Complexity Control for Low-Power HEVC Encoding

Alexandre Mercat

THESE DE DOCTORAT DE

L'INSTITUT NATIONAL DES SCIENCES APPLIQUEES RENNES
COMUE UNIVERSITE BRETAGNE LOIRE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Signal, Image, Vision*

Par

Alexandre MERCAT

Complexity Control for Low-Power HEVC Encoding

Thèse présentée et soutenue à Rennes, le 07.12.18
Unité de recherche : IETR
Thèse N° : 18ISAR 34 / D18 - 34

Rapporteurs avant soutenance :

Marco CAGNAZZO	Maître de conférences HDR, Telecom-ParisTech
François-Xavier COUDOUX	Professeur des Universités, Université de Valenciennes

Composition du Jury :

Président :	Anissa MOKRAOUI	Professeur des Universités, Université Paris13
Examinateurs :	Marco CAGNAZZO	Maître de conférences HDR, Telecom-ParisTech
	François-Xavier COUDOUX	Professeur des Universités, Université de Valenciennes
	Frederic DUFAUX	Directeur de Recherche CNRS, L2S, CentraleSupelec
	Jean-Marc THIESSE	Ingénieur R&D, VITEC
	Maxime PELCAT	Maître de conférences HDR, INSA Rennes
	Wassim HAMIDOUCHE	Maître de conférences, INSA Rennes
Dir. de thèse :	Daniel MENARD	Professeur des Universités, INSA Rennes

Intitulé de la thèse :

Complexity Control for Low-Power HEVC Encoding

Alexandre MERCAT

En partenariat avec :

For my Dad…

Contents

Acknowledgements

First, I would like to thank my advisors Dr. Maxime Pelcat (HDR) and Dr. Wassim Hamidouche, and my thesis director Pr. Daniel Menard for their guidance, their support, and their trust for the last three years. Thank you for giving me the opportunity to pursue a PhD on such an interesting topic with freedom and in such a motivating and friendly working environment. Wassim, thank you for your well help and support and especially for your technical expertise that answered all my questions. Maxime, thank you for your organizational and technical insights and thank you for your open-mindedness on research directions which has been a great help in enhancing the value of this PhD in many research fields. Daniel, thank you for your perfect thesis supervision, you have always been there to help me, to guide me and your advice has been very precious to me.

I want to express my gratitude to Pr. Francois-Xavier Coudoux and Dr. Marco Cagnazzo for being part of the PhD committee and for taking the time to review this thesis. I also want to thank Pr. Anissa Mokraoui for presiding the PhD committee and Dr. Frederic Dufaux and Dr. Jean-Marc Thiesse for being a member of the PhD committee.

Special thanks to Pr. Jean-Francois Nezan, many thank for allowing me to start working on research topics, I owe you a lot! I also would like to thank all members of the VAADER team of the IETR for making me feel part of the team since the beginning. Special thanks to all my office-mates during these three years: Karol Desnos, Julien Heulot, Justine Bonnot and Erwan Nogues.

I want to thank Florian Arrestier, who started as my intern and now flies his own wing in thesis. Working with you has been great and I could never have done this work without your help, thanks again!

I would also like to thank my friends who supported me and with who I went to drink beers throughout my thesis: Maxime Bichon, Alexandre Sanchez, Thomas (Michel) Amestoy, Frédéric, Louis-Paul, Ronan, Quentin, Amaury, Irwin, Olivia, Félicie, Lulu, Juliette, Jessica, Julie, Romane, Aline... and my crossfit partners JF, Séverine, Quentin, Léa, Jérémie.

I thank all the players of the French football team for this second World Cup victory (during my thesis writing), two stars on our jersey now!

I am deeply grateful to my family and friends for their support and for helping me during these last three years. In particular, I would like to express my infinite gratitude to my Mother, my brothers and sisters for their unconditional support and their love.

Last but not least, I would like to thank in particular my roommate and childhood friend (already 24 years of friendship) Victorien Lorcy without whom I would not be here and with whom I spent three wonderful years, thank you!

CHAPTER 1

Introduction

Through traditional broadcasting channels such as television broadcasting or over-the-top online platforms such as Youtube or social networks, digital video is omnipresent in our lives. Video is already the vast majority of data traffic over the world; Cisco [CIS17] reports that 73% of total IP traffic is dedicated to IP video traffic in 2016 and estimates it to go up to 82% by 2021. Sandive [San16] details that 60% of North American downstream traffic in the peak evening hours on fixed access networks is recorded by four VOD services in 2016: Netflix, YouTube, Amazon Video and Hulu. This number has doubled in five years. The expansion of high-speed networks combined with the progress of microelectronics now makes it possible to consume digital video anywhere anytime.

On one end, through the emerging of new video formats like 4K Ultra High Definition (UHD), 360-degree video, multi-view, point clouds and Light Field, the amount of video data is continually increasing over time. Storage and transmission of these premium video experiences become extremely data-heavy. On the other side of the spectrum, devices available to record, display and distribute ultra-high resolutions become affordable for home and mobile consumers. Video compression gains are necessary to fuel the adoption of these immersive video technologies.

Video encoding has motivated academic and industrial researches over the past 30 years and produced appealing products and services. The fundamentals of video encoding have not changed over these years, as last video encoder codecs or standards are based on the same block-based hybrid video coding principles that have been used since Recommendation H.261 in 1988. Nevertheless, improvements of video coding tools have led to impressive reductions in bitrate for a similar objective video quality. In order to satisfy the increasing demands of storage and video transmission, the targeted key number to highlight when developing new standard is "50%"; H.264/Advanced Video Coding (AVC) asserted 50% less bitrate than MPEG-2/H.262 and High Efficiency Video Coding (HEVC) asserted 50% less bitrate than H.264/AVC. At the present time, the HEVC [SBS14, Wie15] standard, designed in early 2013 [SOHW12], represents the state-of-the-art in video coding.

Parallel to the bitrate reduction, most new video coding tools introduced inside standards carry a very heavy computational complexity increase on both decoding and (even more) encoding sides. The community consider as normal that a newer and more efficient video coding standard requires more processing power than former codecs. Globally, a factor of between 5x and 10x in computational complexity of the encoding process is introduced with each generation of video standard. At the same time, the number of devices

connected to IP networks will be more than three times the global population, reaching 27 billion network devices by 2021 [CIS17] and a large share of these devices will be embedded. As a consequence, a new context is emerging where connected devices with video capabilities will be ubiquitous.

The main limitation of recent embedded systems, particularly in terms of computational performance, comes from the bounded energy density of batteries. Smart management of embedded systems and their energy use are therefore crucial to support new features without altering the Quality of Experience (QoE). Energy limitation is a major constraint for image and video applications, video encoding and decoding being for instance the most energy-consuming algorithms on smart phones [CH10]. A large share of systems are likely to integrate HEVC in the long run and will require to be energy efficient, or even energy aware. As a consequence, energy consumption represents a serious challenge for embedded HEVC real-time encoders. For both hardware and software codecs, a solution to reduce the energy consumption is to decrease the computational complexity while limiting compression quality losses.

1.1 Challenges and Hypothesis

The substantial increase of computational complexity becomes a real challenge to the real-time implementation of the new video coding standards on embedded platforms, limited in computing, memory, and energy resources. New 4K UHD and 360-degree video content that enter the market with very high spatial (8K, 16K) and temporal (120 Frames per Second (FPS)) resolutions make it more difficult to achieve real-time coding and decoding performances.

The progress of microelectronics made in the recent years in the field of multi-core embedded platforms does not compensate for the ten-fold increase of computational complexity of encoding process. It is necessary to work on algorithmic aspects to reduce the complexity of the HEVC encoder while preserving the bitrate and quality provided by the new coding tools. The bounded energy density of batteries of recent embedded systems requires designers to go further in methods for reducing the computational complexity and proposed methods to scale and control the complexity, and therefore energy consumption of the HEVC encoding process.

In the context of embedded real-time HEVC encoding, research conducted in this document is focused on the All Intra (AI) configuration. Ultra-low latency video encoding is especially tailored to real-time applications. We choose an AI configuration to take into account the limited computational power of embedded systems in order to achieve ultra-low latency encoding. Moreover, the AI configuration has the lowest memory footprint which is significant in an embedded context.

1.2 Contributions

This document proposes a set of studies and methods which has the final purpose of scaling and controlling the complexity and therefore the energy consumption of the HEVC encoding process. When any energy management mechanism is used, the energy consumption is quasi-proportional to the execution times and thus the complexity. Energy, execution time, complexity metrics are considered in this document.

To properly address the problematic of complexity/energy-aware video coding systems, the initial stage is to study and identify the most computationally intensive process parts. We analyze energy reduction opportunities by considering different levels of granularity

from global video parameters to prediction unit determination [MAH+17b]. The study demonstrates that at the coarsest granularity, the energy of HEVC real-time Intra encoding process is proportional to video resolution. At a lower granularity, the frame partitioning structure determination has the largest potential of energy reduction.

To determine the best frame partitioning structure, called quad-tree partitioning in HEVC, the Rate-Distortion Optimization (RDO) encoding process explores a bounded search space of solutions and selects the solution leading to the minimum cost in terms of bitrate and quality. To control the complexity/energy consumption of this type of algorithm, we propose a methodology, named Smart Search Space Reduction (SSSR), defining a straightforward flow that exploits the computation-skipping approximate computing concept in order to explore the trade-offs between algorithm degradation and computational cost reduction [MBP+17b, MBP+17a]. Using a first prediction solution, the SSSR provides early selection of the best candidate configurations to reduce and scale the search space according to a budget of complexity/energy.

Figure 1.1 illustrates the contribution organization of this document. State-of-the-art chapters are displayed in white and contribution chapters in gray. The main contributions of this document can be summarized in three axes.

1. **"One-shot" quad-tree partitioning prediction:**

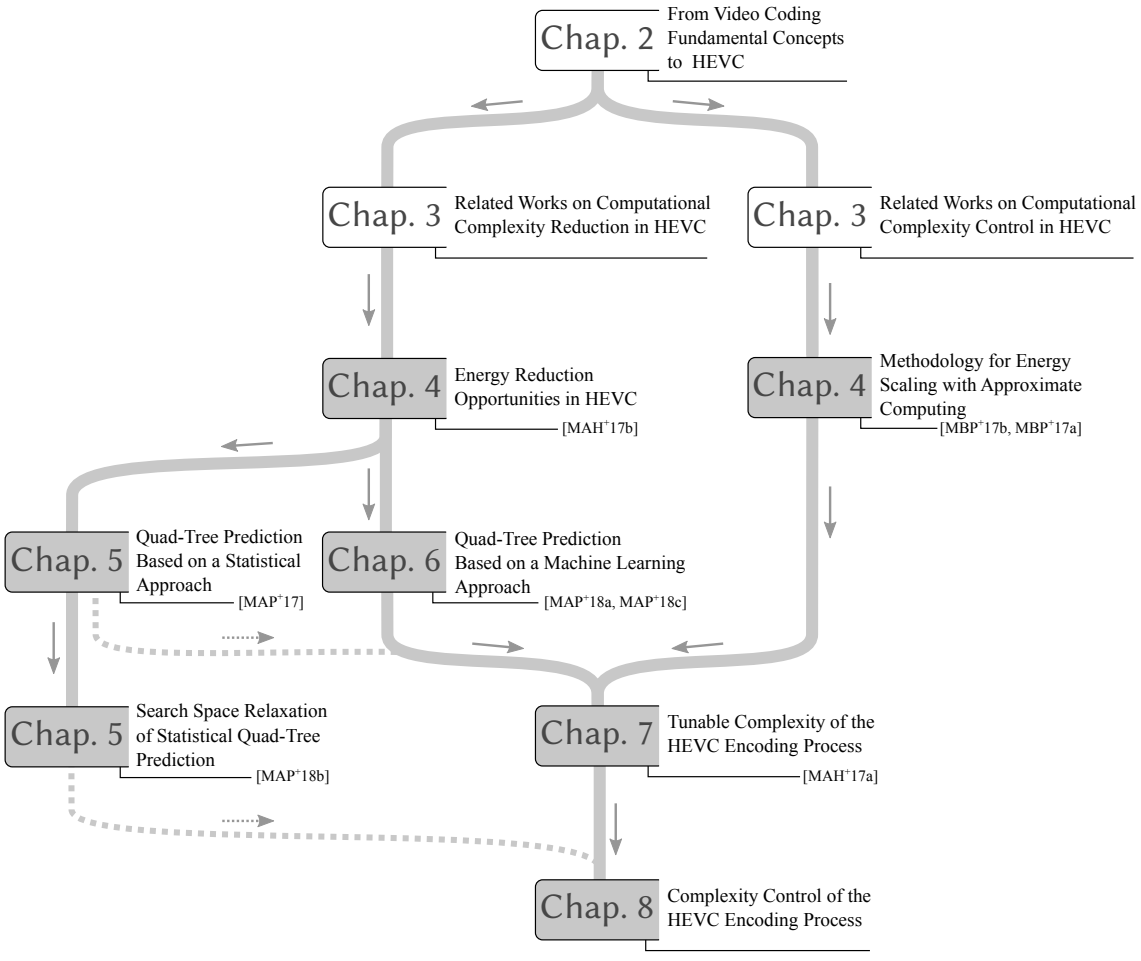


Figure 1.1 – *Summary of the contribution organization of this document. State-of-the-art chapters are displayed in white and contribution chapters in gray.*

- We first propose a method to predict in "one-shot" the quad-tree partitioning based on a variance-aware statistical approach [MAP+17]. The proposed prediction algorithm exploits the correlation between a block partitioning, the complexity of its texture and the variance of its luminance samples.
- Considering that the luminance samples variance can be used to predict the frame partitioning structure, we conduct a fair comparison of characteristics used for the prediction of quad-tree partitioning. Candidate characteristics are extracted from state-of-the-art studies using Machine Learning (ML) approach [MAP+18a]. Machine Learning is an interdisciplinary subfield of computer science that aims to replace the manually engineered solutions for extracting structured information from sensored data, and this in all application fields. The various compared characteristics are used as learning features in a Machine Learning framework. The comparison is realized under a real-time framework considering both information gain for representing characteristic suitability, and computational overhead to obtain the characteristic values. The best characteristics are used in a second method of "one-shot" prediction of quad-tree partitioning based on a Machine Learning approach across data-mining classifiers which achieves more accurate predictions than the statistical one. [MAP+18c].

2. **Complexity/energy scaling of HEVC encoding process:**

- Based on an extended study of the "one-shot" quad-tree partitioning based on a variance-aware statistical approach, the prediction is moreover adjustable with tuning parameters, offering a trade-off between energetic gains and compression efficiency [MAP+18b].
- From the partitioning prediction and following the SSSR methodology, we propose a generic tunable complexity scheme for the HEVC Intra encoding process. The scheme expands the search space around the coarse partitioning prediction and exploits a method, called Constrain the Docile CTU (CDC), to allocate the complexity in a frame while minimizing the performance loss in terms of bitrate and quality [MAH+17a]. The Machine Learning approach of "one-shot" quad-tree partitioning prediction is used as coarse partitioning prediction but the proposed tunable complexity scheme is generic and could take other partitioning prediction as input (as illustrated in Figure 1.1 by the dots line). The proposed solution is able to scale the complexity of the encoding process from full complexity until the coarse partitioning prediction.

3. **Complexity/energy control of HEVC encoding process:**

- To demonstrate the utility and the applicability of the major contributions of this document, we finally propose a real-time control system that dynamically manages the encoding process to keep the encoding complexity under a specific target. The control system uses Proportional–Integral (PI) feedback loop mechanism to maintain the encoding time, expressed in seconds by frame, under the required target.

The contributions of Chapter 7 and 8 are not specific to the previous contributions and could be applied on other quad-tree partitioning predictions and tunable complexity schemes of encoding process, respectively, as illustrated on Figure 1.1 by the dots lines.

1.3 Outline

Chapter 2 presents the fundamental concepts of video compression necessary to understand this document, from the history of video compression across the main standards and codecs, through the classical hybrid video compression scheme and the Rate-Distortion Optimization (RDO) process, to the specific video encoding standard HEVC.

Chapter 3 presents firstly a global overview of state-of-the-art research on computational complexity reduction in the HEVC encoding process. Then, methods of computational complexity control of the HEVC encoding process are detailed in Chapter 3.

Chapter 4 makes the connection between the approximate computing concepts and the applicative HEVC encoding process in order to control its energy consumption. In a first step, the chapter proposes a methodology to exploit the computation-skipping approximate computing concept. The methodology, named SSSR, explores at design time the Pareto relationship between computational complexity and application quality. In a second step, the chapter presents an analysis of the energy reduction opportunities offered by an HEVC encoder. Minimal Energy Point (MEP) is introduced, across energy reduction search space, representing the boundaries of energy consumption for a given configuration. The impact on energy consumption of encoding tools at various levels of granularity is measured.

Chapter 5 proposes a new lightweight method to predict in "one-shot" the quad-tree partitioning with the objective to budget the energy consumption of a real-time HEVC Intra encoder. The proposed energy reduction technique exploits the correlation between the quad-tree partitioning and the variance of the block luminance samples to predict the quad-tree partitioning before starting the RDO process. The predictor is also made adjustable, based on two parameters that provide a trade-off between energy consumption and coding efficiency.

Image features are then compared for their capacity to feed a prediction of quad-tree partitioning under a real-time framework. Chapter 6 presents a second method of "one-shot" prediction of quad-tree partitioning for the HEVC Intra encoders. The prediction is based on an ML approach across data-mining classifiers. This quad-tree partitioning prediction is then used to drastically reduce the energy consumption of the encoding process under a real-time framework.

Chapter 7 presents the algorithm used to expand the search space around the "one-shot" prediction of quad-tree partitioning. Combined with the in-frame complexity allocator, named CDC, the chapter proposes a tunable complexity frame encoding scheme able to scale the complexity of the encoding process until the coarse partitioning prediction.

Chapter 8 presents an encoding time control scheme for HEVC Intra encoder. Gathering the main contributions presented in this document, this chapter is used as a showcase for demonstrating the utility and the applicability of the whole of this document. The proposed solution adds a feedback loop mechanism to the tunable complexity frame encoding scheme to limit the encoding time by frame under a selected target expressed in seconds.

Chapter 9 concludes this document and proposes several research directions to extend this work for future research.

CHAPTER 2

Video Coding Background

2.1 Introduction

Video compression is a specific case of data compression, which consists in reducing the amount of data to represent the video, while minimizing the impact on the visual quality of a video sequence. The main goal of video compression algorithms is to reduce video files size for storage and transmission over constrained bandwidth networks. This chapter introduces the notions of video compression necessary to understand this document. Section 2.2 starts by presenting the fundamental concepts of video compression, before detailing the specific video encoding standard High Efficiency Video Coding (HEVC) in Section 2.3.

2.2 Video Coding Fundamental Concepts

This section firstly walks through the evolution of video compression across the main standards and codecs developed since the 1980s in Section 2.2.1. After presenting the classical hybrid video compression scheme in Section 2.2.2 on which are based on most of the video encoders, the video distortion metrics commonly used are detailed. Finally, Section 2.2.4 focuses on a specific part of the encoding process, the Rate-Distortion Optimization (RDO) process.

2.2.1 Video Compression History

Video coding concept consists in reducing the amount of data used to represent a digital video signal. Video coding is based on a visual degradation/compression ratio trade-off. The coding is lossy or lossless depending on the considered application. In the case of lossy coding, it is necessary to compromise between the amount of data (bitrate) and the distortion (quality) expressed with objective metrics. Figure 2.1 presents a chronological view of the different compression technologies and standards. Only the syntax and decoder are standardized allowing any decoder complying with the standard to decode the bitstream properly and reconstruct the video sequence.

In 1984, International Telephone and Telegraph Consultative Committee (ITTCC) presents the first digital video coding technology standard H.120 [CCI88]. The first version of H.120 used conditional replenishment, scalar quantization, variable-length and Differential Pulse-Code Modulation (DPCM) coding techniques to transmit National Television

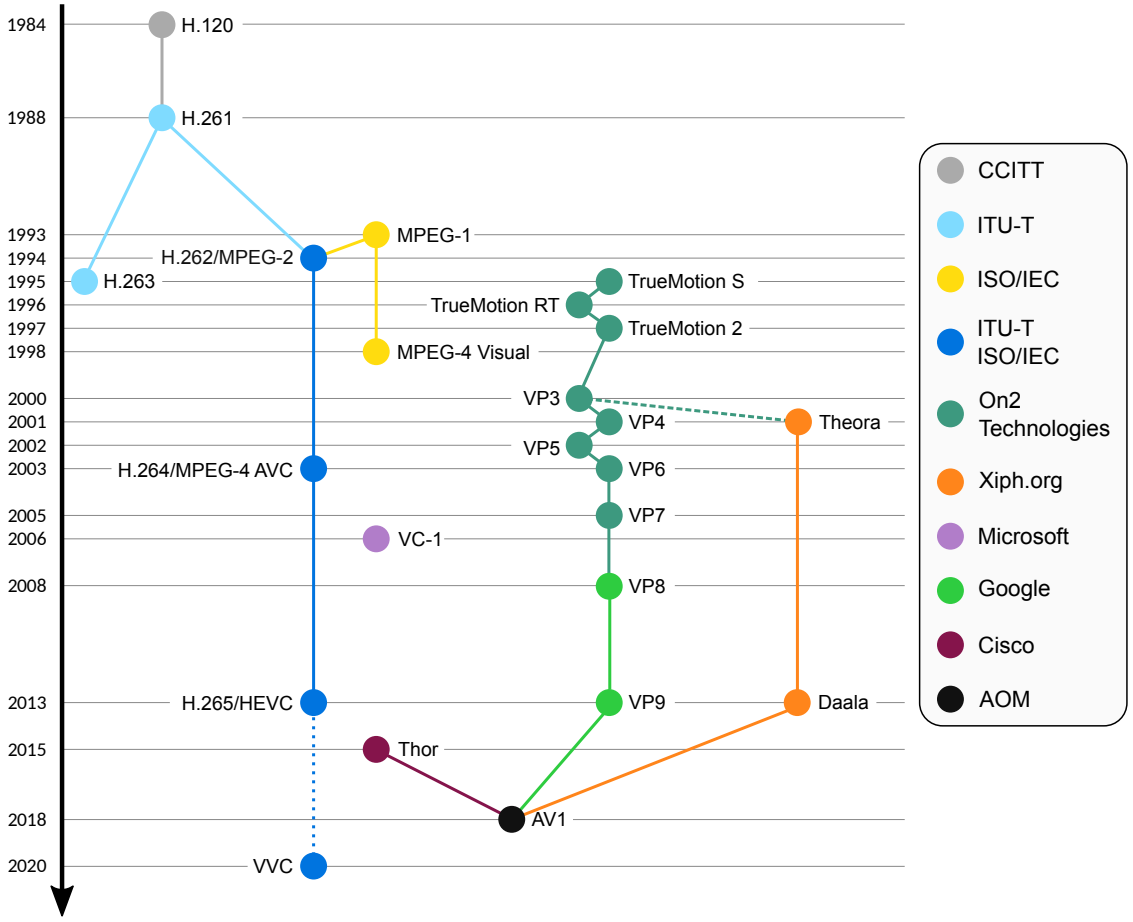


Figure 2.1 – *History of video coding technology and standards over the years.*

System Committee (NTSC) or Phase Alternating Line (PAL) video contents. DPCM predicts the input data, and encodes only the remaining difference. The second version added motion compensation and background prediction in 1988.

The ITTCC has been merged with the International Telecommunication Union (ITU) to form the ITU-T group. The specific group in charge of the video compression projects, called IUT-T Video Coding Experts Group (VCEG) presented the first widespread practical success video encoder, called Recommendation (Rec.) H.261 [IT93] in 1988. The goal of Rec. H.261 was to transmit video with multiples of 64 kbits/s data rates and CIF (352x288 pixels) or QCIF (176x144 pixels) resolutions. Rec. H.261 was the first technology using a hybrid video coding scheme (detailed in Section 2.2.2) that is today still the basis for many video coding standards. The first design of Rec. H261 embodied typical structure still dominating today:

- 16x16 macroblock motion compensation;
- 8x8 Discrete Cosine Transform (DCT), scalar quantization;
- zig-zag scan;
- Huffman-based variable-length coding.

In parallel with VCEG video standardization activity, the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) established a collaborative standardization group in 1988, called ISO/IEC Moving Picture Ex-

perts Group (Mpeg), which aimed at developing standards for the representation of moving pictures and/or audio. In 1993, the ISO/IEC ratified the first standard developed by Mpeg named Mpeg-1 [ISO93]. The Mpeg-1 standard inherited most of technical features of H.261 but also added new ones included:

- bi-directional motion prediction;
- half-pixel motion;
- slice-structured coding;
- DC-only "D" pictures;
- quantization weighting matrices.

The Mpeg-2/H.262 [ISO99, IT99] standard was jointly developed and approved in 1993 by the ISO and ITU standardization organizations across the Mpeg and VCEG teams. Inherited from both H.261 and Mpeg-1, the Mpeg-2/H.262 standard added two primary new technical features: the support for interlaced-scan pictures and the increased DC quantization precision but also various forms of scalability (SNR, Spatial, breakpoint) and I-picture concealment motion vectors. This standard encountered success and massive adoption through DVD, broadcast and broadband industry such as high-definition DTV. This joint collaboration did not prevent these two groups from developing other standards on their own.

The Rec. ITU-T H.263 [IT95] was developed in 1995 by the ITU-T and overtook H.261 as dominant videoconferencing codec. The baseline algorithm features included:

- half-pixel motion compensation;
- 3-D variable length coding of DCT coefficients;
- median motion vector prediction;
- delectable Group of Pictures (GOP) header overhead;
- increased motion vector range with picture extrapolation;
- PB-frames (bi-directional prediction);
- arithmetic entropy coding.

The second version Rec. ITU-T H.263v2 standard improved compression efficiency by 15-25% over the first version. Rec. ITU-T H.263v2 allowed custom and flexible video formats and became error resilient.

In the same way, Mpeg independently developed Mpeg-4 Visual [ISO95] containing the Rec. ITU-T H.263 baseline design. Mpeg-4 Visual also included many new extras:

- error resilience across packet loss enhancements;
- segmented coding of shapes;
- zero-tree wavelet coding of still textures;
- coding of synthetic and semi-synthetic content;
- 10 and 12-bit sampling.

Both Rec. ITU-T H.263 and Mpeg-4 Visual significantly improved compression efficiency, but could not knock over Mpeg-2, especially for broadcasting applications.

The Joint Video Team (JVT) was officially created in 2001, bringing together experts from Mpeg and VCEG, to finalize a more efficient standard than Mpeg-2, called

H.264 [IT03] or Advanced Video Coding (AVC) [ISO03] which has been initiated in 1998 by VCEG. The Rec. H.264 was ratified and published in 2003 and consists in an improvement of Mpeg-2 with additional features such as in-loop filtering for example. H.264/AVC is the most widely used codec on earth, surpassing Mpeg-2, especially in telephone companies and internet video provider (such as Youtube). It is also the codec that drives Blu-ray and many broadcast pipelines and distribution channels have adopted H.264/AVC as well.

In 2010, the Joint Collaborative Team on Video Coding (JCT-VC) group, bringing both Mpeg and VCEG experts, was created with the mandate of reducing by 50% the bitrate for the same visual quality compared to H.264/AVC. High Efficiency Video Coding (Hevc) [IT13] was finalized and ratified as Rec. H.265 [ISO13] in 2013. The Hevc standard reduces the bit rate by 50% on average compared to H.264/AVC for a similar objective video quality [OSS+12, TWM+16]. Hevc is detailed in Section 2.3.

In parallel, other groups also proposed video coding standards. A private company called On2 Technology developed the TrueMotion S video codec for 3D-rendered video in 1995. Two improved versions called TrueMotion RT and TrueMotion 2 were released in 1996 and 1997 respectively. In 2000, On2 Technology proposed a video codec for natural scenes coding called VP3. The VP3 codec became royalty-free allowing the Xiph organization to design its first open-source Theora [Fou01] standard in 2001. Several generations of VP codec were released including VP4, VP5, VP6, VP7 and VP8 in 2001, 2002, 2003, 2005 and 2008, respectively. The VP8 [BWX11] standard was announced in 2008 with a comparable coding efficiency to H.264/AVC [FWRR11] standard. In February 2002, Google acquired On2 Technology and made all its standards royalty-free, under BSD-License agreement, within the WebM project. Google proposed the VP9 [MBG+13] standard which improved the previous one VP8.

In parallel, Microsoft standardized in 2006 its proprietary coding technology called VC-1 [KL07], which became a serious alternative to H.264/AVC because of its use within HD-DVD and Blu-ray discs. The association of Mozilla Foundation and Xiph organization released their latest codec named Daala [VTE+16] in 2013 which was announced as a royalty-free open-source alternative to VP9 and Hevc. In 2015, Cisco Systems proposed the royalty free video codec named Thor [BDFM16] as an open-source alternative to Hevc.

In March 2018, the industry consortium Alliance for Open Media (AOM) composed of Amazon, Apple, ARM, Cisco, Facebook, Google, IBM, Intel Corporation, Microsoft, Mozilla, Netflix, and Nvidia as founding members presented AV1: a new open video standard and format as a successor to VP9 and a royalty-free alternative to Hevc. This new standard is based on the VP9 standard as well as improvements from the Daala and Thor video codecs.

Finally, Joint Video Expert Team (JVET), founded in 2015 and bringing together experts from Mpeg and VCEG, is working on the Hevc successor standard. This new video coding standard is named Versatile Video Coding (VVC) [Sul18] and is expected by the end of 2020.

2.2.2 Hybrid Video Compression

As explained in Section 2.2.1, the hybrid video coding scheme introduced by H.261 [IT93] is the basic encoding structure for all video coding standards and recommendations of the ISO and ITU standardization organizations. The video coding scheme is called *hybrid* to express the combination of temporal prediction between frames of the video sequence and 2D spatial prediction within the frame. Hybrid video compression is based on four signal processing operations: intra and inter frame predictions, transformation, quantization and

entropy coding. Figure 2.2 shows a simplified overview of the basic structure of the hybrid video coding scheme.

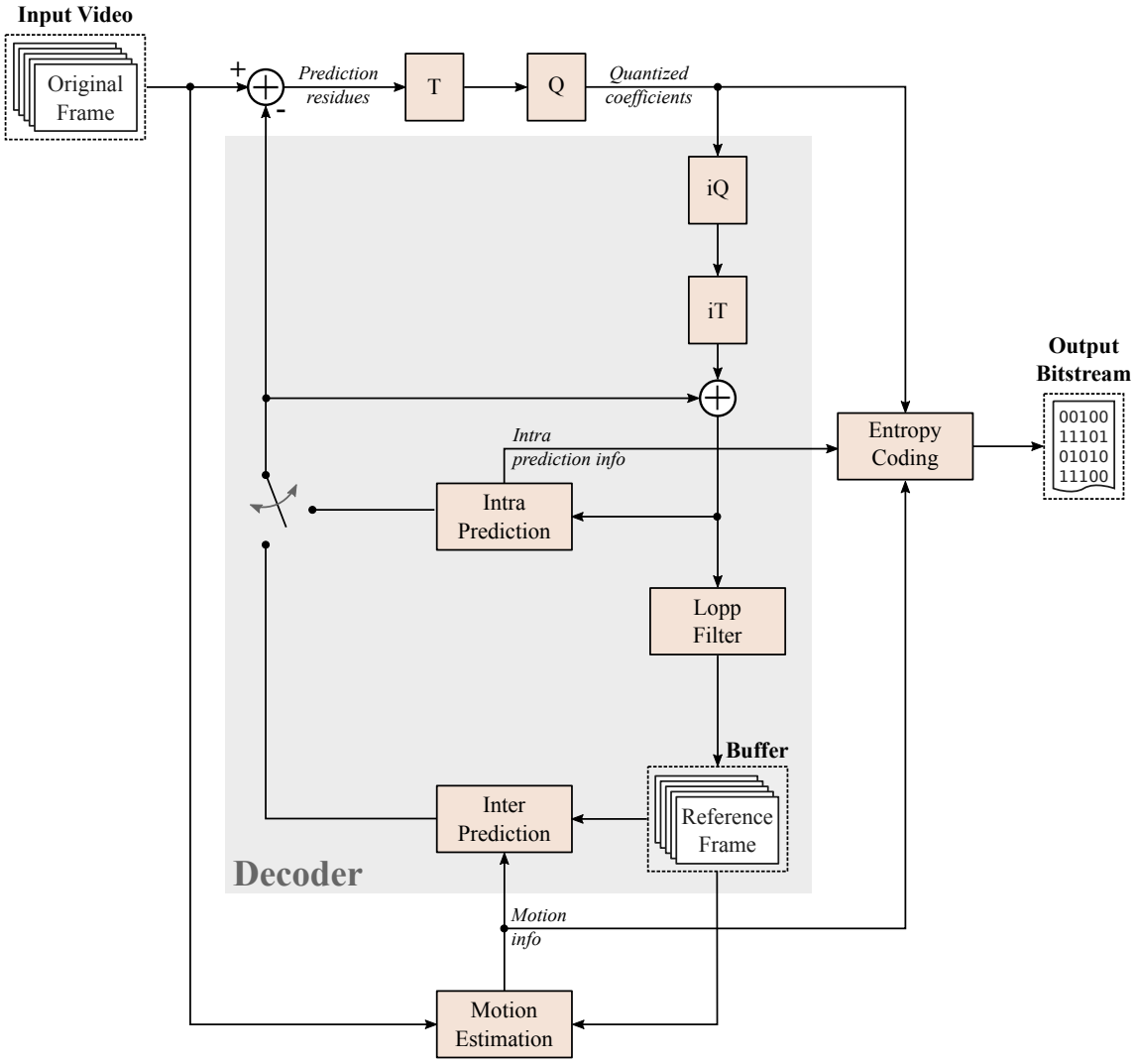


Figure 2.2 – *Hybrid video encoder block diagram.*

The encoder takes as input the frames of the input video sequence. The original input samples are predicted using intra or inter frame prediction and subtracted from the original input samples which eliminates *redundant* information from the input data. The residual signal, presenting the resulting prediction error, is transformed to convert the values from the spatial to the frequency domain in order to also concentrate the energy on a reduced number of coefficients. Then, a quantization step is applied on the transformed coefficients to remove *irrelevant* parts of the information. The *irrelevant* parts of the information are composed of small values associated to spectral components, decreasing the amount of data to be encoded without losing too much important visual information. Finally, the quantized transform coefficients are encoded into the bitstream. The prediction parameters needed to reproduce the prediction signal at the decoder side are also encoded into the bitstream.

Since the data used to compute the residual signal sent to the decoder must be exactly the same data used by the decoder to reconstruct the decoded images, a decoder step is present in the encoder, marked by a gray box in Figure 2.2. This decoder part is

composed by an inverse quantization and inverse transform steps that reconstruct the residual information. The residual is added to the predicted samples to generate the decoded frame (also called reconstructed samples). Using the decoder picture buffer as reference samples for the prediction, this decoder part assures that intra and inter prediction in the encoder uses the same sample values as the decoder.

Inter prediction reduces redundant information using Motion Compensation (MC), which is based on the fact that the difference between adjacent frames in the video sequence results from camera and object motions. The Motion Estimation (ME) stage searches for the best prediction available for the current picture block in the decoded picture buffer. In intra prediction, blocks are predicted from the neighboring pixels of reconstructed (previously encoded) blocks. Depending on the encoder decision, the best prediction is selected which can be either intra or inter prediction.

2.2.3 Distortion Metrics

In order to qualify the quality of a video encoding, it is possible to use evaluation metrics which can be objective or subjective. Objective metrics, such as Peak Signal-to-Noise Ratio (PSNR) or Structural SIMilarity (SSIM), mathematically qualify a video by analyzing the video signal data. On the other hand, subjective metrics qualify a video based on scores given by a set of human testers. Subjective metrics are also used to define psycho-visual tools qualifying the video encoding based on human observations. For example, subjective metrics can identify areas where the eye will look and where it may be interesting to improve quality by increasing the coding rate.

2.2.3.1 Peak Signal to Noise Ratio (PSNR)

The most used and known objective metric is the Peak Signal-to-Noise Ratio (PSNR) [Ric04] which measures the distortion between a signal and its source. The PSNR is based on the Mean-Squared Error (MSE) that is also a distortion metric by itself quantifying the difference between the samples from two images. The MSE is defined by Equation 2.1 where M and N are the spatial dimensions of the 2D video signal, O and R are respectively the original and reconstruct luminance or chrominance samples.

$$MSE = \frac{1}{M \cdot N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (R_{i,j} - O_{i,j})^2 \tag{2.1}$$

The PSNR is computed from the MSE using the Equation 2.2 where 2^{B-1} represents the maximum value of a sample with B the number of bits per sample.

$$PSNR = 20 \cdot \log_{10} \left(\frac{2^{B-1}}{\sqrt{MSE}} \right) \tag{2.2}$$

The PSNR is expressed in dB, the higher its value, the better is the quality. The PSNR can be calculated on an image or on a video sequence.

2.2.3.2 Structural SIMilarity (SSIM)

The Structural SIMilarity (SSIM) [WBSS04] quality metric measures the difference between two images. Unlike the PSNR, the SSIM measures the difference by structural area and not by pixel. The defined processing window can move in the image. The SSIM between two area x and y of samples is computed using Equation 2.3 where μ_x is the average of x,

μ_y is the average of y, σ_x^2 is the variance of x, σ_y^2 is variance of y, σ_{xy} is the covariance of x and y, c_1 and c_2 two stabilization variables given by Equation 2.4 with B the number of bits per sample and k_1 and k_2 two coefficients (set at 0.01 and 0.03 by default).

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2.3}$$

$$c_1 = (k_1 \cdot 2^{B-1})^2 \quad \text{and} \quad c_2 = (k_2 \cdot 2^{B-1})^2 \tag{2.4}$$

2.2.3.3 BD-BR and BD-PSNR

To compare and evaluate the performance of encoding tools, the most widely used methods are called Bjøntegaard metrics [Bjo01]. Based on the PSNR metric detailed in Section 2.2.3.1, the Bjøntegaard metrics measure the average difference between two Rate-Distortion (RD) curves interpolated through four bit-rate points. In general terms, the Bjøntegaard Delta Bit Rate (BD-BR) reports the average bit rate difference in percent for two encodings at the same quality: level measured with PSNR. Similarly, the Bjøntegaard Delta PSNR (BD-PSNR) measure the average PSNR difference in decibels (dB) for two different encoding algorithms considering the same bit rate.

The method approximates the Rate-Distortion (RD) curve using a third-order logarithmic polynomial fitting applying on a set of N bit rate values $(R_1, R_2, \ldots, R_N)$ and their corresponding PSNR measurements $(D_1, D_2, \ldots, D_N)$ as defined in Equation 2.5 and Equation 2.6:

$$D := f_1(R) = a_1 + b_1 \cdot \log(R) + c_1 \cdot \log(R)^2 + d_1 \cdot \log(R)^3 \tag{2.5}$$

$$\log(R) := f_2(D) = a_2 + b_2 \cdot D + c_2 \cdot D^2 + d_2 \cdot D^3 \tag{2.6}$$

where a_1, b_1, c_1, d_1 are the fitting parameters of distortion and a_2, b_2, c_2, d_2 are the fitting parameters of rate. The Bjøntegaard Delta Bit Rate (BD-BR) and Bjøntegaard Delta PSNR (BD-PSNR) are computed using the difference between two RD curves as illustrated in Figure 2.3 and 2.4

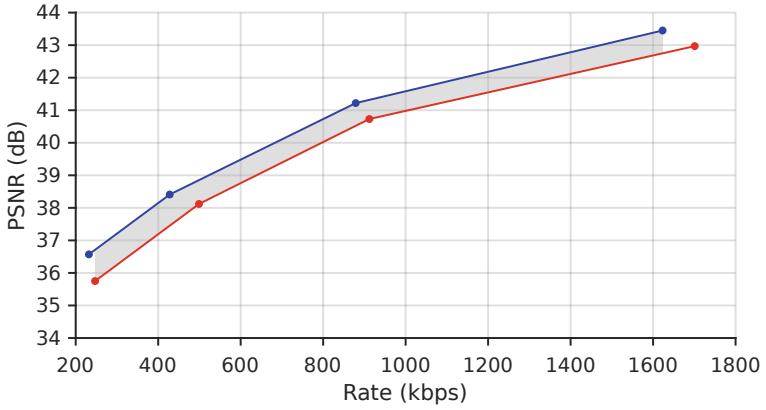


Figure 2.3 – *Illustration of BD-PSNR computation using two RD curves.*

All methods proposed in this document for reducing the complexity/energy consumption of the HEVC encoding process are compared to the original encoder version using the BD-BR and BD-PSNR metrics.

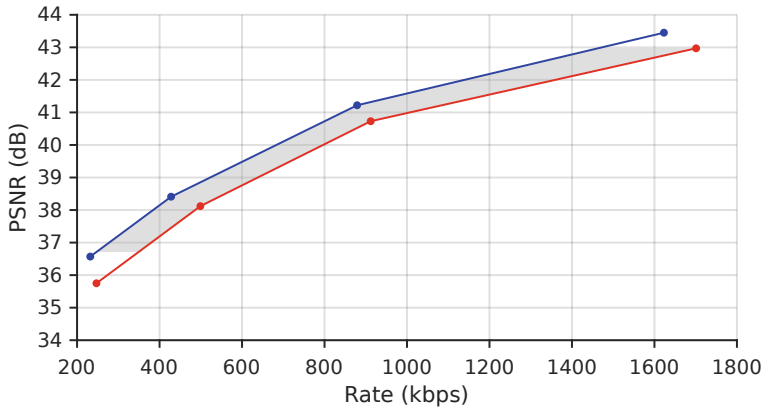


Figure 2.4 – *Illustration of BD-BR computation using two RD curves.*

2.2.4 Rate-Distortion Optimization

During the encoding process, the encoder has to choose between several coding modes including applicable predictions, the corresponding prediction parameters and block partitioning. Knowing all available coding modes and their coding efficiency, the encoder has to choose the best one for each block in each frame of the video sequence, in order to achieve optimal RD efficiency. This process, called Rate-Distortion Optimization (RDO) process [SW98], optimizes the output encoding process in terms of bitrate R and distortion D resulting from the selected coding mode under a given bitrate constraint R_c. This optimization task of the encoder can be modeled by Equation 2.7:

$$\begin{aligned} M_{\text{opt}} &= \arg\min_{M} D(M), \\ &\text{subject to } R(M) \leq R_c, \end{aligned} \tag{2.7}$$

where M represents the available mode set, M_{opt} is the optimal mode that minimizes the distortion, $D(M)$ and $R(M)$ are respectively the distortion and bitrate of the mode M and R_c is the bitrate constraint. Using Lagrangian methods, this constrained optimization problem can be transformed into an unconstrained problem described by Equation 2.8:

$$M_{\text{opt}} = \arg\min_{M} J(M|\lambda), \tag{2.8}$$

where the joint cost criterion $J(M|\lambda)$ is the RD-cost and defines by Equation 2.9:

$$J(M|\lambda) = D(M) + \lambda \cdot R(M) \tag{2.9}$$

with λ being the Lagrangian weighting factor.

Without model to describe the relation between coding modes M and their corresponding RD-cost J, the Rate-Distortion Optimization (RDO) process, by default, tests all possible coding modes to select the best one that minimize the RD-cost J according to Lagrangian factor λ. The RDO process is the source of the high complexity of the encoders.

2.3 High Efficiency Video Coding

This section presents an overview of the HEVC encoding process. An HEVC encoder is based on a classical *hybrid* video encoder that combines Inter and Intra predictions as detailed in the previous section.

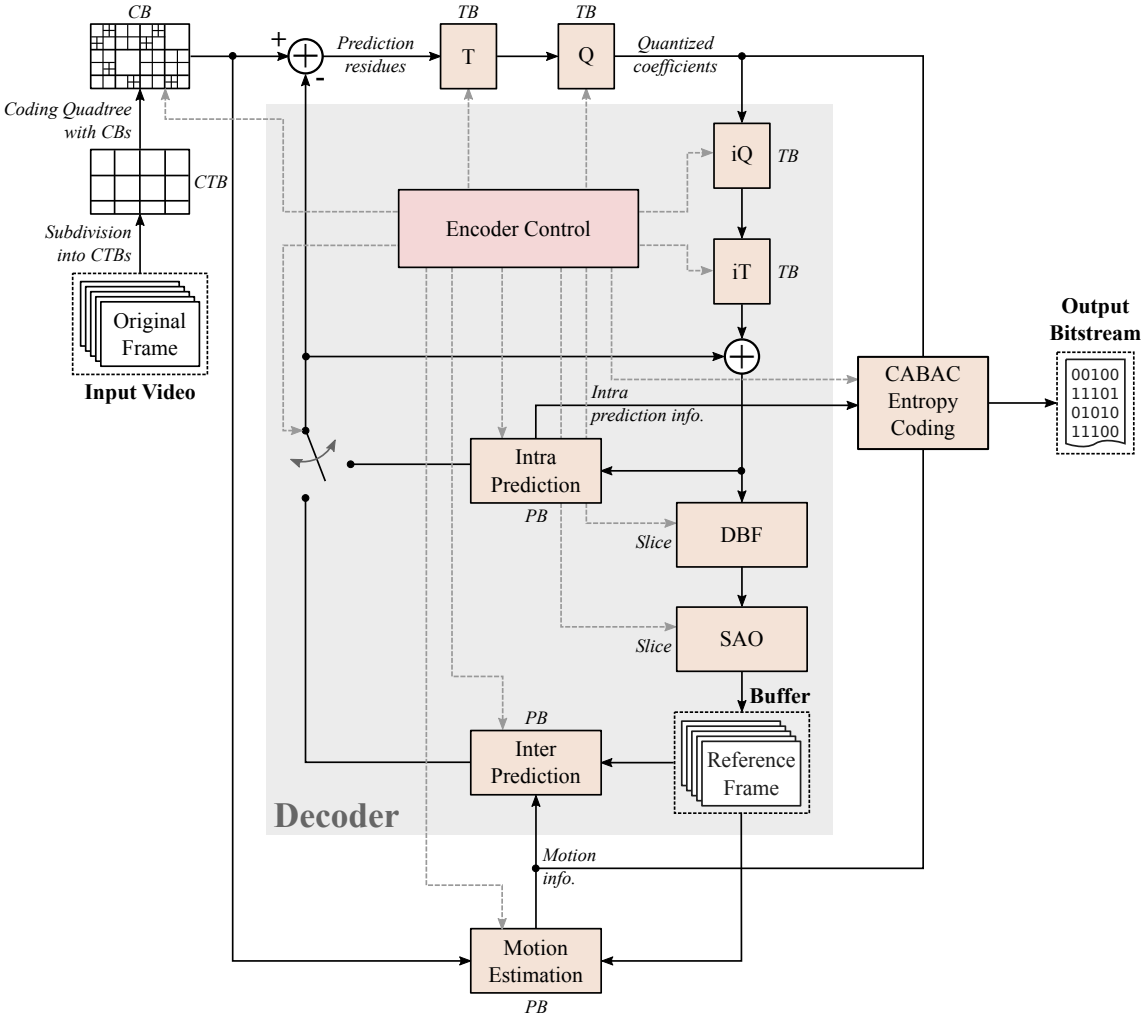


Figure 2.5 – *Block diagram of an* HEVC *encoder.*

2.3.1 Encoding Partitioning

2.3.1.1 Video Partitioning - Temporal Coding Structure

In HEVC, a video sequence is split into GOP that can include a Random Access Point (RAP), i.e. a frame that the decoder can start decoding without needing any previous frame. Figure 2.6 shows an example of a video sequence divided into GOP, where the black frames represent the RAP.

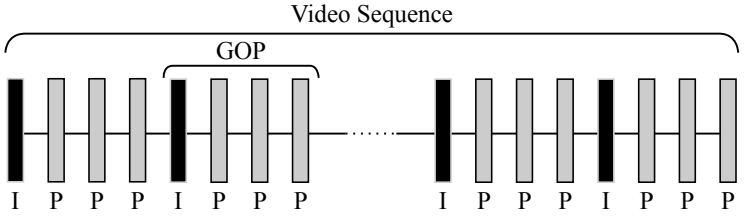


Figure 2.6 – *Example of a video sequence divided into GOP, where I and P correspond to the slice type used to encode the frames, as detailed at the end of this section.*

A GOP is composed of frames itself divided into slices. A slice is a part of the frame that can be decoded independently of other slices constituting the frame. Figure 2.7 illustrates a frame split into slices.

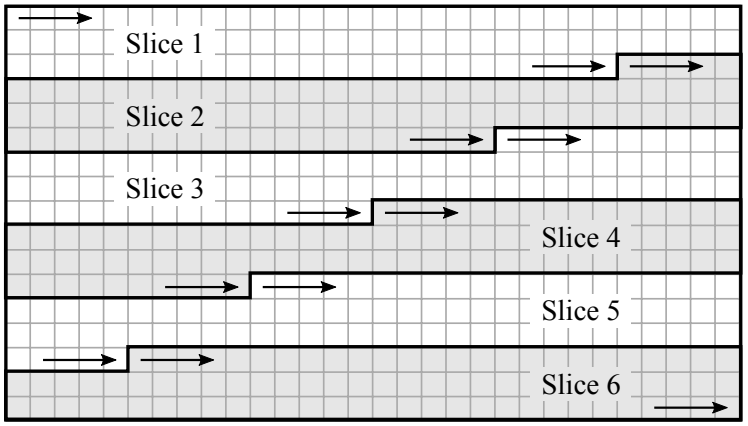


Figure 2.7 – *Example of a frame partitioning into slices.*

While encoding in HEVC, each slice is split into equally-sized blocks named Coding Tree Unit (CTU), which are further split into Coding Unit (CU). This spatial coding structure are explained in detail in the next section. Each CU of CTU belonging to a slice are encoded according to the slice type, which can be I (for Intra), P (for Predicted) or B (for Bi-directionally predicted). In I slices, only intra prediction (detailed in Section 2.3.2.1) can be used to encode CU in the slice. In P slices, CU can be encoded using both intra or inter predictions, but only one reference can be used for the inter prediction (detailed in Section 2.3.2.2); the inter prediction is unidirectional. Finally, in B slices, besides intra prediction and unidirectional inter prediction, CU can be encoded using inter prediction based on two reference frames for MC; the inter prediction may be bidirectional.

2.3.1.2 Picture Partitioning - Spatial Coding Structure

While encoding in HEVC, each slice is split into equally-sized blocks named CTU, as shown in Figure 2.7. A CTU is composed of one luminance Coding Tree Block (CTB) and two YUV chrominance CTB in the case of a 4:2:0 color representation. A CTB is a root node in a quad-tree image decomposition of luminance or chrominance data. Each CTB in a CTU is usually divided into Coding Block (CB), themselves nodes in the quad-tree. In HEVC, the size of CB can be equal $2N \times 2N$ with $N \in \{32, 16, 8, 4\}$. The encoder and decoder need the ability to link luma and chroma data for a given CB, thus the concept of CU is created that gathers luma and chroma CB corresponding to the same pixels. The Largest coding block (LCB) size and Smallest coding block (SCB) size available for the encoder are defined in its configuration file as the maximum and minimum allowed block size in HEVC are 64×64 and 8×8, respectively. Due to the nature of luma and chroma data, the same structure is usually applied to luminance and chrominance CTB, other than if the SCB size for chrominance block is reached.

Figure 2.8 shows an example of a 64×64 CTB split into several CB. The gray blocks in the Figure 2.8 represent the selected decomposition of the CTB after all the possible decompositions tested.

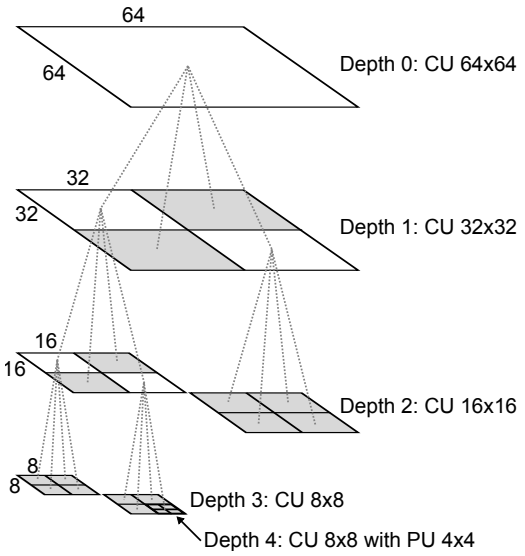


Figure 2.8 – *Quad-tree structure of a CTB divided into CB.*

2.3.2 Block Prediction

To perform Intra or Inter frame prediction, CB may be split into Prediction Block (PB) of smaller size. All PB of a CB are predicted using either Inter or Intra prediction. Figure 2.9 shows all possible PB partitioning of a CB of size $2N \times 2N$ according to the type of prediction available in HEVC.

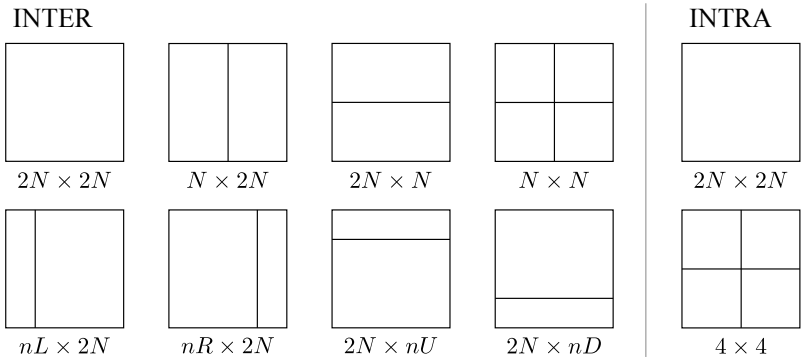


Figure 2.9 – *All possible PB splitting of a CB of size $2N \times 2N$ for Inter and Intra prediction available in HEVC, with $n = N/2$.*

In HEVC, the number of possible PB partitioning depends on the CB size. Table 2.1 specifies the available partitioning of a CB of size $2N \times 2N$ according to the prediction type and the CB size.

Table 2.1 – *PB partitioning available for each CB size in HEVC, with $n = N/2$*

CB size	PB splitting modes	
	Inter	Intra
Larger than SCB	$2N \times 2N$, $2N \times N$, $N \times 2N$, $2N \times nU$, $2N \times nD$, $nL \times 2N$, $2R \times 2N$	$2N \times 2N$
SCB	$2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$	$2N \times 2N$, $N \times N$

The rest of this section present the two type of prediction used in HEVC to predict a PB: the Intra-frame and Inter-frame prediction.

2.3.2.1 Intra-frame Prediction

The Intra-frame prediction is usually responsible for decreasing spatial redundancy by predicting the samples of the current PB from the samples of neighboring blocks already encoded in the current slice. As explained in 2.3.1.1, the CTU of a slice are encoded following the raster scan order, as shown in Figure 2.7. Due to this order, the neighboring PB already encoded in the current slice are the left and above blocks. Figure 2.10 shows an example of an $N \times N$ PB with $N = 8$ and the reference samples used for Intra prediction. The encoder needs $4N+1$ reference samples, including $2N$ samples from the left neighboring block (in blue in Figure 2.11), $2N$ samples from the top neighboring block (in red in Figure 2.11) and 1 sample of the top-left diagonal block (in purple in Figure 2.11) to predict the $N \times N$ samples of the current PB (in white in Figure 2.11).

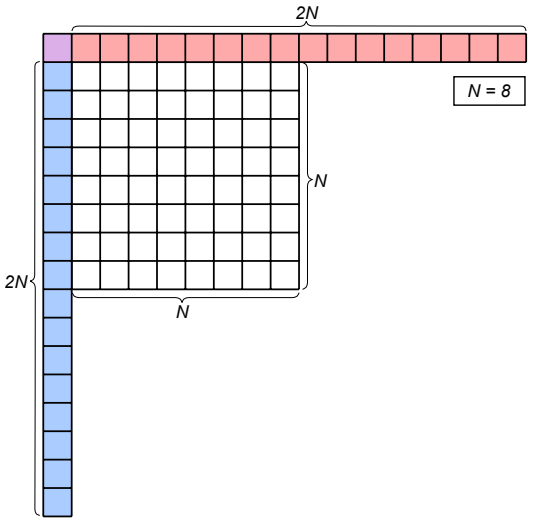


Figure 2.10 – *Neighboring samples used for intra-frame prediction of an $N \times N$ PB of size 8×8.*

The intra-frame prediction of HEVC is extensive and supports a total of 35 modes performed at the level of PB: planar (surface fitting) mode, DC (flat) mode and 33 angular modes [SOHW12] illustrated in Figure 2.11.

In the planar mode, a weighted average of four reference samples generate the prediction of one sample. The reference samples depend on the predicted sample localization in the PB. Alternatively, the DC mode predicts one sample from an average of the neighboring samples. For the 33 angular modes, the predictions copy the neighboring reference samples to predict the current PB according to the angle of the mode.

To select the best intra mode, the RDO process compute the RD-cost J_{RDO} for each prediction mode using Equation 2.10:

$$J_{RDO} = (SSE_Y + 0.57 \cdot SSE_{UV}) + \lambda \cdot R_M \tag{2.10}$$

where SSE_Y and SSE_{UV} are the Sum of Squared Errors between the original block and the predicted block for luminance and chrominance samples respectively, λ is the Lagrangian

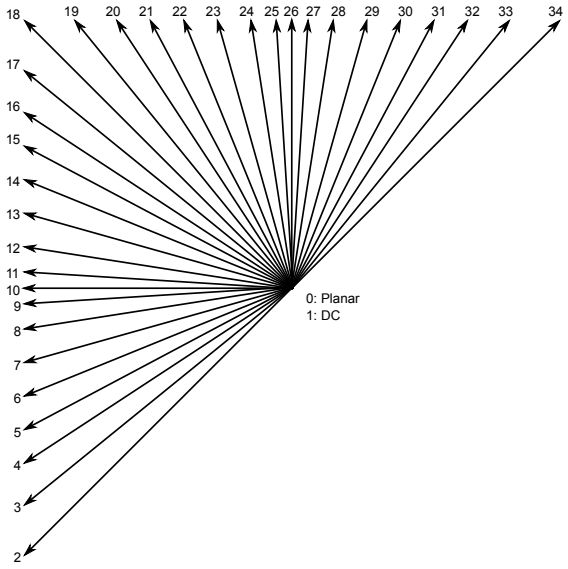


Figure 2.11 – *Intra prediction modes tested for each PB.*

multiplier and R_M is the number of bits necessary to encode the tested mode. The intra prediction mode with the smallest RD-cost J_{RDO} is then selected.

