, ranging from high-end servers to PDAs and mobile phones. The available computational complexity of these devices is often non-static. The complexity budget may vary due to a large number of processes running on a multitasking system. Other devices might experience variable power supply, for example, due to batteries. In such cases, spare complexity can be traded for additional battery lifetime, for example, through techniques such as dynamic voltage scaling [61].

As such, besides rate and distortion, available computational complexity is considered an important parameter in a video coding system. Several techniques have been proposed in the literature for dealing with complexity issues, primarily in the context of predictive video coding. In predictive video coding, motion estimation is a very computationally complex task, due to the high number of coding modes and the high computational complexity of many of these modes. To reduce complexity at the encoder, low-complexity alternatives to conventional solutions have been proposed frequently in the literature. Some of these solutions attempt to reduce complexity by excluding the less likely coding modes in the mode decision process [62–64]. Other techniques try to reduce calculations in the modes, for example, by proposing low-complexity motion estimation techniques [65–67].

A rate-distortion-complexity function is often used as well, which allows,

for example, to select coding modes based on a trade-off between compression performance and decoding complexity [68]. Another interesting contribution describes a complexity-scalable encoder where the operation of several modules is driven by power constraints [69]. Encoder-side complexity scalability has also been studied in the context of wavelet video coding by Turaga et al. [70].

Despite all these techniques, only encoder and/or decoder-side complexity reduction is considered, without the possibility of shifting some of the workload to the other side. Due to the use of a decoder loop at the encoder, the encoder remains more complex than the decoder at any time.

DVC systems feature a reversed complexity distribution, in the sense that it is now the decoder that is significantly more complex than the encoder. However, the main focus of current research is on compression performance, and complexity is only rarely considered in a DVC context. Limited complexity analysis has been provided for DISCOVER [52].

Due to the complexity of the motion estimation process, in this chapter, a codec is presented that is able to share motion estimation flexibly between the encoder and the decoder. While current solutions only allow encoder and/or decoder complexity to be decreased, our system allows redistributing complexity. Such a system is better suited for dynamic environments involving multitasking, battery-constrained devices, or session transfers between devices, for example. In such situations, at a certain point in time the encoder could have more resources available than the decoder, but this could be the other way around later on. The proposed codec is able to follow these changing conditions.

The idea of shifting motion estimation between the encoder and the decoder was new at the time this research was performed, although a few systems could be considered somewhat related. These systems are discussed first in Section 3.2. After this discussion, in Section 3.3.1, the general operation of the codec is described. This codec features several modes for coding inter frames, using either the predictive mode with motion estimation at the encoder (Section 3.3.3), the DVC mode with motion estimation at the decoder (Section 3.3.4) or one of the hybrid modes where motion estimation is shared (Section 3.3.5).

Complexity analysis is provided for each of the coding modes in our system, using the theoretical model for complexity presented in Section 3.3.2. This model is validated by practical measurements in Section 3.5, showing correspondence between the theoretical results and the practical measurements. This complexity analysis is then used for choosing which coding mode to use for each frame in a group of pictures (GOP), while meeting complexity con-

straints at both encoder and decoder. The details of this mode decision strategy can be found in Section 3.6. In Section 3.7 we describe some of the recent developments by other researchers, in the context of flexible distribution of complexity. Finally, conclusions end the chapter (Section 3.8).

3.2 Related work

Several systems in the literature allow motion estimation to be shared between the encoder and the decoder. However, at the time this research was performed, none of these systems allowed to share this task dynamically (to the best of our knowledge). In this section, we limit our discussion to systems where motion estimation is shared statically. This has been the starting point of the work presented in this chapter. The more closely-related dynamic systems are described in Section 3.7, where they are compared directly to the techniques presented in our work.

A first (non-dynamic) system worth mentioning is due to Tom Clerckx et al. [71]. This system applies a simple block-based motion estimation procedure at the encoder. More specifically, for each block, the encoder selects the best predictor out of a reduced set of 3, 6, or 9 candidate predictors. There is no decision scheme for choosing between the number of candidate predictors to take, instead, the different configurations are compared. Only the p most significant bitplanes are considered (out of a total of M bitplanes), and sent to the decoder via skip, entropy-coding (i.e., intra coding), or DVC. At the decoder, the p most significant bitplanes are decoded, while the remaining $M-p$ bitplanes are filled in by decoder-side generated side information.

While this system features some interesting hybrid techniques for sharing motion estimation, complexity can not be shifted between the encoder and the decoder.

Oh et al. [72] propose a system where motion vectors are calculated remotely. A very coarse representation of the current frame to be coded is communicated to the decoder, or to some other component present in the network. This component then calculates motion vectors, and sends them back to the encoder for conventional motion compensated predictive coding. Similar ideas have been used by Liu et al. [73], following a DVC approach for obtaining the motion vectors at the decoder.

In the context of flexible distribution of complexity, a straightforward extension of these two systems would be to choose, for each frame, the component responsible for motion estimation. Based on a complexity target, a method could then be developed to switch between both possibilities.

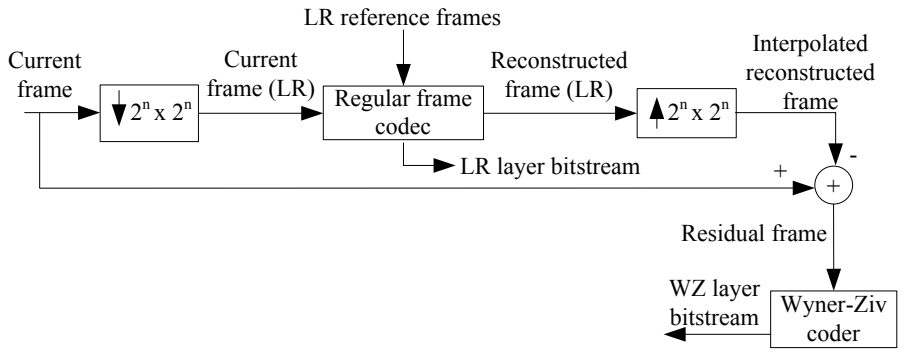

Figure 3.1: Spatially scalable encoding architecture proposed by Mukherjee [74].

Another interesting hybrid system is proposed by Mukherjee [74]. A regular coder is used to code frames that serve as reference frames for motion estimation. The remaining frames are referred to as non-reference Wyner-Ziv (NRWZ) frames, and they are coded using a hybrid approach. At the encoder (Figure 3.1), NRWZ frames are subsampled with a factor $2^n \times 2^n$. The resulting low-resolution (LR) frames are coded using a regular coder (e.g., H.263+) which uses LR reference frames for motion estimation, and the LR bitstream is sent to the decoder. The reconstructed (decoded) LR frame is interpolated to full resolution, and the residual with the original frame is WZ coded.

At the decoder (Figure 3.2), the LR frame is decoded by the regular frame decoder. The frame is interpolated using the same interpolation filter as the encoder, and the result is refined in a procedure called *motion based semi super-resolution*. Next, the noisy residual between the refined frame and the non-refined frame is corrected by the WZ decoder, using the WZ layer bitstream. Finally, the decoded NRWZ frame is obtained by adding the interpolated decoded frame to the corrected residual.

The flexibility for distributing complexity is still quite limited in this system. One of the main reasons is that the proposed hybrid approach is only applied to the inter frames that are not used as references, i.e., the NRWZ frames. Other inter frames are still coded following traditional techniques (with encoder-side motion estimation). It is also unclear from this work how to choose the subsampling factor n, as well as how to adapt the motion based semi super-resolution to changes in n. Intuitively, one would expect more decoder-side calculations as n becomes larger.

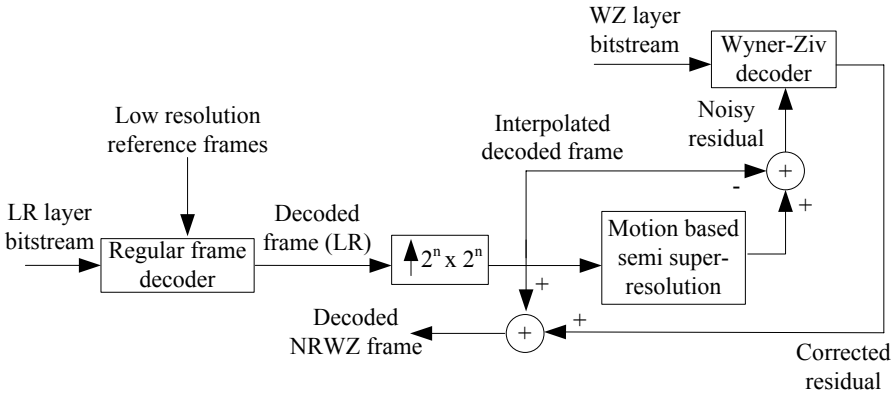

Figure 3.2: Spatially scalable decoding architecture proposed by Mukherjee [74].

3.3 Description of the proposed video codec

In this chapter, a system is presented that features several modes for coding frames. Each mode shares the complex task of motion estimation differently between encoder and decoder: the predictive mode with motion estimation performed by the encoder, the DVC mode with motion estimation performed by the decoder, and the hybrid modes where motion estimation is shared. Two variants for the hybrid modes are developed, using a spatial partitioning technique on the one hand, and a partitioning of the motion search algorithm on the other hand.

The widely-used Stanford DVC architecture [40] has been taken as basis. As such, the frame sequence is partitioned into intra frames I and inter frames W. For the inter frames, motion estimation is performed at the encoder and/or decoder, depending on the mode. Only the part involved in motion estimation operates differently for each mode, therefore, the discussion is split into two parts.

Firstly, those modules are discussed that are functionally independent from the particular coding mode used. This includes the WZ codec, the intra codec and the buffering system. Next, the interaction between these modules is described (Section 3.3.1).

Secondly, the motion estimation part is described in detail for each mode (Section 3.3.3 to Section 3.3.5), and complexity is analyzed using a theoretical model for complexity (Section 3.3.2).

Before we continue, it is probably good to remark that the implementation of the codec is not my own personal achievement exclusively. A large

portion of the (C++) programming work has been done together with my colleagues Jozef Škorupa and Stefaan Mys. Due to his education in mathematics, Jozef had accepted the challenging task to design and implement the turbo codec. Stefaan first integrated H.264/AVC intra coding, while I programmed the framework and basic functions such as transformation and quantization. In a later stage, Stefaan and I closely collaborated for implementing the remaining modules. As the programming and testing phases took a considerable amount of time (about 6 to 9 months), we agreed upon sharing the output publications of the end result. As such, Jozef and Stefaan both published a conference paper as a first author [75, 76]. I had the opportunity to publish a journal paper [77], since the ideas concerning flexible distribution of complexity – as described in this chapter – were originally mine. Using the architecture as a common basis, during the following months and years of our PhD we tried to improve the codec by developing new techniques individually, specializing in different areas.

3.3.1 General codec operation

Our codec operates as follows. At the encoder (Figure 3.3), intra frames I are coded using H.264/AVC intra coding (JM 12.1). Decoded intra frames I' are available anyway after intra coding due to mode decision and rate-distortion optimization, hence, I' frames are stored in the *decoded I frame buffer*.

For each inter frame W, both encoder and decoder generate a prediction Z. How Z is generated depends on the mode, but at all times, the same prediction is obtained both at the encoder and at the decoder. This allows using a residual approach, by coding the residual R between W and Z. Such a residual approach has shown to improve compression, as illustrated in a DVC context by Aaron et al. [78], for example. Z is stored in the *prediction frame buffer*.

R is partitioned into non-overlapping blocks of 4×4 pixels, which are transformed using the H.264/AVC transformation. This transformation, described in Appendix A, is a computationally efficient approximation of the discrete cosine transform (DCT). The result of this transformation is that each block of 16 pixels is converted into a block of 16 transform coefficients. Next, for all blocks, coefficients at the same index are grouped into so-called coefficient bands. For example, each third coefficient in each block will be collected, forming the third coefficient band. Next, each band is quantized using a quantizer with 2^{M_k} quantization levels (or *bins*). The zero bin of this quantizer is larger than the other bins (1.5 times larger in this case), referred to as a *dead-zone* quantizer. Next, for each band, bits at identical positions are grouped into bitplanes BP_i^k. For example, all most significant bits of all DC coefficients

will form bitplane BP_0^0. Finally, parity bits are generated by the turbo coder for each bitplane, and stored in a buffer. These bits are sent in portions to the decoder upon request. More information about the turbo codec is provided in Appendix B.

At the decoder (Figure 3.4), intra frames are decoded into I' and stored in the *decoded I frame buffer*. For each inter frame W, side information Y is generated as well as the prediction Z. The residual Y^R between Y and Z is transformed and used by the turbo decoder. The turbo decoder requests as many bits as needed until Y^R is corrected. When all bitplanes are decoded by the turbo decoder, they are multiplexed and the unquantized coefficients are reconstructed using centroid reconstruction, as in [60]. The result is inverse transformed into R', and Z is added to obtain the decoded frame W'. For future reference, W' is stored in the *decoded W frame buffer*.

The turbo decoder needs information about the reliability of the side information. This reliability – more specifically, the correlation between the original coefficient and the side information coefficient – is modeled online using the method described by Brites and Pereira [79] (at coefficient-frame level). More accurate correlation models will be discussed in the following chapter of this dissertation.

So far, the main operation of the codec has been described, without going into detail about how the prediction Z and the side information Y is actually created. The creation of Z and Y is mode dependent, and more details will be provided further on, along with a theoretical analysis of complexity. Details about how this theoretical analysis will be performed, is provided first in the following section.

3.3.2 Modeling motion estimation complexity

The execution speed of a program depends on parameters such as the number of operations to be executed, the speed of these operations (additions, multiplications, conditional expressions, etc.), the number of memory requests and the delay associated with these requests (which depends on the memory architecture, cache behavior, and so on). Modeling complexity theoretically by taking all these parameters into account is difficult, and hardware dependent.

However, to get an idea of how complexity is distributed between the encoder and the decoder for the different modes of our system, we propose to use a general approach and model computational complexity through the number of data transfers from and to memory. This is motivated by the fact that these data transfers are crucial factors that determine the performance of multimedia applications. This is illustrated for example by Brockmeyer et al., who

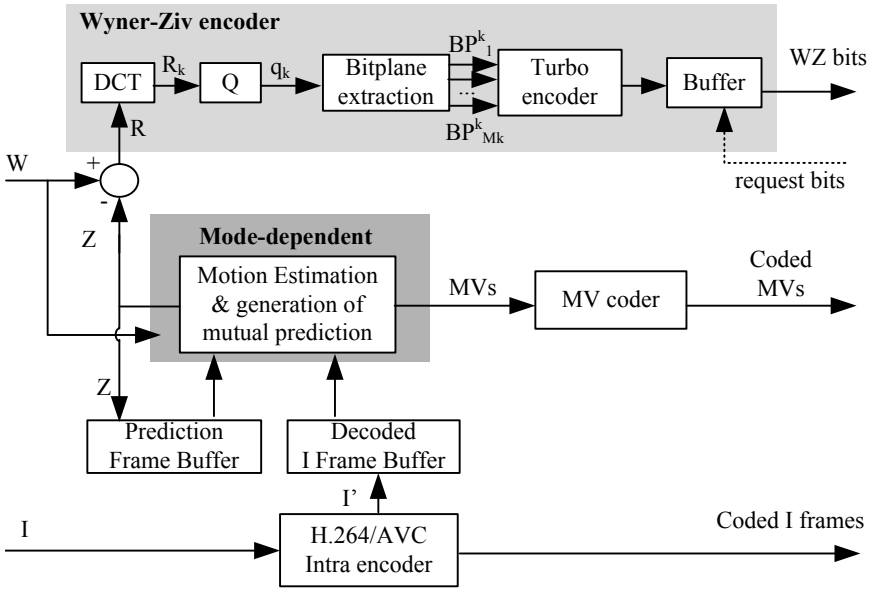

Figure 3.3: Encoder architecture consisting of a WZ encoder, an intra encoder, a mode-dependent part and a mechanism for buffering frames.

estimated that a software implementation of an MPEG-4 video encoder (VM 7.0) typically requires about $5 \cdot 10^9$ memory transfers per second to encode the simple profile level L2 [80].

It is clear that a general model based only on the amount of data transfers will have its limits in describing the complexity balance for a large variety of implementations and platforms. For example, hardware implementations could be more determined by other parameters such as the number and length of pipelines, data caches, etc. These extensions are left as future work.

Figure 3.5 depicts the model that will be used in this chapter for estimating computational complexity. More specifically, each step in the motion estimation process is generalized as an operation performed on pixel[1] data. To perform a particular operation (such as spatial interpolation or Lagrangian cost calculation) a number of pixels needs to be read. These pixels are read from the original frame, from a past or future reference frame, from a temporarily stored frame such as the current version of the side information, etc. After performing the desired operation, the frame store might be updated by writing the new pixel values as output to the desired location, but this step is optional.

[1] In the context of pixel read and write operations, the term "pixel" will be used to indicate one particular luma or chroma value.

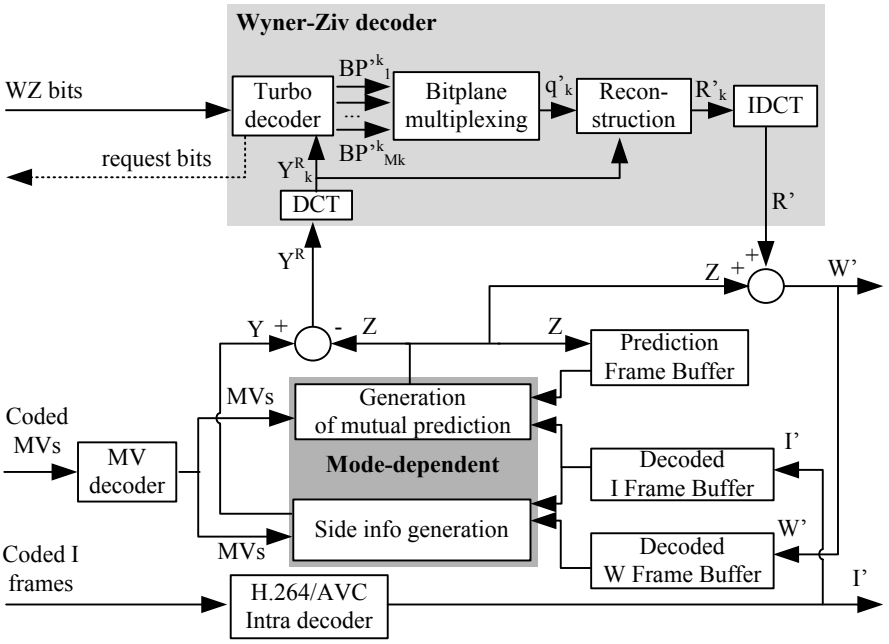

Figure 3.4: Decoder architecture consisting of a WZ decoder, an intra decoder, a mode-dependent part and a mechanism for buffering frames.

As a complexity measure, in the following sections of this chapter the total number of such pixel read/write operations (or briefly referred to as "operations") is counted, for each of the modes. The accuracy of this model is evaluated in Section 3.5.

Frames are assumed to be in YUV format, and unless stated otherwise, motion estimation is performed only on the luma component while motion compensation is performed on both luma and chroma. 4:2:0 subsampling is used, so that frames have a (spatial) luma resolution of H(orizontal) by V(ertical) pixels whereas the chroma components are each $H/2$ by $V/2$.

3.3.3 Motion estimation in the predictive video coding mode

In the predictive video coding mode, motion estimation is solely performed at the encoder. For each inter frame W that needs to be coded, the closest past frame P and closest future frame F are retrieved from the *prediction frame buffer* and/or *decoded I frame buffer*. W is partitioned into non-overlapping blocks of size 8×8 (luma) pixels, called macroblocks. Next, each macroblock in W is compared to a number of candidate blocks in P, and the block corre-

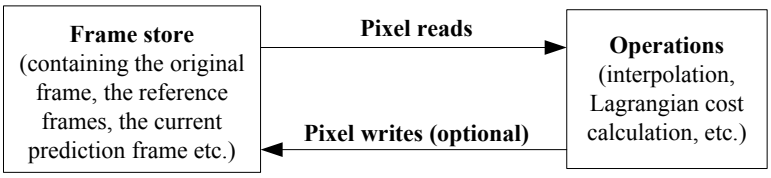

Figure 3.5: Modeling complexity by pixel read and write operations.

sponding to the best match is selected (the criterion used for this selection is described further on). The S candidate blocks in P to consider are defined by the search window.

The complexity of this step is calculated as follows. Let M be the size of the macroblocks in W, e.g., with $M = 64$ in the case of 8×8 blocks. Each of the HV/M macroblocks in W is compared to S candidate blocks in P. Comparing one block of M (luma) pixels in W to one block of M (luma) pixels in P results in $2M$ pixel read operations. Hence, the total complexity for this step is HV/M times S times $2M$, or $2SHV$ operations. This result as well as the results from all following steps are listed in Table 3.1 and Table 3.2 as a reference.

Remark that in this case we have assumed for simplicity that the motion estimation process is not terminated early, for example, in case the current intermediary result has high probability for being equal or equivalent to the final result [81, 82].

The best match in P is found by minimizing a Lagrangian cost function:

$$Cost_i(\vec{v}) = D_i(\vec{v}) + \lambda R_i(\vec{v}) \tag{3.1}$$

where the distortion metric $D_i(\vec{v})$ is the Sum of Squared Errors (SSE) between the current macroblock B_i and the macroblock in P defined by the motion vector $\vec{v}$. λ is a Lagrange multiplier that has been determined offline using several sequences, and which is set to 30, 65, 110, or 180 for quantization matrices Q_0 to Q_3 (which are defined further on in this chapter). $R_i(\vec{v})$ represents the rate to code $\vec{v}$. Motion vectors are coded by first predicting them from their neighbors as in H.264/AVC and coding the residual between $\vec{v}$ and its prediction using signed exponential Golomb coding, resulting in $R_i(\vec{v})$ bits.

After retrieving the best match B^P in P for a block B in W, a second prediction for B is obtained from F. First, the average between each of the candidate blocks in F and B^P is calculated. B is then compared against each of the averages using the same cost function, and the block in F corresponding to the best match is selected.

This operation is similar to the previous step, requiring one additional block to be read. Therefore, a total of $3SHV$ operations are needed.

The result from the motion estimation process is that we have now two motion vectors for each macroblock in W, i.e., one referring to a block in P and one to a block in F. Since the motion vectors calculated at the encoder will also be available at the decoder, they are used to generate the prediction Z through bidirectional motion compensated interpolation.

More specifically, each pixel in Z is constructed as the average of one pixel in P and one pixel in F. Hence, there are two read operations and one write operation, for each of the $3/2HV$ pixels (i.e., luma or chroma values). This results in a total of $9/2HV$ operations.

At the decoder, Z is constructed using the received motion vectors and the reference frames P and F (retrieved from the *prediction frame buffer* and/or *decoded I frame buffer*). Y is taken equal to Z so that the residual Y^R between Y and Z that is used by the turbo decoder contains only zeros.

3.3.4 Motion estimation in the DVC mode

In the DVC mode, no motion estimation is performed by the encoder. The prediction Z equals the closest frame, past or future, that can be retrieved from the *prediction frame buffer* and *decoded I frame buffer*.

At the decoder, side information is generated for each frame W using the closest past frame P and closest future frame F, retrieved from the *decoded W frame buffer* and *decoded I frame buffer* only, since decoded frames have better quality than prediction frames Z.

The side information Y is generated based on the work of Artigas et al. [51], implemented in the DISCOVER codec. This technique consists of several steps, as illustrated by Figure 3.6, and explained next.

Firstly, for better capturing the true motion field, the luma component of P and F is low-pass (LP) filtered by replacing each pixel by the average of a group of $L = 3 \times 3$ pixels having this pixel as a center. As each pixel in an LP filtered frame is constructed by reading L pixels and writing one, this implies $HV(L + 1)$ operations per frame[2].

After LP filtering the reference frames, unidirectional block-based motion estimation is performed between the filtered versions of P and F. The candidate motion vectors are scaled with the distance $\Delta_{P,F}$ between P and F, with $\Delta_{P,F} = 1$ if the frames are adjacent to each other. This is performed for

[2]From here on, complexity analysis will only be provided for steps that are not similar to techniques analyzed earlier in this work. The reader is referred to Table 3.1 and Table 3.2 for a complete overview.

Figure 3.6: Side information generation as adopted in DISCOVER [51]. For each WZ frame, past and future reference frames are retrieved and low pass filtered. Next, unidirectional motion estimation is performed, from future reference frame to past reference frame. For each block in the WZ frame, the closest intersecting motion vector is chosen and treated as a bidirectional motion vector. These vectors are then further refined on a subpixel level, and spatially smoothed. The final motion vectors are used to create the side information through bidirectional motion compensation.

Table 3.1: Number of pixel read/write operations at the **ENCODER**, for the different modes, per frame W. Absolute numbers (expressed in $\cdot 10^6$ operations) are provided as an example, for: $H = 352$ and $V = 288$, $S = 1089$, $M = 64$, $L = 9$, $S_R^{DVC} = 16$, $S_R^{SPAT} = 25$, $S_R^{SUB_1} = 25$, and $S_R^{SUB_2} = 9$.

	Description	**Pixel read/write ops.**	**Example**
Predictive mode	Motion estimation(W, P)	$2SHV$	220.8
	Motion estimation(W, F)	$3SHV$	331.2
	Construct Z	$9HV/2$	0.5
	Total		552.4
DVC mode			
	Total		–
Hybrid spatial mode	Motion estimation(S_1)	$5SHV/2$	276.0
	Construct Z	$9HV/2$	0.5
	Total		276.5
Hybrid subsample mode	Subsample	$15HV/4$	0.4
	Motion estimation	$5SHV/16$	34.5
	Construct Z	$9HV/2$	0.5
	Total		35.3

compensating for possibly larger motion as the distance between the reference frames increases. Matching is performed using the following cost function (CF), as proposed by Ascenso et al. [53]:

$$CF(v_x, v_y) = (1 + 0.05\sqrt{v_x^2 + v_y^2}) \cdot \text{MAD}(v_x, v_y), \tag{3.2}$$

where (v_x, v_y) indicates the motion vector from F to P, and MAD is the Mean Absolute Difference between the corresponding blocks in F and P, defined by (v_x, v_y).

After obtaining the motion vectors from F to P, for each macroblock in Y the motion vector intersecting the block closest to the block center is chosen and treated as a bidirectional motion vector (Figure 3.7).

Table 3.2: Number of pixel read/write operations at the **DECODER**, for the different modes, per frame W. Absolute numbers (expressed in $\cdot 10^6$ operations) are provided as an example, for: $H = 352$ and $V = 288$, $S = 1089$, $M = 64$, $L = 9$, $S_R^{DVC} = 16$, $S_R^{SPAT} = 25$, $S_R^{SUB_1} = 25$, and $S_R^{SUB_2} = 9$.

	Description	**Pixel read/write ops.**	**Example**
Predictive mode	Construct Z	$9HV/2$	0.5
	Total		0.5
DVC mode	LP filtering	$2HV(L+1)$	2.0
	ME (F, P)	$2SHV$	220.8
	Wiener interpolation	$115HV$	11.7
	Refinement(16×16)	$\frac{512HVS_R^{DVC}}{M}$	7.3
	Refinement(8×8)	$\frac{128HVS_R^{DVC}}{M}$	1.8
	Spatial smoothing	$16HV$	1.6
	Bid. interpolation	$9HV/2$	0.5
	Total		252.8
Hybrid spatial mode	LP filtering	$2HV(L+1)$	2.0
	Wiener interpolation	$115HV$	11.7
	Refinement(16×16)	$\frac{256HVS_R^{SPAT}}{M}$	10.1
	Refinement(8×8)	$\frac{64HVS_R^{SPAT}}{M}$	2.5
	Bid. interpolation	$9HV/2$	0.5
	Construct Z	$9HV/2$	0.5
	Total		27.3
Hybrid subsample mode	LP filtering	$2HV(L+1)$	2.0
	Wiener interpolation	$115HV$	11.7
	Refinement(16×16)	$\frac{512HVS_R^{SUB_1}}{M}$	20.3
	Refinement(8×8)	$\frac{128HVS_R^{SUB_2}}{M}$	1.8
	Bid. interpolation	$9HV/2$	0.5
	Construct Z	$9HV/2$	0.5
	Total		36.7

Next, the LP-filtered versions of P and F are upsampled to half pixel precision using a 6-tap Wiener interpolation filter (only the luma). The full quality reference frames P and F are upscaled as well (all color components) for the purpose of motion compensation, explained further on.

The complexity of the half-pixel Wiener interpolation filter is calculated as follows. The luma component of a high-resolution frame is constructed by first copying the HV values from the input frame to the high-resolution output, requiring one read and one write per pixel ($2HV$ operations). Next, the $3HV$ luma pixels in between are constructed as a weighted average of 6 surrounding values. This requires 7 operations for each of the $3HV$ luma pixels, so $21HV$ operations. In total, for the luma component, half pixel interpolation requires $23HV$ read/write operations. Likewise, $11.5HV$ operations are needed for both chroma components together. Hence, for the two luma only frames and the two YUV frames processed in this step of the motion estimation process, a total of $115HV$ operations are needed.

Using the half-pixel LP-filtered reference frames, the motion vector of each block in Y is refined in two passes: first using reference blocks of size 16×16 and next using reference blocks of size 8×8. At all times, the motion vector is assumed linear between P and F, and going through the block center. The refinement window for a certain block is defined by the motion vectors of the neighboring blocks. More specifically, the refinement process is defined as finding the vector that minimizes the MAD between past and future blocks, with the additional constraint that the backward motion vector (v_x, v_y) should satisfy:

$$min(v_x^B, v_x^D) \leq v_x \leq max(v_x^B, v_x^D), \tag{3.3}$$

$$min(v_y^A, v_y^C) \leq v_y \leq max(v_y^A, v_y^C), \tag{3.4}$$

where A is the top neighbor, B the left neighbor, C the bottom neighbor, and D the right neighbor.

Due to the fact that the refinement window size S_R^{DVC} is calculated using the neighboring motion vectors, it is not constant. S_R^{DVC} will be small (on average) if the motion vector field is smooth. On the other hand, S_R^{DVC} will be large if there are a lot of discontinuities in the motion vector field. A value for S_R^{DVC} has been obtained experimentally, using the setup described in the results section (Section 3.4). By averaging over all sequences and rate points, a value of $S_R^{DVC} = 16$ has been obtained. Hence, each of the HV/M motion vectors is refined by reading S_R^{DVC} candidate pairs of 16×16 blocks in the first pass and reading S_R^{DVC} candidate pairs of 8×8 blocks in the second pass. This results in a total of $S_R^{DVC} \cdot 512HV/M$ for the first pass and a total of $S_R^{DVC} \cdot 128HV/M$ operations for the second pass.

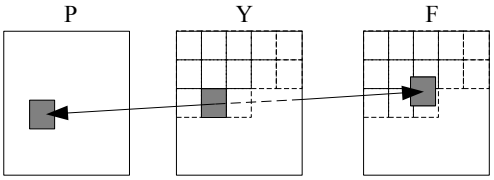

Figure 3.7: For each macroblock in Y, the closest intersecting motion vector is chosen and treated as a bidirectional motion vector.

After motion refinement, the motion vectors are spatially smoothed by weighted vector median filtering of the motion vector for B_i and the motion vectors of neighboring macroblocks applied to B_i.

For each of the HV/M macroblocks, applying the (maximum) eight neighboring motion vectors for calculating the weights used in median filtering results in eight times two blocks of size M to be read (one from the past reference frame and one from the future reference frame), which is a total of $16HV$ operations per frame for this step.

Finally, the calculated motion vectors are used for bidirectional motion compensation to obtain the side information frame Y.

3.3.5 Motion estimation in the hybrid video coding modes

In the hybrid modes, motion estimation is shared between encoder and decoder. Two variants for sharing motion estimation can be identified, based on spatial partitioning on the one hand and splitting of the motion estimation algorithm on the other hand.

A. Spatial partitioning

Motion estimation can be shared between encoder and decoder by partitioning the macroblocks in an inter frame W into two subsets S_1 and S_2, for which motion estimation will be performed at the encoder or decoder, respectively. At the encoder, the techniques developed in the predictive mode can be applied to the elements in S_1. In the DVC mode, however, as indicated in Table 3.2, the most computationally complex step is performing unidirectional motion estimation between the reference frames P and F. This step is performed to enable generating an initial motion vector estimate for all blocks in W at once, which does not directly allow us to leave out S_1, for which calculations have already been performed at the encoder side.

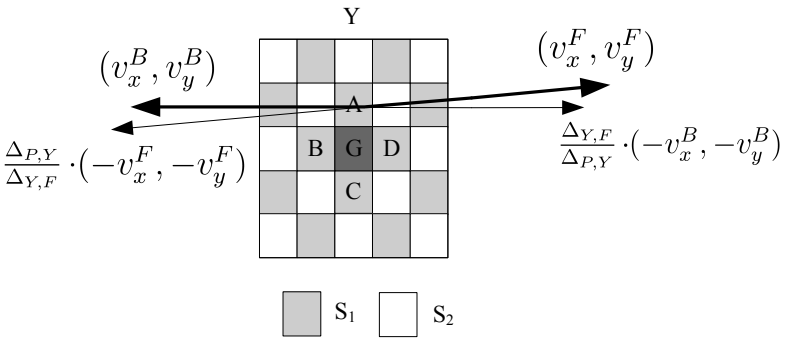

Figure 3.8: The hybrid spatial mode using a checkerboard pattern.

Still, unidirectional motion search between the (LP-filtered) reference frames can be avoided, by constructing S_1 and S_2 intelligently. For example, in this chapter, S_1 and S_2 are constructed as a checkerboard pattern. This way, unidirectional motion search can be avoided, generating initial motion vector estimates for each element in S_2 using neighboring blocks in S_1. Consider for example a non-border block G (Figure 3.8). As in the DVC mode, the motion between the reference frames is approximated as being linear, and an initial motion vector estimate for G is obtained by treating the backward and forward motion vector of each neighbor in S_1 (e.g., A in Figure 3.8) separately. As such, the bidirectional vector $((v_x^F, v_y^F), (v_x^B, v_y^B))$ is split into two bidirectional vectors describing linear motion: $((v_x^F, v_y^F), \frac{\Delta_{P,Y}}{\Delta_{Y,F}} \cdot (-v_x^F, -v_y^F))$ and $((v_x^B, v_y^B), \frac{\Delta_{Y,F}}{\Delta_{P,Y}} \cdot (-v_x^B, -v_y^B))$. This is done for all neighbors A, B, C, and D, resulting into eight vectors. Each of these motion vectors is applied to G, and the one minimizing the Sum of Squared Errors (SSE) is chosen.

This initial estimate is then used as a starting point for half-pixel motion refinement, as in the DVC mode. This operation is restricted to elements in S_2. In this case a refinement window of fixed size 5×5 ($S_R^{SPAT} = 25$) showed better results.

Subsequent to half-pixel refinement, bidirectional motion compensation is performed to construct the side information frame Y. No spatial smoothing is performed, since information from neighboring blocks is already taken into account during initialization of the motion vector.

To construct the prediction Z, each block in S_2 is assigned the motion vector of its left neighbor or if it does not exist, the vector of its right neighbor. This technique adds very little complexity but it enables constructing a prediction frame Z through bidirectional motion compensated interpolation.

B. Splitting the motion estimation algorithm

A second way to combine predictive video coding techniques and DVC is to split up the motion search algorithm, i.e., restrict the encoder search space for each macroblock instead of restricting the number of macroblocks for which encoder-side motion estimation needs to be performed. In other words, the encoder calculates coarse motion vectors which are further refined by the decoder. Due to the generality of this definition, hybrid modes can be constructed in several ways.

In this chapter, a subsampling approach is used. Other techniques can be considered as well, for example, a heuristic motion search algorithm such as a three step search (TSS) [83] could be split up in executing one or two steps at the encoder while executing the remaining steps at the decoder.

For the subsampling approach used here, W and its reference frames are subsampled to one fourth of the resolution. One pixel value at low resolution is obtained as the average of four pixels at full resolution. Next, bidirectional motion search is performed on blocks of size M, as in the predictive mode, but using a down-scaled search window of size $S/4$. Since the number of blocks of size M is reduced with a factor four, as well as the number of candidate vectors to consider, the encoder's computational complexity is reduced drastically. Subsequently, the low resolution motion vectors are coded and sent to the decoder. To create the prediction Z, the motion vectors are upscaled and used for bidirectional motion compensation.

At the decoder, P and F are retrieved from the buffers with decoded frames and LP filtered. The decoded motion vectors are upscaled and used as a starting point for motion refinement. As in the DVC mode, half pixel motion refinement is performed in two passes, operating on blocks of size 16×16 in the first pass and 8×8 in the second pass. The half pixel refinement window is set to a fixed size of 5×5 ($S_R^{SUB_1} = 25$) for the first pass and 3×3 ($S_R^{SUB_2} = 9$) for the second pass. As such, the vector is never refined more than three half pixels (which is less than two pixels, i.e., the accuracy of encoder-side motion estimation). No spatial smoothing step is performed afterward, but bidirectional motion compensation follows directly.

Remark from the complexity analysis in Table 3.1 and Table 3.2 that the hybrid subsampling mode (abbreviated "subs. mode") has low complexity both at the encoder and at the decoder. This is in contrast to the hybrid spatial mode, for example, where encoder-side complexity is much higher.

3.4 Rate-distortion performance

The rate-distortion performance of the system is compared to a number of different configurations. Firstly, the coding efficiency of the different modes is analyzed (Section 3.4.1). Next, the DVC mode is compared to the state-of-the-art found in the literature, i.e., the DISCOVER codec (Section 3.4.2). Finally, the predictive mode is compared to H.264/AVC (Section 3.4.3).

For all these results, tests have been conducted on three video sequences: Mother and Daughter, Foreman, and Table Tennis, containing very little, moderate, and relatively high motion respectively. All sequences have CIF resolution, and a frame rate of 30 fps. A GOP of length four is used, and for each sequence the maximum number of GOPs is coded (i.e., 297 frames: 74 GOPs plus one closing frame). Inter frames are hierarchically coded, meaning that the sequence $I_1W_1W_2W_3I_2$ is coded and decoded in the following order: $I_1I_2W_2W_1W_3$. Four different quantization patterns are used Q_0 to Q_3, quantizing each coefficient from 6 to 3 bits respectively.

The quantization of the intra and inter frames is chosen in an offline setting, as is common in the DVC literature. The sequence is coded several times using different intra quantization parameters per inter quantization pattern. Out of these results, only the one showing the best correspondence between average intra and inter decoded quality is used for reporting the results. A more practical solution has been proposed by Sofke et al. [84], but this solution requires additional complexity at the encoder. Hence, currently, merely all systems described in the literature adopt the offline approach.

Unless stated otherwise, rate distortion plots indicate the PSNR of the luma component as a function of the total rate of all color components.

The specifications of the test system that has been used for these experiments can be found in Table 3.3.

3.4.1 RD performance of the different modes

The rate-distortion results for the different modes are depicted in Figure 3.9. The differences in performance can be explained as follows.

Remark that there is typically a switch between the DVC mode and the predictive mode, in the sense that the DVC mode outperforms the predictive mode at low rates while this is the other way around at high rates. This difference is due to the coding of motion vectors in predictive mode. In predictive mode, motion vectors are used at the decoder side to generate the prediction Y, which is of better quality than the one obtained in the DVC mode. However, sending these motion vectors from encoder to decoder introduces a rate penalty that is not present in the DVC mode. At high rates, the rate of the

Table 3.3: Specifications of the test system used.

CPU	QC CLOVERTOWN XEON X5355 2.66Ghz 8M 1333FSB 120W
Memory	4 × DDR II 4048MB (DDR2-667 ECC FB-DIMM FMHS LP)
Operating sytem	Microsoft Windows Server R2, Standard X64 edition Service Pack 2

motion vectors is negligible compared to the WZ rate. At low rates, however, sending the motion vectors to the decoder is an important penalty, especially for sequences that can be well predicted at the decoder-side (such as Mother and Daughter). This explains why the DVC mode outperforms the predictive mode for Mother and Daughter significantly at low rates.

The same conclusions apply for the hybrid modes, which lie more or less in between the predictive mode and the DVC mode.

3.4.2 RD performance compared to DISCOVER

The DVC mode of the proposed system is compared to a similar system with decoder-side motion estimation, i.e., the state-of-the-art DVC codec from DISCOVER. As for the system proposed in this chapter, experiments are performed with the DISCOVER codec using a fixed GOP of size four, and the intra quantization parameters are chosen so that the quality of the intra decoded frames and WZ decoded frames are the same. Results are generated using WZ quantization patterns 1, 3, 6, and 7 (specified in, e.g. [52]). The results in this section are limited to the luma component only, since the DISCOVER codec does not take the chroma into account.

Similar performance is expected between the DVC mode and the DISCOVER codec, since side information generation and virtual noise estimation are similar in both systems. An important difference compared to DISCOVER is the use of residual coding in the proposed system. Especially for low motion sequences, residual coding is expected to increase compression performance, as illustrated by Aaron et al. [78].

These results are confirmed in Figure 3.10, showing better performance for the DVC mode compared to DISCOVER, for low-motion sequences such

as Mother and Daughter. For other sequences such as Foreman and Table, compression performance is similar.

These results indicate that the DVC mode of the codec described in this chapter is competitive with the state-of-the-art in DVC.

3.4.3 RD performance compared to H.264/AVC

The predictive mode is compared to the state-of-the-art in video coding with encoder-side motion estimation, namely H.264/AVC. To this extent, the H.264/AVC reference software (JM 13.2 [85]) is used to create two reference RD curves for each sequence: one with an IBBB (hierarchical) GOP structure, and one with only intra-coded frames (IIII). The extended profile is used, RDO enabled, one slice per picture, all coded using a fixed QP. Remark that in H.264/AVC often different QP's are used for the different hierarchical layers [86], but such an analysis has not been performed in the context of DVC so far. Hence, for now we use the same QP's throughout (for both I and B frames).

The results in Figure 3.11 indicate that H.264/AVC inter coding significantly outperforms the predictive mode (and the DVC mode). This is due to advanced techniques in H.264/AVC, such as adaptive block sizes, skip modes, etc. Only for simple sequences such as Mother and Daughter, compression performance can be considered comparable.

3.5 Validation of complexity analysis

The complexity for each of the modes has been calculated using the number of pixel read/write operations (Table 3.1 and Table 3.2). How these results should be mapped to the number of CPU cycles or milliseconds spent for coding each inter frame depends on hardware details such as calculation speed, cache behavior, etc.

However, if our complexity model is accurate then we should observe similar relationships between the complexity of each of the modes, in theory as well as in practice. In other words, if theoretical analysis indicates that the encoder in the predictive mode is twice as complex as the encoder in the spatial mode, then this should be observed in practice also. Hence, to compare theory and practice we normalize theoretical and measured complexity to the total complexity in the predictive mode of the encoder and decoder together (Table 3.4). For the practical measurements, we have measured the execution time of the motion estimation process, and averaged the results over all sequences and rate points.

Table 3.4: Comparing calculated and measured complexity of the motion estimation process shows that the complexity model is fairly accurate.

	Calculated		**Measured**	
	Encoder	Decoder	Encoder	Decoder
Predictive mode	>99.5%	<0.5%	99%	1%
DVC mode	<0.5%	46%	<0.5%	48%
Hybrid spat. mode	50%	5%	50%	8%
Hybrid subs. mode	6%	7%	7%	8%

The results given in Table 3.4 show that the model for measuring complexity using the number of pixel read/writes is reasonably accurate. Similar relationships are observed between theoretical complexity and practice. Some inaccuracies can be observed for the modes that have the lowest complexity (e.g., spatial mode, decoder-side). This is most likely due to the simplicity of the model, which does not account for differences in complexity of the different operations, the possible overhead of function calls and other programming constructs, memory behavior, etc. To this respect, it is important to remark that other implementations or platforms could require adjusting the complexity model in some way.

Armed with an accurate complexity model, we can now use these results to define the modes for the different frames in a GOP while respecting encoder and decoder complexity constraints. This will be described in the following section.

3.6 Video coding with controllable complexity

So far we have developed several modes for coding frames, with different distributions of complexity between the encoder and the decoder. The next step is to define a method for choosing how to combine these modes, in order to meet complexity constraints at encoder and decoder side. A technique minimizing total complexity is presented in Section 3.6.1. This technique is illustrated with an example in Section 3.6.2, followed by discussion in Section 3.6.3.

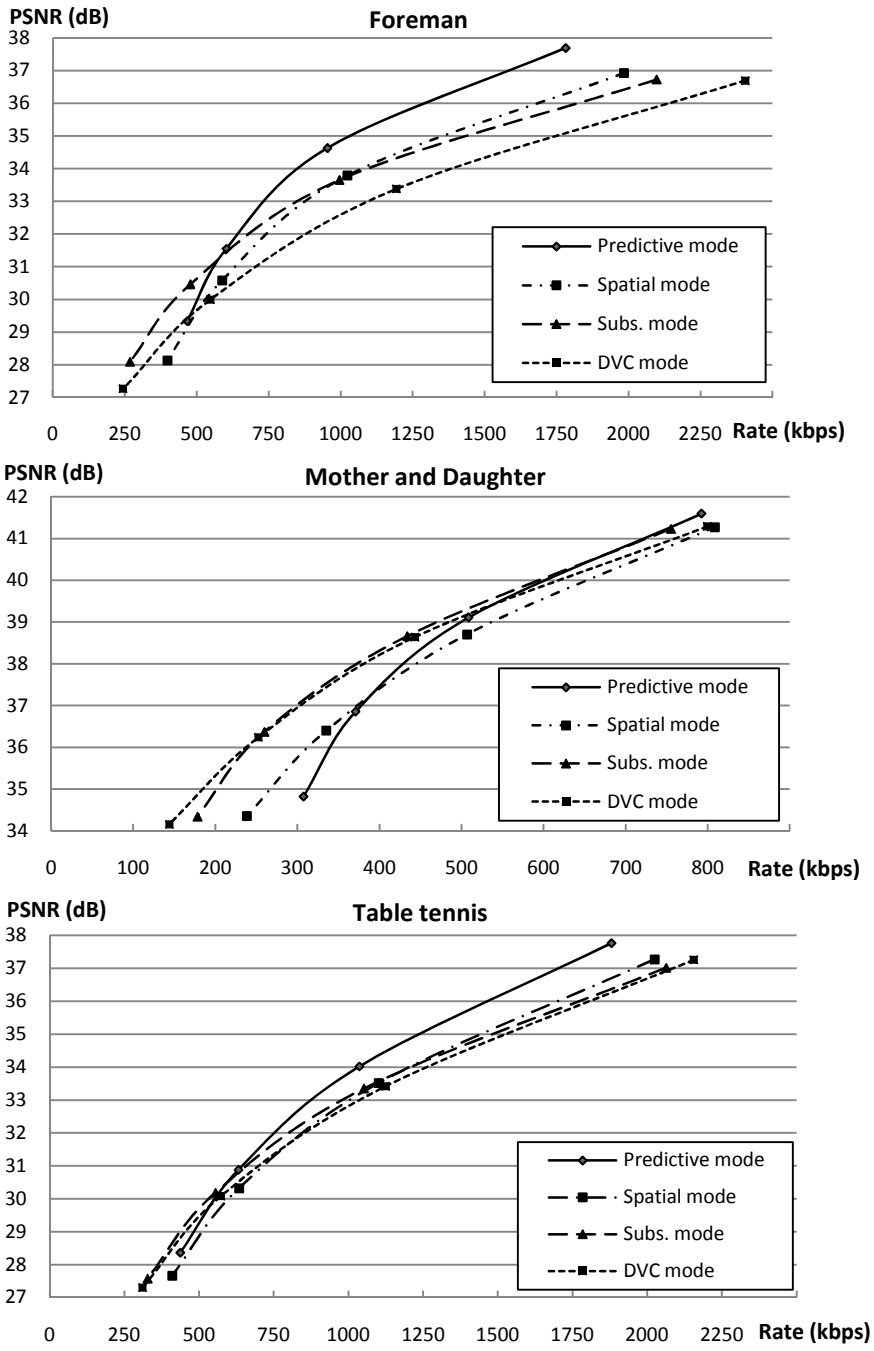

Figure 3.9: Rate-distortion plots for the different modes of the proposed codec.

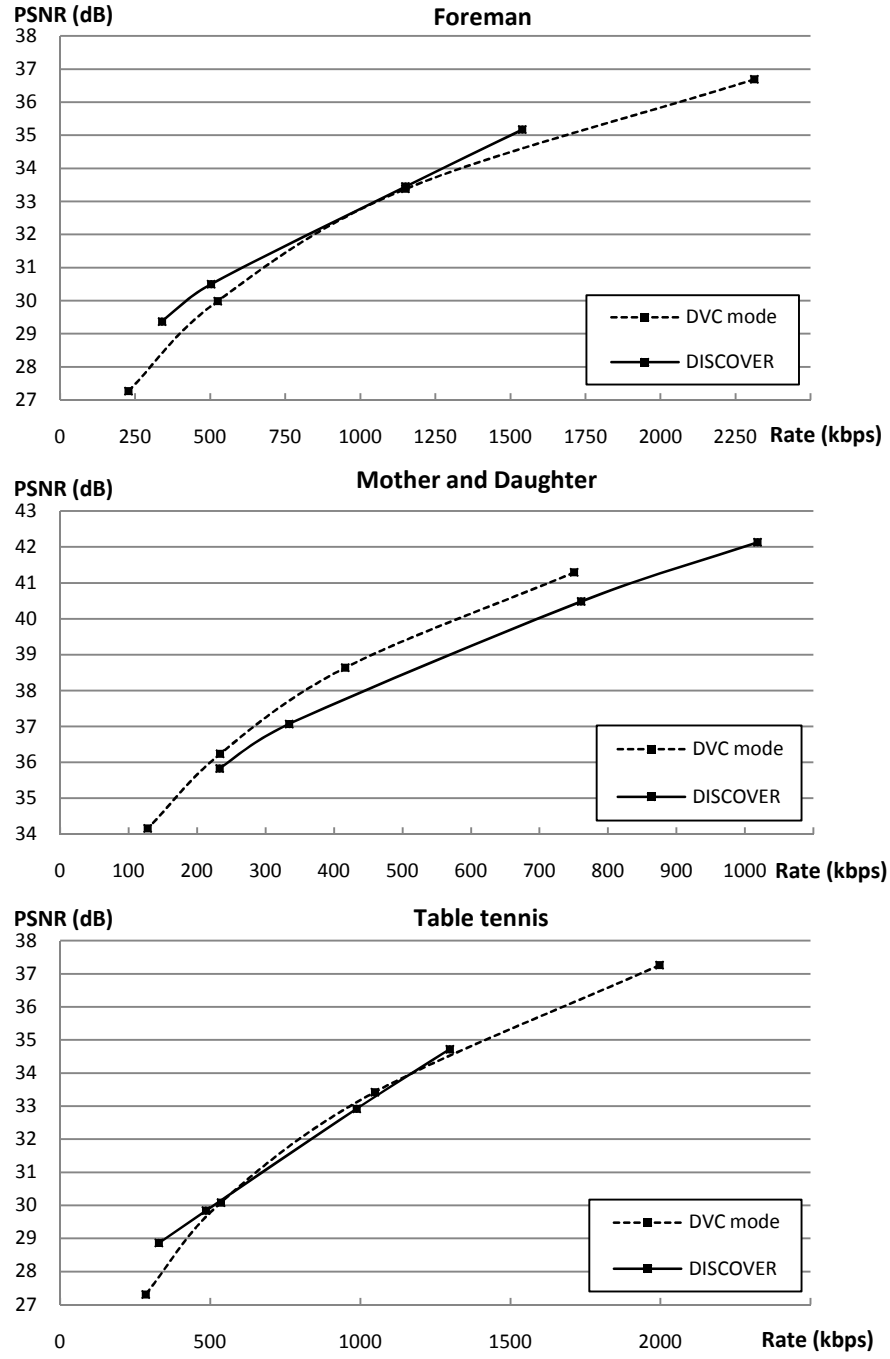

Figure 3.10: Rate-distortion plots comparing the DVC mode of the proposed codec to DISCOVER.

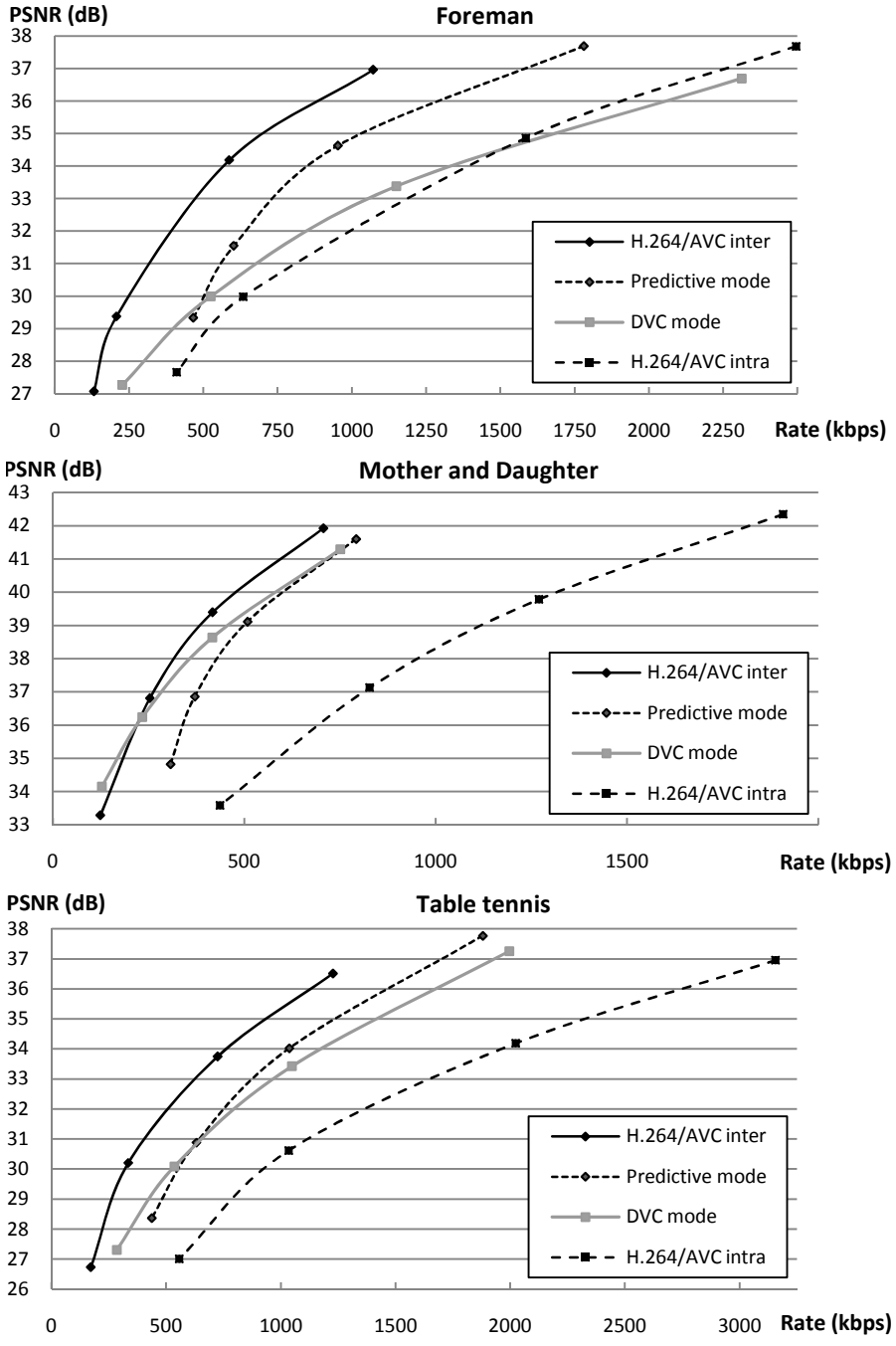

Figure 3.11: Rate-distortion results for the predictive mode compared to H.264/AVC.

3.6.1 Controlling complexity using encoder and decoder complexity constraints

Assume that techniques are available to estimate or calculate the computational resources available at encoder and decoder, and that these values are accurate for coding the following K inter frames W. Denote the available complexity per frame at the encoder (decoder) as C_E (C_D), respectively. A method will be defined to calculate the optimal linear combination of modes that meets this constraint. The method will be optimal in the sense that the total complexity of the system is minimized.

To code one frame using mode m_i (with i the mode index), a complexity budget of M_i^E is needed at the encoder, and a budget of M_i^D is needed at the decoder. From the K inter frames, α_i frames will be coded using mode i. Hence, this optimization problem can be formulated as follows:

Minimize:

$$\sum_i \alpha_i (M_i^E + M_i^D)$$

subject to:

$$\sum_i \alpha_i = K,$$

$$\sum_i \alpha_i M_i^E \leq K \cdot C_E,$$

$$\sum_i \alpha_i M_i^D \leq K \cdot C_D. \tag{3.5}$$

Remark also that one could favor encoder or decoder complexity decrease, by introducing a weighing factor δ (≥ 0) in the cost function:

$$\sum_i \alpha_i (M_i^E + \delta M_i^D). \tag{3.6}$$

This optimization problem can be solved, for example, exhaustively or by using integer linear programming techniques (ILPs) [87]. In the following section, a graphical exhaustive method will be used to find a solution.

3.6.2 Example

The previous is illustrated by means of an example for a particular set of parameters (K, C_E, C_D, and δ). While this is only one out of many configurations, similar reasoning can be used for other parameters.

Consider an example where the coding modes for the following $K = 3$ inter frames need to be decided. Using the results from the complexity analysis

performed in this chapter (Table 3.1 and Table 3.2), available complexity is expressed in terms of the number of pixel read/write operations that can be performed for coding these K frames. Assume for example that $C_E = 325 \cdot 10^6$, $C_D = 225 \cdot 10^6$, and $\delta = 1$.

Each out of K inter frames can be coded using one out of N modes, resulting into $\binom{K+N-1}{K}$ different encoder-decoder complexity distributions for coding the GOP. In this case, i.e., with $K = 3$ and $N = 4$, the binomial resolves to 20. Each of these 20 solutions can be described by the tuple $(\alpha_0, \alpha_1, \alpha_2, \alpha_3)$ indicating how the three inter frames are coded: using α_0 times the predictive mode, α_1 times the spatial mode, α_2 times the subsample mode, and α_3 times the DVC mode. Given the complexity of each mode (Table 3.1 and Table 3.2), each tuple has an associated average encoding complexity per frame $\frac{\sum_i \alpha_i M_i^E}{K}$ and an average decoding complexity of $\frac{\sum_i \alpha_i M_i^D}{K}$. These two values can be used as coordinates for representing the 20 solutions in a plane (Figure 3.12). It is easily verified that solutions featuring only two modes i and j lie on a straight line connecting the solutions where i is used exclusively and where j is used exclusively.

From these 20 points, some are suboptimal in the sense that there exists always a better way to code the GOP, i.e., with lower or at most equal encoder and decoder complexity. In Figure 3.12, these points are indicated in gray. The other points (indicated in black) are so-called pareto-optimal. These 12 pareto-optimal points provide the range for distributing complexity between encoder and decoder. We can see that $(1, 0, 0, 2)$ and $(2, 0, 0, 1)$ are not pareto-optimal, which indicates that using the hybrid modes enables more efficient distribution of complexity than combining only the DVC mode and the predictive mode. In addition, since all modes are present in the optimal set, this figure shows that no mode is redundant.

Given the encoder and decoder constraints ($C_E = 325 \cdot 10^6$, and $C_D = 225 \cdot 10^6$ respectively) illustrated by the gray rectangle, from the pareto-optimal points the point minimizing the cost function is chosen. This means that in this case all three frames should be coded using only the hybrid subsample mode $(0, 0, 3, 0)$. Other solutions satisfying the complexity constraints are suboptimal in this context, since they do not minimize the cost function.

3.6.3 Discussion

While this was only one example for a particular set of parameters and constraints, several advantages for using the proposed system in general can be identified.

A first advantage is that the system supports several complexity configu-

rations. Hence, even if the complexity constraints are static, the most suitable distribution of complexity can be determined and used throughout. Also, it provides a solution in case neither encoder nor decoder have enough resources to perform all motion estimation. This is illustrated by the previous example, where neither the predictive mode $(3, 0, 0, 0)$ nor the DVC mode $(0, 0, 0, 3)$ satisfy C_E and C_D.

Secondly, due to the fact that there is no dependency between the modes, any frame can be coded using any mode at any time. As a consequence, it is possible to adapt rapidly to varying complexity constraints imposed by devices with variable power supply, or by systems featuring multi-tasking. In case encoder complexity constraints are drastically reduced, motion estimation can be shifted to the decoder side yielding a different solution for coding the GOP (Figure 3.13). On the other hand, if decoder complexity constraints are reduced (Figure 3.14), motion estimation can be shifted to the encoder side.

The speed at which the system can adapt to varying complexity constraints, depends on a number of parameters including the network delay. The decision scheme so far assumed that both encoder and decoder complexity constraints are known. In practice, however, one needs to define such constraints based on information that is available. For example, for battery-constrained devices one could predict future power availability based on knowledge about the characteristics of the power supply (e.g. type of battery) and its current status. For multi-tasking systems one could predict the future availability of computational resources by analyzing the current status of each of the processes, and exploiting knowledge about their expected behavior. Such an analysis would then result in the definition of complexity constraints, both at the encoder and at the decoder. After complexity constraints have been defined, the decoder would need to send this information to the encoder so that it can decide upon the modes that need to be used for coding the following frames. The fact that any frame can be coded using any mode at any time is a considerable advantage for our system, since it allows the encoder to immediately incorporate the results of the mode decision process. This way, the updated complexity distribution can be made effective immediately at the encoder. At the decoder, complexity will change only after receiving the corresponding frames. This means that at the decoder there is a latency of at least two times the network delay between signaling constraints and adapting to them.

Remark also that this example assumed a very short time during which encoder and decoder complexity constraints remain constant. In practice, however, complexity constraints are likely to be constant over a larger period of time. As K becomes larger, distributing complexity can be performed more subtle, since the pareto-optimal set will contain more solutions. This is illus-

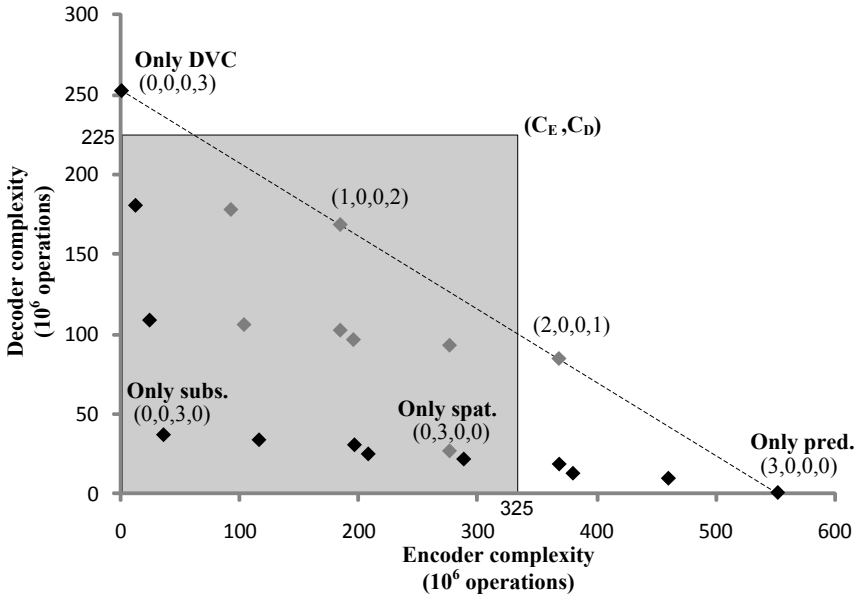

Figure 3.12: Encoder and decoder complexity constraints define a set of possible combinations for coding the GOP (indicated by the gray rectangle). In this case, coding all three inter frames using the subsample mode is optimal.

trated by Figure 3.15, where $K = 10$ results into 40 pareto-optimal solutions, allowing fine-grain adaptivity.

It should be remarked that the decision scheme here presented does not consider the differences in RD performance between the modes. Ideally, one would like all modes to have the same performance, but, from the results' section, this does not always seem to be the case. Hence, instead of selecting the point with the lowest total complexity, one could trade-off rate, distortion and complexity at the same time. This topic is left as future work, as discussed at the end of this chapter.

3.7 Recent developments by other researchers

Other related systems to the system presented in this chapter have been described in the literature, simultaneously or after some of the results in this chapter have been published. These systems will be discussed in this section.

In his PhD thesis, Belkoura [88] describes how to distribute complexity between the encoder and the decoder dynamically, by combining an H.264/AVC codec and a WZ codec. Both codecs are independent from each other, and

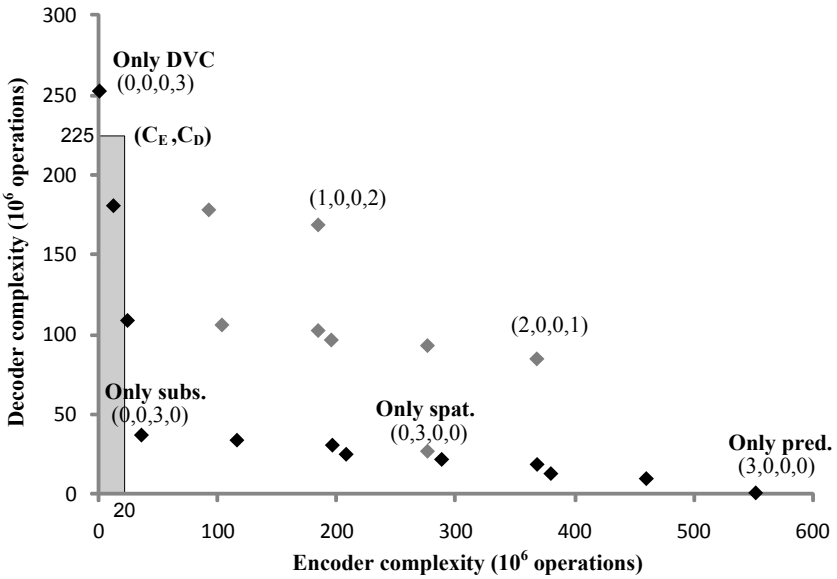

Figure 3.13: If encoder complexity constraints are reduced, motion estimation complexity is shifted to the decoder side, yielding $(0, 0, 1, 2)$ as the optimal solution.

complexity is shared in the temporal sense by switching between both codecs. In the context of this chapter, this corresponds to using the predictive mode and the DVC mode only for distributing complexity.

While our research has been limited to the complexity of the motion estimation algorithms, Belkoura also studies the complexity of the turbo decoder. His analysis shows that the complexity of the turbo decoder decreases with the number of WZ bits used for decoding. A significant reduction in turbo decoding complexity can be achieved if more than the minimal amount of WZ bits are received by the decoder. As such, he shows that complexity can also be controlled by adapting the WZ bit rate.

A related preliminary study using our system revealed that the complexity of the turbo decoding process varies between the modes as well. This is due to the fact that the number of decoding iterations that need to be executed by the turbo decoder, depends on the quality of Y_R compared to the original R available at the encoder. When shifting the motion estimation from the encoder to the decoder, the quality of Y_R will typically decrease due to the absence of the original at the decoder. As a result, less bitplanes will be skipped by the turbo decoder and more decoding iterations will be needed per bitplane, as illustrated in Table 3.5. To avoid evaluating a possibly non-optimal rate request strategy,

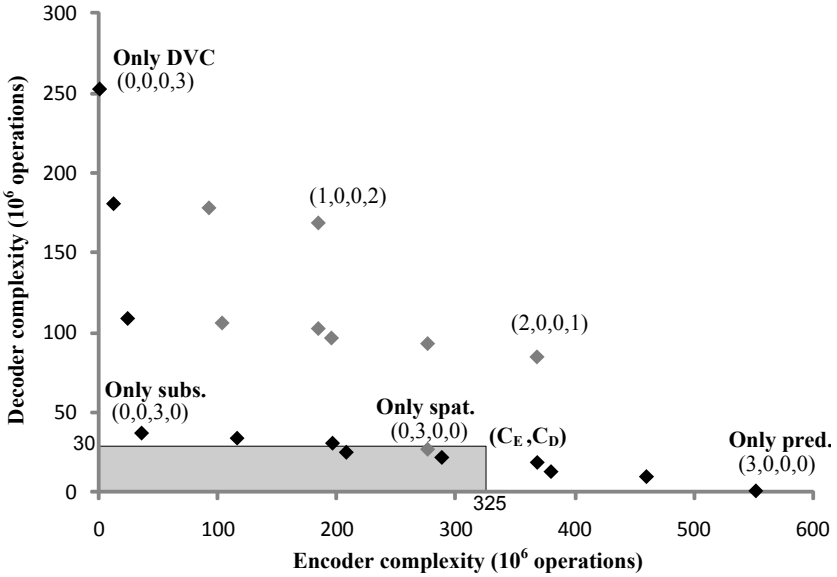

Figure 3.14: If decoder complexity constraints are reduced, motion estimation complexity is shifted to the encoder side, yielding $(1, 0, 2, 0)$ as the optimal solution.

only the final decoding pass is considered (i.e. when the correct number of WZ bits is received from the encoder).