In order to decrease the computational complexity of HEVC Intra encoding, a fast intra mode decision called Rough Mode Decision (RMD) [YJJ10, ZZMZ11] was added in the reference software HEVC test Model (HM) [JV16]. This technique splits the Intra prediction process into two successive steps: RMD and RDO as illustrated in Figure 2.12.

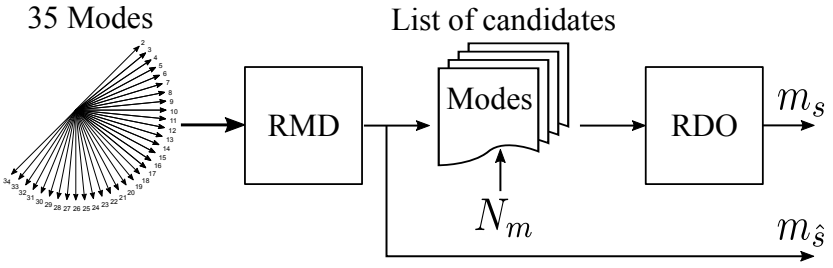


Figure 2.12 – *Intra prediction steps*

RMD consists in constructing a candidate mode list which is then tested in the full RDO process. RMD method computes for each mode m a cost $J_{RMD}(m)$ described by Equation 2.11, where D_P is the sum of absolute values of the Hadamard transformed coefficients of the prediction residue, λ_P is a Lagrangian multiplier (which depends on the Quantization Parameter (QP)) and R_P is the number of bits necessary to encode the prediction mode information.

$$J_{RMD}(m) = D_P + \lambda_P \cdot R_P \tag{2.11}$$

R_P is constant and equal for all modes different from the three Most Probable Mode (MPM), which require lower signaling (in the bitstream). The N_m modes with the lowest costs $J_{RMD}(m)$ are then evaluated by the full RDO process to select the best among them. N_m depends on the CU size N and is defined by Equation 2.12. The RDO step is much

more complex than the RMD step. As the RMD step orders the modes according to their costs, the RDO step can be skipped to limit the encoding complexity.

$$N_m \begin{cases} 8, N \in \{4, 8\} \\ 3, N \in \{16, 32\} \\ 3, N \in \{64\} \end{cases} \tag{2.12}$$

2.3.2.2 Inter-frame Prediction

In HEVC, the inter prediction is based on MC, which aims to predict PB using equal-sized areas localized according to the Motion Vector (MV) from previously encoded frames as reference. The MV vertical and horizontal components describes the displacement of the PB from the reference to the current frame. The ME process determines the MV for a PB, which is the most complex operation in terms of computational complexity of the HEVC encoding process.

In the case of 4:2:0 format, HEVC supports ME with one quarter of pixel accuracy for luminance samples and one eight of pixel accuracy for chrominance samples. When the position samples are integer values, the localization of reference pixels are directly referred to existing samples. When the position samples is fractional, the reference frame has to be re-sampled to enable the quarter/eighth pixel accuracy. Figure 2.13 shows positions of integer (in gray) and fractional (in white) luminance reference samples.

The fractional position luminance samples is obtained using one 8-tap filter and one 7-tap filter applied horizontally and vertically which generate the half-pixel and quarter-pixel samples, respectively. Table 2.2 resumes the filter coefficients for quarter-pixel interpolation $qfilter$ and for half-pixel interpolation $hfilter$ of the luma component. The fractional position chrominance samples is obtained using one 4-tap interpolation filter. A set of four filters are defined according to the distance between the fractional and the integer pixels. Table 2.2 resumes the filter coefficient $filter1$, $filter2$, $filter3$ and $filter4$ of the four 4-tap filters.

Table 2.2 – *Filter coefficients for luminance and chrominance sample interpolation.*

		Index i							
		-3	**-2**	**-1**	**0**	**+1**	**+2**	**+3**	**+4**
Luma	$hfilter[i]$	-1	4	-11	40	40	-11	4	1
	$qfilter[i]$	-1	4	-10	58	17	-5	1	–
Chroma	$filter1[i]$	–	–	-2	58	10	-2	–	–
	$filter2[i]$	–	–	-4	54	16	-4	–	–
	$filter3[i]$	–	–	-6	46	28	-4	–	–
	$filter4[i]$	–	–	-4	36	36	-4	–	–

As example, the samples $a_{0,0}$ and $n_{0,0}$ are calculated by the Equations 2.13 and 2.14 respectively where B is the bit depth of the reference samples and $qfilter$ is the filter coefficients vector for quarter-pixel interpolation.

$$a_{0,0} = \left(\sum_{i=-3}^{3} A_{i,0} \cdot qfilter[i] \right) \gg (B - 8) \tag{2.13}$$

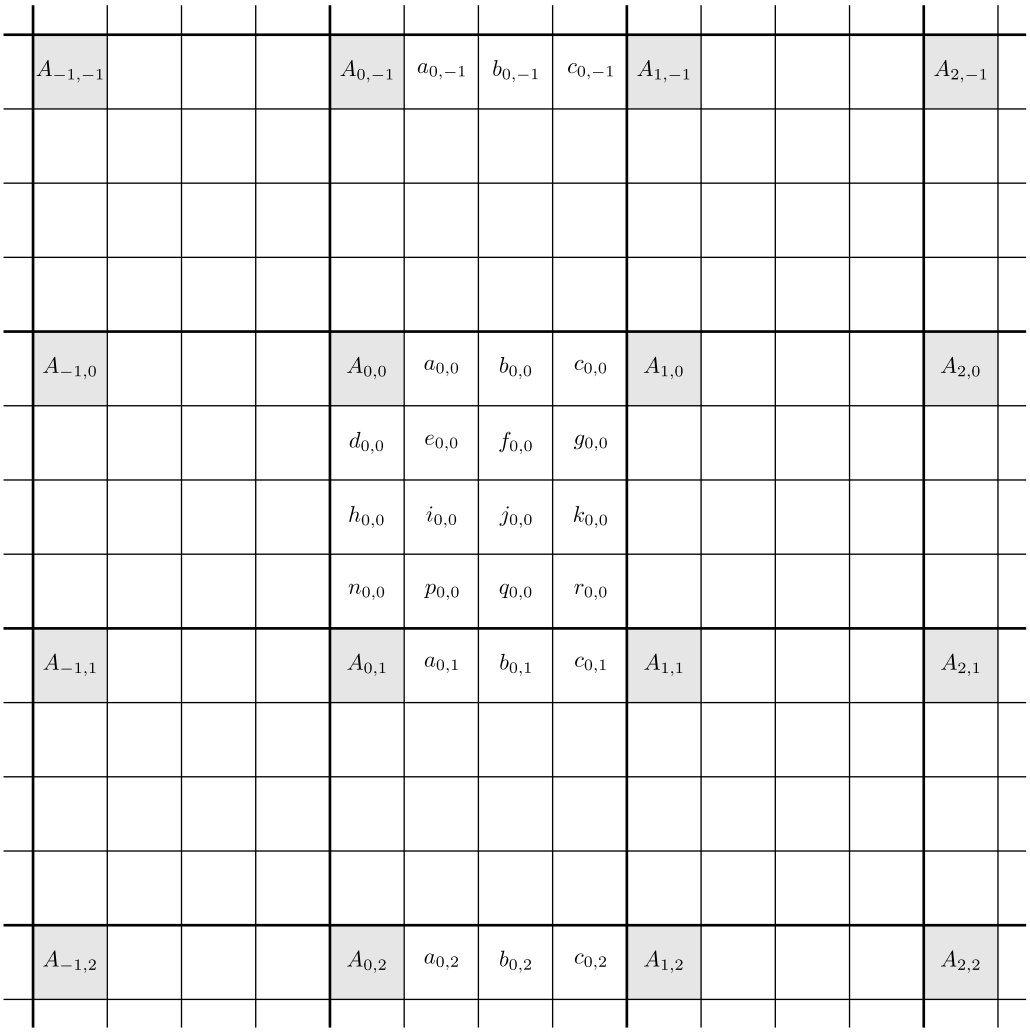


Figure 2.13 – *Arrangement of luma sample locations $A_{i,j}$ and quarter-samples locations from $a_{i,j}$ to $r_{i,j}$*

$$n_{0,0} = \left(\sum_{j=-2}^{4} A_{0,j} \cdot qfilter[1-j] \right) \gg (B-8) \tag{2.14}$$

In HEVC, ME process uses multiple reference frames. The reconstructed frames are split into two lists: *List 0* and *List 1* which are used as references for the inter prediction. *List 0* is composed of the indices of past encoded frames in display order whereas *List 1* is composed of the indices of future encoded frames in display order. A type P slice is only predicted using reference frames of *List 0* (as explain in Section 2.3.1.1), a type B slice can use one or two reference frames of both *List 0* and *List 1* for the inter prediction. In the case of bi-prediction of type B slice, the predictions performed from *List 0* and from *List 1* are averaged. Figure 2.14 give an example of available variations of uni-prediction and bi-prediction.

In HEVC, ME process aims at determining the actual MV representing the displacement of PB between the current frame and the best matching area in the reference frame. Many block-matching ME algorithms have been proposed since mid-1980's. In the search area, the best matching block is determined by searching all possible candidate blocks which leads

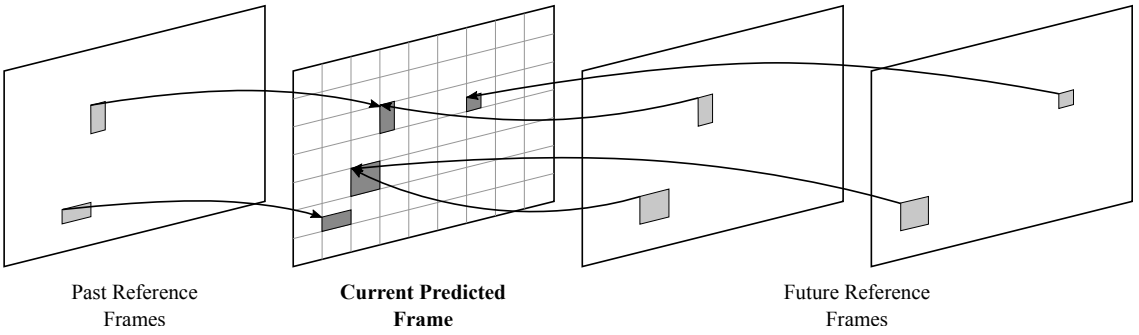


Figure 2.14 – *Illustration of uni-prediction (P slices) and bi-prediction (B slices) for the current predicted frame.*

to the optimal solution. This algorithm is called *Full Search*, it corresponds to the mathematically simplest algorithm but it requires the most computational complexity. Faster approaches have been developed to reduce the computational complexity of this process. More information about inter prediction can be found in [SOHW12, Wie15, SBS14].

2.3.3 Transformation and Quantization

In both Intra and Inter predictions, the predicted block and the current block are rarely exactly equal. The difference between them, called *residues*, are computed, encoded and transmitted. Conceptually, HEVC provides four different methods to code the residuals.

- The residues are transformed and the transformed coefficients are quantized before to be encoded into the bitstream.
- The transform step is skipped and the quantization is directly applied on the non-transformed residual signal.
- The transform and the quantization steps are skipped and thus the residues are directly encoded without transform and quantization processes.
- The transform and the quantization steps are skipped and no residue is transmitted.
- The CU samples are directly coded into the bitstream without any prediction, transform and quantization.

In the following paragraphs, the transform and quantization processes are detailed.

Transformation: HEVC uses a two-dimensional transform which is computed by applying one-dimensional transforms in the horizontal and vertical directions of the block. Transforms are applied for all Transform Block (TB) tested in the RDO process. The transformation operation is performed on the TB with permissible sizes of 4×4, 8×8, 16×16 and 32×32 samples.

The transformation is applied using integer DCT-based transform according to the TB size. In HEVC, only the 32×32 transform matrix is specified and the smaller transform matrices are derived from it by using part of its coefficients.

An alternative 4×4 integer Discrete Sine Transform (DST)-based transform is applied to 4×4 luma residue blocks resulting from Intra prediction. Due to the statistical property of the residual signal, the use of DST-based transform to encode Intra predicted blocks of size 4×4 obtains better results.

Quantization: After transformation, the transformed coefficients of a TB are quantized using a non-linear discrete mapping of the coefficient values into integer quantization indices. The quantization implementation consists in a multiplication by a constant, an addition of a rounding factor and a right shift controlled by the QP value. The quantization is implemented according to Equation 2.15, where L is the output quantized coefficient, C is the input transformed coefficient, $Q_{QP\%6}$ is the constant values, *offset* is the rounding factor and N is the TB size [GL].

$$L = \frac{C \times Q_{QP\%6} + \text{offset}}{2^{21+\frac{\text{QP}}{6}-\log_2 N}} \tag{2.15}$$

The constant $Q_{QP\%6}$ value depends on the QP value as detailed in Table 2.3 with QP%6 the remained of the division between QP and 6.

Table 2.3 – *Constant $Q_{QP\%6}$ values used in the coefficient quantization according to QP values.*

QP%6	0	1	2	3	4	5
$Q_{QP\%6}$	26.214	23.302	20.560	18.396	16.384	14.564

As a non-normative contribution, Rate-Distortion Optimization Quantization (RDOQ) [KYC08] has been adopted in HEVC to improve the encoding efficiency. When RDOQ is enabled, the encoder computes four quantized coefficient candidates per transformed coefficient in each TB and selects the best one in terms of RD efficiency.

2.3.4 Inverse Quantization and Inverse Transformation

As the HEVC encoder includes the decoder processing loop, the quantized residues are transmitting to the inverse quantization and inverse transformation.

Inverse Quantization: The inverse quantization is implemented according to Equation 2.16:

$$CQ = \frac{L \times IQ_{QP\%6} \times 2^{\frac{QP}{6}}}{2^{\log_2 N-1}} \tag{2.16}$$

where CQ is the output inverse coefficient, L is the input quantized coefficient, N is the TB size [KYC08]. The constant $IQ_{QP\%6}$ value depends on the QP value as detailed in Table 2.4 with QP%6 the remained of the division between QP and 6.

Table 2.4 – *Constant $IQ_{QP\%6}$ values used in the coefficient inverse quantization according to QP values.*

QP%6	0	1	2	3	4	5
$IQ_{QP\%6}$	40	45	51	57	64	72

Inverse Transformation: The inverse transformation is applied on the inverse quantized coefficients to generate the inverse transformed residue prediction of the TB. The inverse transformation matrix is the transposed of the corresponding matrix used in the transformation step.

2.3.5 Entropy Coding

The entropy coding in HEVC is performed using the Context-Adaptative Binary Arithmetic Coding (CABAC) process. Each syntax element is firstly converted into binary symbols. Each binary symbol is then associated to a probability model according to its context. Finally, each binary symbol is encoded on a specific number of bits according to its probability. As the CABAC process does not affect and is not directly affected by any contribution proposed of this work, it will not be detailed here. More information can be found in [SOHW12, Wie15, SBS14].

2.3.6 In-Loop Filtering

In order to improve the quality of reconstructed samples, two filtering steps are specified in HEVC: Deblocking Filter (DBF) and Sample Adaptive Offset (SAO) filters. The two filters are applied into the encoding and decoding processes in order to improve subjective/objective quality of the reconstructed images. First, DBF filter is applied to PB and TB edges to reduce visible artifacts of block structure due to the block-based approach of the coding process in HEVC. The DBF filter applies a selected filter according to the detected artifacts at the coded block boundaries. In a second step, SAO filter is applied to all samples in slice and consider more than only the block boundaries. The main function of SAO filtering is to reduce visible ringing artifact effects. SAO filter classifies the reconstructed samples in two categories: Band Offset (BO) or Edges Offset (EO). The reconstructed samples are classified in EO categories based on sample value differences of the current sample and its neighboring samples. According to edge direction on the block boundary, an estimated offset is applied to the reconstructed sample to avoid the artifacts located at the block boundaries. The BO classified samples are only evaluated according to the band samples, and an offset is also applied to the reconstructed sample.

2.3.7 Software HEVC Encoders and Kvazaar

For embedded applications, hardware encoding solutions [Qua14] consume lower energy than software solutions. However, when the considered system does not embed a hardware coprocessor, a software HEVC encoder [JV16, Mul17, Van17, Ult17] is necessary.

The HEVC reference software model (HM) [JV16] is widely used in research, as it has been designed to achieve an optimal coding efficiency (in terms of RD). However, the computational complexity of HM is high and not adapted to embedded applications. To fill this gap, the *x265* [Mul17], *f265* [Van17] and *Kvaazar* HEVC [Ult17] software encoders enables real-time encoding using parallel processing and low-level Single Instruction Multiple Data (SIMD) optimizations for different specific platforms.

Since all works of this document have been conducted under a real-time embedded framework, methods for reducing and controlling the complexity/energy consumption of the HEVC encoding process have been implemented in the real time HEVC software encoder Kvazaar. Kvazaar is an academic cross-platform HEVC encoder coordinated by Ultra Video Group [Ult17] at *Tampere University of Technology (TUT)*. Kvazaar encoder is targeted at x86, x64, PowerPC, and ARM processors on Windows, Linux, and Mac Operating System (OS). Currently, Kvazaar supports HEVC Main and Main 10 profiles for 8- and 10-bit 4:2:0 video sequence. Table 2.5 lists the main coding parameters and tools of Kvazaar.

Table 2.5 – *Kvazaar features included in the current version.*

Feature	Kvazaar HEVC Encoder
Profiles	Main, Main 10
Internal bit depth	8, 10
Color format	4:2:0
Coding configurations	AI, LP, RA, LB
Slice types	I, P, B
Parallel processing	Tiles, Wavefront
Sizes of CU	64×64, 32×32, 16×16, 8×8
Sizes of TU	32×32, 16×16, 8×8, 4×4
Sizes of PU, Intra	32×32, 16×16, 8×8, 4×4
Sizes of PU, Inter	64×64, 32×32, 16×16, 8×8, AMP, SMP
Intra prediction modes	DC, planar, 33 angular
Coding modes	Intra, Inter, Skip, Merge
ME algorithm	HEXB, FS, TZ
# of reference pictures	Up to 15
Mode decision metrics	SATD, SAD, SSD
Rate-distortion optimization	Partial (includes RDOQ)
Entropy coding	CABAC
Loop filtering	DBF, SAO
Residual coding	Coefficients, 4×4 transform skip, Trans/quant Bypass
Bitrate target	1-pass

2.4 Conclusion

This chapter firstly walks through the evolution of video compression across the main standards and codecs developed since the 1980s in Section 2.2.1. To improved compression efficiency, the complexity of the encoding process have been increase over time and standards. The Rate-Distortion Optimization (RDO) process, presented in Section 2.2.4, testing more and more modes and partitioning, is the greatest source of complexity in the HEVC encoding process (presented in Section 2.3). The next chapter presents a global overview of state-of-the-art research on computational complexity reduction in the HEVC encoding process.

CHAPTER 3

Related Works

3.1 Introduction

The state-of-the-art video coding standard HEVC was designed to substantially improve the coding efficiency with respect to its predecessor H.264/AVC. When compared with the previous H.264/AVC standard, HEVC Main profile requires less than half the bit-rate on average for a similar objective video quality [TWM+16, VVHH12]. Following the fast evolution of video telecommunication technologies, HEVC was specifically designed for higher resolution and frame rate videos, which considerably increases the computational needs of the video encoding processes. Moreover, more efficient signal processing tools were included in HEVC, further increasing the complexity of the encoding process.

This chapter presents a global overview of state-of-the-art research on computational complexity reduction in the HEVC encoding process. First, techniques of complexity reduction that constitute the state-of-the-art for HEVC are presented in Section 3.2. Then, methods of computational complexity control of the HEVC encoding process are summarized in Section 3.3. Finally, Section 3.4 concludes this chapter.

3.2 Computational Complexity Reduction in HEVC

Since the HEVC standard has been standardized, many works have been published in the literature, presenting methods to reduce the computational complexity of the encoding process. Most of these techniques aim at reducing the computational complexity due to the exhaustive RDO process. These techniques are classified into four main categories covered in the next sections:

- quad-tree decomposition (Section 3.2.1),
- Prediction Unit (PU) mode selection (Section 3.2.2),
- intra mode selection (Section 3.2.3),
- Residual Quad-Tree (RQT) decomposition (Section 3.2.4).

Figure 3.1 illustrates a classification of the computational complexity reduction techniques with their corresponding number of works that will be summed up in the following sections.

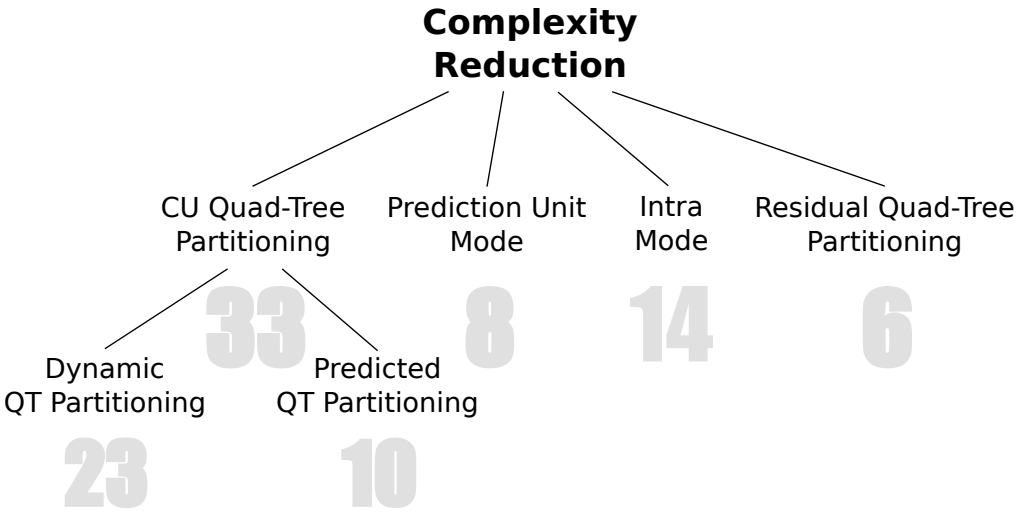


Figure 3.1 – *Overview of the computational complexity reduction techniques categories and the number of works that will be described by category.*

3.2.1 Quad-Tree Partitioning Complexity Reduction Techniques

Quad-tree partitioning, i.e. choosing the appropriate encoding block size for each part of the image, is the most time consuming operation in an HEVC encoder and thus offers the biggest opportunities for complexity reduction (up to 78% in the considered embedded encoder) [MAH+17b], as presented later in Section 4.3. Indeed, the encoding block size of a CTU has a strong impact on the Rate-Distortion (RD) ratio and the obtained visual quality. Choosing the right encoding block size results from a costly RDO process. Block sizes are also referred to as *depths*, where a shallow depth denotes a large block and a deep depth denotes a small block, as illustrated in Figure 2.8 on page 19. The exhaustive search partitioning solution is optimal in terms of RD-cost. It is obtained by encoding all possible partitioning configurations and select the one that minimizes the RD-cost. To reduce the computational complexity of HEVC encoders, several algorithmic solutions have been proposed at the level of quad-tree partitioning for reducing the number of levels to test.

Works on low complexity CTU quad-tree partitioning can be classified into two categories: the dynamic quad-tree partitioning reduction techniques and the prediction-based complexity reduction techniques. Dynamic quad-tree partitioning techniques are applied during the RDO process and are based on encoder-related information. They dynamically terminate processing when further gains are unlikely. Prediction-based complexity reduction techniques can be applied before starting the RDO process. They try to predict the quad-tree partitioning with lower complexity processing than the full RDO process. For the two categories, existing approaches are presented in the following sections. For each complexity reduction technique, the performance in terms of BD-BR and complexity reduction as well as the encoder used for the experiments are summarized in Table 3.1 and 3.2 at the end of each section.

3.2.1.1 Dynamic Quad-Tree Partitioning Complexity Reduction Techniques

Dynamic methods refer to methods that have access to the current encoded data and by-products when taking complexity reduction-oriented decisions. They are opposed to predictive methods that use the relative data-independence of the current encoding process to infer, from simple picture-based features, complexity-related information.

RDO processes traditionally start from the encoding of large blocks and recursively split blocks into sub-blocks in order to find the appropriate block size to encode. Early termination consists in stopping this process before the smallest possible size is reached. Early pruning or early split consists in skipping large blocks encoding which are probably not selected in the RDO process.

Cassa et al. [CNP12] introduce an *Early Termination* technique for fast RDO. This technique stops the CU partitioning process when the best found RD cost is already lower than a given dynamic threshold based on both spatial and temporal correlations. They add a Top Skip fast RDO techniques to reduce the complexity of the HEVC encoder by skipping some depth levels of the quad-tree partitioning. The skipped depths are selected based on the correlation between the minimum depth of the co-located CTU in the current and previous frames.

As another strategy, many complexity reduction techniques [LTLHTY12, HZLB13, HZ14, GKJ+14, TK15] early terminate the quad-tree decomposition process based on RD costs. These techniques stop the quad-tree partitioning process when the sum of RD costs of the sub-CU is higher than the RD cost of the upper CU in the quad-tree decomposition.

An early terminate CU partitioning method based on available encoding information is proposed by Ahn et al. [AKP13]. The CU partitioning decision technique uses SAO parameter values, PU size, MV size and coded block flag (cbf) data to decide whether or not the CU should be split. They improved the previous technique [ALK15] using spatial and temporal encoding parameters such as MV data.

Shen et al. [SLZ+13] propose a fast CU size decision based on image content characteristics and intermediate encoding results such as motion homogeneity, PU mode selected in the upper coding tree depth and RD costs of neighboring blocks.

A fast partitioning and pruning method for intra-coding is described by Cho et al. [CK13]. Authors propose a low-complexity RD cost calculation to decide whether or not the CU partitioning and pruning processes have to be applied. They define a Bayes-based decision rule method using statistical parameters which are periodically updated on-line, i.e. during the encoding process, to take into account the changing characteristics of the video content.

Xiong et al. [XLWM14] propose a method based on Pyramid Motion Divergence (PMD) that early terminates the RDO partitioning process of the CTU. Before encoding the CTU, the technique starts by down-sampling the entire frame and the optical flow of this frame is estimated based on the MV in order to determine the motion divergence features. For each CU, the motion divergence is evaluated as the variance of the optical flows of the current CU and its sub-CU. Xiong et al. [XLM+14] also present a method to determine whether a current CU should be further split or not, based on a Markov Random Field (MRF) inference problem, which can be optimized by a graph-cut algorithm. A maximum a posteriori approach based on RD cost is conducted to evaluate whether the non-split CU should be further split or not.

Lim et al. [KJSS15] present a computational reduction techniques composed of three part: early skip algorithm, PU skip algorithm, and PU split termination algorithm. The early skip algorithm compares the above and left PU sizes to decide whether the RD cost of the large PU should be computed or not. Both PU skip and PU split termination algorithms

are based on the RD cost ratio of the current PU and either its spatially adjacent PU or its upper depth PU. The PU skip algorithm skips the RD cost computation of the current PU and goes to the next depth PU. The PU split termination algorithm aims to terminate the encoding process of the remaining PU.

Kim and Park [KP16] propose to early terminate the CU partitioning process using a Bayesian decision rule based on on-line and off-line training. The classification problem is composed by two classes: non-partitioning and partitioning classes. The features vector of the two-class problem is composed of RD costs of inter-prediction and intra-prediction of each CU. The on-line training takes into account video content properties whereas the off-line training helps to refine the false decision making of the minimum-error-based Bayesian decision rule.

A fast CU size decision algorithm which early splits CU and early terminates partitioning process is proposed by Lee and Jeong [LJ17b]. The early partitioning method decides to perform an early-split using statistical data, including variance difference and depth level of neighboring CU. To early terminate the partitioning process, authors use an on-line learning phase to compute a RD cost threshold based on full RD cost and the variance difference.

In addition to an intra mode prediction technique (detailed in Section 3.2.3), Fernandez et al. [FDBB+18] proposed a method to identify smooth regions on the input frame to reduce the computational complexity of these specific regions. A smooth region is defined as an area within the image that possesses a homogeneous texture, which is identified using DC and AC coefficients extracted from the DCT coefficients. The method stops the iterative division process when a current block is classified as smooth. Indeed, homogeneous region should not be divided into smaller partitions due to the fact that a larger block will have a lower RD cost and splitting homogeneous block does increase the RD ratio. Authors add temporal analysis to propose a second method of fast CU size decision based on spatial and temporal homogeneity detection [FDBBG18].

Machine Learning (ML) dynamic techniques for reducing complexity of quad-tree partitioning: All previously presented methods reduce encoding complexity by assuming a given relationship between some dynamically computable features and the potential RD cost improvements related to testing more block size configurations. These methods are then tested experimentally for performance. The next presented reduction techniques partly automate the determination of features-to-complexity relationship by using Machine Learning (ML) methods to reduce encoding computational complexity. The complexity reduction techniques based on ML are separated here from other methods because ML will be used in Chapter 6 to predict quad-tree decomposition.

Shen and Yu [SY13] propose a CU partitioning early termination algorithm using Support Vector Machine (SVM) which solves CU partitioning binary classification problem. The off-line training process uses RD cost loss due to the misclassification as a weighting factor. The set of features is composed by the cbf flag (the cbf flag indicates whether or not the TB includes residual information), CU depth information of neighboring CU, gradient magnitude of current CU, and RD costs. Authors employ a wrapper method based on F-score to select the features for SVM.

Early termination methods have been presented by Correa et al. [CAdSC+14, CAVAdSC15]. Authors use classification trees obtained through Data Mining techniques. The classifiers are trained using intermediate encoding results such as RD costs or flags and used to early terminate the quad-tree partitioning, the PU mode selection and the RQT partitioning.

Zhang et al. [YKX+15] propose a fast CU depth decision using a three-level hierarchical SVM classifier. The features used for the off-line training are composed by the information of the SKIP or Merge Mode in current CU, the MV, the neighboring CU RD cost, the neighboring CU depth and QP. To consider the risk of false prediction, a joint classifier consisting of multiple SVM classifiers is proposed.

A fast CU size decision, reducing the computational complexity of the encoding process, is proposed by Zhang et al. [ZZL17]. Authors use linear SVM to early split and early terminate quad-tree partitioning process using a combination of on-line update and off-line training strategy. The two features designed for the linear SVM are composed by the depth difference between current CU and its neighboring CU and the ratio of RD costs between the current CU and its neighboring CU.

Zhu et al. [ZZK+17] proposed a CU decision method based on ML. They model the quad-tree partitioning of CTU as a multi-level classification problem which is solved by a fuzzy SVM. The sets of features are mainly composed by coding information such as flags and RD costs but differ according to classifiers. Authors incorporate a misclassification cost, composed of a classification prediction accuracy and an RD performance degradation, into a joint rate-distortion-complexity scheme to be minimized under a configurable constraint related to RD performance degradation.

Finally, Lee and Jeong [LJ17a] propose an algorithm for fast CU partitioning using statistical analysis of image complexity and coding information. The proposed CU size decision algorithm is based on Fisher's linear discriminant analysis and k-nearest neighbors (k-NN) classifier strengthened by an adaptive on-line learning phase.

All these methods are evaluated in their respective publications in terms of BD-BR and complexity reduction. These numbers are gathered in Table 3.1. The different encoding setups and configurations make the numbers difficult to compare with each other. Moreover, complexity reduction numbers are by nature optimistic, as they are obtained from very complex encoding setups. A global analysis of methods' performance is proposed at the end of this chapter.

3.2.1.2 Prediction-Based Quad-Tree Partitioning Reduction Techniques

Compared to the methods developed in previous section, the prediction-based complexity reduction methods described in this section use information that does not require an encoding. As a consequence, prediction-based methods, contrary to dynamic methods, do not require an iterative encoding process. The features used by prediction-based techniques are thus image-derived features.

Shen et al. [SZA13] propose a fast CU size decision algorithm for HEVC intra coding. They predict an interval of depth level for a given CTU based on correlations between the current CTU quad-tree decomposition and its neighboring quad-tree decomposition. Authors improve this technique by adding an early termination methods based on coding information [SLZ+13], already explained in Section 3.2.1.1.

A method using texture variance to efficiently predict the CTU quad-tree decomposition is proposed by Khan et al. [KSH13]. Authors model the Probability Density Function (PDF) of variance populations by a Rayleigh distribution to estimate some variance thresholds and determine the quad-tree partitioning.

Several works [BC15, FDZX16, WX16, KSH13, PCL16] use CTU texture complexity to predict the quad-tree partitioning. Min et al. [BC15] propose to decide whether a CU has to be split, non-split or if it is undetermined, using the global and local edges complexities

Table 3.1 – *Dynamic Quad-Tree Partitioning Reduction Technique Performance.*

Category	Reference	Complexity Reduction (%)	Degradation BD-BR (%)	Encoder
Quad-Tree Dynamic	Cassa [CNP12]	45	1.91	HM2.0
	Tan [LTLHTY12]*	40	1	HM4.0
	Ahn [AKP13]	43.2	1.58	HM9.0
	Ahn [ALK15]	49.6	1.4	HM12.0
	Huang [HZLB13]	19.9	0.46	HM8.2
	Shen [SY13]	45	1.9	HM6.0
	Liquan [SLZ+13]	42	1.49	HM2.0
	Cho [CK13]	63.5	3.5	HM6.0
	Correa [CAdSC+14]*	37	0.28	HM12.0
	Xiong [XLWM14]	43	1.9	HM9.0
	Goswami [GKJ+14]	38.03	1.68	HM10.0
	Xiong [XLM+14]	59	2.74	HM9.0
	Zhang [YKX+15]	51.45	1.98	HM12.0
	Lim [KJSS15]	44.64	1.28	HM14.0
	Tariq [TK15]*	40	0.37	HM10.0
	Kim [KP16]	53.4	0.75	HM15.0
	Zhu [ZZK+17]	55.3	2.67	HM16.5
	Lee [LJ17b]	55.4	1.01	HM16.0
	Zhang [ZZL17]*	54	0.7	HM14.0
	Lee [LJ17a]	57.9	1.42	HM16.15
	Fernandez [FDBB+18]*	25.4	1.02	HM16.2
	Fernandez [FDBBG18]	32.69	1.2	HM16.2
	Yang [YG17]*	65	1.3	HM10.0

**Techniques using multiple categories to reduce the complexity*

in four different directions (horizontal, vertical, 45° and 135° diagonals) of CU and sub-CU. An edge complexity represents a large variation of pixel values, and especially variations in a variety of directions. Feng et al. [FDZX16] use information entropy of CU and sub-CU saliency maps to predict the adequate CU size. Wang et al. [WX16] used the Otsu's method to measure the texture complexity of each CTU to skip some CU depth.

Both Shang et al. [SWFL15] and Zhao et al. [ZOS15] leverage the depth information of neighboring CU to make an early CU split decision or CU pruning decision.

Machine Learning (ML) prediction-based techniques for reducing the complexity of quad-tree partitioning : The two following reduction techniques use an ML approach to automate feature-to-complexity relationship determination when reducing encoding complexity.

Ruiz-Coll et al. [RCAFE+14, RFEA+15] present an intra CU size classifier based on data-mining for off-line classifier training. The methods select the optimal coding block size for Intra-Prediction by using a data mining classifier, based on an off-line training. The classifier is a three-node decision tree using mean and variance of CU and sub-CU as input features.

A fast CU machine learning partitioning using features that describe CU statistics and sub-CU homogeneity for screen content compression is presented by Duanmu et

al. [DMW15]. Authors employ many features such as CU luma variances, color Kurtosis of CU and gradient Kurtosis of CU as input features.

Table 3.2 shows the published performance of the prediction-based complexity reduction methods in terms of BD-BR and complexity reduction.

Table 3.2 – *Performance of the State-of-the-art Prediction-Based Quad-Tree Partitioning Reduction Techniques*

Category	Reference	Complexity Reduction (%)	Degradation BD-BR (%)	Encoder
Quad-Tree Predicted	Khan [KSH13]	44	1.7	HM7.2
	Shen [SZA13]	37.9	0.54	Kvazaar
	Ruiz [RCAFE+14]	28	0.6	HM10.0
	Ruiz [RFEA+15]	52	2	HM16.6
	Duanmu [DMW15]	36.8	3	SCM-3.0
	Min [BC15]	52.3	0.82	HM6.0
	Feng [FDZX16]	37.9	0.62	HM11.0
	Wang [WX16]*	33.32	0.81	HM12.0
	Shang [SWFL15]*	37.91	0.66	HM14.0
	Zhao [ZOS15]*	53.3	2.59	HM10.0

**Techniques using multiple categories to reduce the complexity*

3.2.2 Prediction Unit (PU) Determination Reduction Techniques

Previous sections focused on the encoding quad-tree decomposition. Another method to reduce the computational complexity of the HEVC encoder is to predict or early terminate the PU mode decision schemes. As explain in Section 2.3.2, each CB may be divided into PB, which are separately predicted with either inter or intra prediction. The next paragraphs describe the most relevant works on this category, to the best of our knowledge. The performance in terms of BD-BR and complexity reduction as well as the encoder used for the experiments of each complexity reduction technique is summarized at the end of the section in Table 3.3.

In addition to an early termination technique of the quad-tree decomposition process based on RD costs (described in Section 3.2.1.1), Tan et al. [LTLHTY12] propose to reduce the PU splitting modes tested using the correlation of the best PU mode between CU and sub-CU.

Vanne et al. [VVH14] propose an optimized PU mode decision scheme that evaluates sets of modes according to intermediate encoding information. This technique always evaluates square, intra and SKIP PU modes but proposes a conditional evaluation of the Symmetric Motion Partition (SMP) modes. They also propose a range limitations of the SMP and Asymmetric Motion Partition (AMP) sizes selected as a function of the QP value.

A method to reduce the complexity of the encoding process, based on the correlations among neighboring CU (spatial, temporal and inter-depth neighbors) is defined by Shen et al. [LZZ14]. They propose an early SKIP detection using the spatial, temporal and inter-level correlation and an early determination of the best mode according to the predicted RD-cost.

Blasi et al. [BPM+15] propose an algorithm to reduce the number of tested PU. Authors use the first frame of each GOP as a learning frame to collect the occurrence frequency for each PU mode and their RD-costs. The algorithm uses these pieces of information to

decide the minimum number of PU modes to test in descending order of their probabilities. Another method is proposed by Blasi et al. [BZIP15] that allows the quad-tree decomposition to follow a bottom-up order, i.e. from the smallest CU to the larger CU. Thanks to reverse CU visiting, the method uses MV information found in the four sub-CU to limit the number of PU modes tested at the current CU.

From the relationship between PU modes and the distribution of the distortions in the current CU, Zhang et al. [ZLL15] propose a method to skip further PU modes testing for the current depth.

Zhao et al. [ZOS15] use inter-prediction residuals to determine whether to skip unnecessary PU modes or to terminate the PU mode decision process.

A multi-objective Design Space Exploration (DSE) method is proposed by Herglotz et al. [HRGs+16] to optimize the coding PU modes evaluation of the PU mode decision process. The DSE jointly explores early skip conditions to minimize bitrate, distortion, encoding time and decoding energy.

Table 3.3 shows the published performance of the PU determination-based complexity reduction methods in terms of both BD-BR and complexity reduction.

Table 3.3 – *Performance of the State-of-the-art Prediction Unit (PU) Determination Reduction Techniques*

Category	Reference	Complexity Reduction (%)	Degradation BD-BR (%)	Encoder
PU mode	Tan [LTLHTY12]*	40	1	HM4.0
	Vanne [VVH14]	51	1.3	HM8.0
	Shen [LZZ14]	52	0.88	HM10.0
	Blasi [BPM+15]	36.7	1.33	HM??
	Blasi [BZIP15]	20.6	0.7	HM??
	Zhang [ZLL15]	48	2.9	HM16.4
	Zhao [ZOS15]*	53.3	2.59	HM10.0
	Herglotz [HRGs+16]	50	3.15	HM16.0

**Techniques using multiple categories to reduce the complexity*

3.2.3 Intra Mode Determination Reduction Techniques

Since HEVC standard enables up to 35 different intra prediction modes, there is also work in the literature that aims at decreasing the computational complexity by reducing the number of candidate intra prediction modes tested during the RDO process. Table 3.4 sums up the BD-BR, complexity reduction and encoder used for the experiments for all complexity reduction techniques.

The vast majority of works proposed to reduce the candidates modes for Rough Mode Decision (RMD) or RDO process use the direction of image edges [JMC12, CPS+13, dSAC12, LdSdSCAo13, YLL16, LLW+17, ZZL17, YG17, WX16]. Jiang et al. [JMC12] calculate the gradient directions of the current PU's samples and generate the gradient-mode histogram to select a set of candidate modes. A similar method is proposed by Chen et al. [CPS+13].

Da Silva et al. [dSAC12] calculate the predominant orientation of the edges to reduce the number of intra mode tested from 33 to 9. They improve this technique [LdSdSCAo13] by using the correlation of intra modes across depth level in the quad-tree decomposition.

Yao et al. [YLL16] use the Dominant Edge Assent (DEA), defined in [TWLY07], in directions of degree 0, 45, 90 and 135 to decrease the number of tested intra prediction modes.

Using both texture complexity and texture directions, Liu et al. [LLW+17] propose to reduce the number of candidates mode in the RMD and RDO process from 35 to 13 and 8 to 3, respectively. The texture complexity is defined as the difference between the current pixel and its surrounding pixels and the standard deviation of the current PU block. The texture directions are computed using oriented pixel value deviation in 8 directions: Vertical, Horizontal, Diagonal Down Left, Diagonal Down Right, Vertical Right, Horizontal Down, Vertical Left and Horizontal Up.

In addition to their early CU split and termination techniques detailed in Section 3.2.1.1, Zhang et al. [ZZL17] propose a gradient-based method to reduce the number of candidates mode in the RMD and RDO process. They use the average of gradients in the horizontal and vertical direction and their ratio to set a group of intra prediction modes.

In addition to the fast CU size decision, described in Section 3.2.1.1, Shang et al. [SWFL15] use the correlation between the first three modes in RMD of the current PU and the selected mode of the highest layer in the quad-tree decomposition. If one of the three modes selected in the RMD process is exactly equal to the higher layer mode in the quad-tree decomposition, they stop testing other modes. In the other case, they also stop testing other modes when the RD cost obtained for the current PU mode is smaller than the average of the left, up, left-up and right-up PU RD costs.

Yang and Grecos [YG17] propose, among other methods, a fast intra mode decision to reduce the encoding time. They observe that the curve of angular RMD cost can be classified into one of these three categories: plain, unimodal or multimodal according to the edges contained in the PU samples. From this observation, they select a set of intra prediction modes based on local minimal costs.

Part of the proposed HEVC flow of Fernandez et al. [FDBB+18] to reduce the computational complexity is composed of a fast intra prediction mode decision. As explained in Section 3.2.1.1, authors identify smooth regions on the input frame to reduce the computational complexity of these specific regions. Based on the intra modes usually selected for these smooth regions, authors propose to only evaluate DC (mode 1), planar (mode 2), Horizontal (mode 10), Vertical (mode 26) and the first MPM modes.

Tariq et al. propose in [TK15] an hybrid approach to reduce the intra coding complexity based on an early PU split termination technique (detailed in Section 3.2.1.1) and a fast intra prediction mode determination. For the fast intra prediction mode determination, authors propose to only apply DC prediction mode when the variance of the current PU samples is close to zero. They also propose to only test a set of modes in the RMD process according to the PU size.

Finally, Zhang and Zhan [HZ14] propose a complexity reduction technique composed of a micro-level and a macro-level schemes. At the macro-level, already explained in Section 3.2.1.1, authors introduce an early CU split termination based on the RD cost. At the micro-level, authors employ a Hadamard cost to prune the intra prediction mode candidates. Another similar method is proposed by Zhang et al. [ZSZG17].

Table 3.4 shows the published performance of the Intra Mode Determination-based complexity reduction methods in terms of BD-BR and complexity reduction.

Table 3.4 – *Intra Mode Determination Reduction Techniques Performance*

Category	Reference	Complexity Reduction (%)	Degradation BD-BR (%)	Encoder
Intra mode	Jiang [JMC12]	20	0.74	HM4.0
	Chen [CPS+13]	28.45	0.53	HM8.0
	Da Silva [dSAC12]	20	1.3	HM4.0
	Da Silva [LdSdSCAo13]	37.81	1.7	HM10.0
	Yao [YLL16]	36.26	1.86	HM9.1
	Liu [LLW+17]	56	1.0	HM16.0
	Zhang [ZZL17]*	54	0.7	HM14.0
	Shang [SWFL15]*	37.91	0.66	HM14.0
	Yang [YG17]	11.88	0.43	HM10.0
	Wang [WX16]*	33.32	0.81	HM12.0
	Fernandez [FDBB+18]*	25.4	1.02	HM16.2
	Tariq [TK15]*	40	0.37	HM10.0
	Zhang [HZ14] (micro-)	26	0.4	HM10.0
	Zhang [ZSZG17]*	53	1.7	HM16.2

**Techniques using multiple categories to reduce the complexity*

3.2.4 Residual Quad-Tree (RQT) Complexity Reduction Techniques

Other solutions to reduce computational complexity consist in decreasing the computational complexity needed in the process of deciding the best RQT structure. The BD-BR, complexity reduction and the encoder used for the experiments of all complexity reduction techniques are summarized in Table 3.5 at the end of this section.

Several works [CC13, SGZ13, CJ12] aim to reduce the complexity needed to decide the RQT decomposition based on the number of nonzero transformed coefficients. Chiang and Chang [CC13] proposed a fast TB zero block detection of sizes from 32×32 to 4×4. The method early decides the Transform Unit (TU) size based on Sum of Absolute Differences (SAD) values from inter prediction results.

Another method is proposed by Shi et al. [SGZ13] based on the concept of quasi-zero-blocks (QZB). They defined a criteria based on the sum of absolute values of coefficients and the number of nonzero coefficients to reduce the number of tested TU depths.

Similarly, Choi and Jang [CJ12] used the correlation between the TU block sizes and the number of nonzero coefficients to define a threshold to stop further RQT decomposition processes.

Shen et al. [SZZ+15] propose a method to reduce the complexity of the TU partitioning process. The method early terminates the TU size decision process based on Bayesian decision theory and the correlation between the variance of residual coefficients and the TU block size. They reduce the number of candidate TU block sizes for a given CU.

Similarly, Wu et al. [WWW16] accelerate TU depth decision using Bayesian model. They employ the correlations between a TU and its adjacent TU, which are used as referential features for TU depth decision.

The works of Yang and Grecos [YG17] conclude this section of computational complexity reduction techniques in HEVC with a technique combining complexity reduction techniques extracted from each category: a fast CU skip decision, a fast CU early termination, a fast PU mode decision and a fast TU size decision. They decide whether TU

partitioning should be avoided or not comparing TU scores with thresholds. TU scores are computed as the SAD of quantized transformed coefficients and weighting factor.

Table 3.5 shows the published performance of the Residual Quad-Tree (RQT) complexity reduction methods in terms of BD-BR and complexity reduction.

Table 3.5 – *Performance of Residual Quad-Tree (RQT) Complexity Reduction Techniques*

Category	Reference	Complexity Reduction (%)	Degradation BD-BR (%)	Encoder
Other	Chiang [CC13]	-	-	HM4.0
	Shi [SGZ13]	22.82	-	HM10.0
	Choi [CJ12]	-	-	HM3.0
	Shen [SZZ+15]	13	0.5	HM10.0
	Wu [WWW16]	20.36	0.78	HM16.0
	Yang [YG17]	8.65	0.17	HM10.0

3.2.5 Overview of Computational Complexity Reduction Techniques

The 4 previous Sections (3.2.1, 3.2.2, 3.2.3 and 3.2.4) provide an overview of computational complexity reduction techniques proposed for HEVC. However, it is important to notice that most of the approaches cannot be directly compared with each other w.r.t the achieved complexity reductions and encoding degradations (BD-BR or BD-PSNR) due to the encoding conditions. Indeed, in addition to missing information, different software encoder or encoder versions, encoding configurations, sets of test sequences, set-ups and experimental conditions (e.g. HW architecture, processor, caches, memory, etc.) are used. Nevertheless, Tables 3.1, 3.2, 3.3, 3.4 and 3.5 summarize available information to have an overview of the complexity reduction strategies for HEVC and reachable gains.

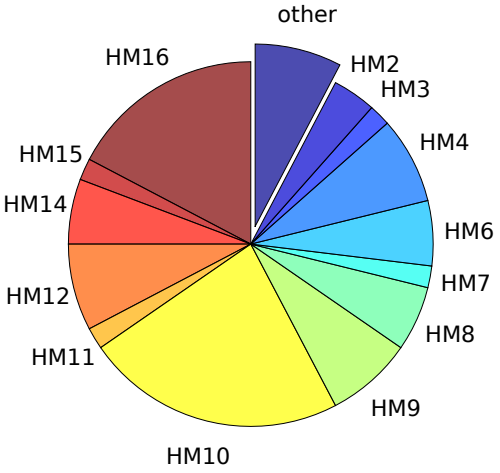


Figure 3.2 – *Percentage of each encoder version extracted from the state-of-the-art.*

Figure 3.2 represents the percentage of each encoder version used in literature. As explained in the previous paragraph, more than 13 different versions of HM are used. The two most used versions are HM10 and HM16 corresponding respectively to the version

used for the standardization (in 2013) and the latest version at the time of writing this document.

Figure 3.3 shows the percentage of complexity reduction techniques for each category: quad-tree partitioning (QT) representing 73%, PU determination (PU) representing 15%, Intra Mode prediction (IM) representing 29% and RQT prediction (RQT) representing 15%.

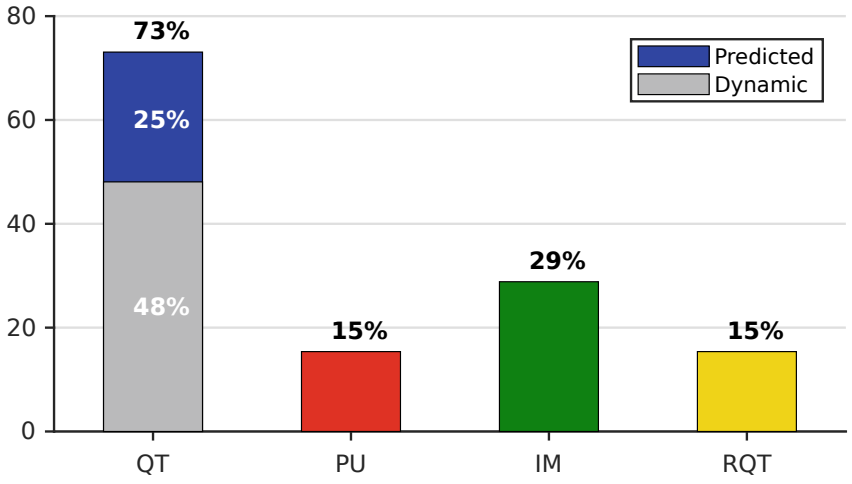


Figure 3.3 – *Percentage of complexity reduction technique for each category.*

As shown in Figure 3.3, the most common way to reduce the computational complexity of the HEVC process is to reduce the complexity of the quad-tree partitioning process. Indeed, the CU, which can be of size from 64×64 to 8×8, is the basic encoding unit and the quad-tree determination is the most complexity-impacting part of the HEVC encoding process, as Chapter 4 will show in details. Within the quad-tree determination category of techniques, 48% of techniques are *Dynamic* (i.e. using encoder-related information) and 25% are *Predicted* (i.e. applicable before the RDO process). This difference can be explained by the direct relevance of the encoder-related information to predict the quad-tree partitioning, while image-based features are only indirectly related to the optimal block partitioning.

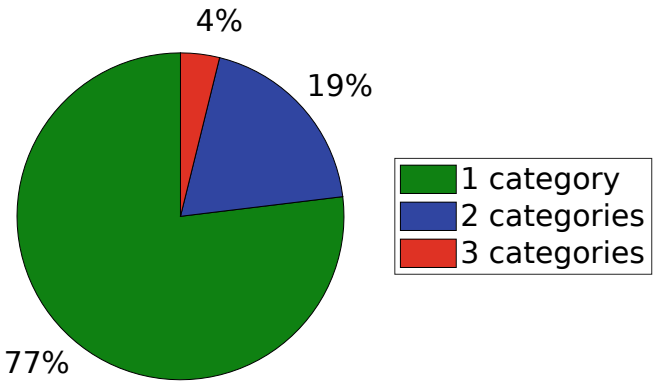


Figure 3.4 – *Percentage of techniques using 1, 2 or 3 categories at the same time.*

The sum of percentages of Figure 3.3 exceeds 100% due to the fact that some contributions do not limit to one category and try to reduce the computational complexity of the HEVC encoding process by tackling several categories at the same time. As shown in Figure 3.4, 77% of state-of-the-art techniques work on only one category of complexity reduction. 19% of techniques use 2 categories in conjunction, including most of the time the quad-tree determination category. Only two works combine 3 categories in the same complexity reduction technique.

Figures 3.5 and 3.6 show the distribution of the computational complexity reduction and the encoding degradation in terms of BD-BR of the techniques summarized in this section, respectively.

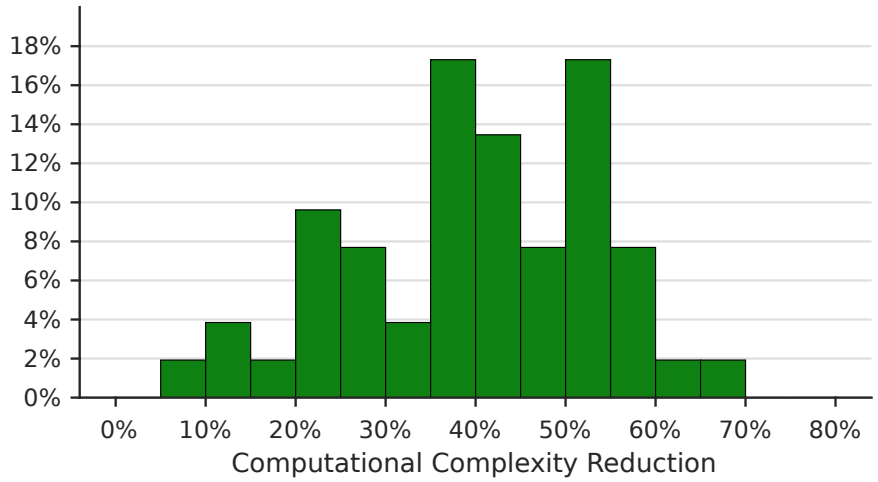


Figure 3.5 – *Distribution of the computational complexity reduction of the state-of-the-art.*

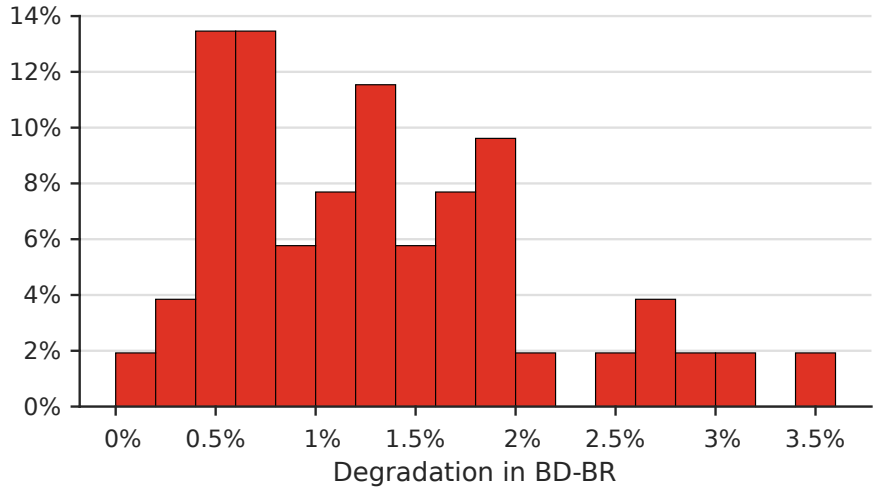


Figure 3.6 – *Distribution of the encoding degradation in terms of BD-BR of the state-of-the-art.*

As shown in Figure 3.5, all computational complexity reductions are located between 5% and 70%. Half of the techniques (55%) reduce the complexity from 35% to 55%. Figure 3.6 shows that all complexity reduction techniques have a BD-BR increase of less than 3.5%. Half of the techniques (51%) have an increased BD-BR included between 0.4% and 1.4%.

3.3 Computational Complexity Control in HEVC

The reference HEVC software HM [noa] includes every tool defined in the JCT-VC work since its inception. The HM software is not intended to be a suitable option for HEVC encoder implementation. In other words, HM software is not intended to be real-time; it disregards computational costs or energetic constraints to concentrate on encoding efficiency. Real-time HEVC encoding implies a scenario in which available resources (computational power and/or energy) of the platform are limited.

All the techniques of computational complexity reduction presented previously in this section are not sufficient to dynamically control the encoding complexity when running the encoder on a specific target. Most of the techniques do not achieve the same complexity reduction from one video sequence to another. Specific algorithms of complexity control have to be designed. Several complexity control algorithms for the HEVC encoding process have been published in the literature in the last years and are detailed hereafter.

A first approach to control the computational complexity of the HEVC encoding process is to dynamically adapt the encoding parameters according to a complexity target value [PMP+16, GSK+13, GZS+17].

Penny et al. [PMP+16] propose a dynamic Pareto-based energy consumption controller for HEVC encoders. Authors start by analyzing the energy consumption of the encoding process according to a combination of the following HEVC parameters: maximum partition depth, Search Range (SR), Asymmetric Motion Partition (AMP), Hadamard Motion Estimation (ME), maximum TU Depth and Fractional Motion Estimation (FME). They extracted 31 configurations from a Pareto frontier analysis that are used to control the energy consumption of the encoding process.

Similarly, an Adaptive Workload Management Scheme (AWMS) proposed by Grellert et al. [GSK+13] dynamically configures different parameters of the HEVC encoder to manage the complexity under a specific target. The parameters used are composed by: the maximum CU depth, Search Range (SR), Hadamard ME and AMP. They realize the feedback loop control by a Proportional–Integral–Derivative (PID) controller. They improved the previous work in a new method, called Complexity Control Scheme (CCS) [GZS+17]. They added the number of reference frames and the maximum of TU depth as input parameters of the controller.