This means that, when going from the predictive mode to the DVC mode over the hybrid modes, the complexity of the decoder is not only increased by motion estimation, but also by turbo decoding. To improve the accuracy of the complexity model, it could therefore be necessary to incorporate turbo coding. This is left as a topic for future work.

After two papers of the work in this chapter had been published [75, 76], Chen and Steinbach [89] published a conference paper describing similar ideas for implementing hybrid modes for coding frames. More specifically, they

Table 3.5: Average number of turbo decoding iterations per bitplane, for the Foreman sequence under the same test conditions as described in the results section.

	Pred. mode	Hybrid subs.	Hybrid spat.	DVC mode
Q_5	0.2	0.8	1.3	1.7
Q_4	1.6	2.9	3.1	4.2
Q_3	4.0	6.1	5.8	7.6
Q_2	8.0	9.9	9.7	11.5

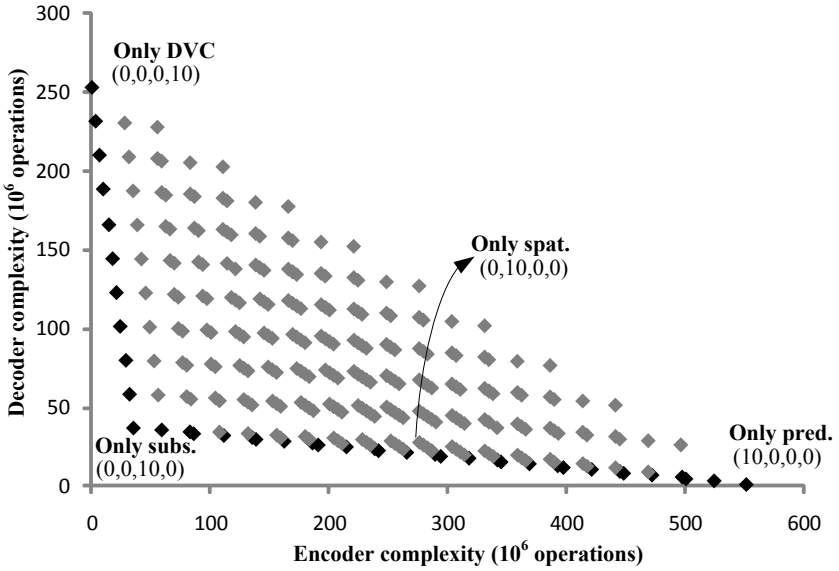

Figure 3.15: Complexity can be distributed more subtle if K is large. For example, $K = 10$ results in 40 points being part of the pareto-optimal set.

propose a combination of spatial partitioning and coarse encoder-side motion estimation with decoder-side refinement.

At the encoder, each block is compared to the colocated block in the previous frame, using the SSE as a metric. All SSEs are listed for all blocks in the frame, and the N blocks with the highest SSE are selected for motion estimation at the encoder. Coarse motion estimation is performed for these blocks, using a 2D logarithmic search [90]. The obtained motion vectors are coded using Huffman coding, and sent to the decoder. It is unclear, however, how the encoder indicates the blocks for which coarse motion estimation has been performed.

The decoder decodes the motion vectors and refines them using bilateral motion estimation [91], combined with a spatial smoothing step similar to the one used in this chapter [54]. If no motion vector has been received for a certain block, decoder-side motion estimation is performed using the same techniques but with larger search windows.

The ratio of blocks that need to be predicted at the encoder is used as a parameter to illustrate flexible distribution of complexity. A switching scheme is however not defined.

Extending the work of Chen and Steinbach, and our work, Forte and Srivastava recently defined a scheme for distributing complexity based on energy

and thermal constraints [92]. The complexity balance between the encoder and the decoder is expressed through a parameter α, where the extremes $\alpha = 0$ and $\alpha = 1$ result in strictly DVC and predictive coding respectively. For other values of α (i.e., $0 < \alpha < 1$), motion estimation is shared between the encoder and the decoder by restricting the search space in a way that depends on α. Specifically, instead of doing a full search, the encoder only performs a portion of the raster scan while the decoder analyzes the remaining positions.

A complexity controller is deployed at the encoder. This component dynamically chooses the α that minimizes the energy spent for the harder working coder side while maintaining thermally safe operating conditions. To be able to solve this constrained optimization problem, energy and thermal models are established.

To this extent, the encoder's and decoder's energy consumption is modeled for the different components of the system. The energy spent by some of these components is a function of α (e.g., the component performing motion estimation, parity bit transmission, etc.), and offline analysis is used to obtain approximations of the behavior of the system as a function of α. Next, for a particular α, the energy models can be used to describe the power dissipation at the encoder and at the decoder as a function of time. In turn, given the current operating temperatures at the encoder and the decoder, we can use this to estimate the future operating temperatures. As such, the optimization problem can be solved by evaluating future operating temperatures and energy consumption profiles for different α's.

The results indicate that such a flexible system – similar to the one described in our work – shows a better compromise between energy and temperature as it is better able to adapt.

3.8 Conclusions, future work, and original contributions

The system presented in this chapter allows distributing the complex task of motion estimation between the encoder and the decoder in a flexible way. While typical RDC optimized systems only allow encoder and/or decoder complexity to be reduced, this system allows shifting motion estimation workload from the encoder to the decoder, and vice versa. As a result, this type of codec is extremely useful in cases where available computational complexity is non-static, for example, in the case of multi-tasking systems or battery-constrained devices.

While other researchers have started to propose alternative solutions or

improvements, we still see much room for further analysis. One of the main problems is that the compression performance of the different modes is still quite low compared to solutions such as H.264/AVC. While the proposed system has the capability of distributing complexity – in contrast to H.264/AVC – improving its compression performance is still desirable to make it competitive. Some of the modes, such as the predictive mode, can be easily extended by borrowing techniques from H.264/AVC. Other modes, such as the DVC mode, require new research since they already represent the state-of-the-art.

Another topic requiring additional research is the mode decision strategy. The coding mode for each inter frame is now decided by minimizing the total complexity of the system, while respecting encoder and decoder complexity constraints. Due to the differences in rate-distortion performance of the different modes, an improved strategy would be to define a rate-distortion-complexity (RDC) optimization technique. For example, instead of selecting the pareto-optimal point that minimizes the total complexity for coding the GOP, the point corresponding to the best relation between complexity and compression performance could be chosen. The technique proposed by Forte and Srivastava [92] can also be extended in this context.

RDC optimization requires developing RDC models for each of the modes. This effort is probably better postponed until after we are pleased with the performance of each of the modes. Therefore, the main focus of the remainder of this dissertation is on the weakest link, i.e., the DVC mode. Improving the compression performance of this mode is advantageous for DVC use case scenarios, and the results can be fed back to architectures such as the system proposed in this chapter. Other architectures might benefit as well by new insights in DVC. For example, as shown in collaborative work, H.264/AVC can be extended with DVC-like coding modes, leading to improved compression performance [93].

The author's work on flexible distribution of complexity has led to the following publications:

- Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Flexible distribution of complexity by hybrid predictive-distributed video coding. *Signal Processing: Image Communication*, 25(2):94–110, February 2010.

- Peter Lambert, Stefaan Mys, Jozef Škorupa, Jürgen Slowack, Rik Van de Walle, and Christos Grecos. Distributed video coding for video communication on mobile devices and sensors. In *Handheld computing for mobile commerce: applications, concepts and technologies*, pages 375–402. Information Science Publishing, 2010.

- Jozef Škorupa, Stefaan Mys, Jürgen Slowack, Peter Lambert, and Rik Van de Walle. Heuristic dynamic complexity coding. In *Proc. SPIE*, volume 7000, pages 70000G–70000G–8, April 2008.

- Stefaan Mys, Jürgen Slowack, Jozef Skorupa, Peter Lambert, and Rik Van de Walle. Motion compensated interpolation as a new inter coding mode for 8x8 macroblock partitions in H.264/AVC B slices. In *Lecture Notes in Computer Science*, volume 5353, pages 168–177. Springer Verlag, 2008.

- Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Dynamic complexity coding: Combining predictive and Distributed Video Coding. In *Proc. Picture Coding Symposium (PCS)*, November 2007.

Chapter 4

Modeling the correlation channel

4.1 Introduction

Estimating the correlation between the original frame X (at the encoder) and the side information Y (at the decoder) is a difficult problem in DVC. Usually, this correlation is expressed through the difference $N = X - Y$, referred to as the *correlation noise*. Modeling the distribution of N is difficult, due to non-stationary properties of the noise. This is illustrated by a number of examples. In these examples, the side information Y is generated using the techniques adopted in DISCOVER [51].

A first example comes from the Football sequence. This sequence features a lot of motion, making the generation of Y difficult, as illustrated in Figure 4.1. One can observe that N is spatially non-stationary, meaning that different regions have different error distributions. Some regions are well predicted (such as the grass in the background), while the prediction for other regions is poor. Hence, when estimating the distribution of N, one should take into account the fact that the noise is spatially non-stationary. Apart from spatially varying statistics, also temporal non-stationarity can be observed. This is illustrated by two examples taken from the Foreman sequence, provided in Figure 4.2 and Figure 4.3. Comparing the quality of the side information for both examples illustrates that a different amount of errors is observed for frames at different time instances. This also suggests differences between sequences, which is illustrated by a third example from Akiyo in Figure 4.4. Comparing these results with other sequences such as Football clearly illustrates the point.

While accurate correlation modeling is difficult due to temporal and spatial non-stationarity of the correlation noise N, having accurate information about

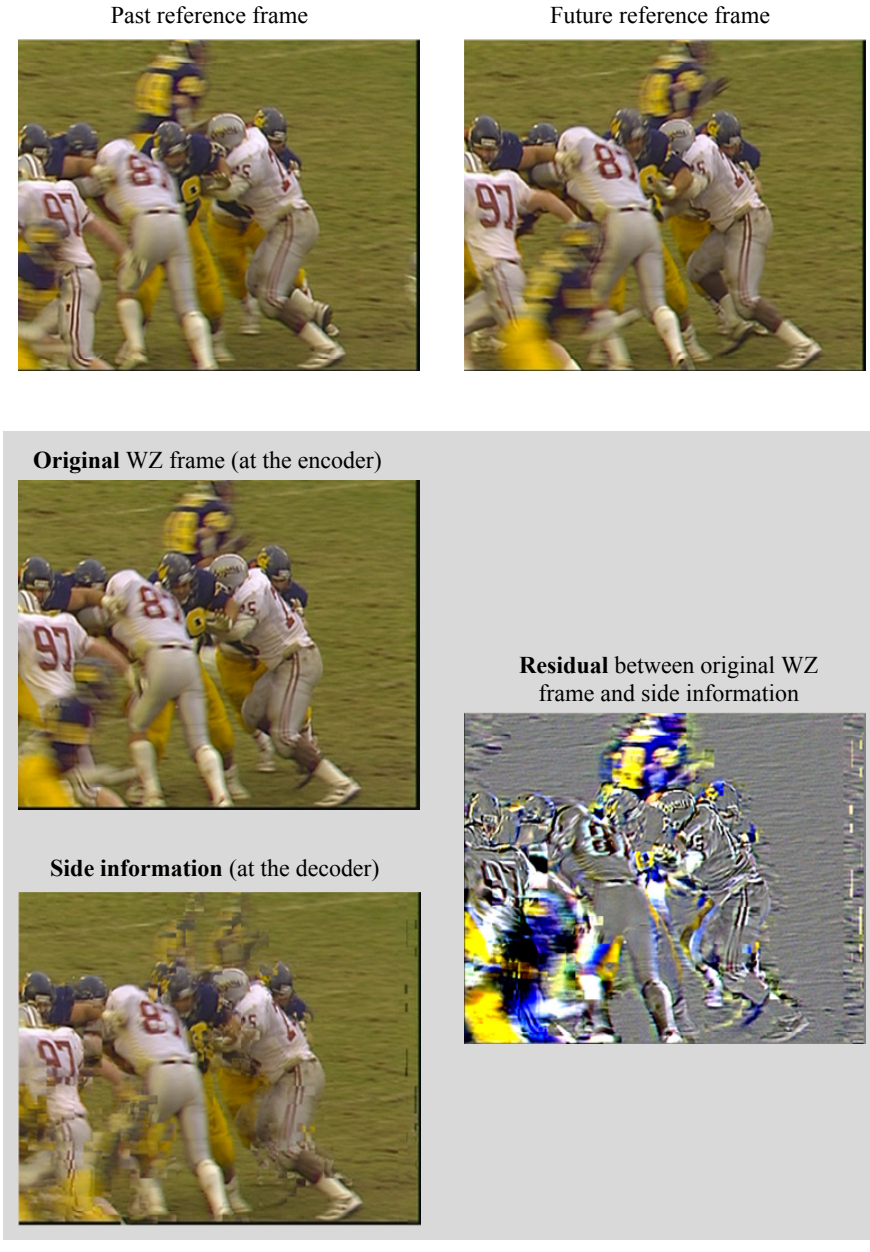

Figure 4.1: Illustration of the correlation noise for the Football sequence, frame index 40 (past frame index 39, future frame index 41).

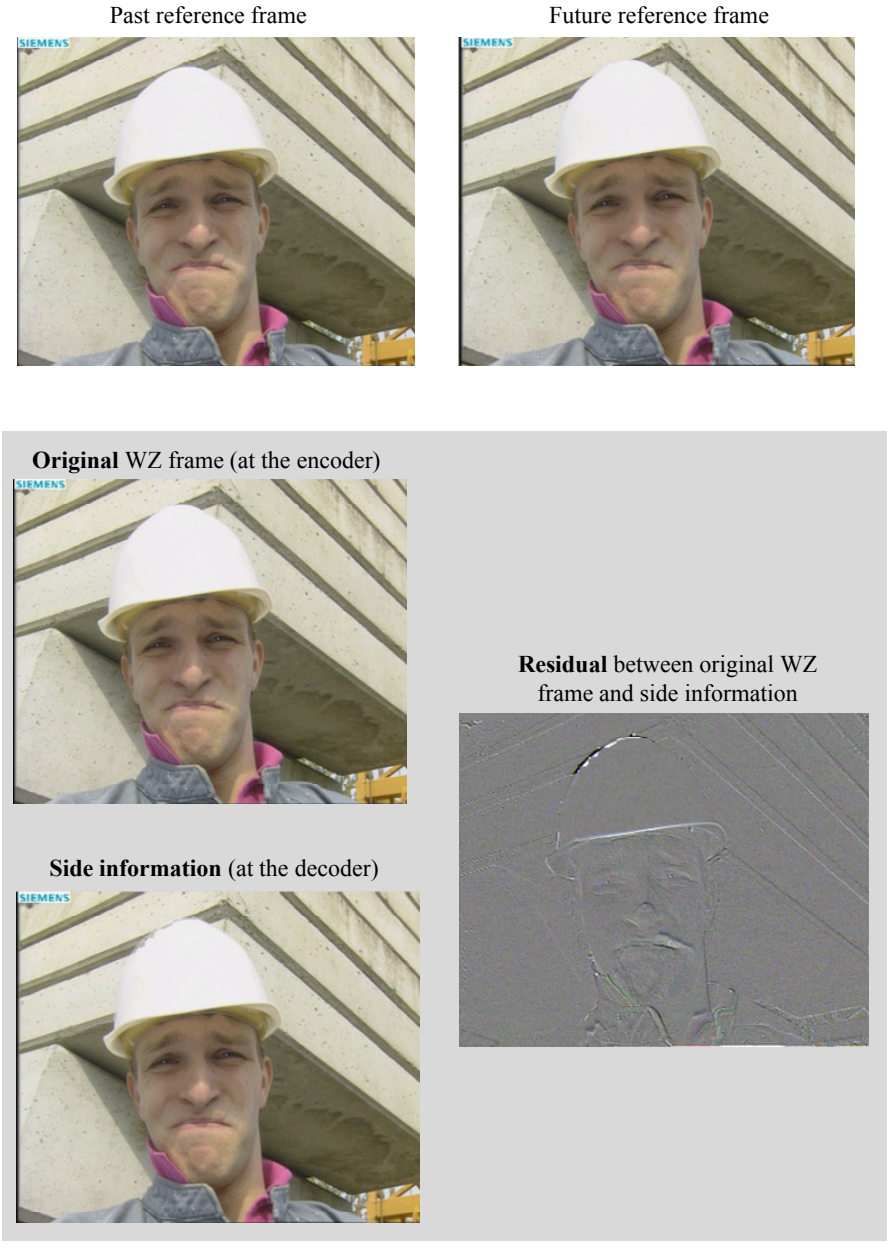

Figure 4.2: Illustration of the correlation noise for the Foreman sequence, frame index 38 (past frame index 37, future frame index 39).

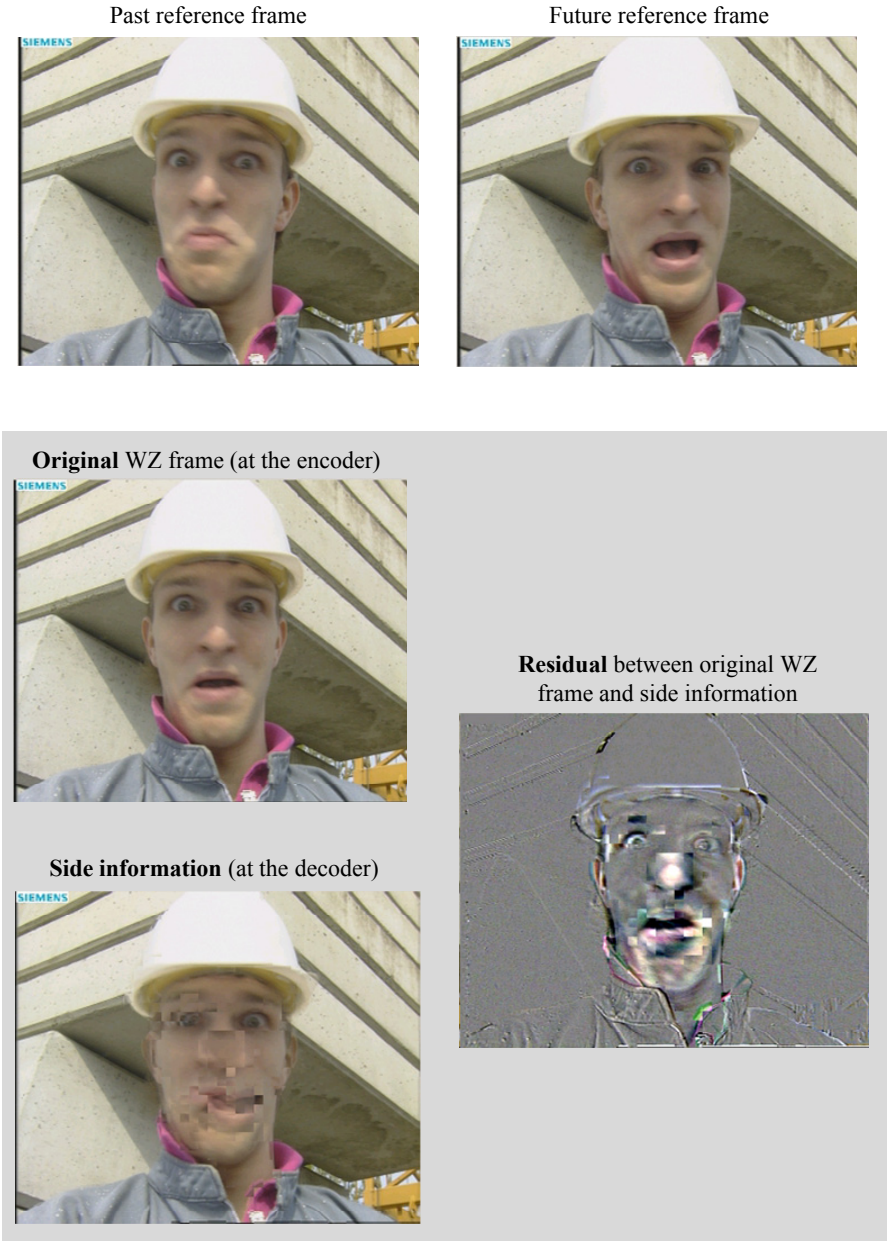

Figure 4.3: Illustration of the correlation noise for the Foreman sequence, frame index 94 (past frame index 93, future frame index 95).

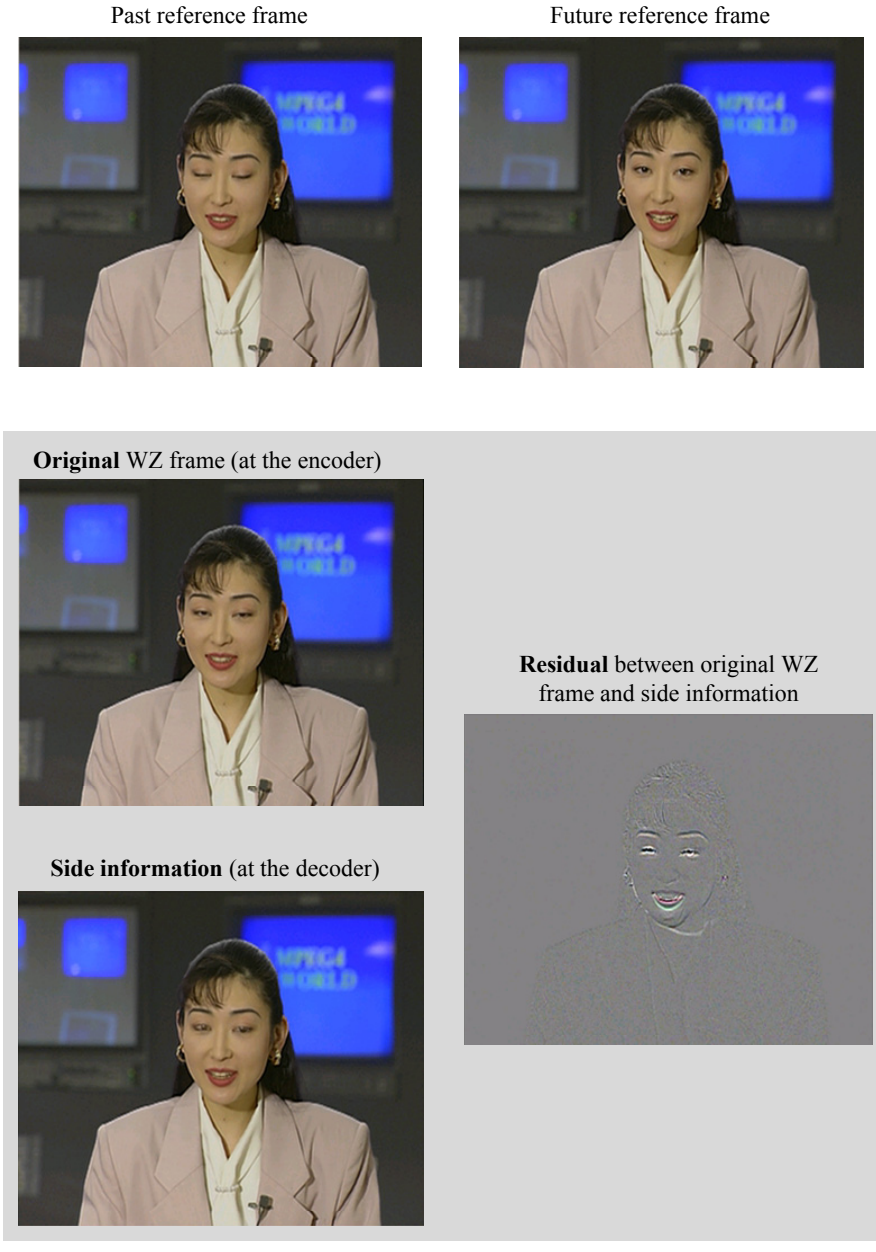

Figure 4.4: Illustration of the correlation noise for the Akiyo sequence, frame index 72 (past frame index 71, future frame index 73).

the correlation noise is of critical importance both at the encoder and at the decoder.

At the encoder, knowledge about the correlation between X and Y can be used to estimate the number of bits that the decoder will need for decoding. Little correlation between X and Y means that the decoder has a poor prediction, and therefore, more parity bits will be required for reliable decoding. On the other hand, if the correlation between X and Y is high, less bits are needed. Hence, if the encoder has knowledge about the correlation between X and Y, it can estimate/calculate the number of parity bits needed by the decoder, hereby eliminating the need for a feedback channel from decoder to encoder. This is a major practical advantage, but it comes at the cost of increased encoder complexity, since the encoder needs to estimate the correlation between X and Y. In addition, inaccuracies in this estimation often result in a bit rate overhead. More information about feedback-free DVC systems can be found in the literature [44–47].

At the decoder, knowledge about the correlation is needed for high-performance decoding (e.g., turbo or LDPC decoding). Typically, when receiving parity bits, these channel decoders update statistics about the reliability of each bit and the reliability of an output sequence. If sufficient reliability is achieved, channel decoding is terminated and the most likely output is delivered. In these calculations, the correlation model is used as reliability information by the channel decoder. As a result, the correlation model has an impact on the efficiency of the decoding process, and inaccuracies in the correlation model result in a bit rate penalty. When channel decoding terminates, for each coefficient, the most likely quantization bin is returned. Next, the most likely value in this quantization bin is selected by the reconstruction module, resulting in the final decoded coefficient. This decision is often based on the correlation model as well [59], so that the correlation model impacts both bit rate and distortion.

In this chapter we focus on decoder-side correlation noise estimation, since it improves compression performance without increasing the complexity at the encoder-side.

Techniques for correlation noise estimation have been proposed frequently in the literature. Initially, stationary models and/or offline approaches have been used. For example, in the Stanford DVC system, parameters for each coefficient band are obtained through an initial training stage performed on several sequences [40–42]. A stationary model is adopted as the same parameter set is used throughout the entire sequence. A different offline technique allows access to the original WZ frame for calculating the correlation distribution parameters [94]. Variants on both approaches have been discussed by

Meyer et al. [95,96], illustrating the non-stationary properties of the correlation noise in the specific case of occlusion.

While such techniques are impractical or lack flexibility, efficient online techniques have been proposed only recently. An important contribution in this context is due to Brites and Pereira [79]. In their contribution, both offline and online methods are proposed, in the pixel-domain as well as in the transform-domain. Since each block in the side information is generated as the average between a past reference block and a future reference block, the similarity between these two reference blocks is used to estimate the correlation between the original X and its estimation Y. If the side information generation process is unable to find good matches between past and future reference blocks, then X and Y is assumed poorly correlated. On the other hand, if there is a good match, then the correlation between X and Y is assumed strong. As such, this technique allows online estimation of the correlation noise by analyzing the similarities between past and future motion-compensated frames.

An alternative solution has been proposed in the context of this dissertation, in co-authorship with Jozef Škorupa and others [97]. This technique – described further on in detail – also uses the motion-compensated residual between the past and future reference frames for estimating the correlation noise. However, one of the major differences with the previous technique is that the transform-domain noise is estimated by converting the pixel-domain noise estimates. Results show increased performance compared to the work of Brites and Pereira [79].

Converting pixel-domain correlation noise estimates to their transform-domain equivalents has been proposed as well by Fan et al. [98]. Here, instead of using the motion-compensated residual, they exploit information available from the previously decoded WZ frame, as well as the previously decoded coefficient bands.

Information from decoded coefficient bands is used also by Huang and Forchhammer [99], in order to improve the method proposed by Brites and Pereira [79]. The decoded information is used to classify coefficients, applying different estimators for different categories.

In Section 4.2 we first comment on the method proposed by Brites and Pereira [79], as this contribution can be considered one of the first in the context of online correlation noise modeling in DVC. Next, the alternative solution of Škorupa et al. [97] is discussed in detail, as this method has been extended in the context of this dissertation.

Two contributions are presented in this chapter. A first contribution, described in Section 4.3, refines the method of Škorupa et al. [97] by compensating for inaccuracies due to quantization noise. This improvement realizes a re-

duction of the WZ bit rate up to 19.5%. A second contribution compensates for complex motion by introducing multiple predictors, as shown in Section 4.4. This extension leads to additional bit rate gains up to 8%.

4.2 Online transform-domain correlation noise estimation

The correlation between X and Y is commonly modeled as a Laplacian distribution (e.g., [40, 59, 79]) or a Gaussian distribution (e.g., [100]). The Laplacian distribution is more often used to model the correlation, as a good trade-off between model accuracy and complexity. The use of the Laplacian was validated by Brites et al. [79], through a chi-square goodness-of-fit test, delivering chi-square values of 0.12 for Foreman and 0.04 for Hall Monitor.

Using the Laplace distribution, the correlation between X and Y is described through the conditional probability density function, i.e.:

$$f_{X|Y}(x|y) = \frac{\alpha}{2} e^{-\alpha|x-y|}. \tag{4.1}$$

At the decoder, the realization of the side information, y, is available, and so only the distribution scale parameter α needs to be estimated. The relation between α and the variance σ^2 is given by:

$$\alpha = \frac{\sqrt{2}}{\sigma}. \tag{4.2}$$

4.2.1 Using the motion-compensated residual (Brites and Pereira)

An interesting description of offline and online estimation techniques has been provided by Brites and Pereira [79]. Both pixel-domain as well as transform-domain techniques are proposed, with different granularities for estimating α. The best performing method is the online transform-domain technique at coefficient/frame level. This technique estimates one α for each transformation coefficient in the WZ frame, as follows.

First, the residual frame R is generated by using the motion vectors applied for generating the side information. More precisely, each residual pixel at location (x, y) is calculated as

$$R(x,y) = \frac{X'^F(x + dx_F, y + dy_F) - X'^B(x + dx_B, y + dy_B)}{2} \tag{4.3}$$

where X'^F and X'^B denote the forward (future) and backward (past) reference frame, and (dx_F,dy_F) and (dx_B,dy_B) denote the associated forward and backward motion vector, respectively.

This motion-compensated residual R is used for estimating N. The intuition behind this approach is that if the side information generation process is unable to find good matches between past and future reference blocks, then R will be large. In such cases, the accuracy of the prediction will typically be low, so using a large variance for N seems appropriate. On the other hand, if a good match is found, then it seems more suitable to use a smaller variance. Hence, there is a relation between R and N that can be exploited. Since R is available at the decoder, N can be estimated online, i.e., after side information generation.

Using R, the estimation of α proceeds as follows. Each 4-by-4 block in R is transformed by a DCT, delivering the transformed frame T. Next, all coefficients in T are replaced by their absolute values, leading to $|T|$. The variance of the b-th coefficient band $|T|_b$ in $|T|$ is calculated, using:

$$\hat{\sigma}_b^2 = E_b[(|T|_b)^2] - (E_b[|T|_b])^2, \tag{4.4}$$

where E_b denotes the expectation taken over the coefficient band $|T|_b$. The band average value is denoted $\hat{\mu}_b$.

Finally, the α parameter for the DCT coefficient at position (u, v), in the b-th coefficient band, is estimated by:

$$\hat{\alpha}_b(u,v) = \begin{cases} \sqrt{\frac{2}{\hat{\sigma}_b^2}}, & [D_b(u,v)]^2 \leq \hat{\sigma}_b^2 \\ \sqrt{\frac{2}{[D_b(u,v)]^2}}, & [D_b(u,v)]^2 > \hat{\sigma}_b^2, \end{cases} \tag{4.5}$$

where $D_b(u, v)$ given by:

$$D_b(u,v) = |T|_b(u,v) - \hat{\mu}_b. \tag{4.6}$$

This technique yields different values for α at different spatial and temporal positions. This is important due to the spatial and temporal non-stationarity of N, as discussed in the introduction of this chapter.

4.2.2 From pixel-domain to transform-domain (Škorupa et al.)

An alternative transform-domain technique has been proposed in the context of this dissertation, in co-authorship with Jozef Škorupa and others [97]. This technique is based on the same logic, in the sense that it also estimates the distribution of N using the motion-compensated residual R. However, better performance is achieved compared to the previous method proposed by Brites and Pereira [79]. Using this technique, the α parameter is first estimated in the pixel domain. Next, for transform-domain systems, the α parameter is transformed in order to find its transform-domain correspondence. The remainder

of this section will describe this method in detail, as it is extended further in this chapter.

In the pixel domain, the α parameter is calculated as follows. First, the residual R is (re)defined as:

$$R(x,y) = X'^{B}(x+dx_B, y+dy_B) - X'^{F}(x+dx_F, y+dy_F). \tag{4.7}$$

For each block at index k, the central moment is calculated by:

$$Mom_k(R) = E_k\left[|R(x,y)|^{0.5}\right], \tag{4.8}$$

with E_k denoting the expectation taken over the block at index k only. Next, the average central moment $Mom(R)$ is obtained through expectation over the entire residual frame R:

$$Mom(R) = E\left[|R(x,y)|^{0.5}\right]. \tag{4.9}$$

The central moment of the correlation noise N for the block at index k in the WZ frame is then estimated as:

$$\widetilde{Mom_k}(N) = \frac{Mom_k(R) + Mom(R)}{2}. \tag{4.10}$$

The rationale behind this formula is that it estimates N by combining local and global information from R. The authors are motivated to use the central moment through the observation that the noise distribution is better fitted with this method [97].

Finally, for each pixel in the WZ block at index k, the following α is used as an estimation:

$$\alpha_k^P = \frac{\pi}{4\widetilde{Mom_k}(N)^2}, \tag{4.11}$$

where the upper-index P indicates that this α parameter is defined in the pixel-domain.

The pixel-domain α parameter can be converted to a transform-domain version by using a scaling step. As such, the α parameter of the coefficient at index (i,j) in block k is given by:

$$\alpha_{k,(i,j)}^T = \frac{\alpha_k^P}{\sqrt{s_{i,j}}}, \tag{4.12}$$

with $s_{i,j}$ defined as in Table 4.1 ($0 \leq i \leq 3$ and $0 \leq j \leq 3$). The use of the square root in Eq. 4.12 is explained by the fact that the scaling factors $s_{i,j}$ have been obtained for the distribution's variance.

Table 4.1: Scaling parameters $s_{i,j}$ used for pixel to transform-domain conversion of the α parameter estimation, following the techniques proposed by Škorupa et al. [97].