Zhao et al. [ZWK13] proposed to control the computational complexity for a specific target by adjusting the number of evaluated PU splitting modes. Based on a MV correlations analysis, they used a mode mapping method to select PU splitting modes. The computational complexity allocation is based on linear programming to select the number of tested PU modes.

A complexity control scheme is presented by Ukhanova et al. [UMF13] based on game-theory approach. According to a given budget, the method defines a maximum coding tree depth for each encoded frame to adjust the complexity. The first 4 frames of the video sequence are encoded without any constraints and then a game-theory based approach is used to select the maximum coding tree depth for the other frames.

Deng et al. [DXJ+16] propose a Subjective-driven Complexity Control (SCC) approach based on visual attention model to control the encoding complexity with a minimal visual distortion. They study two relationships: the relationship between the maximum depth and encoding complexity and the relationship between the maximum depth and visual distortion. The method varies the maximal depth of each CTU to achieve target encoding complexity.

A complexity control algorithm using an early termination condition of the quad-tree partitioning is proposed by Jimenez et al. [JMMEDdM16]. Taking into account the target of complexity reduction, the early termination condition determines for every CU size whether sub-CU size should be tested. The threshold used in the early termination condition are estimated on the fly to adapt to video content properties, encoding configuration and complexity target.

Correa et al. [CAAC16] present a set of methods to control the computational complexity for the HEVC encoding process. All the methods limit the quad-tree exploration to adjust encoding complexity. The first method introduces the Fixed Depth Complexity Scaling (FDCS) [CAdSC+12] that limits the computational complexity of the system by decreasing or increasing the maximum coding tree depth by frame. At the end of the current frame encoding, if the complexity of this current frame is lower (resp. higher) than the target, the method increases (resp. decreases) by one the maximum coding tree depth for the next frame.

The Variable Depth Complexity Scaling (VDCS) method, proposed by Correa et al. [CAdSC+12, CAAC11], uses the depth used in previously encoded frames to constrain the maximum coding tree depth. They define two types of frame: the unconstrained frames in which all possible coding tree structures are tested and the constrained frames in which the maximum coding tree depth is set as the depth obtained at the last unconstrained frame. According to the target and current complexity, the method adjusts the number of constrained frames between two unconstrained frames. In the Motion-Compensated Tree Depth Limitation (MCTDL) method [CAA+12, CAA+13], authors improve the previous method by taking into account the video content motion between frames. The method stores a weighted average of MV for each CTU of unconstrained frames and uses them to adjust the maximum coding tree depth of constrained frames. They then propose to use, in addition to the temporal depth information of MCTDL, the spatial neighboring CTU depth information in the Coding Tree Depth Estimation (CTDE) method [CAdSC+13a, CAAC13]. For each constrained frame, the maximum allowed coding tree depth is defined to be the largest of the maximum coding tree depths used in the Left, Top, Top-Left neighboring CTU of the current frame plus the co-localized and motion-compensated CTU from the previous frame. Finally, they analyze the difference in complexity reduction and encoding degradation between enabling and disabling PU size smaller than $2N \times 2N$ according to maximum coding tree depth values. The results show that it is more profitable in terms of complexity reduction and encoding ratio to disable PU size smaller than $2N \times 2N$ in comparison to reducing the maximum coding tree depth by one level. They proposed a new method [CAAdSC16, CAdSC+13b], called Constrained Coding Units and Prediction Units (CCUPU), in which in addition to the maximum coding tree depth constraint of the CTDE method, they add a PU size constraint for each CTU of each constrained frame.

Last but not least, Correa et al. [CAAdSC15, CAdSCA15, CAAC16] gather the last previous method and the early termination techniques [CAdSC+14, CAVAdSC15] (detailed in Section 3.2.1.1) to build a control system able to keep encoding time under a specific target value. This system is split into three parts: the coarse-, medium- and fine- granularity time control. The coarse-granularity computes the ratio between the target time and the current encoding time to configure the two other parts. The medium-granularity has to set the configuration of the following parameters: the Search Range (SR) and Bi-Prediction Refinement (BPR) values plus the Hadamard ME, coding tree early termination, PU early termination and RQT early termination activation. After testing all the 240 configurations, the parameter configurations used in the system are extracted from the Pareto frontier.

Finally, the fine-granularity applies the previously evoked CCUPU method. The results shown that the system limits the encoding time per GOP with an average deviation from the target of 2.9%.

Table 3.6 summarizes the performance of the presented complexity controllers, listing their range of complexity reduction and their range of BD-BR degradation. The currently most performing solution is certainly the one from [CAAdSC15, CAdSCA15]. However, it is still evaluated on the off-line, non-optimized HM encoder.

Table 3.6 – *Computational Complexity Control Performance*

Category	Reference	Complexity Reduction (%)	Degradation BD-BR (%)	Encoder
Gray Box Parameter Tuning	Penny [PMP+16]	-	-	HM16.0
	Grellert - AWMS [GSK+13]	31-43	2.6-3.5	HM8.2
	Grellert - CCS [GZS+17]	20.1-61.45	0.45-4.9	HM10.0
White Box PU Mode Tuning	Zhao [ZWK13]	24.5-54	0.7-5.9	HM8.0
White Box CU Depth Tuning Level	Ukhanova [UMF13]	≈10-40	-	HM7.0
	Deng [DXJ+16]	≈20-80	0.84-17.5	HM14.0
	Jimenez [JMMEDdM16]	9.79-42.23	0.23-6.83	HM13.0
	Correa - FDCS [CAdSC+12]	11-37	1.24-12.02	HM4.0
	Correa - VDCS [CAdSC+12, CAAC11]	10-38	0.58-6.29	HM4.0
	Correa - MCTDL [CAA+12, CAA+13]	10-39	0.54-5.42	HM4.0
	Correa - CTDE [CAdSC+13a, CAAC13]	11-37	1.05-4.74	HM8.2
	Correa* - CCUPU [CAAdSC16, CAdSC+13b]	7-84	0.03-39.47	HM9.2
	Correa* [CAAdSC15, CAdSCA15]	≈10-90	0.16-9.84	HM13.0

**Techniques using multiple categories to reduce the complexity*

3.4 Conclusion

As shown in the previous sections, active research is conducted around computational complexity for video encoding process, especially focusing on HEVC in the recent years. Nevertheless, the substantial increase of networked devices and their fast evolution in terms of computational power and display capabilities on one side, and the development of more complex video coding algorithm on the other side, create novel research challenges.

The Internet of Things (IoT) will include more than 27 billion connected devices in 2021 [CIS16] across plethora of heterogeneous architectures and embedded constraints such as computational power, bounded energy density of batteries, power consumption, etc. Among Internet of Things (IoT) devices, video-based services will raise, together with the technological capacity to manage them. At the same time, Internet video traffic will represent 80% of all Internet traffic in 2021 [CIS16]. These estimates, showing the im-

portance that embedded video will take in the next years, represent only one part of the challenge.

Evolution of display and camera technologies make possible ever higher resolutions — up to 8K at the time of writing this document— and ever higher frame rates —up to 120Hz at the moment—. Video sequences which should be recorded, transmitted, stored and played are thus individually requiring an increasing amount of data. Video processing including compression, applied to these big data, will have an ever-greater environmental impact in terms of power consumption.

A large share of systems are likely to integrate HEVC in the long run and will require to be energy efficient, or even energy aware. As a consequence, energy consumption represents a serious challenge for embedded HEVC encoders. For both hardware and software codecs, a solution to reduce the energy consumption is to decrease the computational complexity while limiting compression quality losses.

This chapter has presented a study on the main contributions to computational complexity reduction and control for HEVC. Section 3.2 summarized computational complexity reduction techniques, classifying them into 4 categories according to the impacted part of the HEVC encoding process. Section 3.3 focused on HEVC computational complexity control methods which is an emerging field. Complexity control of the HEVC encoding process under energy constraint is the main motivation for the research work presented in the next chapters.

CHAPTER 4

Energy Reduction Context and Opportunities in HEVC

4.1 Introduction

Optimizing the energy consumption of embedded systems is a primary concern when designing autonomous devices. The emerging domain of the IoT requires ultra low-power processing and communicating systems. The main goal of this thesis is to optimize the energy consumption of HEVC encoding process. More specifically, this work aims at controlling the energy or power consumption of the encoding process.

When bit accurate energy and power management techniques have been implemented, further steps in energy reduction can only be achieved by degrading the quality of the application output. In HEVC encoding, degrading the quality results in either an increase of the bit-rate for a given video quality, or a decrease of the video quality for a given bit-rate. Exploring the trade-offs between energy and quality is at the heart of the approximate computing paradigm. Approximate computing relies on the ability of many applications and systems to tolerate some loss of quality in the computed result. The aim is to consider energy reduction of HEVC encoder through the prism of approximate computing paradigm. This paradigm has been emerging for a decade and can produce significant improvements from technological level to system level. In our work, approximate computing at algorithmic level is considered. At this level, approximate computing aims at reducing the computational complexity by approximating or skipping part of the processing.

This chapter makes the connection between the approximate computing concepts and the applicative HEVC encoding process in order to control its energy consumption. A methodology is proposed in Section 4.2 to exploit the computation-skipping approximate computing concept. The methodology, named Smart Search Space Reduction (SSSR), explores at design time the Pareto relationship between computational complexity and application quality. Section 4.3 presents an analysis of the energy reduction opportunities offered by an HEVC encoder. The energy reduction search space is defined, and the impact on energy consumption of encoding tools at various levels of granularity is measured.

4.2 Energy Reduction with Approximate Computing

The main limitation of recent embedded systems, particularly in terms of computational performance, comes from the bounded energy density of batteries. The power consumption of processors is generally split into two parts: the dynamic part coming from the switching

activity of transistors and the static part coming from transistor leakage current. Jejurikar et al. [JPG04] defined the power model where the total power consumption P_{tot} is given by equation 4.1:

$$P_{tot} = P_{stat} + P_{dyn} \tag{4.1}$$

with

$$P_{stat} = C_{eff} \cdot f_{proc} \cdot V^2 \tag{4.2}$$

and

$$P_{dyn} = V \cdot I_{leak} \tag{4.3}$$

where V is the supply voltage matching with the processing frequency f_{proc}, I_{leak} is the leakage current, and C_{eff} is the circuit effective capacitance.

To reduce the energy consumption of digital systems, new techniques from circuit to system levels have been proposed in the last two decades. Technologies such as FinFet [SFB+08] and Fully Depleted Silicon on Insulator (FD-SOI) [NMD+09] continue the trend of shrinking down transistors and reducing their leakage current, providing more energy efficient hardware. At system level, methods are developed to adapt the instantaneous processing capacity to the requirements of the running applications. Two power management techniques are mainly used to minimize the energy consumption of modern Systems-on-Chips, namely DPM and DVFS. Dynamic Power Management (DPM) [BDM98] consists in combining clock gating and power gating to turn a processing core into a low-power state when it is idle. Dynamic Voltage and Frequency Scaling (DVFS) [MDVPO90] minimizes the consumed dynamic power by reducing both hardware clock frequency and supply voltage until real-time constraints are met.

The complexity of HEVC encoding process is due to the solving of several discrete optimization problems like quad-tree decomposition, PU mode selection, intra mode selection, RQT decomposition, motion estimation. The aim is to find the solution minimizing the RD-cost as expressed in Eq. 2.8. Thus, in this section, the focus is put on algorithms that use discrete optimization techniques, exploring a parameter search space, to minimize a cost function. The optimization process contributes in creating costly algorithms and most existing optimization methods tend to raise the data dependency of the considered algorithm complexity. These algorithms are referred to as Minimization based on Search Space Exploration (MSSE) algorithms in the rest of the document. To decrease the energy consumption of MSSE algorithms, a challenge is to reduce the search space.

4.2.1 Approximate Computing Paradigm

4.2.1.1 Approximate Computing Dimensions

The approximate computing paradigm exploits the error resilience of numerous data processing applications (signal processing, image processing, etc.) to explore the energy-quality trade-off. Approximate computing provides three degrees of freedom [RSNP12] to act on : data, hardware, and computation (or algorithm) levels, as illustrated in Figure 4.1.

Approximation on the processed data dimension consists in reducing the quality of the application in a controlled way by using: less up-to-date data (temporal decimation), less input data (spatial decimation), less accurate data (word-length optimization) or even corrupted data. For the HEVC encoding process, the video resolution and the frame rate can be adapted for spatial and temporal decimation.

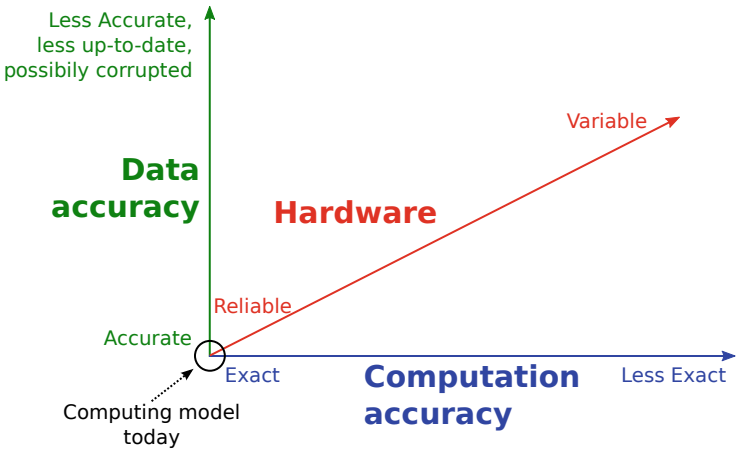


Figure 4.1 – *The dimensions of approximate computing [RSNP12]*

On the hardware dimension of approximate computing techniques, the exactness of the computation support can be relaxed with different techniques. At the technological level, a voltage over-scaling technique can be used to reduce energy consumption. To switch on an N-type Metal Oxide Semiconductor (NMOS) (resp. P-type Metal Oxide Semiconductor (PMOS)) transistor reliably, its gate-to-source (resp. source-to-gate) voltage has to be higher than the threshold voltage of the transistor. Working at near threshold regime or even at sub threshold regime achieves major energy consumption reductions but leads to computation errors due to an increase in circuit delay. Nevertheless, these timing errors may be tolerable if their probability of occurrence is sufficiently low. Techniques have been proposed to compensate these errors [MCRR11] or to maintain these errors within reasonable bounds [KP11]. At a logic level, a functionality can be approximated by simplifying logic. An approximate hardware module is implemented with a truth table that slightly differs from the exact hardware module.

The third dimension related to algorithm level is investigated in this document and analyzed in the next section.

4.2.1.2 Approximate Computing at Algorithm Level

A few approximate computing techniques have been proposed to decrease the computational complexity with the objective to reduce energy consumption. On the one hand, the inherently error resilient blocks can be modified or skipped, computations can be stopped early to save power, or memory accesses can be ignored. On the other hand, some intricate computations inside these blocks can be approximated.

The concept of incremental refinement, introduced in [LNC96], consists in reducing the number of iterations of an iterative processing. By skipping part of the processing, the energy can be reduced. Chippa et al. [CMR+14] use early termination and convergence-based pruning for k-means clustering. For early termination, the iterative process is stopped before the full convergence, when the number of points changing clusters in successive iterations falls below a threshold. For convergence-based pruning strategy, the points, that have not changed their clusters for a predefined number of iterations are considered to have converged, and they are skipped from further computations. The concept of early termination is also widely used to stop the search process in discrete optimization problem

when the chance to find a better solution is very small. As shown in part 3.2.1.1, this concept is used to reduce the complexity of Quad-Tree Partitioning.

Loop perforation is proposed in [SDMHR11]. The loops that can tolerate approximation errors are identified in a first step, and then transformed to execute a certain number of iterations. In [SDMHR11] a Pareto front enables the application developer to select the best perforation strategy depending on his requirements and constraints.

Another solution to reduce computational complexity consists of identifying blocks that can be skipped with a minimum impact on the output [NMP16]. Selected blocks can be permanently or periodically skipped depending on the quality constraints of the application. Once a given trade-off between the required precision and the energy consumption has been reached, a parameter can be used to activate or not the selected blocks. This approximate computing technique can be applied to a domain conservation process [NMP16]. This process is used to enhance a signal property. For example, in an HEVC decoder, the different filters have been approximated to decrease its energy consumption.

4.2.1.3 Approximate Computing for MSSE Algorithms

HEVC encoding process is complex in terms of computation because it integrates several tools carrying-out discrete optimization like quad-tree decomposition, PU mode selection, intra mode selection, RQT decomposition, motion estimation. These tools aim at minimizing a cost function in order to select the solution leading to the minimal cost. The test of the numerous candidate solutions, for each tool, leads to high computational complexity.

Thus, in this section, we focus on processes achieving discrete optimization based on Minimization based on Search Space Exploration (MSSE) algorithms for which the main purpose is to minimize a cost function by exploring a search space. As illustrated in Figure 4.2, the studied MSSE algorithms consists in testing different candidate solutions S_i ($i \in [1, n]$) to select the optimal one that minimizes the cost function.

Numerous applications in the image and signal processing domain integrate MSSE algorithms. In telecommunications, channel decoding and MIMO decoding such as sphere decoding use MSSE algorithms. For example, in [HOPP13], authors use the properties of an MSSE algorithm to select the best transmission configuration including the modulation spectral efficiency and Error Code Correction code rate that maximizes the quality of a wireless received scalable video over MIMO channels. MSSE algorithms are also used in image processing applications for classification operations such as in the Nearest Neighbor classifier.

An exhaustive search, presented in Figure 4.2, is a straightforward approach to process MSSE but it may require much computation, depending on the search space size. A challenge for an MSSE algorithm implementation is to reduce the search space while minimizing the impact on the selected solution $\tilde{S}_{opt}$ compared to the optimal solution under the full search space S_{opt}. Ideally, the optimal solution must be contained in the reduced search space ($\tilde{S}_{opt} = S_{opt}$). To the best of our knowledge, no approach has been proposed in literature that focuses on approximate computing based on MSSE algorithms.

4.2.2 Approximate Computing Methodology for MSSE algorithm

4.2.2.1 Search Space Reduction Techniques

For MSSE algorithms, the search space reduction techniques can be classified into two categories: branch-and-bound and SSSR. The branch-and-bound techniques initially consider

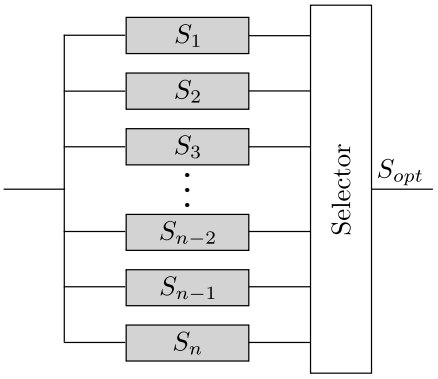


Figure 4.2 – *MSSE algorithm with exhaustive search. Each branch represents a full solution computation.*

the entire solution search space. Then, based on intermediate results, the branch-and-bound technique automatically prunes the search space by excluding the least likely solutions. Hence, parts of the search space which can not lead to the optimal solution are removed. The discrete optimization problem can be formulated with a tree representing the different solutions S_i. The branch-and-bound technique can be used to reduce the search space. This technique exploits the concept of early termination. In HEVC, dynamic quad-tree partitioning complexity reduction techniques detailed in Section 3.2.1.1 belong to the branch-and-bound technique category. The exploration of the branch is stopped if the minimal cost which can be obtained for the exploration of this branch is higher than the best cost which has already been obtained during the exploration of the previous branches, as illustrated in Figure 4.3.

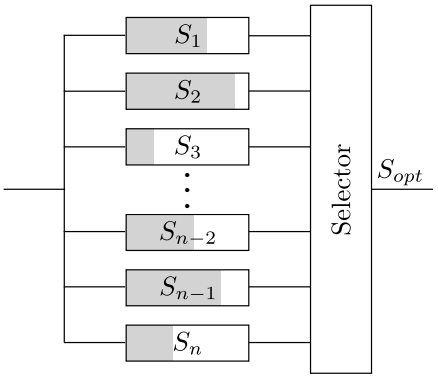


Figure 4.3 – *MSSE algorithm with branch & bound technique. Early termination stop the exploration of branches which can not lead to the best solution.*

The efficiency of this technique is based on the availability of a heuristic that quickly finds a good solution. This technique can guarantee that the optimal solution S_{opt} is found, even though the search space is pruned. However, the drawback of branch-and-bound techniques is the unpredictability of their execution time [HOPP13] which is problem dependent. Thus, the gain in terms of energy can not be predicted or adjusted with parameters controlling the approximation.

The Smart Search Space Reduction (SSSR) techniques first select a subset of initial solutions in the search space, based on a coarse estimation of their cost, called prediction. Then a refinement of selected initial solutions is computed to find the best solution among them, as depicted in Figure 4.4.

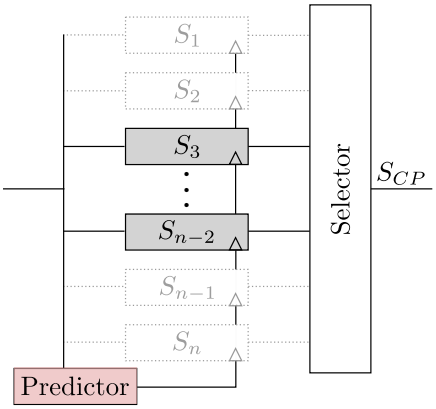


Figure 4.4 – *MSSE algorithm with SSSR technique. Coarse estimation is carried-out and a refinement is applied to the best candidates.*

An efficient coarse solution predictor providing a good estimation with low computational complexity can improve the quality of such solutions. In HEVC, predicted quad-tree partitioning complexity reduction techniques detail in Section 3.2.1.2 belong to the SSSR technique category. With SSSR techniques applied to energy minimization, the gain in terms of energy can be controlled by adjusting the search space around the coarse estimation. Nevertheless, optimality can not be ensured which depends on the accuracy of the prediction step. This SSSR second category of search space reduction technique is investigated in this section.

4.2.2.2 The Smart Search Space Reduction (SSSR) Method

As explained previously, the branch-and-bound techniques aim to converge as fast as possible towards the best solution in a search space, with the objective to reduce the search cost. However, contrary to the SSSR design method, they do not offer features to control the search cost. Contrary to branch-and-bound techniques, the new SSSR design methodology detailed in this section is control-oriented and lets the system, at run time, choose a searching method in a set of solutions ranging from a minimal search to a full search.

The new SSSR design method for applications based on MSSE algorithms is depicted in Figure 4.5 and formalized by Algorithm 4.1. It consists of the 3 first steps performed at design time to manage the approximation at the run time, as detailed below. The goal of the first step is to identify the MSSE in the application. Then, these MSSE are classified according to their potential cost gain. The third step consists in designing a predictor to manage the approximation and reduce the cost of the MSSE according to an acceptable quality degradation. In the fourth step, a run-time management of the approximation is set-up.

4.2.2.2.1 Step 1: MSSE Identification

In the first step of the Smart Search Space Reduction (SSSR) method, the developer must manually identify the MSSE algorithms in the application $\mathcal{A}$. MSSE can be independent

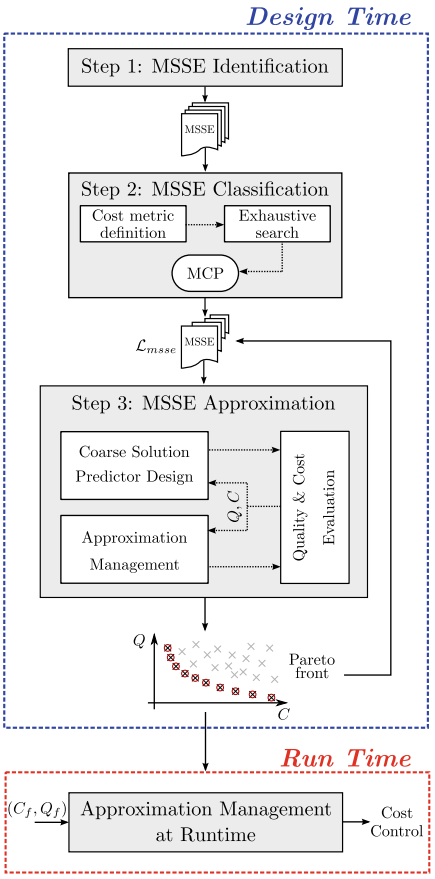


Figure 4.5 – *The Smart Search Space Reduction (SSSR) method*

or nested. The nesting of MSSE algorithms in the application has no consequence on the methodology, each MSSE being studied independently.

4.2.2.2.2 Step 2: MSSE Classification

The second step aims at ordering the identified MSSE according to the cost reduction opportunity they offer. The cost metric, defined by the developer, can be energy, complexity, execution time, or a combination of several of these metrics. It is an implementation metric to optimize. The nature of quality metric, also defined by the developer, depends on the application domain and purpose. In the case of the HEVC encoding process, the quality metric is the BD-BR.

Depending on cost and quality metrics, different Pareto curves will be obtained, offering trade-off between cost and quality. Thus, the reduction of the search space will not lead to the same application configurations. Let the Minimal Cost Point (MCP), associated to an MSSE algorithm, be the theoretical lower bound of the implementation cost (e.g. the minimal energy) that enables the optimal quality (e.g. the optimal BD-BR). Let $C_{M_{CP}}$ be the cost that can be obtained if the optimal solution is perfectly predicted. In this case of optimal prediction, only one solution is computed, the optimal one. It leads to the minimum cost for an optimal quality.

The MCP is obtained by a two-pass approach. The first pass is an exhaustive search identifying the optimal solution in terms of quality over a representative training data set $\mathcal{D}$. This exhaustive search has two objectives: obtaining the worst case cost by exhaustive

Algorithm 4.1: The Smart Search Space Reduction (SSSR) method

Data: An application $\mathcal{A}$ to optimize and a training data set $\mathcal{D}$
Result: A Pareto point cloud over NFP cost C and application quality Q, with points tagged by approximation parameters.

1 Identify the MSSE in $\mathcal{A}$;
2 Define the Cost metric C;
3 **for** *all* MSSE **do**
4 Apply an exhaustive search on the MSSE using $\mathcal{D}$;
5 Get the optimal solutions of the MSSE;
6 Force the MSSE to only compute the optimal solution and measure the minimum cost;
7 Derive the MCP of the MSSE
8 Order the MSSE according to their Minimal Cost Point (MCP);
9 **while** *the approximation result is not satisfactory for the specific use case* **do**
10 Select the next MSSE in ascending MCP order;
11 Design a coarse solution predictor for this MSSE;
12 Define the approximation parameters Γ;
13 Define a set i of Γ values;
14 Evaluate the resulting cost and quality (C_i, Q_i) on $\mathcal{D}$;
15 Extract the set f of Γ configurations on the Pareto front;
16 **if** (C_f, Q_f) *is not satisfactory and* i *can be refined* **then**
17 Redefine a set i of Γ values;
18 Go back to line 14;

search, and the optimal solution over the training set $\mathcal{D}$ that will be used in the second pass. The exhaustive search reveals at design time a solution close to the optimal solution (under the hypothesis that the training data set $\mathcal{D}$ is representative of the run time processed data) that will be approached at run time by the Pareto prediction. In the second pass, the MSSE algorithm only executes the "optimal over training data" solution identified in the previous pass. The C_{MCP} is the cost of this second pass. The cost reduction opportunity is then defined by the C_{MCP} from the cost of the exhaustive search.

The output of this MSSE Classification step is $\mathcal{L}_{msse}$, the ordered list of MSSE according to the cost reduction opportunities.

4.2.2.2.3 Step 3: MSSE Approximation

The approximation process is carried out for each MSSE algorithm of the ordered list $\mathcal{L}_{msse}$ until sufficient cost reduction is obtained. This process starts with the MSSE algorithms having the highest opportunity in terms of cost reduction and progressively the approximations associated to each MSSE algorithm can be combined.

Substep 3.1: Coarse Solution Predictor Design In this step, the developer has to design and develop an efficient coarse solution predictor. The challenge of this step is to define a predictor model with a moderate computation complexity overhead and able to provide a solution as close as possible to the optimal solution. In the context of approximate computing, coarse solution predictors with a limited computation complexity are preferred

to precise and expensive cost reduction techniques in order to reduce significantly the implementation cost. Complex coarse solution predictors can annihilate the cost reduction obtained by the search space reduction. Moreover, a complex predictor solution requires both long design and development times. Let C_{CP} and Q_{CP} be respectively the normalized computation cost of Coarse Predictor (CP) and the quality degradation associated to this configuration. The difference between $C_{M_{CP}}$ and C_{CP} is likely to be mainly due to the overhead of the Coarse Predictor computation.

Substep 3.2: Approximation Management This sub-step aims at expanding the search space around the solution obtained with the coarse solution predictor. Expanding the search space improves the quality because it increases the probability to include the optimal solution in the set of tested solutions. Nevertheless, this expansion is performed at the expense of some complexity cost. Firstly, the different approximation parameters Γ involved in exploring the cost-quality trade-off are enumerated and a set of parameter values i is defined. These approximation parameters Γ serve the coarse solution predictor and define how large the search space is from the coarse estimation. Secondly, a fast quality evaluation approach is used to extract the configurations that are close to the multi-objective Pareto front. This fast approach is used to quickly remove the configurations (C_f, Q_f) in all the tested configurations (C_i, Q_i) which are far from the Pareto front. The quality is evaluated on a subset of the approximation parameters Γ values to select the values leading to configurations (C_f, Q_f) close to the Pareto front. The fast quality evaluation is carried-out by limiting the amount of processed input data. This leads to a statistical estimation with a moderate accuracy, yet sufficient to detect configurations (C_f, Q_f) close to the Pareto front. Thirdly, the most interesting configurations of Γ values are refined by testing more approximation parameters values on the complete input data set.

At the end of this step, the developer can choose to go back at the beginning of the Step 3 to combine another MSSE in the approximation. Three main reasons can bring the developer to work on multiple MSSE in an energy oriented use case:

- if the developer has energy constraints not achievable with one MSSE alone,
- if the developer wants to have a fine grain management of the energy to match an energy budget,
- if the developer wants to achieve the best possible quality for a given energy reduction.

4.2.2.2.4 Run-time Approximation Management

The aim of the last step is to design and implement the run-time management of the approximation. This controller defines the strategy to control the approximation parameters at run-time. It determines the best parameter value configuration according to the requirements in terms of quality and cost. These parameter values are determined from the Pareto front obtained in the previous step. To measure the global gain in terms of cost (e.g. energy) the quality/cost trade-off is evaluated on a real scenario.

4.2.3 Conclusion

To conclude, by using the SSSR methodology, a designer follows a straightforward flow that exploits the computation-skip approximate computing concept to explore the trade-offs between algorithm degradation (quality metric) and computational cost reduction. The identification of the MSSE parts of the algorithm is the only prerequisite to apply

the methodology. SSSR requires manual design steps that complicate the design process. However, they are the price to pay for obtaining large energy gains.

The next section presents an analysis of the energy reduction opportunities offered by an HEVC encoder. The impact on energy consumption of encoding tools at various levels of granularity is measured in order to identify and evaluate the MSSE of HEVC Intra encoding process.

4.3 Energy Reduction Opportunities in an HEVC Real-Time Encoder

In this section we analyze energy reduction opportunities by considering different levels of granularity from global video parameters to prediction unit determination. The purpose is to identify and evaluate the MSSE of HEVC Intra encoding process. In our use case, the energy consumption and BD-BR are respectively used as the cost and quality metric, as defined in the SSSR methodology presented in Section 4.2.2.2.

The aim of this section is to study energy consumption and RD degradation in a real-time HEVC encoder when constraining the encoding at different levels of granularity. The specific case of MCP, named Minimal Energy Point (MEP), is introduced representing the boundaries of energy consumption for a given configuration.

4.3.1 Experimental Setup

The following sections give the experimental setup used to conduct the experiments and compute the results.

4.3.1.1 Video Sequences Setup

A set of 18 sequences defined by Joint Collaborative Team on Video Coding (JCT-VC) in the Common Test Conditions (CTC) [Bos13] is selected for this analysis, as summarized in Table 4.1. Two sequences from class A, 5 sequences from class B, 4 sequences from class C, 4 sequences from class D and 3 sequences from class E. Each class corresponds to a specific resolution. The sequences of class E are considered to represent conversational applications. This set of sequences is composed by video sequences that strongly differ from one another in terms of frame rate, motion, texture and spatial resolution as illustrated in Appendix B.

BD-BR and BD-PSNR [WSBL03], detailed in Section 2.2.3, are used to measure the compression efficiency difference between two encoding configurations. The BD-BR reports the average bit rate difference in percent for two encodings at the same quality: PSNR. Similarly, the BD-PSNR measure the average PSNR difference in decibels (dB) for two different encoding algorithms considering the same bit rate. BD-BR and BD-PSNR are computed using the four Quantization Parameter (QP) values: 22, 27, 32 and 37. The used HEVC software encoder is the real time Kvazaar [KVL+15, KVV+15, VKL+15], detailed in Section 2.3.7, in All Intra (AI) configuration.

4.3.1.2 Embedded Platform

All experimentations are performed on one core of the embedded *EmETXe-i87M0* platform from *Arbor Technologies*, illustrated in Figure 4.6, and equipped with the *PBC-900J* processing board.

Table 4.1 – *Test sequences*

Class	Sequence name	Resolution	Nb frame	Frame rate	Source
A	Traffic	2560 × 1600	150	30	CTC
A	PeopleOnStreet	2560 × 1600	150	30	CTC
B	Kimono	1920 × 1080	240	24	CTC
B	ParkScene	1920 × 1080	240	24	CTC
B	Cactus	1920 × 1080	500	50	CTC
B	BQTerrace	1920 × 1080	600	60	CTC
B	BasketballDrive	1920 × 1080	500	50	CTC
C	RaceHorses	832 × 480	300	30	CTC
C	BQMall	832 × 480	600	60	CTC
C	PartyScene	832 × 480	500	50	CTC
C	BasketballDrill	832 × 480	500	50	CTC
D	RaceHorses	416 × 240	300	30	CTC
D	BQSquare	416 × 240	600	60	CTC
D	BlowingBubbles	416 × 240	500	50	CTC
D	BasketballPass	416 × 240	500	50	CTC
E	FourPeople	1280 × 720	600	60	CTC
E	Johnny	1280 × 720	600	60	CTC
E	KristenAndSara	1280 × 720	600	60	CTC

The embedded *EmETXe-i87M0* platform is based on an *Intel Core i5-4402E* processor clocked at 1.6 GHz and running the *Ubuntu 16.04.1 LTS* operating system. The Kvazaar real-time HEVC encoder is compiled by *gcc version 5.4.0* using the -O2 optimization option.

4.3.1.3 Energy Profiling Setup

To measure the energy consumed by the platform, the Intel Running Average Power Limit (RAPL) interfaces are used that retrieve the energy consumption of the CPU package, including cores, IOs, DRAM and integrated graphic chipset. As shown in [HSI+15], RAPL power measurements are coherent with external measurements and [EAD+12] proves the reliability of this internal measure across various applications. We confirm the coherency of the RAPL power measurements using a National Instrument (NI) PXI-6280 external data acquisition board used to measure the energy of the entire platform. The power was evaluated by measuring the current drawn by the board through a shunt 0.1Ω resistor added for this purpose and by knowing the power voltage of the platform.

The operating system running on the platform chooses the power states of the processor, which one corresponding to a power-performance trade-off mode. The Advanced Configuration and Power Interface (ACPI) standard [Bro05] specifies power states for devices. The state of a device is either working, stand-by or hibernating and noted as its *global state*. When in working mode, the processor can be in different states noted as C-states. The core or set of cores is then active. More power saving actions are taken for numerically higher C-states. Lower power C-states have longer entry and exit latency.

In this work, the power gap between a non-processing period, putting the processor in an IDLE state, and a video encoding period is measured. The CPU is considered to be

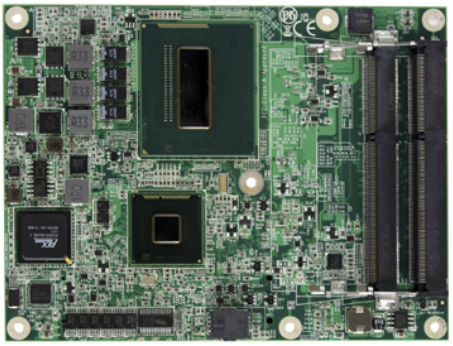


Figure 4.6 – *Embedded EmETXe-i87M0 platform from Arbor Technologies*

in IDLE state when it spends more than 90% of its time in the C7 C-states mode. The C7 state is the deepest C-state of the CPU, characterized by all core caches being flushed, the PLL and core clock being turned off as well as all uncore domains. The power of the entire board is measured to 16.7W when the CPU is in idle mode and goes up to 31W during video encoding in average. RAPL shows that 72% of this gap is due to the CPU package, the rest of the power going to the external memory, the voltage regulators and other elements of the board.

In the rest of the document, RAPL are used to obtain the energy consumption values of the experiments.

4.3.2 Coarse-Grain Energy consumption analysis

Video encoders, and especially HEVC encoders, include numerous configurations and tools, each one having different impacts on energy consumption. This section studies the impact of the input video stream properties and the coarse-grain features of the encoder.

4.3.2.1 Resolution and Frame Rate Level

First of all, the resolution of a video is what holds the most impact on the energy consumption of its encoding. The upper part of Figure 4.7 shows the normalized energy consumption average versus video resolution (2K, 1080p, 720p, 480p, 240p) for 100 frames.

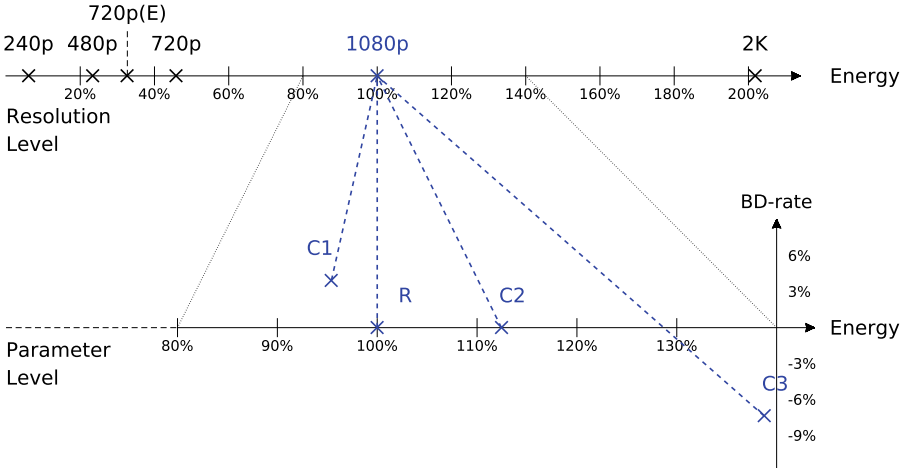


Figure 4.7 – *Normalized energy according to resolutions and configurations*

Encoding the 1080p video sequence consumes in average 2592 Joules, which is used to normalize the results.

The energy consumption of HEVC encodings increases linearly with the number of pixels by frame. Since the energy-per-pixel is stable, the frame rate becomes a significant parameter to reduce the energy consumption. For example, for two videos, with one having two times the resolution in number of pixels of the other and half the frame rate, the energy consumption will be approximately the same. Videos of class E (720p(E)) are a special case and do not follow the linear consumption since their content have specific properties such as a static background.

4.3.2.2 Encoder Parameter Level

The first HEVC-specific parameter level being used to reduce the energy consumption is based on three independent processes of the Intra prediction: the SAO and DBF filters, the *Transform skipping* test and the RDOQ. Four configurations are derived from these parameters due to their significant impact on the energy consumption, as show in Table 4.2.

Table 4.2 – *Encoder parameter configurations*

	Configuration			
Tool	C1	R	C2	C3
Filters (SAO+DF)	D	E	E	E
Transform skipping	D	D	E	D
RDOQ	D	D	D	E

D: Disable, E: Enable

The baseline configuration is defined as R using the default set of option of kvazaar (preset: medium) and the three others by *C1*, *C2* and *C3*. The lower part of Figure 4.7 shows the average normalized energy consumption of all configurations for the 1080p resolution. On the vertical axis, the average BD-BR are computed for four values of QP (22, 27, 32, 37). The BD-BR and the energy consumption, which is the result of the sum of the energies for the four QP, are normalized by the R configuration. In HEVC encoders, the tools increasing the energy consumption improve significantly the RD performance and thus the BD-BR. Figure 4.7 shows that the *Transform Skip* tool (C2) is not relevant in a real-time application due to its bad BD-BR/energy consumption trade off (the complexity increases without significant quality improvement).

4.3.2.3 Quantized Parameter Level

Figure 4.8 plots the average energy consumption (normalized by QP32) as a function of QP when the R configuration is used for encoding. Figure 4.8 shows that the energy consumption decreases as QP increases. This is because an encoding with higher QP quantizes data more aggressively, leading to a larger share of null coefficients after quantization and the RDO process is stopped sooner. This leads to faster encoding, and in turns less energy spent.

As a first conclusion, the coarse-grain analysis performed in a hierarchical approach shows that the energy consumption of a video encoding is linearly impacted by the resolution of the input video and differently impacted by high level HEVC features. In the

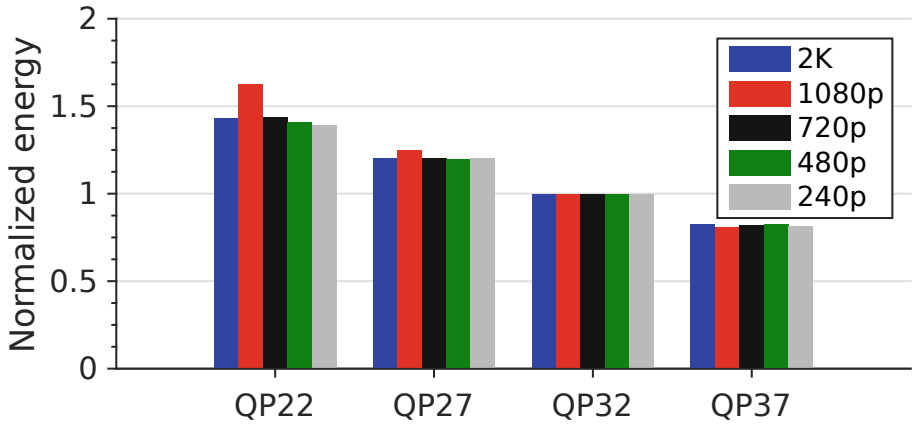


Figure 4.8 – *QP impacts of energy consumption according to resolutions*

next section, we analyze the energetic impact of lower granularity parameters. These parameters are driven by the RDO process and can significantly reduce complexity, as shown in the related works presented in Chapter 3. We also define the MEP which bounds the energy reduction search space of the MSSE algorithm part.

4.3.3 Energetic Impact of the Rate-Distortion Optimization Process

Complexity reduction techniques aim to improve the selection of the best configurations in the HEVC encoder in order to lower the energy consumption while trying to limit the deterioration of the rate distortion performance.

4.3.3.1 Determination of the Minimal Energy Point

The RDO selects CTU partitioning and prediction mode which lead to the minimal RD-cost. Depending on the entropy of the encoded block, different sizes of CTU are suitable. The RDO block carries out an exhaustive search in the partition set S by testing all possible CTU split partitioning and coding modes. As described in Chapter 3, complexity reduction techniques reduce the number of tested solutions (set S) and thus can reduce significantly the energy consumption.

We define the theoretical lower bound of the energy consumption for the two levels of the RDO: *CTU level* and *Intra prediction (IP) level*. These lower bounds are called respectively the Minimal Energy Point of *CTU level* (MEP-CTU) and the Minimal Energy Point of *IP level* (MEP-IP). As explained in Section 4.2.2.2.2, the MEP is the energy obtained when the encoder is able to predict perfectly the best solution and thus only this solution is considered to encode the CTU. Therefore, for a given MSSE algorithm, the energy consumption of the search process is reduced to the energy consumption of the solution and so the MEP is the minimal energy consumption point that can be achieved.

4.3.3.2 Energy Reduction Search Space

According to the HEVC encoding process detailed in Chapter 2 and the state-of-the-art of complexity reduction techniques detailed in Chapter 3, two MSSE algorithms, corresponding to an iterative RDO processes, are studied: *CTU level* and *Intra prediction (IP) level*. In the Kvazaar HEVC encoder, two features have been developed to reduce the number of iterative searches for the two levels: *early_split_termination* and *full_intra_search*. The

first feature stops the RDO splitting process when all transform coefficients are inferred to be equal to zero. When *full_intra_search* is disabled, the number of angular modes searched is reduced using Rough Mode Decision (RMD). These two features lead to distinct configurations linked to corresponding levels: *CTU level* and *IP level*. To each configuration and level is matched a MEP that defines the 9 configurations described in Table 4.3 and their respective energy consumptions are plotted in Figure 4.9.

Table 4.3 – *CTU and IP configurations*

	Configuration								
Tool	M2	M1	M	T2	T1	T	R2	R1	R
early_split_termination	-	-	-	E	E	E	D	D	D
full_intra_search	-	D	E	-	D	E	-	D	E
MEP-CTU	E	E	E	D	D	D	D	D	D
MEP-IP	E	D	D	E	D	D	E	D	D

D: Disable, E: Enable, -: no impact

Figure 4.9 shows the average energy consumption and BD-BR of the different configurations organized in the two levels: *CTU level* and *IP level* for 1080p sequences. The energy reference point (R) is based on the R configuration of the Table 4.2. As shown by the T point on Figure 4.9, the *early_split_termination* tool reduces the energy consumption without degrading the BD-BR (+0.16%). The *full_intra_search* does not affect the BD-BR as the RDO process can compensate by adjusting the CTU splitting shown by *T1* and *R1*. This compensation does not apply to *M1* because its CTU splitting is fixed and thus an important increase of 2.03% of BD-BR is observed. Each *CTU level* energy point serves as a MEP reference point for several *IP level*s. The MEP show the maximum achievable energy reduction.

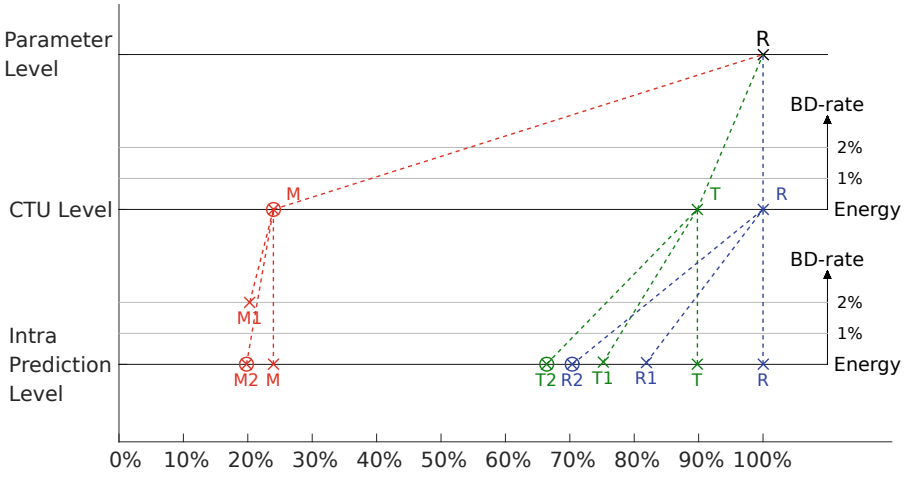


Figure 4.9 – *Normalized energy according to CTU and IP configurations , with ⊗ for MEP*

The *energy reduction search space* is defined by the percentage of energy reduction between a reference point and its associated MEP. All complexity reduction techniques summarized in Chapter 3 are constrained within this space of achievable energy reduction. Table 4.4 summarizes the normalized energy reduction for the different video resolutions. The results show that the search space is consistent across all resolutions and that the largest energy reduction search space occurs at the *CTU level* with up to 78.1% of potential energy reduction whereas working on the *IP level* offers 30% at best.

Table 4.4 – *Normalized energy consumption according to resolution (in %)*

	Configuration								
Res.	M2	M1	M	T2	T1	T	R2	R1	R
2k	18.3	19.7	23.4	61.5	73.9	90.9	65.0	80.1	100
1080p	19.0	20.8	23.9	63.4	73.5	89.3	67.6	80.8	100
720p	16.3	18.5	21.9	49.2	60.7	75.5	58.7	77.1	100
720p(E)	18.4	20.0	23.9	66.3	74.8	87.8	71.0	83.2	100
480p	20.4	21.6	25.6	69.1	78.3	93.4	71.7	82.9	100
240p	22.8	23.8	27.9	74.6	80.5	95.5	76.3	83.4	100

From this analysis we can conclude that in real-time software HEVC encoder, the energy problematic can be more efficiently addressed by reducing complexity at the *CTU level* rather than at the *IP level*.

4.4 Conclusion

This chapter has introduced the concept of Minimization based on Search Space Exploration (MSSE) and studied the HEVC encoding process within this context to infer bounds on maximum achievable energy reduction.

Section 4.2 has detailed a new approach exploiting the computation-skipping concept by using a Smart Search Space Reduction (SSSR) technique. This technique aims at classifying MSSE algorithms according to their potential cost gain to reduce the search space. Designing efficient SSSR techniques and adjusting the parameters that balance complexity reduction and efficiency of the approximate application has been shown to be not trivial. It requires an in-depth knowledge of the application.

Complementary, Section 4.3 has presented an analysis of energy reduction opportunities by considering different levels of granularity, from global video parameters to prediction units. We measured the search space in terms of energy reduction according to parameters of these different granularity levels. This study demonstrates several elements: at the coarsest granularity, the energy of HEVC real-time Intra encoding is proportional to video resolution. At medium granularity, the transform skipping method is not effective in reducing encoding energy. At a lower granularity, two MSSE algorithms have been studied; the *CTU level* has a potential of energy reduction up to 78.1% whereas the *IP level* offers at best 30% of energy reduction.

According to the SSSR methodology defined in Section 4.2.2.2 and due to its potential of energy reduction, we focus the next chapters on the *CTU level* MSSE. The next step is then to design a *Coarse Solution Predictor* for the purpose of playing with the size of the search space of this MSSE: from 100% of energy consumed to the MEP-CTU (21.9%) defined in Section 4.3.3.2 and illustrated in Figure 4.9.

The two next chapters propose two methods that predict the quad-tree partitioning in "one-shot", i.e. without iterating on the encoding of a CTU. Chapter 5 presents a statistical approach based on variance-aware quad-tree prediction whereas Chapter 6 presents a second method based on a *Machine Learning (ML)* approach.

CHAPTER 5

Quad-Tree Prediction Based on Statistical Approach

5.1 Introduction

As shown in Chapter 3, to reduce the computational complexity of HEVC encoders, several algorithmic solutions have been proposed at the level of quad-tree partitioning, which test less partitioning configurations than exhaustive method. Exhaustive search solution chooses the optimal solution in terms of RD-cost after encoding and testing all possible partitioning configurations. The study carried out in Section 4.3 of Chapter 4 has shown that this process is the most time consuming operation in an HEVC encoder and thus it offers the biggest opportunity of complexity reduction (up to 78%) [MAH+17b]. Complexity reduction solutions at the quad-tree level consist in predicting, before encoding, the adequate level of partitioning that offers the lowest RD-cost. Authors in [SZA13] and [CNP12] propose to use the correlation between the minimum depth of the co-located CTU in the current and previous frames to skip computing some depth levels during the RDO process. Authors in [BC15, FDZX16, WX16, KSH13, PCL16] use CTU texture complexities to predict the quad-tree partitioning. However, these solutions are based on reducing the complexity of the best effort (i.e. non-real-time) HM encoder and do not offer the possibility to tune the compression quality and complexity. Moreover, a real-time encoder such as Kvazaar is up to 10 times faster than HM [VKL+15]. The performance of state-of-the-art solutions based on HM are biased because they are measured comparatively to a gigantic compression time. The complexity overhead of state-of-the-art solutions is thus comparatively higher in the context of a real-time encoder. Published results can thus not be directly applied to reduce the energy consumption in a real-time encoder without a knowledge of the complexity overhead induced by prediction features computation.

This chapter proposes a new lightweight method to predict the CTU quad-tree partitioning with the objective to budget the energy consumption of a real-time HEVC encoder. The proposed approach is used to limit the recursive RDO process, which drastically impacts the energy consumption of HEVC Intra encoders [MAH+17b]. Based on the SSSR methodology defined in Section 4.2 of Chapter 4, the predictor is also made adjustable, based on two parameters that provide a trade-off between energy consumption and coding efficiency. The proposed energy reduction technique exploits the correlation between the CTU partitioning and the variance of the CU luminance samples to predict the quad-tree partitioning of a CTU before starting the RDO process. Compared to state-of-the-art so-

lutions, the originality of the presented work comes from its focus on energy consumption, real-time encoders and adjustable energy gains targeted towards energy budgeting.

The rest of this chapter is organized as follows. Section 5.2 presents an analysis of CTU variance for predicting quad-tree partitioning in one-shot. Section 5.3 study the prediction of quad-tree partitioning performances. Section 5.4 details the proposed algorithm of quad-tree partitioning prediction based on variance studies. The proposed energy reduction scheme and its performance are investigated in Section 5.5. Finally, Section 5.6 concludes this chapter.

5.2 CTU Variance Analysis for Predicting an HEVC Quad-Tree Partitioning

The aim of this chapter is to replace the brute force scheme usually employed in HEVC encoders by a low complexity algorithm which predicts the CTU partitioning for Intra prediction without testing all possible decompositions. Following a bottom-up approach (from CU 4×4 to 32×32), the main idea is to determine whether a CU at a current depth $d \in [1,4]$ should be encoded or need to be further merged into lower depth $d-1$.

It has been already shown that the CTU partitioning during the RDO process is highly linked to the QP value and the texture complexity which can be statistically represented by the variance of blocks in Intra coding [KSH13, BC15, WX16]. Figure 5.1 shows the CU boundaries of the 6^{th} frame of *BasketballDrive* video sequence.

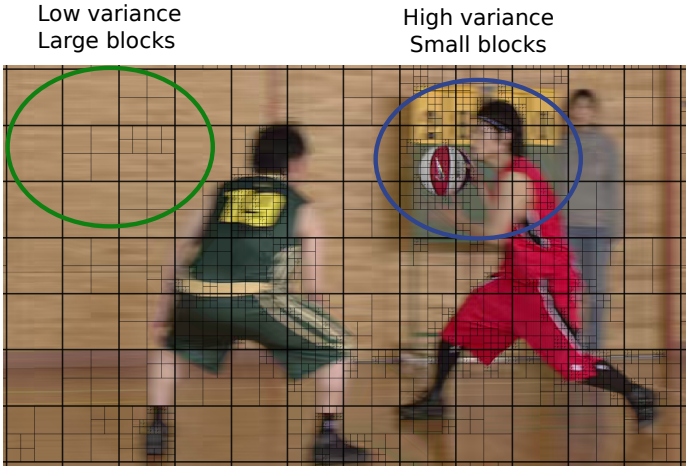


Figure 5.1 – *Quad-tree partitioning of the 6th* HEVC *intra coded frame of the* BasketballDrive *sequence. The green (resp. blue) circle shows that the lowest (resp. highest) variance regions tend to be encoded with larger (resp. smaller) units.*

It is noteworthy that the regions with the lowest variance (smooth) tend to be encoded with larger blocks, as illustrated by the green circle in Figure 5.1, while the blue circle shows a region with a high variance (high local activity), which are encoded with smaller blocks. This existing correlation between the pixel's values of a block (variance) and its encoding decomposition can be used to predict the quad-tree decomposition of a CTU and thus drastically reduce the encoding complexity.

5.2.1 Variance-Based Decision for Quad-Tree Partitioning

To study how to predict the quad-tree partitioning from the variance values of CU luminance samples, two populations of CU at a current depth d are defined: *Merged (M)* and *Non Merged (NM)*. The CU belongs to the *Non Merged* population when the full RDO process chooses to encode the CU at the current depth d, while the CU belongs to the *Merged* population when the RDO process choose to encode the CU at a new depth d' with $d' < d$. However, with a bottom-up approach (i.e. d from 4 to 1), all CU of the quad-tree decomposition of all CTU can be classified into one of these two populations.

In the following paragraphs, two approaches are investigated to classify CU in one of these two populations based on the variance criteria.

5.2.1.1 Probability Density Function (PDF) approach

Figure 5.2 shows the PDF of the variance of the CU 8×8 of the two populations for the 6th frame of *BasketballDrive* video sequence. Statistically, for two PDF with sufficiently distinct populations, a threshold can be derived such that, given an element — here the variance of a 8×8 CU — the element can be classified to one of the two populations. The classification threshold is given by the abscissa of the intersection of the two PDF, represented by the green dotted line in Figure 5.2. The threshold is determined such as to minimize the miss detection and false alarm probability.

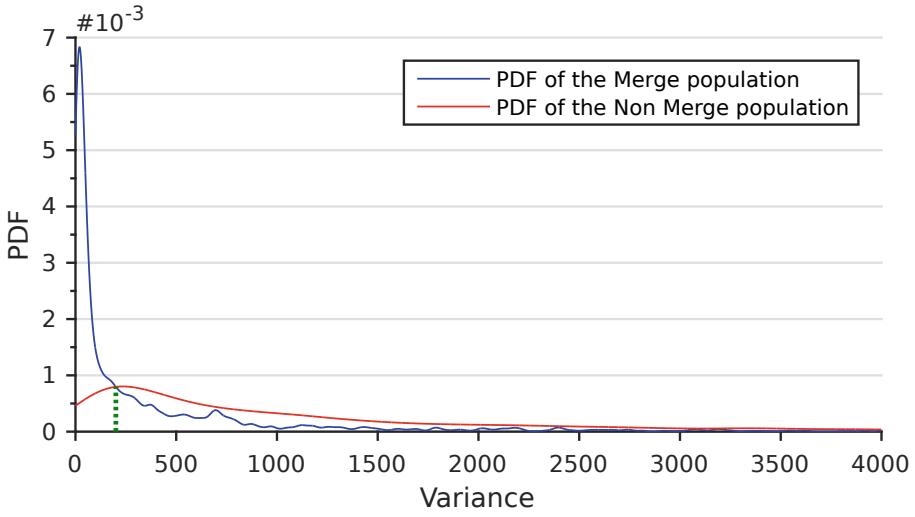


Figure 5.2 – *PDF of CU* 8×8 *variances of the two populations* Merged *and* Not Merged *of the sixth frame of the sequence* BasketballDrive. *A variance threshold (the green dotted line) can be used to classify a block into one of the two populations.*

5.2.1.2 Cumulative Distribution Function (CDF) approach

An alternative approach consists in using only the Cumulative Distribution Function (CDF) of the *Non Merged* population to decide whether a CU has to be merged or not. In our case, the CDF defines the probability of the variance population of a given CU size being less or equal to a given value.

Figure 5.3 shows the CDF of CU variances depending on CU size for the sixth frame of the *BasketballDrive* video sequence. The CDF curves show that the probability for a CU size to be selected during the RDO process decreases when the variance of the CU

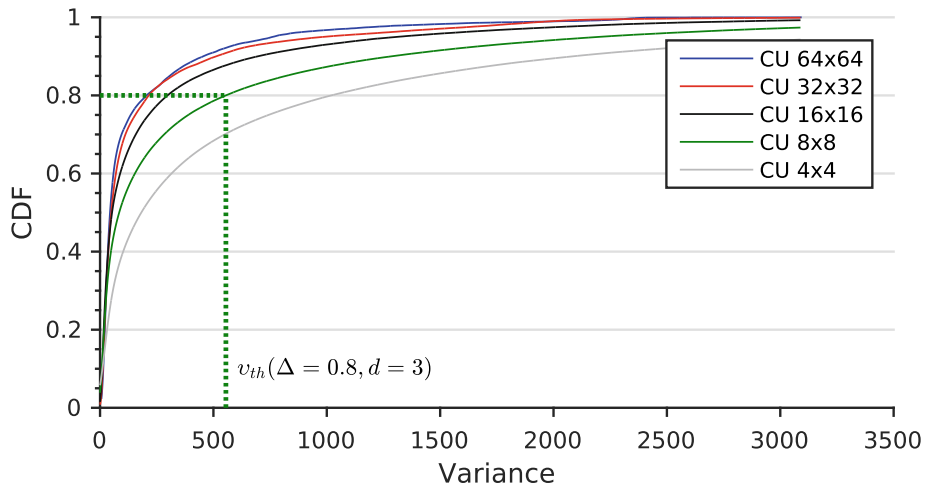


Figure 5.3 – *CDF of the Merged population depending on CU size for the sixth frame of the sequence* BasketballDrive. *Under a specific probability* Δ*, a variance threshold can be extracted from the inverse CDF curve to classify a block as Merged.*

increases. In other terms, it is rare for a CU to have a variance greater than a certain threshold. From this observation, a variance threshold $v_{th}(\Delta, d)$ for each depth d can be extracted from the inverse CDF curve for a specific probability Δ. For example, Figure 5.3 shows that 80% ($\Delta = 0.8$) of CU 8×8 ($d = 3$) have a variance less than the value 555 represented by the green dotted line in Figure 5.3. Δ is the percentage of coding units whose variance is under the threshold v_{th}, i.e. the variance threshold that triggers unit split.

Table 5.1 shows the thresholds $v_{th}(\Delta, d)$ for $d \in \{1, 2, 3, 4\}$ extracted from the CDF using the second previous approach for the 50th frame of the two sequences *PeopleOnStreet* and *ParkScene*.

Table 5.1 – *Variance thresholds* $v_{th}(\Delta, d)$ *of the 50th frame of two example sequences versus* d *and* Δ

Sequence name	Δ	$d = 1$	$d = 2$	$d = 3$	$d = 4$
PeopleOnStreet	0.3	31.8	31.1	51.4	97.0
	0.5	40.8	40.1	66.4	127.0
	0.7	49.8	50.1	84.4	166.0
	0.9	58.8	61.1	109.4	219.0
ParkScene	0.3	41.2	24.3	29.1	53.5
	0.5	51.2	31.3	37.1	70.5
	0.7	62.2	40.3	48.1	90.5
	0.9	74.2	50.3	62.1	117.5

The Table illustrates the large variation of the threshold value across different video contents. In fact, the thresholds rely on the video contents and thus have to be determined on-the-fly from the Learning Frame (F_L).

5.2.2 Variance Threshold Modeling

Since the thresholds have to be adapted based on the video content, they have to be determined on-the-fly from learning frames. The modeling of these thresholds can be conducted using variance PDF with an approximation of the distribution based on common known probability distribution or directly using population features. Neither of those two approaches were conclusive in our experimental results.

However, an approximation of the thresholds directly from *Non Merged* population features provides good results for the CDF curve. In Figure 5.4, the variance thresholds $\upsilon_{th}(d = 3, \Delta = 0.8)$ is plotted versus the mean $\mu_\upsilon(d = 3)$ and the standard deviation $\sigma_V(d = 3)$ of CU 8×8 variances. The values are extracted for 4 QP values 22, 27, 32 and 37 of 100 frames selected randomly from 5 different sequences: *BasketballDrive*, *BQTerrace*, *Cactus*, *ParkScene*, *PeopleOnStreet* and *Traffic*. Similar results are obtained for other CU sizes. The results show that for a fixed value of Δ, $\upsilon_{th}(\Delta, d)$ depends linearly on $\mu_\upsilon(d)$ and standard deviation $\sigma_\upsilon(d)$, and this independently of the QP value.

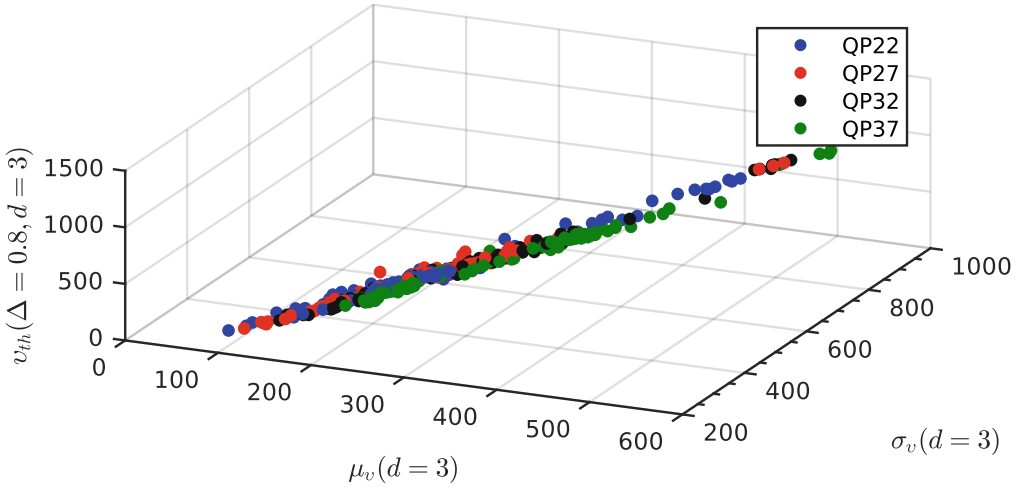


Figure 5.4 – *Variance thresholds $\upsilon_{th}(d = 3, \Delta = 0.8)$ versus the mean $\mu_\upsilon(d = 3)$ and the standard deviation $\sigma_V(d = 3)$ of CU 8×8 variances. For a fixed value of Δ, the variance threshold values form a plane (independently of the QP value).*

From this observation, $\upsilon_{th}(\Delta, d)$ can be modeled using the following linear relation:

$$\upsilon_{th}(\Delta, d) = a(\Delta, d) \cdot \mu_\upsilon(d) + b(\Delta, d) \cdot \sigma_\upsilon(d) + c(\Delta, d) \tag{5.1}$$

where $a(\Delta, d)$, $b(\Delta, d)$ and $c(\Delta, d)$ are coefficients modeling the threshold for each probability Δ and for each depth d. The coefficients are computed off-line for each Δ and d values using a linear regression on all frames of *BasketballDrive*, *BQTerrace*, *Cactus*,*ParkScene*, *PeopleOnStreet* and *Traffic* for 4 QP values: 22, 27, 32 and 37.

The Coefficient of Determination (*Rsq*) is a metric that quantifies the accuracy of a predicting model. It takes values between 0 and 1 and the more the *Rsq* is close to 1, the more the predicted data fits the actual data. *Rsq* is defined by Equation (5.2) where $\hat{y}$ represents the estimated values of y, n is the number of y value used for the measure and $\bar{y}$ is the mean of y.

$$Rsq = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5.2}$$

Table 5.2 summarizes the average of the *Rsq* for $\Delta \in \{0.3, 0.35, \ldots, 0.9\}$ for each depth d. The average *Rsq* value is always larger than 72%, which confirms that the model fits the $v_{th}(\Delta, d)$, regardless of the video content and QP value.

Table 5.2 – *Rsq average of* $V_{th}(\Delta, d)$ *modeling*

Depth	Average *Rsq*
$d = 0$	0.8774
$d = 1$	0.8482
$d = 2$	0.7214
$d = 3$	0.8942
$d = 4$	0.9436

The following items summarize the above analysis:

- Thresholds from CDF of variances can be predicted from reference Learning Frame (F_L).
- Look-Up Table (LUT) requires slight computation and memory overhead for the determination of the threshold. The computation overhead of the proposed method presented in section 5.5 is between 1% and 1.9% depending on the QP and video sequence.
- The prediction of thresholds is independent of the QP value (Figure 5.4).
- Threshold modeling is accurate with a mean *Rsq* of 0.86 for the different depths.
- Thresholds can be precomputed according to Δ value as a parameter.

5.3 Statistical Prediction Algorithm for Coding Tree Unit Partitioning

This following section describes the proposed statistical algorithm and prediction scheme which predict in "one-shot" the CTU partitioning, i.e. before starting the RDO process of the CTU, using a variance criterion and the thresholds $v_{th}(\Delta, d)$.

5.3.1 Coding Tree Unit partitioning representation

A description of the CTU partitioning is needed to explicitly depict the prediction of the quad-tree and then force the encoder to only encode this specific decomposition. Figure 5.5 illustrates the chosen representation of a CTU partitioning in the form of a CTU Depth Map (CDM) matrix 8 by 8. Each element of the matrix represents the depth d of a 8 by 8 square samples of the CTU. Since the CTU size is 64×64 and the minimum size of CU is 4×4, a matrix 8 by 8 can be used to describe all partitioning of a CTU. A depth of 4 in the CDM corresponds to 4 CU 4×4 in the CTU decomposition.

5.3.2 Variance-Aware Prediction Algorithm for Coding Tree Unit Partitioning

Algorithm (5.1) describes the proposed statistical algorithm that predicts the CTU partitioning in one-shot. The algorithm takes as inputs the luminance samples of CTU and the

3	3	2	2	1	1	1	1
3	3	2	2	1	1	1	1
2	2	3	3	1	1	1	1
2	2	3	4	1	1	1	1
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2

Figure 5.5 – CDM *matrix of the CTU partitioning of Figure 2.8 on page 19. Each element of the matrix represents the depth of an 8 by 8 pixel square in the CTU.*

table of thresholds v_{th} previously computed by Equation (5.1) to generate the CDM associated to the input CTU. In other terms, the goal of this algorithm is to use the variances of the luminance samples to determine the CDM matrix of the CTU. Then, the encoder only uses the predicted depths instead of RDO to encode the video, reducing significantly the complexity.

For a given CTU, let $v^d(i,j)$ be the variance of the luminance sample blocks of size $2^{6-d} \times 2^{6-d}$ at the depth level d and the local coordinates (i,j) into the CTU as illustrated in Figure 5.6.

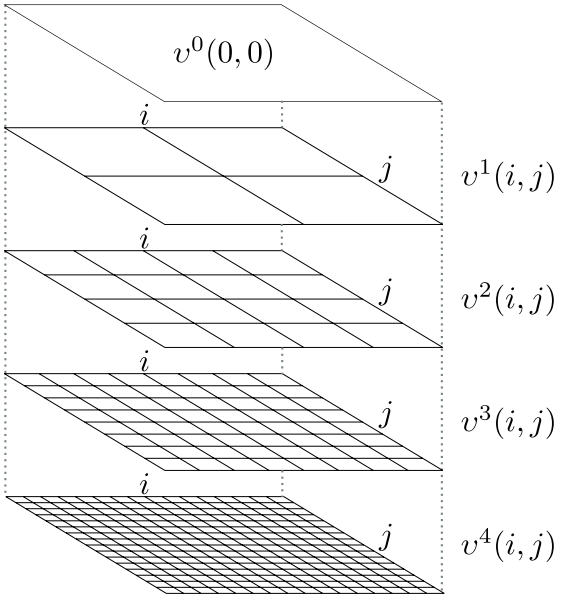


Figure 5.6 – $v^d(i,j)$*: variance of the luminance sample blocks of size* $2^{6-d} \times 2^{6-d}$ *of a CTU versus the depth level* d*.*

First of all, the full CDM is initialized with the depth value 4 (line 1) and all the variances $v^4(x,y)$ of the CU 4×4 (line 2) are computed using Equation (5.3) where $p_{x,y}(i,j)$ is the luminance component of the samples at the coordinate (i,j) in the CU 4×4 at the position (x,y) and $\bar{p}_{x,y}$ the average value of the block.

$$v^4(x,y) = \frac{1}{16} \sum_{i=0}^{3} \sum_{j=0}^{3} (p_{x,y}(i,j) - \bar{p}_{x,y})^2 \tag{5.3}$$

Algorithm 5.1: Statistical CTU Depth Map (CDM) Generation

Data: Samples of CTU, $v_{th}(\Delta, d)$
Result: CDM matrix

1 $\text{CDM}(x, y) = 4, \forall x, y \in [0, 7]$; // Initialization
2 Compute: $v^4(x, y), \forall x, y \in [0, 15]$; // cf Eq. 5.3
3 **for** $(d = 4; d > 0; d--)$ **do**
4 $\quad \delta = 2^{4-d}$; // CDM matrix block size
5 $\quad$ **for** $(y = 0; y < 8; y \mathrel{+}= \delta)$ **do**
6 $\quad\quad$ **for** $(x = 0; x < 8; x \mathrel{+}= \delta)$ **do**
7 $\quad\quad\quad$ // Test if the 4 neighbor blocks have the same depth d when $d < 4$
8 $\quad\quad\quad$ **if** $(\text{CDM}(x, y) = d \;\&\&$
$\quad\quad\quad\quad \text{CDM}(x + \lfloor \frac{\delta}{2} \rfloor, y) = d \;\&\&$
$\quad\quad\quad\quad \text{CDM}(x, y + \lfloor \frac{\delta}{2} \rfloor) = d \;\&\&$
$\quad\quad\quad\quad \text{CDM}(x + \lfloor \frac{\delta}{2} \rfloor, y + \lfloor \frac{\delta}{2} \rfloor) = d)$ **then**
9 $\quad\quad\quad\quad$ // Test if the variances of blocks are lower than the threshold
10 $\quad\quad\quad\quad$ $x' = x/(2^{3-d}); y' = y/(2^{3-d})$;
$\quad\quad\quad\quad$ **if** $(v^d(x', y') < v_{th}(\Delta, d) \;\&\&$
$\quad\quad\quad\quad$ $v^d(x' + 1, y') < v_{th}(\Delta, d) \;\&\&$
$\quad\quad\quad\quad$ $v^d(x', y' + 1) < v_{th}(\Delta, d) \;\&\&$
$\quad\quad\quad\quad$ $v^d(x' + 1, y' + 1) < v_{th}(\Delta, d))$ **then**
11 $\quad\quad\quad\quad\quad$ // Block merging in the CDM
12 $\quad\quad\quad\quad\quad$ $\text{CDM}(x + i, y + j) = d - 1,$
$\quad\quad\quad\quad\quad\quad \forall i, j \in [0, \delta - 1]$;
13 $\quad\quad\quad\quad\quad$ Compute: $v^d(x'/2, y'/2)$; // cf Eq. 5.4

Then, the algorithm explores the CTU decomposition with a bottom-up approach: from $d = 4$ to $d = 1$ (line 3). For the current depth d, the algorithm browses the CDM (lines 5-6) taking the block size δ in the CDM (line 4) into account. Afterwards, the algorithm tests whether the 4 neighbor blocks in the Z-scan order have the same depth d, except when $d = 4$ (line 8). The algorithm does not try to merge neighbor blocks if they have different depths. Since the algorithm is bottom-up, there is no need to test the condition when $d = 4$ because the CDM is initialized at $d = 4$ which is the starting depth of the algorithm. If the previous condition is true, then the algorithm tests whether the blocks have to be merged or not using the variance criteria (line 10) previously detailed in Section 5.2.1. If the 4 blocks variances v^d are lower than the given threshold $V_{th}(\Delta, d)$ then the blocks are merged and the corresponding elements in the CDM are set to $d - 1$ (line 12) and the variance of the merged block is calculated three times using the combined variance Equation 5.4.

$$v_{a \cup b} = \frac{(2n - 1)(v_a + v_b) + n(\mu_a - \mu_b)^2}{4n - 1} \tag{5.4}$$

Equation (5.4) [CGL82] computes the variance of two sets of data a and b containing the same number of observations n with μ and v corresponding to the mean and the variance of the specified data set, respectively.

5.4 Relationship between the Prediction Performance and the Δ Parameter

As described in Section 5.3, Algorithm (5.1) takes as input a set of variance thresholds v_{th} corresponding to the parameter Δ. As detailed in Section 5.2.1, the parameter Δ is the normalized probability of *Non Merged* population with a variance less or equal to its associated threshold $v_{th}(\Delta, d)$. The following section is a baseline study of the impact of parameter Δ.

5.4.1 CTU Depth Map Distance Γ Definition for Performance Analysis

The main objective of Algorithm (5.1) is to generate a CDM that minimizes the prediction error when compared to what the full RDO process would generate. To evaluate the accuracy of our predictions, we define the normalized L_1 *distance* Γ between two CDM in terms of depth levels as follows:

$$\Gamma(A,B) = \left[\sum_{i=0}^{7}\sum_{j=0}^{7}|A(i,j) - B(i,j)|\right] /64 \tag{5.5}$$

where A and B are the two compared CDM. In other terms, the metric $\Gamma(A,B)$ measures the average gap in number of depth levels between two CDM A and B of a given CTU. Let use Figures 5.7 as example, the distance Γ between the CDM Figure 5.7a and Figure 5.7b is equal to $\Gamma = (4+3+16)/64 = 0.3594$. Distance Γ is used in the following sections to evaluate the accuracy of the prediction.

3	3	2	2	1	1	1	1
3	3	2	2	1	1	1	1
2	2	3	3	1	1	1	1
2	2	3	4	1	1	1	1
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2

(a) *Example of* CDM *matrix A.*

2	2	2	2	1	1	1	1
2	2	2	2	1	1	1	1
2	2	4	3	1	1	1	1
2	2	4	3	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

(b) *Example of* CDM *matrix B*

Figure 5.7 – *Example of* CDM *matrix A and B to illustrated the distance* Γ *metric. Gray blocks represent the blocks that differ in terms of the CU size between the two* CDM *A and B.*

5.4.2 CTU Depth Map Distance Versus Δ

In Figure 5.8 are plotted in blue the average and the standard deviation of $\Gamma_\Delta(P,R)$ where $\Gamma_\Delta(P,R)$ is the distance between the predicted CDM P and the reference CDM R [1], generated by a full RDO process (optimal) according to $\Delta \in \{0.3, 0.35, ..., 0.95\}$. The results are extracted from the 100 first frames of the 5 following sequences: *BasketballDrive*, *BQTerrace*, *Cactus*, *ParkScene*, *PeopleOnStreet* and *Traffic* for 4 QP values: 22, 27, 32 and 37. The more $\Gamma(P,R)$ is close to 0, the more precise the predicted CDM P becomes.

[1] exhaustive search leading to the optimal solution

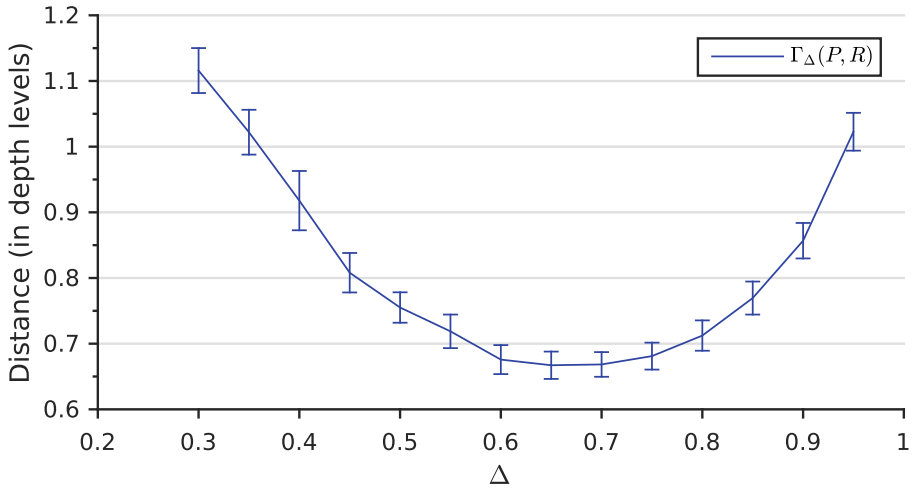


Figure 5.8 – *Averages and standard deviations of* $\Gamma_\Delta(P,R)$*, the distance between the predicted* CDM P *and the reference* CDM R *generated by a full RDO process versus* Δ*. The distance is minimized, i.e. the prediction accuracy is maximized for* $\Delta \in [0.6, 0.7]$.

Figure 5.8 shows that $\Gamma_\Delta(P,R)$ has the property of being convex when plotted over Δ. Furthermore, the low standard deviations values (< 0.45) of $\Gamma_\Delta(P,R)$ (represented by the vertical bars in Figure 5.8) show that the results are stable across the video contents and QP value. As can be seen in Figure 5.8, $\Gamma_\Delta(P,R)$ is minimized and thus the accuracy of the predictions is maximized for $\Delta \in [0.6, 0.7]$. Moreover, the value of $\Gamma_\Delta(P,R)$ is below 0.7 which demonstrates the accuracy of the proposed predictions.

5.4.3 Decomposing the CTU Depth Map Distance into Lower and Upper Distances

To extend the previous analysis, the distance $\Gamma(A,B)$ can be decomposed into two independent distances: $\overline{\Gamma(A,B)}$ and $\underline{\Gamma(A,B)}$ complying with Equation (5.6):

$$\Gamma(A,B) = \overline{\Gamma(A,B)} + \underline{\Gamma(A,B)} \tag{5.6}$$

where $\overline{\Gamma(A,B)}$ (respectively $\underline{\Gamma(A,B)}$) is called *upper distance* (respectively *lower distance*) and is the normalized distance between the two CDM A and B only when the depth of A is lower (respectively higher) than the depth of B, as described by Equation (5.7) (respectively Equation (5.8)). $\overline{\Gamma(A,B)}$ is called the *upper distance* as it represents the fact that the quad-tree decomposition of CDM A is shallower than the one of B, i.e the CDM of A has lower depth values. The same reasoning can be applied to $\underline{\Gamma(A,B)}$.

$$\overline{\Gamma(A,B)} = \left[\sum_{i=0}^{7} \sum_{j=0}^{7} |\min\left(A(i,j) - B(i,j), 0\right)| \right] /64 \tag{5.7}$$

$$\underline{\Gamma(A,B)} = \left[\sum_{i=0}^{7} \sum_{j=0}^{7} \max\left(A(i,j) - B(i,j), 0\right) \right] /64 \tag{5.8}$$

In green and red in Figure 5.9 are plotted the averages and the standard deviations of respectively $\overline{\Gamma_\Delta(P,R)}$ and $\underline{\Gamma_\Delta(P,R)}$, i.e. the *upper distance* and *lower distance* between

the predicted CDM P and the reference CDM R generated by a full RDO process with $\Delta \in \{0.3, 0.35, ..., 0.95\}$. The decomposition of $\Gamma_\Delta(P,R)$ into $\overline{\Gamma_\Delta(P,R)}$ and $\underline{\Gamma_\Delta(P,R)}$ is interesting as it tells the difference between the two following types of prediction errors: the error when the predicted depth is lower than the optimal depth and when the predicted depth is higher than the optimal depth. Figure 5.9 shows that $\overline{\Gamma_\Delta(P,R)}$ and $\underline{\Gamma_\Delta(P,R)}$ are respectively strictly decreasing and increasing when Δ increases.

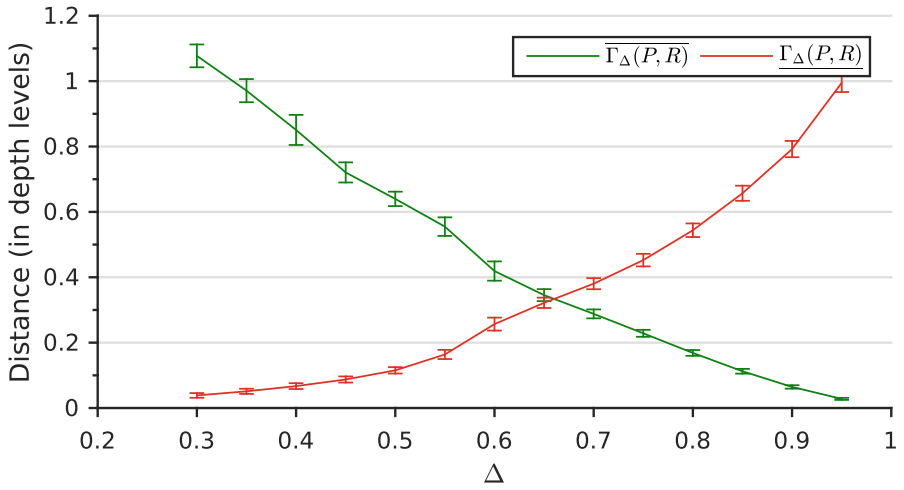


Figure 5.9 – *Averages and standard deviations of* $\overline{\Gamma_\Delta(P,R)}$*, the* upper *and* lower distance *between the predicted* CDM P *and the reference* CDM R *generated by a full RDO process versus* Δ.

For a low value of Δ, such as $\Delta = 0.3$ in Figure 5.9, the main part of errors is due to low values of depth in the predicted CDM P, i.e. $\underline{\Gamma_\Delta(P,R)}$ close to 0. Indeed, as shown by Figure 5.3, a low value of Δ implies low values of thresholds $v_{th}(\Delta, d)$ which according to Algorithm (5.1) lead to fewer merges during the CDM generation. The resulting CDM has therefore a finer-grained splitting and minimizes the prediction errors on small CU in the CTU. In contrast, for a high value of Δ, such as $\Delta = 0.95$ in Figure 5.9, the resulting CDM has a coarser-grained splitting and minimizes the prediction errors of large CU in the CTU.

As a result of the previous analysis, we propose to force the HEVC encoder to encode between two CDM: $\text{CDM}_{\overline{\Delta}}$ and $\text{CDM}_{\underline{\Delta}}$ generated from two different values of Δ with $\overline{\Delta} < \underline{\Delta}$ to minimize both errors $\overline{\Gamma_\Delta(P,R)}$ and $\underline{\Gamma_\Delta(P,R)}$.

5.4.4 Refinement Algorithm for Coding Tree Unit Partitioning

To increase the accuracy of the one-shot depth map prediction with a limited impact on the complexity, a second algorithm is designed that refines the CDM.

The *Refinement* algorithm, described by Algorithm (5.2), takes as input a CDM matrix from Algorithm (5.1) and generates a second CDM called Refined CTU Depth Map (RCDM). The RCDM is the result of merging all groups of four neighboring blocks (in the Z-scan order) having the same depth in the input CDM. Algorithm (5.2) details the process as follows.

The first step is to check whether the input CDM's depth is equal to 0, if so then no merge can be applied and thus the RCDM is also set to 0 (line 2). If not, the CDM is analyzed element by element (lines 4-5). Due to the fact that a depth of 4 in a CDM corresponds to 4 CU 4×4, they are always merged to a depth 3 and thus the value in

Algorithm 5.2: CDM Refinement

Data: CDM matrix
Result: RCDM matrix

```
1  if CDM(0,0) = 0 then
2      RCDM(x,y) = 0, ∀x,y ∈ [0,7];                          // Set RCDM at 0
3  else
4      for (y = 0; y < 8; y++) do
5          for (x = 0; x < 8; x++) do
6              if CDM(x,y) = 4 then
7                  RCDM(x,y) = 3                              // Automatic merge
8              else
9                  d = CDM(x,y)                               // Get the depth
10                 N = 16/2^d                                 // Offset definition
11                 // Test if it is the last blocks in the Z-scane order for
                      the depth d
12                 if ((x % N) = N/2) && ((y % N) = N/2) then
13                     // Test if the three other blocks have the same depth
14                     if (CDM(x − N/2, y) = d &&
                           CDM(x, y − N/2) = d &&
                           CDM(x − N/2, y − N/2) = d) then
15                         RCDM(x + i, y + j) = d,
                               ∀i,j ∈ [0, N/2];                // Fill the RCDM
```

the RCDM is automatically set to 3 (line 7). For the general case (i.e $d \in 1,2,3$), if the evaluated element in the matrix corresponds to the fourth block (in the Z-scan order) of the given depth d (line 12) and if the 3 others blocks are of depth d (line 14), then the algorithm fills the corresponding blocks of the RCDM with the upper depth $d-1$ (line 15).

3	3	2	2	1	1	1	1
3	3	2	2	1	1	1	1
2	2	3	3	1	1	1	1
2	2	3	4	1	1	1	1
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2

(a) CDM *matrix example. Each element of the matrix represents the depth of an 8 by 8 pixel square in the CTU.*

2	2	2	2	1	1	1	1
2	2	2	2	1	1	1	1
2	2	3	3	1	1	1	1
2	2	3	3	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

(b) CDM *and its associate* RCDM *matrices of the CTU partitioning of Figure 5.10a*

Figure 5.10 – CDM *and its associate* RCDM *matrices of the CTU partitioning of Figure 5.10a*

Figures 5.10 show an example of a CDM (Figure 5.10a) and its associated RCDM (Figure 5.10b) matrices. The gray blocks in the RCDM Figure 5.10b represent the merged blocks. The next section describes our proposed energy reduction scheme.

5.5 Using Statistical Quad-Tree Partitioning Prediction to Reduce the Encoder Energy Consumption

To reduce the energy consumption of HEVC encoder, we propose to limit the exhaustive search of the RDO process on the CTU quad-tree decomposition. The proposed energy reduction technique aims to predict the coding-tree partitioning based on video frame content properties. We propose a variance-aware quad-tree partitioning prediction.

To limit the exploration of the quad-tree decomposition of a CTU, two CDM, i.e. *refined upper quad-tree constraints* ($\text{RCDM}_{\overline{\Delta}}$) and *lower quad-tree constraints* ($\text{CDM}_{\underline{\Delta}}$), are predicted from the variance of its samples before starting any encoding computation. Then, the HEVC encoder is forced to only apply the RDO process between the two CDM.

5.5.1 Overall Algorithm Scheme

Figure 5.11 presents a high-level diagram of our overall algorithm scheme. Firstly, the video sequence is split into Group of Frames (GOF).

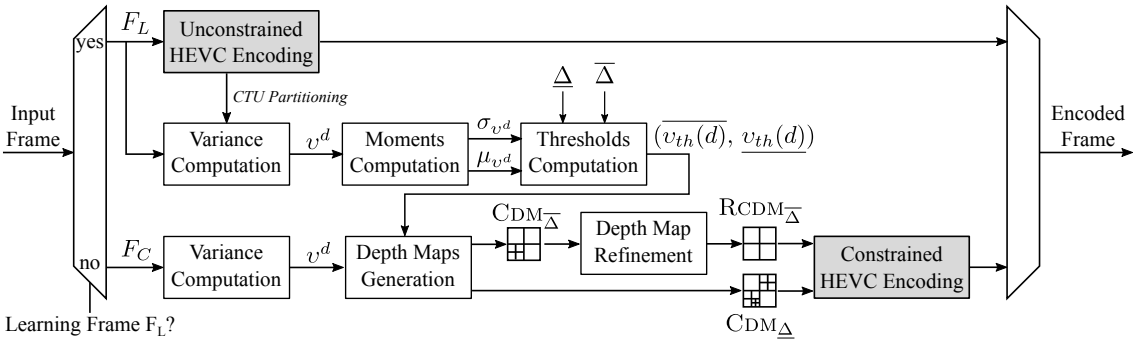


Figure 5.11 – *Diagram of the proposed method. The F_L are encoded with a full RDO process (unconstrained) and the block variances of the resulting quad-tree decomposition is used to compute the set of thresholds $\underline{\upsilon_{th}(d)}$ and $\overline{\upsilon_{th}(d)}$. The constrained frames (F_C) use these thresholds to generate two CDM and constrain the encoder to only apply the RDO process in the interval formed by the two CDM.*

The first frame of each GOF, called Learning Frame (F_L) is encoded with a full RDO process. From this encoding are extracted the variances υ^d for each depth $d \in \{1, 2, 3, 4\}$ selected during the full RDO process. Then, the two following statistical moments according the depth d are computed: the means μ_{υ^d} and the standard deviations σ_{υ^d} of the variance populations υ^d. Based on two parameters $\underline{\Delta}$ and $\overline{\Delta}$, two sets of thresholds $\underline{\upsilon_{th}(d)}$ and $\overline{\upsilon_{th}(d)}$ are calculated using Equation (5.1) and the LUT of the coefficients $a(\Delta, d)$, $b(\Delta, d)$ and $c(\Delta, d)$ computed off-line (cf. Section 5.2.2).

The other frames of the GOF, called *constrained frames* (F_C), are encoded with a limited RDO process. For each CTU, Algorithm (5.1) is applied twice using the two sets of thresholds previously computed from frame F_L. Algorithm (5.1) takes as input the set of thresholds $\underline{\upsilon_{th}(d)}$ the first time and $\overline{\upsilon_{th}(d)}$ the second one to generate respectively the $\text{CDM}_{\underline{\Delta}}$ and $\text{CDM}_{\overline{\Delta}}$. To finish, the $\text{CDM}_{\overline{\Delta}}$, being the *upper quad-tree constraint*, is refined by Algorithm (5.2) to build $\text{RCDM}_{\overline{\Delta}}$. This refinement process gives more slack to the constrained depth decision that will test coarser grain units.

To conclude this section, our proposed energy reduction scheme takes as input two parameters $\overline{\Delta}$ and $\underline{\Delta}$ with $\overline{\Delta} < \underline{\Delta}$ to generate the two CDM: $\text{CDM}_{\overline{\Delta}}$ and $\text{CDM}_{\underline{\Delta}}$ predicted from the variance of its samples. Then, the $\text{CDM}_{\overline{\Delta}}$ is refined to generate the $\text{RCDM}_{\overline{\Delta}}$.

The CDM$_{\underline{\Delta}}$ and RCDM$_{\overline{\Delta}}$ constitute a smart interval of quad-tree partitioning. The HEVC encoder is then forced to only apply the RDO process in the interval between CDM$_{\underline{\Delta}}$ and RCDM$_{\overline{\Delta}}$.

5.5.2 Experimental Setup and Results

The following section gives the experimental metrics and parameters and then the results obtained for the proposed energy reduction scheme on a real time HEVC encoder. The experimental set-up has been detailed in Section 4.3.1.

5.5.2.1 Experimental Metrics and Parameters

The performance of the proposed energy reduction scheme is evaluated by measuring the trade-off between Energy Reduction (ER) in % and RD efficiency using the BD-BR and BD-PSNR. ER is defined by Equation (5.9):

$$ER = \frac{100}{4} \sum_{QP_i \in \{22,27,32,37\}} \frac{E_{Ref}(QP_i) - E_{red}(QP_i)}{E_{Ref}(QP_i)} \tag{5.9}$$

where $E_{Ref}(QP_i)$ is the energy spent to encode the video sequence without constraint and $E_{red}(QP_i)$ the energy to encode the same sequence with our proposed energy reduction scheme, both with $QP = QP_i$.

Table 5.3 – *Configuration of* $(\underline{\Delta}, \overline{\Delta})$

Configuration	$(\underline{\Delta}, \overline{\Delta})$
C1	(0.90,0.30)
C2	(0.85,0.35)
C3	(0.80,0.40)
C4	(0.75,0.45)
C5	(0.70,0.50)
C6	(0.65,0.55)
C7	(0.60,0.60)

Table 5.3 summarizes the configurations of $(\underline{\Delta}, \overline{\Delta})$ used to evaluate the performance of the proposed energy reduction scheme. We vary the couple of parameters $(\underline{\Delta}, \overline{\Delta})$, starting from the base value $(\underline{\Delta}, \overline{\Delta}) = (0.60, 0.60)$ (C7 configuration), corresponding approximately to the crossing point of curves in Figure 5.9. As the error is close to symmetric, we keep the sum $\underline{\Delta} + \overline{\Delta}$ constant until $(\underline{\Delta}, \overline{\Delta}) = (0.30, 0.90)$ (C1 configuration), with incremental steps of 0.05. Based on previous study [MBP+17a], the size of the GOF is set to 50.

As detailed in Section 5.5, the proposed reduction scheme is composed by two distinct algorithms: CDM *Generator* (Algorithm (5.1)) and CDM *Refinement* (Algorithm (5.2)). To evaluate the gain of the second one, each encoding was carried out with and without applying the CDM *Refinement* algorithm. In other words, the HEVC encoder is forced to only apply the RDO process between CDM$_{\underline{\Delta}}$ and CDM$_{\overline{\Delta}}$ (respectively RCDM$_{\overline{\Delta}}$) when the scheme does not apply (respectively apply) the CDM *Refinement* algorithm (see Figure 5.11).

5.5.2.2 Experimental Results

Figures 5.12 and 5.13 show the results in terms of BD-BR and BD-PSNR versus ER of the configurations detailed in Table 5.3 when the CDM *Refinement* algorithm is and is not

applied. The energy values include the energy overhead due to the entire energy reduction scheme as the variance computation. The results of Figures 5.12 and 5.13 are the average of the results obtained when applying our proposed energy reduction scheme on the 18 video sequences [Bos13]. Moreover, Table 5.4 details the results for the 18 different sequences belonging to 5 classes (A, B, C, D and E) for configurations C7 and C1 (when the RCDM algorithm is applied).

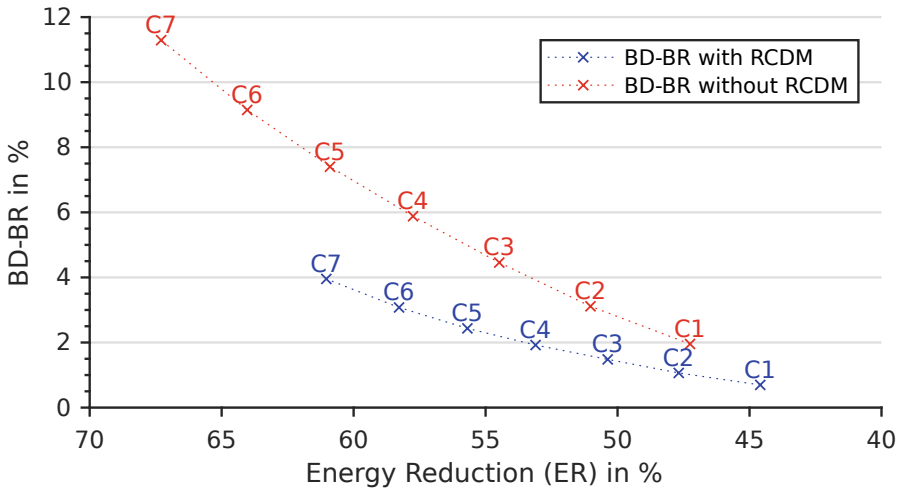


Figure 5.12 – *BD-BR and ER versus* $(\underline{\Delta}, \overline{\Delta})$ *configurations summarized in Table 5.3.*

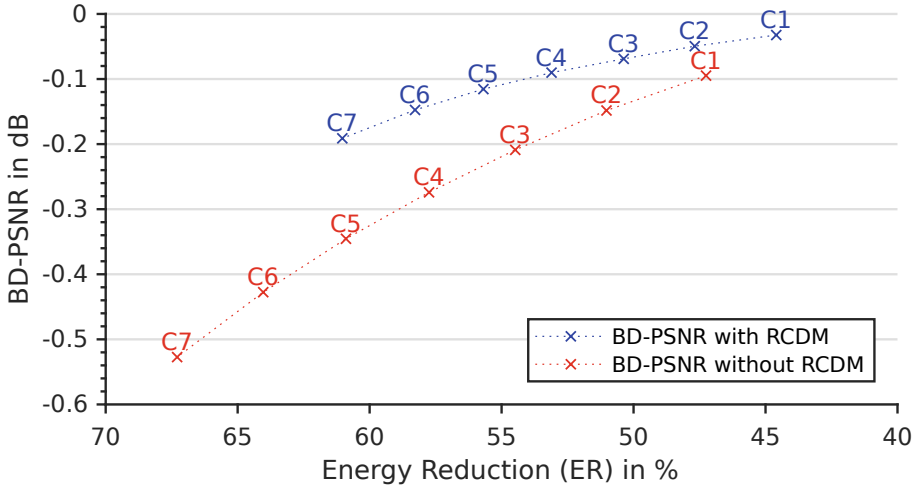


Figure 5.13 – *BD-Psnr and ER versus* $(\underline{\Delta}, \overline{\Delta})$ *configurations summarized in Table 5.3.*

First of all, the results of Figures 5.12 and 5.13 show that applying the Cdm *Refinement* algorithm consumes between 2.7% (C1) and 6.3% (C7) more energy with not negligible gains of encoding performance. The gains obtained by applying the Cdm *Refinement* algorithm are between:

- -1.3% (C1) and -7.33% (C7) in terms of BD-BR,
- +0.06 dB (C1) and +0.34 dB (C7) in terms of BD-Psnr.

Table 5.4 – *BD-BR, BD-PSNR and ER of the proposed energy reduction scheme according to the sequences for configurations C7 and C1*

				C7 configuration			C1 configuration		
Class	**Sequence name**	**Resolution**	**Nb Frame**	**BD-BR** (in %)	**BD-PSNR** (in dB)	**ER** (in %)	**BD-BR** (in %)	**BD-PSNR** (in dB)	**ER** (in %)
A	*Traffic*	2560x1600	150	4.59	-0.24	63.11	0.97	-0.05	45.58
A	*PeopleOnStreet*	2560x1600	150	4.28	-0.24	62.12	0.63	-0.04	47.47
B	*Kimono*	1920x1080	240	13.28	-0.43	55.66	4.17	-0.14	37.77
B	*ParkScene*	1920x1080	240	4.29	-0.19	58.79	0.92	-0.04	40.67
B	*BasketballDrive*	1920x1080	500	3.71	-0.11	63.53	0.52	-0.02	45.71
B	*Cactus*	1920x1080	500	3.56	-0.13	63.19	0.46	-0.02	45.03
B	*BQTerrace*	1920x1080	600	2.26	-0.15	63.76	0.16	-0.03	48.66
C	*RaceHorses*	832x480	300	3.12	-0.18	62.60	0.42	-0.02	44.47
C	*PartyScene*	832x480	500	1.88	-0.13	61.78	0.12	-0.01	47.42
C	*BasketballDrill*	832x480	500	2.74	-0.13	61.65	0.04	-0.00	42.96
C	*BQMall*	832x480	600	3.46	-0.19	60.98	0.44	-0.02	45.55
D	*RaceHorses*	416x240	300	2.47	-0.15	61.75	0.14	-0.01	40.88
D	*BQSquare*	416x240	600	2.74	-0.21	60.62	0.38	-0.03	50.44
D	*BlowingBubbles*	416x240	500	1.45	-0.10	58.12	0.03	-0.00	41.59
D	*BasketballPass*	416x240	500	2.39	-0.14	62.42	0.15	-0.01	43.85
E	*FourPeople*	1280x720	600	4.79	-0.27	59.57	0.99	-0.06	45.71
E	*Johnny*	1280x720	600	6.02	-0.24	59.24	1.37	-0.06	41.68
E	*KristenAndSara*	1280x720	600	4.15	-0.21	59.73	0.62	-0.03	47.25
			Average	**3.95**	**-0.19**	**61.03**	**0.70**	**-0.03**	**44.59**

These results confirm the interest of the CDM *Refinement* algorithm, thus the rest of this section is focused only on the results with the CDM *Refinement* algorithm.

The results show that our energy reduction scheme is able to reduce the energy consumption of the HEVC encoder from 45% (C1) to 61% (C7). In configuration C7, an average of 61% of energy reduction is achieved with a 3.95% BD-BR increase and a quality degradation of -0.19dB BD-PSNR. Whereas in configuration C1, an average of 45% of energy reduction is achieved with a slight BD-BR increase of 0.69% and a quality degradation of -0.03dB BD-PSNR

Table 5.4 shows that for the two configuration C7 and C1, the ER has a range lower than $\approx$ 10% across the sequences with a maximum of 63.76% in *BQTerrace* and a minimum of 55.65% in *Kimono* for configuration C7 for example. The variations of ER for all configurations (from C1 to C7), are due to the number of blocks merged to build the $\text{RCDM}_{\overline{\Delta}}$, which depends on the prediction of the $\text{CDM}_{\overline{\Delta}}$. This decomposition depends on the texture characteristics of the sequence. This instability is especially significant for configuration C7 because this configuration has the most severe constraint, i.e. the $\text{CDM}_{\underline{\Delta}}$ and $\text{CDM}_{\overline{\Delta}}$ are the same (generated by the same Δ value: 0.6). Only the CDM *Refinement* algorithm creates a small unpredictable interval between the two CDM.

In the other hand, it can be noticed that classes C and D for the C7 and C1 configurations have less degradation in terms of BD-BR and BD-PSNR than other classes. The same results are observed for all configurations (from C1 to C7). It can be explained by the condition line 8 of Algorithm (5.1) and the small resolution of these two classes which tend to have a finer-grained splitting during the RDO process. Indeed, Algorithm (5.1) does not try to merge the block at the current depth if the four neighbor blocks in the Z-scan order do not have the same depth that tends to predict more finer-grained splitting. It is also noticeable in Table 5.4 than the Kimono sequence has more degradation than the other sequences: 13.28% of BD-BR increasing. This can be explained by the texture specificity of the Kimono video sequence which is composed by a traveling of trees and vegetation in the background.

5.5.2.3 Comparison with Related Works

Since the goal of this work is to predict CTU quad-tree partitioning in one-shot (before starting the RDO process), the performance comparison of our proposed energy reduction scheme is focus on predicted quad-tree partitioning reduction techniques detailed in Section 3.2.1.2.

Figure 5.14 presents a comparison between our proposed energy reduction technique and techniques presented in Section 3.2.1.2. Results from techniques implemented of HM Khan[KSH13], Ruiz14[RCAFE+14], Ruiz15[RFEA+15], Duanmu[DMW15], Min[BC15] and Feng[FDZX16] are plotted in red. We reimplemented in the real-time software encoder Kvazaar two complexity reduction techniques extracted from Shen*[SZA13] and Zhang*[ZLL15] plotted in blue in Figure 5.14. Crosses and circles correspond respectively to results in terms of energy and complexity reductions.

This comparison is unfair to our method in terms of time/energy, as the complexity overhead of the proposed complexity reduction techniques in literature would be higher in the context of a real time encoder, thus reducing their complexity reduction. In a real-time configuration (see 4.3.1), the computational overhead of our proposed method in the real-time encoder Kvazaar is between 1% and 1.9%.

Figure 5.14 shows that, even with an unfair comparison, the energy reduction technique proposed in this paper outperforms state-of-the-art both in terms of average encoding

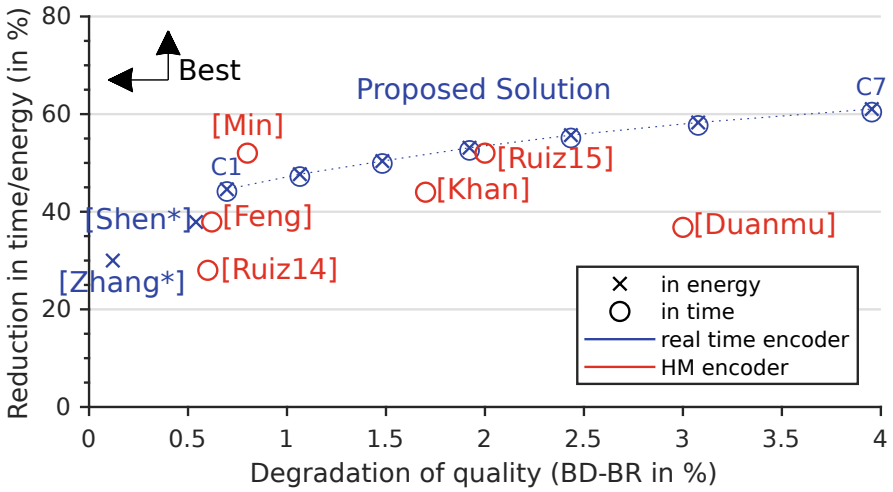


Figure 5.14 – *Comparison with related works in terms of time/energy reduction (in %) and quality degradation (BD-BR in %).*

energy/time reduction and compression efficiency across the different configurations (from C1 to C7). In configuration C1 for example, our method reduces the energy consumption by up to 7% more when compared to Shen*[SZA13] for the same bit-rate degradation of $\approx$ 0.5% of BD-BR. Regarding Zhang*[ZLL15], our results show that the complexity reduction technique reduces the energy consumption of 30% in average for a BD-BR increase of 0.12%. An energy reduction of 30% cannot be obtained with our current configurations of $(\underline{\Delta}, \overline{\Delta})$. Only Min[BC15] has better results than our configurations. This technique uses 8 filtered images to compute their local and global edge complexities and predict the partitioning of CU. We reimplemented in the real-time software encoder Kvazaar only the filtering to estimate the overhead of this complexity reduction technique. We do not have optimized all the computation but the results show that the overhead of the filtering is between 20% and 30% for the real-time configuration describe in Section 4.3.1 depending on the QP value. This overhead would lead to a decrease of the energy/time reduction under our results.

5.6 Conclusion

This Chapter has presented an energy reduction scheme for HEVC encoders based on a "one-shot" CTU partitioning prediction technique that limits the recursive RDO process. The proposed energy reduction technique exploits the correlation between a CTU partitioning, its texture complexity and its luminance samples variance. The technique explores trade-offs between energy reduction and compression efficiency. Experimental results show that the proposed algorithm is able to reduce the energy consumption of the HEVC encoder by 45% to 61% — including the algorithm overhead — for a bit rate increase ranging between 0.69% and 3.95% respectively.

As explained in the conclusion of the previous chapter (Section 4.4), the main objective of this chapter is to design the *Coarse Solution Predictor* defined in the SSSR methodology in Section 4.2.2.2 for the MSSE corresponding to the CTU level. A challenge of the *Coarse Solution Predictor Design* step (see 4.2.2.2.3) is to design a predictor model with a moderate computation complexity overhead and able to provide a solution as close as possible to the optimal solution. With a computational overhead in a real-time framework of between

1% and 1.9% and the performance results demonstrated in Section 5.5.2.2, the solution presented in this chapter appears to match with the objective.

However, the *Approximation Management* step requires approximation parameters Γ to define how large the search space is from the coarse estimation, corresponding in our case to the number of explored depths per CTU. The two parameters proposed in the solution of this Chapter, namely $(\underline{\Delta}, \overline{\Delta})$, offer a trade-off between energetic gains and compression efficiency but do not offer the possibility to explore the entire search space of the MSSE. In other words, the solution proposed in this chapter does not offer the possibility to set the energy consumed by the encoding process from 100% to the Minimal Energy Point (MEP) of the CTU level (21.9%) defined in Section 4.3.3.2 and illustrated in Figure 4.9 page 61.

A solution to this problem is to only use the CDM corresponding to the C7 configuration (for the C7 configuration, $\text{CDM}_{\underline{\Delta}} = \text{CDM}_{\overline{\Delta}}$ because $\underline{\Delta} = \overline{\Delta} = 0.6$) as coarse solution prediction and define an algorithm to search around this CDM in terms of depth levels. Nevertheless, this statistical approach is not optimal because it has to use costly Learning Frame (F_L) to determine the variance thresholds $v_{th}(\Delta, d)$. As a consequence, the *Approximation Management* would not only play with the number of depths explored by CTU but also with the size of the GOF between two learning frames.

The next chapter presents a "one-shot" quad-tree partitioning prediction technique based on *Machine Learning (ML)* that removes the need of learning frames. The quad-tree prediction is then used to drastically limit the energetic cost of the RDO process.

CHAPTER 6

Quad-Tree Prediction Based on Machine Learning Approach

6.1 Introduction

As the study presented in Section 4.3 of Chapter 4 has shown, and as the results of the quad-tree partitioning prediction technique based on variance studies proposed in Chapter 5 have confirmed, quad-tree partitioning exploration offers the biggest opportunity of complexity/energy reduction.

According to the SSSR methodology presented in Section 4.2 of Chapter 4, the method presented in Chapter 5 has two main limitations. First, when the energy reduction algorithm scheme (Section 5.5 and Figure 5.11) is used, the window of achievable energy reduction values do not cover the entire possible reduction range: until 78% of energy reduction (see Section 4.3.3.2). Second, the use of Learning Frame (F_L) makes the *Approximation Management* step of the SSSR methodology more difficult. To address these specific issues, we choose to limit our *Coarse Solution Predictor* to a "one-shot" quad-tree partitioning prediction and then design an algorithm for exploring the search space around this prediction in terms of depth levels.

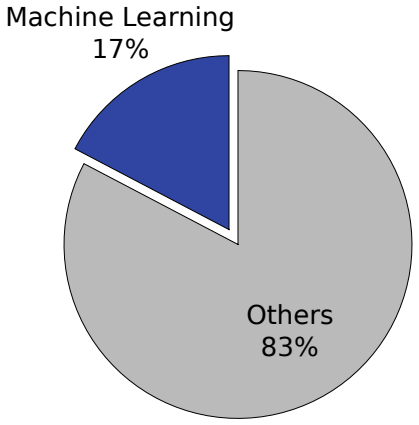


Figure 6.1 – *Percentage of complexity reduction techniques using ML tools in the state-of-the-art presented in Chapter 3.*

As shown in Chapter 3, especially in Section 3.2.1, several works have been proposed to reduce the complexity of the encoding process using ML-based HEVC optimizations. ML is an interdisciplinary subfield of computer science that aims to replace the manually engineered solutions for extracting structured information from sensored data, and this in all application fields. Figure 6.1 show the share of complexity reduction techniques using ML tools, representing 17%, in the state-of-the-art presented in Chapter 3.

This chapter presents a new "one-shot" prediction of quad-tree partitioning for the HEVC Intra encoders. For a given CTU, this technique predicts a CTU Depth Map (CDM) before starting the RDO process and without using an on-line learning phase. The prediction is based on an ML approach across data-mining classifiers. Compared to state-of-the-art solutions, the originality of the presented work comes from its one-shot prediction of quad-tree partitioning before starting the RDO process. This quad-tree partitioning prediction is then used to drastically reduce the energy consumption of the encoding process under a real-time framework.

The rest of this chapter is organized as follows. Section 6.2 compares features, using an ML approach, for their capacity to feed a prediction of quad-tree partitioning under a real-time framework. Section 6.3 details the ML classifiers, which are then used in Section 6.4 to predict the quad-tree. Section 6.5 extends the analysis of previous sections to study the pros/cons of using co-located depths as features. The one-shot quad-tree prediction is used in Section 6.6 to drastically reduce the energy consumption of the encoding process for reasonable coding efficiency losses. Section 6.7 compares the performance of the two proposed energy reduction schemes: the statistical approach presented in Chapter 5 and the Machine Learning approach detailed in this chapter. Finally, Section 6.8 concludes this chapter.

6.2 Evaluating Characteristics for their Capacity to Determine an Effective Coding Tree Structure

The goal of this section is to perform an exhaustive comparison of characteristics extracted from state-of-the-art and used to predict a quad-tree decomposition of a CTU before starting the encoding process. By characteristic, we mean a metric value available at the encoder side, computed from current frame or previously encoded frames, that can help determining the correct block sizes in the coding tree structure. The study is based on a Machine Learning approach under a real-time framework and the coding tree decomposition is first transformed into classification problems.

6.2.1 Coding Tree Structure Determination Expressed as Classification Problems

As detailed in Chapter 3, the main approach to reduce the complexity of a quad-tree decomposition is to determine the best partitioning of a given CU between $2N \times 2N$ pixels and $N \times N$ pixels sub-blocks. To cover both Bottom-up (grouping blocks) and Top-Down (splitting blocks) approaches, the classification problem is composed of two different predictions at each depth d:

- the *Merge* prediction which predicts whether the CU of the depth d has to be merged in CU of the depth $d-1$, as illustrated by Figure 6.2,
- the *Split* prediction which predicts if a CU of the depth d has to be split into 4 CU of depth $d+1$ as illustrated by Figure 6.3.

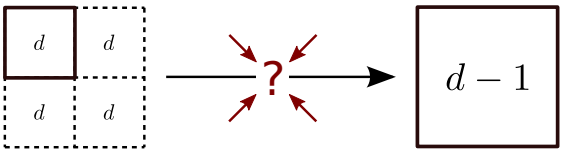


Figure 6.2 – *Merge prediction of a CU of depth d (belonging to a group of 4 neighboring CU in the Z-scan order) into a CU of depth d − 1.*

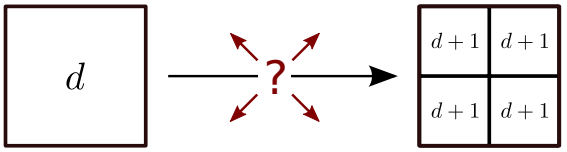


Figure 6.3 – *Split prediction of a CU at depth d into 4 CU at depth d + 1.*

The next section details the training setup used for the characteristics evaluation of the two previous classification problems.

6.2.2 Training Set-Up for the Coding Tree Structure Determination

Machine Learning efficiency is highly correlated with the diversity and relevance of the training data set. Video sequences used to train the Machine Learning framework are chosen to cover a vast space of content types. To select this training data set including a large range of video contents and complexities, the Spatial Information (SI) and Temporal Information (TI) metrics [ITU99] are used to characterize video sequences. The TI and SI give respectively the amount of movement and spatial details in the video sequence. Since compression complexity is highly linked to these SI and TI metrics, the set of training sequences for the Machine Learning feature evaluation should span a large range of both SI and TI.