$s_{i,j}$	j			
i	4.25	2.06	1.16	0.77
	2.06	1.00	0.56	0.38
	1.16	0.56	0.32	0.21
	0.77	0.38	0.21	0.14

For future reference in the extensions provided in this chapter, define the average pixel-domain α (with $\widetilde{Mom(N)}$ the coefficient band average of $\widetilde{Mom_k(N)}$):

$$\alpha^P = \frac{\pi}{4\widetilde{Mom(N)}^2}, \tag{4.13}$$

and its transform-domain counterpart:

$$\alpha^T_{(i,j)} = \frac{\alpha^P}{\sqrt{s_{i,j}}}. \tag{4.14}$$

4.3 Accounting for quantization noise

This chapter presents two contributions in the context of online correlation noise estimation in DVC. A first contribution, discussed in this section, extends the previous method explained in Section 4.2.2, by accounting for quantization noise in the reference frames used for generating the side information.

The technique is based on the observation that quantization noise in the decoded reference frames X'_B and X'_F has a different impact on R than it has on N. As a result, inaccuracies occur in the previous method for correlation noise estimation, especially when the quantization noise is high (i.e., for coarse quantization). This is illustrated using the results from coding frame 93 of Foreman (I-WZ-I-WZ GOP structure). Any other frame could have been used to illustrate this point as well.

For fine quantization (Figure 4.5), the residual R, defined by Eq. 4.7, and the correlation noise N show resemblance. Areas that are well predicted also show low (i.e., mid-gray) values for R. On the other hand, R also provides reasonably accurate information about the average mismatch in areas that are poorly predicted. Hence, R can indeed be used to estimate N.

However, as shown in Fig. 4.6, at coarse quantization of the intra-frames, there is a mismatch between R and N. The distribution of N has a much larger

variance than the distribution of R. This is a consequence of the quantization noise present in the reference frames. Due to this noise, some of the fine detail is lost in the past and future reference frame. However, the side information generation process is still able to find good matches between past and future blocks, as the details have been erased in a similar way in both frames. As a result, the residual R is typically low, and the side information Y that is constructed through interpolation does not contain some of the fine detail either. For example, when comparing the original frame and the side information in Figure 4.6, one can observe that some of the texture of the concrete in the background is lost. As a result, the lost texture is found in N, but not in R, which results in higher energy for N compared to the energy in R.

The current technique compensates insufficiently for quantization noise. To illustrate this, measurements have been performed for the luma DC component of Foreman (CIF, first 101 frames, I-WZ-I-WZ GOP structure). The distribution of N measured offline is compared against the average noise distribution estimated online using the method described in Section 4.2.2. The results are presented in Figure 4.7, including the (offline measured) distribution of R. These results show that – for coarse quantization (i.e., IQP 40) – there is a clear mismatch between the measured distribution of N and the estimated distribution.

4.3.1 Proposed solution

As a solution to this problem, quantization information is calculated at the encoder. This information is sent to the decoder, where it is used to improve the estimation of N.

For high resolution quantization, i.e., when the width of the quantization bin is small compared to the variance of the distribution of the signal, the average distortion due to uniform quantization can be approximated by the distortion of a random variable that is uniformly distributed over the quantization bin, which has a variance of $d^2/12$, where d denotes the bin width [101]. In our case, this approximation is too inaccurate since low rates (high quantization noise) are specifically targeted. At low rates, the quantization noise depends on the distribution of the signal and hence it is sequence dependent, and non-stationary in time (and space[1]). Therefore, the quantization noise of the intra frames is calculated at the encoder (Section A.) and this information is used at the decoder to improve the estimation of the correlation noise (Sec-

[1] In the technique proposed here, the average quantization noise will be calculated per frame, hereby ignoring spatial differences. Accounting for spatial differences comes at the cost of increased signaling overhead.

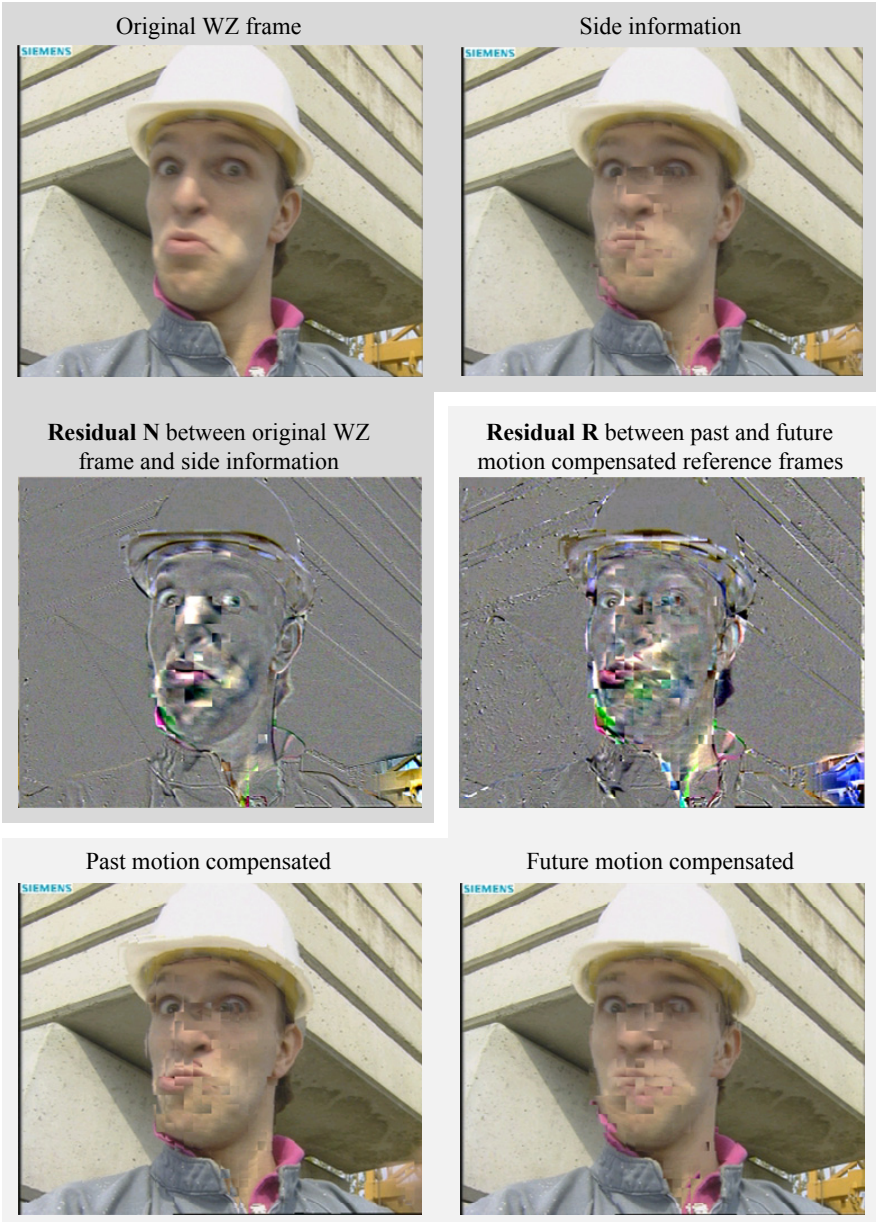

Figure 4.5: Correlation noise N compared to the motion compensated residual R, for fine quantization, i.e., an intra quantization parameter (IQP) of 10 for the reference frames. In the case of fine quantization, R is reasonably well comparable to N.

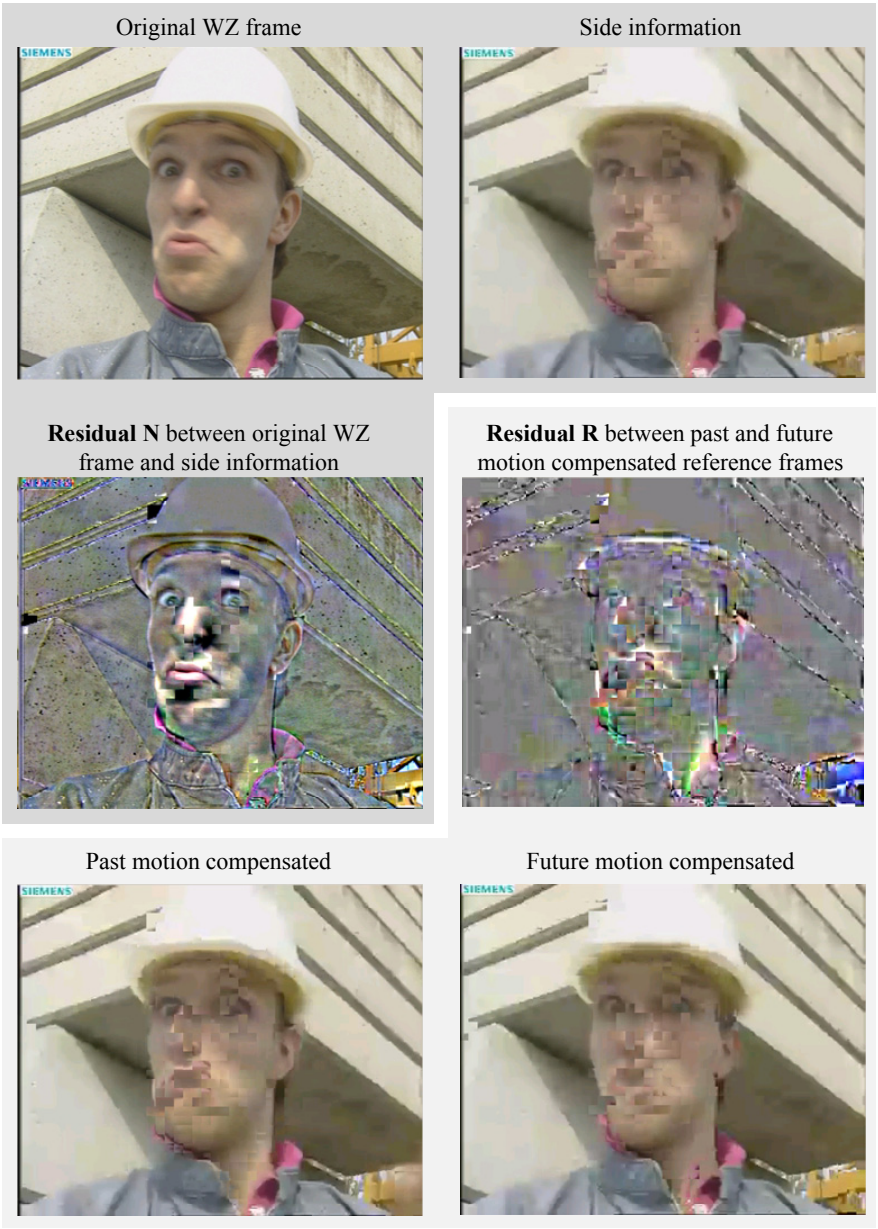

Figure 4.6: Correlation noise N compared to the motion compensated residual R, for coarse quantization, i.e., an intra quantization parameter (IQP) of 40 for the reference frames. In this case, some of the residual texture in N is not present in R, since quantization has removed this information from the reference frames.

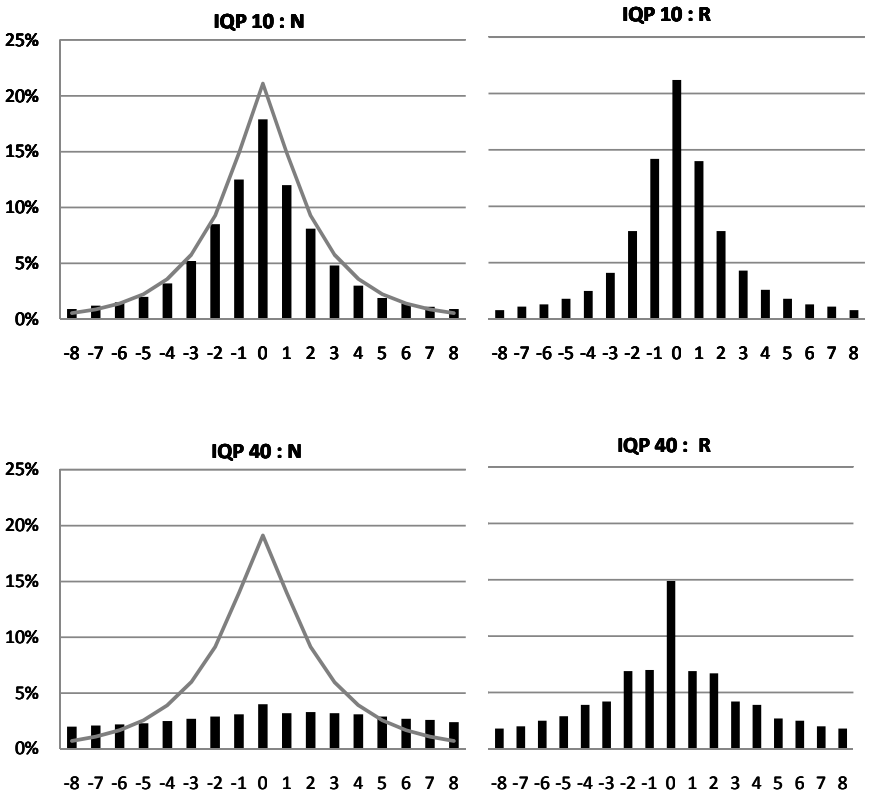

Figure 4.7: Measured distribution of N and R, for the luma DC component of Foreman, for different quantization levels (H.264/AVC intra qp given). The average estimated distribution of N is drawn on top. Clearly, we can see that while the estimated noise is close to the true noise for fine quantization (IQP 10), there is a significant mismatch for coarse quantization (IQP 40).

tion B.).

The architecture of the codec is depicted in Figure 4.8. It is based on the DVC mode of the codec discussed in the previous chapter. Apart from the extensions proposed in this section, the only difference with the DVC mode discussed in Section 3.3.4 is that no residual coding approach is employed, as in most common DVC-only systems. One of the reasons for not using a residual approach is that residual coding establishes a link between the key frame encoder and the WZ encoder, and in this sense, the system is not a DVC system (i.e., where frames are encoded independently from other frames). Remark also that the correlation model is explicitly provided as input to the turbo decoder, whereas in previous chapters this link was not depicted although it was implicitly assumed to be present.

A. Encoder-side

For each coefficient band, the variance of the quantization noise of the intra frames is calculated by calculating the variance of the difference between the transformed original intra frame and the transformed intra decoded version. The computational overhead remains low, since H.264/AVC intra decoded frames are generated anyway by the intra encoder in the context of mode decision and rate-distortion optimization.

To limit the overhead of sending the quantization noise variances to the decoder, some additional processing steps are performed. Firstly, it was observed that the variances of the chroma components (U and V) are usually very similar. Therefore, only the average of both is used. Next, the variances are quantized using a uniform quantizer having 2^M bins. It can be assumed that the variances are never much larger than the variance of a random variable that is uniformly distributed over the quantization bin, so that the quantizer range can be restricted to the interval [0, d^2/12]. d can be easily calculated from the H.264/AVC intra QP. For M, the largest integer is taken that is not greater than 5 and for which d is at least 1. Since a 4-by-4 DCT transformation is used, the result of processing the variances is that at most $5 \cdot (16 + 16) = 160$ bits need to be sent to the decoder per I frame.

Since the quantization noise statistics do not always change drastically from intra frame to intra frame, information is only sent if the average difference between the newly quantized variances and the previously-sent quantized values is at least 0.5. This ensures that only significant updates are communicated.

In our experiments, the above processing steps proved to be efficient. The overhead of sending the quantization noise information to the decoder only

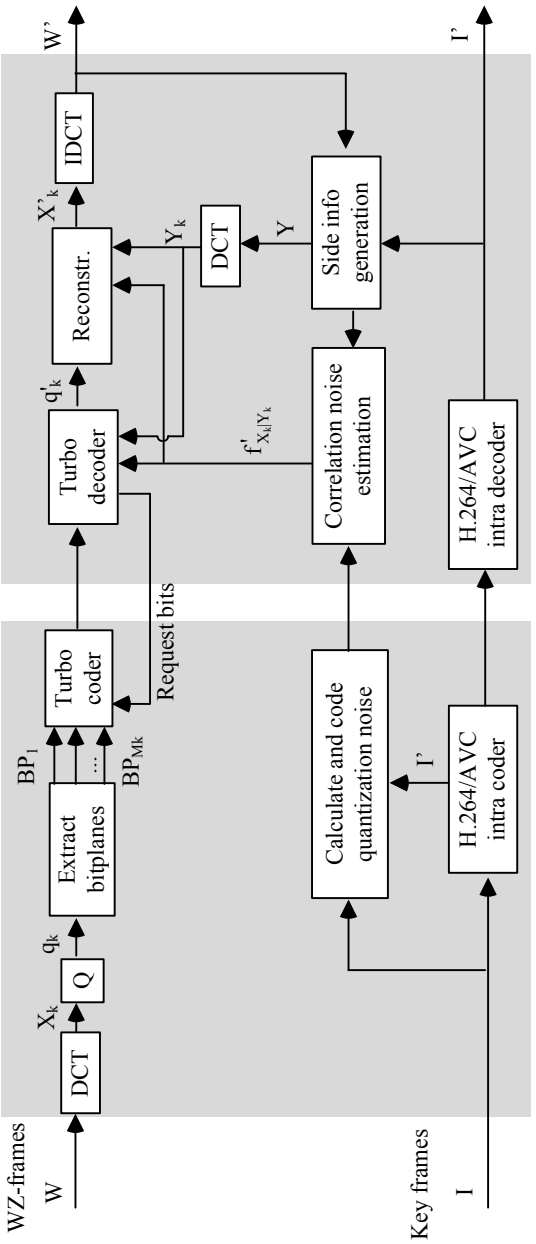

Figure 4.8: Functional diagram of the DVC codec featuring correlation noise estimation with quantization noise compensation.

accounts for maximum 0.05% of the total bit rate, for each rate point.

B. Decoder-side

At the decoder, the coded variances for the intra frames are received from the encoder, and reconstructed. Next, the decoder uses these variances to improve the modeling of the correlation noise for a particular WZ frame with associated side information Y. As in most DVC systems, quantization of intra frames and WZ frames is chosen in such a way that all frames have more or less the same quality. Therefore, it is assumed that the decoded intra frames and decoded WZ frames are all affected by more or less the same noise. In addition, assume that the quantization noise in Y is similar to the noise in the reference frames, so that the quantization noise variances σ_Q^2 received from the encoder can be applied to the side information Y.

Now that an approximation of the quantization noise in Y has been obtained, this needs to be combined with the noise induced by motion, deformation, illumination changes, etc. Since the current methods for correlation noise estimation provide a good approximation when quantization noise is low, as shown in our analysis (Figure 4.7), both methods are combined. The standard deviation σ of the correlation noise N associated with a coefficient at index (i, j) in block k, is thus estimated as:

$$\sigma = \sigma^T_{k,(i,j)} + C \cdot \sigma_Q, \tag{4.15}$$

with

$$C = \begin{cases} 1 - \frac{\sigma^T_{(i,j)}}{2\sigma_Q} & \text{, if } \frac{\sigma^T_{(i,j)}}{2\sigma_Q} < 1 \\ 0 & \text{, otherwise} \end{cases} \tag{4.16}$$

where $\sigma^T_{k,(i,j)}$ and $\sigma^T_{(i,j)}$ relate to Eq. 4.12 and Eq. 4.14, respectively, since:

$$\sigma^T_{k,(i,j)} = \frac{\sqrt{2}}{\alpha^T_{k,(i,j)}}, \tag{4.17}$$

and

$$\sigma^T_{(i,j)} = \frac{\sqrt{2}}{\alpha^T_{(i,j)}}. \tag{4.18}$$

To justify this experimentally derived formula, similar measurements are performed as in Section 4.3. Comparing the new model for correlation noise estimation to the actual correlation noise in Figure 4.9 clearly shows that our estimation has become significantly more accurate.

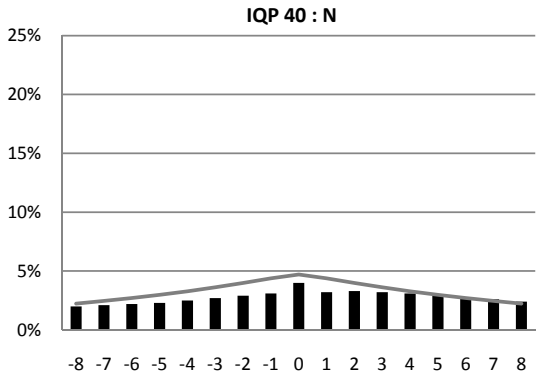

Figure 4.9: The new method for correlation noise estimation is more accurate for coarse quantization (i.e., low rates).

4.3.2 Results

Tests have been performed on sequences with different motion characteristics: Mother and Daughter, Foreman, Table Tennis, and Football. All are at CIF resolution, 30 fps, and with a GOP length of 4. Results are given in Table 4.2 for Wyner-Ziv frames only.

The Bjøntegaard delta metric [102] is used to illustrate the rate difference at a given level of quality. This metric shows that this new technique performs particularly well at low rates (i.e., coarse quantization), with average rate gains up to 19.5% per Wyner-Ziv frame for Mother and Daughter.

The reason why the gain is largest for Mother and Daughter can be explained as follows. Mother and Daughter is a sequence with not much motion, hence, the side information generation process is able to find very good matches, resulting in small values for R and consequently for $\sigma^T_{k,(i,j)}$ (and $\sigma^T_{(i,j)}$). Therefore, σ_Q is relatively large, which results in a large impact on σ. For sequences with more movement, $\sigma^T_{k,(i,j)}$ and $\sigma^T_{(i,j)}$ are larger so that the impact of our update is smaller.

The results obtained for our technique are interesting, but some areas still need further exploring. For example, we have assumed so far that the quantization noise in the decoded intra frames and the decoded WZ frames is similar. This might not always be true since the reconstruction of the intra frames and the reconstruction of the WZ frames is performed using different techniques. Some of the other remaining issues have been addressed in collaborative work, as described next.

		Proposed		Previous work (Section 4.2.2)		BJM delta rate	
		PSNR (dB)	rate (kbps)	PSNR (dB)	rate (kbps)	PSNR (dB)	rate (%)
MD	Q0 , IQP 25	41.8	513	41.8	540	41	-6.1
	Q1 , IQP 29	38.9	261	38.9	288	38	-11.6
	Q2 , IQP 33	36.4	129	36.3	151	36	-16.6
	Q3 , IQP 37	34.3	69	34.2	88	35	-19.5
Table	Q0 , IQP 26	37.5	1666	37.5	1668	37	0.1
	Q1 , IQP 32	33.4	827	33.3	835	33	-2.0
	Q2 , IQP 38	30.2	405	30.1	421	30	-6.4
	Q3 , IQP 43	27.8	216	27.7	235	28	-10.3
Foreman	Q0 , IQP 27	38.2	1434	38.1	1445	38	-1.4
	Q1 , IQP 32	34.8	708	34.7	722	34	-3.9
	Q2 , IQP 38	31.6	357	31.4	374	31	-7.2
	Q3 , IQP 43	28.7	188	28.6	201	29	-9.1
Football	Q0 , IQP 29	37.0	2899	37.0	2902	36	0
	Q1 , IQP 36	32.7	1578	32.7	1580	32	-0.4
	Q2 , IQP 42	29.1	787	28.9	794	29	-3.5
	Q3 , IQP 48	26.0	357	25.5	365	27	-8.7

Table 4.2: Average results per WZ frame, for Mother and Daughter (MD), Table Tennis (Table), Foreman, and Football. Bjøntegaard delta rate metrics (BJM) illustrate the evolution of the gain for different levels of quality (negative values indicate decrease in bit rate).

4.3.3 Recent developments in co-authorship

The interesting gains obtained by this technique have motivated us to refine it further. A number of problems have been dealt with by my colleague Jozef Škorupa, leading to several contributions as a co-author.

In a first improvement, a model for motion-compensated interpolation has been established first. Based on this model, the experimentally justified formula in Eq. 4.15 could be refined to a formula that is theoretically justified as well. This provides a solid basis, and it improves upon the work described in this section with an additional bit rate gain between 1.1% and 3.7% [103].

In a second improvement, sending information regarding quantization distortion from the encoder to the decoder is avoided. This leads to a more elegant system in which the decoder estimates the quantization noise by itself, without having to rely on the encoder. Despite small inaccuracies in the estimation performed at the decoder-side, avoiding to send quantization information even realizes small bit rate gains between 0.2% and 0.5% [104].

4.3.4 Conclusions

Using only the motion-compensated residual between the reference frames (used for generating the side information) does not always deliver an accurate estimation of the correlation noise. Instead, in this section we have shown that the quantization distortion in the reference frames needs to be taken into account as well. Using this additional information has resulted into significant performance gains.

On the one hand, the results in this section underline the importance of accurate correlation noise modeling in DVC. On the other hand, the results encourage researchers to investigate other sources of additional information, and develop new correlation noise models.

4.4 Compensating for motion estimation inaccuracies

Learning from the experience that the correlation model can be further improved by using more information than only the motion-compensated residual, in this chapter, a second contribution is presented. Here, we compensate for inaccuracies in the generation of the side information, by using a correlation model based on multiple predictors instead of one.

Current techniques for side information generation commonly assume that the motion between the past and future reference frames can be approximated as linear. This assumption is made in, for example, the Stanford DVC architecture [40] as well as in DISCOVER [51]. Even in cases where more complex motion models are used, motion interpolation is still performed in a linear fashion. For example, Kubasov et al. [105] use mesh-based techniques to obtain a model of the motion between the past and future reference frame. The side information is then created through linear interpolation along the motion trajectories described by this model, assuming uniform motion between the key frames.

The assumption that the motion between the reference frames can be approximated as linear, becomes less valid when the distance between the reference frames increases. This is illustrated for a GOP of size eight. Side information has been generated for the 5th frame of the Foreman sequence, using the first frame as a past reference, and the 9th frame as a future reference. When analyzing the residual between the side information and the original frame in Figure 4.10, it is clear that a lot of errors need to be corrected. Judging from the side information itself, it could already be expected that the accuracy of estimating the face is low. However, the residual also reveals that errors need to be corrected in the background. More specifically, we can see that edges in the side information are not predicted accurately. This is due to non-linear camera motion.

To compensate for inaccuracies in side information generation, most recent techniques apply a refinement approach. Decoding is performed partially, and the partially decoded frame is used to improve the side information. The improved side information is then used for further decoding. For example, Martins et al. [106] propose to refine the side information after each coefficient band has been decoded. A layered approach is used by Fan et al. [107], dividing each frame into a low-resolution base layer and higher resolution refinement layers. The decoded base layer can be used for improving side information accuracy of the following, higher resolution refinement layer, and so on. Information about the (partially) decoded frame can also be used to re-evaluate the side information generation process, for example, by identifying

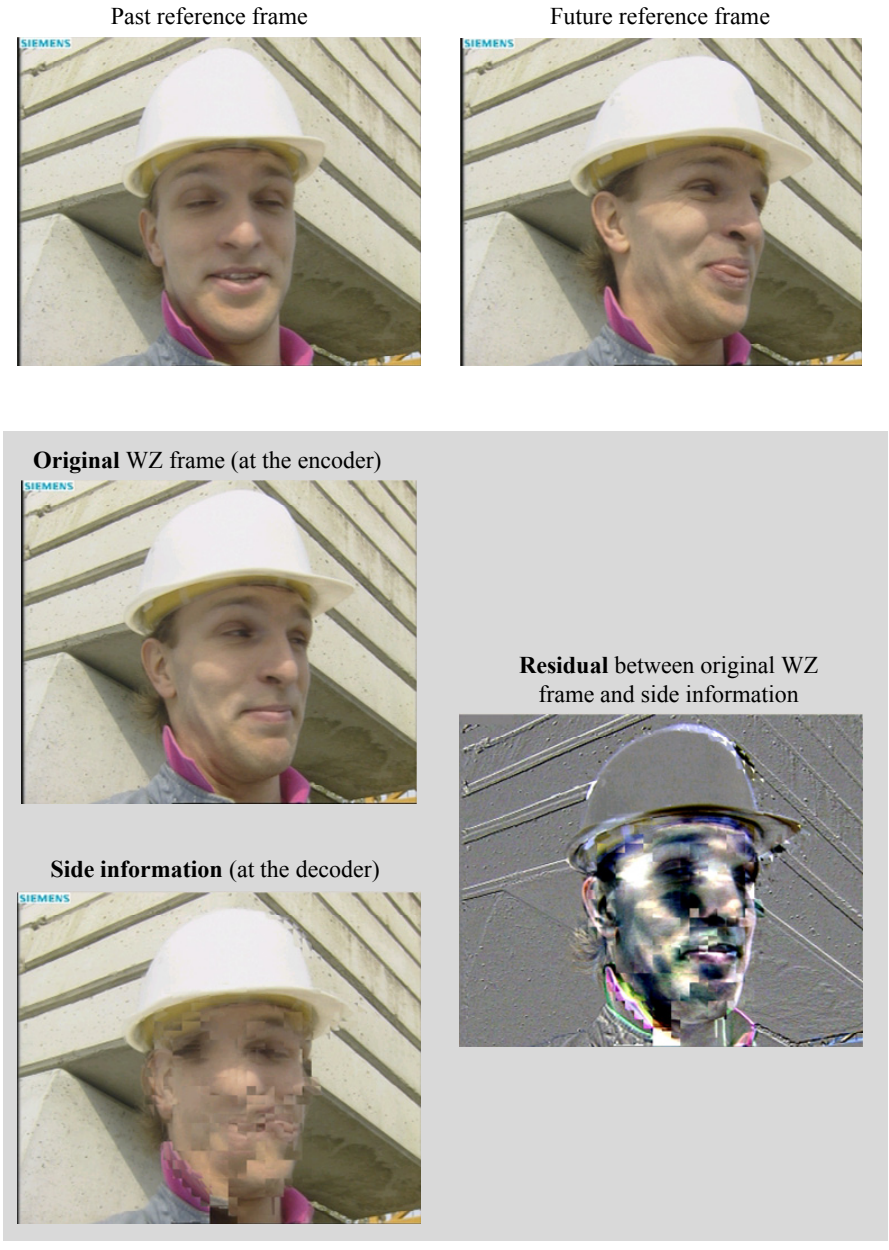

Figure 4.10: Especially for large GOP's, the assumption of linear motion becomes less valid.

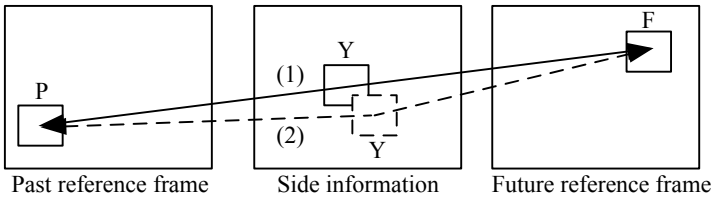

Figure 4.11: The linear motion vector (1) could be inaccurate, so that the interpolation between P and F is located on a different spatial position (2).

possibly inaccurate motion vectors [108].

While these techniques show good results, what they have in common is that they can compensate for errors only *after* some information has been decoded. Therefore, in this section, a technique is proposed where some of these uncertainties are quantified *prior to decoding*. This is realized by extending the correlation model, improving compression with an additional 8% gain in bit rate.

4.4.1 Proposed technique

The main idea is to compensate for some inaccuracies by using more than one prediction for each block. Recall that a particular block in the side information Y is generated by averaging past and future reference blocks P and F, using a linear motion vector. However, if the motion is non-linear, then the prediction should appear on a different spatial position in the side information (Figure 4.11). Hence, to predict a block at position (x_0, y_0), the block at position (x_0, y_0) in Y can be used, together with some of the surrounding blocks in Y. This strategy can also be beneficial in other cases with complex motion such as occlusion and deformation.

Before explaining this method, a description of the codec is provided.

A. Codec description

The proposed codec is depicted in Figure 4.12, highlighting the extensions for compensating for motion estimation inaccuracies.

As before, the techniques for side information generation are adopted from DISCOVER. The output of this process is the side information Y, and for each block the (linear) motion vector MV_{SI}, as well as the residual R_{SI} between the past and future reference blocks. This information is used as input for the proposed extensions. First, for each block, multiple predictors are generated

(Section B.). Next, each of these predictors is assigned a weight (Section C.), and the correlation between the predictors and the original is modeled (Section D.). This distribution is used by the turbo decoder, which requests bits until the decoded result is sufficiently reliable. Finally, the quantized coefficients are reconstructed (Section E.) and inverse transformed to obtain the decoded frame W'.

B. Generation of predictors

A block at position (x_0, y_0) is predicted using multiple predictors, obtained from the side information frame Y. The first predictor is the predictor corresponding to linear motion, i.e., the block at the co-located position in Y. To compensate for motion inaccuracies such as non-linear motion, surrounding blocks in Y are taken into account as well. As a compromise between complexity and performance, eight additional predictors are used, namely the ones corresponding to positions $(x_0 \pm m, y_0 \pm m)$ in Y ($m \in \{0, 1\}$). This results in a total of 9 predictors per block.

Not every predictor is equally likely, so that weights are calculated for each predictor, as explained in the following section.

C. Online calculation of the predictor weights

Each of the 9 predictors is assigned a weight, according to the probability that this predictor is the best one out of the set. This probability is estimated using the results from previously decoded frames. In a previously decoded frame W', given the side information Y, the best predictor for a block is obtained using the following procedure.

Each block in W' is compared to each of the 9 predictors in Y. More specifically, the mean absolute difference (MAD) is calculated between the block at a certain position (x_0, y_0) in W' and the co-located block in Y. This MAD indicates the amount of errors corrected when using the linear predictor. Likewise, the MAD for other predictors is calculated, for example, comparing the block at position (x_0, y_0) in W' to the block at position $(x_0 + 1, y_0 + 1)$ in Y etc. The predictor with the lowest MAD is then considered the best one out of the set.

However, a non-linear predictor is only considered best in case its MAD is lower than 0.9 times the MAD of the linear predictor. Otherwise, the linear predictor is considered to be the best. This criterion is used to ensure that only significant improvements over the linear predictor are taken into account. For example, in a region with not much texture, one of the non-linear predictors could have a lower MAD than the linear predictor, because the quantization

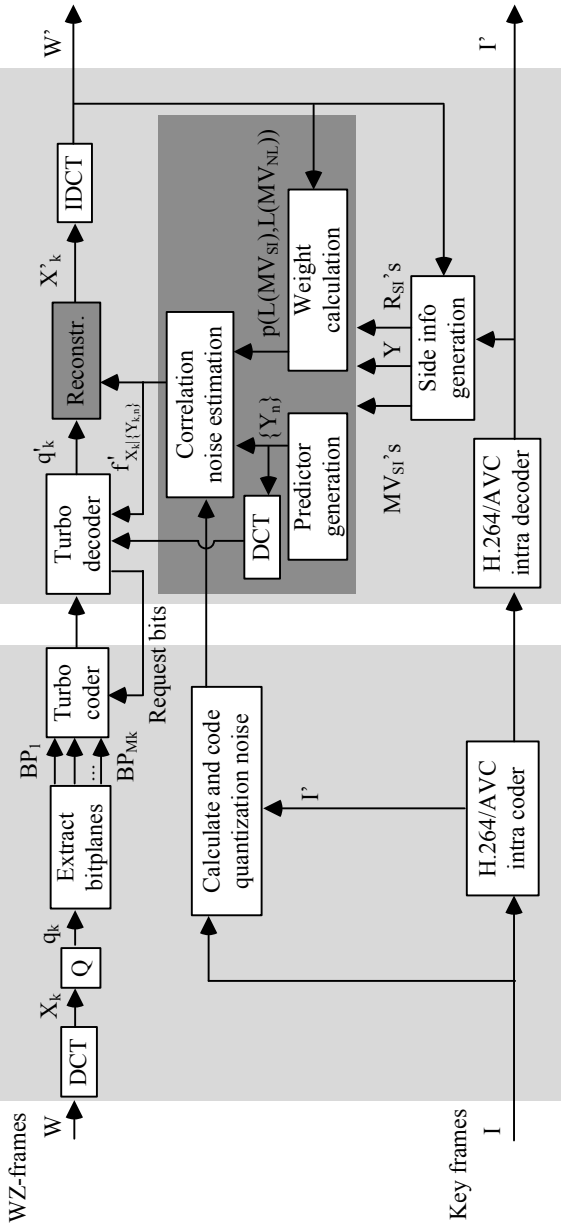

Figure 4.12: The DVC codec featuring a correlation model that compensates for quantization noise and motion estimation inaccuracies.

noise in this predictor has distorted the block in such a way that it resembles better the decoded result. To avoid these situations, the MAD of a non-linear predictor must be lower than 0.9 times the MAD of the linear predictor. The value of 0.9 has been experimentally obtained.

Given the best predictor, a table is updated, based on a characterization of the predictor using two parameters.

The first parameter is the amplitude of the motion. For example, the linear predictor could be highly reliable in static regions (e.g., in the background), but its reliability could be much lower for fast moving objects in the foreground. To discriminate between such cases, the amplitude of the (linear) motion vector MV_{SI} is used. To this extent, the following amplitude metric $L()$ is defined:

$$L((x, y)) = max(|x|, |y|). \tag{4.19}$$

The second parameter discriminates between different predictors, through the amplitude of the non-linearity of the predictor. Denote MV_{NL} as the predictor offset compared to the linear predictor. For example, if the linear predictor corresponds to the block at position (x_0, y_0) in Y, then the predictor at position $(x_0 + 1, y_0 - 1)$ in Y has $MV_{NL} = (1, -1)$.

Due to the use of the amplitude metric for this second parameter, all 8 non-linear predictors have a value of one for $L(MV_{NL})$. Only the linear-motion predictor has a different value, namely zero. As such, the statistics of the predictor having $MV_{NL} = (1, -1)$ are assumed to be similar to those of the predictor having $MV_{NL} = (0, 1)$. This simplification can be refined in future work, for example, by assigning higher weights to the non-linear predictors in the direction of the motion MV_{SI}.

Given the best predictor and its parameters $L(MV_{SI})$ and $L(MV_{NL})$, a table T is updated. This table only covers the statistics of the current frame. All elements have been initialized to zero before any updating takes place. The parameters $L(MV_{SI})$ and $L(MV_{NL})$ serve as coordinates in T, and the corresponding value in T is incremented by one, for the best predictor.

After all blocks in W' have been processed, the result is combined with the result from previously decoded frames, by updating global statistics:

$$p_{i,j} = K \cdot p_{i,j} + (1 - K) \cdot \frac{T(i, j)}{\sum_k T(i, k)}, \tag{4.20}$$

where $p_{i,j}$ is a shorthand for $p(L(MV_{SI}) = i, L(MV_{NL}) = j)$. The update parameter K is set to 0.8. This value – which has been obtained through experiments – remains fixed for all test sequences. A more detailed study of the update parameter is left as future work.

The global statistics are used for calculating the weights for the predictors in the following WZ frame to be decoded. More specifically, the weight $w_{i,j}$ for a predictor characterized by $L(MV_{SI}) = i$, and $L(MV_{NL}) = j$ is calculated as:

$$w_{i,j} = \frac{p_{i,j}}{N_j}, \tag{4.21}$$

where N_j denotes the number of predictors (for that block) having a value of j for $L(MV_{NL})$. Hence, N_j equals one for the linear-motion predictor, and 8 for the remaining ones.

D. The correlation model

The goal is to model the correlation between the original X and the set of predictors denoted $\{Y_n\}$ (with $0 \leq n \leq 8$). This is modeled in the (DCT) transform-domain. For each 4-by-4 block in the original frame, 16 distributions are generated, i.e., one for each coefficient X_k (with $0 \leq k \leq 15$). The predictors are transformed, and all coefficients at the same index are grouped. Denote the predictors for X_k as $\{Y_{k,n}\}$.

As explained previously in this chapter, the correlation between the original and the side information is commonly modeled using a Laplace distribution. Hence, with multiple predictors, the conditional distribution $f_{X_k|\{Y_{k,n}\}}$ is modeled as a combination of weighted Laplace distributions, i.e.:

$$f_{X_k|\{Y_{k,n}\}}(x|\{y_{k,n}\}) = \sum_m w_m \cdot \frac{\alpha}{2} e^{-\alpha|x-y_{k,m}|}, \tag{4.22}$$

where $y_{k,m}$ indicates the k-th coefficient of the m-th predictor. w_m is the weight of the m-th predictor.

The scaling parameter α is calculated based on the reference residual of the linear predictor, using the techniques proposed in Section 4.3.

E. Coefficient reconstruction

After turbo decoding, the quantization bin q'_k containing the original value (with very high probability) is known at the decoder. The following step is to choose a value in this quantization bin as the decoded coefficient X'_k. This is done through optimal centroid reconstruction [60]:

$$X'_k = \frac{\sum_m w_m \int_{q'_L}^{q'_H} x \cdot \frac{\alpha}{2} e^{-\alpha|x-y_{k,m}|} dx}{\sum_m w_m \int_{q'_L}^{q'_H} \frac{\alpha}{2} e^{-\alpha|x-y_{k,m}|} dx}, \tag{4.23}$$

where q'_L and q'_H indicate the low and high border of q'_k, respectively.

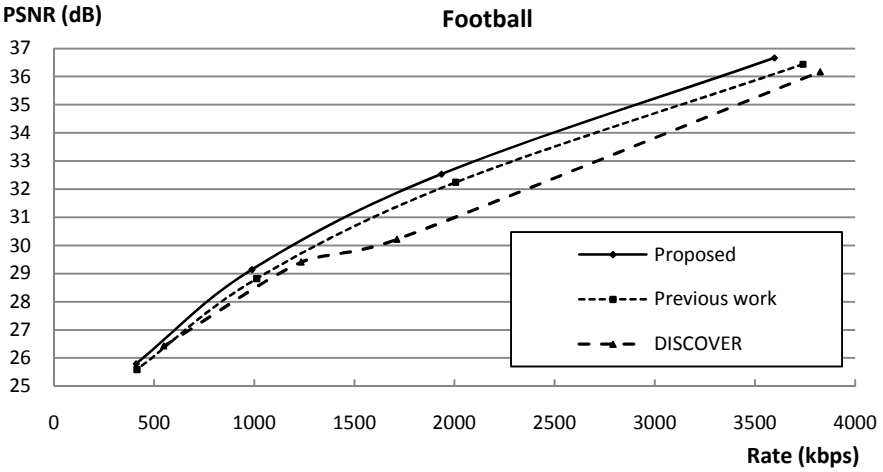

Figure 4.13: For Football, average quality gains up to 0.4 dB are reported over our previous work.

4.4.2 Results

Tests have been conducted on three different sequences: Foreman, Football and Coastguard. All are in CIF resolution, at a frame rate of 30 frames per second. A GOP of size 8 is used, and only the luma component is coded for comparison with the DISCOVER codec. The system is also compared to our previous method described in Section 4.3, which applies a better correlation noise model than DISCOVER, but still uses only one predictor per block.

The results indicate improvements over both systems. The gains are the largest for sequences with complex motion such as Football (Figure 4.13) and Foreman (Figure 4.14), where the linear predictor does not always provide an accurate prediction. In these cases, using multiple predictors to compensate for inaccuracies shows average Bjøntegaard [102] quality gains up to 0.4 dB over our previous work, and 1.0 dB over DISCOVER (both for Football and Foreman).

For sequences with rather simple motion characteristics, such as Coastguard (Figure 4.15), less gain is observed. For such sequences, an accurate prediction is already provided by the linear-motion predictor, and so using additional predictors provides less gain. Over our previous work, average quality gains of 0.1 dB are reported, and 1.4 dB over DISCOVER.

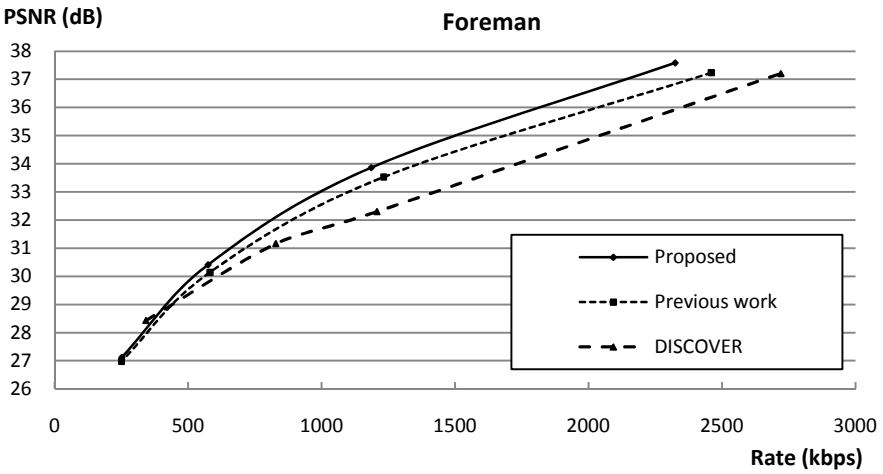

Figure 4.14: For Foreman, average quality gains up to 0.4 dB are reported over our previous work.

4.5 Conclusions and original contributions

Modeling the correlation between the original frame available at the encoder, and its prediction available at the decoder, is an important but difficult problem in DVC. While most current techniques rely on the motion-compensated residual between the reference frames, in this chapter, two techniques have been proposed that use more than this residual. The first technique exploits information about the quantization of the reference frames, while the second technique compensates for inaccurate assumptions made when generating the side information. Both approaches provided gains over classical techniques in the literature.

The author's work on correlation noise estimation has led to the following publications:

- Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Correlation noise estimation in distributed video coding. Accepted for publication in: *Video coding*, IN-TECH, 2011.

- Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Compensating for motion estimation inaccuracies in distributed video coding. In *Proc. In-*

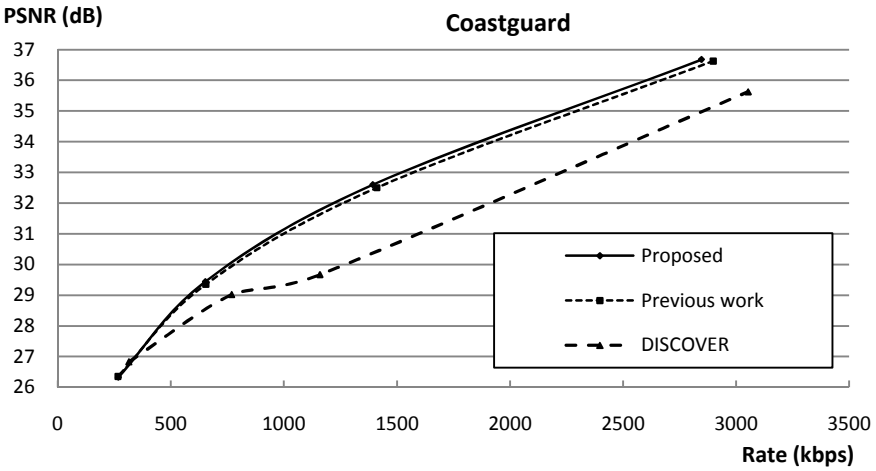

Figure 4.15: As expected, less gain is achieved for sequences with simple motion characteristics, although still 0.1 dB over our previous work.

ternational Conference on Image and Signal Processing (ICISP), pages 324–332, June 2010.

- Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Peter Lambert, Christos Grecos, and Rik Van de Walle. Accounting for quantization noise in online correlation noise estimation for distributed video coding. In *Proc. Picture Coding Symposium (PCS)*, May 2009.
- Jozef Škorupa, Jan De Cock, Jürgen Slowack, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Correlation modeling with decoder-side quantization distortion estimation for distributed video coding. Accepted for publication in: *Proc. Picture Coding Symposium (PCS)*, December 2010.
- Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Nikos Deligiannis, Jan De Cock, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Exploiting quantization and spatial correlation in virtual-noise modeling for distributed video coding. *Signal Processing: Image Communication*, 25(9):674 – 686, 2010.
- Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Peter Lambert, and Rik Van de Walle. Accurate correlation modeling for transform-domain Wyner-Ziv video coding. In *Proc. Pacific-Rim Conference on Multimedia (PCM)*, pages 1–10, December 2008.

Chapter 5

RD driven decoder-side bitplane mode decision

5.1 Introduction

In conventional video compression with encoder-side motion estimation, significant performance gains are achieved by using a large number of coding modes. Frames are divided into blocks, and for each block, the encoder generates a (usually large) number of predictors. Next, the best predictor out of the set is selected, which is the one reaching the best trade-off between the number of bits to spend and the quality after decoding. By implementing a decoder loop at the encoder, each of the predictors can be used in a coding-decoding operation, to obtain the coded bit rate and the end distortion associated with each of the predictors. This is done beforehand, i.e., without sending any information to the decoder. Next, only the best predictor is used for the actual coding, which results in the highest compression performance for the considered set of predictors.

Extending these techniques to DVC is not straightforward. The main problem here is that the encoder does not have the predictors for the WZ frames available, since they are (or should be) generated at the decoder only. Generating the predictors at the encoder as well would leave us with a complex encoder, which is not desirable in a DVC context (specifically targeting low-complexity encoding). On the other hand, while the complexity of the decoder is considered less of an issue in DVC, the decoder has only the predictor Y (or set of predictors), so that rate and distortion cannot be obtained either. Hence, estimations/approximations for rate and distortion need to be used. To solve this problem, two approaches can be identified: an encoder-side approach, or a decoder-side approach.

Most current techniques apply an encoder-side approach. At the encoder, an estimation Y' of Y is generated using low-complexity techniques. A well-known example is the PRISM architecture, where the past neighboring frame is used, and thresholds on the difference between the current frame and the past neighboring frame define the mode for coding the current block. Together with Stefaan Mys and others [109] we proposed similar ideas in a Stanford-alike DVC architecture for the introduction of block-based skip. Error correcting information is still sent for an entire WZ frame, but the turbo decoding procedure is adjusted to take the information about skipped blocks into account. Liu et al. [110] propose an iterative method for deciding on a block-basis between intra mode and WZ mode, assuming Y' to be available. Both distribution parameters as well as the modes are determined through an iterative procedure. The models that are used are spatially stationary, having the same variance for all DCT coefficients at a given index and a given mode. Ascenso et al. [48] use fast motion estimation techniques to generate Y', and propose rate-distortion based mode decision for deciding between intra mode and WZ mode. Blocks classified as intra are more coarsely quantized than WZ blocks, and they are used for enhancing the side information at the decoder. Next, the enhanced side information is corrected using error correcting information calculated on the entire frame (intra + WZ blocks).

The disadvantage of these encoder-side techniques is that the compression performance depends on the quality of Y' at the encoder. This introduces a trade-off between compression performance and encoder complexity, since the best results will be obtained in case the estimated Y' coincides with the actual side-information Y.

As a solution, the decoder can be made responsible for performing mode decision. However, very few techniques have been proposed so far. Chien et al. [111] use rate-distortion based decoder-side mode decision, deciding between skipping or WZ coding of a given bitplane. This decision is based on a threshold. While their system outperforms the DISCOVER codec for sequences with low motion, objective results are inferior for sequences with more motion.

The idea of decoder-side mode decision is carried forward in this chapter. First, formulas are established for estimating rate and distortion at the decoder (Section 5.2). In contrast to previous approaches (e.g. [48]), the position of the realization of the side information in the quantization bin is taken into account. Next, based on these theoretical derivations, techniques for rate-distortion-based decoder-side mode decision are developed. Two levels for mode decision are introduced. At the coefficient level, the decoder decides if the entire coefficient band should be skipped or not (Section 5.3.1). At the

bitplane level (Section 5.3.2), the decoder decides between bitplane skip, bitplane WZ coding and bitplane intra coding. These techniques have been used to extend the codec developed so far (Section 5.4), and significant improvements are reported for different test sequences (Section 5.5). Conclusions are provided in Section 5.6.

5.2 Rate-distortion modeling

Most of the challenges in DVC are due to the inherent problem of not having the side information Y available at the encoder. One consequence is that, at the encoder, one has to choose a quantizer for coding X, without knowledge of Y. The latter is typically solved by quantizing and coding X independently from Y, in contrast to conventional systems where the residual $X-Y$ would be quantized and coded. However, not using the residual approach is less efficient, especially at reasonably low rates. This will be illustrated theoretically, and these results will be used to develop a method for decoder-side skipping of coefficients that would be too inefficiently coded.

Consider a uniform quantizer with step size Δ and quantization bins labeled q_i ($i \in \mathbb{Z}$). The low and high borders of each quantization bin are given by $q_i^L = \Delta(i-0.5)$ and $q_i^H = \Delta(i+0.5)$, respectively. As before, the correlation between X and Y is modeled by a Laplace distribution:

$$f_{X|Y}(x|y) = \frac{\alpha}{2} e^{-\alpha|x-y|}, \tag{5.1}$$

where α is the distribution scale parameter. Denote the quantization bin containing the realization of the side information as q_K. A parameter is introduced that describes the position of the side information in q_K, i.e., $y_N = \frac{y}{\Delta} - K$ taking values in $[-0.5; 0.5]$. As shown in appendix C.1, the minimum rate that is needed to communicate a quantized version of X from the encoder to the decoder is given by[1]:

$$\begin{aligned} H(Q(X)|Y_N = y_N) =& A \cdot \cosh(\alpha\Delta y_N) + B \cdot y_N \cdot \sinh(\alpha\Delta y_N) \\ & -(1 - e^{-\alpha\Delta/2}\cosh(\alpha\Delta y_N)) \\ & \cdot \log_2(1 - e^{-\alpha\Delta/2}\cosh(\alpha\Delta y_N)), \end{aligned}$$

[1] Eq. C.10 is repeated for convenience.

where,

$$A = e^{-\alpha\Delta/2}\left(\frac{\alpha\Delta}{\ln 2\cdot(1-e^{-\alpha\Delta})} - \log_2\sinh(\alpha\Delta/2)\right),$$
$$B = -\frac{\alpha\Delta}{\ln 2}e^{-\alpha\Delta/2}. \tag{5.2}$$

Next, after the decoder receives the index of the bin containing the original X, a particular value in this bin needs to be chosen as a reconstruction X'. When using the reconstruction employed in the Stanford codec [42] and Mean Absolute Difference (MAD) as a distortion measure (as in [48]), as shown in appendix C.2, the following expected distortion[2] is obtained:

$$E[|X-X'||Y_N=y_N] = -\cosh(\alpha\Delta y_N)\cdot\Delta e^{-\alpha\Delta/2}\cdot\left(\frac{e^{-\alpha\Delta}}{1-e^{-\alpha\Delta}}+0.5\right) + \Delta y_N\sinh(\alpha\Delta y_N)\cdot e^{-\alpha\Delta/2}+\frac{1}{\alpha}. \tag{5.3}$$

Note that $H(Q(X)|Y_N=y_N)$ as well as $\alpha\cdot E[|X-X'||Y_N=y_N]$ are both functions of $\alpha\Delta$. Hence, the rate-distortion curves can be plotted for different positions y_N, by choosing any value for α and varying Δ. The result is depicted in Figure 5.1. The accuracy of the rate-distortion model has been verified using a sample set of 100000 coded samples generated by a Laplacian source, with $\alpha = 1$. The rate-distortion curves generated experimentally and the ones generated using the proposed formulas for rate and distortion, coincide. The average absolute error is not higher than 0.002.

The shape of the curves in Figure 5.1 is explained as follows. At the one end, when Δ is very small (i.e., for high rates), the rate and distortion become independent from the position of the side information. The curves coincide and no coding loss is observed. When Δ goes to infinity, the distortion approaches $\frac{1}{\alpha}$ while the entropy goes to zero, except for the case where $y_N = 0.5$, as explained further.

Consider the extreme case where the realization of the side information lies exactly on the bin border ($y_N = 0.5$). If Δ goes to infinity, this means that we end up with two quantization bins, each of infinite length, and having y exactly on the border between the two quantization bins. Due to the symmetry of the Laplace distribution, each of the quantization bins will have a probability of 50%. Therefore, the rate required to communicate the correct bin from the encoder to the decoder equals one (instead of zero as in all other cases depicted). At the decoder side, given the correct quantization bin, the reconstruction process will either select the side information y or the bin border (which is again

[2] Eq. C.19 is repeated for convenience.

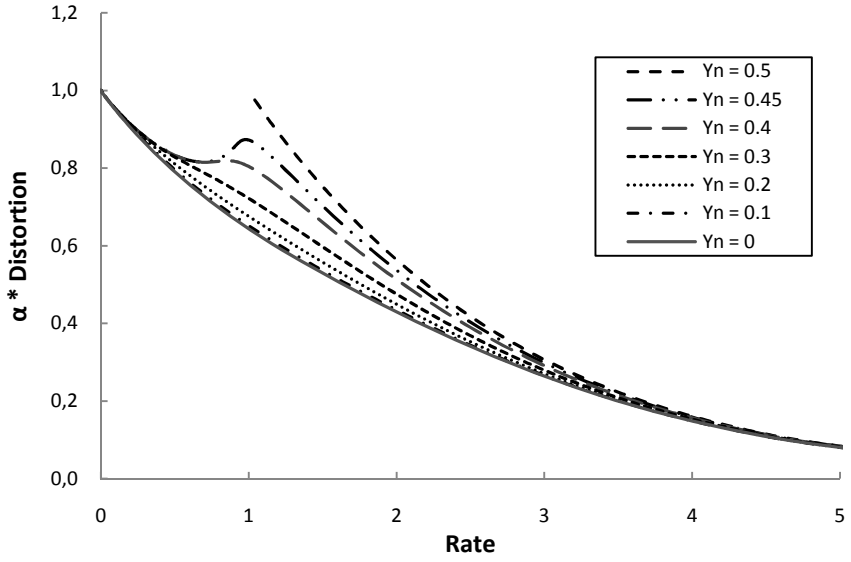

Figure 5.1: Theoretical rate-distortion results for different positions of the side information in the quantization bin.

y). Hence, this reconstruction is the same as in the case where the coefficient would have been skipped. As a consequence, the distortion is also the same in both cases, i.e., $\frac{1}{\alpha}$.

For other positions of the side information (e.g., $y_N = 0.45$), similar reasoning can be applied to explain the occurrence of distortion peaks around a bit rate of one.

To summarize, we can conclude that a coding loss is observed if the realization of the side information is not in the center of the quantization bin. If Y lies closer to bin border, more bits are needed for achieving the same amount of distortion . This coding loss is significant for relatively low rates. This is the case in most DVC systems where coefficients are quantized to a low number of bits (e.g., 2 to 7 bits [52]).

Now that coding inefficiency has been identified, the question arises whether or not it is possible to avoid this loss. One solution is to perform residual coding by generating an estimation Y' of the side information both at the encoder as well as at the decoder. Next, the residual between X and Y' is coded, as in [78]. An intuitive explanation for the gains achieved using this approach is that coding the residual between X and Y' is equivalent to coding X with a quantizer that is shifted so Y' is in the center of the quan-

tization bin. Hence, if there is strong correlation between Y' and Y, then Y has high probability to lie in the center of the bin as well. As such, there is high probability that the best coding performance is achieved. This introduces a trade-off between encoder complexity and rate-distortion performance, since the best results are obtained if $Y' = Y$.

A different strategy is to avoid inefficient coding by choosing between several coding modes. This technique is adopted in this chapter. Since the mode decision is performed at the decoder-side, the complexity of the encoder is not increased.

To develop this method, first some additional problems are discussed concerning rate and distortion in a practical DVC system.

5.2.1 Additional problems

The WZ rate in a practical DVC system is higher than the conditional entropy, due to a number of factors. A first problem is that the correlation between X and Y (i.e., the conditional distribution $f_{X|Y}$) needs to be estimated (e.g., by $f'_{X|Y}$), as discussed in chapter 4. Inaccuracies in this estimation result in a bit rate penalty. This rate penalty can be significant in some cases, especially for higher rates where the mismatch between the bin probabilities increases.

A second issue is that – even if the conditional distribution $f_{X|Y}$ would be known – the conditional entropy can only be reached by a perfect conditional entropy coder, while in practice efficient but suboptimal solutions are used such as LDPC and turbo coding.

Finally, additional rate penalties are imposed by the feedback channel as well as the puncturing period (which determines the smallest chunk of bits that can be sent between encoder and decoder), but this overhead can usually be neglected.

Hence, to summarize, it can be said that estimating the bit rate using the conditional entropy (Eq. 5.2) will not always be entirely accurate. Our techniques compensate for this problem by the use of an intra mode for bitplane coding, and more strict criteria for coefficient band skipping and bitplane decoding, as explained further.

As far as distortion is concerned, the only assumption in Eq. 5.3 is that the correct quantization bin (i.e., the one containing the original) is successfully communicated from the encoder to the decoder. This is usually a valid assumption, especially when using a CRC or hash code for additional security [41, 106, 112].

5.3 Proposed solution

The techniques proposed in this chapter are entirely decoder-driven so that the complexity of the encoder remains fairly unaffected.

To perform mode decision, typically, two approaches can be used. A first approach is to apply mode decision at the block level. Each WZ frame is partitioned into blocks and each block is classified into one of several classes (e.g., skip, WZ, intra). The advantage of this technique is that spatial variations can be taken into account. Regions well predicted by the decoder can be skipped, requiring no rate, while other regions can be WZ or intra coded. A second approach is to classify coefficient bands and bitplanes instead of blocks. The advantage of such a strategy is that high frequency information can often be discarded, as well as the most significant bitplane(s) of a coefficient band, which are often well predicted by the decoder. Since both techniques have their advantages, in this chapter, a coefficient band/bitplane-level approach is adopted, which is combined with a (block-like) coefficient classification method to differentiate spatially.

At coefficient level (Section 5.3.1) a conservative coefficient-band skip criterion is used, deciding between skipping the entire coefficient band or not. If the coefficient band can not be skipped, one or more bitplanes are decoded. At the bitplane level (Section 5.3.2), the decoder is granted the choice between three different coding modes for bitplane coding. A first option is to skip the bitplane. Secondly, the bitplane can be coded in a WZ fashion, by using a turbo coding procedure with selective early stopping and rate request criteria. A third option is to use a conventional arithmetic coder. In that case, the bitplane is coded and decoded without using the side information or any other information, being hence referred to as a bitplane intra mode. While this mode has a theoretical minimal rate equal to the entropy of the bitplane (which can never be lower than the conditional entropy, i.e., given Y), the main idea is that this mode can be used in case the WZ coding mode is inefficient, for example, due to poor correlation noise estimation $f'_{X|Y}$ or poor side information.

5.3.1 Coefficient band coding

The theoretical results in Figure 5.1 illustrate that the rate-distortion performance of quantizing and coding a coefficient depends on the position of the side information in the quantization bin. Some points on the rate-distortion curves seem much less interesting, in the sense that a reasonable amount of bits is spent while the decrease in distortion seems rather limited. In these cases, it could have been better to skip the coefficient and spend no rate at all.

This trade-off can be expressed as a Lagrangian cost, using the results from Eq. 5.2 and Eq. 5.3. As such, the cost to WZ code a coefficient X_i at index i in the coefficient band is defined as:

$$C^i_{WZ} = H(Q(X_i)|Y_i = y_i) + \lambda \cdot E[|X_i - X'_i||Y_i = y_i], \tag{5.4}$$

while skipping the coefficient and using the side information as a reconstruction results in a cost of

$$C^i_{skip} = \lambda \cdot \frac{1}{\alpha}. \tag{5.5}$$

The Lagrange parameter λ has been obtained through experiments on several sequences. First, the optimal Lagrange parameter has been determined offline, for each sequence and quantization level. Next, a curve has been fitted to these results, delivering the following formula for the Lagrange parameter:

$$\lambda = 7.6e^{-0.1 \cdot IQP}, \tag{5.6}$$

where IQP indicates the intra quantization parameter applied for the intra coded frames.

From a theoretical point of view, it seems advantageous to skip the entire coefficient band in case

$$\sum_i C^i_{skip} \leq \sum_i C^i_{WZ}. \tag{5.7}$$

However, accounting for practical uncertainties, a more conservative coefficient band skipping method is proposed for use in practice, so that a coefficient band is skipped only if

$$C^i_{skip} \leq C^i_{WZ}, \forall i. \tag{5.8}$$

If this criterion is fulfilled, all bitplanes are skipped and decoding proceeds with the next coefficient band. If this criterion is not fulfilled, one or more bitplanes are decoded as explained in the following section.

5.3.2 Bitplane coding

The second level after the coefficient band level is the bitplane level. A labeling strategy is used to label coefficients (and their corresponding bits in the bitplane) as *relevant* and *non-relevant*. The *relevant* coefficients are those coefficients for which spending parity bits is necessary, i.e., for which $C^i_{skip} > C^i_{WZ}$. The *non-relevant* coefficients are the remaining ones, i.e., for which $C^i_{skip} \leq C^i_{WZ}$. These coefficients should be ignored in the decoding process – as much as possible – since it is better to skip them.

Decoding is considered successful if the relevant bits have high reliability, i.e., a probability of being correct greater than or equal to a confidence threshold T (set to 90% in our experiments).

After describing how the decoder chooses between the different modes (Section A.), details are provided for bitplane skip (Section B.), bitplane WZ mode (Section C.), and bitplane intra mode (Section D.).

A. Bitplane mode decision

If all relevant bits in the current bitplane show high reliability, the bitplane is not coded, i.e., it is skipped.

If the bitplane is not skipped, a choice is made between intra coding and WZ coding. This choice is based on the best coding option for the co-located bitplane in the previous frame F, in hierarchical order[3]. This is inspired by the results from offline experiments, indicating high probability that the best coding mode in the temporally-equivalent frame F will also be the best in the current frame G (Figure 5.2).

In an online setup, only one coding mode has been used for coding a certain bitplane in F. Hence, the result of coding this bitplane using other modes needs to be estimated. For example, if the intra mode has been used, one needs to estimate bit rate and distortion for the WZ mode. This estimation can be limited to bit rate only, since the distortion of the relevant coefficients is similar for each mode.

The following algorithm is used at the decoder for estimating the rate for coding bitplane k in F, denoted BP_k^F, given the decoded bitplane $BP_k'^F$:

- if BP_k^F was WZ coded, the intra rate is calculated by intra coding $BP_k'^F$,
- if BP_k^F was intra coded, the WZ rate is calculated by turbo coding and decoding of $BP_k'^F$.

Given these bit rate estimations for the previous frame F, mode decision for the current frame G is performed as follows. If BP_k^F was skipped, the correlation model is considered accurate. Therefore, the WZ mode is preferred over the intra mode for coding BP_k^G. If BP_k^F was not skipped, the (possibly estimated) intra rate is compared to the (possibly estimated) WZ rate. If the intra rate is less than the WZ rate, the intra mode is selected for BP_k^G, and vice versa.

[3]For example, consider a GOP of size four: $I_1 - WZ_2 - WZ_3 - WZ_4 - I_5 - WZ_6 - WZ_7 - WZ_8 - I_9$. Since a hierarchical GOP structure is used [39], the mode decision process for bitplane k in WZ_7 will employ the mode used to code bitplane k in WZ_3, while for WZ_8 it will use the results from WZ_6.

Now that an algorithm for mode decision has been defined, details are provided for the different coding modes, starting with bitplane skip.

B. Bitplane skip

In the case of bitplane skip, the side information is already considered to be accurate enough. Hence, no bits are sent from encoder to decoder.

C. Bitplane WZ coding

If the bitplane needs to be WZ coded, a turbo coding strategy is applied, similar to existing solutions in the literature. However, the criteria that are used at the decoder during this process are evaluated only on a subset of the bits in the bitplane, instead of all. This subset Ω contains the bits corresponding to relevant coefficients, as well as some additional elements. The latter are the bits corresponding to irrelevant coefficients that have high reliability before starting to decode the bitplane. This condition is used to compensate for inaccuracies in the correlation noise estimation. It has no effect if the correlation noise estimation is accurate.

Two criteria are used for defining when to stop the iterative turbo decoding process. Firstly, an early stopping criterion is used that is only evaluated for elements of Ω. If the probabilities of these elements remain fairly constant over four iterations (which is called *bit convergence*), the turbo decoding process is stopped. Secondly, if the early stopping criterion is not triggered, the turbo decoding process is stopped if a maximum of 20 decoding iterations has been reached.

If the turbo decoding process stops, the decoder determines if more bits need to be requested from the encoder through a selective rate request strategy. More specifically, additional parity bits are requested from the encoder if there are still bits in Ω for which convergence is not reached, or if they have a reliability less than T.

It is important to remark that – after bitplane WZ decoding – some of the non-relevant bits might have a reliability less than T. This is in contrast to existing techniques in the literature where decoding is performed until all bits are reliably decoded [84, 99, 113]. This will be discussed further when performing coefficient reconstruction.

D. Bitplane intra coding

If the bitplane needs to be intra coded, a binary arithmetic entropy coder is used. This coder operates with an adaptive model that is initialized to a uniform

distribution before each bitplane is coded.

At the decoder, intra coded bitplanes can be decoded without using the side information. In contrast to the WZ mode, all decoded bits can be considered reliable.