Figure 6.4 shows the SI and TI for the video sequences according to the classes (from A to E). The chosen training sequence set (circled in Figure 6.4) is composed of one video sequence of each class, well distributed in terms of SI and TI: *Traffic* (class A), *Cactus* (class B), *BQMall* (class C), *BasketballPass* (class D) and *Johnny* (class E).

Overfitting, i.e. overspecializing a model to a training set, constitutes one of the main risks for the quality of a Machine Learning-based models [Dom12]. Thus, the data set used for training should result in a low bias. In our case, due to the broad range of resolutions and frame rates across the training sequences, the total number of CTU for each class is not uniformly distributed. For instance, sequences with high resolution are likely to contain a high number of CTU with low texture complexities when compared to sequences with low resolution. To avoid such bias, data sets used for training are forced to be composed of a fixed number of CTU from each class. To avoid the temporal bias, which would lead to redundant information, the sampled CTU comes from frames uniformly distributed throughout the sequences: 13 frames of the class A, 25 frames of class B, 55 frames for class E, 125 frames of class C and 500 frames of class D. For each depth d, 80000 instances are randomly sampled from the previous defined data pool and composed of 40000 instances of each prediction decision. The instances are uniformly extracted from 4 QP values: 22, 27, 32 and 37.

The open source Waikato Environment for Knowledge Analysis (WEKA) Machine Learning framework is used for experiments [HFH+09]. WEKA includes a large number of

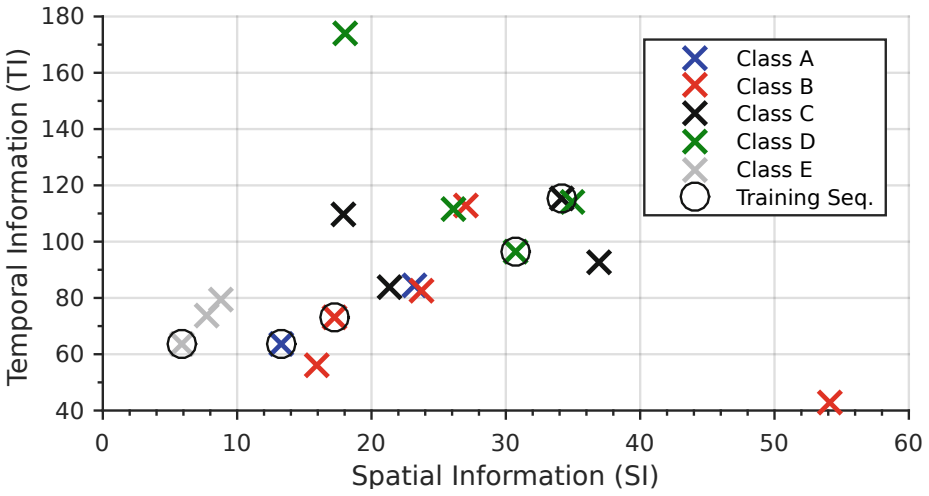


Figure 6.4 – *Spatial Information (SI) and Temporal Information (TI) of the video sequences according to the classes.*

Machine Learning algorithms based on decision tree for data mining tasks, such as *REPTree*, *LMT*, *RandomForest*, *BFTree* and *C4.5* among others. WEKA also provides several useful tools for features evaluation that use strategies according to a search algorithm to rank the features depending on their usefulness. For the current study, the characteristic evaluation is done using the *information gain* provided by the WEKA software. The *information gain* is based on the Kullback-Leibler Divergence (KLD) [KL51], also called relative entropy, which measures the divergence between two probability distributions.

6.2.3 Characteristics Information Gain Evaluation

State-of-the-art studies described Chapter 3 gather many characteristics used to predict the coding tree decomposition of a CTU. As we plan to predict the coding tree in one-shot, this work is only focused on characteristics independent of the encoding process, i.e. the characteristics extracted from the prediction-based complexity reduction techniques detailed in Section 3.2.1.2. The compared characteristics are listed below and the publications mentioning them are provided.

- **CU var** [MAP+17, KSH13, PCL16, RFEA+15, RCAFE+14, DMW15] : the variance of the CU luminance samples of depth d (1 characteristic).
- **Lower-CU var** [MAP+17, KSH13, RFEA+15, RCAFE+14, DMW15]: the variances of the 4 sub-CU luminance samples of depth $d+1$ (4 characteristics).
- **Upper-CU var** [MAP+17, KSH13, PCL16, RFEA+15, RCAFE+14]: the variance of the upper CU luminance samples of depth $d-1$ (1 characteristic).
- **Nhbr-CU var** [KSH13, DMW15]: the variances of the neighboring CU luminance samples of the depth d in the Z-scan order (3 characteristics).
- **Local cplx V, H, 45, 135** [BC15, SY13, DMW15]: respectively the horizontal, vertical, 45° diagonal and 135° diagonal local edge complexity of the CU luminance samples of the depth d as described in [BC15]; they are obtained by the SAD between the sample values and the mean of the sample values after filtering by the corresponding oriented gradient filter (4 characteristics).

- **Global cplx V, H, 45, 135** [BC15]: respectively the horizontal, vertical, 45° diagonal and 135° diagonal global edge complexity of the CU luminance samples of the depth d as describe in [BC15] (4 characteristics).
- **Var of lower-CU mean** [RFEA+15, RCAFE+14]: the variance of the mean of the 4 sub-CU luminance samples of the depth $d+1$ (1 characteristic).
- **Var of lower-CU var** [RFEA+15, RCAFE+14]: the variance of the variance of the 4 sub-CU luminance samples of the depth $d+1$ (1 characteristic).
- **Kurtosis V, H, 45, 135** [DMW15]: the kurtosis of the local *cplx V, H, 45, 135* characteristics [DMW15] (4 characteristics).
- **Kurtosis of samples** [DMW15]: the kurtosis of the CU luminance samples of the depth d [DMW15] (1 characteristic).
- **Prev-CU depth** [SZA13, ZLL15, CNP12]: the mean of the CU depth d of the previous frame (1 characteristic).
- **Prev-CTU depth** [SZA13, ZLL15, CNP12]: the mean of the CTU depth including the CU of the depth d of the previous frame (1 characteristic).
- **Above-CTU depth** [SZA13, ZLL15, CNP12]: the mean of the CTU depth of the above CTU (1 characteristic).
- **Left-CTU depth** [SZA13, ZLL15, CNP12]: the mean of the CTU depth of the left CTU (1 characteristic).

Tables 6.1 and 6.2 show the information gain of the evaluated characteristics for receptively the *Merge* and *Split* predictions according to the depth d. For both Tables 6.1 and 6.2, the characteristics are ranked according to the average (Av.) of information gain across the depths. Information gain values are presented separately for each depth d in the two Tables. Some characteristics (as *Lower-CU var* or *Local cplx V, H, 45, 135* for example) are grouped and the mean of information gain of the group is used for the evaluation.

Table 6.1 – *Information gain of the characteristics for Merge decisions according to depth d.*

Characteristics	$d=4$	$d=3$	$d=2$	$d=1$	**Av.**
Prev-CU depth	0.35	0.41	0.48	0.48	0.43
Prev-CTU depth	0.24	0.29	0.40	0.56	0.37
Upper-CU var	0.34	0.32	0.30	0.33	0.32
Var of lower-CU var	-	0.19	0.21	0.27	0.22
CU var	0.19	0.17	0.19	0.24	0.20
Nhbr-CU var	0.18	0.17	0.19	0.24	0.20
Above-CTU depth	0.11	0.14	0.19	0.27	0.18
Var of lower-CU mean	-	0.14	0.16	0.18	0.16
Local cplx V, H, 45, 135	0.19	0.14	0.14	0.17	0.16
Global cplx V, H, 45, 135	0.12	0.12	0.14	0.19	0.14
Left-CTU depth	0.09	0.11	0.15	0.20	0.14
Lower-CTU var	-	0.10	0.12	0.18	0.13
Kurtosis V, H, 45, 135	0.02	0.05	0.13	0.26	0.11
Kurtosis of samples	0.02	0.06	0.08	0.13	0.07

There is no absolute threshold to evaluate a characteristic information gain in the literature but information gain can be used to evaluate a characteristic w.r.t the others. In the experimental data, the characteristics that have an information gain larger than

Table 6.2 – *Information gain of the characteristics for Split decisions according to depth d.*

Characteristics	$d=3$	$d=2$	$d=1$	$d=0$	Av.
Prev-CU depth	0.36	0.42	0.51	0.56	0.46
Prev-CTU depth	0.24	0.29	0.41	0.56	0.38
Var of lower-CU var	0.39	0.38	0.36	0.35	0.37
CU Var	0.35	0.32	0.31	0.32	0.32
Upper-CU var	0.30	0.26	0.23	-	0.27
Var of lower-CU mean	0.29	0.25	0.24	0.24	0.25
Local cplx V, H, 45, 135	0.31	0.26	0.21	0.21	0.25
Global cplx V, H, 45, 135	0.25	0.23	0.23	0.24	0.24
Kurtosis V, H, 45, 135	0.05	0.16	0.30	0.37	0.22
Lower-CTU var	0.18	0.17	0.20	0.25	0.20
Above-CTU depth	0.12	0.14	0.19	0.26	0.18
Nhbr-CU var	0.14	0.14	0.17	-	0.15
Left-CTU depth	0.09	0.11	0.15	0.21	0.14
Kurtosis of samples	0.07	0.09	0.12	0.13	0.10

0.3 (representing 26% of all characteristics) constitute a group of particularly relevant characteristics. The well represented category of information gains comprised between 0.1 and 0.3 (representing 64% of all characteristics) can be considered as average-performing solutions. Finally, the characteristics with information gain less than 0.1 (representing 10% of all characteristics) can be considered as low-performing.

First of all, Table 6.1 and 6.2 show that each characteristic extracted from state-of-the-art can be considered relevant to both Merge and Split predictions for at least one of the four depths of decision tree prediction. However, information gain of some characteristics heavily depend on the value of d. For instance, *Kurtosis V, H, 45, 135* has good information gain only for depth $d = 1$ and depth $d = 0$ for the Split prediction. Results show that the most relevant characteristics for both Merge and Split prediction are *Prev CU depth* and *Prev CTU depth* which are respectively the CU and CTU mean of the depth at the previous frame. Indeed, according to the frame rate, the quad-tree decomposition of consecutive frames are highly correlated due to the relative temporal stability of the video content. On the other hand, *Kurtosis* characteristics can be considered not relevant for the classification problems.

The performance of complexity reduction techniques advertised by state-of-the-art publications are based on the non real-time HM encoder. Computational gains are measured comparatively to a very large, unoptimized compression time. The complexity overhead of state-of-the-art solutions for computing characteristics is thus comparatively higher in the context of a real-time encoder than the published numbers. Next section discusses the previously covered characteristics in terms of computational overhead for a real-time encoder.

6.2.4 Selection of the Most Relevant Characteristics in Terms of the Complexity-Accuracy Trade-off

The computation of characteristics is not part of the native encoding process. This computation results in an overhead that should be fairly less than the complexity reduction offered by one-shot coding tree decisions. In our experiments, the overheads are computed on one sequence of each class corresponding to a specific resolution: *PeopleOnStreet* (class A: 2560×1600), *ParkScene* (class B, 1920×1080), *PartyScene* (class C, 832×480), *Race-*

Horses (class D, 416×240), *FourPeople* (class E, 1280×720). The experimental set-up has been defined in Section 4.3.1. The overhead is measured from the difference of computation time of the full encoding process with and without the computations of each characteristic for $QP \in \{22, 27, 32, 37\}$. Table 6.3 summarizes the results.

Table 6.3 – *Average computational complexity overheads of characteristics computation.*

Characteristic or group of characteristics	Computational Overhead
Var of lower-CU mean	$\approx 1.0\%$
Prev-CU depth / Prev-CTU depth Above-CTU depth / Left-CTU depth	$\approx 1.0\%$
CU var / Lower-CU var Upper-CU var / Nhbr-CU var	$\approx 2.5\%$
Var of lower-CU var	$\approx 2.5\%$
Kurtosis of samples	$\approx 4\%$
Local cplx V, H, 45, 135	$\approx 30\%$
Global cplx V, H, 45, 135	$\approx 30\%$
Kurtosis V, H, 45, 135	$\approx 40\%$

As shown in Section 4.3.2.3 , the computational complexity decreases as QP increases. However, the complexity of the characteristics computation only depends on the raw data of the video sequence and thus is stable across QP values. Table 6.3 shows a global average of computational overheads to evaluate the suitability of characteristics. The overheads are considered in their most optimistic case, i.e. they are measured along with a full encoding process, i.e. without any complexity reduction applied to the encoder.

Table 6.3 shows that the overhead of *Local cplx V, H, 35, 145*, *Global cplx V, H, 35, 145* and *Kurtosis V, H, 35, 145* characteristics are up to 40% of the encoding time. Indeed, these 3 groups of characteristics require multiple filtering processes of all frame samples. These overheads considerably reduce the potential computation gains of these characteristics in a real-time encoder. On the other hand, results show that other characteristics have an acceptable overhead, around 4% for *Kurtosis* and less than 2.5% for the others.

To conclude, Section 6.2.3 shows that all characteristics extracted from state-of-the-art are relevant to predict the quad-tree decomposition of a CTU for some depth levels. However, within a real-time encoder, characteristics that require filtering are not suitable to reduce the computational complexity of the encoding process due to their high computation overhead. Considering both factors: information gain and computational overhead, characteristics extracted from depth and variance information are the most relevant to be used to predict the quad-tree decomposition of a CTU with a Machine Learning approach.

Moreover, the depth features including *Prev-CU depth*, *Prev-CTU depth*, *Above-CTU depth* and *Left-CTU depth* are highly dependent on the quad-tree partitioning of the neighboring CTU. Therefore, in a context of one-shot quad-tree prediction, these characteristics have to be carefully studied as they can produce a propagation of errors. In Section 6.5, the impact of such characteristics on the quad-tree partitioning prediction is studied. In the rest of the document, we refer as *Prev-CU depth*, *Prev-CTU depth*, *Above-CTU depth* and *Left-CTU depth* as the co-located "depth features".

In the first instance, let the feature vector for CU at coordinates (x, y) and depth d of a given CTU, $\mathcal{F}^d_{x,y}$, be composed of the following 12 features:

- **CU var** (1 feature).
- **Lower-CU var** (4 features).
- **Upper-CU var** (1 feature).
- **Nhbr-CU var** (3 features).
- **Var of lower-CU mean** (1 feature).
- **Var of lower-CU var** (1 feature).
- **QP** (1 feature).

6.3 Decision Trees Analysis for their Capacity to Determine an Effective Coding Tree Structure

A decision tree is a model of a classification algorithm where the root represents the starting point of the decision process, a node represents a test, a branch represents a possible outcome of this test and a each leaf corresponds to a class label. The open source WEKA software includes numerous tree classifiers, such as REPTree, LMT, RandomForest, BFTree and C4.5 among others. In this section, the training of the decision trees is performed with the C4.5 algorithm [Qui14], providing information gain on each feature. As information gain, the C4.5 algorithm uses KLD to select the best features for each decision. The C4.5 algorithm is iterating among all training instances and searches for each feature the threshold that achieves the best classification, i.e. with the highest information gain. Then, the features and their corresponding thresholds are used to divide the training instances into two subsets. To finish, the process is recursively iterated on the two different subset of training instances.

To measure the accuracy of the decision trees, a 10-fold cross-validation is performed on the training instances. The cross-validation technique evaluates a predictive model by partitioning the original instances into a training set to train the model, and a test set to evaluate it. In 10-fold cross-validation, the original instances are randomly split into 10 equal size subsets. Among the 10 subsets of instances, one subset is used as the validation instances for testing the model, and the remaining 9 subsets are used as training instances. The cross-validation process is then repeated 10 times (the folds), with each of the 10 subsets used exactly once as the validation instances. Let the Percentage of Correctly Classify Instances (PCCI) given by the 10-fold cross-validation be the accuracy of the decision trees.

6.3.1 Training of a Decision Tree to Predict the Quad-Tree Partitioning

As explained in Section 6.2.3 and illustrated by Figures 6.2 and 6.3, two problems of classification are defined for each depth d: the *Merge* and *Split* predictions, which are associated respectively with the *Merge* and *Split* decision trees. In practice, the features used for the prediction is the main difference between *Merge* and *Split* decision trees. The *Merge* decision trees use the features linked to a CU of depth d to predict if the 4 CU (in the Z-scan order) of depth d have to be merged in the CU of depth $d-1$. The Split decision trees use the features linked to CU of depth $d-1$ to predict if the current CU of depth $d-1$ has to be split in 4 CU of depth d.

To control the size of decision trees, the WEKA software offers the possibility of setting the minimum number of instances per leaf. For both *Merge* and *Split* decision trees and for each depth d, two decision trees are trained called *Long* and *Short* decision trees. *Long* decision trees are trained with the default minimum number of instances per leaf equal to 2. To reduce the size of trees and generate *Short* decision trees, the minimum number of instances per leaf is set at 1000. Table 6.4 summarizes the trained tree size and number of leaves of the 16 decision trees. The *Short* decision trees are 150 times smaller in average than *Long* decision trees in terms of both number of leaves and size.

Table 6.4 – *Decision Trees dimensions according to the depth d. The Short decision trees are 150 times smaller in average than Long decision trees in terms of both number of leaves and size.*

	Merge Decision Trees				**Split Decision Trees**			
	Long Decision Trees							
Depth	$d=4$	$d=3$	$d=2$	$d=1$	$d=3$	$d=2$	$d=1$	$d=0$
Nb Leaves	2302	2430	2459	1058	1928	2161	2067	1529
Size	4603	4859	4917	2115	3855	4321	4133	3057
	Short Decision Trees							
Depth	$d=4$	$d=3$	$d=2$	$d=1$	$d=3$	$d=2$	$d=1$	$d=0$
Nb Leaves	18	15	10	9	15	13	13	15
Size	35	29	19	17	29	25	25	29

To evaluate the accuracy of the predictions, two PCCI are considered: expected and experimental. Expected PCCI are given by the 10-fold cross-validation provided by the WEKA software in the data set extracted from the learning sequences (as described in Section 6.2.2). Experimental PCCI are computed during the encoding process and correspond to the real accuracy of the decision trees. To compute experimental PCCI, the features computation and the testing decision trees are implemented in the encoder. The encoder applies the full RDO process and compares the quad-tree partitioning resulting from the decision of the different classifiers. Experimental PCCI are generated from all the non-training sequences.

Figure 6.5 and 6.6 show experimental and expected PCCI of the *Long* and *Short* decision trees according to the depth d for respectively the *Merge* and *Split* decision trees.

Results of both Figure 6.5 and 6.6 show that for all decision trees, the expected results are better (above 80% of good predictions) than experimental results (below 80% of good predictions). The results show that the expected PCCI of the *Long* and *Split* trees (in green in Figures 6.5 and 6.6) are higher than the expected PCCI of *Short* trees (in blue Figure 6.5 and 6.6) for all depths d. On the other hand, the experimental PCCI show that the accuracy of the experimental prediction is better with the *Short* trees (in red Figure 6.5 and 6.6) compared to the *Long* trees (in gray Figure 6.5 and 6.6). These results can be explained by an overfitting of the *Long* trees due to a number of instances per leaf too low which makes the *Long* trees too specific to the training set. To conclude, based on the comparison of experimental and expected results for the accuracy of prediction of the decision trees, *Short* decision trees are selected over the *Long* trees in the following.

Let $P_S(\mathcal{F}^d_{x,y})$ and $P_M(\mathcal{F}^d_{x,y})$ respectively be the prediction results of the *Short Split* and *Merge* trees for the feature vector $\mathcal{F}^d_{x,y}$. In the next section, the merging of the *Merge* and *Split* trees is being investigated to increase the prediction accuracy at each depth d.

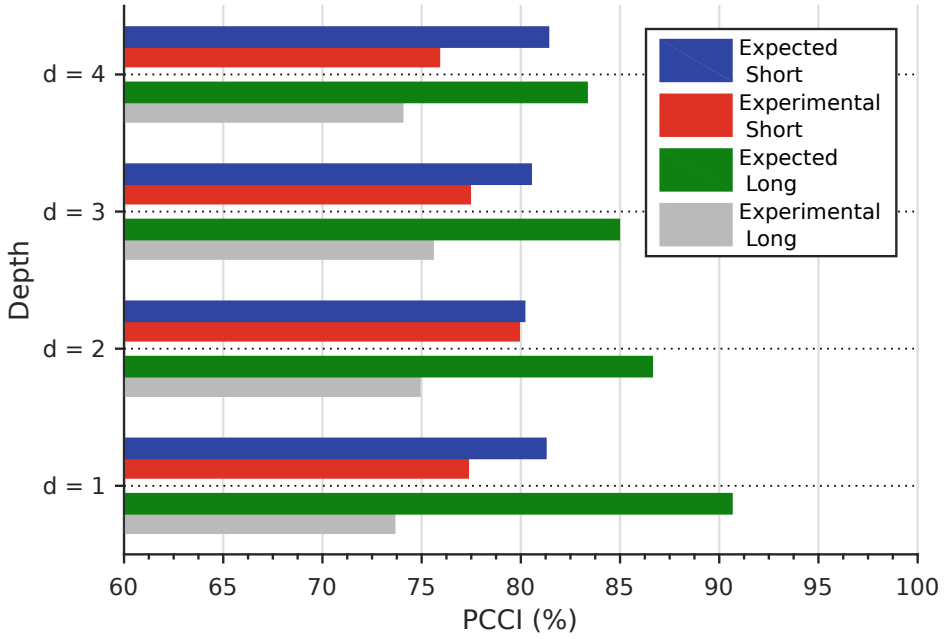


Figure 6.5 – *Expected and experimental PCCI of the Merge decision trees according to the depth d*

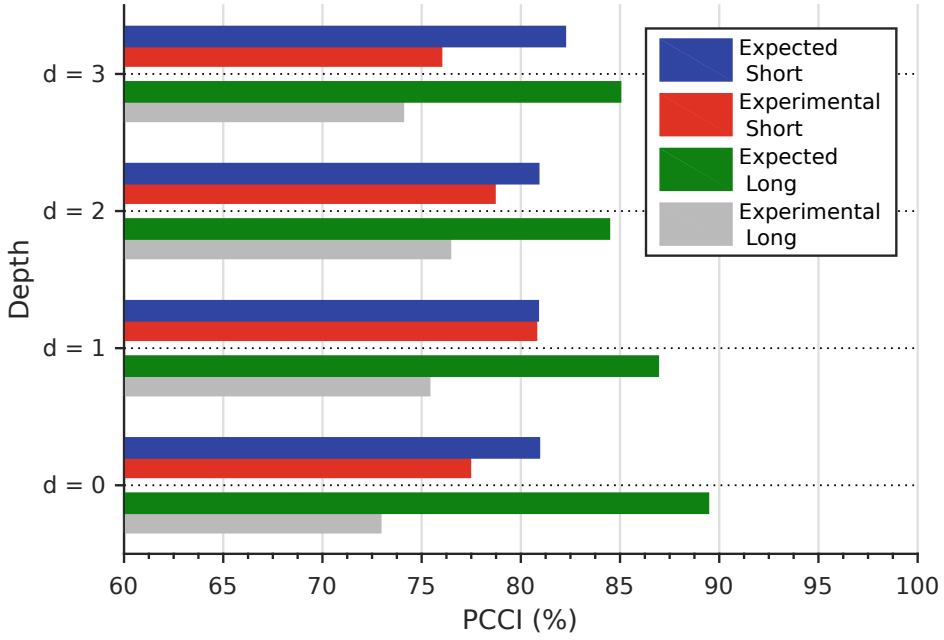


Figure 6.6 – *Expected and experimental PCCI of the Split decision trees according to the depth d*

6.3.2 Combination of Decision Trees to Increase Prediction Accuracy

In the case of a Bottom-up (BU) approach for example, to decide if a group of 4 neighboring CU at the depth d have to be merged in a CU at the depth $d-1$, our Machine Learning approach gives 5 predictions: 4 from the *Merge* tree applied on the 4 neighboring CU of

the depth d and 1 from the *Split* tree applied on the equivalent CU (in terms of samples) at the depth $d-1$.

For a given CTU, $\mathcal{F}^d_{x,y}$ is the feature vector of the CU of size $2^{6-d} \times 2^{6-d}$ at the depth level d and the local coordinates (x, y) into the CTU as illustrated in Figure 6.7.

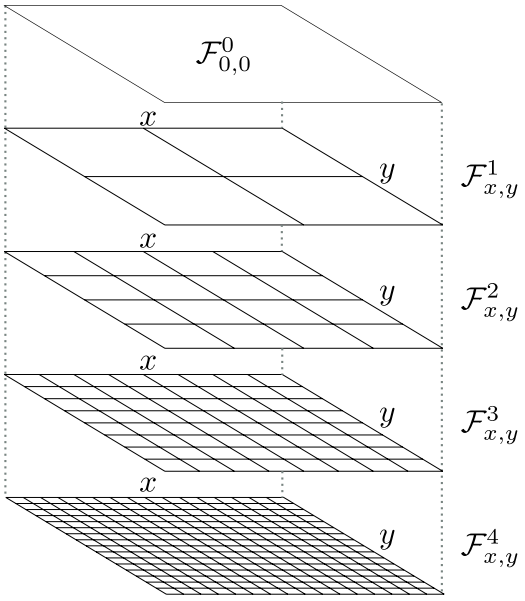


Figure 6.7 – $\mathcal{F}^d_{x,y}$*: feature vector of the CU of size* $2^{6-d} \times 2^{6-d}$ *of a CTU versus the depth level* d*.*

Let $\mathbb{S}_d(x, y)$ and $\mathbb{M}^k_d(x, y)$ be respectively the booleans associated to predictions of the *Split* and *Merge* trees for the CU at the coordinates (x, y) and depth d, defined by the following equations:

$$\mathbb{S}_d(x, y) = P_S(\mathcal{F}^d_{x,y}) \tag{6.1}$$

$$\mathbb{M}^k_d(x, y) = \begin{cases} 0 & \text{if } (\sum_{i=0}^{1}\sum_{j=0}^{1} P_M(\mathcal{F}^d_{x+i,y+i})) < k \\ 1 & \text{otherwise} \end{cases} \tag{6.2}$$

with $P_S(\mathcal{F}^d_{x,y})$ and $P_M(\mathcal{F}^d_{x_i,y_i})$ the prediction results of the *Split* and *Merge* trees for the feature vector $\mathcal{F}^d_{x,y}$. $\mathbb{M}^k_d$ predicts to non merge (when equal to 0) if at least k *Merge* trees of the 4 neighboring CU predict to non-merge.

At each depth d, a total of 13 combinations of trees, described in Table 6.5, are compared to determine the best one. The first four combinations use only the *Merge* decision trees and the fifth only the *Split* one. The next four combinations combine the *Merge* and *Split* decisions by AND ($\wedge$) logical operator. The last four combinations combine the *Merge* and *Split* decisions by OR ($\vee$) logical operator.

Table 6.5 details the experimental PCCI of the decision tree combinations according to the depth d. The results are obtained during the encoding process by applying the full RDO process and by comparing the results of the prediction with the encoder decision. The results show that for the upper depth $d \in \{1, 2\}$, the combination $\mathbb{M}^1_d \vee \overline{\mathbb{S}}_{d-1}$ of decision trees achieve the best results. This combination corresponds to have at least one non-merge prediction across the five prediction trees to decide to not merge. For the lower

Table 6.5 – *Experimental PCCI in % of decision tree combinations according to the depth d. The best results are achieved: for the upper depth $d \in \{1,2\}$ with the combination $\mathbb{M}_d^1 \vee \overline{\mathbb{S}}_{d-1}$, for the lower depth $d \in \{3,4\}$ with the combination $\mathbb{M}_d^4 \wedge \overline{\mathbb{S}}_{d-1}$.*

	$d=1$	$d=2$	$d=3$	$d=4$
$\mathbb{M}_d^1$	87.05%	82.18%	76.12%	73.65%
$\mathbb{M}_d^2$	83.44%	81.06%	77.40%	75.43%
$\mathbb{M}_d^3$	78.48%	79.68%	78.04%	76.00%
$\mathbb{M}_d^4$	72.77%	77.57%	78.94%	76.57%
$\overline{\mathbb{S}}_{d-1}$	79.77%	80.90%	78.76%	74.80%
$\mathbb{M}_d^1 \wedge \overline{\mathbb{S}}_{d-1}$	79.01%	80.80%	79.32%	77.17%
$\mathbb{M}_d^2 \wedge \overline{\mathbb{S}}_{d-1}$	77.22%	79.77%	79.44%	77.75%
$\mathbb{M}_d^3 \wedge \overline{\mathbb{S}}_{d-1}$	74.83%	78.85%	79.56%	77.97%
$\mathbb{M}_d^4 \wedge \overline{\mathbb{S}}_{d-1}$	71.23%	77.15%	**79.70%**	**78.22%**
$\mathbb{M}_d^1 \vee \overline{\mathbb{S}}_{d-1}$	**87.82%**	**82.29%**	75.57%	71.28%
$\mathbb{M}_d^2 \vee \overline{\mathbb{S}}_{d-1}$	85.99%	82.19%	76.72%	72.47%
$\mathbb{M}_d^3 \vee \overline{\mathbb{S}}_{d-1}$	83.42%	81.72%	77.24%	72.84%
$\mathbb{M}_d^4 \vee \overline{\mathbb{S}}_{d-1}$	81.31%	81.31%	78.01%	73.15%

depth $d \in \{3,4\}$, the combination $\mathbb{M}_d^4 \wedge \overline{\mathbb{S}}_{d-1}$ of decision trees achieved the best results. This combination corresponds to having all the decision trees that predict to non-merge to avoid merging.

Let $\mathbb{D}_d(x,y)$ be the prediction resulting of the combination of the decision trees according to the depth d defined as follows:

$$\mathbb{D}_d(x,y) = \begin{cases} \mathbb{M}_d^1(x,y) \vee \overline{\mathbb{S}}_{d-1}(x,y) & \text{if } d \in \{1,2\} \\ \mathbb{M}_d^4(x,y) \wedge \overline{\mathbb{S}}_{d-1}(x,y) & \text{if } d \in \{3,4\} \end{cases} \tag{6.3}$$

Table 6.6 shows a comparison of experimental PCCI between the *Merge* and *Split* trees and the tree combination selected previously. These results highlight the increasing accuracy of the predictions by combining the decision trees for each depth d.

Table 6.6 – *Experimental PCCI comparison between the Short* Merge *and* Split *trees and the tree combination*

	Merge Decision Trees			
Depth	$d=4$	$d=3$	$d=2$	$d=1$
Experimental PCCI	75.89%	77.44%	79.91%	77.34%
	Split Decision Trees			
Depth	$d=3$	$d=2$	$d=1$	$d=0$
Experimental PCCI	76.00%	78.69%	80.78%	77.45%
	Combination of Decision Trees			
Experimental PCCI	78.22%	79.70%	82.29%	87.82%

The next section uses the combined Machine Learning prediction $\mathbb{D}_d(x,y)$ to perform one-shot predictions of a quad-tree partitioning.

6.4 Machine Learning Approach for Predicting an HEVC Quad-Tree Partitioning

CTU Depth Map (CDM) is also used in his chapter as description of the quad-tree partitioning. As detailed in Section 5.3.1, a CDM is a 8×8 matrix that represents the depths d of an 8x8 square samples of the CTU.

From the Machine Learning prediction $\mathbb{D}_d(x, y)$ defined in Section 6.3.2, 4 different algorithms of quad-tree partitioning prediction are investigated in the following sections: two following a Bottom-up (BU) approach and two following a Top-Down (TD) approach. Following a Bottom-up (BU) approach, the algorithms of quad-tree partitioning prediction use the prediction $\mathbb{D}_d(x, y)$ to decide if a group of 4 CU at depth d have to be merged in a CU at depth $d-1$. Similarly, following a Top-Down (TD) approach, the algorithms use the same prediction $\mathbb{D}_{d-1}(x, y)$ to decide if a CU at depth d has to be split in 4 CU of depth $d+1$.

For both BU and TD approaches, two versions are investigated, namely the *Restrained* and *Unrestrained* versions. The *Restrained* version never calls the decision previously made into question. In other words, the *Restrained* BU algorithm does not try to merge a group of 4 CU if all the 4 CU have not been merged at the previous depths. Similarly, the *Restrained* TD algorithm does not try to split a current CU if the CU have not been split previously. Symmetrically, the *Unrestrained* versions of BU and TD approaches always try every decision despite any previously made decision.

The 4 tested algorithms for quad-tree partitioning prediction are detailed in the following sections.

6.4.1 Bottom-Up Approach for Quad-Tree Partitioning Prediction

Algorithm (6.1) describes our two proposed BU algorithms that predict the quad-tree partitioning using a Machine Learning approach. The algorithm takes as inputs all the features $\mathcal{F}^d_{x,y}$ defined in Section 6.2.4 previously computed and the flag $\mathcal{R}_{flag}$ which is true when the algorithm is *Restrained* and false when the algorithm is *Unrestrained*. The algorithm generates the CDM associated to the input CTU.

First, the full CDM is initialized with the depth value 4 (line 1). Then, the algorithm explores the CTU decomposition with a BU approach: from $d = 4$ to $d = 1$ (line 2). For the current depth d, the algorithm browses the CDM (lines 4-5) taking the block size δ in the CDM (line 3) into account.

Afterwards, when the algorithm is *Restrained* ($\mathcal{R}_{flag} = 1$), it tests if the 4 neighbor blocks in the Z-scan order have the same depth d (line 7). The *Restrained* algorithm does not try to merge neighbor blocks if they have different depths, which corresponds to not having merged at least one CU in the group of 4 CU at the previous depth. In the other hand, when the algorithm is *Unrestrained* ($\mathcal{R}_{flag} = 0$), the algorithm just ignore the condition (line 7) and does not take into account the decisions taken at the previous depth.

Then the algorithm tests whether the blocks have to be merged or not using the prediction $\mathbb{D}_d(x', y')$ (line 10) previously detailed in Section 6.3.2. The blocks are merged if the prediction is true and the corresponding elements in the CDM are set to $d-1$ (line 12).

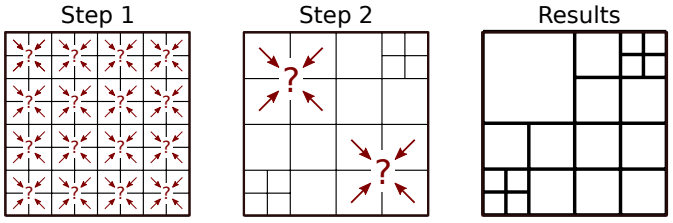
Figures 6.8a and 6.8b illustrate the difference between the Restrained and Unrestrained BU algorithm.

The first step in both algorithms is the same, all groups of CU are tested to be merged. In the other steps, the Restrained BU algorithm only try to merge the groups of CU previously merged, whereas, the Unrestrained BU algorithm try to merge all the CU.

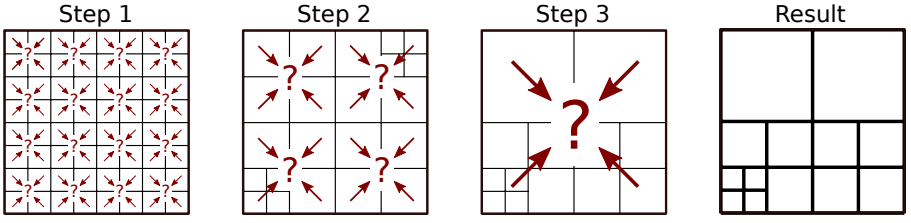
Algorithm 6.1: Bottom-Up CDM Algorithm

Input: $\mathcal{F}^d_{x,y}$ // All features of the CTU
Input: $\mathcal{R}_{flag}$ // True when restrained algorithm
Result: CDM matrix

```
CDM(x, y) = 4, ∀x, y ∈ [0, 7];                                   // Initialization
for (d = 4; d > 0; d − −) do
    δ = 2^(4−d);                                                 // CDM matrix block size
    for (y = 0; y < 8; y += δ) do
        for (x = 0; x < 8; x += δ) do
            // Test if the 4 neighbor blocks have the same depth d when
               the algorithm is restrained (R_flag = 1)
            if ((CDM(x, y) = d  &&
             CDM(x + ⌊δ/2⌋, y) = d  &&
             CDM(x, y + ⌊δ/2⌋) = d  &&
             CDM(x + ⌊δ/2⌋, y + ⌊δ/2⌋) = d)   ||   (¬R_flag)) then
                x' = x/δ; y' = y/δ;                                          // F idx
                Compute D_d(x', y');                             // cf Eq. 6.3, 6.2, 6.1
                if D_d(x', y') then
                    // Block merging in the CDM
                    CDM(x + i, y + j) = d − 1,
                     ∀i, j ∈ [0, δ − 1];
```



(a) *Restrained Bottom-up algorithm illustration example*



(b) *Unrestrained Bottom-up algorithm illustration example*

Figure 6.8 – *Illustration example of the difference between the Restrained and Unrestrained BU algorithms.*

6.4.2 Top-Down Approach for Quad-Tree Partitioning Prediction

Algorithm (6.2) describes our two proposed TD algorithms that predict the quad-tree partitioning using a Machine Learning approach. As for the BU algorithms, the TD algorithm takes as inputs all the features $\mathcal{F}^d_{x,y}$ defined in Section 6.2.4 previously computed and the Restrained/Unrestrained flag $\mathcal{R}_{flag}$.

Algorithm 6.2: Top-Down CDM Algorithm

```
   Input: F^d_{x,y}                                   // All features of the CTU
   Input: R_flag                                      // True when restrained algorithm
   Result: CDM matrix
1  CDM(x, y) = 0, ∀x, y ∈ [0, 7];                     // Initialization
2  for (d = 0; d < 4; d++) do
3      δ = 2^{3−d};                                   // CDM matrix block size
4      for (y = 0; y < 8; y += δ) do
5          for (x = 0; x < 8; x += δ) do
6              // Test if the block has the current depth d when the
                  algorithm is constrained (R_flag = 1)
7              if (CDM(x, y) = d)  ||  (¬R_flag)) then
8                  x' = x/δ; y' = y/δ;                // F idx
9                  Compute D_{d−1}(x', y');           // cf Eq. 6.3, 6.2, 6.1
10                 if D_{d−1}(x', y') then
11                     // Block splitting in the CDM
12                     CDM(x + i, y + j) = d,
                        ∀i, j ∈ [0, δ − 1];
```

First, the full CDM is initialized with the depth value 0 (line 1). Then, the algorithm explores the CTU decomposition with a TD approach: from $d = 0$ to $d = 3$ (line 2). For the current depth d, the algorithm browses the CDM (lines 4-5) taking the block size δ in the CDM (line 3) into account.

Afterwards, when the algorithm is Restrained ($\mathcal{R}_{flag} = 1$), it tests if the depth of the CU is equal to the current depth d (line 7). The *Restrained* algorithm does not try to split a block if the depth of the CU is not equal to the current depth d which corresponds to not having split the CU at the previous depth. In the other hand, when the algorithm is Unrestrained ($\mathcal{R}_{flag} = 0$), the algorithm just ignore the condition (line 7) and does not take into account the decisions taken at the upper depth.

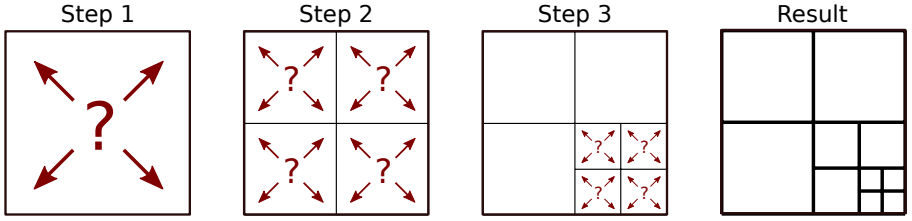
Then, the algorithm tests whether the blocks have to be split or not using the prediction $\mathbb{D}_{d-1}(x', y')$ (line 10) previously detailed in Section 6.3.2. If the prediction is true, the block is split and the corresponding elements in the CDM are set to d (line 12).

Figures 6.9a and 6.9b illustrate the difference between the Restrained and Unrestrained TD algorithm.

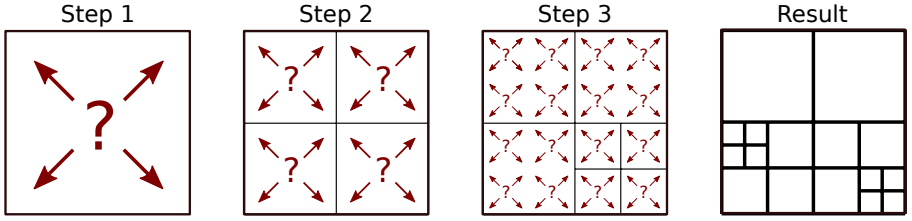
The two first steps in both algorithms are the same. In the second step, since the CU has been split at the previous step, the two algorithms try to split the 4 CU. In the third step, the Restrained BU algorithm only tries to split the groups of CU previously split, i.e. the 4 CU in the bottom left of the CTU. Whereas, the Unrestrained TD algorithm tries to split all the CU.

6.4.3 Accuracy Evaluation of the Quad-Tree Partitioning Prediction

The main objective of Algorithms (6.1) and (6.2) is to generate a CDM that minimizes the prediction error when compared to what the full RDO process would generate. To evaluate the accuracy of our predictions, in addition to the *distance* metric Γ defined in Section 5.4.1, we define a complementary second metric in the next section.



(a) *Restrained Top-Down algorithm illustration example*



(b) *Unrestrained Top-Down algorithm illustration example*

Figure 6.9 – *Illustration example of the difference between the Restrained and Unrestrained TD algorithms.*

6.4.3.1 CTU Depth Map Recall ρ Definition for Performance Analysis

In addition to the *distance* metric Γ defined in Section 5.4.1 page 71, we define the *recall* ρ between two CDM A and B:

$$\rho(A,B) = \left[\sum_{i=0}^{7}\sum_{j=0}^{7}\delta(A(i,j) - B(i,j))\right] \times \frac{1}{64}, \tag{6.4}$$

$$\text{with } \delta(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases} \tag{6.5}$$

The recall $\rho(A,B)$ represents the share of correct quad-tree decomposition in terms of pixel area between predicted CTU A and reference CTU B. Let us use Figure 6.10 as example, the recall between the CDM Figure 6.10a (considered as predicted) and Figure 6.10b (considered as reference) is equal to $\rho = \frac{41}{64} = 64.06\%$.

3	3	2	2	1	1	1	1
3	3	2	2	1	1	1	1
2	2	3	3	1	1	1	1
2	2	3	4	1	1	1	1
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2

(a) *Example of CDM matrix A.*

2	2	2	2	1	1	1	1
2	2	2	2	1	1	1	1
2	2	4	3	1	1	1	1
2	2	4	3	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

(b) *Example of CDM matrix B*

Figure 6.10 – *Example of CDM matrices A and B to illustrate the distance ρ metric. Gray blocks represent the blocks that differ in terms of the CU size between the two CDM A and B.*

The recall $\rho(P,R)$ and the distance $\Gamma(P,R)$ (defined in Section 5.4.1) are used in the following sections to evaluate the accuracy of the prediction with P being the predicted

CDM and R the reference CDM[1], generated by a full RDO process (optimal). The average of $\rho(P,R)$ measurements gives the percentage of perfect prediction in terms of pixel area, it falls between 0% and 100% and the more $\rho(P,R)$ is close to 100%, the more the predicted CDM accurately fit the reference CDM. The average distance $\Gamma(P,R)$ represents the mean error in terms of depth levels between the predicted CDM and the reference one, the more $\Gamma(P,R)$ is close to 0, the more precise the predicted CDM P is.

6.4.3.2 Quad-Tree Partitioning Prediction Evaluation

Table 6.7 details the accuracy of the four proposed quad-tree partitioning prediction algorithms: Restrained BU, Unrestrained BU, Restrained TD and Unrestrained TD, for 18 different sequences belonging to the 5 classes A, B, C, D and E, each one corresponding to a specific resolution or video content. Table 6.7 details the recall $\rho(P,R)$ and the distance $\Gamma(P,R)$, respectively defined in Section 6.4.3.1 and Section 5.4.1, averaged across four QP values: $QP \in \{22, 27, 32, 37\}$.

First of all, it is interesting to notice that learning sequences (marked by * in Table 6.7) used to constitute the training data-set (cf Section 6.2.2) do not yield higher prediction accuracy compared to other sequences. These results demonstrate the non-overfitted behavior of the classifiers.

In terms of quad-tree prediction accuracy in one-shot, Table 6.7 shows that BU algorithms achieve overall better results than the TD ones. More precisely, results show that the Unrestrained BU algorithm obtains the best results with a recall $\rho(P,R)$ around 53% for a distance $\Gamma(P,R)$ of 0.67 depth level. However, it can be noticed that the Restrained BU algorithm has good results for the sequences of small resolution (class D). The small resolution of these sequences tends to have a finer-grained splitting during the RDO process and the Restrained BU algorithm do not try to merge the block at the current depth if the four neighboring blocks in the Z-scan order have not been merged in the previous depth level that tends to predict more finer-grained splitting.

To conclude, the Unrestrained BU algorithm achieves better results on average than the others to predict the quad-tree partitioning in one-shot based on Machine Learning decision. From this quad-tree prediction, it is now possible to evaluate the depth features previously set aside. In the rest of this paper, the Unrestrained BU algorithm using $\mathcal{F}^d_{x,y}$ feature vector is defined as the *Default* solution.

6.5 Testing the Influence of Co-located Depth Features on Quad-Tree Prediction

In Section 6.2.4, we have proposed a fair comparison of state-of-the-art characteristics under a real-time encoder, considering both the information gain for representing the characteristic suitability and the computational overhead of the characteristic. The main conclusion of the experiments is that characteristics extracted from co-located depth and variance information are the most relevant to predict the coding tree decomposition. However, in Section 6.3 and 6.4, the depth features composed by *Prev-CU depth*, *Prev-CTU depth*, *Above-CTU depth* and *Left-CTU depth* were not used due to their direct links to the neighboring CTU quad-tree partitioning, only predicted (and thus not fully available) in a one-shot prediction scheme.

[1] Computed with exhaustive search leading to the optimal solution

Table 6.7 – *The recall $\rho(P,R)$ and distance $\Gamma(P,R)$ according to the sequences of the quad-tree partitioning prediction algorithms: Bottom-up restrained/unrestrained and TD restrained/unrestrained.*

Sequence	Class	Restrained BU ρ (in %)	Restrained BU Γ (in d)	Unrestrained BU ρ (in %)	Unrestrained BU Γ (in d)	Restrained TD ρ (in %)	Restrained TD Γ (in d)	Unrestrained TD ρ (in %)	Unrestrained TD Γ (in d)
*Traffic**	A	44.66	0.87	**46.96**	**0.79**	42.70	0.92	38.88	1.05
PeopleOnStreet	A	50.46	0.75	**51.34**	**0.72**	45.91	0.85	43.83	0.90
Kimono	B	29.73	1.19	**40.82**	**0.94**	39.45	1.10	26.29	1.49
ParkScene	B	42.22	1.05	**47.09**	**0.84**	44.66	0.98	36.30	1.29
BasketballDrive	B	50.42	0.71	**50.74**	**0.69**	47.66	0.78	45.01	0.85
*Cactus**	B	46.45	0.86	**50.03**	**0.73**	46.88	0.86	41.70	1.06
BQTerrace	B	50.30	0.71	**50.35**	**0.69**	47.35	0.76	45.27	0.83
RaceHorses 480p	C	50.13	0.82	**54.59**	**0.64**	51.38	0.73	43.94	0.99
PartyScene	C	52.53	**0.62**	**52.54**	0.63	48.72	0.69	48.24	0.69
BasketballDrill	C	47.25	0.75	43.62	0.85	43.66	0.86	**48.07**	**0.70**
*BQMall**	C	**53.15**	0.67	52.62	**0.66**	49.54	0.75	48.48	0.78
RaceHorses 240p	D	54.21	0.63	**57.88**	**0.54**	54.11	0.61	45.25	0.88
BQSquare	D	**69.35**	**0.39**	67.81	0.42	66.45	0.44	61.80	0.55
BlowingBubbles	D	**53.43**	**0.60**	51.90	0.64	48.42	0.69	47.83	0.69
*BasketballPass**	D	57.10	0.57	**57.46**	**0.56**	52.93	0.66	47.14	0.77
FourPeople	E	49.65	0.76	**50.03**	**0.73**	47.20	0.82	46.43	0.85
*Johnny**	E	**60.52**	**0.54**	60.17	0.55	59.40	0.58	55.59	0.69
KristenAndSara	E	61.08	0.52	**61.32**	**0.50**	59.78	0.55	58.32	0.60
Average		51.26	0.72	**52.63**	**0.67**	49.79	0.76	46.02	0.87

* learning sequences (cf Section 6.2.2)

For the purpose of our study, let us consider a new feature vector $\mathcal{F}2^d_{x,y}$ for a CU at coordinates (x, y) and depth d of a given CTU, composed of the following 16 features, including co-located depths:

- CU var (1 feature).
- Lower-CU var (4 features).
- Upper-CU var (1 feature).
- Nhbr-CU var (3 features).
- Var of lower-CU mean (1 feature).
- Var of lower-CU var (1 feature).

\+ **Prev-CU depth** (1 feature).

\+ **Prev-CTU depth** (1 feature).

\+ **Above-CTU depth** (1 feature).

\+ **Left-CTU depth** (1 feature).

- QP (1 feature).

The training set-up of the new *short* decision trees using the $\mathcal{F}2^d_{x,y}$ feature vector is the same as in Section 6.2.2. Table 6.8 shows the expected PCCI given by the WEKA software of the decision trees trained from the two different feature vectors $\mathcal{F}^d_{x,y}$ and $\mathcal{F}2^d_{x,y}$, according to depth d.

Table 6.8 – *Decision trees accuracy (expected PCCI) using $\mathcal{F}^d_{x,y}$ and $\mathcal{F}2^d_{x,y}$ according to the depth d.*

	Merge Decision Trees			
Depth	$d=4$	$d=3$	$d=2$	$d=1$
PCCI using $\mathcal{F}^d_{x,y}$	81.39%	80.52%	80.19%	81.26%
PCCI using $\mathcal{F}2^d_{x,y}$	83.99%	84.46%	87.41%	88.34%
	Split Decision Trees			
Depth	$d=3$	$d=2$	$d=1$	$d=0$
PCCI using $\mathcal{F}^d_{x,y}$	82.24%	80.89%	80.87%	80.93%
PCCI using $\mathcal{F}2^d_{x,y}$	84.49%	85.13%	88.20%	88.16%

Results show that all the decision trees have better accuracy in terms of percentage of good prediction (PCCI) using the $\mathcal{F}2^d_{x,y}$ with depth features instead of $\mathcal{F}^d_{x,y}$. PCCI increases by at least 2.2% and up to 7.3% across depths.

From the quad-tree prediction algorithm detailed in Section 6.4, it is now possible to evaluate the depth features for predicting quad-tree. Indeed, depth features are strongly correlated to the quad-tree partitioning of nearby CTU. To evaluate the quad-tree prediction accuracy of the added depth features, three cases are considered, inputting one of the three following feature vectors:

- *Default* $\mathcal{F}^d_{x,y}$: no depth feature is employed in this case.
- *Optimal* $\mathcal{F}2^d_{x,y}$: using $\mathcal{F}2^d_{x,y}$ with optimally computed co-located depths. The 4 depth features result from an exhaustive search of the RDO process and leading to

the optimal quad-tree partitioning. This case is a reference, using the best possible co-located depths.

- *Predicted* $\mathcal{F}2^d_{x,y}$: using $\mathcal{F}2^d_{x,y}$ with predicted co-located depths. The 4 depth features result from a one-shot quad-tree partitioning of our Unrestrained BU algorithm (6.1) detailed in Section 6.4 and exploit the new decision trees based on the $\mathcal{F}2^d_{x,y}$ feature vector.

The used quad-tree prediction algorithm is the Unrestrained BU (6.1) (Section 6.4) and the combined prediction $\mathbb{D}_d(x,y)$ is unchanged as defined in Section 6.3.2. Table 6.9 details the recall $\rho(P,R)$ and distance $\Gamma(P,R)$, respectively defined in Section 6.4.3.1 and Section 5.4.1, averaged across the four values of $QP \in \{22,27,32,37\}$, which evaluate the accuracy of the one-shot quad-tree prediction.

Table 6.9 – *The recall* $\rho(P,R)$ *and distance* $\Gamma(P,R)$ *according to the sequences for Default, Optimal and Predicted cases.*

Sequence	**Default** $\mathcal{F}^d_{x,y}$		**Optimal** $\mathcal{F}2^d_{x,y}$		**Predicted** $\mathcal{F}2^d_{x,y}$	
	ρ (in %)	Γ (in d)	ρ (in %)	Γ (in d)	ρ (in %)	Γ (in d)
Traffic	46.96	0.79	51.31	0.68	35.90	1.10
PeopleOnStreet	51.34	0.72	51.36	0.68	34.49	1.12
Kimono	40.82	0.94	65.88	0.49	44.60	1.03
ParkScene	47.09	0.84	54.04	0.68	28.17	1.44
BasketballDrive	50.74	0.69	50.29	0.67	27.80	1.36
Cactus	50.03	0.73	57.39	0.59	40.09	1.09
BQTerrace	50.35	0.69	51.25	0.63	20.74	1.74
RaceHorses480	54.59	0.64	54.44	0.63	29.39	1.42
PartyScene	52.54	0.63	51.47	0.62	42.93	0.81
BasketballDrill	43.62	0.85	52.14	0.63	36.10	1.08
BQMall	52.62	0.66	53.99	0.61	26.47	1.59
RaceHorses240	57.88	0.54	56.60	0.56	46.31	0.81
BQSquare	67.81	0.42	69.11	0.38	58.85	0.63
BlowingBubbles	51.90	0.64	51.85	0.62	45.56	0.76
BasketballPass	57.46	0.56	56.06	0.58	39.90	1.09
FourPeople	50.03	0.73	53.92	0.63	42.63	0.92
Johnny	60.17	0.55	64.39	0.46	55.78	0.67
KristenAndSara	61.32	0.50	63.82	0.45	51.46	0.72
Average	52.63	0.67	56.07	0.59	39.29	1.08

Regarding the *Optimal* $\mathcal{F}2^d_{x,y}$ case, results show that the prediction accuracy increases on average using the new decision trees including the depth features compared to the *Default* setting. The recall $\rho(P,R)$ increases on average of 3.44% for a reduction of the distance $\Gamma(P,R)$ of 0.08 depth level. These results are obtained when the depth features used by the decision trees are without degradation (optimal). However, the results of the *Predicted* $\mathcal{F}2^d_{x,y}$ case, based on the prediction with the same techniques, show a significant decrease in quad-tree prediction accuracy. The recall $\rho(P,R)$ decreases on average by 13.34% for an increase of the distance $\Gamma(P,R)$ of 0.41 depth levels compared to the *Default* $\mathcal{F}^d_{x,y}$ case. This significant degradation of the quad-tree prediction is due to error propagation. Indeed, the depth approximation of the neighboring CTU, due to prediction

errors of the quad-tree partitioning, are used as features by the decision trees; approximations lead to more and more errors. In other words, when the algorithm predicts too fine splitting for example, and thus generates errors for a CTU, these errors are used to predict the quad-tree partitioning of the next CTU which tends to continue to predict too fine splitting (or vise-versa).

Results in Table 6.9 show that depth features significantly increase the one-shot quad-tree prediction accuracy in the optimal case. However, to be used in a real set-up, propagation of errors must be considered and our results show that depth features rapidly become inefficient when the RDO process is drastically limited. They are thus not suited in the context of a one-shot quad-tree prediction but will be studied when the constraint is more relaxed in the next chapter.

Based on the previously presented one-shot prediction of the quad-tree partitioning, the next section proposes a method to drastically reduce the energy consumption of the recursive RDO process. The method is then evaluated in terms of BD-BR.

6.6 Using Quad-Tree Prediction to Drastically Reduce Encoding Energy Consumption

To reduce the energy consumption of an HEVC encoder, our proposed energy reduction technique uses the one-shot prediction of the quad-tree partitioning based on Machine Learning. We propose to use this prediction to limit the exhaustive search of the RDO process on the CTU quad-tree decomposition.

6.6.1 Energy Reduction Scheme

Figure 6.11 presents a high-level diagram of our resulting Machine Learning-based HEVC encoding energy reduction. First, the features ($\mathcal{F}^d_{x,y}$ or $\mathcal{F}2^d_{x,y}$) are computed for the whole frame to minimize the computational complexity overhead. Then, the Unrestrained BU Algorithm (6.1) is applied using the features to generate the CDM.

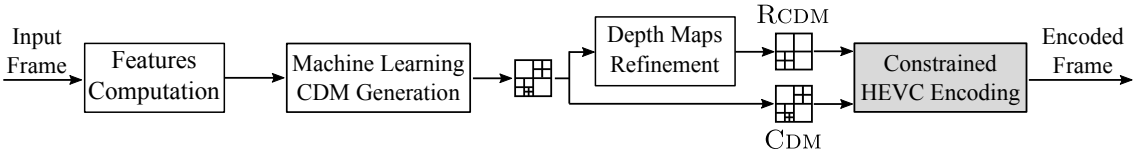


Figure 6.11 – *Diagram of the Machine Learning proposed energy reduction scheme. The features are first computed for the whole input frame. They are then used to generate the* CDM. *The* CDM *is refined to generate the Refined CTU Depth Map (*RCDM*) and the encoder is finally constrained to only apply the RDO process in the interval formed by the two* CDM.

As already explain in Chapter 5, the CDM generated with Algorithm (6.1) is very restrictive for the RDO process as it allows only one depth to be searched for each CU of a CTU. To increase the accuracy of the depth map prediction with limited impact on the complexity, the *Refinement* algorithm detailed in Section 5.4.4 is used, and the HEVC encoder is forced to only apply the RDO process between the two CDM. As a reminder, the RCDM is generated from the CDM by merging all the group of four neighboring blocks (in the Z-scan order) with the same depth in the input CDM. Finally, the HEVC encoder is forced to only apply the RDO process between the previously generated CDM and RCDM.