5.3.3 Coefficient reconstruction

The goal of the reconstruction process is to select – for each coefficient – one particular value as a decoded value. In conventional approaches, all bits are reliably decoded so that the reconstruction process consists of selecting one particular value in the decoded quantization bin. In the system proposed here, in general, one employs an entire set of quantization bins instead of only one value. This is a consequence of the skip mode and the WZ mode, where only the bits of the relevant coefficients can be considered reliable after decoding. The bits of the non-relevant coefficients might have been corrected as well, as a side-effect, but there is no guarantee. Hence, in general, one employs a whole set S_i of possible quantization bins that can contain the coefficient X_i at index i in the coefficient band. The way such a set is constructed is explained next.

Prior to decoding the first bitplane, S_i is initialized to contain all bins (e.g., 2^M bins when using a quantizer with 2^M levels). Next, after each bitplane is processed, the number of elements in S_i is reduced if the corresponding bit in the bitplane is considered reliably decoded. For example, if the third bit of the coefficient at position 78 in the coefficient band is reliably decoded as being '1', then the quantization bins having '0' in the third bit will be removed from S_{78}. If the corresponding bit is not reliably decoded, S_{78} is left untouched. This elimination process is performed for all coefficients, both relevant and non-relevant.

Finally, after processing the last bitplane (and updating S_i accordingly), the coefficient is reconstructed using centroid reconstruction [60]:

$$x'_i = \frac{\sum_{q_k \in S_i} \int_{q_k^L}^{q_k^H} x_i \cdot f'_{X_i|Y_i}(x_i|y_i) dx_i}{\sum_{q_k \in S_i} \int_{q_k^L}^{q_k^H} f'_{X_i|Y_i}(x_i|y_i) dx_i}, \tag{5.9}$$

where q_k^L and q_k^H denote the low and high border of the quantization bin q_k, respectively.

Remark that centroid reconstruction is more accurate than the reconstruction technique used for obtaining Eq. 5.3, so that the average distortion is slightly overestimated.

5.4 Test setup

The techniques proposed in this chapter have been used to extend the codec developed in Section 4.3. These extensions will be discussed using the block diagram depicted in Figure 5.3.

The encoder's extensions are limited, which is in favor of the DVC scenario featuring low-complexity encoders. Only an intra encoder is added so that – depending on the information received from the decoder – all bitplanes in the coefficient band are either (1) discarded, (2) only the current bitplane is discarded, (3) the current bitplane is intra coded, or (4) WZ coded.

At the decoder, the side information Y and the estimated correlation $f'_{X|Y}$ are used to estimate the bit rate (Eq. 5.2) and the distortion (Eq. 5.3). Next, mode decision is performed using the techniques discussed in this chapter. The mode information is communicated to the encoder using the feedback channel. The overhead of sending the coding modes to the encoder is negligible[4]. If the WZ bitplane coding mode is selected, the turbo decoder uses the transformed side information for decoding. If the bitplane is intra decoded using the arithmetic decoder, the side information is not used. When all bitplanes are decoded, the coefficients are reconstructed, and the inverse DCT is applied to the results, delivering the decoded frame W'. This frame can then be used for side information generation of future frames to be decoded.

5.5 Results

Tests have been conducted on three sequences: Mother and Daughter, Table Tennis and Foreman. All sequences have CIF resolution, at a frame rate of 30 Hz. Only the luma component is coded, allowing for comparison with the DISCOVER codec [51].

First, the gains realized by introducing bitplane skip and bitplane intra mode are investigated, by comparing different configurations of our system (Section 5.5.1). Next, the performance of our system is evaluated against comparable systems found in the literature (Section 5.5.2).

5.5.1 Studying the gains realized by intra and skip

To illustrate the gains obtained by the different coding modes, results have been generated for a GOP of length four, for a number of different configura-

[4]For example, 16 luma coefficient bands each quantized to 5 bits results in 80 modes to be communicated. Using two bits to signal skip, WZ, or intra results in a marginal additional bit rate of 3.6 kbps for a GOP of size 4 at 30 fps.

tions. A first configuration is the system proposed in Section 4.3. This system has been taken as a basis in this chapter. It supports bitplane WZ decoding and bitplane skip based on a threshold on the bit probabilities. A second configuration is obtained by using the system proposed in this chapter, but excluding the intra mode. This system is labeled as *RD-based WZ + skip*. The third configuration, labeled *RD-based WZ + intra + skip*, supports all modes proposed in this chapter.

Comparing rate-distortion performance of the different configurations (Figure 5.4) indicates that most of the gain at low rates is due to the skip mode, while the gains at high rates are mostly due to the bitplane intra mode. This is rather expected. At low rates, for large quantization bins, the decoder is often able to predict to correct bin, so that the information can be skipped. At high rates, predicting the correct bin becomes difficult. In addition, inaccuracies in the correlation noise model make the WZ mode less efficient, especially at high rates where the accuracy of the estimation of the bin probabilities decreases. As a result, the intra mode is selected more often.

More details concerning the mode decision are provided in Table 5.1, for different sequences. For each sequence, results for the first eight coefficient bands (labeled CB0 to CB7) in raster scan are provided, from fine to coarse quantization (Q2 to Q5, respectively). Analyzing the occurrence of each mode, we can indeed see that the skip mode is selected more often at low rates, while the intra mode is more often selected at high rates. Across coefficient bands, we observe that high frequency information is often skipped. More skip is used for sequences that can be well predicted by the decoder, such as Mother and Daughter. On the other hand, for difficult sequences, more intra is used.

5.5.2 Rate-distortion results compared to other systems

The system is compared to a number of reference systems. A first reference system is the DISCOVER codec, which can still be considered among the state-of-the-art in DVC. We also compare with the best performing conventional video compression scheme, i.e., H.264/AVC. It is also popular in DVC literature to compare with a low-complexity version of H.264/AVC, so-called H.264/AVC *no motion*. In this configuration, we use the same settings as for H.264/AVC inter coding, but exclude computationally expensive motion estimation at the encoder by setting the search range to zero. This way, the computational complexity of the encoder is drastically reduced, making it comparable to the complexity of the DVC encoder. Finally, we also compare with H.264/AVC intra coding.

Results for GOP sizes two (Figure 5.5), four (Figure 5.6), and eight (Fig-

ure 5.7) are presented. We will discuss these results by comparing our system with each of the reference systems in turn.

DISCOVER is outperformed significantly. This is due to the improved correlation noise model proposed in Section 4.3 as well as the addition of several coding modes as proposed in this chapter. For example, for a GOP of size four, average Bjøntegaard [102] rate improvements are achieved of 28% for Foreman, 13% for Table, and 24% for Mother and Daughter. The largest gains over DISCOVER are observed for those sequences where either the skip mode or the intra mode is used a lot, as in Mother and Daughter, and Foreman, respectively (see Table 5.1). This shows that the use of additional non-WZ modes can really boost performance.

To compare with H.264/AVC inter coding, we have used similar settings as for our system (i.e., same fixed GOP size, hierarchical coding, only two reference frames used). The extended profile was used, having one slice per frame. As illustrated by the results, there is still a performance gap between our (DVC) system and conventional video compression with encoder-side motion estimation, for large GOP sizes. Nonetheless, compared to a similar DVC system such as DISCOVER, we have made significant progress and narrowed the gap between conventional video coding and DVC with several decibels. For low motion sequences such as Mother and Daughter, we even get similar results at low bit rates.

A comparison with H.264/AVC no motion is often provided in the literature as this configuration shows similar encoding complexity, while still being difficult to beat for DVC systems. Our system is able to outperform H.264/AVC no motion merely everywhere, except for the Table Tennis sequence. This is probably caused by inaccuracies in estimating the parabolic motion of the ball, and zooming of the camera. The techniques used here still assume linearity of motion between the reference frames used for side information generation. While we have proposed a technique to compensate for this effect in Chapter 4, we believe that there is still work left in this area.

It is interesting to notice for the Foreman sequence that the performance of the DISCOVER system gets worse as the GOP size increases. While DISCOVER performs better than H.264/AVC intra coding for a GOP of size two, performance is only comparable for a GOP of size four, and even worse than intra for a GOP of size eight. This is due to the fact that the average quality of the side information decreases with the GOP size. For example, for GOP of size eight, the side information for the middle WZ frame will be generated using two decoded I frames at a distance of 8 apart (i.e., 7 WZ frames in between). Due to this distance this side information will typically be of bad quality, requiring a lot of error correcting bits from the encoder during

the decoding process. To illustrate the performance of the WZ frames as a function of the GOP size, we have constructed graphs containing results for the WZ frames only. The be able to compare the curves, the WZ bit rates have been normalized, by first calculating the average number of bits per WZ frame and multiplying the result by 30 (i.e., the total number of frames per second). The results provided in Figure 5.8 confirm that the average compression performance of the WZ frames decreases as the GOP size increases, as expected. For more information about the impact of the GOP size on the compression performance of the DISCOVER system, the reader is referred to the literature [58].

In contrast to DISCOVER, the compression performance of the WZ frames in our system does not suffer that much, as shown in Figure 5.9. Although there is definitely a decrease in performance when comparing the results for a GOP of size 2 to a GOP of size 4, it is important to notice that the performance drop for a GOP of size 8 is much smaller. This is due to the effectiveness of the bitplane intra mode, which enables to compensate for some performance losses by switching to more efficient bitplane coding techniques. DISCOVER, however, does not apply such a scheme and so it is stuck with inefficient WZ coding for these bitplanes at all times. As a consequence, our system is able to consistently outperform H.264/AVC intra coding in contrast to DISCOVER, even for a GOP of size eight (Fig. 5.7).

5.6 Conclusions and original contributions

In this chapter closed-form expressions for rate and distortion in DVC have been derived that take into account the position of the side information in the quantization bin. These formulas indicate that a significant coding penalty is paid if the side information lies close to the border of the quantization bin. A number of approaches can be used to compensate for this effect. The approach followed in this chapter is to use a Lagrange cost function for mode decision, skipping cases of inefficient WZ coding.

Apart from the coding loss caused by inefficient quantization, a second issue addressed in this chapter is the fact that the side information Y and correlation noise estimation $f_{X|Y}$ can be inaccurate. For these cases, the decoder is given the possibility to select a bitplane intra coding mode, avoiding the use of unreliable Y and $f_{X|Y}$. The results indicate that this strategy improves significantly the coding performance. On the one hand, this illustrates that there is a need for further improving of side information generation and correlation noise estimation in DVC. On the other hand, this shows that adding additional coding modes can have a significant impact, improving the performance in

DVC, without the need of performing motion estimation at the encoder side.

The author's work on mode decision has led to the following publications:

- Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Rate-distortion driven decoder-side bitplane mode decision for distributed video coding. *Signal Processing: Image Communication*, 25(9):660 – 673, October 2010.
- Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Bitplane intra coding with decoder-side mode decision in distributed video coding. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages 3733 – 3736, September 2010.
- Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Distributed video coding with decoder-driven skip. In *Proc. Mobimedia*, September 2009.
- Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Decoder-driven mode decision in a block based distributed video codec. Accepted for publication in: *Multimedia Tools and Applications*, 2010.
- Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Introducing skip mode in distributed video coding. *Signal Processing: Image Communication*, 24(3):200–213, March 2009.

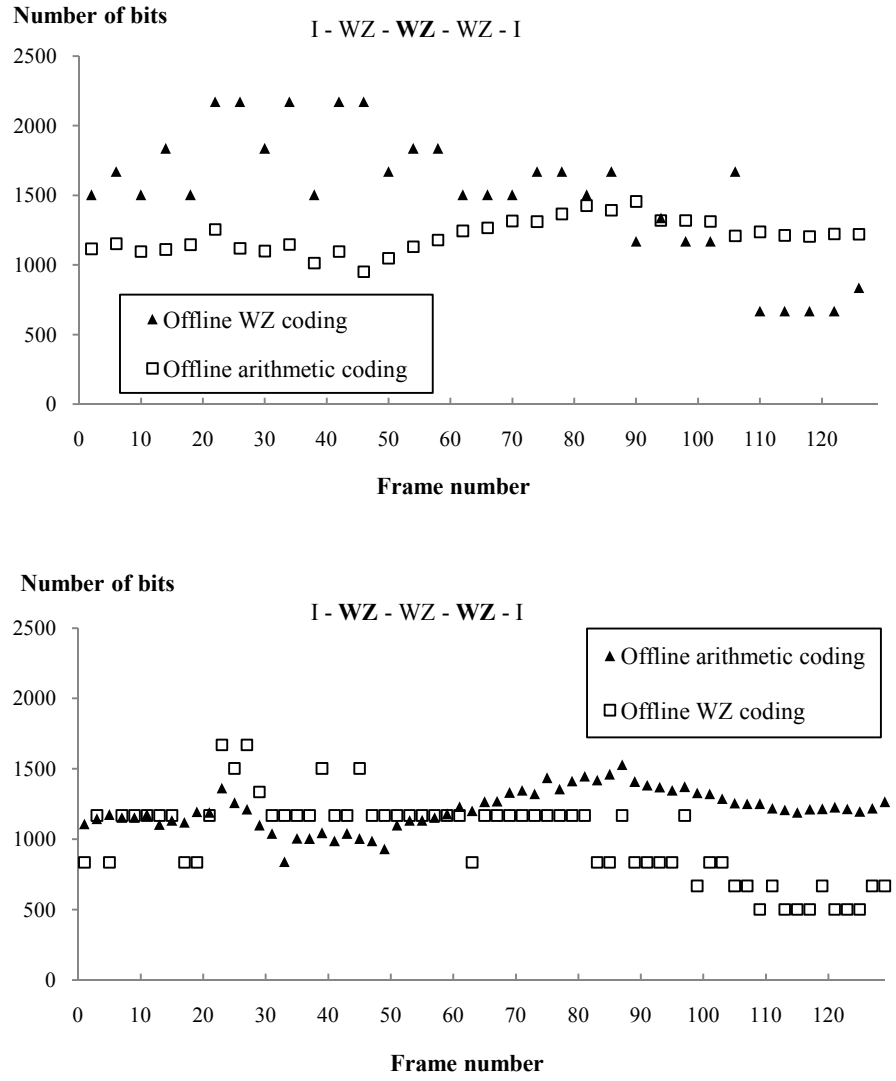

Figure 5.2: Results for the least significant bitplane of the luma DC for the Table Tennis sequence. In an offline setup, for each frame, bitplanes are coded using both the intra mode, and the WZ mode, delivering two rate points per frame index. The graphs depict the results from two hierarchical layers of WZ frames. These results indicate high probability that the best mode of the current bitplane is the same as in the temporally-equivalent frame located either in the previous or current GOP.

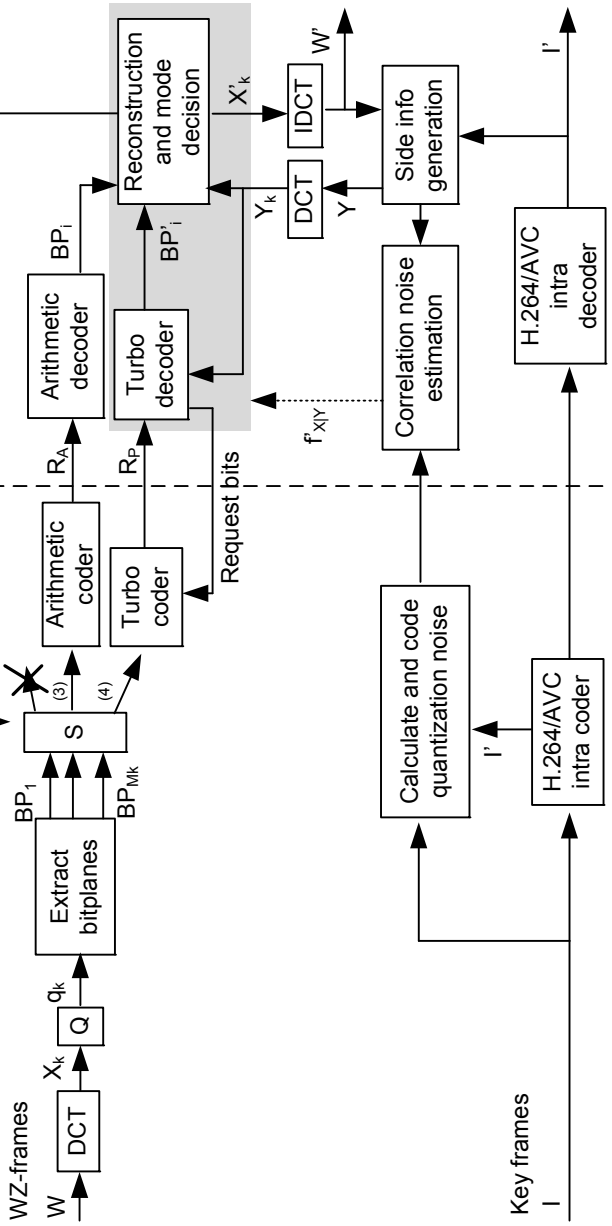

Figure 5.3: Codec architecture, featuring several modes for coding bitplanes.

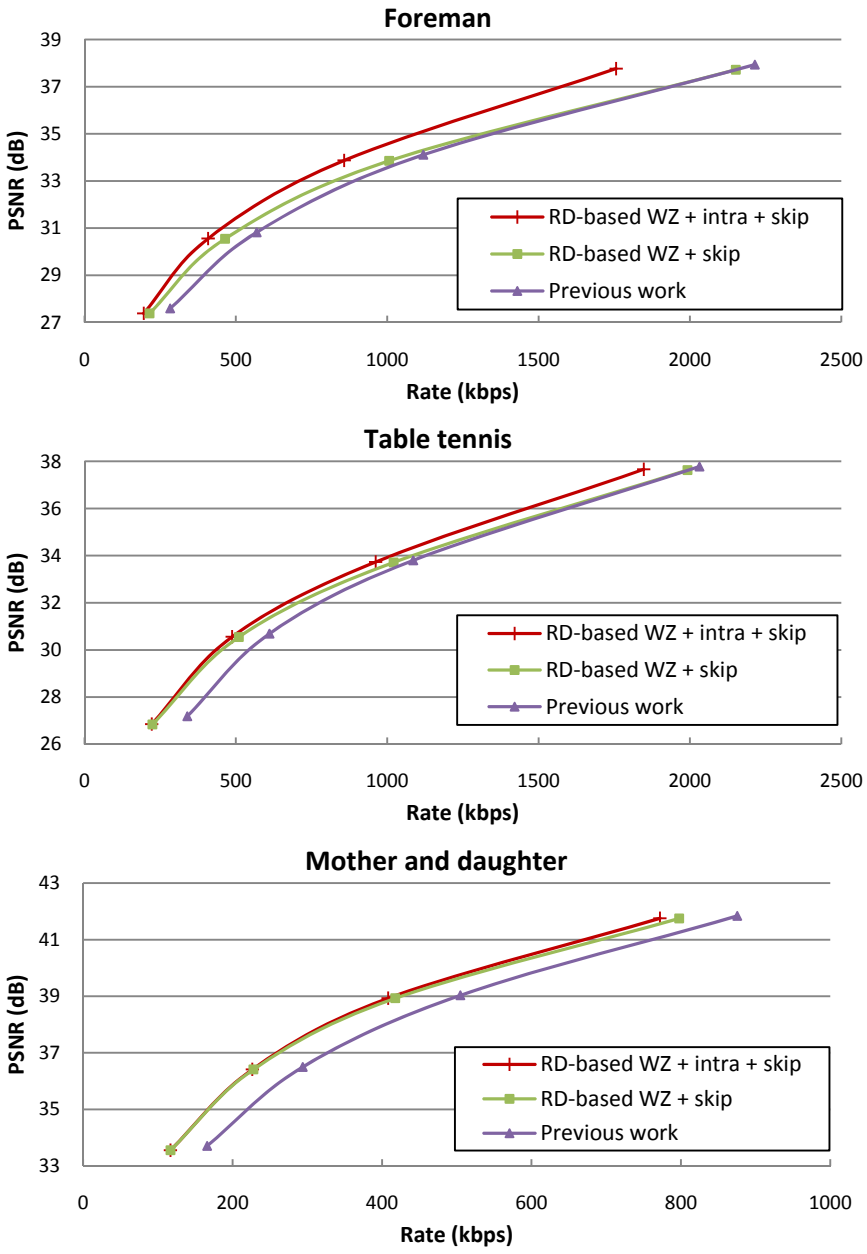

Figure 5.4: Comparing the system proposed in this chapter to the same configuration applying only WZ and skip mode, and to our previous work (in Section 4.3) [114].

Table 5.1: Average percentage of modes used for different coefficient bands (i.e., CB0 to CB7) and different quantization levels (i.e., Q2 to Q5).

FOREMAN

		CB0	CB1	CB2	CB3	CB4	CB5	CB6	CB7
	Skip	0%	6%	30%	43%	9%	28%	41%	53%
Q2	WZ	68%	54%	27%	12%	43%	37%	29%	9%
	Intra	32%	40%	42%	45%	48%	35%	29%	38%
	Skip	0%	14%	46%	63%	16%	45%	65%	78%
Q3	WZ	74%	56%	24%	7%	46%	33%	14%	6%
	Intra	26%	30%	30%	30%	38%	22%	20%	17%
	Skip	0%	41%	74%	84%	36%	68%	86%	92%
Q4	WZ	82%	40%	7%	5%	38%	16%	4%	3%
	Intra	18%	19%	20%	12%	26%	17%	10%	5%
	Skip	5%	78%	93%	97%	71%	91%	100%	100%
Q5	WZ	79%	10%	3%	1%	11%	4%	0%	0%
	Intra	16%	11%	4%	2%	18%	6%	0%	0%

TABLE TENNIS

		CB0	CB1	CB2	CB3	CB4	CB5	CB6	CB7
	Skip	0%	6%	19%	36%	3%	16%	28%	43%
Q2	WZ	85%	69%	63%	50%	63%	58%	52%	40%
	Intra	15%	24%	18%	14%	35%	26%	20%	17%
	Skip	0%	8%	27%	44%	4%	22%	41%	59%
Q3	WZ	90%	72%	59%	43%	63%	55%	44%	29%
	Intra	10%	20%	14%	13%	33%	23%	15%	12%
	Skip	0%	18%	47%	69%	10%	46%	72%	91%
Q4	WZ	92%	64%	44%	20%	61%	40%	22%	6%
	Intra	8%	18%	9%	11%	30%	15%	6%	2%
	Skip	6 %	60%	96%	99%	49%	93%	99%	100%
Q5	WZ	88%	35%	3%	1%	37%	5%	1%	0%
	Intra	7%	5%	1%	0%	15%	2%	0%	0%

MOTHER AND DAUGHTER

		CB0	CB1	CB2	CB3	CB4	CB5	CB6	CB7
	Skip	1%	32%	55%	74%	29%	52%	72%	87%
Q2	WZ	97%	55%	33%	21%	67%	19%	9%	4%
	Intra	2%	13%	12%	5%	3%	29%	18%	8%
	Skip	4 %	50%	79%	93%	50%	77%	93%	98%
Q3	WZ	96%	35%	14%	3%	49%	5%	3 %	1%
	Intra	0%	14%	8%	4%	1%	17%	4 %	1%
	Skip	22%	78%	97%	99%	81%	97%	100%	100%
Q4	WZ	78%	13%	2%	1%	16%	2%	0%	0%
	Intra	0 %	9%	1%	0%	3%	1%	0%	0%
	Skip	64%	99%	100%	100%	100%	100%	100%	100%
Q5	WZ	36%	1%	0%	0%	0%	0%	0%	0%
	Intra	0%	0%	0%	0%	0%	0%	0%	0%

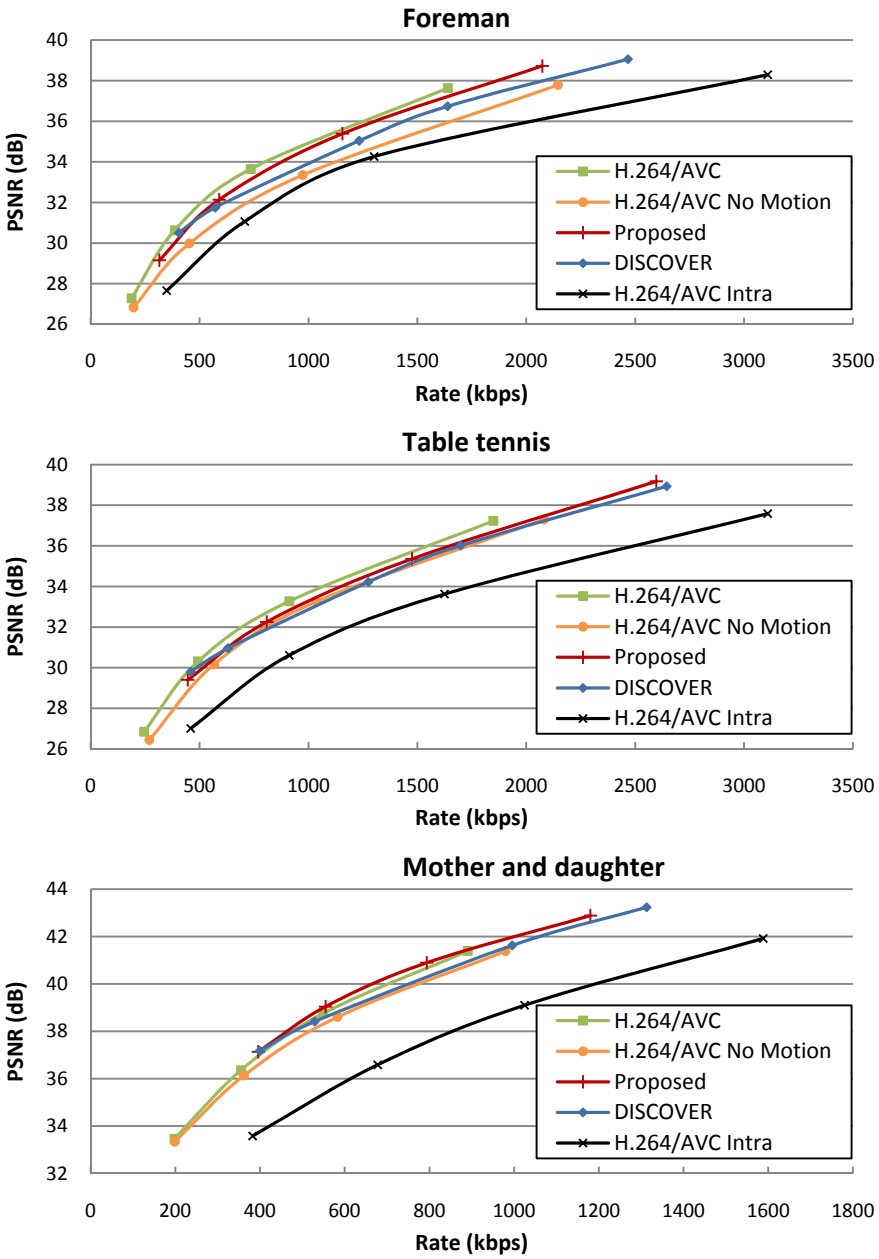

Figure 5.5: Rate-distortion results for a GOP of size two.

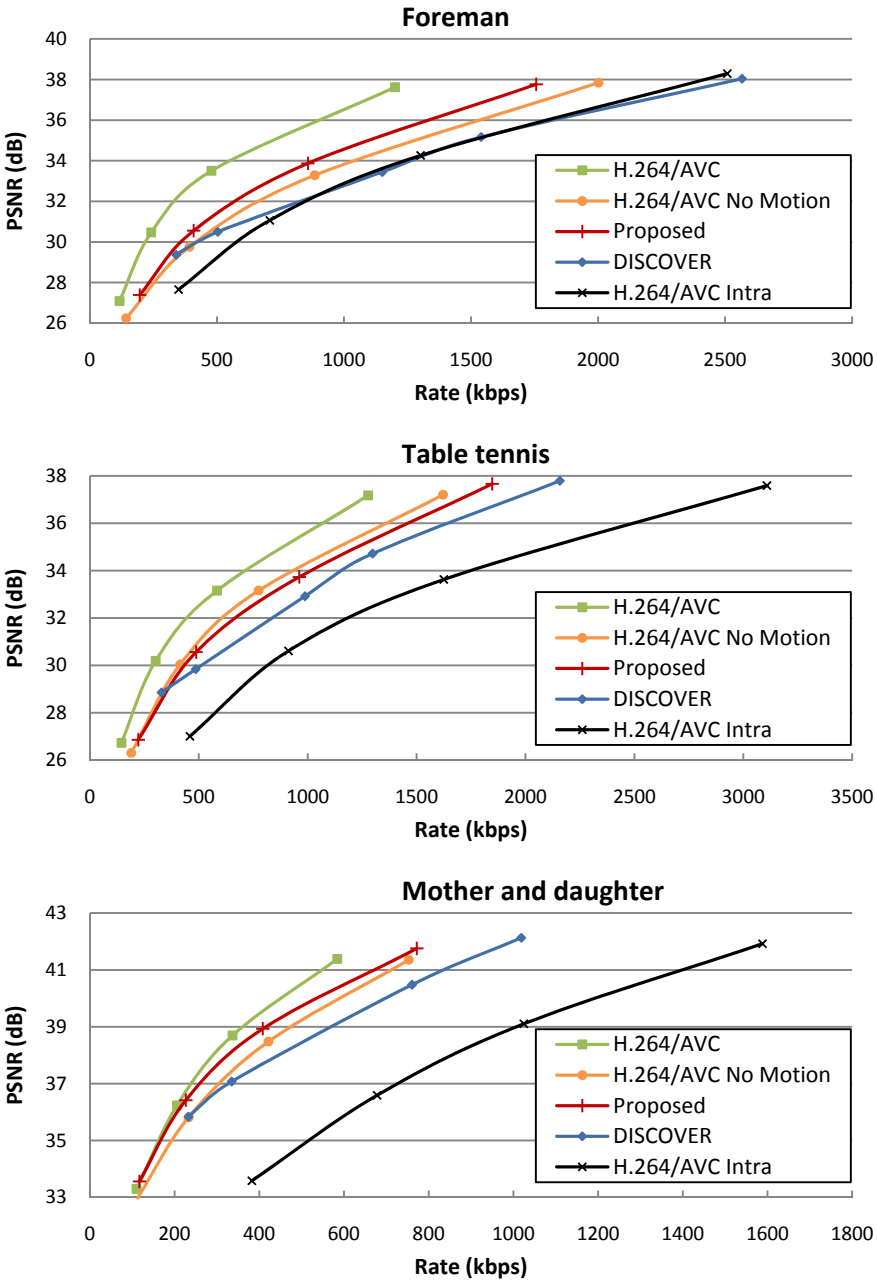

Figure 5.6: Rate-distortion results for a GOP of size four.

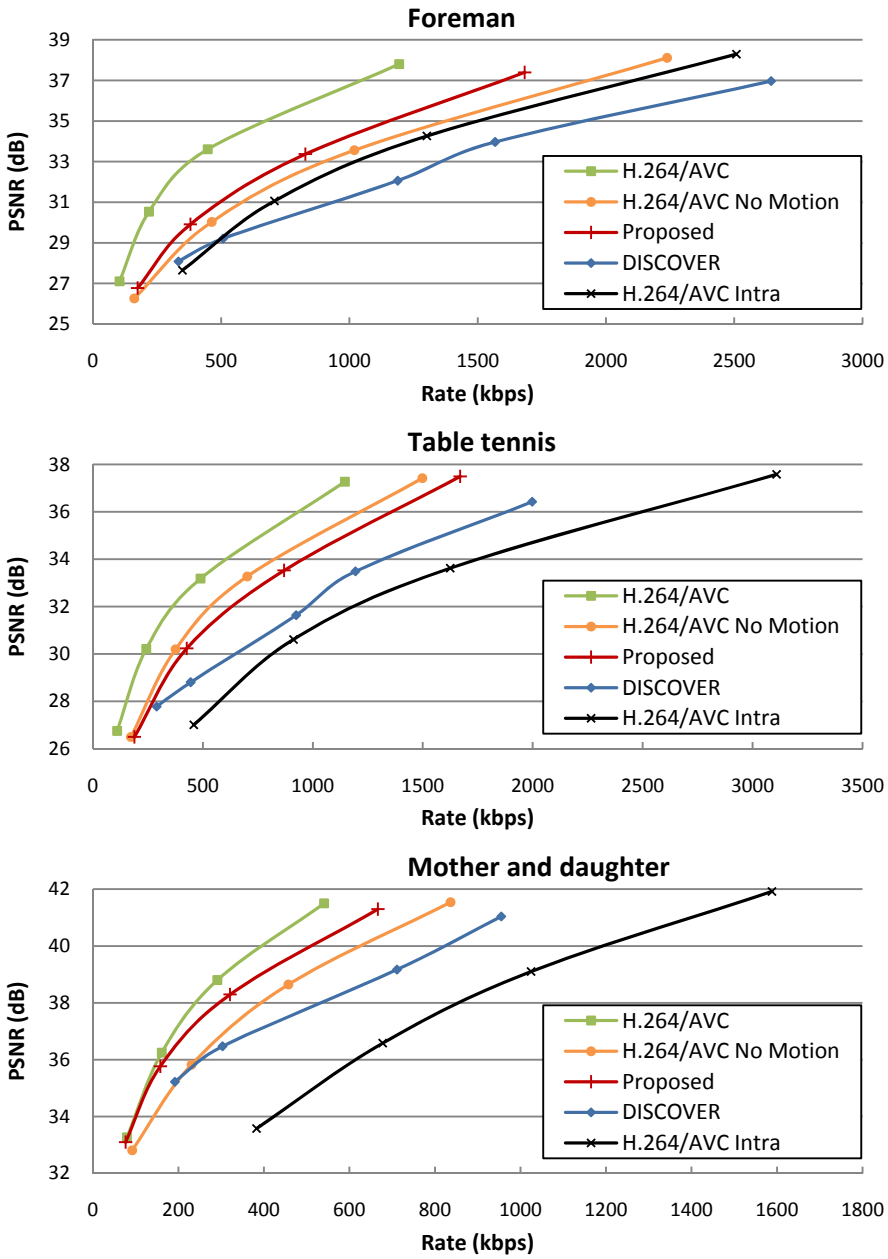

Figure 5.7: Rate-distortion results for a GOP of size eight.

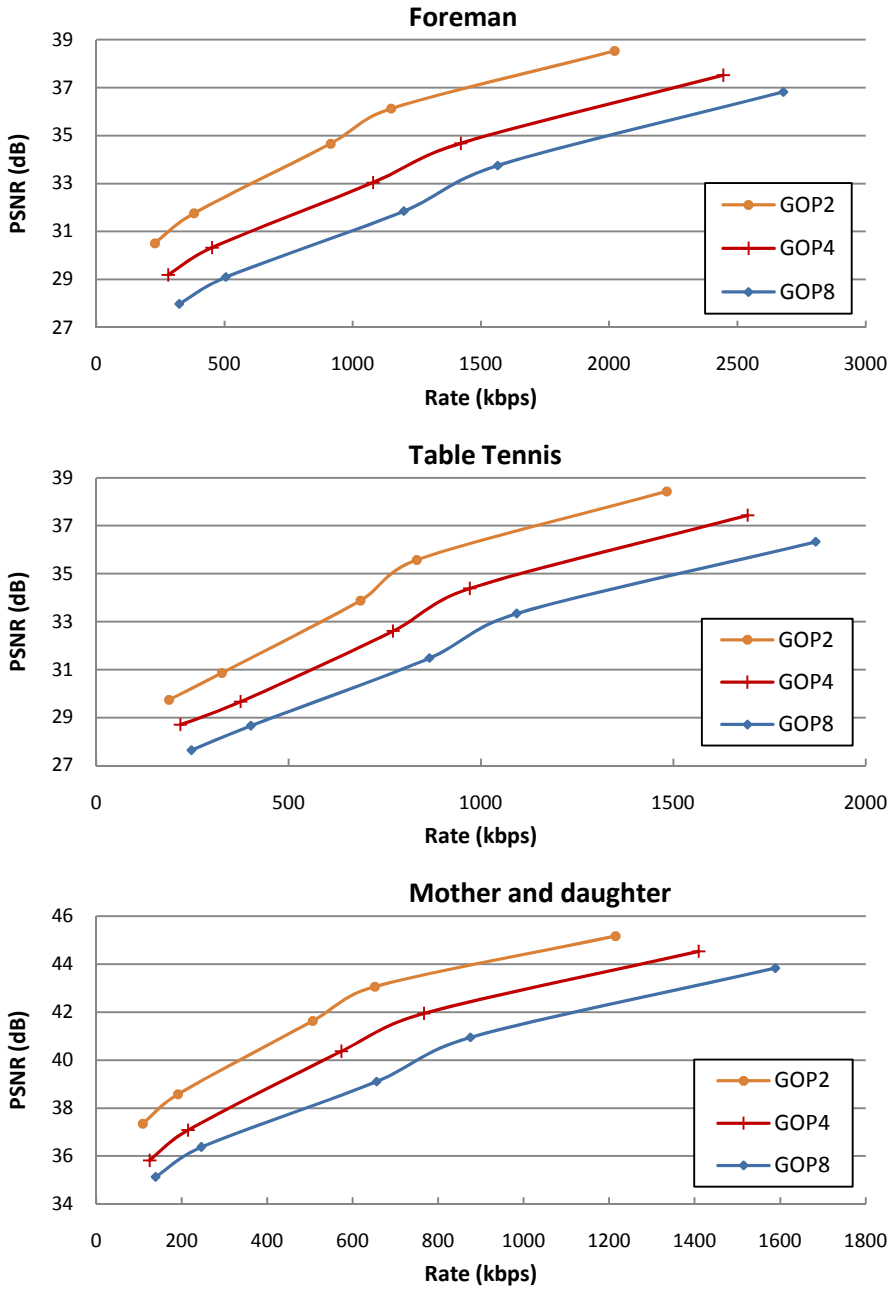

Figure 5.8: Normalized rate-distortion results for the WZ frames coded using DISCOVER.

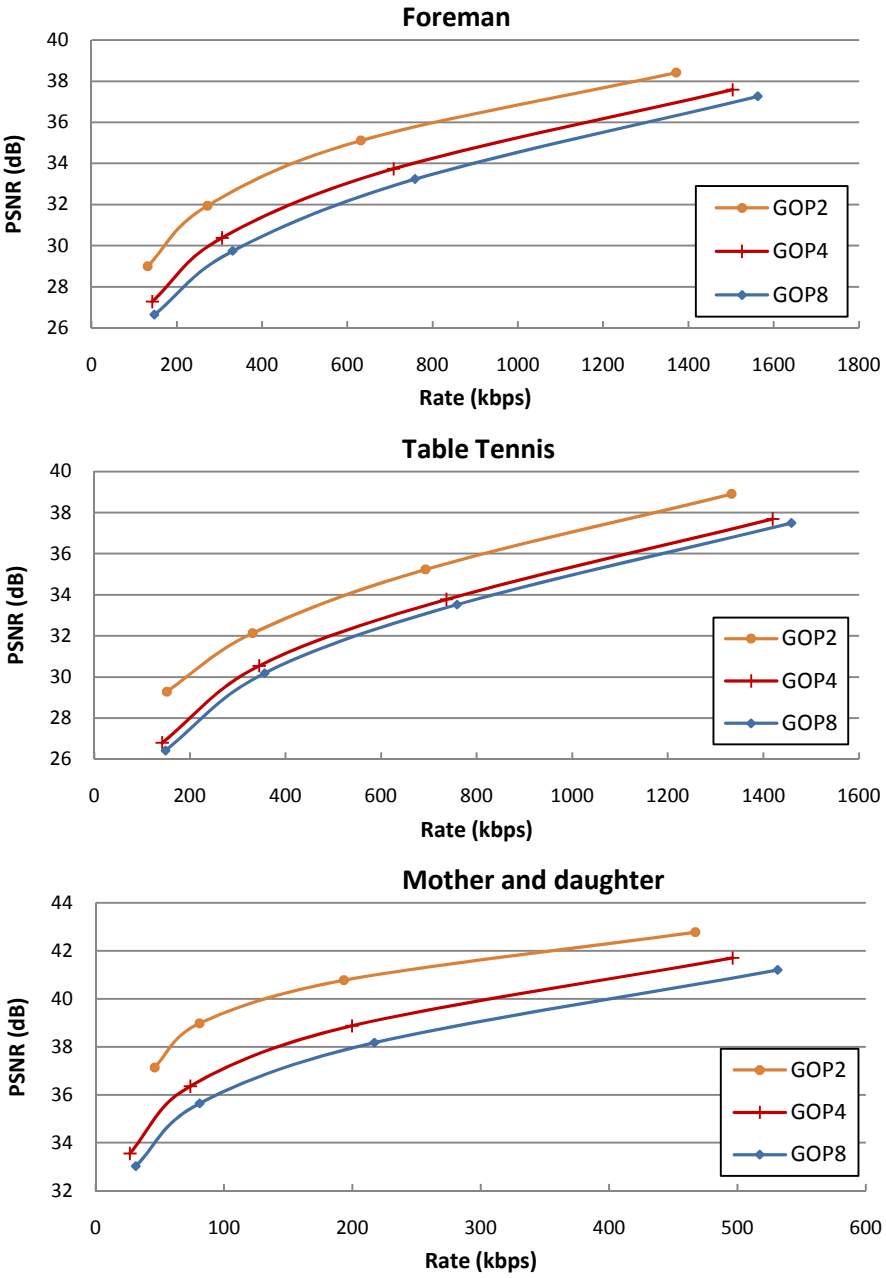

Figure 5.9: Normalized rate-distortion results for the WZ frames coded using the system proposed in this chapter.

Chapter 6

Conclusions

Video compression has always been an important area of research due to practical limits on the amount of information that can be stored, processed, or transmitted. Conventional video coding systems show a complexity imbalance, with an encoder that is significantly more complex than the decoder. On the other hand, during the past decade DVC has emerged as a new video coding paradigm, featuring simple encoders at the cost of a complex decoder.

Given these two extremes, a general architecture was presented in Chapter 3. This architecture was able to operate as a conventional video coding system, as a DVC system, or as an intermediary system, by shifting and/or sharing the complex task of motion estimation between the encoder and the decoder. By defining several modes for coding frames, the distribution of complexity between the encoder and the decoder was made dynamic. If the encoder had more computational resources, it could take over some of the workload from the decoder and vice versa.

Such a flexible video coding system can be particularly useful in environments where computational complexity varies over time, for example, in the case of multi-tasking, variable power supply, or session mobility. Our flexible video coding system can also be used in cases where available complexity remains static as it matches computational requirements to the amount of resources available. This way, it offers a solution for a high number of situations, including cases where standard video coding can not be applied directly.

A number of problems arose when evaluating the architecture described in Chapter 3. A first category of problems was related to complexity modeling and adaptation. For simplicity, we had assumed that complexity constraints are readily available at the encoder and decoder. In practice, it will often be necessary to define such constraints during the coding-decoding process, by monitoring current resources (such as battery status, amount of CPU-time

available, etc.) and predicting future behavior. After communicating these constraints from the decoder to the encoder, the encoder needs to decide upon the set of modes to use for coding the next group of pictures. The way this was handled in Chapter 3 was by considering estimated complexity only, and a clear extension of this technique would be to incorporate compression performance. This would result in selecting the best mode based on a trade-off between complexity and rate-distortion performance. Finally, we also remarked in Chapter 3 that the complexity model could be further improved, for example by accounting for turbo (de)coding complexity.

A second category of problems was related to the performance of the system. Despite the advantage of being more flexible, the compression performance of our system could still be considered inferior to conventional solutions such as H.264/AVC. Increasing the performance of the different coding modes seemed therefore a necessary step to improve its competitiveness. Although some of the modes could be improved in a relatively straightforward manner by borrowing concepts from solutions such as H.264/AVC, improving the DVC mode seemed to be a more challenging task as this mode already represented the state-of-the-art.

Out of all remaining challenges associated with the developed system, the last problem was considered the most crucial. As such, the remainder of this dissertation has focused on improving compression performance in a DVC context. These improvements can be fed back to our flexible architecture in a later stage.

DVC is still quite a young research area, and compression performance can be increased by focusing on different components of the codec. The main focus in this dissertation has been on correlation noise modeling and (bitplane) mode decision, although other areas are worth investigating as well (as mentioned further on).

The correlation model in DVC describes the relationship between the original frame (available at the encoder), and its prediction (available at the decoder). If the decoder is capable of modeling the correlation more accurately, less parity bits are needed for reliable decoding. Hence, there is a strong relationship between the accuracy of the correlation model and the compression performance of the system.

Several techniques have been proposed in the literature to establish correlation models at the decoder's side. In Chapter 4, two main contributions were presented that improve upon current techniques found in the literature. A first improvement extends current techniques by compensating for quantization noise in the reference frames (used for generating the side information). An experimentally-derived formula was proposed, showing large im-

provements especially for low bit rates. This work has been taken further in co-authorship, in order to refine and justify it theoretically as well.

A second refinement of the correlation model addressed uncertainties in the generation of the side information. Although merely all known techniques assume that the motion between the references frames can be approximated as being linear, this assumption becomes less valid if the distance between the reference frames is large. To compensate for inaccuracies, the technique in Chapter 4 adopted multiple predictors per block, each taken from the neighborhood of the current block. As a result, in addition to being spatially adaptive the proposed correlation model was also temporally adaptive. This is an additional feature that is currently not supported by techniques found in the literature.

Improving the accuracy of the correlation model is only one way to improve compression performance in DVC. Another important topic – addressed in Chapter 5 – is the development of different coding modes. This research was motivated by the observation that a WZ strategy sometimes resulted in inefficient coding, due to poor-quality side information or inaccurate correlation modeling. Surprisingly, significant gains can be achieved by using bitplane entropy coding instead in such cases, disregarding the side information (and the correlation model). Inspired by these results we developed new techniques for bitplane-level mode decision in DVC, deciding between bitplane skip, bitplane WZ coding, and bitplane intra coding. A rate-distortion model was used to decide whether to skip a coefficient band or not, whereas the decision between bitplane WZ coding and bitplane intra coding was driven by online measurements. This enabled adapting to the statistics of the sequence, by fine-tuning the relation between the number of WZ coded bitplanes and the number of intra coded bitplanes.

When comparing the compression performance of the system in Chapter 5 to alternative DVC solutions such as DISCOVER, we can conclude that we have succeeded in creating a DVC system with increased compression performance. While the DVC mode of the flexible architecture in Chapter 3 was still comparable to DISCOVER, we are now outperforming this system significantly hereby realizing the main research goal in this dissertation.

Although we have narrowed the performance gap with conventional video coding solutions, we still believe that compression can be improved even further. Besides trying to extend the work described in this dissertation, future work should focus on other areas as well, such as the generation of high-quality side information, optimizing the channel codes in a DVC context, intelligent reference frame selection, adaptive GOP structures, etc. However, when lifting DVC compression performance to a higher level, the need for dealing with

practical issues increases as well.

Possibly the most important problem in practice is the use of a feedback channel from the decoder to the encoder in most DVC systems (including the one described in this dissertation). Clearly, storage applications can not support the use of a feedback channel. Some streaming scenarios might support a certain degree of feedback from the decoder to the encoder, but most likely not in the same way as it is assumed in most DVC-related contributions. Hence, we need to modify current high-performing solutions in order to make them more practical.

Developing feedback-free DVC architectures is a challenging task, since the encoder needs to estimate the bit rate without the support of the decoder. Moreover, the techniques used by the encoder should have low complexity, since using high-complexity techniques at the encoder would violate DVC use case scenarios. Due to this trade-off we believe that dealing with the feedback channel is one of the most challenging problems that need to be dealt with in future contributions.

Another problem in practice is the complexity of the turbo decoder. Currently, a lot of time is spent during turbo decoding, due to the iterative decoding process and the rate request strategy. To alleviate this problem, low-complexity alternatives need to be studied, as well as opportunities for parallelism. For example, in most systems, different coefficient bands are decoded independently from each other, enabling parallel decoding in such cases.

Finally, remark that improvements in the context of DVC can also be used in other video coding approaches as well. For example, H.264/AVC can be extended to incorporate DVC-like coding modes, leading to improved compression performance (as already shown in collaborative work [93]). New techniques can be fed back to the flexible architecture described in Chapter 3, or to other architectures such as multi-view systems. Therefore, various domains can benefit from new insights in DVC.

Appendix A

Using H.264/AVC transformation and quantization

The H.264/AVC transformation [115] is an approximation of the discrete cosine transformation (DCT). This approximation is computationally more efficient since it can be executed using integer operations only. For this reason, the H.264/AVC transformation has been integrated in the codec developed in the context of this dissertation.

This appendix provides more details about the H.264/AVC transformation, its impact on the quantization, and its integration in the codec.

A.1 Forward transformation

For a 4-by-4 input block X, the DCT is given by:

$$X_T = AXA^T = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \begin{bmatrix} & & & \\ & X & & \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix} a & b & a & c \\ a & c & -a & -b \\ a & -c & -a & b \\ a & -b & a & -c \end{bmatrix} \tag{A.1}$$

where:

$$a = \frac{1}{2}, b = \sqrt{\frac{1}{2}} \cos\left(\frac{\pi}{8}\right), c = \sqrt{\frac{1}{2}} \cos\left(\frac{3\pi}{8}\right). \tag{A.2}$$

This matrix multiplication can be approximated by [115]:

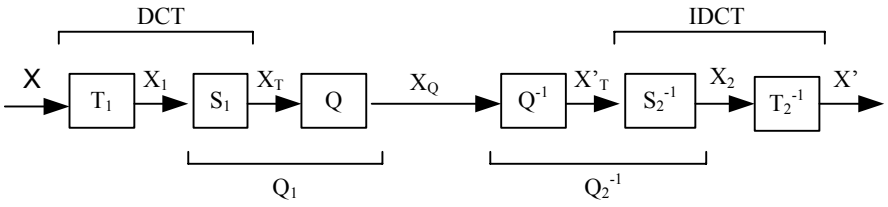

Figure A.1: The H.264/AVC transformation and quantization scheme.

$$X_T \approx T_1(X) \otimes S_1 =$$

$$\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} \\ X \\ \\ \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

where:

$$a = \frac{1}{2}, \text{ and } b = \sqrt{\frac{2}{5}} \tag{A.3}$$

and $\otimes$ indicates that the elements at corresponding positions need to be multiplied.

This result indicates that the DCT can be split up into a forward transformation T_1 and a scaling part S_1. As explained further on, this also applies to the inverse discrete cosine transform (IDCT), which can be split into a backward scaling operation S_2^{-1} and a backward transformation step T_2^{-1}. This relation is illustrated by Fig. A.1.

T_1 and T_2 can be executed using integer operations only, making the transformation computationally efficient since no floating point operations need to be performed. The scaling steps S_1 and S_2^{-1} can be integrated into the quantization operation, resulting into Q_1 and Q_2^{-1}, respectively. One of the consequences of this approach is that the backward transformation T_2^{-1} and quantization Q_2^{-1} are not the inverse operations of the forward transformation T_1 and quantization Q_1, respectively.

A.2 Quantization

While the input and output of the H.264/AVC transformation is a matrix, quantization is performed on scalars. So for ease of notation, consider x_T to be an arbitrary element in X_T. Forward quantization Q is defined as:

$$x_Q = round\left(\frac{x_T}{Q_{step}}\right), \tag{A.4}$$

where Q_{step} is the quantizer step size. To integrate the scaling step into the quantization, the operation for Q_1 becomes:

$$x_Q = round(\frac{x_T}{Q_{step}}) = round(\frac{x_1 \cdot PF}{Q_{step}}) = round(\frac{x_1 \cdot MF}{2^{qbits}}),$$

where:

$$\frac{MF}{2^{qbits}} = \frac{PF}{Q_{step}}$$

and

$$qbits = 15 + \left\lfloor \frac{QP}{6} \right\rfloor. \tag{A.5}$$

PF is the *post-scaling factor*, which is the corresponding element in S_1. QP is the *quantization parameter* that drives the quantization, and MF is the *multiplication factor*.

In integer arithmetic, the scaling-quantizing operation can be implemented as follows:

$$|x_Q| = (|x_1| \cdot MF + f) >> qbits$$
$$sign(x_Q) = sign(x_1) \tag{A.6}$$

where f is the deadzone constant that defines the width of the zero quantization bin[1]. This formula indicates that scaling and quantization can be performed by integer multiplication, addition and shift operations only. Hence, computationally expensive divisions are avoided. The multiplication factor MF is precalculated for each coefficient position, and stored in a table.

A.3 Backward transformation

The IDCT is approximated by a backward transformation T_2^{-1} and backward scaling S_2^{-1}:

[1] In the reference model software, f is defined as $2^{qbits}/3$ for intra blocks and as $2^{qbits}/6$ for inter blocks.

$$X' \approx T_2^{-1}(X'_T \otimes S_2^{-1}) =$$

$$\begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} \left(\begin{bmatrix} \\ X'_T \\ \\ \end{bmatrix} \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \right)$$
$$\cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix} \quad \text{(A.7)}$$

As before, the backward scaling step S_2^{-1} is integrated in the backward quantization step Q_2^{-1}.

A.4 Backward quantization

The backward quantization operation Q_2^{-1} is given by:

$$x_2 = x_q \cdot V \cdot 2^{\lfloor QP/6 \rfloor} \quad \text{(A.8)}$$

where V is called the *scaling factor* which is equal to $Q_{step} \cdot PF \cdot 64$. In the latter, the factor 64 is used to avoid rounding errors. This factor is removed again after the backward transformation T_2^{-1}. As for the multiplication factor in the operation of Q_1, V is precalculated for each coefficient and stored in a table.

Appendix B

The turbo codec

This appendix provides details about the turbo coding and decoding process applied in the codec. Originally, turbo codes have been designed for protecting data sent across error prone communication channels (such as wireless links). When used in DVC, turbo codes are used for compression purposes. Due to these differences, small modifications to the existing algorithms are required.

B.1 Turbo Encoding

The turbo encoder consists of two convolutional encoders and a pseudo-random interleaver. Before discussing the functional diagram of the turbo encoder, the operation of the convolutional encoders is described.

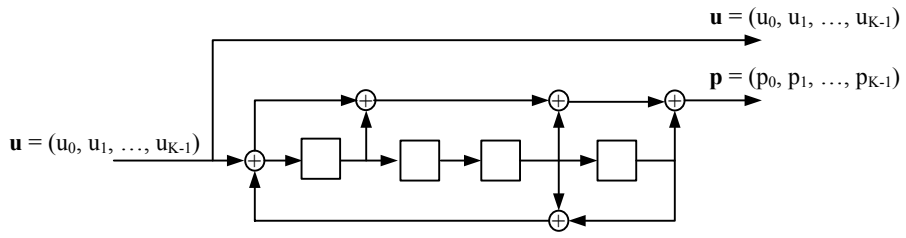

Figure B.1: Systematic feedback convolutional encoder with memory size 4. The □'s represent memory blocks which are able to store one bit. The ⊕'s represent the logical XOR operation.

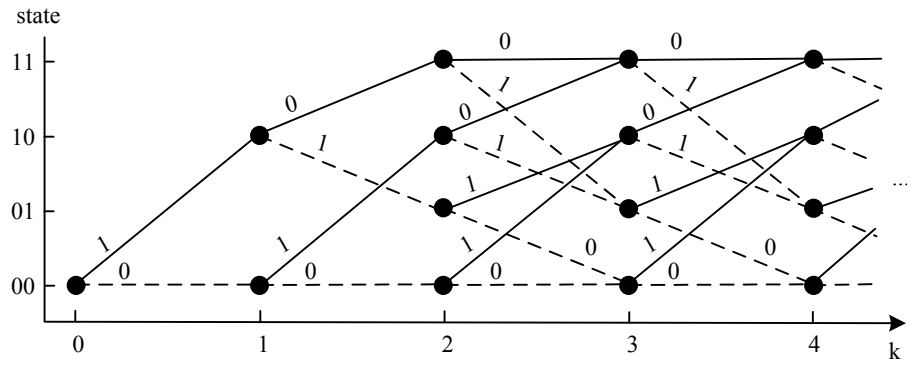

Figure B.2: Trellis diagram of a simple convolutional code with two memory blocks. On the vertical axis, the encoder states are given. These correspond to the content of the memory blocks. On the horizontal axis, the time steps k are given. A dotted line corresponds to an input bit being zero, a full line to an input bit being one. The systematic output bits are identical to the input bits, the corresponding parity output bits are written as labels on the edges.

B.1.1 Systematic convolutional encoders

A convolutional encoder mainly consists of a number of memory blocks, connected in a certain way. Different configurations exist, depending on the number of memory blocks and their interconnection. Figure B.1 depicts the convolutional encoder used in the context of this dissertation. It consists of 4 memory blocks. Each of the K input bits from $\mathbf{u}$ is fed into the encoder one by one, and each time a bit is fed as input, the content of the memory blocks is refreshed (according to the input to this memory block).

The output of the convolutional encoder consists of two K-bit sequences. One is the so-called *systematic* output $\mathbf{u}$, which equals the input. The other output sequence is the parity sequence $\mathbf{p}$. Each parity bit in $\mathbf{p}$ is generated as a logical exclusive or (XOR) of bits contained in the convolutional encoder's system. Remark that the output of the right-most memory block is fed back, from right to left. As a result, the value of each parity bit in $\mathbf{p}$ depends on all previous input bits. This property attributes highly to the performance of the turbo coder.

A convolutional code can be represented by a trellis diagram, describing the content of the convolutional encoder and the generation of the parity bits. This trellis is mainly used for analysis and decoding. Figure B.2 shows a trellis diagram for a simple systematic convolutional code having two memory blocks.

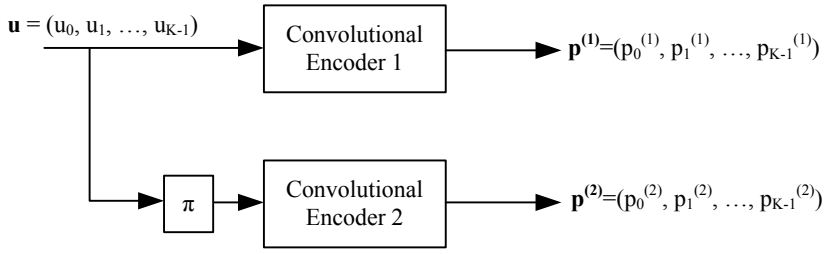

Figure B.3: The turbo encoder consists of two identical convolutional coders and an interleaver.

B.1.2 General encoder structure

The turbo encoder (Figure B.3) consists of two systematic convolutional encoders, and an interleaver denoted π. In this dissertation, the two convolutional encoders are identical and their functional diagrams are given by Figure B.1.

When used for channel coding, both the systematic bits and the parity bits are sent across the error-prone channel. If the systematic part is not corrupted by transmission errors, it can be used as is. If the systematic part is corrupted, then the parity bits can be used for correcting the errors.

In the context of DVC, the equivalent of the systematic part is the side information. This information is generated at the decoder, so it does not need to be sent. Therefore, the systematic outputs of the convolutional encoders in Figure B.1 are discarded. If the decoder is able to generate an accurate prediction without errors, then no parity bits are needed. On the other hand, less accurate side information can be corrected at the decoder using the parity bits. Hence, the number of parity bits needed depends on the quality of the side information. Better compression is achieved if the quality of the side information is higher and vice versa.

The task of the interleaver is to generate a scrambled version of $\mathbf{u}$ that can be used as input for the second convolutional encoder. This scramble operation is in fact a pseudo-random permutation, as described in [116] (page 778, Eq. 16.12 with k = 7). The same interleaver will be used at the decoder as well. The interleaver introduces some kind of randomness in the code, which contributes to the high performance of turbo coding.

At the output of the turbo decoder, two K-bit parity sequences are delivered denoted $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$. Thus, no compression has been achieved so far. In fact, the output of the turbo encoder is twice the size as the input. Compression is actually achieved by sending only a subset of the parity bits to the decoder, through a process called *puncturing*, described next.

B.2 The puncturing process

The word *puncturing* refers to the removal of bits from the parity sequences $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$. A probably more intuitive way of thinking is how to select a subset of the parity bits from $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$.

Note that the total number of parity bits that need to be selected is provided by the decoder, through the use of the feedback channel. Parity bits are requested by the decoder until a certain criterion is satisfied, as discussed later on.

To extract the parity bits, each of the parity sequences $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$ is divided into blocks. Each block has the same size M, except for the last block. The value of M is determined so that the size of the last block is at least $0.7M$, and $M \geq 64$.

The total number of parity bits to extract is divided over the blocks. To select only a subset of size N_p of the parity bits in one block, a pseudo-random selection strategy is used. More specifically, a binary mask is created, having the same size as the block size. This mask contains N_p '1's followed by zeros. After interleaving the mask, the result is used to select the parity bits. If the interleaved binary mask has a value of '1' at index k, the parity bit at index k in the block will be selected. In the case of '0', the parity bit will not be selected.

Finally, all selected parity bits are sent to the turbo decoder.

B.3 Turbo Decoding

At the turbo decoder (Figure B.4), an iterative decoding process is applied involving two soft-input, soft-output (SISO) convolutional decoders. While several decoding algorithms exist, one of the most efficient solutions is the so-called *log-Map* or *log-domain BCJR* algorithm [116]. This algorithm is used in the context of this dissertation. One of the key features of the log-Map algorithm is that it operates on logarithms of probabilities. For a complete description of the decoding process, the interested reader is referred to the literature [116].

In each iteration, one of the SISO decoders calculates the logarithm of the a posteriori probability (LAPP)[1]

$$L(u_k) = \ln\left(\frac{P\left(u_k = 1 \mid \mathbf{r}\right)}{P\left(u_k = -1 \mid \mathbf{r}\right)}\right) \tag{B.1}$$

for each encoded bit u_k.

[1] It is common to represent the binary values 1 and 0 by the symbols 1 and -1. This convention will be used in this section as well.

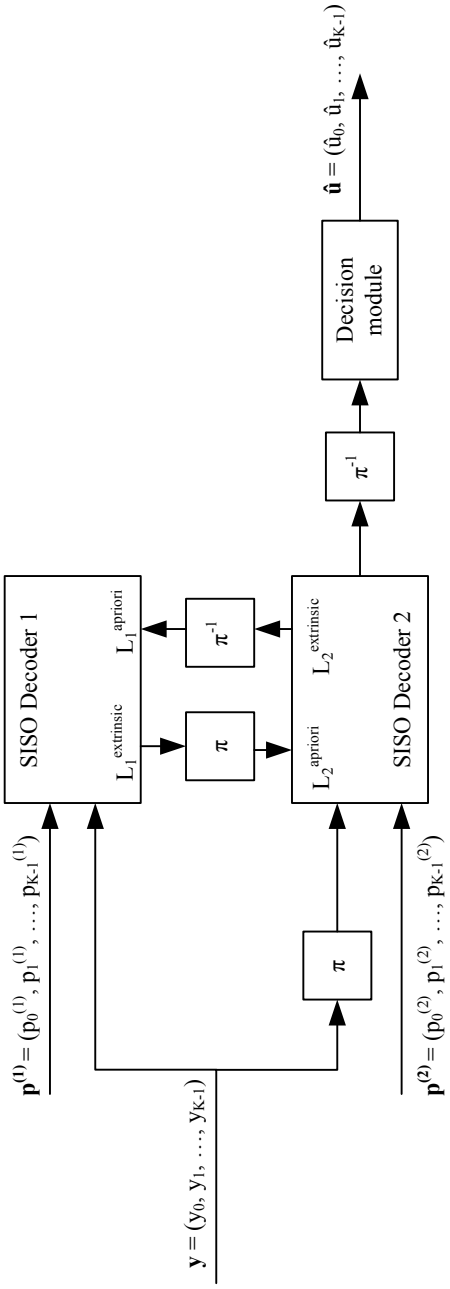

Figure B.4: Turbo decoding is an iterative process involving two soft-input, soft-output (SISO) convolutional decoders passing information to each other.

$\mathbf{r}$ represents all information known to the SISO decoder, i.e., the side information, the reliability of the side information (i.e., the correlation model), the parity bits, and the a priori information $L_j^{apriori}$ ($j = 1$ or 2). The SISO decoders exchange information, called *a priori information* when received as input, or *extrinsic information* when generated as output (and serving as input for the other SISO decoder). The extrinsic information generated at one of the SISO decoders reflects new insights that have been obtained from the parity input sequence. These need to be communicated to the other SISO decoder, since the parity sequence is not available at the other SISO.

Remark that – when passing information from one SISO decoder to the other – interleaving π or deinterleaving π^{-1} must be performed, in order to have meaningful information at the target SISO decoder.

The iterative turbo decoding process is terminated when a certain stopping criterion is met, as explained in Section B.3.1. If turbo decoding is terminated, the LAPP ratios calculated in the last iteration serve as decoded bits, by converting them into bit values using the following interpretation:

$$\hat{u}_k = 1 \text{ if } L(u_k) >= 0 \text{ and } \hat{u}_k = 0 \text{ if } L(u_k) < 0. \tag{B.2}$$

B.3.1 Stopping criterion for turbo decoding

Offline stopping criteria are used frequently, especially in the early DVC implementations. An offline strategy is also adopted in the system described in Chapter 3, and Section 4.3. In these systems, the decoder is granted access to the original for deciding whether to stop turbo decoding for the current bitplane, or not. If the current output matches the original bitplane, turbo decoding is stopped and the current output is returned as a result. If this is not the case for 20 turbo decoding iterations, the turbo decoding process is stopped anyway, and more parity bits are requested from the encoder.

An analysis of online stopping criteria has been provided in collaborative work [117]. This work concludes that the so-called sign-difference ratio [118] is the best criterion to use in a DVC context. More specifically, let D denote the number of sign differences between the a priori and the extrinsic values at the SISO decoder:

$$D = |\{k; 1 \leq k \leq N, L_2^{apriori}(u_k) \cdot L_2^{extrinsic}(u_k) < 0\}|. \tag{B.3}$$

Then, after each iteration, decoding is considered successful if $D < q \cdot N$. The threshold q was experimentally obtained as being in the interval $[0.001, 0.01]$. As before, turbo decoding is stopped anyway if the criterion is not valid for 20 decoding iterations. When turbo decoding stops, more bits are requested from

the encoder if the sign-difference is still not met. This strategy is adopted in Section 4.4.

A different online method is applied in Chapter 5, focusing on a subset of the bits instead of all. This is driven by a rate-distortion model, which enables classifying bits as relevant or non-relevant. Reliability of decoding is only required for the relevant bits. More details are provided in the chapter.