6.6.2 Results on Encoding Degradation and Energy Reduction

The experimental set-up has been detailed in Section 4.3.1. Table 6.10 details the performance of the proposed energy reduction scheme, for 18 different sequences belonging to the 5 classes A, B, C, D and E, each one corresponding to a specific resolution or video content. The results are composed by the BD-BR and BD-PSNR which are the common metrics used in video compression and the ER values including the energy overhead due to the entire energy reduction scheme.

Table 6.10 – *BD-BR, BD-PSNR and ER of the Machine Learning drastic energy reduction schemes according to the sequences using the $\mathcal{F}^d_{x,y}$ and $\mathcal{F}2^d_{x,y}$ feature vectors.*

Sequence	**Predicted** (using $\mathcal{F}2^d_{x,y}$)			**Default** (using $\mathcal{F}^d_{x,y}$)		
	BD-BR (in %)	BD-PSNR (in dB)	ER (in %)	BD-BR (in %)	BD-PSNR (in dB)	ER (in %)
Traffic	7.83	-0.41	58.95	4.05	-0.21	58.69
PeopleOnStreet	10.67	-0.59	59.01	3.73	-0.21	57.09
Kimono	11.40	-0.38	57.26	9.51	-0.31	61.33
ParkScene	10.08	-0.43	58.90	3.86	-0.17	60.28
BasketballDrive	32.72	-0.84	56.89	4.65	-0.13	60.16
Cactus	11.51	-0.40	59.66	3.79	-0.14	61.34
BQTerrace	25.84	-1.15	52.83	1.99	-0.13	58.93
RaceHorses 480p	28.89	-1.52	56.13	2.95	-0.17	60.37
PartyScene	6.05	-0.42	55.22	1.10	-0.08	57.10
BasketballDrill	17.12	-0.78	59.72	4.97	-0.24	62.26
BQMall	34.00	-1.65	57.23	3.16	-0.17	57.13
RaceHorses 240p	19.11	-1.10	58.07	1.93	-0.12	58.57
BQSquare	13.43	-1.03	56.82	1.00	-0.08	55.51
BlowingBubbles	5.90	-0.40	53.12	1.24	-0.08	53.54
BasketballPass	27.73	-1.49	56.16	2.31	-0.14	58.01
FourPeople	9.69	-0.53	56.11	4.51	-0.25	55.13
Johnny	12.89	-0.51	51.85	5.49	-0.22	52.26
KristenAndSara	16.49	-0.79	52.10	4.62	-0.23	53.39
Average	16.74	-0.80	56.45	3.60	-0.17	57.84

Two cases of the proposed energy reduction scheme are evaluated: the *Default* case using the decision trees described in Section 6.3 and the *Predicted* case using the depth features and the decision trees detailed in Section 6.5.

The results show that the proposed energy reduction scheme achieves an average of 58% energy reduction with 3.6% BD-BR increase, and a quality degradation close to -0.17dB BD-PSNR in the *Default* case. As may be expected, using the depth features produces a significant BD-BR increasing of 16.74% and quality degradation close to -0.80dB, due to the error propagation explain in Section 6.5. By comparing Table 6.9 and Table 6.10, the results show that the two metrics $\rho(P,R)$ and $\Gamma(P,R)$ of quad-tree prediction accuracy are well correlated with the impact in encoding degradation.

Considering the *Default* case, it is noticeable in Table 6.10 than the Kimono sequence has more degradation than the other sequences: 9.51% of BD-BR increasing. This can be explained by the texture specificity of the Kimono video sequence which is composed by a

traveling with trees and vegetation in the background. This video sequence has the highest Spatial Information (54.1) due to the details (cf Table 6.4 in Section 6.2.2).

6.7 Performance Comparison of Predicting HEVC Quad-Tree Partitioning

This section presents a detailed comparison of our two proposed energy reduction schemes based on one-shot quad-tree partitioning: the statistical and the Machine Learning approaches and extends it to the related works.

6.7.1 Comparison of our Statistical and Machine Learning Approaches

Table 6.11 details the performance of the two proposed energy reduction schemes: the statistical and the Machine Learning approaches, for 18 different sequences belonging to the 5 classes A, B, C, D and E, each one corresponding to a specific resolution or video content. Two types of metrics are detailed in Table 6.11. The first one is composed by the $\rho(P,R)$ measure and the distance $\Gamma(P,R)$, respectively defined in Section 6.4.3.1 and Section 5.4.1, averaged across the four QP values which evaluate the precision of the quad-tree prediction for all constrained frames. The second one is composed by BD-BR and BD-PSNR [WSBL03] that are the usual metrics used in video compression to measure the compression efficiency difference between two encodings. The ER values include the energy overhead due to the entire energy reduction scheme (features or variance computation and CDM prediction).

In terms of quad-tree prediction accuracy in one-shot, Table 6.11 shows that the Machine Learning energy reduction techniques achieve better results (around 53% of $\rho(P,R)$ for a distance $\Gamma(P,R)$ of 0.67 depth level) than the statistical energy reduction techniques (around 50% of $\rho(P,R)$ for a distance $\Gamma(P,R)$ of 0.79 depth level).

The results show that both energy reduction techniques achieve an average of 60% of energy reduction. In fact, the overhead due to the unconstrained Learning Frame (F_L) and the variance computations of the statistical approach is approximately equal to the overhead of the features computations of the Machine Learning approach. However, even if the statistical approach does not constrain all the frames (only 49 every 50 frames), this approach causes higher encoding degradation: +0.35% of BD-BR and -0.02dB of BD-PSNR, than the Machine Learning approach. These results show that the two metrics $\rho(P,R)$ and $\Gamma(P,R)$ of quad-tree prediction accuracy are well correlated with the impact in encoding degradation.

It is noticeable in Table 6.11 than the Kimono sequence has more degradation than the other sequences: 13.28% of BD-BR increasing with the statistical approach and 9.51% of BD-BR increasing with the Machine Learning approach. This can be explained by the texture specificity of the Kimono video sequence which is composed by a traveling with trees and vegetation in the background. This video sequence has the highest Spatial Information (54.1) due to the details. Nevertheless, the results show that the Machine Learning approach reduces the degradation of 3.77% of BD-BR compare to the statistical approach.

To conclude, the Machine Learning approach achieves better results on average than the statistical approach and does not require unconstrained learning frames to predict the quad-tree partitioning, learning process is achieved off-line. These two points make the proposed Machine Learning approach a good candidate to build energy reduction methods for real-time HEVC encoders.

Table 6.11 – *The recall $\rho(P,R)$, distance $\Gamma(P,R)$, BD-BR, BD-PSNR and ER of the statistical and Machine Learning drastic energy reduction schemes according to the sequences. For the same energy reduction, the Machine Learning energy reduction techniques achieve better results than the statistical energy reduction techniques for both quad-tree prediction accuracy and encoding degradation.*

			Statistical					Machine Learning				
Class	**Sequence name**	**Resolution**	ρ (in %)	Γ (in d)	BD-BR (in %)	BD-PSNR (in dB)	ER (in %)	ρ (in %)	Γ (in d)	BD-BR (in %)	BD-PSNR (in dB)	ER (in %)
A	*Traffic*	2560x1600	44.76	0.86	4.59	-0.24	63.11	46.96	0.79	4.05	-0.21	58.69
A	*PeopleOnStreet*	2560x1600	51.92	0.70	4.28	-0.24	62.12	51.34	0.72	3.73	-0.21	57.09
B	*Kimono*	1920x1080	18.87	2.06	13.28	-0.43	55.66	40.82	0.94	9.51	-0.31	61.33
B	*ParkScene*	1920x1080	38.66	1.24	4.29	-0.19	58.79	47.09	0.84	3.86	-0.17	60.28
B	*BasketballDrive*	1920x1080	47.68	0.76	3.71	-0.11	63.53	50.74	0.69	4.65	-0.13	60.16
B	*Cactus*	1920x1080	43.86	0.95	3.56	-0.13	63.19	50.03	0.73	3.79	-0.14	61.34
B	*BQTerrace*	1920x1080	51.83	0.66	2.26	-0.14	63.76	50.35	0.69	1.99	-0.13	58.93
C	*RaceHorses*	832x480	46.86	0.91	3.12	-0.18	62.60	54.59	0.64	2.95	-0.17	60.37
C	*PartyScene*	832x480	55.38	0.56	1.88	-0.13	61.78	52.54	0.63	1.10	-0.08	57.10
C	*BasketballDrill*	832x480	51.40	0.64	2.74	-0.13	61.65	43.62	0.85	4.97	-0.24	62.26
C	*BQMall*	832x480	53.78	0.66	3.46	-0.19	60.98	52.62	0.66	3.16	-0.17	57.13
D	*RaceHorses*	416x240	52.01	0.69	2.47	-0.15	61.75	57.88	0.54	1.93	-0.12	58.57
D	*BQSquare*	416x240	65.92	0.41	2.74	-0.21	60.62	67.81	0.42	1.00	-0.08	55.51
D	*BlowingBubbles*	416x240	55.70	0.55	1.45	-0.10	58.12	51.90	0.64	1.24	-0.08	53.54
D	*BasketballPass*	416x240	57.55	0.55	2.39	-0.14	62.42	57.46	0.56	2.31	-0.14	58.01
E	*FourPeople*	1280x720	49.80	0.74	4.79	-0.27	59.57	50.03	0.73	4.51	-0.25	55.13
E	*Johnny*	1280x720	53.61	0.67	6.02	-0.23	59.24	60.17	0.55	5.49	-0.22	52.26
E	*KristenAndSara*	1280x720	58.50	0.56	4.15	-0.21	59.73	61.32	0.50	4.62	-0.23	53.39
		Average	**49.89**	**0.79**	**3.95**	**-0.19**	**61.03**	**52.63**	**0.67**	**3.60**	**-0.17**	**57.84**

6.7.2 Comparison with Related Works

Since the goal of the statistical and the Machine Learning approaches is to predict CTU quad-tree partitioning in one-shot (before starting the RDO process), the performance comparison of our proposed energy reduction schemes is focus on predicted quad-tree partitioning reduction techniques detailed in Section 3.2.1.2.

Figure 6.12 presents a comparison between our two proposed energy reduction techniques based on one-shot quad-tree partitioning prediction and techniques presented in Section 3.2.1.2. The results of configuration C7 presented in Chapter 5 and the new results of the proposed energy reduction schemes based on ML are plotted in Figure 6.12. Results from techniques implemented on HM Khan[KSH13], Ruiz14[RCAFE+14], Ruiz15[RFEA+15], Duanmu[DMW15], Min[BC15] and Feng[FDZX16] are plotted in red. We reimplemented in the real-time software encoder Kvazaar two complexity reduction techniques extracted from Shen*[SZA13] Zhang*[ZLL15] plotted in blue in Figure 6.12. Crosses and circles correspond respectively to results in terms of energy and complexity reductions.

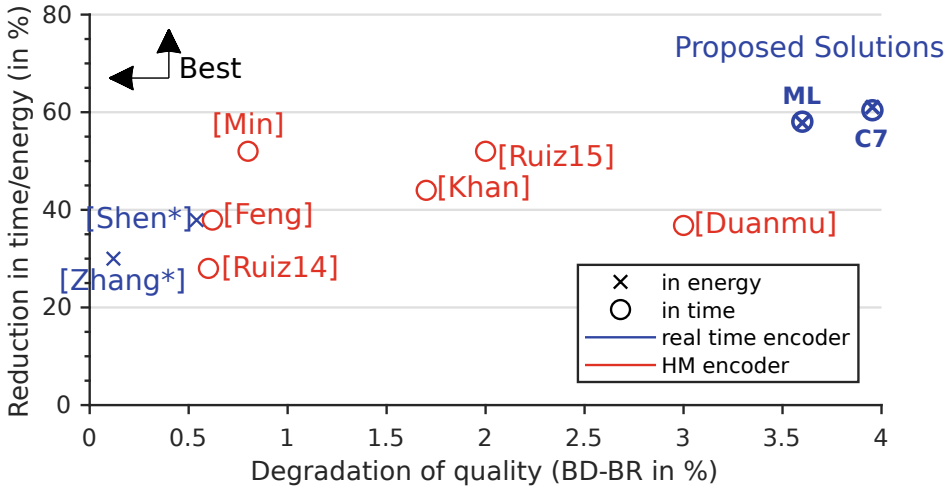


Figure 6.12 – *Comparison with related works in terms of time/energy reduction (in %) and quality degradation (BD-BR in %).*

This comparison is unfair to our methods in terms of time/energy, as the complexity overhead of the proposed complexity reduction techniques in literature would be higher in the context of a real time encoder, thus reducing their complexity reduction.

Figure 6.12 shows that the energy reduction of our two proposed energy reduction schemes are superior to the complexity reduction of all state-of-the-art studies and measured in the more difficult conditions of a real-time encoder. Only Min[BC15] has an energy reduction close to our configurations. This technique uses 8 filtered images to compute their local and global edge complexities and predict the partitioning of CU. For the characteristics selection presented in Section 6.2.4, we reimplemented in the real-time software encoder Kvazaar the local and global edge complexities computations to estimate the overhead of these characteristics. We did not optimize all computation but, as shown in Table 6.3 of Section 6.2.4, the computational overhead of the filtering is around 30% for the real-time configuration described in Section 4.3.1. This overhead would lead to a significant decrease of energy/time reduction for this method.

6.8 Conclusion

In this chapter we have proposed a new "one-shot" quad-tree prediction technique based on *Machine Learning*. Quad-tree prediction is used to drastically limit the energetic consumption of the recursive RDO process. Experimental results show that the new proposed technique is able to reduce the energy consumption of the HEVC encoder by 58% — including the additional algorithm overhead in a real-time encoder — for a bit rate increase of 3.6%. Such a large energy reduction is obtained by tailoring the features used by Machine Learning for inferring the quad-tree.

Then, this chapter proposes a comparison of our two energy reduction methods for real-time HEVC Intra encoders. The statistical method proposed in Chapter 5 exploits the correlation between a CTU partitioning and the variance of the CTU luminance samples to predict the quad-tree decomposition in one-shot. The second method proposed in this chapter uses a Machine Learning method to predict in one-shot the quad-tree decomposition.

Experimental results show that the Machine Learning method has a slight edge over the statistical method and that this performance has a direct impact on the encoding degradations. Both energy reduction techniques are capable of reducing the energy consumption of the HEVC encoder around 60% — including the additional algorithm overhead under a real-time encoder — for a bit rate increase of respectively 3.95% and 3.6%.

The main objective of this chapter was to design the *Coarse Solution Predictor* defined in the SSSR methodology in Section 4.2.2.2 for the MSSE corresponding to the CTU level. For this reason we chose to work on a new "one-shot" quad-tree prediction technique without using Learning Frame (F_L). The Machine Learning approach achieves better results on average than the statistical approach and does not require unconstrained learning frames to predict the quad-tree partitioning. These two points make the proposed Machine Learning approach a good candidate as *Coarse Solution Predictor* to build energy control methods for real-time HEVC encoders, as shown in next chapter.

CHAPTER 7

Tunable Complexity for HEVC Intra Encoding

7.1 Introduction

Chapter 6 has presented a "one-shot" quad-tree prediction technique based on *Machine Learning* illustrated by Figure 7.1. For a given CTU, this technique predicts a CTU Depth Map (CDM) before starting the RDO process. This one-shot quad-tree prediction is used in this chapter as *Coarse* CDM *Solution Predictor* defined in Section 4.2.2.2 in order to control the complexity of the HEVC Intra encoding process. The goal of this chapter is to propose a technique to explore the trade-off between encoding efficiency and complexity. From the one-shot quad-tree partitioning prediction, the search space is relaxed to expend it. As the proposed solutions do not include hardware-related features such as DVFS or DPM [NPMM16], the energy reduction is linear to the complexity reduction. In a real-time context, DVFS or DPM mechanisms allow amplifying the gain in energy when the complexity is reduced. The contributions proposed in this chapter are focused on complexity reduction while the same results could be represented in energy saving.

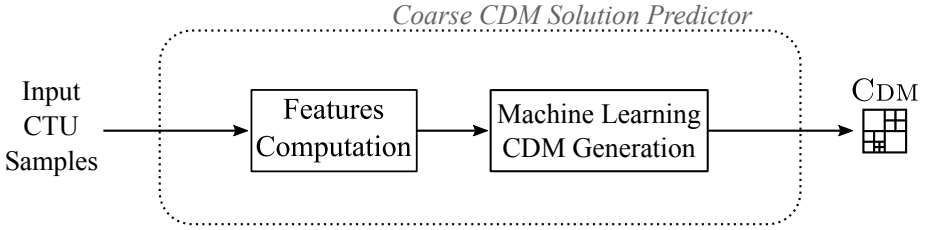


Figure 7.1 – *Coarse* CDM *solution predictor scheme.*

The rest of this chapter is organized as follows. Section 7.2 presents the CDM *Search Space Relaxation* algorithm used to expand the search space around the solution obtained with the coarse solution predictor, in our case, around the CDM generated by the *Coarse* CDM *Solution Predictor* illustrated in Figure 7.1. Section 7.3 proposes a solution to allocate the complexity more finely at the frame level. Section 7.4 presents the results of the proposed tunable complexity solution of HEVC Intra encoding process proposed in this chapter. Finally, Section 7.5 concludes this chapter.

7.2 CDM Search Space Relaxation

In HEVC, exploring the entire search space of CTU quad-tree partitioning process corresponds to test every CU block size, i.e. all the depth levels of the CTU quad-tree decomposition (see Figure 2.8). To limit and control the complexity of the HEVC encoding process, the proposed solution is to test a selected number of depth levels around the CDM generated by the *Coarse* CDM *Solution Predictor* as illustrated in Figure 7.2.

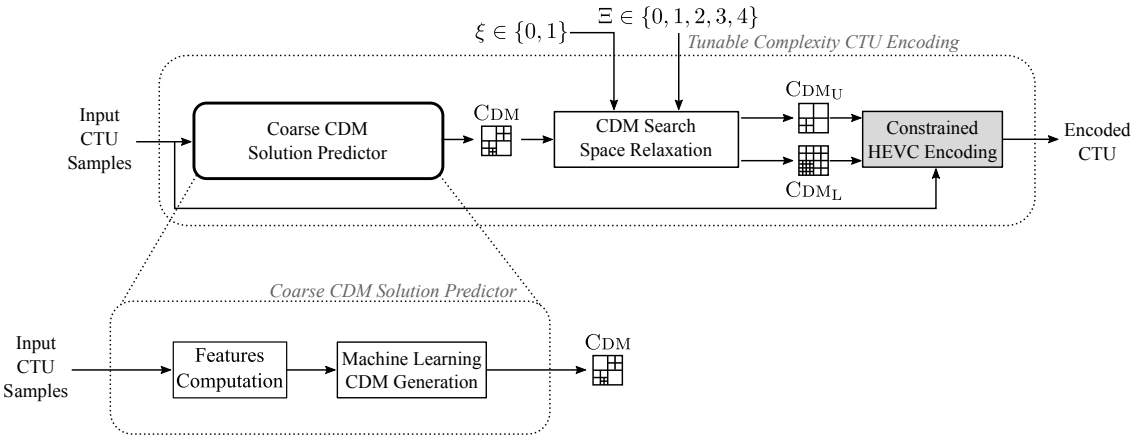


Figure 7.2 – *Tunable complexity CTU encoding scheme.*

The *Tunable Complexity CTU Encoding* takes as inputs the CTU samples and uses the *Coarse* CDM *Solution Predictor* to predict the associated CDM. The CDM is then used to generate two CDM: the Upper CDM_U and the Lower CDM_L. The two CDM_U and CDM_L are generated from an input CDM by the CDM *Search Space Relaxation* algorithm described by Figure 7.3. Finally, the HEVC encoder explores the depths between CDM_L and CDM_U, i.e. the HEVC encoder is forced to only apply the RDO process between the interval defined by CDM_L and CDM_U.

The CDM *Search Space Relaxation*, illustrated in Figure 7.3, takes a CDM and a couple of parameters (Ξ, ξ) as inputs to generate the $\text{CDM}_L(\Xi, \xi)$ and $\text{CDM}_U(\Xi, \xi)$. The parameter Ξ sets the number of depth levels between $\text{CDM}_L(\Xi, \xi)$ and $\text{CDM}_U(\Xi, \xi)$; the CTU quad-tree decomposition having 5 depth levels (from $d = 0$ to $d = 4$), Ξ is included in the interval $\Xi \in \{0, 1, 2, 3, 4\}$. When $\Xi = 0$, only one depth level, corresponding to the input CDM, is tested whereas when $\Xi = 4$, all depth levels (from $d = 0$ to $d = 4$) are tested. The second parameter ξ is a flag used to select a second version of the CDM *Search Space Relaxation*. This parameter is explained later and considered equal to 0 (disabled) for the moment.

To add one depth level around the input CDM, $\Xi = 1$, the algorithm is split into two steps:

1. The first step applies the *Refinement* algorithm described in Section 5.4.4 on the CDM to generate $\text{CDM}_U(\Xi, \xi)$. As a reminder, the *Refinement* algorithm merges all the group of four neighboring blocks (in the Z-scan order) with the same depth in the input CDM. Chapters 5 and 6 have indeed demonstrated the quality improvements of the *Refinement* algorithm in terms of BD-BR. The number of blocks merged by the *Refinement* algorithm depends on the input CDM and is not predictable. The distance $\Gamma(\text{CDM}, \text{CDM}_U(1, 0))$ is unknown and fills in the interval $[0, 1]$. The merged blocks are represented in red in Figure 7.3.

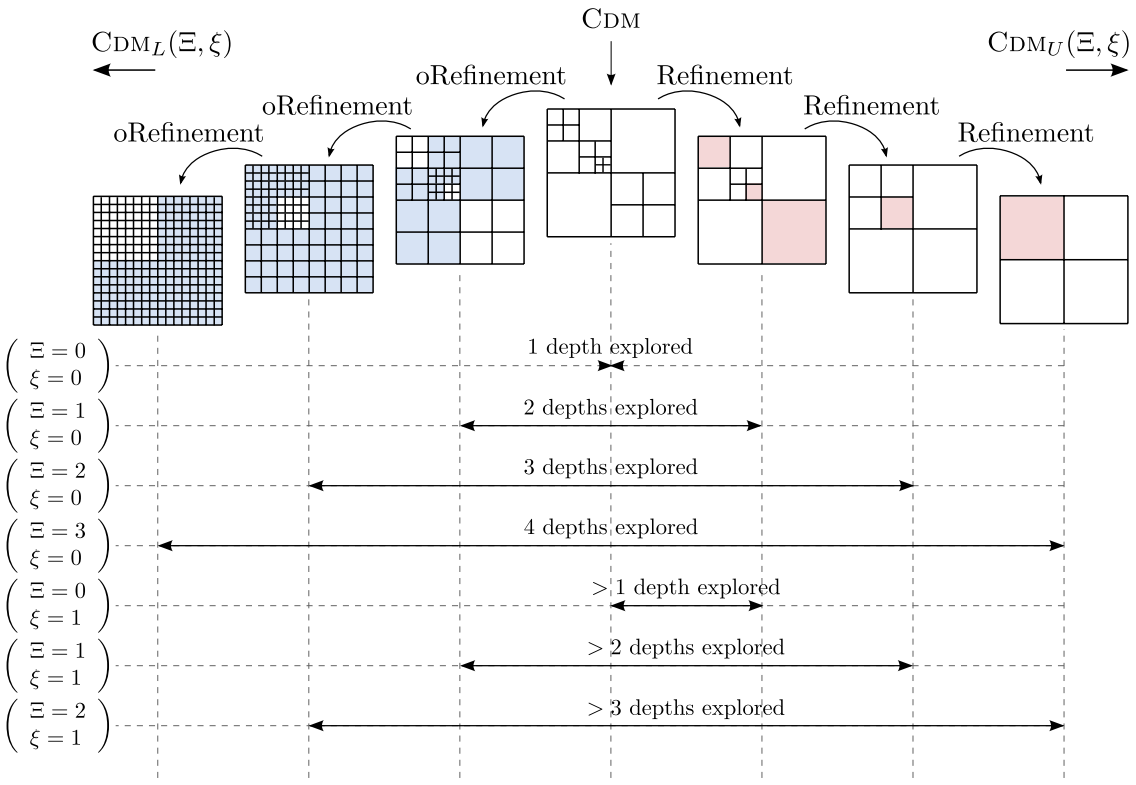


Figure 7.3 – CDM *Search Space Relaxation. The merged and split blocks are represented in red and blue, respectively. The number of depth levels tested when the quad-tree partitioning process is constrained is illustrated according to the couple of parameters* (Ξ,ξ).

2. The second step applies the "opposite" algorithm, called *oRefinement* algorithm, to generate CDM$_L(\Xi,\xi)$. The *oRefinement* algorithm generates the CDM$_L(\Xi,\xi)$ from the CDM by splitting all blocks that have not been merged during the *Refinement* algorithm of step 1. The split blocks are represented in blue in Figure 7.3. The distance $\Gamma($CDM$_L(1,0)$, CDM$_U(1,0))$ is exactly equal to 1. When constrained between the CDM$_L(1,0)$ and CDM$_U(1,0))$, the encoding process tests exactly 2 depth levels spread around the input CDM.

To add more than one depth level, the steps 1 and 2 are repeated on the CDM$_U(\Xi,\xi)$ and CDM$_L(\Xi,\xi)$, respectively, as illustrated in Figure 7.3. When $\xi=0$, this two pass algorithm generates an interval of CDM having a predictable and exact number of depths levels spread around the input CDM.

As specifically shown in Section 5.5.2.2, applying the *Refinement* algorithm to relax the quad-tree constraint consumes between 2.7% and 6.3% more energy for not negligible gains of encoding performance, between 1.3% and 7.33% in terms of BD-BR. The flag $\xi\in\{0,1\}$ is defined to apply an optional extra *Refinement* algorithm pass on CDM$_U(\Xi,\xi)$ as illustrated in Figure 7.3, when the parameter ξ is equal to 1. The main drawback of this second version of the CDM *Search Space Relaxation* is the unpredictability of the number of depth levels between CDM$_U(\Xi,\xi)$ and CDM$_L(\Xi,\xi)$ that can prevent from an accurate future control of the encoding complexity.

When $\Xi=3$, the RDO process tests 4 depth levels, i.e. does not test either the depth $d=0$ or the depth $d=4$. It is why, when $\Xi=3$, ξ is forced to 0 because applying an extra *Refinement* algorithm pass on this specific case would force the RDO process on all

depth levels (if depth $d = 0$ was not tested) or would not impact the interval of CDM. The two versions of the CDM *Search Space Relaxation* ($\xi = 0$ and $\xi = 1$) are studied in this chapter.

7.2.1 Results of CDM Search Space Relaxation Algorithm

The next section presents the performance in terms of BD-BR of the CDM *Search Space Relaxation* according to the two parameters Ξ and ξ. The experimental set-up has been detailed in Section 4.3.1. The results presented in this section are the average results obtained for the 18 video sequences presented in Table 4.1.

7.2.1.1 Testing the performance of Co-located Depth Features on Quad-Tree Prediction

In Chapter 6, two features vectors $\mathcal{F}^d_{x,y}$ and $\mathcal{F}2^d_{x,y}$ were considered for the one-shot quad-tree prediction scheme (*Coarse CDM Solution Predictor*). The second features vector adds the features extracted from depth composed by *Prev-CU depth*, *Prev-CTU depth*, *Above-CTU depth* and *Left-CTU depth* as detailed in Section 6.5. These features are directly linked to the neighboring CTU quad-tree partitioning and thus impacted by the constraints applied on the neighboring CTU. The results presented in Section 6.5 have shown that using $\mathcal{F}2^d_{x,y}$ features vector leads to significant degradation of the quad-tree prediction due to error propagation in the context of a one-shot quad-tree prediction. Indeed, the depth approximations of the neighboring CTU, due to prediction errors of the quad-tree partitioning, are used as features by the decision trees. Approximations thus lead to more and more errors.

Figure 7.4 shows the results in terms of BD-BR according to Ξ (with $\xi = 0$) when the *Coarse CDM Solution Predictor* uses $\mathcal{F}^d_{x,y}$ (without depth features) and $\mathcal{F}2^d_{x,y}$ (with depth features) for the one-shot quad-tree partitioning prediction, in blue and green, respectively.

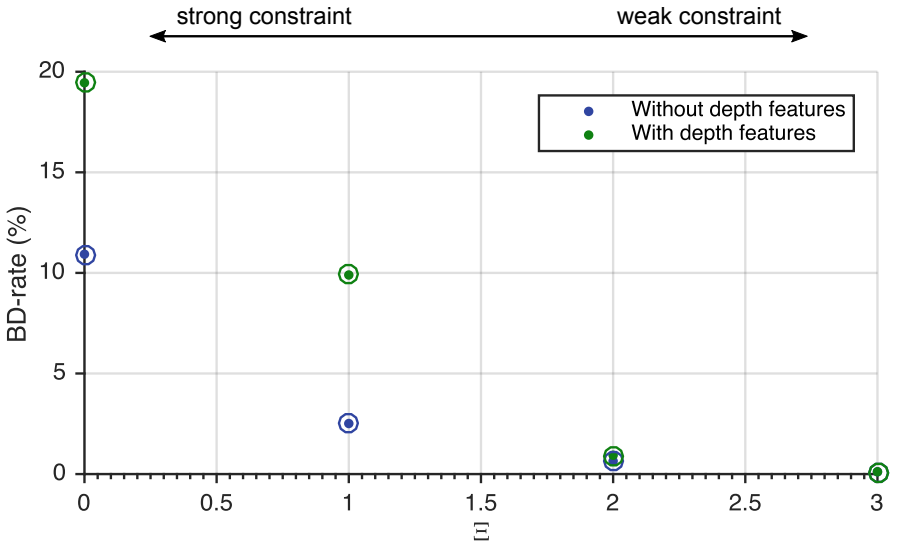


Figure 7.4 – *BD-BR according to* Ξ *when the* Coarse CDM Solution Predictor *uses* $\mathcal{F}^d_{x,y}$ *or* $\mathcal{F}2^d_{x,y}$*, i.e. when the* Coarse CDM Solution Predictor *uses (in blue) or does not use (in green) the depth features.*

Results show that for $\Xi \in \{0, 1, 2, 3\}$, the features vector $\mathcal{F}2^{d}_{x,y}$ causes more degradation in terms of BD-BR than using $\mathcal{F}^{d}_{x,y}$, with up to 8.52% of BD-BR increase for $\Xi = 0$. Although the CDM *Search Space Relaxation* is not as restrictive as the energy reduction scheme presented in Section 6.6.1, the error propagation due to the quad-tree constraints of the neighboring CTU is too important and leads to significant degradation of BD-BR. More results are shown latter in Section 7.4.

7.2.1.2 Performance Results of CDM Search Space Relaxation

Figures 7.5 and 7.6 show the results in terms of BD-BR and BD-PSNR according to parameter Ξ when $\xi = 0$ and $\xi = 1$, i.e. when an extra pass of the *Refinement* algorithm is not and is applied, respectively.

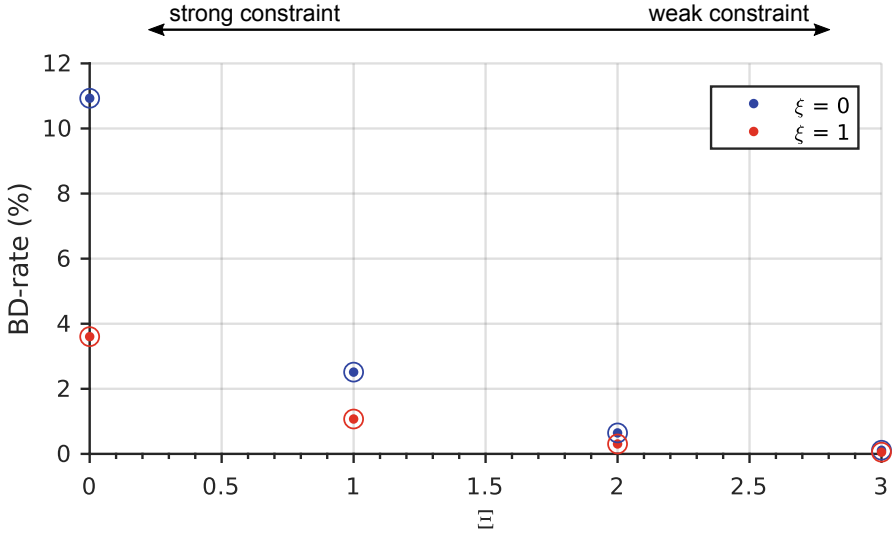


Figure 7.5 – *BD-BR according to parameter Ξ when $\xi = 0$ and $\xi = 1$, i.e. when an extra pass of the* Refinement *algorithm is not (in blue) and is applied (in red) on* CDM_U.

As observed in Chapters 5 and 6, applying an extra pass of *Refinement* algorithm ($\xi = 1$) on the CDM_U significantly improves the encoding performance in terms of BD-BR and BD-PSNR. The results of Figures 7.5 and 7.6 show that the gains obtained by applying an extra *Refinement* algorithm on the CDM_U are between:

- -0.34% ($\Xi = 2$) and -7.33% ($\Xi = 0$) in terms of BD-BR,
- +0.015 dB ($\Xi = 2$) and +0.34 dB ($\Xi = 0$) in terms of BD-PSNR.

The results show that the differences of BD-BR and BD-PSNR when $\xi = 0$ and $\xi = 1$ decrease as the parameter Ξ increases because the extra pass of *Refinement* algorithm has less and less impact. These results confirm the interest of setting $\xi = 1$ in terms of encoding quality. The impact of ξ on the complexity reduction of the system will be discussed in Section 7.4.

To conclude, this section presented the CDM *Search Space Relaxation* used to expand the search space around the CDM generated by the *Coarse* CDM *Solution Predictor*, i.e. to generate an interval of depth levels across the CDM_U and CDM_L. However, the number of depth levels set by the parameter Ξ is constrained to be an integer value between 0 and 4. Due to the fact that setting a distance between two CDM of any value is not feasible without

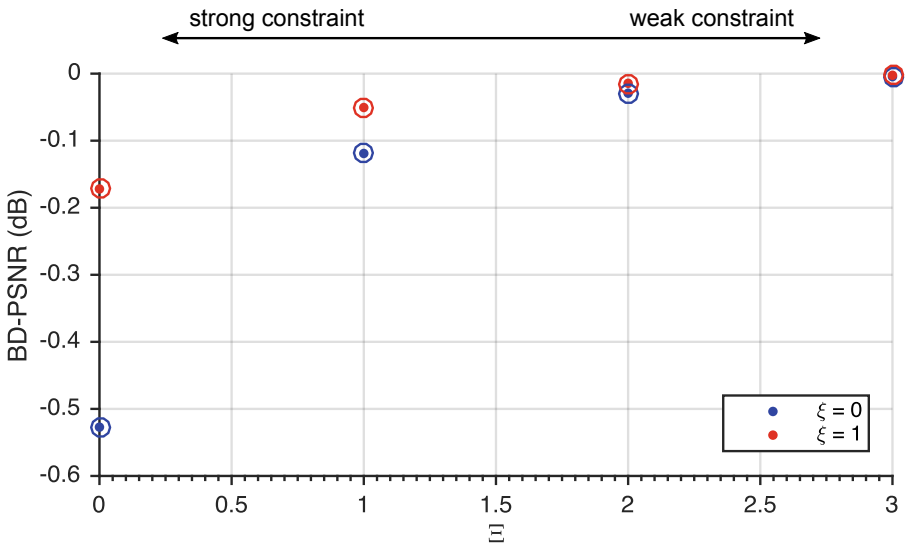


Figure 7.6 – *BD-PSNR according to parameter Ξ when $\xi = 0$ and $\xi = 1$, i.e. when an extra pass of the* Refinement *algorithm is not (in blue) and is applied (in red) on* CDM_U.

arbitrary choice on the generated CDM, a complementary tool allowing fine-grained control is presented in the next section.

7.3 In-Frame Complexity Allocator for HEVC Intra Encoders

As the quad-tree decomposition of a CTU has a limited number of depth to explore, making the control of complexity of the RDO process at the CTU level rough, this section proposes to allocate the complexity more finely at the frame level. Using RD-costs to choose the CTU to constrain is briefly evoked by Correa, et al. in [CAAC16, CAdSC+13b, CAAdSC16] without defining the method and evaluating the gains. We define and assess in this section a method named Constrain the Docile CTU (CDC) that allocates the complexity in a frame while minimizing the RD performance loss.

The rest of this section is organized as follows. Section 7.3.1 analyzes the correlation between CTU partitioning depths and the RD-cost of CTU for the purpose of studying impacts of a quad-tree decomposition constraint on the RD-cost. This analysis motivates the complexity allocator introduced in Section 7.3.2.

7.3.1 Relation between CTUs partitioning depths and the RD-cost

When encoding under a bit rate constraint, an HEVC encoder must select the optimal intra prediction mode i with $i \in \{1, ..., 35\}$ that minimizes a rate-distortion criterion. As explained by Sullivan and Weerakkody [SW98], this optimization problem can be represented by an unconstrained problem using the Lagrangian method in Equation 7.1 where $J(i)$ is the RD-cost, $D(i)$ and $R(i)$ are respectively the resulting distortion and bit rate when using intra prediction mode i and λ is the Lagrangian multiplier [OR98, WSBL03]. The RDO process consists in testing all possible coding modes and selecting the one that minimizes the RD-cost:

$$\min_i J(i) = D(i) + \lambda \cdot R(i) \tag{7.1}$$

In this section, the relation between partitioning depth and RD-cost is analyzed in order to check whether an encoder can use the RD-cost to allocate the complexity within the frame.

7.3.1.1 Correlation between a CTU's partitioning depths and its RD-cost

To estimate the correlation between the partitioning depths and the RD-cost of a CTU, we formalized a novel depth metric $\mathbf{D}_{p,x,y} \in \mathbb{N}$ to quantify the partitioning depths of each CTU:

$$\mathbf{D}_{p,x,y} = \sum_{i \in [1, N_{p,x,y}]} d_i \tag{7.2}$$

where x and y are respectively the horizontal and vertical coordinates of the CTU in the frame p and $d_x \in [0, 4]$ is the depth of the considered PB with $N_{p,x,y}$ the number of PB in the CTU. $\mathbf{D}_{p,x,y}$ takes one of 687 values in the range $[0, 1024]$. A high partitioning complexity for a CTU (i.e. many small CU) implies a high value of $\mathbf{D}_{p,x,y}$. For the CTU in Figure 7.7, $\mathbf{D}_{p,x,y}$ is equal to $2 \times 1 + 6 \times 2 + 6 \times 3 + 8 \times 4 = 64$. $\mathbf{J}_{p,x,y}$ is defined as the RD-cost of the corresponding CTU.

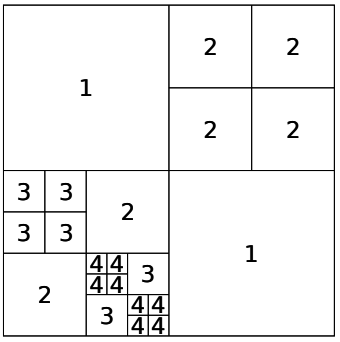


Figure 7.7 – *Example of a CTU quad-tree decomposition to illustrate the* $\mathbf{D}_{p,x,y}$ *metric.* $\mathbf{D}_{p,x,y} = 2 \times 1 + 6 \times 2 + 6 \times 3 + 8 \times 4 = 64$

To measure the correlation between $\mathbf{D}_{p,x,y}$ and $\mathbf{J}_{p,x,y}$, a normalized cross-correlation $\tilde{\phi}_{\mathbf{j}\,\mathbf{d}}(n)$ defined by Equations 7.3 and 7.4 is performed. $\mathbf{d}$ and $\mathbf{j}$ are two N length vectors with N the total number of CTU in the sequence, respectively formed by $\mathbf{D}_{p,x,y}$ and $\mathbf{J}_{p,x,y}$, reshaped in 1 dimension for each video sequence.

$$\tilde{\phi}_{\mathbf{j}\,\mathbf{d}}(n) = \frac{\phi_{\mathbf{j}\,\mathbf{d}}(n)}{\sqrt{\phi_{\mathbf{j}\,\mathbf{j}}(0)\phi_{\mathbf{d}\,\mathbf{d}}(0)}} \quad \text{, with} \tag{7.3}$$

$$\phi_{\mathbf{j}\,\mathbf{d}}(n) = \sum_{k=0}^{N-1} \mathbf{j}(k)\mathbf{d}^*(k-n), \forall n \in [-(N-1), (N-1)] \tag{7.4}$$

When the displacement $n = 0$, $\tilde{\phi}_{\mathbf{j}\,\mathbf{d}}(0)$ correspond to the normalized cross-correlation between $\mathbf{d}$ and $\mathbf{j}$ aligned to match $\mathbf{d}$ and $\mathbf{j}$ by CTU. Figure 7.8 shows $\tilde{\phi}_{\mathbf{j}\,\mathbf{d}}(0)$ for a set of 18 sequences with $QP \in \{22, 27, 32, 37, 42\}$ (from high to low quality).

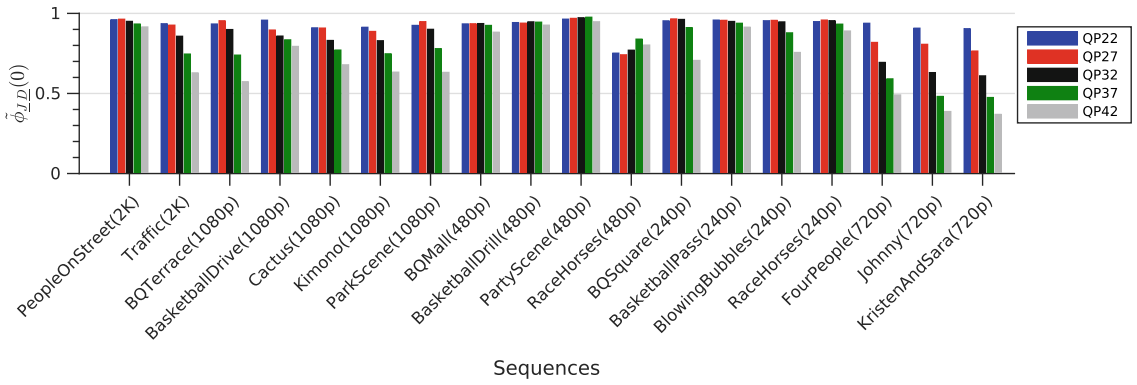


Figure 7.8 – *Correlation coefficient between the CTU depth metric* **d** *and the RD-cost* **j**

A high correlation is observed between the RD-cost **j** and the depth metric **d** for most of the configurations. The degree of correlation decreases when the QP increases. This effect is more prominent in the sequences of class E [Bos13]. These results lead to the conclusion that a strong link exists between the RD-cost and the partitioning depths of the CTU in HEVC which slightly decreases at low bitrate. Intuitively, CTU with low entropy have low RD-cost (they require low information to be coded at low distortion) and lead to coarse grain tree partitioning. We call these CTU "docile CTU".

7.3.1.2 Impacts of a CTU constraint on the RD-cost

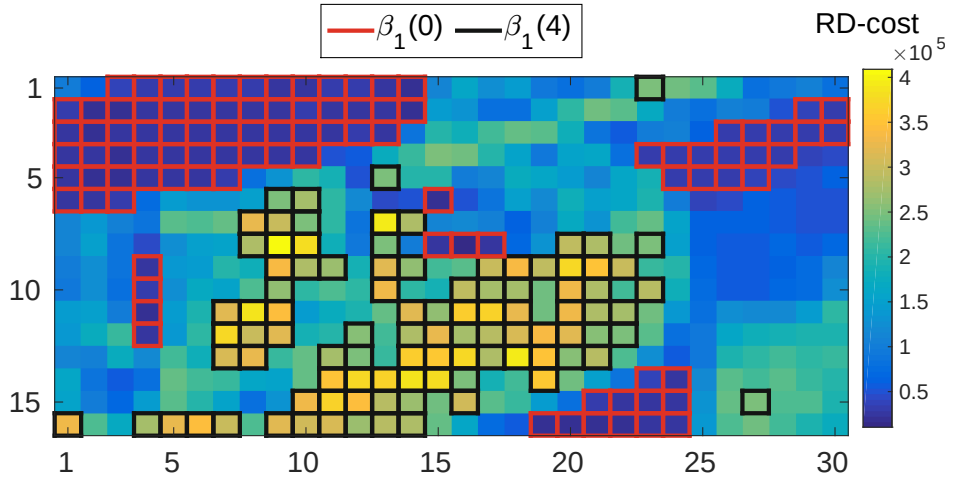
In the next sections, as the major part of quad-tree partitioning complexity reduction techniques presented in Section 3.2.1 used *Early Termination* techniques, a CTU is constrained by preventing the use of depth level $d = 4$, i.e. the last tested depth. This constraint is used as an example of a local complexity reduction technique. Since the RD-cost and the partitioning depths of a CTU are correlated, this section analyzes the impact of the CTU partitioning constraint on the RD-cost. Two encodings are performed for that purpose: one with constraint (lowering max depth from $d_{max} = 4$ to 3) in the RDO process and one without any constraint (anchor). The RD-cost obtained for both constrained and anchor encodings, $\mathbf{J}'_{p,x,y}$ and $\mathbf{J}_{p,x,y}$ respectively, are compared.

Each frame is split into bins, grouping CTU with similar RD-costs, each bin including the same number of CTU. The following equation defines the interval of bin $\beta_p(k)$:

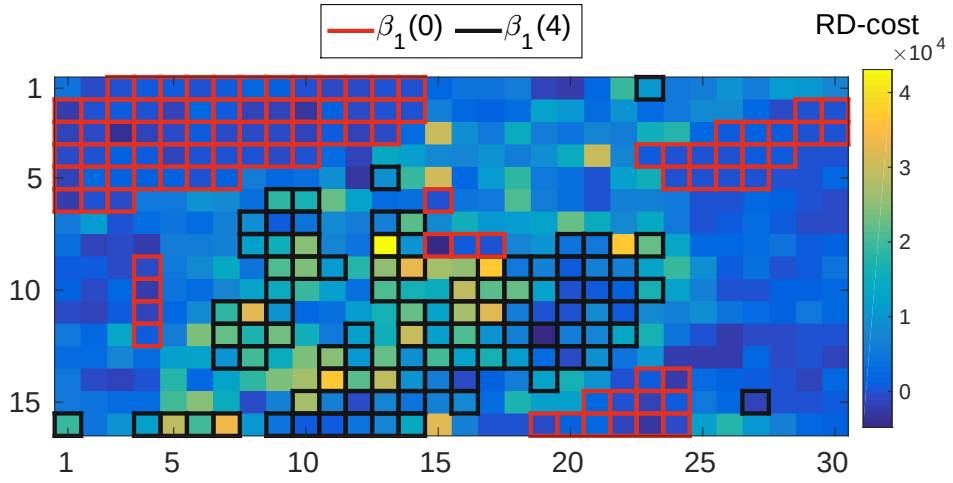
$$\beta_p(k) = \left[F_p^{-1}\left(\frac{k}{N_b}\right), F_p^{-1}\left(\frac{k+1}{N_b}\right) \right], \text{ with } k \in [0, N_b - 1] \tag{7.5}$$

where F_p is the cumulative distribution function of the RD-cost $J_{p,x,y}$ for frame p and N_b is the number of bins (uniform CTU intervals). To illustrate the impact of the CTU constraint on the RD-cost, Figure 7.9 presents the RD-cost of the 1^{st} frame of the *BQTerrace* sequence (in 1080p) coded in the AI configuration at QP32.

Figure 7.9a is a CTU heat map of the RD-costs. These costs are split into 5 bins ($N_b = 5$), with $\beta_1(0)$ and $\beta_1(4)$ (i.e. the bins including CTU with respectively lowest and highest RD-costs) being highlighted. Figure 7.9b shows the difference $\mathbf{J}_{1,x,y} - \mathbf{J}'_{1,x,y}$, representing the impact of the CTU constraint. The RD-cost increase due to removing the constraint is not uniformly distributed in the frame: it is much higher for $\beta_1(4)$ than it is for $\beta_1(0)$. As a consequence, the RD-cost of docile CTU is less impacted by a constraint than the RD-cost of the non docile CTU.



(a) *Absolute RD-cost per CTU with unconstrained encoding* $\mathbf{J}_{1,x,y}$ *of the* 1^{st} *frame of the* BQTerrace *sequence coded in the AI configuration at QP32.*



(b) *Relative RD-cost gain per CTU when applying a constraint* $\mathbf{J}_{1,x,y} - \mathbf{J}'_{1,x,y}$ *of the* 1^{st} *frame of the* BQTerrace *sequence coded in the AI configuration at QP32.*

Figure 7.9 – *Heat maps of the RD-cost of the first frame of the sequence BQTerrace(1080p) with* $QP = 32$.

The total RD-cost increase $\delta(k)$ for bin k and for all the sequences is defined by Equation 7.6. It is used to quantify the impact of the CTU constraint on the sequences.

$$\delta(k) = \sum_{p,x,y} \left(\mathbf{J}_{p,x,y} - \mathbf{J}'_{p,x,y}\right), \forall \mathbf{J}_{p,x,y} \in \beta_p(k) \tag{7.6}$$

Figure 7.10 shows $\delta(k)$ for all sequences of class C encoded with $N_b = 7$. The figure presents the results for QP=32, other values were also tested and show similar behavior. The histograms show that $\delta(k)$ follows a monotonically increasing curve. For the constrained CTU, the RD-cost increase is thus minimal for docile CTU, i.e. CTU with low RD-cost in unconstrained encoding.

Further results that consider a wider range of QP values (27, 32 and 37) are presented in Table 7.1. The number of bins is fixed to 3. Table 7.1 presents results for the sequences

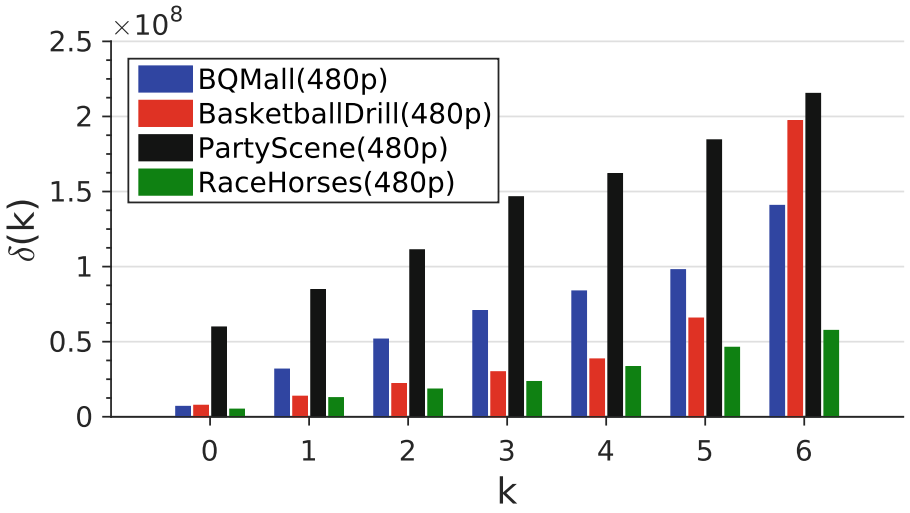


Figure 7.10 – *RD-cost increase per bin when constraint is removed with $Nb = 7$ bins*

Traffic, Cactus, RaceHorses, BQSquare, FourPeople (one per class). The cost increase for bin k is normalized to $\delta(0)$:

$$\tilde{\Delta}_{0,k} = \frac{\delta(k) - \delta(0)}{\delta(0)} \times 100 \tag{7.7}$$

$\tilde{\Delta}_{0,1}$ and $\tilde{\Delta}_{0,2}$ represent the increase (in %) of RD-cost between the first bin $\delta(0)$ and respectively the second $\delta(1)$ and third $\delta(2)$ bins. The results show that in most cases the constraint has a larger impact on the RD-cost for the last bin than for the second one, for every sequence and QP.

Table 7.1 – *Increase cost (%) by bin for $N_b = 3$*

Class	Sequence	QP 27		QP 32		QP 37	
		$\tilde{\Delta}_{0,1}$	$\tilde{\Delta}_{0,2}$	$\tilde{\Delta}_{0,1}$	$\tilde{\Delta}_{0,2}$	$\tilde{\Delta}_{0,1}$	$\tilde{\Delta}_{0,2}$
A	Traffic	237	596	259	723	214	591
B	Cactus	162	671	135	383	1028	1522
C	RaceHorses	165	278	207	408	247	568
D	BQSquare	169	647	890	3059	116	479
E	FourPeople	13	127	108	135	1015	826

These experiments show that the CTU with lowest RD-cost have less increase of bit rates and/or distortion if constrained encoding is applied to them. As a conclusion, when a CTU complexity reduction technique has to be applied on a part of a frame, CTU with the lowest RD-cost should be constrained, i.e. less encoding effort should be spent to encode them.

7.3.1.3 Temporal RD-cost stability of consecutive frames

In previous section, the RD-cost of unconstrained encoding is used to choose which CTU to constrain. Yet, an encoder applying complexity reduction techniques does not compute unconstrained RD-cost. Consequently, RD-cost prediction should be performed from existing information. A precise distribution modeling of RD-costs is a challenging problem

because RD-cost distribution depends on both QP values and video content such as texture complexity, noise, and irregular illumination.

The RD-cost of the previous frame in the video can however be used to predict the RD-cost of the current frame. To justify the RD-cost temporal stability, a normalized auto-correlation (as in Equation 7.3) is performed on every vector $\mathbf{j}_{x,y}$ formed of the RD-cost of the CTU located at coordinates i, j for the entire sequence. To observe the temporal stability, the correlations are taken at $n = 1$ which correspond to frame p being correlated to frame $p - 1$. The average of every correlation coefficient $\tilde{\phi}_{\mathbf{jj}}(1)$ is performed as shown in Equation 7.8.

$$\overline{\phi}_{\mathbf{jj}} = \underset{x,y}{mean}(\tilde{\phi}_{\mathbf{j}_{x,y}\mathbf{j}_{x,y}}(1)) \tag{7.8}$$

The results are presented in Table 7.2 considering five values of QP (22, 27, 32, 37 and 42) and averaged by class. A high correlation (with correlation coefficient higher than 0.95) between RD-cost on consecutive frames is observed. It is thus justified to use the RD-cost of the previous frame to predict the RD-cost of the current one. Furthermore, the Constrain the Docile CTU (CDC) allocator does not need the exact RD-cost values and approximations are sufficient for comparison against dynamic thresholds.

Table 7.2 – *Average of correlation coefficient of CTU cost of consecutive frames*

	QP22	QP27	QP32	QP37	QP42	**Av.**
Class A	0.991	0.989	0.987	0.985	0.983	**0.987**
Class B	0.988	0.987	0.986	0.985	0.985	**0.986**
Class C	0.986	0.986	0.986	0.985	0.985	**0.986**
Class D	0.985	0.985	0.985	0.985	0.985	**0.985**
Class E	0.986	0.986	0.986	0.986	0.986	**0.986**
Average	**0.987**	**0.987**	**0.986**	**0.985**	**0.985**	**0.986**

The next part describes the CDC allocation method to improve RD performance of HEVC.

7.3.2 The CDC Complexity Allocator

The previous analysis leads to three observations:

1. RD-cost is linked to the partitioning depths of CTU.

2. CTU with low RD-cost have less increase of bit rates and/or distortion than CTU with high RD-cost when constrained (i.e. when the partitioning depth is limited).

3. The RD-costs of the previous frame can be used to predict the RD-costs of the current frame.

Motivated by these observations, we propose a method to allocate complexity in a frame.

7.3.2.1 Complexity Allocator Presentation

The proposed complexity allocator can be adapted to different CTU complexity reduction techniques that require the HEVC encoder to constrain X_p% of the frame p.

Initially, the first frame of the video sequence is encoded unconstrained and the RD-cost of CTU of that frame is stored. Then, for each constrained frame, the CDC method constrains CTU which RD-cost belongs to the following interval:

$$J_{p-1,x,y} \in \left[0; F_{p-1}^{-1}(X_p/100)\right[\tag{7.9}$$

where F_{p-1} is the cumulative distribution function of the RD-cost $J_{p-1,x,y}$ of the frame $p-1$ and X_p is the percentage of constrained CTU in the current frame p. CDC is called "Constrain the Docile CTUs" because it consists of reducing the encoding effort for the CTU that lend themselves the most to encoding.

The quantization step of the CDC complexity allocator depends on the number of CTU per frame and thus the resolution of the video sequence. Figure 7.11 shows the quantization step of the CDC complexity allocator according to the number of pixel per frame for each resolution.

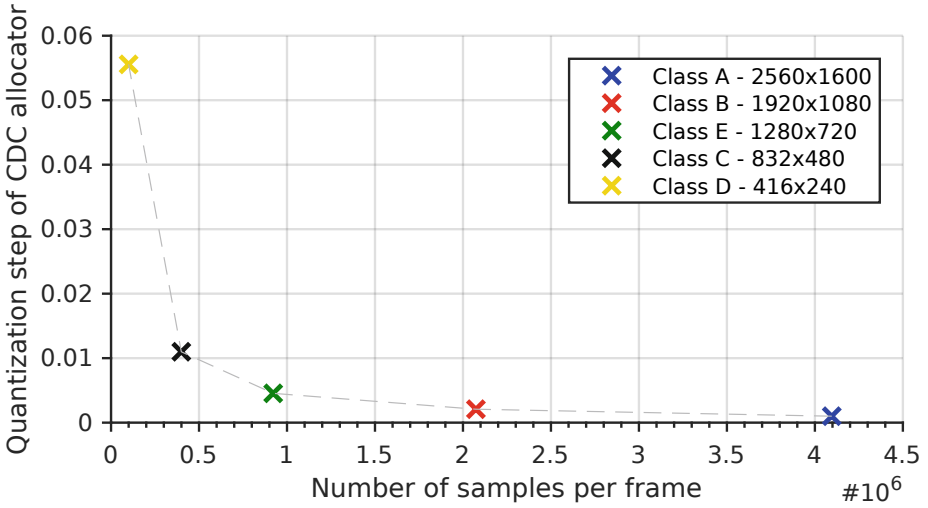


Figure 7.11 – *Quantization step of the CDC complexity allocator according to the number of pixel per frame.*

Video sequences with the lowest resolution (416×240) include 18 full CTU and have a quantization step of 0.056 while video sequences with the highest resolution (2560×1600), including 1000 full CTU, have a quantization step of 0.001.

7.3.2.2 Experimental Results

The performance of the CDC allocator is evaluated by measuring the BD-BR between CDC and four methods described below:

1. *Upper*: The first X_p% of the CTU in raster scan order of the frame p are constrained.
2. *Lower*: The last X_p% of the CTU in raster scan order of the frame p are constrained.
3. *Tick*: every CTU out of a percentage (for example $X_p = 33\%$ means every three CTU) is constrained.

4. *Inverse*: The exact inverse of our allocator method, i.e. the CTU with the highest RD-cost in the previous frame are constrained.

CDC and the four reference allocators are implemented in the real-time HEVC encoder *Kvazaar*. CTU complexity reduction is implemented by removing the last depth level ($d = 4$) in the RDO process on the constrained CTU. The proposed method performance is evaluated with three shares of constrained CTU per frame (30%, 50% and 70%) and four different QP values (22, 27, 32 and 37). Table 7.3 presents the BD-BR between CDC and the four others for 18 standard sequences. For all the tests, CDC has better RD performance with a negligible memory footprint and a very low computational overhead. CDC saves a significant amount of bitrate compared to the *inverse* one, which confirms the approach.

7.3.3 Tunable Complexity Frame Encoding

Using the CDC complexity allocator, to determine the Ξ parameter of the *Tunable Complexity CTU Encoding*, our system is able to select a real number[1] of depth levels by frame $\hat{\Xi} \in [0, 4]$. Figure 7.12 presents the hierarchical block diagram of the *Tunable Complexity Frame Encoding*.

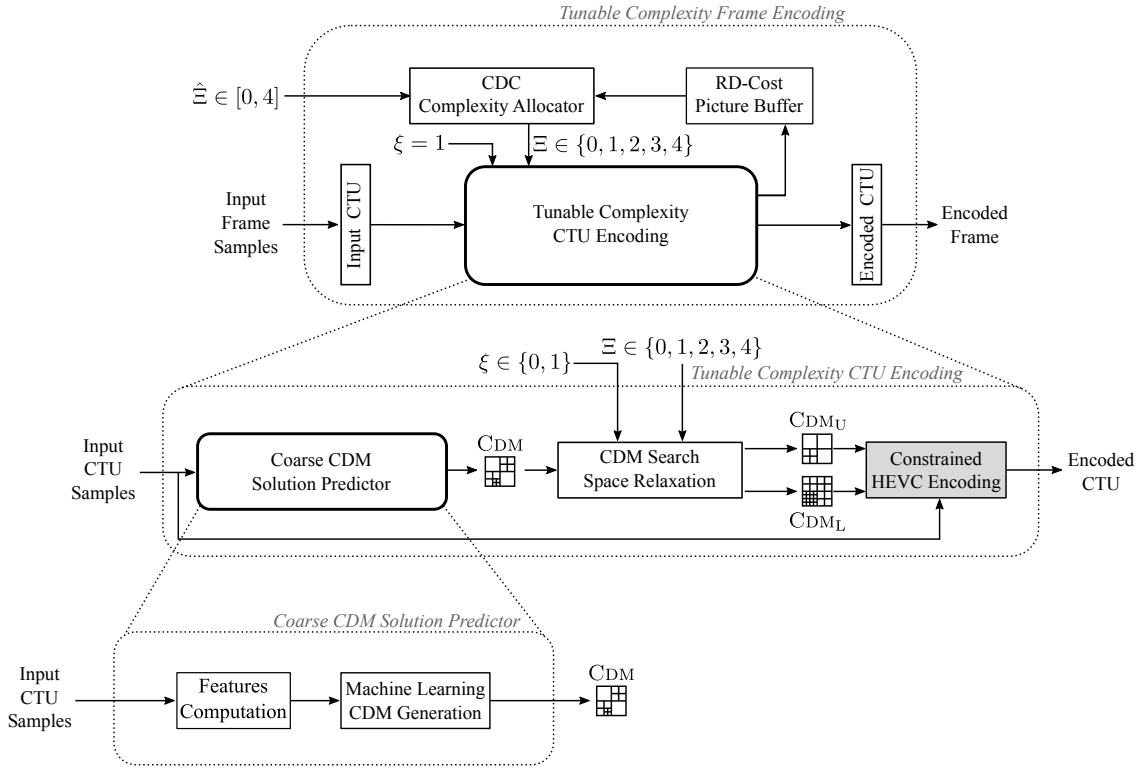


Figure 7.12 – *Tunable complexity frame encoding scheme.*

The *Tunable Complexity Frame Encoding* takes a frame and the couple of parameters $(\hat{\Xi}, \xi)$ as input and constrain the encoding process to test the real number of depth levels $\hat{\Xi} \in [0, 4]$ in the input frame. The CDC complexity allocator splits the real number $\hat{\Xi}$ in two parts: the integer part and the fractional part. The integer part of $\hat{\Xi}$ sets the minimum number of depth levels $\lfloor \hat{\Xi} \rfloor$ tested for all CTU of the input frame. The fractional part of $\hat{\Xi}$

[1] number composed of an integer part and a fractional part

Table 7.3 – *BD-BR between our CDC allocator and four reference allocators (in %). The proposed CDC allocator is better than other allocators in all cases.*

Class	Sequences	Upper			Lower			Tick			Inverse		
		30%	**50%**	**70%**	**30%**	**50%**	**70%**	**30%**	**50%**	**70%**	**30%**	**50%**	**70%**
Class A	Traffic	0.134	0.408	0.684	0.500	1.024	1.038	0.680	0.813	0.810	1.278	1.498	1.212
	PeopleOnStreet	1.306	1.702	1.573	0.638	0.955	0.930	1.380	1.523	1.495	2.472	2.816	2.262
Class B	ParkScene	0.326	0.459	0.266	0.701	0.822	0.640	0.728	0.744	0.576	1.073	1.334	1.000
	Cactus	0.814	1.588	1.684	0.858	1.261	1.717	1.115	1.522	1.867	2.666	2.886	2.544
	BQTerrace	0.818	1.175	0.692	1.861	1.872	1.723	1.382	1.757	1.659	2.769	3.174	2.570
	BasketballDrive	0.467	0.470	0.628	0.674	0.949	0.825	1.026	0.829	0.861	1.422	1.512	1.342
Class C	RaceHorses	0.167	0.828	1.351	1.503	2.258	2.554	1.386	1.649	1.860	2.746	3.097	2.667
	BQMall	0.973	1.302	1.007	1.274	1.504	1.372	1.752	1.644	1.515	2.276	2.720	2.159
	PartyScene	1.025	1.117	1.005	3.514	3.661	3.525	2.224	2.878	3.449	4.588	4.941	4.482
	BasketballDrill	1.155	1.761	1.913	1.816	2.507	2.618	2.367	2.344	2.185	3.714	4.239	3.511
Class D	RaceHorses	0.326	0.965	1.515	2.003	2.682	2.910	1.659	1.916	1.787	2.783	3.190	2.752
	BQSquare	2.495	3.798	3.205	2.827	2.963	3.346	4.011	3.642	3.147	4.749	5.527	4.601
	BlowingBubbles	0.908	1.469	1.205	1.040	1.055	1.541	1.444	1.485	1.204	1.713	2.068	1.762
	BasketballPass	1.028	2.480	3.436	2.601	3.714	4.114	2.299	2.944	3.530	4.736	5.489	4.734
Class E	FourPeople	0.261	1.098	0.958	2.280	3.133	2.953	2.041	2.457	1.999	3.456	4.215	3.200
	Johnny	2.107	2.919	4.073	1.018	2.847	2.956	2.299	2.756	3.880	5.313	5.791	5.056
	KristenAndSara	1.398	2.317	2.983	4.054	5.318	5.662	2.845	3.876	5.357	7.121	7.780	6.952
	Average	0.891	1.462	1.580	1.659	2.186	2.281	1.743	1.976	2.098	3.108	3.534	2.989

sets the number of CTU that tests one more depth level $\lceil\hat{\Xi}\rceil$. The CDC complexity allocator uses the RD-cost picture buffer of the previous frame to select which CTU test $\lfloor\hat{\Xi}\rfloor$ or $\lceil\hat{\Xi}\rceil$ depth levels according to the protocol defined in Section 7.3.2.1. The CTU obtaining the highest RD-costs in the previous image are forced to test $\lceil\hat{\Xi}\rceil$ depth levels, the rest of the CTU are forced to test $\lfloor\hat{\Xi}\rfloor$ depth levels. Finally, the CDC complexity allocator sets the appropriate Ξ parameter value to the *Tunable Complexity CTU Encoding* for the current CTU.

7.4 Results of the Tunable Complexity Frame Encoding Scheme

The following section gives the experimental results obtained for the proposed complexity reduction scheme on a real time HEVC encoder. The experimental set-up has been detailed in Section 4.3.1. Results presented in this section are average results obtained for the 18 video sequences presented in Table 4.1.

7.4.1 Testing the performance of Co-located Depth Features on Quad-Tree Prediction

Section 7.2.1.1 has presented a first evaluation of the impact of depth features used for the one-shot quad-tree partitioning prediction within the *Tunable Complexity CTU Encoding* (according to Ξ parameter values). To complete the results presented by Figure 7.4 in Section 7.2.1.1, Figure 7.13 presents the results in term if BD-BR according to the $\hat{\Xi}$ (with $\xi = 0$) when the *Coarse* CDM *Solution Predictor* uses $\mathcal{F}^d_{x,y}$ and $\mathcal{F}2^d_{x,y}$ for the one-shot quad-tree partitioning prediction.

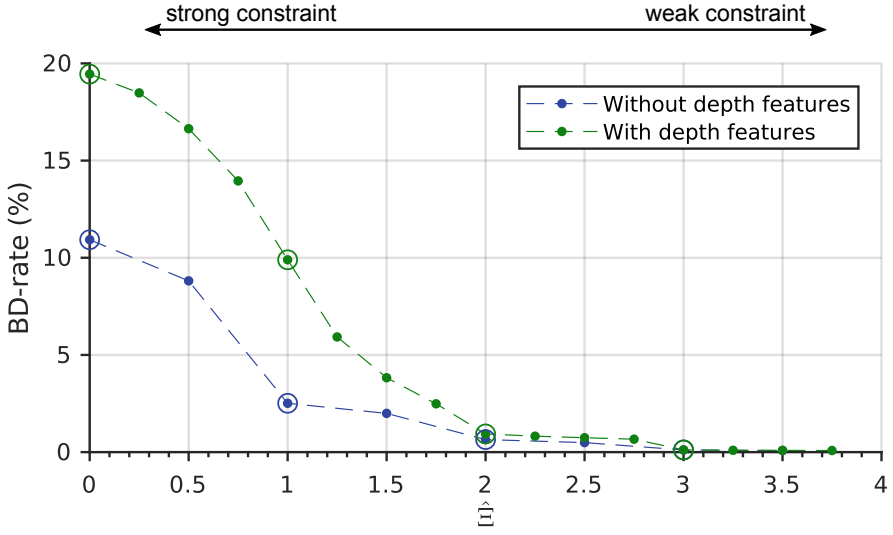


Figure 7.13 – *BD-BR according to* $\hat{\Xi}$ *when the* Coarse CDM Solution Predictor *uses* $\mathcal{F}^d_{x,y}$ *or* $\mathcal{F}2^d_{x,y}$*, i.e. when the* Coarse CDM Solution Predictor *uses (in blue) or does not use (in green) the depth features.*

Results show that for $\hat{\Xi}$ from 0 to 3, the features vector $\mathcal{F}2^d_{x,y}$ causes more degradation in terms of BD-BR than using $\mathcal{F}^d_{x,y}$, up to 8.52% of BD-BR increase for $\hat{\Xi} = 0$. For $\hat{\Xi}$ from 3 to 4, the BD-BR degradations of the two versions are very close and the encoding

performance can be considered equal. The results leads to conclude that error propagation due to the quad-tree constraints of the neighboring CTU is too important and thus the depth features can not be used for the one-shot quad-tree partitioning prediction regardless the degree of depth level constraint. In the rest of this section, the features vector $\mathcal{F}^d_{x,y}$ is used in the *Coarse* CDM *Solution Predictor* to predict the first CDM prediction.

7.4.2 Accuracy Evaluation of the CDM Interval Prediction

Let CDM P be the CDM obtained after constraining the RDO process between CDM_U and CDM_L, i.e. the best quad-tree partitioning included in the interval of predicted CDM and the reference CDM R be the CDM generated by a full RDO process (optimal), i.e. when the RDO process realizes an exhaustive search leading to the optimal solution.

To analyze the prediction accuracy of the CDM interval (composed by CDM_U and CDM_L) generated by the *Tunable Complexity Frame Encoding*, the recall $\rho(P,R)$, whose gives the percentage of perfect prediction in terms of pixel area, is decomposed into two independent recalls (as the distance $\Gamma(P,R)$ in Section 5.4.3): $\overline{\rho(P,R)}$ and $\underline{\rho(P,R)}$ complying with Equation (7.10):

$$\rho(P,R) = 100 - (\overline{\rho(P,R)} + \underline{\rho(P,R)}) \tag{7.10}$$

where $\overline{\rho(P,R)}$ (respectively $\underline{\rho(P,R)}$) is called *upper recall* (respectively *lower recall*) and is the percentage of bad prediction in terms of pixel area only when the depth of CDM P is lower (respectively higher) than the depth of CDM R, as described by Equation (7.11) (respectively Equation (7.13)). $\overline{\rho(P,R)}$ is called the *upper recall* as it represents the fact that the quad-tree decomposition of CDM P is shallower than the one of R, i.e the CDM of P has lower depth values. The same reasoning can be applied to $\underline{\rho(P,R)}$.

$$\overline{\rho(P,R)} = \left[\sum_{i=0}^{7}\sum_{j=0}^{7}\overline{\delta}(P(i,j) - R(i,j))\right] \times \frac{1}{64}, \tag{7.11}$$

$$\text{with } \overline{\delta}(x) = \begin{cases} 0 & \text{if } x \geq 0 \\ 1 & \text{if } x < 0 \end{cases} \tag{7.12}$$

$$\underline{\rho(P,R)} = \left[\sum_{i=0}^{7}\sum_{j=0}^{7}\underline{\delta}(P(i,j) - R(i,j))\right] \times \frac{1}{64}, \tag{7.13}$$

$$\text{with } \underline{\delta}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0 \end{cases} \tag{7.14}$$

The 6 following summary metrics are used in this section to evaluate the prediction accuracy of the CDM interval (composed by CDM_U and CDM_L) generated by the *Tunable Complexity Frame Encoding*:

- the average of $\rho(P,R)$ measurements, defined in Section 6.4.3.1, gives the percentage of perfect prediction in terms of pixel area, it falls between 0% and 100% and the more $\rho(P,R)$ is close to 100%, the more the predicted CDM accurately fit the reference one,
- the average of $\overline{\rho(P,R)}$ measurements, defined previously, gives the percentage of bad prediction in terms of pixel area when the depth of P is lower than the depth of R, it falls between 0% and 100% and the more $\overline{\rho(P,R)}$ is close to 0%, the more the predicted CDM accurately fit the reference one,

- the average of $\underline{\rho(P,R)}$ measurements, defined previously, gives the percentage of bad prediction in terms of pixel area when the depth of P is higher than the depth of R, it falls between 0% and 100% and the more $\underline{\rho(P,R)}$ is close to 0%, the more the predicted CDM accurately fit the reference one,
- the average distance $\Gamma(P,R)$, defined in Section 5.4.1 represents the mean error in terms of depth between the predicted CDM P and the reference one R, it is in $[0,4]$, the more $\Gamma(P,R)$ is close to 0, the more accurate the predicted CDM P becomes,
- the average $\overline{\Gamma(P,R)}$, defined in Section 5.4.3, represents the fact that the quad-tree decomposition of the predicted CDM P has lower depth values than the reference CDM R,
- the average $\underline{\Gamma(P,R)}$, defined in Section 5.4.3, represents the fact that the quad-tree decomposition of the predicted CDM P has upper depth values than the reference CDM R.

Figures 7.14 and 7.15 present respectively the distances $\Gamma(P,R)$, $\overline{\Gamma(P,R)}$, $\underline{\Gamma(P,R)}$ and recalls $\rho(P,R)$, $\overline{\rho(P,R)}$, $\underline{\rho(P,R)}$ according to the parameter $\hat{\Xi}$ when $\xi = 0$, averaged across the four $QP \in \{22, 27, 32, 37\}$ values and the 18 sequences presented by Table 4.1 in Section 4.3.1.1. The results evaluate the accuracy of the CDM interval generated by the *Tunable Complexity Frame Encoding*.

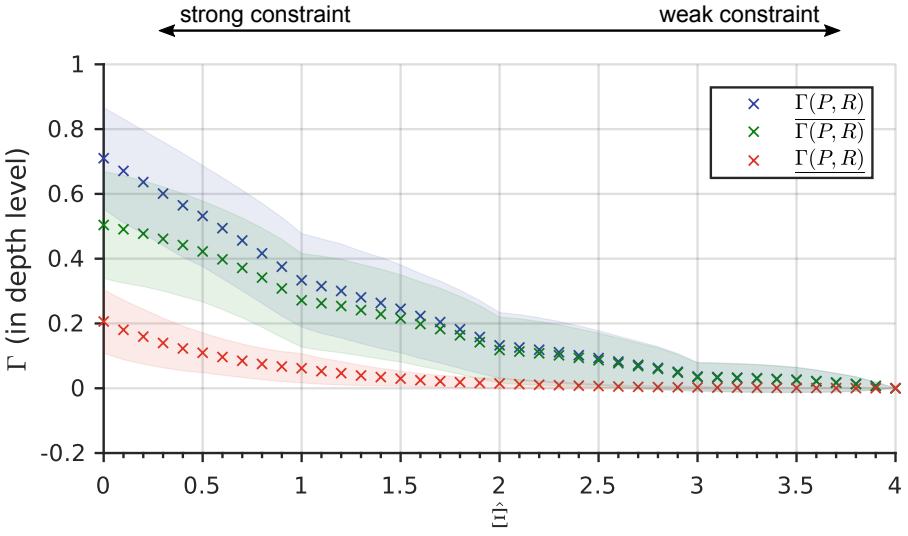


Figure 7.14 – *Averages and standard deviations of* $\Gamma(P,R)$ *in blue,* $\overline{\Gamma(P,R)}$ *in green and* $\underline{\Gamma(P,R)}$ *in red between the predicted* CDM P *and the reference* CDM R *generated by a full RDO process versus* $\hat{\Xi}$*. The color areas represent the standard deviation of the according metrics.*

Results in Figure 7.14 show that the distance Γ in depth level between P and R is not equally distributed between $\overline{\Gamma(P,R)}$ and $\underline{\Gamma(P,R)}$. The main part of the error in terms of distance Γ is composed by *upper distances*. In other words, when the interval of CDM does not include the optimal quad-tree partitioning, the optimal partitioning is composed by larger CU blocks. The *lower distance* even tends very quickly towards 0, especially when $\hat{\Xi}$ is higher than 2. Figure 7.15 confirms the previous results showing that the main part of the errors are due to miss prediction of big CU blocks. For $\hat{\Xi} = 0$, 48% of CTU quad-tree partitioning are not included in the interval of CDM in terms of pixel area of which 33%

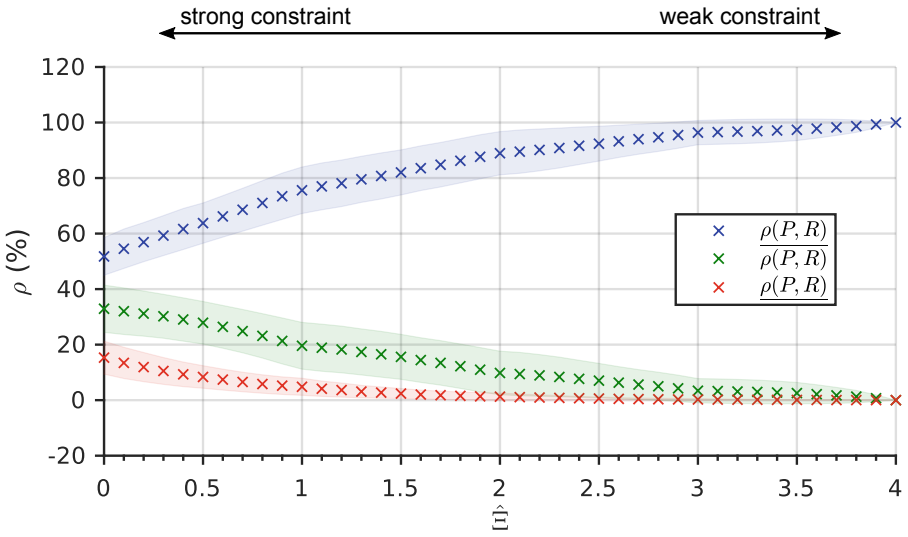


Figure 7.15 – *Averages and standard deviations of $\rho(P,R)$ in blue, $\overline{\rho(P,R)}$ in green and $\underline{\rho(P,R)}$ in red between the predicted* CDM *P and the reference* CDM *R generated by a full RDO process versus $\hat{\Xi}$. The color areas represent the standard deviation of the according metrics.*

are too small CU blocks predicted. The next section presents the results of the *Tunable Complexity Frame Encoding* in terms of encoding degradation and complexity reduction.

7.4.3 Results of Encoding Degradation and Complexity Reduction of the Tunable Complexity Frame Encoding

Figures 7.16 and 7.17 show the results in terms of BD-BR and BD-PSNR according to the complexity consumed for $\hat{\Xi} \in [0,4]$ and $\xi = \{0,1\}$, i.e. when an extra pass of the *Refinement* algorithm is not and is applied on CDM_U respectively. These results extend results presented in Section 7.2.1.2 by Figures 7.5 and 7.6 where the parameter Ξ could only take integer values (circle points in Figures 7.16 and 7.17). The complexity values include the complexity overhead due to the entire complexity reduction scheme as the features computation. Results of Figures 7.16 and 7.17 are the average results across the 18 video sequences detailed in Table 4.1 following the experimental set-up described in Section 4.3.1.

First of all, the results of Figures 7.16 and 7.17 show that applying an extra pass of *Refinement* algorithm consumes between 6% ($\hat{\Xi} = 0$) and 4.7% ($\hat{\Xi} = 2.5$) more complexity with non negligible gains of encoding performance. When $\hat{\Xi} \geq 3$, the results for $\xi = 0$ and $\xi = 1$ are strictly the same in terms of encoding degradation, the extra pass of *Refinement* algorithm being not applied as explained in Section 7.2. The slight shifting in terms of complexity between the results for $\xi = 0$ and $\xi = 1$ and $\hat{\Xi} \geq 3$ is due to the fuzziness of complexity profiling but does not exceed 0.6%. The gains obtained by enabling the parameter ξ are between:

- -7.33% ($\hat{\Xi} = 0$) and -0.28% ($\hat{\Xi} = 2.5$) in terms of BD-BR,
- +0.36dB ($\hat{\Xi} = 0$) and +0.012dB ($\hat{\Xi} = 2.5$) in terms of BD-PSNR.

These results confirm the interest of setting the parameter $\xi = 1$ to limit the encoding degradation of our complexity reduction scheme. On the other hand, setting ξ equal to 1 reduces the maximal complexity reduction achievable from 33% to 39%.

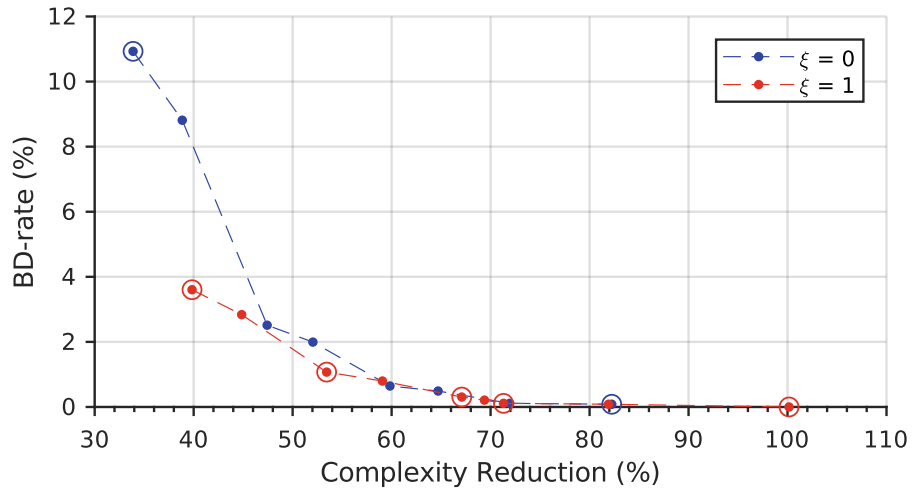


Figure 7.16 – *BD-BR according to parameter $\hat{\Xi}$ when $\xi = 0$ and $\xi = 1$, i.e. when an extra pass of the* Refinement *algorithm is not (in blue) and is applied (in red) on* CDM_U.

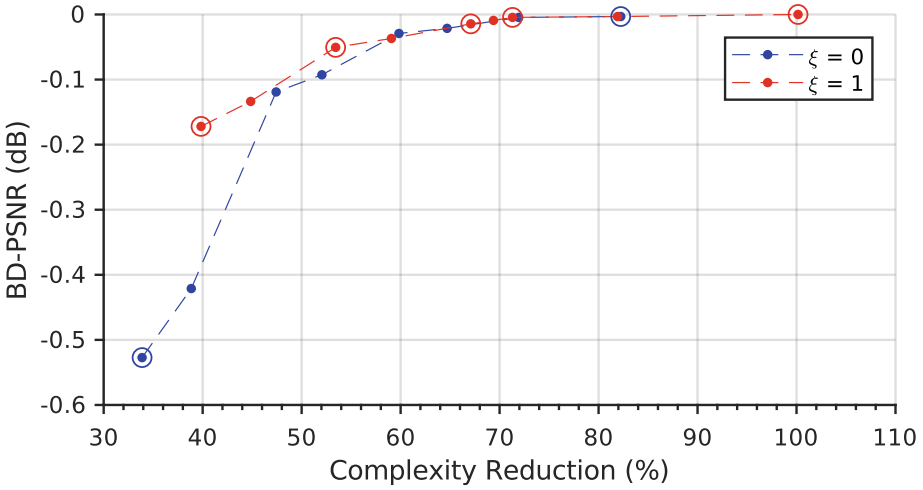


Figure 7.17 – *BD-*PSNR *according to parameter $\hat{\Xi}$ when $\xi = 0$ and $\xi = 1$, i.e. when an extra pass of the* Refinement *algorithm is not (in blue) and is applied (in red) on* CDM_U.

Figure 7.18 presents the complexity according to the parameter $\hat{\Xi} \in [0, 4]$ with $\xi = 1$. The results are the average of complexity across the 18 different sequences belonging to 5 classes (A, B, C, D and E) presented in Table 4.1 for four QP values: 22, 27, 32, 37. The blue area on Figure 7.18 represents the standard deviation of the complexity across all the sequences and QP values.

The results show that the complexity can be considered as following a piecewise linear function across $\hat{\Xi}$ with four pieces: for $\hat{\Xi} \in [0, 1]$, $\hat{\Xi} \in [1, 2]$, $\hat{\Xi} \in [2, 3]$ and for $\hat{\Xi} \in [3, 4]$. Moreover, it is interesting to notice that the standard deviation for all $\hat{\Xi}$ values is very low, close to 2.5% in average. This observation will be important in the next section to use the parameter $\hat{\Xi}$ to control the system under a specific complexity target.

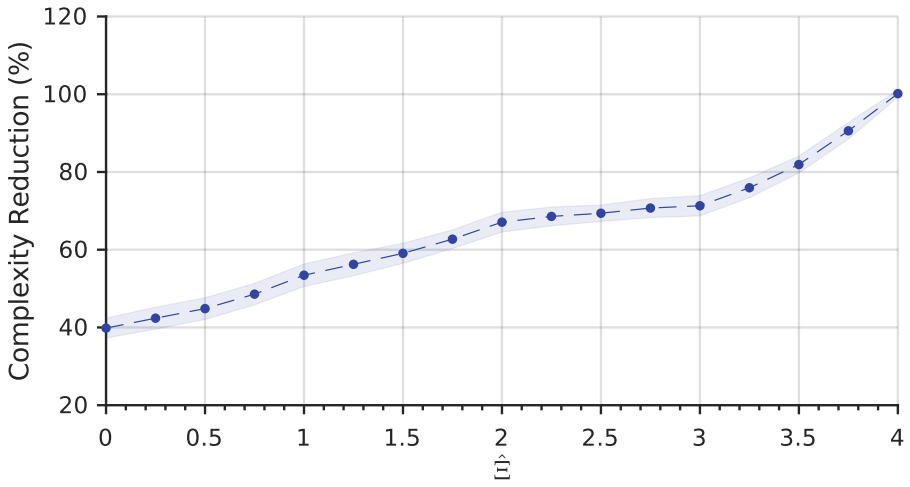


Figure 7.18 – *complexity consumed versus* $\hat{\Xi}$ *with* $\xi = 1$. *The blue area shows the standard deviation of the energy consumed across the all sequences and QP values.*

7.4.4 Comparison of our Statistical and Tunable Complexity Frame Encoding scheme

To conclude this section, Figure 7.19 details the performance of the two proposed tunable complexity reduction schemes of this document: the statistical approach detailed in Section 5.5.1 and the *Tunable Complexity Frame Encoding*. The statistical complexity reduction scheme exploits the correlation between CTU partitioning, its texture complexity and its luminance samples variance to predict an interval of CDM using a couple of parameters $(\underline{\Delta}, \overline{\Delta})$ which offer a trade-off between energetic gains and compression efficiency (see Section 5.4.3 and 5.5.1). The new scheme presented in this section uses ML approach to predict CTU quad-tree decomposition in one-shot and then searches around this prediction using the CDM *Search Space Relaxation* described in Section 7.2. Figure 7.19 shows the

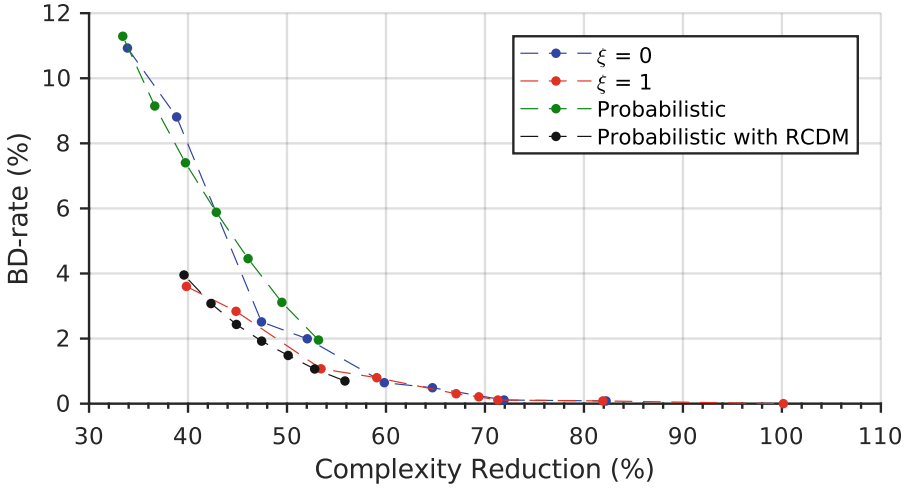


Figure 7.19 – *BD-BR of the two tunable complexity reduction schemes: the statistical approach and the* Tunable Complexity Frame Encoding *with and without the extra* Refinement *algorithm pass.*

results of the two tunable complexity reduction schemes, when the *Refinement* algorithm is not or is applied on the upper CDM of the interval.

First of all, it is interesting to notice that the results of the two schemes without the extra *Refinement* algorithm pass have the same order of magnitude. Figure 7.19 shows that when the *Refinement* algorithm is applied on the upper CDM of the interval, the two proposed tunable complexity reduction schemes are increased with the same gap: from the green curve to the black curve and from the blue curve to the red curve for the statistical and the *Tunable Complexity Frame Encoding* scheme, respectively.

On the other hand, the *Tunable Complexity Frame Encoding* scheme is able to reduce the complexity from the MEP to 100% which is not possible to achieve with the statistical scheme due to the limitation of the approach parameters. In the next section, the *Tunable Complexity Frame Encoding* will be used with $\xi = 1$ to control the complexity of the encoding process under a specific target using control systems engineering.

7.5 Conclusion

This chapter has firstly presented the CDM *Search Space Relaxation* which generates an interval of CDM around an input CDM according to a selected integer number of depth levels. The proposed CDM *Search Space Relaxation* scheme is used in the *Tunable Complexity CTU Encoding* to limit and scale the complexity of the HEVC Intra encoding process setting an integer number of depth levels tested by RDO process for a given CTU.

This chapter has secondly presented the *Tunable Complexity Frame Encoding* scheme which, using the CDC complexity allocator, is able to finely scale the complexity by frame of the Intra encoding process up to the MEP. The *Tunable Complexity Frame Encoding* scheme uses the CDC complexity allocator to select a real number of depth levels by frame tested by the RDO process.

The next chapter presents a real-time control system that dynamically manages the encoding complexity under a specific target. The proposed solution will use the *Tunable Complexity Frame Encoding* scheme and a feedback loop mechanism to limit the encoding time by frame under a selected target in second.

CHAPTER 8

Complexity Control for HEVC Intra Encoding

8.1 Introduction

The main goal of this chapter is to use the *Tunable Complexity Frame Encoding* presented in the previous section to build a real-time control system that dynamically manages the encoding process to keep the encoding "complexity", in the broadest sense, under a specific target. By "complexity", we imply as well energy in joule as time in seconds or number of clock cycles. As this control system is used as a showcase for demonstrating the utility and the applicability of the major contributions of this document, we choose to design a control system able to target a specific time by frame for the encoding process. Nevertheless, the design of the control system does not depend on this target metric, only settings would change if the metric was changed.

8.2 Control System Design

The goal of the control system is to set the right value of the parameter $\hat{\Xi} \in [0, 4]$ to manage the complexity of the current encoding frame according to the target encoding time, in second. The parameter $\hat{\Xi} \in [0, 4]$ corresponds to the number of depth levels tested during the RDO process for the current frame. The parameter $\hat{\Xi}$ scales the complexity of the Intra encoding process for the current frame from the full encoding complexity to the MEP. Figure 8.1 presents the control system overview.

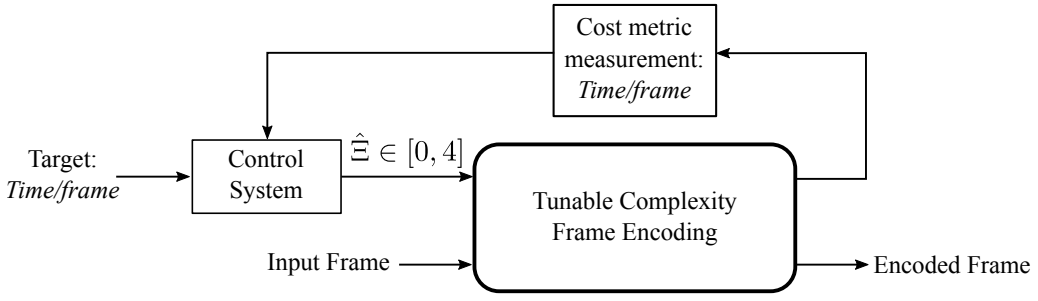


Figure 8.1 – *Overview of the control system to manage the complexity of the current encoding frame according to the time target in seconds.*

The control system takes as input the target encoding time per frame in seconds and the measured time of the previous frame. The control system then computes and sets the corresponding value of parameter $\hat{\Xi}$.

8.2.1 Encoding Process System Modeling

Based on the nature of the *Tunable Complexity Frame Encoding* system and the results observed in Section 7.4.3, the system is modeled as unit delay with a gain K_h, as illustrated in Figure 8.2. Between the output time of the encoding process and the system output y, a disturbance w is added, which corresponds to the profiling fuzziness, hardware and software disturbances or encoding time variations due to the input video sequence content.

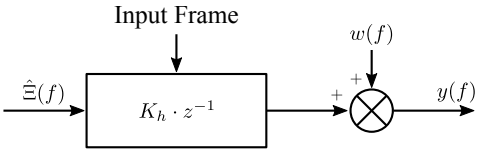


Figure 8.2 – Tunable Complexity Frame Encoding *system modeled as a unit delay with a gain* K_h.

As the system has to manage the absolute encoding time per frame of the encoding process, the gain K_h depends on every factor that have an impact on the frame encoding time, such as the input video sequence resolution or the QP value (as shown in Section 4.3). Moreover, Figure 7.18 shows that the energy consumed according to the command $\hat{\Xi}$ is not linear, the value of the command $\hat{\Xi}$ also affects the gain K_h.

To control the system, the gain K_h is modeled according to three factors:

- the number of pixels by frame,
- the QP value,
- the command $\hat{\Xi}$ value.

The gain K_h corresponds to the gap between the encoding time of a frame when the command $\hat{\Xi}$ goes from the minimum to the maximum value, i.e. from $\hat{\Xi} = 0$ to $\hat{\Xi} = 4$. Because the gain K_h is not stable across the command $\hat{\Xi}$, K_h is modeled for each interval $[0, 0.5[$, $[0.5, 1[$, $[1, 1.5[$, $[1.5, 2[$, $[2, 2.5[$, $[2.5, 3[$, $[3, 3.5[$ and $[3.5, 4[$. The K_h values for each interval of command $\hat{\Xi}$ used for the modeling are computed from all the frames of the 18 sequences described in Table 4.1 for 4 QP values 22, 27, 32 and 37. The gain K_h is averaged across the frames of the sequences having the same resolution for each QP value and interval of $\hat{\Xi}$. Table 8.1 shows as example the gain K_h according to the resolution and $\hat{\Xi}$ intervals for $QP = 32$. The same variation of results is observed for other QP values.

The modeling of the gain K_h can be directly conducted using a LUT. Nevertheless, this approach leads to too high gaps of the gain K_h when $\hat{\Xi}$ is up to 2.5 which has a non-negligible impact on the control.

However, a polynomial approximation of the gain K_h according to the command $\hat{\Xi}$ and the number of pixels by frame Nb_{px} for each QP value provides good results. Figure 8.3 shows the gain K_h versus the command $\hat{\Xi}$ and the number of pixels by frame Nb_{px} for $QP = 32$ (blue dots) and the polynomial approximation of degree (4,4) of K_h (the surface). Similar results are obtained for other QP values.

Table 8.1 – *Average of gain K_h by resolution by frame and $\hat{\Xi}$ intervals for $QP = 32$.*

$\hat{\Xi} \in$	$[0, 0.5[$	$[0.5, 1[$	$[1, 1.5[$	$[1.5, 2[$	$[2, 2.5[$	$[2.5, 3[$	$[3, 3.5[$	$[3.5, 4[$
Class A	0.598	0.871	0.645	0.845	0.644	0.736	1.324	2.225
Class B	0.259	0.494	0.255	0.431	0.261	0.491	0.425	1.110
Class E	0.078	0.193	0.044	0.163	0.057	0.153	0.090	0.419
Class C	0.075	0.125	0.077	0.100	0.072	0.091	0.175	0.284
Class D	0.024	0.035	0.022	0.031	0.019	0.025	0.051	0.064

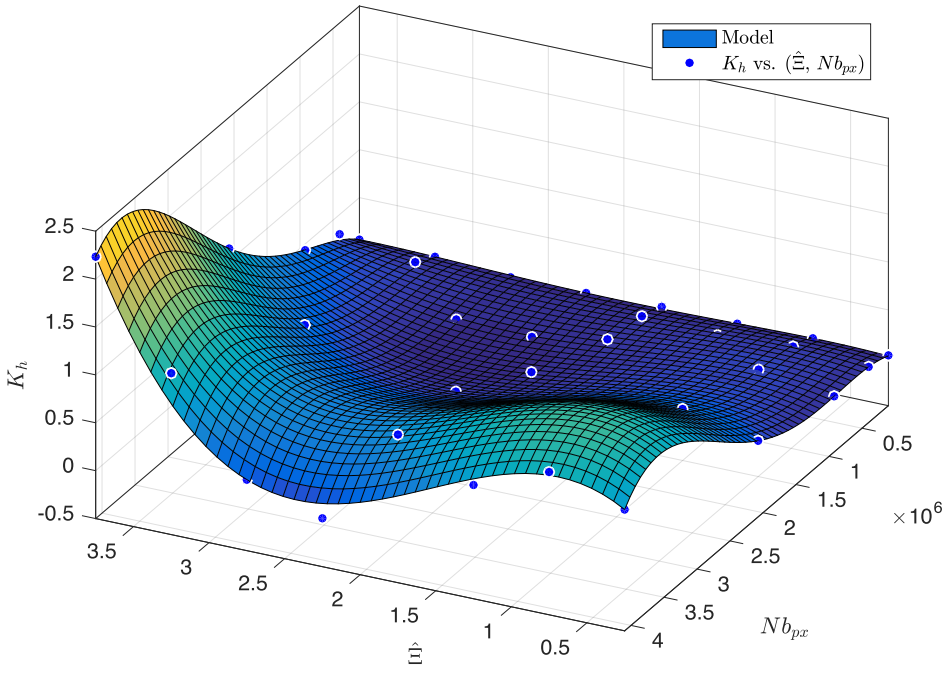


Figure 8.3 – *Gains K_h versus the command $\hat{\Xi}$ and the number of pixels by frame Nb_{px} for $QP = 32$ (blue dots) and polynomial approximation of degree (4,4) of K_h (the surface).*

The gain K_h is modeled per QP value across the command $\hat{\Xi}$ and the number of pixels by frame Nb_{px} by polynomial approximation of degree (4,4). The *Rsq* defined in Section 5.2.2 by Equation 5.2 is used as a metric to quantify the accuracy of a predicting model. As a reminder, it falls between 0 and 1 and the more the *Rsq* is close to 1, the more the predicted data fits the actual data. Table 8.2 summarizes the *Rsq* for $QP \in \{22, 27, 32, 37\}$. The *Rsq* value never goes down below 88%, which is sufficient in this context for the K_h modeling.

Table 8.2 – *Rsq average of K_h modeling according to the QP value.*

	QP22	QP27	QP32	QP37
Rsq	0.9076	0.8899	0.9435	0.9328

8.2.2 PI-based feedback control - Parameters Tuning

For the control system, a solution based on a Proportional–Integral (PI) controller [Ast95] is implemented. PI controller is a control feedback loop mechanism that is widely known in the industry due to its applicability in many systems. The control loop does not include a derivative term to limit the impact of perturbations $w(f)$ of the profiling fuzziness, hardware and software disturbances, or time variation due to the input video sequence content which induce much noise and sudden signal change on the output signal $y(f)$. The PI controller involves two separate constant parameters: the proportional K_p and integral K_i terms. The proportional value K_p depends on the current error ϵ, whereas K_i depends on the past values of ϵ. Figure 8.4 shows the block diagram of the PI controller in a feedback loop of the *Tunable Complexity Frame Encoding* system using trapezoid rule to approximate the integral.

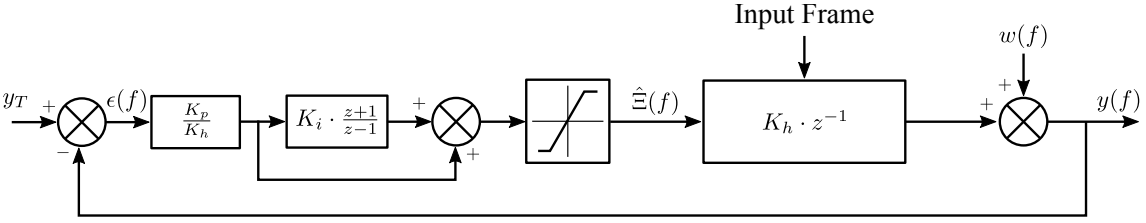


Figure 8.4 – *PI controller in a feedback loop of the* Tunable Complexity Frame Encoding *system.*

y_T is the target time value by frame in seconds, $y(f)$ is the time in seconds obtained for the current encoded frame f, $\epsilon(f) = y_T - y(f)$ is the error in seconds between the target y_T and the current time $y(f)$, K_p and K_i are respectively the proportional and integral terms and $\hat{\Xi}$ is the command computed by the PI control loop system.

Following the block diagram of the PI controller described by Figure 8.4, the transfer function of the control system between command $\hat{\Xi}(z)$ and error $\epsilon(z)$ is described by Equation 8.1.

$$\frac{\hat{\Xi}(z)}{\epsilon(z)} = \frac{K_p}{K_h}\left(1 + K_i\frac{z+1}{z-1}\right) \tag{8.1}$$

By closing the loop, transfer function of the control system can be described by Equation 8.2.

$$\frac{\hat{\Xi}(z)}{\epsilon(z)} = \frac{K_p}{K_h}\left(\frac{(K_i+1) + (K_i-1)z^{-1}}{1 - z^{-1}}\right) \tag{8.2}$$

An anti-windup compensator is added to avoid integrator windup due to the saturation of the command $\hat{\Xi}$, which is limited in the interval $\{0, 4\}$. From Equation 8.2, the command $\hat{\Xi}(f)$ for a frame f computed by the PI controller is described by the recurrence relation given by Equation 8.3.

$$\hat{\Xi}(f) = \begin{cases} 0 & , \text{if } \hat{\Xi}(f) < 0 \\ \hat{\Xi}(f-1) + \frac{K_p}{K_h}\left[(K_i+1)\,\epsilon(f) + (K_i-1)\,\epsilon(f-1)\right] & , \text{if } 0 \leq \hat{\Xi}(f) \leq 4 \\ 4 & , \text{if } \hat{\Xi}(f) > 4 \end{cases} \tag{8.3}$$

To find the operating point of K_p and K_i, two metrics are used to evaluate their impact on the control system: the normalized Absolute Time Error (ATE) and the normalized Time Standard Deviation (Tstd).

- The normalized ATE measures the time control accuracy of the encoding process and is defined by the Equation 8.4:

$$ATE = \frac{1}{y_T} \cdot \left(\sum_{f=1}^{Nb_f} \frac{|y(f) - y_T|}{Nb_f} \right) \tag{8.4}$$

where Nb_f is the number of frames for the encoding video sequence, $y(f)$ is the encoding time of the frame f in seconds and y_T is the target frame duration in seconds.

- The normalized Tstd measures the time control stability of the encoding process and is defined by Equation 8.5:

$$Tstd = \frac{1}{\bar{y}} \cdot \left(\sqrt{\frac{\sum_{f=1}^{Nb_f} (y(f) - \bar{y})^2}{Nb_f - 1}} \right) \tag{8.5}$$

where $\bar{y}$ is the average encoding time per frame in seconds defined by $\bar{y} = \frac{1}{Nb_f} \sum_{f=1}^{Nb_f} y(f)$.

Five video sequences composed of one video sequence of each class are used to find the K_p and K_i operating points: *Traffic* (class A), *BasketballDrive* (class B), *BQMall* (class C), *BasketballPass* (class D) and *Johnny* (class E). All video sequences are encoded using 4 QP values: $\{22, 27, 32, 37\}$, for 3 time targets y_T: 50%, 70% and 90%. The target times y_T are computed for every sequence according the QP value proportionally to the time of the full encoding process. According to a PI tuning by successive approach, the best operating point of the proportional term K_p and integral term K_i are found in three steps using the average normalized ATE and Tstd:

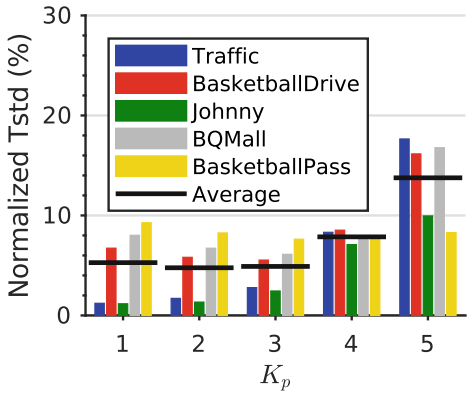
1. The integral term K_i is firstly set to 0 and a first set of K_p values is tested. The best value of K_p, i.e. the one that offer the best stability, is selected using the Tstd.
2. The proportional term K_p is set to the previous selected best value and the integral term K_i value that offers the best accuracy and stability, i.e. that minimizes both ATE and Tstd, is selected.
3. Finally, The integral term K_i is set to the previous selected value and the proportional term K_p is refined around the value selected in the step 1 using both ATE and Tstd.

8.2.2.1 Proportional Term K_p Exploration

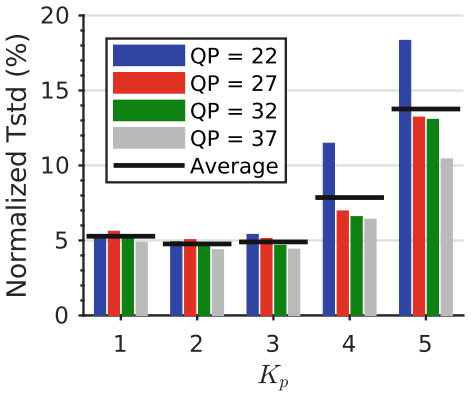
To find the operating point of the proportional term K_p, the integral term K_i is firstly set to 0. Without the integral term K_i, the control system cannot reach the target y_T due to the static error and thus the ATE metric cannot be used to fix the K_p value. The proportional term K_p is increased until the system becomes unstable and the best value of K_p is the one that offers the best stability, i.e. the one that minimizes the Tstd metric. A set of 5 K_p values composed by $\{1, 2, 3, 4, 5\}$ is tested.

Figure 8.5 presents the average normalized Tstd results of the control system for the set of K_p values. Figure 8.5a shows results for each video sequence when the normalized Tstd is averaged across the QP and time target values. Figure 8.5b presents the results

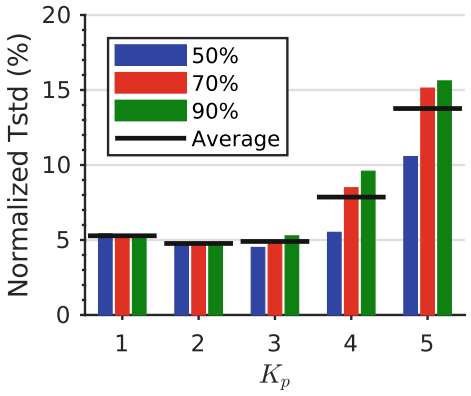
for each QP value when the normalized Tstd is averaged across the video sequences and time target values. Finally, Figure 8.5c shows the results for each time target when the normalized Tstd is averaged across the video sequences and QP values. On all figures, the global averages of the normalized Tstd are plotted in black according to the K_p value.



(a) *Averaged normalized Tstd across the QP and time target values according to the video sequences and K_p values.*



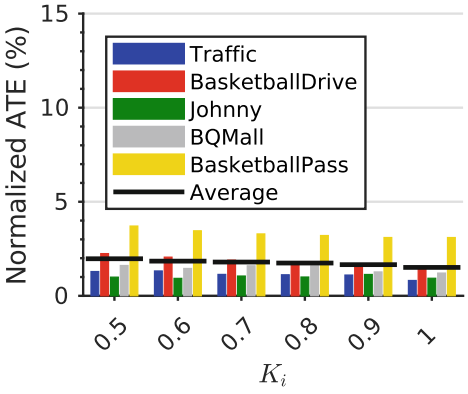
(b) *Averaged normalized Tstd across the video sequences and time target values according to the QP values and K_p values.*



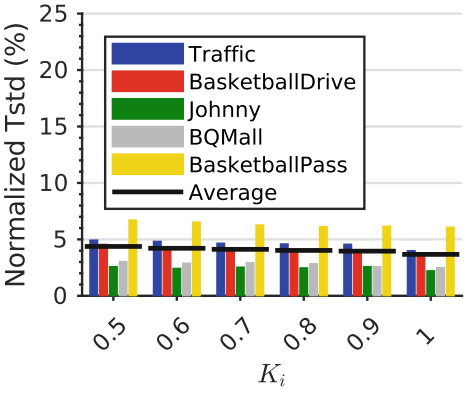
(c) *Averaged normalized Tstd across the video sequences and QP values according to the time target values and K_p values.*

Figure 8.5 – *Averaged normalized Tstd according to the K_p values. The global average of the normalized Tstd is plotted in black according to the K_p value.*

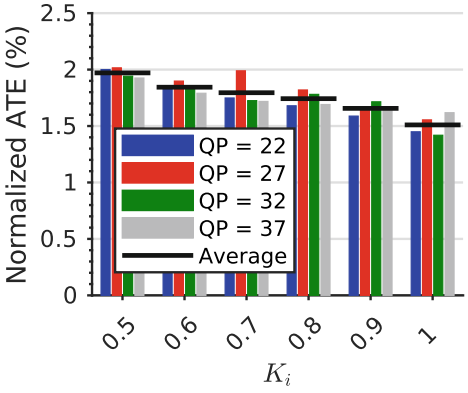
The results show that the best set of K_p values that minimizes averages of normalized Tstd is $K_p \in \{1, 2, 3\}$ with an average of normalized Tstd round 5%. For the three values of K_p, the difference of averages of normalized Tstd is less than 0.5% and therefore not significant enough to be taken into account. Indeed, results of Figure 7.16 and 7.17 in Section 7.4.3 have shown (when $\hat{\Xi} > 3$) that for the same encoding configuration, the results of complexity can vary by 1% between several experiences. As the proportional term K_p acts on the integral term K_i, the K_p value is firstly set to 1 to limit its impact on the integral part. The K_p value will be refined in Section 8.2.3 after setting the integral term K_i.



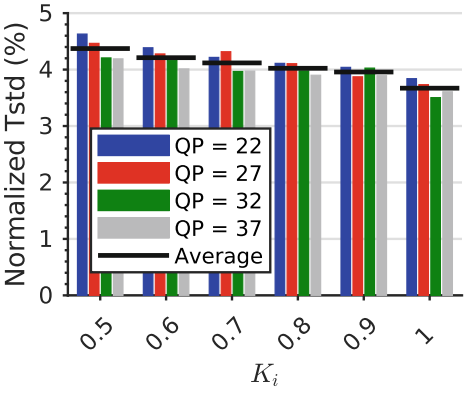
(a) *Averaged normalized ATE across the QP and time target values according to the video sequences and K_i values.*



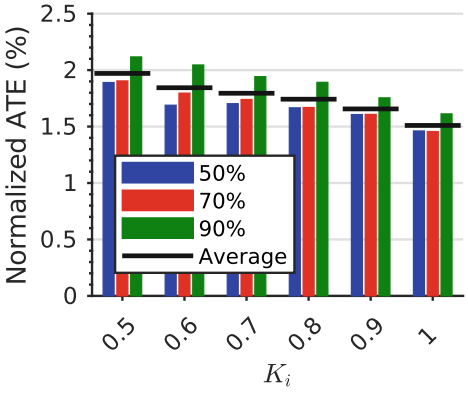
(b) *Averaged normalized Tstd across the QP and time target values according to the video sequences and K_i values.*



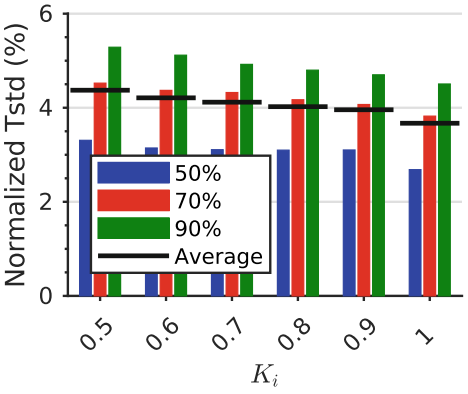
(c) *Averaged normalized ATE across the video sequences and time target values according to the QP values and K_i values.*



(d) *Averaged normalized Tstd across the video sequences and time target values according to the QP values and K_i values.*



(e) *Averaged normalized ATE across the video sequences and QP values according to the time target values and K_i values.*



(f) *Averaged normalized Tstd across the video sequences and QP values according to the time target values and K_i values.*

Figure 8.6 – *Averaged normalized ATE and Tstd according to the K_i values with $K_p = 1$. The global average of the normalized ATE and Tstd are plotted in black according to the K_i value.*

8.2.2.2 Integral Term K_i Exploration

The proportional term K_p is now set to 1, and the integral term K_i impact is explored. A set of 6 K_i values composed by $\{0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ is tested. Figure 8.6 presents the

average normalized ATE and Tstd results of the control system for the previous set of K_i values. Figures 8.6a and 8.6b show results for each video sequence when the normalized ATE and Tstd are averaged across the QP and time target values. Figures 8.6c and 8.6d show the results for each QP value when the normalized ATE and Tstd are averaged across the video sequences and time target values. Finally, Figures 8.6e and 8.6f show the results for each time target when the normalized ATE and Tstd are averaged across the video sequences and QP values. On all figures, the global average of the normalized ATE and Tstd are plotted in black according to the K_i value.

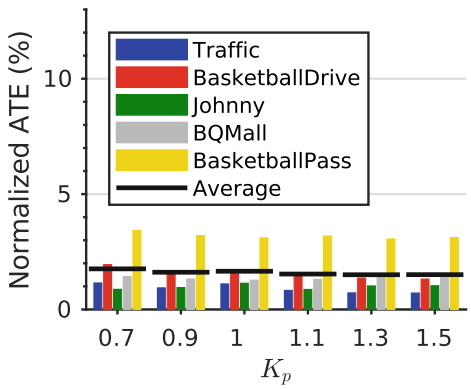
The results show that the more K_i increases, the more averages of the normalized ATE and Tstd decrease. We did not increase the integral term K_i more than 1 after observing that the system became unstable in some cases. The K_i value is set to 0.9 obtaining averages of normalized ATE and Tstd of 1.66% and 3.96%, respectively. The value $K_i = 1$ is not chosen to avoid the removal of the error at the previous frame $\epsilon(f-1)$ in the PI command (see Equation 8.3).