Appendix C

Developing an RD model

C.1 Coefficient rate calculation

Using the notations introduced in the beginning of Section 5.2 (concerning the uniform quantizer and Laplacian correlation model), the probability of q_i containing X given $Y = y$ is written as:

$$p_i = \int_{q_i^L}^{q_i^H} f_{X|Y}(x|y)dx. \tag{C.1}$$

It is quite straightforward to calculate p_i, by discriminating between different cases for i:

$$p_i = \begin{cases} \frac{e^{-\alpha(y-q_i^H)} - e^{-\alpha(y-q_i^L)}}{2}, & i < K \\ 1 - \frac{e^{-\alpha(y-q_i^L)} + e^{-\alpha(q_i^H-y)}}{2}, & i = K \\ \frac{e^{-\alpha(q_i^L-y)} - e^{-\alpha(q_i^H-y)}}{2}, & i > K. \end{cases} \tag{C.2}$$

This can be rewritten by introducing $y_N = \frac{y}{\Delta} - K$ and using the expressions for q_i^L and q_i^H:

$$p_i = \begin{cases} e^{\alpha\Delta(i-K)} \cdot \gamma_1, & i < K \\ 1 - e^{-\alpha\Delta/2} \cdot \cosh(\alpha\Delta y_N), & i = K \\ e^{-\alpha\Delta(i-K)} \cdot \gamma_2, & i > K \end{cases}$$

with

$$\begin{aligned} \gamma_1 &= \sinh(\alpha\Delta/2) \cdot e^{-\alpha\Delta y_N}, \\ \gamma_2 &= \sinh(\alpha\Delta/2) \cdot e^{\alpha\Delta y_N}. \end{aligned}$$

This result is used to derive a closed-form expression for the conditional entropy, given the position of the side information. With $j = i - K$, we get:

$$\begin{aligned} H(Q(X)|Y_N = y_N) &= -\sum_{i=-\infty}^{\infty} p_i \log_2 p_i, \\ &= -\sum_{i=-\infty}^{K-1} p_i \log_2 p_i - (p_K \log_2 p_K) - \sum_{i=K+1}^{\infty} p_i \log_2 p_i \\ &= -\sum_{j=-\infty}^{-1} p_{j+K} \log_2 p_{j+K} - (p_K \log_2 p_K) \\ &\quad -\sum_{j=1}^{\infty} p_{j+K} \log_2 p_{j+K} \end{aligned} \tag{C.3}$$

The first term of this sum is calculated as:

$$\begin{aligned} &\sum_{j=-1}^{-\infty} p_{j+K} \log_2 p_{j+K} \\ &\quad = \sum_{j=-1}^{-\infty} \gamma_1 e^{\alpha\Delta j} \log_2 \gamma_1 e^{\alpha\Delta j} \\ &\quad = \sum_{j=1}^{\infty} \gamma_1 e^{-\alpha\Delta j} \log_2 \left(\gamma_1 e^{-\alpha\Delta j}\right) \qquad \text{(C.4)} \\ &\quad = \gamma_1 \sum_{j=1}^{\infty} e^{-\alpha\Delta j} \left(\frac{\ln \gamma_1}{\ln 2} + \frac{\ln e^{-\alpha\Delta j}}{\ln 2}\right) \\ &\quad = \frac{\gamma_1}{\ln 2} \left(\ln \gamma_1 \sum_{j=1}^{\infty} e^{-\alpha\Delta j} - \alpha\Delta \sum_{j=1}^{\infty} j e^{-\alpha\Delta j}\right) \\ &\quad = \frac{\gamma_1}{\ln 2} \left(\ln \gamma_1 \cdot \frac{e^{-\alpha\Delta}}{1 - e^{-\alpha\Delta}} - \alpha\Delta \cdot \frac{e^{-\alpha\Delta}}{(1 - e^{-\alpha\Delta})^2}\right) \qquad \text{(C.5)} \end{aligned}$$

The third term is calculated in a similar way, recognizing that:

$$\sum_{j=1}^{\infty} p_{j+K} \log_2 p_{j+K} = \sum_{j=1}^{\infty} \gamma_2 e^{-\alpha\Delta j} \log_2 \left(\gamma_2 e^{-\alpha\Delta j}\right), \tag{C.6}$$

has a similar form as Eq. C.4, so that the following expression is obtained:

$$\begin{aligned} H(Q(X)|Y_N = y_N) = &-\frac{e^{-\alpha\Delta}}{\ln 2 \cdot (1 - e^{-\alpha\Delta})} \cdot (\gamma_1 \ln \gamma_1 + \gamma_2 \ln \gamma_2) \\ &+\frac{\alpha\Delta e^{-\alpha\Delta}}{\ln 2 \cdot (1 - e^{-\alpha\Delta})^2} \cdot (\gamma_1 + \gamma_2) \\ &-(1 - e^{-\alpha\Delta/2} \cosh(\alpha\Delta y_N)) \\ &\cdot \log_2(1 - e^{-\alpha\Delta/2} \cosh(\alpha\Delta y_N)). \end{aligned} \tag{C.7}$$

By using the expressions for γ_1 and γ_2, the first term of this sum results in:

$$\begin{aligned} -\frac{e^{-\alpha\Delta/2}}{\ln 2} \cdot \alpha\Delta y_N \cdot \sinh(\alpha\Delta y_N) \\ - e^{-\alpha\Delta/2} \cdot \log_2 \sinh(\alpha\Delta/2) \cdot \cosh(\alpha\Delta y_N) \end{aligned} \tag{C.8}$$

while the second term simplifies to:

$$\frac{\alpha\Delta}{\ln 2} \cdot \frac{e^{-\alpha\Delta/2}}{1 - e^{-\alpha\Delta}} \cdot \cosh(\alpha\Delta y_N), \tag{C.9}$$

which results in the following closed-form expression:

$$\begin{aligned} H(Q(X)|Y_N = y_N) = &A \cdot \cosh(\alpha\Delta y_N) + B \cdot y_N \cdot \sinh(\alpha\Delta y_N) \\ &-(1 - e^{-\alpha\Delta/2} \cosh(\alpha\Delta y_N)) \\ &\cdot \log_2(1 - e^{-\alpha\Delta/2} \cosh(\alpha\Delta y_N)) \end{aligned}$$

with:

$$\begin{aligned} A =& e^{-\alpha\Delta/2} \left(\frac{\alpha\Delta}{\ln 2 \cdot (1 - e^{-\alpha\Delta})} - \log_2 \sinh(\alpha\Delta/2) \right), \\ B =& -\frac{\alpha\Delta}{\ln 2} e^{-\alpha\Delta/2}. \end{aligned} \tag{C.10}$$

C.2 Coefficient distortion calculation

In this subsection, we develop an expression for the distortion using the mean absolute difference (MAD) as distortion metric. We assume that the output of the channel decoder is perfect, so that the decoder knows the bin containing the original value. Reconstruction is performed as in the Stanford codec [42]:

$$x'_i = \begin{cases} q_i^H & i < K, \\ y & i = K, \\ q_i^L & i > K. \end{cases} \tag{C.11}$$

As such, average distortion given the specific realization of the side information, is calculated as follows:

$$\begin{aligned}
E[|X - X'||Y = y] = &\sum_{i=-\infty}^{\infty} \int_{q_i^L}^{q_i^H} f_{X|Y}(x|y)|x - x_i'|dx \\
= &\sum_{i=-\infty}^{K-1} \int_{q_i^L}^{q_i^H} \frac{\alpha}{2} e^{-\alpha(y-x)} \cdot (q_i^H - x)dx \\
&+ \int_{q_K^L}^{y} \frac{\alpha}{2} e^{-\alpha(y-x)} \cdot (y - x)dx \\
&+ \int_{y}^{q_K^H} \frac{\alpha}{2} e^{-\alpha(x-y)} \cdot (x - y)dx \\
&+ \sum_{i=K+1}^{\infty} \int_{q_i^L}^{q_i^H} \frac{\alpha}{2} e^{-\alpha(x-y)} \cdot (x - q_i^L)dx. \qquad \text{(C.12)}
\end{aligned}$$

Solving the integrals in the first term of this sum results in:

$$\sum_{i=-\infty}^{K-1} \int_{q_i^L}^{q_i^H} \frac{\alpha}{2} e^{-\alpha(y-x)} \cdot (q_i^H - x)dx = \frac{\alpha}{2} e^{-\alpha y} \cdot C \sum_{i=-\infty}^{K-1} e^{\alpha\Delta i}, \qquad \text{(C.13)}$$

with

$$C = \frac{e^{\alpha\Delta/2}}{\alpha^2} - e^{-\alpha\Delta/2}\left(\frac{\Delta}{\alpha} + \frac{1}{\alpha^2}\right). \qquad \text{(C.14)}$$

Using $y_N = \frac{y}{\Delta} - K$, and $j = i - K$, we can further simplify:

$$\begin{aligned}
&\sum_{i=-\infty}^{K-1} \int_{q_i^L}^{q_i^H} \frac{\alpha}{2} e^{-\alpha(y-x)} \cdot (q_i^H - x)dx \\
&= \frac{\alpha}{2} e^{-\alpha y} \cdot C \sum_{j=-\infty}^{-1} e^{\alpha\Delta(j+K)} \\
&= \frac{\alpha}{2} e^{-\alpha\Delta y_N} \cdot C \sum_{j=1}^{\infty} e^{-\alpha\Delta j} \\
&= e^{-\alpha\Delta y_N} \cdot \frac{e^{-\alpha\Delta/2} - (\alpha\Delta + 1)e^{-3\alpha\Delta/2}}{2\alpha(1 - e^{-\alpha\Delta})} \qquad \text{(C.15)}
\end{aligned}$$

Similarly, the final term in Eq. C.12 evaluates to:

$$e^{\alpha\Delta y_N} \cdot \frac{e^{-\alpha\Delta/2} - (\alpha\Delta + 1)e^{-3\alpha\Delta/2}}{2\alpha(1 - e^{-\alpha\Delta})}. \qquad \text{(C.16)}$$

The second and third term are given by:

$$\frac{-2\alpha\Delta y_N - \alpha\Delta - 2}{4\alpha} \cdot e^{-\alpha\Delta(0.5+y_N)} + \frac{1}{2\alpha}, \tag{C.17}$$

and

$$\frac{2\alpha\Delta y_N - \alpha\Delta - 2}{4\alpha} \cdot e^{-\alpha\Delta(0.5-y_N)} + \frac{1}{2\alpha}, \tag{C.18}$$

respectively. Adding these results together, we get the following expression:

$$\begin{aligned} E[|X - X'||Y_N = y_N] = &- \cosh(\alpha\Delta y_N) \cdot \Delta e^{-\alpha\Delta/2} \cdot \left(\frac{e^{-\alpha\Delta}}{1 - e^{-\alpha\Delta}} + 0.5\right) \\ &+ \Delta y_N \sinh(\alpha\Delta y_N) \cdot e^{-\alpha\Delta/2} + \frac{1}{\alpha}. \end{aligned} \tag{C.19}$$

Publications

Journal Papers

1. Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Rate-distortion driven decoder-side bitplane mode decision for distributed video coding. *Signal Processing: Image Communication*, 25(9):660 – 673, October 2010.

2. Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Flexible distribution of complexity by hybrid predictive-distributed video coding. *Signal Processing: Image Communication*, 25(2):94–110, February 2010.

3. Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Nikos Deligiannis, Jan De Cock, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Exploiting quantization and spatial correlation in virtual-noise modeling for distributed video coding. *Signal Processing: Image Communication*, 25(9):674 – 686, 2010.

4. Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Decoder-driven mode decision in a block based distributed video codec. Accepted for publication in *Multimedia Tools and Applications*, 2010.

5. Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Introducing skip mode in distributed video coding. *Signal Processing: Image Communication*, 24(3):200–213, March 2009.

Book chapters

1. Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Correlation noise es-

timation in distributed video coding. Accepted for publication in *Video coding*, IN-TECH, 2011.

2. Peter Lambert, Stefaan Mys, Jozef Škorupa, Jürgen Slowack, Rik Van de Walle, and Christos Grecos. Distributed video coding for video communication on mobile devices and sensors. In *Handheld computing for mobile commerce: applications, concepts and technologies*, pages 375–402. Information Science Publishing, 2010.

3. Peter Lambert, Stefaan Mys, Jozef Škorupa, Jürgen Slowack, Rik Van de Walle, and Christos Grecos. Fast mode decision in H.264/AVC. In *Handheld computing for mobile commerce: applications, concepts and technologies*, pages 403–424. Information Science Publishing, 2010.

Papers in conference proceedings

1. Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Bitplane intra coding with decoder-side mode decision in distributed video coding. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages 3733 – 3736, September 2010.

2. Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Compensating for motion estimation inaccuracies in distributed video coding. In *Proc. International Conference on Image and Signal Processing (ICISP)*, pages 324–332, June 2010.

3. Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Distributed video coding with decoder-driven skip. In *Proc. Mobimedia*, September 2009.

4. Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Peter Lambert, Christos Grecos, and Rik Van de Walle. Accounting for quantization noise in online correlation noise estimation for distributed video coding. In *Proc. Picture Coding Symposium (PCS)*, May 2009.

5. Jozef Škorupa, Jan De Cock, Jürgen Slowack, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Correlation modeling with decoder-side quantization distortion estimation for distributed video coding. Accepted for publication in: *Proc. Picture Coding Symposium (PCS)*, December 2010.

6. Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Stopping criterions for turbo coding in a Wyner-Ziv video codec. In *Proc. Picture Coding Symposium (PCS)*, May 2009.

7. Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Peter Lambert, and Rik Van de Walle. Accurate correlation modeling for transform-domain Wyner-Ziv video coding. In *Proc. Pacific-Rim Conference on Multimedia (PCM)*, pages 1–10, December 2008.

8. Jozef Škorupa, Stefaan Mys, Jürgen Slowack, Peter Lambert, and Rik Van de Walle. Heuristic dynamic complexity coding. In *Proc. SPIE*, volume 7000, pages 70000G–70000G–8, April 2008.

9. Stefaan Mys, Jürgen Slowack, Jozef Skorupa, Peter Lambert, and Rik Van de Walle. Motion compensated interpolation as a new inter coding mode for 8x8 macroblock partitions in H.264/AVC B slices. In *Lecture Notes in Computer Science*, volume 5353, pages 168–177. Springer Verlag, 2008.

10. Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Dynamic complexity coding: Combining predictive and Distributed Video Coding. In *Proc. Picture Coding Symposium (PCS)*, November 2007.

11. Saar De Zutter, Jürgen Slowack, Wesley De Neve, and Rik Van de Walle. Efficient Context Management in Context-Aware Environments. In *Proceedings of the Third International Mobile Multimedia Communications Conference, The Context Aware Mobile Multimedia Systems and Applications Workshop*. ICSP, 2007.

References

[1] M. Maitre, Y. Shinagawa, and M.N. Do. Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering. *IEEE Transactions on Image Processing*, 17(6):946 –957, June 2008.

[2] Wenfeng Li, Jin Zhou, Baoxin Li, and M.I. Sezan. Virtual view specification and synthesis for free viewpoint television. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(4):533 –546, April 2009.

[3] P. Benzie, J. Watson, P. Surman, I. Rakkolainen, K. Hopf, H. Urey, V. Sainov, and C. von Kopylow. A survey of 3DTV displays: Techniques and technologies. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(11):1647 –1658, November 2007.

[4] A.M. Tekalp, E. Kurutepe, and M.R. Civanlar. 3dtv over ip. *IEEE Signal Processing Magazine*, 24(6):77 –87, November 2007.

[5] ITU-T. *Video codec for audiovisual services at* $p \times 64$ *kbits/s*, 1993. ITU-T Recommendation H.261, Version 2.

[6] ITU-T. *Video coding for low bit rate communications*, 1998. ITU-T Recommendation H.263, Version 2.

[7] ITU-T and ISO/IEC JTC1. *Generic coding of moving pictures and associated audio information – Part 2: Video*, 1994. ITU-T recommendation H.262 and ISO/IEC 13818-2 (MPEG-2).

[8] ISO/IEC. *Coding of audiovisual objects – Part 2: Visual*, 1999. ISO/IEC 14496-2 (MPEG-4).

[9] 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.

[10] Fernando Pereira, Luis Torres, Christine Guillemot, Touradj Ebrahimi, Riccardo Leonardi, and Sven Klomp. Distibuted video coding: Selecting the most promising application scenarios. *Signal Processing: Image Communication*, pages 339–352, 2008.

[11] Mohammad Rahimi, Rick Baer, Obimdinachi I. Iroezi, Juan C. Garcia, Jay Warrior, Deborah Estrin, and Mani Srivastava. Cyclops: In situ image sensing and interpretation in wireless sensor networks. In *In SenSys*, pages 192–204. ACM Press, 2005.

[12] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh. Deploying a wireless sensor network on an active volcano. *Internet Computing, IEEE*, 10(2):18 – 25, March-April 2006.

[13] Benny P L Lo and Guang-Zhong Yang. Key technical challenges and current implementations of body sensor networks. In *Proceedings of the Second International Workshop Wearable and Implantable Body Sensor Networks*, April 2005.

[14] G. Simon, Á. Lédeczi M. Maróti, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. In *Proc. Conf. Embedded Networked Sensor Systems*, page 112, November 2004.

[15] Jing Wu and Ye Li. Low-complexity video compression for capsule endoscope based on compressed sensing theory. *Conference of the IEEE Engineering in Medicine and Biology Society*, 1:3727–30, 2009.

[16] Feng Gong, Paul Swain, and Timothy Mills. Wireless endoscopy. *Gastrointestinal Endoscopy*, 51(6):725 – 729, 2000.

[17] Xun Guo, Yan Lu, Feng Wu, Debin Zhao, and Wen Gao. Wyner-Ziv-based multiview video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(6):713 – 724, June 2008.

[18] J.L. Martinez, G. Fernandez-Escribano, H. Kalva, W.A.C. Fernando, and P. Cuenca. Wyner-ziv to h.264 video transcoder for low cost video encoding. *IEEE Transactions on Consumer Electronics*, 55(3):1453 –1461, August 2009.

[19] E. Peixoto, R.L. de Queiroz, and D. Mukherjee. A wyner-ziv video transcoder. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(2):189 –200, February 2010.

[20] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, July 1948.

[21] F. Hekland. A review of joint source-channel coding. Technical report, Norwegian University of Science and Technology, February 2004.

[22] M. Stoufs, A. Munteanu, J. Cornelis, and P. Schelkens. Scalable joint source-channel coding for the scalable extension of H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(12):1657 –1670, December 2008.

[23] A. Gabay, M. Kieffer, and P. Duhamel. Joint source-channel coding using real bch codes for robust image transmission. *IEEE Transactions on Image Processing*, 16(6):1568 –1583, jun. 2007.

[24] S. Rane, P. Baccichet, and B. Girod. Systematic lossy error protection of video signals. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(10):1347 – 1360, October 2008.

[25] Mourad Ouaret, Frdric Dufaux, and Touradj Ebrahimi. Error-resilient scalable compression based on distributed video coding. *Signal Processing: Image Communication*, 24(6):437 – 451, 2009.

[26] David Slepian and Jack K. Wolf. Noiseless coding of correlated information sources. *IEEE Trans. Inf. Theory*, 19(4):471–480, July 1973.

[27] Aaron D. Wyner. On source coding with side information at the decoder. *IEEE Trans. Inf. Theory*, 21(3):294–300, 1975.

[28] Aaron D. Wyner and Jacob Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Trans. Inf. Theory*, 22(1):1–10, 1976.

[29] Aaron D. Wyner. The rate-distortion function for source coding with side information at the decoder-II: General sources. *Information and Control*, 38:60–80, 1978.

[30] S. Sandeep Pradhan, Jim Chou, and Kannan Ramchandran. Duality between source coding and channel coding and its extension to the side information case. *IEEE Transactions on Information Theory*, 49(5):1181–1203, May 2003.

[31] Ram Zamir. The rate loss in the Wyner-Ziv problem. *IEEE Trans. Inf. Theory*, 42(6):2073–2084, 1996.

[32] S. Sandeep Pradhan and Kannan Ramchandran. Distributed source coding using syndromes (DISCUS): Design and construction. In *Proc. IEEE Data Compression Conference (DCC)*, pages 158–167, March 1999.

[33] S. Sandeep Pradhan, Julius Kusuma, and Kannan Ramchandran. Distributed compression in a dense microsensor network. *IEEE Signal Processing Magazine*, 19:51–60, March 2002.

[34] Rohit Puri and Kannan Ramchandran. PRISM: A new robust video coding architecture based on distributed compression principles. In *Proc. Allerton Conference on Communication, Control and Computing*, October 2002.

[35] Rohit Puri and Kannan Ramchandran. PRISM: A "reversed" multimedia coding paradigm. In *Proc. IEEE International Conference on Image Processing (ICIP)*, September 2003.

[36] Rohit Puri and Kannan Ramchandran. PRISM: an uplink-friendly multimedia coding paradigm. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, volume 4, pages 856–859, April 2004.

[37] Anne Aaron and Bernd Girod. Compression with side information using turbo codes. In *Proc. IEEE Data Compression Conference (DCC)*, pages 252–261, April 2002.

[38] Anne Aaron, Rui Zhang, and Bernd Girod. Wyner-Ziv coding of motion video. In *Proc. Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 240–244, November 2002.

[39] Anne Aaron, Eric Setton, and Bernd Girod. Towards practical Wyner-Ziv coding of video. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages 869–872, September 2003.

[40] Anne Aaron, Shantanu Rane, Eric Setton, and Bernd Girod. Transform-domain Wyner-Ziv codec for video. In *Proc. SPIE Visual Communications and Image Processing*, volume 5308, pages 520–528, January 2004.

[41] Anne Aaron, Rui Zhang, and Bernd Girod. Wyner-Ziv video coding with hash-based motion compensation at the receiver. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages 3097–3100, October 2004.

[42] Bernd Girod, Anne Aaron, Shantanu Rane, and David Rebollo-Monedero. Distributed Video Coding. In *Proc. IEEE, Special Issue on Video Coding and Delivery*, volume 93, pages 71–83, January 2005.

[43] Claude Berrou, Alain Glavieux, and Punja Thitimajshima. Near Shannon limit error-correcting coding and decoding: turbo codes. In *Proc. IEEE International Conference on Communications (ICC)*, volume 2, pages 1064–1070, May 1993.

[44] M. Morbée, J. Prades-Nebot, A. Pizurica, and W. Philips. Feedback channel suppression in pixel-domain Distributed Video Coding. In *Annual Workshop on Circuits, Systems and Signal Processing (ProRISC)*, pages 154–157, November 2006.

[45] M. Morbée, J. Prades-Nebot, A. Pizurica, and W. Philips. Rate allocation algorithm for pixel-domain Distributed Video Coding without feedback channel. In *Proc. ICASSP*, volume 1, pages I–521–I–524, April 2007.

[46] C. Yaacoub, J. Farah, and B. Pesquet-Popescu. Content adaptive gop size control with feedback channel suppression in distributed video coding. In *IEEE International Conference on Image Processing (ICIP)*, November 2009.

[47] T. Sheng, X. Zhu, G. Hua, H. Guo, J. Zhou, and C. W. Chen. Feedback-free rate-allocation scheme for transform domain wyner-ziv video coding. *Multimedia Systems*, January 2010. DIO 10.1007/s00530-009-0179-8.

[48] J. Ascenso and F. Pereira. Low complexity intra mode selection for efficient distributed video coding. In *Proc. International Conference on Multimedia and Expo (ICME)*, pages 101–104, July 2009.

[49] C. Tonoli, P. Migliorati, and R. Leonardi. Error resilience in current distributed video coding architectures. *EURASIP Journal on Image and Video Processing*, 2009, 2009. article ID 946585.

[50] *DISCOVER-Distributed Coding for Video Services*, September 2005. http://www.discoverdvc.org/ (Accessed Dec. 1, 2009).

[51] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret. The DISCOVER codec: Architecture, techniques and evaluation. In *Proc. Picture Coding Symposium (PCS)*, November 2007.

[52] C. Brites, J. Ascenso, J. Q. Pedro, and F. Pereira. Evaluating a feedback channel based transform domain wyner-ziv video codec. *Signal Processing: Image Communication*, pages 269–297, 2008.

[53] J. Ascenso, C. Brites, and F. Pereira. Content adaptive Wyner-Ziv video coding driven by motion activity. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages 605–608, October 2006.

[54] João Ascenso, Catarina Brites, and Fernando Pereira. Improving frame interpolation with spatial motion smoothing for pixel domain Distributed Video Coding. *5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services*, July 2005.

[55] Sven Klomp, Yuri Vatis, and Jörn Ostermann. Side information interpolation with sub-pel motion compensation for Wyner-Ziv decoder. *International Conference on Signal Processing and Multimedia Applications (SIGMAP)*, August 2006.

[56] Robert G. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, pages 21–28, january 1962.

[57] David Varodayan, Anne Aaron, and Bernd Girod. Rate-adaptive codes for distributed source coding. *EURASIP Signal Processing Journal, Special Issue on Distributed Source Coding*, 86(11):3123–3130, November 2006.

[58] F. Pereira, J. Ascenso, and C. Brites. Studying the GOP size impact on the performance of a feedback channel-based Wyner-Ziv video codec. In *IEEE Pacific-Rim Symposium on Image and Video Technology*, December 2007.

[59] D. Kubasov, K. Lajnef, and C. Guillemot. A hybrid encoder/decoder rate control for a Wyner-Ziv video codec with a feedback channel. In *IEEE MultiMedia Signal Processing Workshop*, pages 251–254, October 2007.

[60] D. Kubasov, J. Nayak, and C. Guillemot. Optimal reconstruction in Wyner-Ziv video coding with multiple side information. In *IEEE MultiMedia Signal Processing Workshop*, pages 183–186, October 2007.

[61] E. Akyol and M. van der Schaar. Complexity model based proactive dynamic voltage scaling for video decoding systems. *IEEE Transactions on Multimedia*, 9(7):1475 –1492, November 2007.

[62] D. Wu, F. Pan, K.P. Lim, S. Wu, Z.G. Li, X. Lin, S. Rahardja, and C.C. Ko. Fast intermode decision in h.264/avc video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(7):953 – 958, July 2005.

[63] C.S. Kannangara, I.E.G. Richardson, M. Bystrom, J.R. Solera, Yafan Zhao, A. MacLennan, and R. Cooney. Low-complexity skip prediction for h.264 through lagrangian cost estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(2):202 – 208, February 2006.

[64] Yu-Ming Lee and Yinyi Lin. Zero-block mode decision algorithm for h.264/avc. *IEEE Transactions on Image Processing*, 18(3):524 –533, March 2009.

[65] M.G. Sarwer and Q.M.J. Wu. Efficient two step edge based partial distortion search for fast block motion estimation. *IEEE Transactions on Consumer Electronics*, 55(4):2154 –2162, November 2009.

[66] Shao-Wei Liu, Shou-Der Wei, and Shang-Hong Lai. Fast optimal motion estimation based on gradient-based adaptive multilevel successive elimination. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(2):263 –267, February 2008.

[67] Ki Beom Kim, Young Jeon, and Min-Cheol Hong. Variable step search fast motion estimation for h.264/avc video coder. *IEEE Transactions on Consumer Electronics*, 54(3):1281 –1286, August 2008.

[68] Yih Han Tan, Zhengguo Li, Keng Pang Lim, and Susanto Rahardja. Simplified motion-refined schame for fine-granularity scalability. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(9):1212 – 1222, September 2008.

[69] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu. Power-rate-distortion analysis for wireless video communication under energy constraints. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(5):645–658, May 2005.

[70] D.S. Turaga, M. van der Schaar, and B. Pesquet-Popescu. Complexity scalable motion compensated wavelet video encoding. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(8):982 – 993, August 2005.

[71] Tom Clerckx, Adrian Munteanu, Jan Cornelis, and Peter Schelkens. Distributed video coding with shared encoder/decoder complexity. In *IEEE International Conference on Image Processing (ICIP)*, volume 6, pages 417–420, September 2007.

[72] Hwa-Yong Oh, S. Erturk, and Tae-Gyu Chang. Low complexity video encoding with one-bit transform based network-driven motion estimation. *IEEE Transactions on Consumer Electronics*, 53(2):601 –608, may 2007.

[73] Limin Liu, Zhen Li, and E.J. Delp. Efficient and low-complexity surveillance video compression using backward-channel aware wyner-ziv video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(4):453 – 465, April 2009.

[74] Debargha Mukherjee. A robust reversed-complexity Wyner-Ziv video codec introducing sign-modulated codes. Technical Report HPL-2006-80, HP Laboratories Palo Alto, May 2006.

[75] Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Dynamic complexity coding: Combining predictive and Distributed Video Coding. In *Proc. Picture Coding Symposium (PCS)*, November 2007.

[76] Jozef Škorupa, Stefaan Mys, Jürgen Slowack, Peter Lambert, and Rik Van de Walle. Heuristic dynamic complexity coding. In *Proc. SPIE*, volume 7000, pages 70000G–70000G–8, April 2008.

[77] Jürgen Slowack, Jozef Škorupa, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Flexible distribution of complexity by hybrid predictive-distributed video coding. *Signal Processing: Image Communication*, 25(2):94–110, February 2010.

[78] Anne Aaron, David Varodayan, and Bernd Girod. Wyner-Ziv residual coding of video. In *Proc. Picture Coding Symposium (PCS)*, April 2006.

[79] C. Brites and F. Pereira. Correlation noise modeling for efficient pixel and transform domain Wyner-Ziv video coding. *IEEE Trans. Circuits Syst. Video Technol.*, 18:1177–1190, September 2008.

[80] E. Brockmeyer, L. Nachtergaele, F. V. M. Catthoor, J. Bormans, and H. J. De Man. Low power memory storage and transfer organization for the mpeg-4 full pel motion estimation on a multimedia processor. *IEEE Transactions on Multimedia*, 1(2), June 1999.

[81] E.A.A. Qaralleh and Tian-Sheuan Chang. Fast variable block size motion estimation by adaptive early termination. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(8):1021–1026, Augustus 2006.

[82] M.G. Sarwer and Q.M.J. Wu. Adaptive variable block-size early motion estimation termination algorithm for H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(8):1196 –1201, Augustus 2009.

[83] Reoxiang Li, Bing Zeng, and M. L. Liou. A new three-step search algorithm for block motion estimation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 4(4):438–442, 1994.

[84] Sören Sofke, Fernando Pereira, and Erika Müller. Dynamic quality control for transform domain wyner-ziv video coding. *J. Image Video Process.*, 2009:1–15, 2009.

[85] Joint Video Team. *Joint Model reference software*. `http://iphome.hhi.de/suehring/tml/index.htm`.

[86] H. Schwarz, D. Marpe, and T. Wiegand. Analysis of hierarchical b pictures and mctf. pages 1929 – 1932, July 2006.

[87] Alexander Schrijver. *Theory of linear and integer programming*. Wiley, 1998.

[88] Z. M. Belkoura. *Analysis and Application of Turbo Coder based Distributed Video Coding*. PhD thesis, Technical University of Berlin, 2007.

[89] Hu Chen and Eckehard Steinbach. Flexible distribution of computational complexity between the encoder and the decoder in distributed video coding. In *Proc. International Conference on Multimedia and Expo (ICME)*, pages 801 – 804, June 2008.

[90] J. Jain and A. Jain. Displacement measurement and its application in interframe image coding. *Communications, IEEE Transactions on*, 29(12):1799 – 1808, December 1981.

[91] Byeong-Doo Choi, Jong-Woo Han, Chang-Su Kim, and Sung-Jea Ko. Motion-compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(4):407 –416, april 2007.

[92] Domenic Forte and Ankur Srivastava. Energy and thermal-aware video coding via encoder/decoder workload balancing. In *ISLPED '10: Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design*, pages 207–212. ACM, 2010.

[93] Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Motion compensated interpolation as a new inter coding mode for 8x8 macroblock partitions in H.264/AVC B slices. In *Lecture Notes in Computer Science*, volume 5353, pages 168–177. Springer Verlag, 2008.

[94] A. Trapanese, M. Tagliasacchi, S. Tubaro, J. Ascenso, C. Brites, and F. Pereira. Improved correlation noise statistics modeling in frame-based pixel domain Wyner-Ziv video coding. *International Workshop on Very Low Bitrate Video*, September 2005.

[95] P.F.A. Meyer, R.P. Westerlaken, R. Klein Gunnewiek, and R.L. Lagendijk. Distributed source coding of video with non-stationary side-information. In *Visual Communications and Image Processing (VCIP)*, 2005.

[96] Ronald P. Westerlaken, Rene Klein Gunnewiek, and R. L. Lagendijk. The role of the virtual channel in distributed source coding of video. In *IEEE International Conference on Image Processing (ICIP)*, 2005.

[97] Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Peter Lambert, and Rik Van de Walle. Accurate correlation modeling for transform-domain Wyner-Ziv video coding. In *Proc. Pacific-Rim Conference on Multimedia (PCM)*, pages 1–10, December 2008.

[98] Xiaopeng Fan, Oscar C. Au, and Ngai Man Cheung. Adaptive correlation estimation for general Wyner-Ziv video coding. In *IEEE International Conference on Image Processing (ICIP)*, November 2009.

[99] Xin Huang and Søren Forchhammer. Improved virtual channel noise model for transform domain Wyner-Ziv video coding. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 921–924, April 2009.

[100] B. Macchiavello, D. Mukherjee, and R. L. Querioz. Iterative side-information generation in a mixed resolution wyner-ziv framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(10):1409–1423, October 2009.

[101] Allen Gersho and Robert M. Gray. *Vector quantization and signal compression*. Kluwer Academic Publishers, 1992.

[102] Gisle Bjøntegaard. Calculation of average PSNR differences between RD-curves. Technical report, VCEG, April 2002. Contribution VCEG-M33.

[103] Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Nikos Deligiannis, Jan De Cock, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Exploiting quantization and spatial correlation in virtual-noise modeling for distributed video coding. *Signal Processing: Image Communication*, 25(9):674 – 686, 2010.

[104] Jozef Škorupa, Jan De Cock, Jürgen Slowack, Stefaan Mys, Nikos Deligiannis, Peter Lambert, Adrian Munteanu, and Rik Van de Walle. Correlation modeling with decoder-side quantization distortion estimation for distributed video coding. In *Accepted for publication in: Proc. Picture Coding Symposium (PCS)*, December 2010.

[105] Denis Kubasov and Christine Guillemot. Mesh-based motion-compensated interpolation for side information extraction in Distributed Video Coding. In *Proc. IEEE International Conference on Image Processing (ICIP)*, October 2006.

[106] R. Martins, C. Brites, J. Ascenso, and F. Pereira. Refining side information for improved transform domain wyner-ziv video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(9):1327–1341, september 2009.

[107] Xiaopeng Fan, Oscar C. Au, Ngai Man Cheung, Yan Chen, and Jiantao Zhou. Successive refinement based Wyner-Ziv video compression. *Signal Processing: Image Communcation*, 2009. doi:10.1016/j.image.2009.09.004.

[108] S. Ye, M. Ouaret, F. Dufaux, and T. Ebrahimi. Improved side information generation for distributed video coding by exploiting spatial and temporal correlations. *EURASIP Journal on Image and Video Processing*, 2009, 2009. Article ID 683510.

[109] Stefaan Mys, Jürgen Slowack, Jozef Škorupa, Peter Lambert, and Rik Van de Walle. Introducing skip mode in distributed video coding. *Signal Processing: Image Communication*, 24(3):200–213, March 2009.

[110] L. Liu, D.-K. He, A. Jagmohan, L. Lu, and E. J. Delp. A low-complexity iterative mode selection algorithm for Wyner-Ziv video compression. In *IEEE International Conference on Image Processing (ICIP)*, pages 1136–1139, October 2008.

[111] Wei-Jung Chien and Lina J. Karam. BLAST-DVC: Bitplane selective distributed video coding. *Multimedia Tools and Applications*, July 2009. doi: 10.1007/s11042-009-0314-8.

[112] M. Tagliasacchi and S. Tubaro. Hash-based motion modeling in Wyner-Ziv video coding. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, volume 1, pages 509–512, 2007.

[113] J. Ascenso, C. Brites, and F. Pereira. A flexible side information generation framework for distributed video coding. *Multimedia Tools and Applications*, July 2009. doi: 10.1007/s11042-009-0316-6.

[114] Jürgen Slowack, Stefaan Mys, Jozef Škorupa, Peter Lambert, Christos Grecos, and Rik Van de Walle. Accounting for quantization noise in online correlation noise estimation for distributed video coding. In *Proc. Picture Coding Symposium (PCS)*, May 2009.

[115] H.S. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky. Low-complexity transform and quantization in h.264/avc. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):598 – 603, july 2003.

[116] Shu Lin and Daniel J. Costello. *Error control coding*. Prentice Hall, 2 edition, 2004.

[117] Jozef Škorupa, Jürgen Slowack, Stefaan Mys, Peter Lambert, Christos Grecos, and Rik Van de Walle. Stopping criterions for turbo coding in a Wyner-Ziv video codec. In *Proc. Picture Coding Symposium (PCS)*, May 2009.

[118] Y. Wu, B. D. Woerner, and W. J. Ebel. A simple stopping criterion for turbo decoding. *IEEE Communications Letters*, 4(8):258–260, 2000.