8.2.3 Proportional Term K_p Refinement

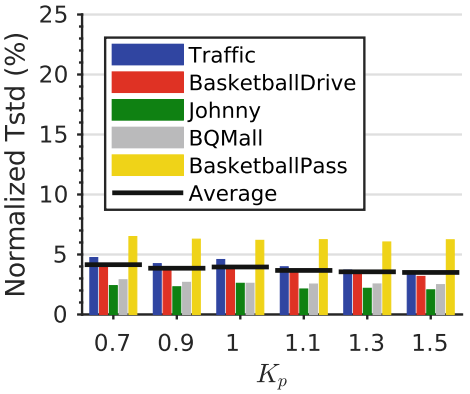
The last step to set up the PI feedback loop control system is to refine the proportional term K_p around 1 with K_i fixed to 0.9. A set of 6 K_p values composed by $\{0.7, 0.9, 1, 1.1, 1.3, 1.5\}$ are tested. Figure 8.7 presents the average normalized ATE and Tstd results of the control system for the previous set of K_p values. Figures 8.7a and 8.7b show results for each video sequence when the normalized ATE and Tstd are averaged across the QP and time target values. Figures 8.7c and 8.7d show the results for each QP value when the normalized ATE and Tstd are averaged across the video sequences and time target values. Finally, Figures 8.7e and 8.7f show the results for each time target when the normalized ATE and Tstd are averaged across the video sequences and QP values. On all figures, the global average of the normalized ATE and Tstd are plotted in black according to the K_p value.

First of all, the results show that the averaged normalized ATE and Tstd decrease slightly as the proportional term K_p increases. However, the difference of results between K_p values is not sufficiently high to be greater than the imprecision of the profiling (as explained in the previous section). In other words, the control accuracy and stability are stable regardless of the value of K_p. Considering the results, the proportional term K_p is finally fixed to 1.3.

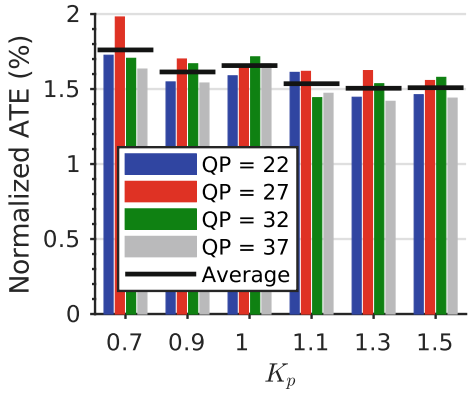
It is interesting to notice that the control accuracy is stable across the QP and time target values y_T but not across the video sequences. The sequence with the smallest resolution, *BasketBallPass*, has an average normalized ATE and Tstd higher than the other sequences. This difference is due to the number of CTU per frame that limit the control quantization step (see Figure 7.11), i.e. degree of liberty (finesse), of the CDC complexity allocator presented in Section 7.3.



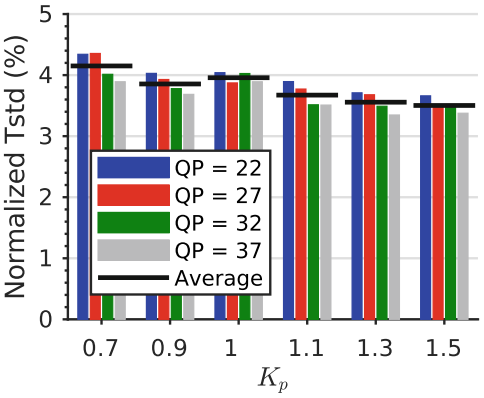
(a) *Averaged normalized ATE across the QP and time target values according to the video sequences and K_p values with $K_i = 0.9$.*



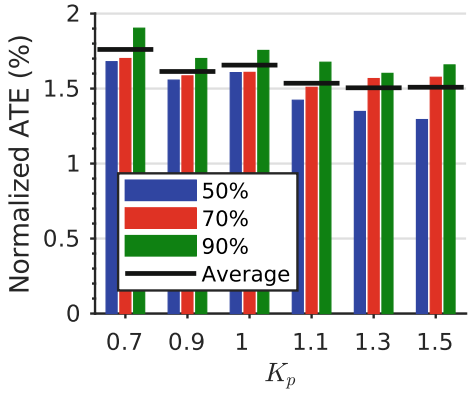
(b) *Averaged normalized Tstd across the QP and time target values according to the video sequences and K_p values with $K_i = 0.9$.*



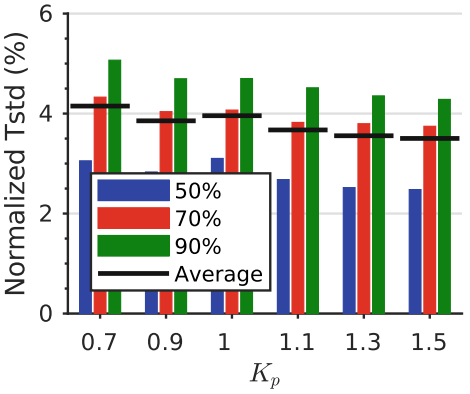
(c) *Averaged normalized ATE across the video sequences and time target values according to the QP values and K_p values with $K_i = 0.9$.*



(d) *Averaged normalized Tstd across the video sequences and time target values according to the QP values and K_p values with $K_i = 0.9$.*



(e) *Averaged normalized ATE across the video sequences and QP values according to the time target values and K_p values with $K_i = 0.9$.*



(f) *Averaged normalized Tstd across the video sequences and QP values according to the time target values and K_p values with $K_i = 0.9$.*

Figure 8.7 – *Averaged normalized ATE and Tstd according to the K_p values with $K_i = 0.9$. The global average of the normalized ATE and Tstd are plotted in black according to the K_p value.*

8.3 Encoding Time Control System of HEVC Intra Encoder

The proposed encoding time control system of HEVC Intra encoder is hierarchically depicted in Figure 8.8 gathering the main contributions of this document. The system takes as input the raw video sequence and an encoding target time per frame in seconds and adjusts the HEVC intra encoding process to respect the time constraint per frame.

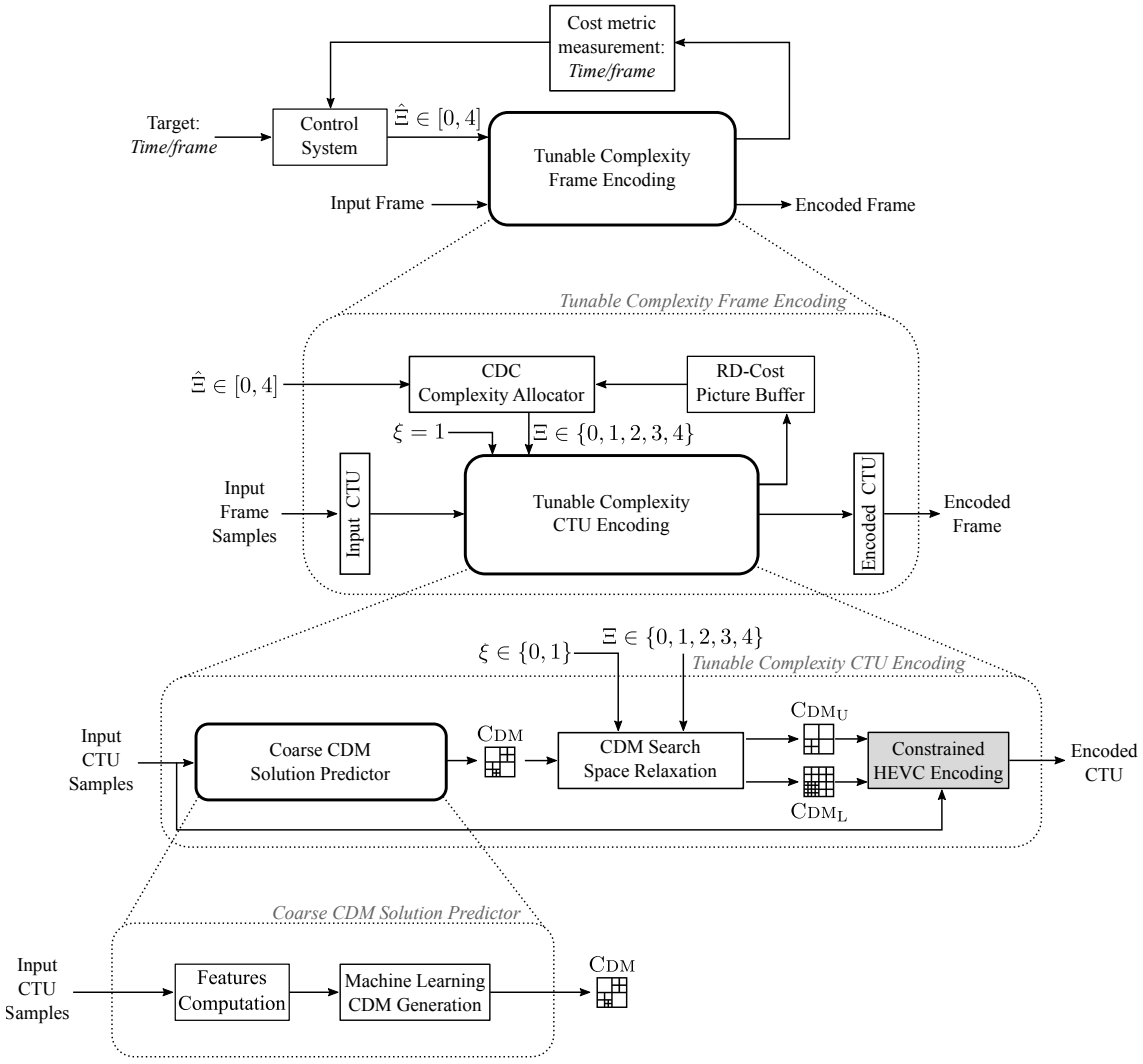


Figure 8.8 – *Overview of the encoding time control system of the HEVC Intra encoder.*

At the top-level, the PI based control system, detailed in Section 8.2, takes as input the target time per frame in seconds and the measured time of the previous frame to set the right value of the parameter $\hat{\Xi}$ for the *Tunable Complexity Frame Encoding* scheme. The parameter $\hat{\Xi} \in [0,4]$ corresponds to the number of depth levels tested during the RDO process for the current frame. The parameter $\hat{\Xi}$ scales the complexity of the Intra encoding process for the current frame from the full encoding complexity to the MEP.

The *Tunable Complexity Frame Encoding* scheme, detailed in Section 7.3.3, receives the current frame and the parameter $\hat{\Xi}$ and constrains the encoding process to test only the real number of depth levels $\hat{\Xi} \in [0,4]$ for the current frame. The real number of depth levels $\hat{\Xi}$ is sent to the CDC complexity allocator, detailed in Section 7.3, to set the appropriate $\Xi \in \{0,1,2,3,4\}$ parameter value to the *Tunable Complexity CTU Encoding* for each CTU.

The CDC complexity allocator splits the real number $\hat{\Xi}$ into two parts: the integer part and the fractional part. The integer part of $\hat{\Xi}$ sets the minimum number of depth levels $\lfloor\hat{\Xi}\rfloor$ tested for all CTU of the input frame. The fractional part of $\hat{\Xi}$ sets the number of CTU that tests one more depth level $\lceil\hat{\Xi}\rceil$. The CDC complexity allocator uses the RD-cost picture buffer of the previous frame to select which CTU is constrained to test $\lfloor\hat{\Xi}\rfloor$ or $\lceil\hat{\Xi}\rceil$ depth levels according to the protocol defined in Section 7.3.2.1. The CTU obtaining the highest RD-costs in the previous image are forced to test $\lceil\hat{\Xi}\rceil$ depth levels while the rest of the CTU are forced to test $\lfloor\hat{\Xi}\rfloor$ depth levels.

The *Tunable Complexity CTU Encoding* receives the integer number of depth levels Ξ and the current CTU samples to be encoded. The *Coarse* CDM *Solution Predictor*, detailed in Section 6.4, is used to predict in one-shot the quad-tree partitioning of the current CTU across the associated CDM. The CDM *Search Space Relaxation* algorithm, detailed in Section 7.2, receives the generated CDM and the number of depth levels Ξ to compute an interval of two CDM formed by the Upper CDM_U and the Lower CDM_L. Finally, the HEVC encoder is forced to only apply the RDO process between the interval defined by CDM_L and CDM_U.

The next sections present the performance of the proposed encoding time control system of HEVC Intra encoder summarized by Figure 8.8. They give the experimental results obtained for the proposed encoding time control system on a real time HEVC encoder. The experimental set-up has been detailed in Section 4.3.1. The evaluation of the performance is realized on the 18 video sequences presented in Table 4.1 for targets ranging from 90% to 40% (the MEP). In the experiments presented in next sections, the target times in seconds are based on the reference full encoding time of the video sequences according to the QP value. For each sequence and QP value, the full encoding time is divided by the number of frames of the video sequences and the targets are computed by weighting the obtained averaged time per frame by the target coefficient. All results of the following sections include the time overhead due to the entire encoding time control system scheme.

8.3.1 Control System Response to Set-Point

Table 8.3 and Figure 8.9 present the average real Normalized Encoding Time (NET) according to the target NET. The results are the average of NET across the 18 different sequences belonging to 5 classes (A, B, C, D and E) presented in Table 4.1 for four QP values: 22, 27, 32, 37. The NET error is the difference between the real and the target NET. The NET std is the standard deviation of the real NET across all the sequences and QP values. The NET std is represented in Figure 8.9 by the blue area.

Table 8.3 – *Average real NET, error and Std across the video sequences and QP values according to the target NET.*

Target NET (%)	Real NET (%)	NET Error (%)	NET Std (%)
90	90.87	0.87	0.77
80	81.24	1.24	0.51
70	71.22	1.22	0.48
60	61.20	1.20	0.55
50	51.40	1.40	0.56
40	43.28	3.28	2.56

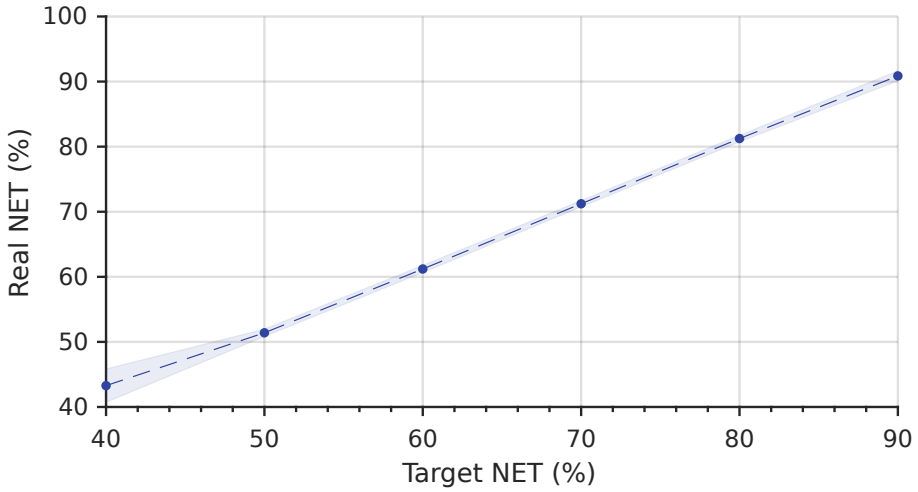


Figure 8.9 – *Average real NET across the video sequences and QP values according to the target NET. The blue area shows the standard deviation of the real NET.*

The results show that the proposed encoding time control system of HEVC Intra encoder is able to reach the target with precision. The NET error does not exceed 1.5% in average for targets ranging from 90% to 50%. Moreover, the NET std show that the results are stable regardless of the video sequence content, resolution and the QP value. The NET error rises to 3.28% for the target NET of 40% because the *Tunable Complexity Frame Encoding* system reaches its limit. Reducing the average NET by 60% is not always possible with our proposed system for all frames of the video sequences, as shown in the following figures.

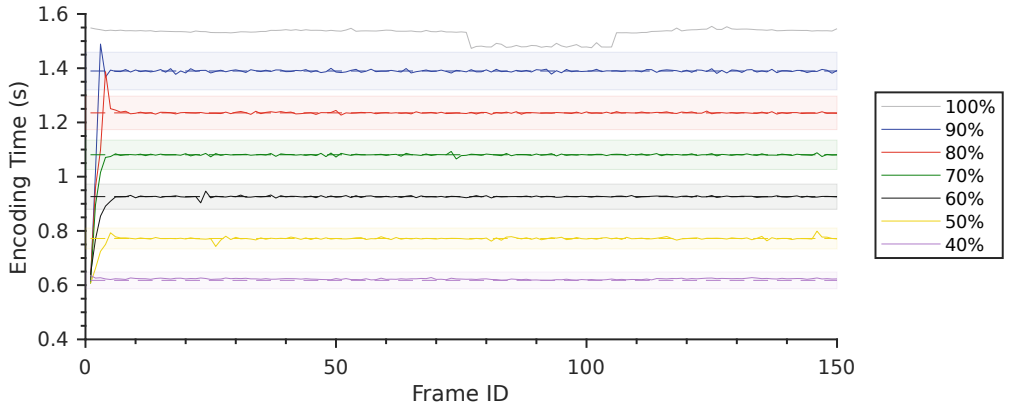
Figures 8.10, 8.11, 8.13, 8.14 and 8.12 show the NET and the associated command $\hat{\Xi}$ for targets ranging from 90% to 40% of five video sequences: *Traffic* (class A), *Kimono* (class B), *RaceHorses* (class C), *BasketballPass* (class D) and *KristenAndSara* (class E). The QP value is fixed to 32. The color area indicates the interval of ±5% for each time target. The same type of results are observed for all QP values and video sequences.

For the two highest resolution, 2560×1600 (Figure 8.10) and 1920×1080 (Figure 8.11), the NET control is stable regardless of the NET target. The lower the resolution, the lower the stability. This is due to the number of CTU available per frame which decreases the quantization step of the CDC complexity allocator, as shown in Figure 7.11. For example, for the video sequence *BasketballPass* (416×240), only 18 full CTU can be used by our control system to adapt the encoding complexity. A low number of CTU by frame does not allow to reach any value of time reduction. It is why the NET oscillates around the time target in Figure 8.14a.

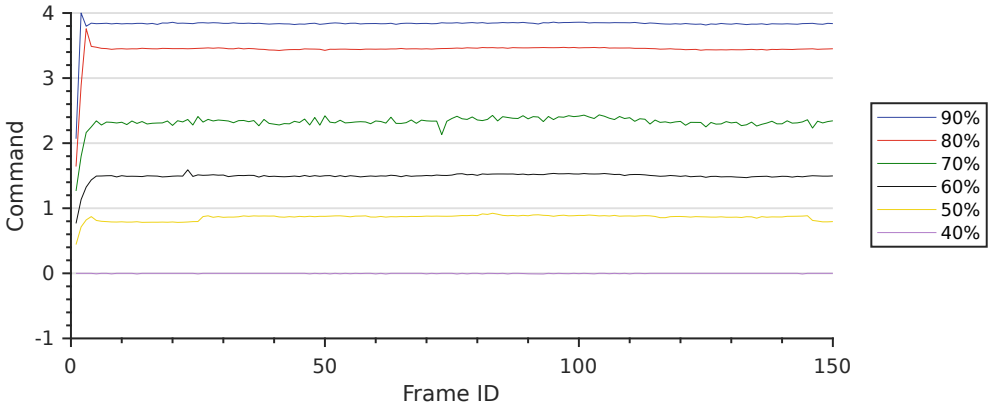
As explained before, a NET target of 40% is not always reachable for all frames of the video sequence. Figure 8.13b shows that to reach the time target of 40%, the command $\hat{\Xi}$ saturates to the minimum value 0 from the first frame to the 140^{th}.

Figure 8.13 and 8.14 show the robustness of the control system. The full NET (100%) is spread between frames but the control system is able to properly adapt the command $\hat{\Xi}$ to keep the NET in the interval of ±5% of the NET target.

The specificity of the *Kimono* video sequence is to include a cut scene in the 140^{th} frame (illustrated by Figures B.3 and B.4 in Appendix B). The cut scene leads to a sudden change in encoding complexity. Figure 8.11a shows that our control system just needs one or two frames to reduce the NET in the interval of ±5% of the NET target. The proposed



(a) *Encoding time in seconds for each frame of the Traffic sequence according to the time target. The color area indicates the interval of ±5% of the time target.*

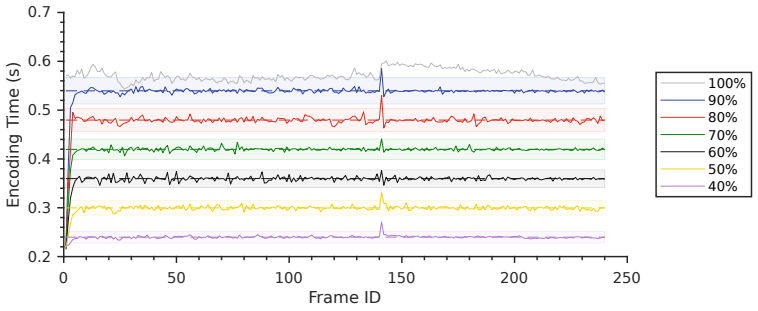


(b) *Command $\hat{\Xi}$ for each frame of the Traffic sequence according to the target.*

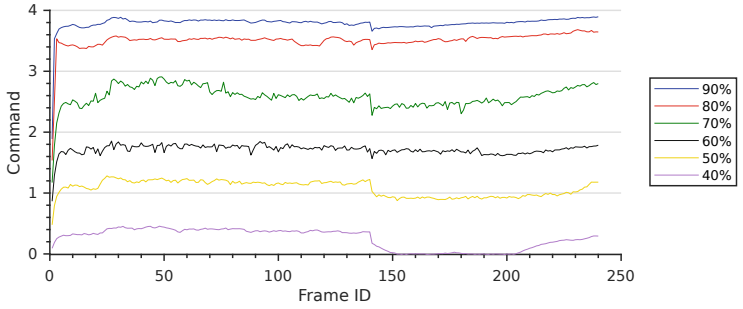
Figure 8.10 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the Traffic sequence (Class A – 2560 × 1600) according to the target from 100% to 40% (QP = 32).*

encoding time control system is able to quickly adapt to sudden changes in video content properties.

Finally, the robustness of the control system is tested by dynamically changing the NET target during the encoding process. Figures 8.15 and 8.16 show the NET according to the target NET and the associated command $\hat{\Xi}$ for dynamic target from 100% to 40% of two video sequences: *BasketballDrill* (class C) and *Cactus* (class B). The first 20 frames are encoded without constraint, i.e. with $\hat{\Xi} = 4$, and the average of encoding time of these frames is then used as the reference time (100 %) to compute the target NET for the rest of the video sequence. The number of frames between two target changes is fixed to 20. Every 20 frames, the target NET increases or decreases by 10%. Results show that our control system adapts the command $\hat{\Xi}$ (in green in Figures 8.15b and 8.16b) to target changes with precision and rapidity regardless of video content and resolution. The NET (in blue in Figures 8.15a and 8.16a) follows the dynamic NET target (in red in Figures 8.15a and 8.16a) throughout the encoding process of the video sequence.

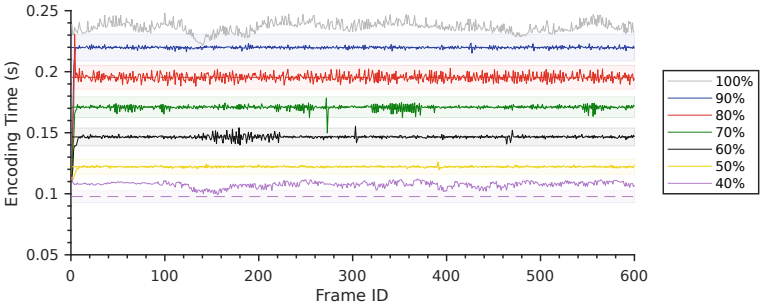


(a) *Encoding time in seconds for each frame of the Kimono sequence according to the time target. The color area indicates the interval of ±5% of the time target.*

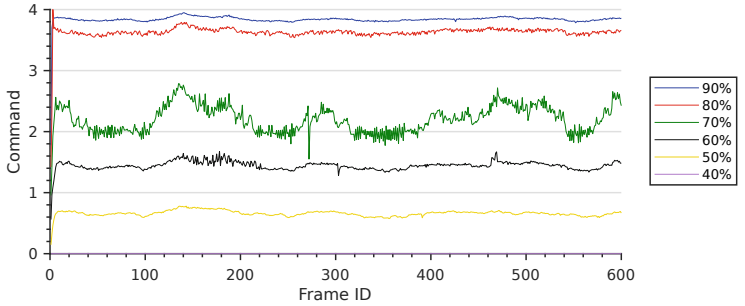


(b) *Command $\hat{\Xi}$ for each frame of the Kimono sequence according to the target.*

Figure 8.11 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the Kimono sequence (Class B – 1920×1080) according to the target from 100% to 40% (QP = 32).*

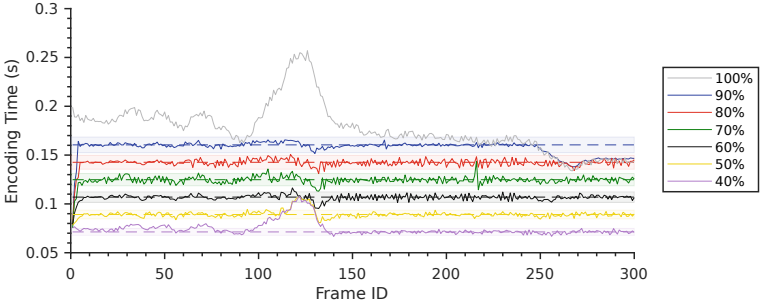


(a) *Encoding time in seconds for each frame of the KristenAndSara sequence according to the time target. The color area indicates the interval of ±5% of the time target.*

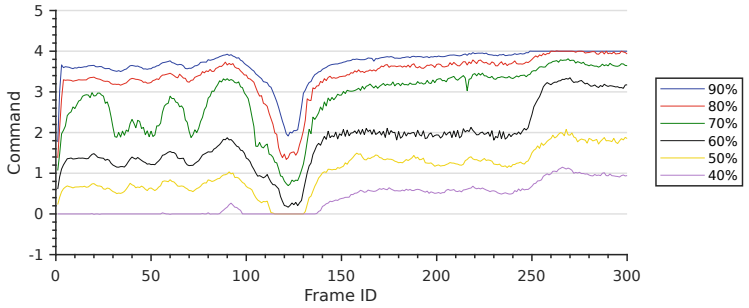


(b) *Command $\hat{\Xi}$ for each frame of the KristenAndSara sequence according to the target.*

Figure 8.12 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the KristenAndSara sequence (Class E – 1280×720) according to the target from 100% to 40%. (QP = 32)*

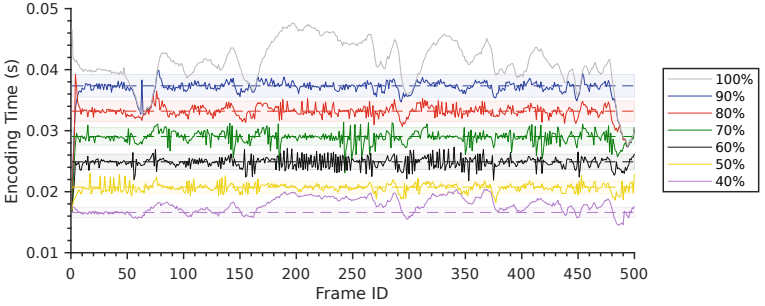


(a) *Encoding time in seconds for each frame of the RaceHorses sequence according to the time target. The color area indicates the interval of ±5% of the time target.*

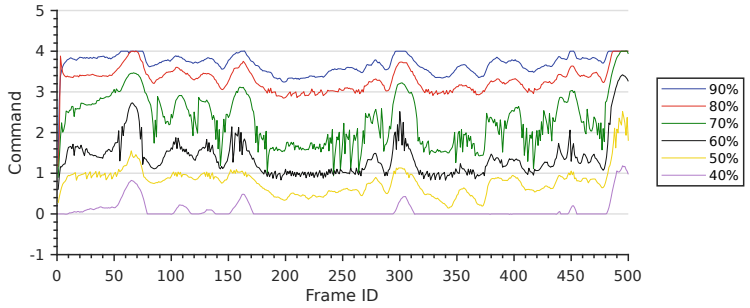


(b) *Command $\hat{\Xi}$ for each frame of the RaceHorses sequence according to the target.*

Figure 8.13 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the RaceHorses sequence (Class C – 832 × 480) according to the target from 100% to 40% (QP = 32).*

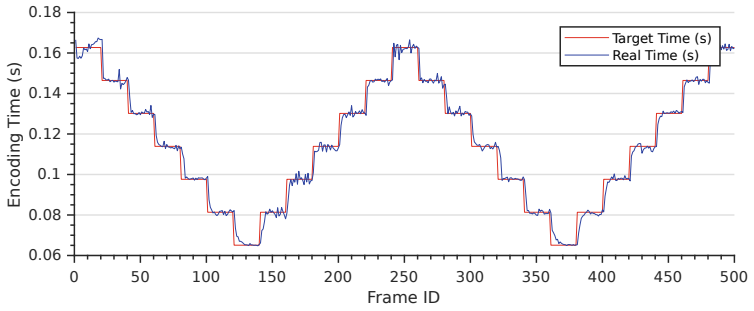


(a) *Encoding time in seconds for each frame of the BasketballPass sequence according to the time target. The color area indicates the interval of ±5% of the time target.*

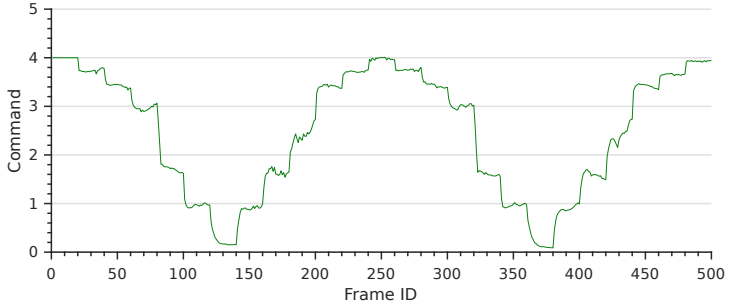


(b) *Command $\hat{\Xi}$ for each frame of the BasketballPass sequence according to the target.*

Figure 8.14 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the BasketballPass sequence (Class D – 416 × 240) according to the target from 100% to 40% (QP = 32).*

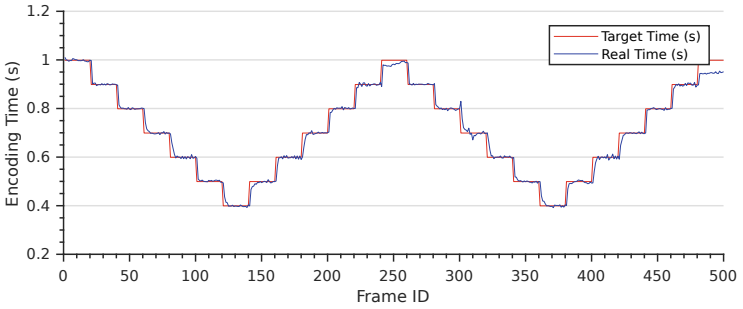


(a) *Encoding time in seconds for each frame of the BasketballDrill sequence according to the dynamic time target.*

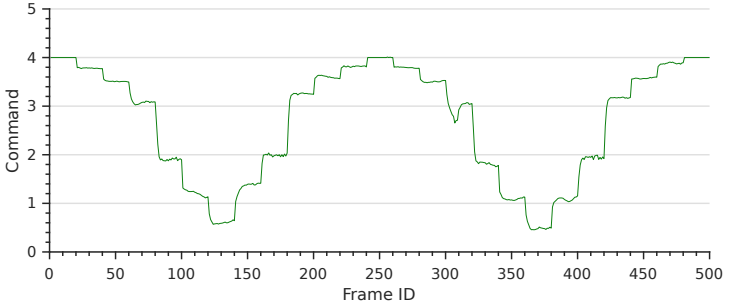


(b) *Command $\hat{\Xi}$ for each frame of the BasketballDrill sequence according to the dynamic target.*

Figure 8.15 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the BasketballDrill sequence (Class C – 832×480) according to the dynamic target from 100% to 40% changing every 20 frames ($QP = 32$).*



(a) *Encoding time in seconds for each frame of the Cactus sequence according to the dynamic time target.*



(b) *Command $\hat{\Xi}$ for each frame of the Cactus sequence according to the dynamic target.*

Figure 8.16 – *Encoding time in seconds and associated command $\hat{\Xi}$ of the Cactus sequence (Class B – 1920×1080) according to the dynamic target from 100% to 40% changing every 20 frames ($QP = 27$).*

8.3.2 Comparison with the HEVC reference encoding

To show the control system impact on encoding degradation, the BD-BR of the *Tunable Complexity Frame Encoding* and the *Encoding Time Control System* according to the NET are plotted in Figure 8.17.

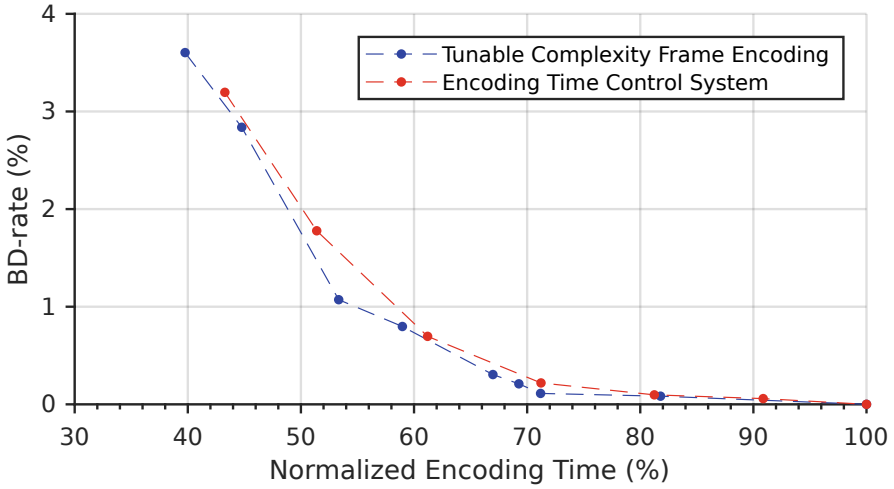


Figure 8.17 – *BD-BR of the* Tunable Complexity Frame Encoding *and the* Encoding Time Control System *according to the NET.*

In the case of the *Tunable Complexity Frame Encoding*, the command $\hat{\Xi}$ corresponding to the encoding constraint is fixed along the frames. In the case of the *Encoding Time Control System* the command $\hat{\Xi}$ and the encoding constraint are dynamically adjusted. The results show that the control system introduces a slight degradation in BD-BR due to the variation of the command $\hat{\Xi}$ along the frames.

Figures 8.18 and 8.19 show the results in terms of BD-BR and BD-PSNR versus NET reduction for targets ranging from 100% to 40%, these results are detailed in Table 8.4. These results are the average by class of the results obtained on the 18 video sequences presented by Table C.1 in Appendix C.

Results show that the encoding degradation in terms of BD-BR and BD-PSNR differ according to the class and resolution of the video sequence. Video sequences belonging to the class D with the smallest resolution obtained the best results in average than other class for all target NET, with an increase of 1.34% of BD-BR for a target NET of 40%. The worst degradation in average in terms of BD-BR and BD-PSNR are observed for video sequences of class B for targets ranging from 90% to 70% and for video sequences of class E for targets ranging from 60% to 40%. Table C.1 in Appendix C shows that the *Kimono* video sequence has a much higher degradation than the others. As it was explained in Section 6.7.1, this can be explained by the texture specificity of the Kimono video sequence which is composed by a traveling with trees and vegetation in the background. This video sequence has the highest Spatial Information (54.1) due to the details.

The bottom of Table 8.4 presents the average results in terms of real NET, error NET, BD-BR and BD-PSNR for the six targets NET tested in the experiments. The results show that the NET errors of our proposed encoding time control system of HEVC Intra encoder varied from 0.87% to 3.20% (with an average of 1.53%) according to the target NET for encoding degradation from 0.06% to 3.20% of BD-BR. The next section details a

Table 8.4 – *Real NET, NET Error, BD-BR and BD-PSNR averaged by class according to the target NET.*

Target NET (%)		Real NET(%)	NET Error (%)	BD-BR (%)	BD-PSNR (dB)
Class A 2560x1600	90	90.93	0.93	0.04	-0.00
	80	81.00	1.00	0.04	-0.00
	70	70.93	0.93	0.07	-0.00
	60	61.09	1.09	0.48	-0.03
	50	51.21	1.21	1.72	-0.10
	40	42.41	2.41	3.59	-0.20
Class B 1920x1080	90	90.79	0.79	0.14	-0.00
	80	81.27	1.27	0.25	-0.01
	70	71.33	1.33	0.37	-0.01
	60	61.36	1.36	0.93	-0.04
	50	51.41	1.41	2.00	-0.08
	40	42.44	2.44	4.10	-0.15
Class C 832x480	90	90.51	0.51	0.02	-0.00
	80	81.07	1.07	0.04	-0.00
	70	71.11	1.11	0.14	-0.01
	60	61.09	1.09	0.65	-0.03
	50	51.19	1.19	1.38	-0.08
	40	42.52	2.52	2.51	-0.14
Class D 416x240	90	90.63	0.63	0.00	-0.00
	80	81.11	1.11	0.01	-0.00
	70	70.95	0.95	0.08	-0.01
	60	60.85	0.85	0.28	-0.02
	50	51.35	1.35	0.76	-0.05
	40	44.24	4.24	1.34	-0.09
Class E 1280x720	90	91.75	1.75	0.07	-0.00
	80	81.78	1.78	0.07	-0.00
	70	71.76	1.76	0.34	-0.02
	60	61.61	1.61	1.08	-0.05
	50	51.84	1.84	3.33	-0.16
	40	44.97	4.97	4.81	-0.23
Average	90	90.87	0.87	0.06	-0.00
	80	81.24	1.24	0.10	-0.00
	70	71.22	1.22	0.22	-0.01
	60	61.20	1.20	0.70	-0.03
	50	51.40	1.40	1.78	-0.09
	40	43.28	3.28	3.20	-0.15

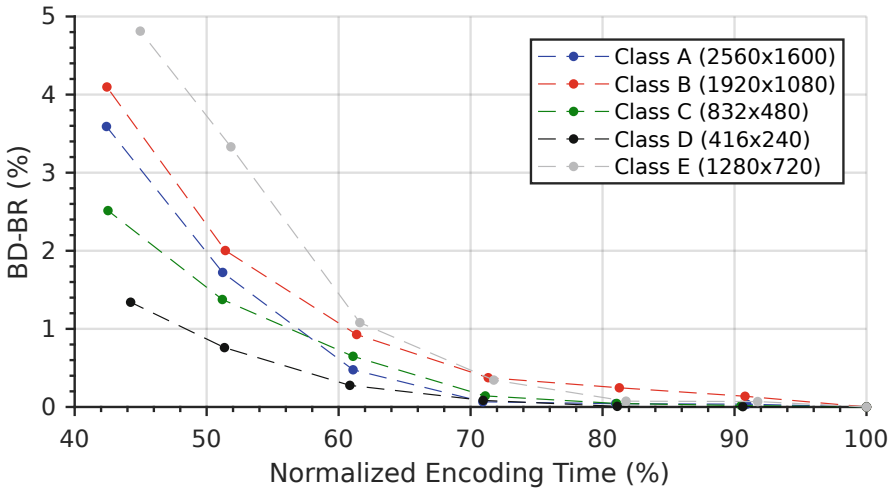


Figure 8.18 – *Average of BD-BR by class of the* Encoding Time Control System *according to the NET.*

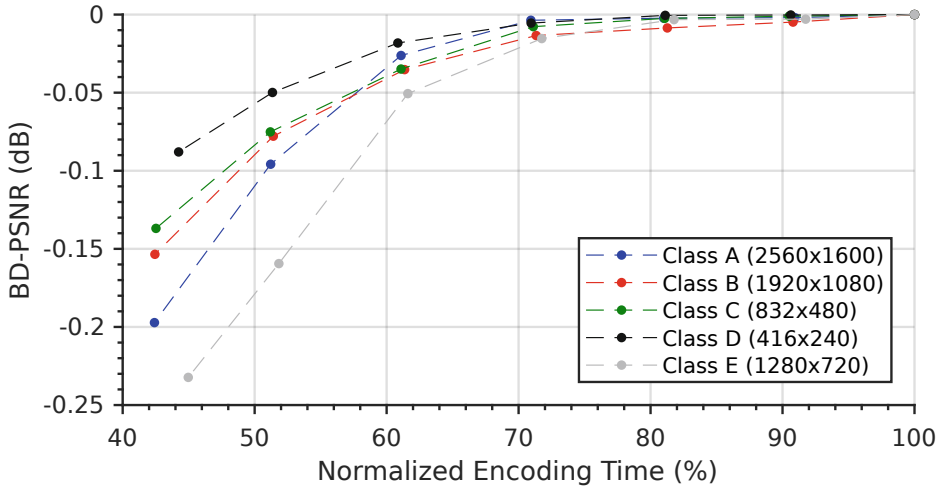


Figure 8.19 – *Average of BD-PSNR by class of the* Encoding Time Control System *according to the NET.*

comparison of our proposed encoding time control system of HEVC encoding and proposed methods found in the literature.

8.3.3 Comparison to State-of-the-Art

This section compares our *Encoding Time Control System* with the most competitive state-of-the-art solutions summarized in Section 3.3. It is important to remember that most of the approaches cannot be directly compared with each other w.r.t the achieved complexity reduction and encoding degradation (BD-BR or BD-PSNR) due to the encoding conditions. Indeed, in addition to missing information, different software encoder or encoder versions, encoding configurations, sets of test sequences, set-ups and experimental conditions (e.g. HW architecture, processor, caches, memory, etc.) are used. Nevertheless, Tables 8.5 and 8.6 summarize available information to have an overview of the complexity control strategies for HEVC and reachable gains.

Table 8.5 – *Comparison with Grellert[GSK+13], Deng[DXJ+16] and Jimenez [JMMEDdM16] in terms of NET Error (in %) and BD-BR (in %) according to the target NET.*

	Proposed		Grellert		Deng		Jimenez	
Target NET	NET Error (%)	BD-BR (%)	NET Error (%)	BD-BR (%)	NET Error (%)	BD-BR (%)	NET Error (%)	BD-BR (%)
90%	0.87	0.06	-	-	-	-	0.21	0.23
80%	1.24	0.10	0.16	0.65	2.40	0.84	1.50	0.69
70%	1.22	0.22	-	-	-	-	0.83	2.02
60%	1.20	0.70	0.14	3.35	0.60	2.27	2.23	6.83
50%	1.40	1.78	-	-	-	-	-	-
40%	3.28	3.20	1.52	7.13	1.00	7.30	-	-
Av.	1.53		0.61		1.33		1.19	

Table 8.6 – *Comparison with Correa[1][CAAdSC16, CAdSC+13b] and Correa[2][CAAdSC15, CAdSCA15] in terms of NET Error (in %) and BD-BR (in %) according to the target NET.*

	Proposed		Correa[1]		Correa[2]	
Target NET	NET Error (%)	BD-BR (%)	NET Error (%)	BD-BR (%)	NET Error (%)	BD-BR (%)
90%	0.87	0.06	3.00	0.03	4.70	0.16
80%	1.24	0.10	3.00	0.23	6.37	0.35
70%	1.22	0.22	2.00	0.46	2.45	0.51
60%	1.20	0.70	1.00	0.80	1.41	0.74
50%	1.40	1.78	1.00	1.28	1.68	0.97
40%	3.28	3.20	2.00	3.98	3.42	1.19
Av.	1.53		2.00		3.34	

Tables 8.5 and 8.6 presents a comparison between our proposed *Encoding Time Control System* and complexity control techniques presented in Section 3.3 for target NET ranging from 90% to 40%. Results from techniques found in literature are implemented on reference software HM: Grellert[GSK+13], Deng[DXJ+16], Jimenez [JMMEDdM16], Correa[1][CAAdSC16, CAdSC+13b] and Correa[2][CAAdSC15, CAdSCA15].

This comparison is unfair to our method in terms of time, as the complexity overhead of the proposed complexity control techniques in literature would be higher in the context of a real time encoder, thus reducing their complexity reduction. In a real-time configuration (see 4.3.1), the computational overhead of our proposed method in the real-time optimized encoder Kvazaar is between 1.5% and 2.5%.

Table 8.5 shows that, even with an unfair comparison, our proposed *Encoding Time Control System* outperforms Grellert, Deng and Jimenez methods in terms of average compression efficiency across all the targets NET for available results. However, these three complexity control techniques are more accurate in terms of time control with an average NET Error of 0.61%, 1.33% and 1.19%, respectively, while our proposed solution achieved 1.53% of NET Error in average.

On the other hand, Table 8.6 shows that the time control accuracy of our proposed solution obtains better results than Correa[1] and Correa[2], with an average of NET error of 2% and 3.34%, respectively. Our proposed solution achieves better results compared to Correa[1] in terms of compression efficiency across the different targets NET except for target NET of 90%. Nevertheless, the experiments of Correa[1] are conducted on a subset of ten video sequences not including *Kimono*. As explained before, *Kimono* obtained outlier

results compared to all the video sequences. If *Kimono* video sequence is removed of the set of experiential video sequences, our results become 0.026%, 0.041%, 0.15%, 0.59%, 1.62% and 2.89% of BD-BR for targets NET of 90%, 80%, 70%, 60%, 50% and 40%, respectively.

Last but not least, Correa[2] is the most relevant and mature complexity control technique in literature. The technique is composed of coarse-, medium- and fine-granularity time control which reduce the complexity of the quad-tree partitioning but also of other parts of the encoding process using for example PU or RQT early termination. Our proposed method achieves better results in terms of BD-BR, from 90% to 60% target NET. For target ranging from 50% to 40%, Correa[2] obtains less encoding degradation. Our proposed method only adjusts the quad-tree partitioning process to scale and control the complexity of the encoding process which makes our approach generic (not linked to the encoding parameters such as the encoding presets of kvazaar) and easily portable. Correa[2] proposed technique affects many more parts of the encoding process which makes it more intrusive. In addition, it is important to notice that the experiments of Correa[2] were realized on only six high-resolution (1920×1080) sequences while our results are the average of 18 different video sequences belonging to the 5 classes A, B, C, D and E, each one corresponding to a specific resolution or video content.

8.4 Conclusion

The *Encoding Time Control System* presented in this chapter uses PI-based feedback control to reduce and control the encoding time in seconds by frame under a specific target. The control system sets the command $\hat{\Xi}$ of the *Tunable Complexity Frame Encoding* detailed in Chapter 7 to scale the complexity of the encoding process by adjusting the quad-tree partitioning process.

Experimental results have shown that our proposed method can limit the normalized encoding time per frame from 100% to 40%, with an average deviation from the target of only 1.53% in average. Target NET that required low reduction of complexity are performed with negligible encoding degradation, less than 0.7% of BD-BR for target NET ranging from 90% to 60%.

The method presented in this chapter is the completion of the main contributions presented in this document. The results confirm the utility and the applicability of our contributions in a concrete context. Next chapter concludes this document and presents main outlines of proposed research directions.

CHAPTER 9

Conclusion

Over the last few years, the Internet of Things (IoT) has become a reality. Forthcoming applications are likely to boost mobile video demand to an unprecedented level. In this context, designing energy-efficient HEVC real-time encoders is becoming a major challenge for software and hardware designers. This document has proposed a set of studies and methods aiming at scaling and controlling the complexity, and therefore the energy consumption, of the HEVC encoding process.

9.1 Research Contributions

Based on the state-of-the-art summarized in Chapter 3, the first contribution of this document, presented in Chapter 4, makes the connection between the approximate computing concepts and the applicative HEVC encoding process in order to control its energy consumption. A methodology is proposed to exploit the computation-skipping approximate computing concept. The methodology, named SSSR, explores at design time the Pareto relationship between computational complexity and application quality. The energy reduction *search space* is defined, and energy reduction opportunities are analyzed by considering different levels of granularity from global video parameters to prediction unit determination. This study demonstrates several elements: at the coarsest granularity, the energy of HEVC real-time Intra encoding is proportional to the resolution of the encoded video sequence. At medium granularity, the transform skipping method is not effective for reducing encoding energy. At a lower granularity, two MSSE algorithms have been studied; the *CTU level* has a potential of energy reduction up to 78.1% whereas the *IP level* offers at best 30% of energy reduction.

Following the methodology and the study of energy reduction opportunities carried out in Chapter 4, Chapters 5, 6, 7 and 8 are focused on quad-tree partitioning to reduce, scale, and finally control the complexity/energy consumption of the HEVC intra encoding process. The main contributions of this document can be partitioned into three categories.

1. **"One-shot" quad-tree partitioning prediction:**

    - Chapter 5 has presented an energy reduction scheme for HEVC encoders based on a "one-shot" CTU partitioning prediction technique that limits the recursive RDO process. The proposed energy reduction technique exploits the correlation between a CTU partitioning, its texture complexity and its luminance samples

variance. Experimental results showed that the proposed overall algorithm reduces the energy consumption by 60% on average for a bit rate increase of 3.95%.

- Considering that the variance of luminance samples can be used to predict the frame partitioning structure, Chapter 6 first proposes a fair comparison of characteristics extracted from the state-of-the-art, used for prediction of quad-tree partitioning. This comparison is performed using Machine Learning. These different characteristics are used as learning features in a Machine Learning framework. The comparison is realized under a real-time framework considering both information gain for representing characteristic suitability and computational overhead to compute the characteristic. Based on the best characteristics, Chapter 6 proposed a second method of "one-shot" prediction of quad-tree partitioning using a Machine Learning approach across data-mining classifiers. Quad-tree prediction is used to drastically limit the energetic consumption of the recursive RDO process. Experimental results showed that the new proposed technique is able to reduce the energy consumption of the considered HEVC encoder by 58% — including the additional algorithm overhead in a real-time encoder — for a bit rate increase of 3.6%.

2. **Complexity/energy scaling of HEVC encoding process:**

   - Based on an extended study of the "one-shot" quad-tree partitioning prediction based on a variance-aware statistical approach presented in Chapter 5, the prediction is made adjustable by parameters, offering a trade-off between energetic gains and compression efficiency. Experimental results show that the proposed algorithm is able to reduce the energy consumption of the HEVC encoder from 45% to 61% — including predictor overhead — for a bit rate increase ranging between 0.69% and 3.95%.
   - From the "one-shot" quad-tree partitioning prediction and following the SSSR methodology, Chapter 7 proposed a generic tunable complexity scheme of HEVC Intra encoding process, called *Tunable Complexity Frame Encoding*. The proposed CDM *Search Space Relaxation* scheme is used in the *Tunable Complexity CTU Encoding* to limit and scale the complexity of the HEVC Intra encoding process setting an integer number of depth levels tested by RDO process for a given CTU. Furthermore, the *Tunable Complexity Frame Encoding* exploits a method, called Constrain the Docile CTU (CDC), to allocate the complexity in a frame while minimizing the performance loss in terms of bitrate and quality allowing to select a real number of depth levels by frame tested by the RDO process. The proposed solution is able to scale the complexity of the encoding process down to the coarse partitioning prediction.

3. **Complexity/energy control of HEVC encoding process:**

   - Finally, to demonstrate the utility and the applicability of the major contributions of this document, Chapter 8 proposes a real-time control system that dynamically manages the encoding process to keep the encoding complexity under a specific target. The control system uses PI feedback loop mechanism to maintain the encoding time by frame under the required target. Experimental results show that our proposed method can limit the normalized encoding time per frame from 100% to 40%, with an average deviation from the target of only

1.53% in average. Target of encoding time that required low reduction of complexity are performed with negligible encoding degradation, e.g. less than 0.7% of BD-BR for normalized encoding time per frame ranging from 90% to 60%.

The contributions of Chapter 7 and 8 are not specific to the previous contributions and could be applied on other quad-tree partitioning predictions and tunable complexity schemes of encoding process, respectively.

9.2 Prospects – Future Works

The work presented in this document opens opportunities for future research on low-power video compression. This section presents main outlines of proposed research directions.

9.2.1 Dissemination and Exploitation

Kvazaar [Ult17], the real-time HEVC encoder software on which the contributions of this document have been developed, is intended to continue to improve in performance. Integrating all the tools described in this document into the open source project can improve encoding performance for the same complexity or achieve new configurations for emerging video formats, such as 4K, high frame rate (120FPS), according to HW capacities.

9.2.2 Performance Improvement

9.2.2.1 Encoding Quality Improvement

Results presented in Chapter 7 and Chapter 8 are directly linked to the performance of the *Coarse* CDM *Solution Predictor*. Figure 7.19 of Chapter 7 explicitly shows that if the performance of the starting point is improved, i.e. from the leftmost point, the overall results are improved.

A potential direction for future research would be to improve the quality of *Coarse* CDM *Solution Predictor* prediction using deep learning approach [KKS+18]. Using Convolutional Neural Networks (CNN), for example, to predict quad-tree partitioning in one-shot can be interesting and achieve good results. On the other hand, CNN implementation requires much computational resource and becomes a challenge in the context of embedded real-time video encoding.

9.2.2.2 Visual Quality Improvement

In a highly competitive market, the Quality of Experience (QoE) is one of the main concerns of communication service providers. In the context of video compression, traditional metrics such as BD-BR and BD-PSNR are not sufficient to assess the quality of a video as perceived by the human visual system. An entire research community works on representative metrics to monitor the video quality.

An interesting perspective for future work would be to ensure that the systems proposed in this document scale and control the complexity/energy consumption maximizing the subjective quality of the video sequence. Two approaches could be explored.

- **Temporal complexity distribution:** in a context of high frame rate video sequence (60 or 120 Frames per Second (FPS) for example), visual quality does not necessarily have to be constant across all frames for maximizing the Quality of Experience (QoE). The optimal setup depends on the human visual system and the used display device

technology. In this case, it would be possible to add a higher level to our control system (see Figure 8.8) in order to optimize visual quality of the encoding.

- **Visual quality-aware complexity allocator** (in frame): our system currently uses the CDC complexity allocator (detailed in Section 7.3) to distribute quad-tree constraints among the CTU within a frame. Using a salience map for example, it would be possible to distribute the quad-tree constraints in order to maximize the visual quality of the most viewed areas of the frame [CZHD18] and reach higher visual quality.

9.2.3 Energy-Aware Rate Control

Rate-Control (RC) and Adaptive Rate Control (ARC) are important stakes in video application industry. In a broadcasting environment for example, programs are often jointly encoded into a fixed-bandwidth channel by taking advantage of statistical-multiplexing techniques. Rate control algorithms are mainly composed of three steps which can be considered at the top of an existing encoder: bitrate allocation, model estimation and encoding parameters selection which are based on feedback loop construction. However, numerous algorithms are reported in the literature, typically improving on the past results and also taking into account the complexity of new video codecs such as HEVC.

A potential direction for future works would be to use the tunable complexity system proposed in this document coupled with an energy model of the encoding process [ABIAFC+17] to build an energy-aware rate control for embedded real-time encoding.

Further research could take into account energy consumption of the transmission and decoding part in the energy model to design a end-to-end energy-aware real-time HEVC solution from video capture to display.

9.2.4 Extension of Proposed Works

9.2.4.1 Inter frame Encoding Extension

Due to the real-time embedded context, the main contributions of this document targeted HEVC Intra software encoding. Nevertheless, the HEVC Inter encoding process complexity has significantly increased compared to the previous standard H.264/AVC. As detailed in Section 2.3.1, different from intra PU splitting modes whose shape must be square, inter PU splitting modes can be rectangle, including 8 possible splitting modes (detailed in Table 2.1 and illustrated in Figure 2.9).

Possibilities for future works would be to extend the current quad-tree prediction to the HEVC inter encoding process in order to scale and control the encoding process complexity. New features will have to be taken into account such as motion information (across MV for example) to maximize the prediction accuracy [CAA+13].

9.2.4.2 Future Video Coding Extension

The Joint Video Expert Team (JVET) has been recently investigating several new coding solutions to show the evidence of developing a new standard Versatile Video Coding (VVC) [Sul18] with coding capability beyond HEVC. These new tools increase the coding efficiency by up to 40% compared to HEVC. However, this gain comes with a significant complexity increase of 10 times the HEVC complexity at both encoder and decoder sides [SHDP17]. Inter and Intra coding complexities become an obstacle to the development of the VVC standard and its deployment on embedded platforms.

Possible angular modes for Intra predicted blocks went from 33 modes in HEVC to 65 modes in VVC. VVC inherits the block-based hybrid video coding architecture existing in HEVC, but it contains a new and more flexible block partition structure. VVC firstly adopts the Quad-Tree plus Binary-Tree (QTBT) structure [ACZ+15] for the CU sizes from 8×8 to 128×128. In addition to the Quad-Tree (QT), present in HEVC, and Binary-Tree (BT) partitions, VVC currently performs a third partition type called Triple-Tree (TT). The BT partition consists of symmetric horizontal splitting and symmetric vertical splitting while the TT partition allows horizontal triple-tree splitting and vertical triple-tree splitting corresponding to split the CU in three blocks with the middle blocks size equal to the half of the CU.

A potential direction for future research would be to extend works presented in this document to reduce and, in the long term, scale the encoding complexity of the future video coding standards. QT partitioning prediction presented in this document could be extended to BT and TT partitioning. VVC being under development, only few techniques have been published regarding complexity reduction of new partitioning structure [JASY17, WWZ+17, LJHC18]. Considering the recent breakthroughs of deep learning as classification techniques, we have initiated works on QT/BT/TT partitioning prediction using CNN-based networks for the future video coding standard.

ANNEXE A

French Summary

A.1 Introduction

A travers les canaux de diffusion traditionnels comme la télévision ou les nouvelles plateformes en ligne comme Youtube ou les réseaux sociaux, la vidéo numérique est omniprésente dans nos vies. La transmission de vidéos représente déjà la grande majorité du trafic de données dans le monde ; Cisco [CISCO] rapporte que 73% du trafic IP total est consacré à la vidéo en 2016 et l'étude estime que cette part du trafic atteindra 82% en 2021. Sandive [SANDIVE] précise que 60% du trafic nord-américain aux heures de pointe le soir est dû à quatre services de vidéo à la demande en 2016 : Netflix, YouTube, Amazon Video et Hulu. Ce nombre a doublé en cinq ans. L'expansion des réseaux à haut débit et les progrès de la micro-électronique permettent aujourd'hui de regarder des vidéos partout et à tout moment.

D'un côté, grâce aux nouveaux formats émergents tels que le format 4K Ultra High Definition (UHD), les vidéo 360°, le multi-vues et la réalité virtuelle, la quantité de données vidéo augmente continuellement avec le temps. Le stockage et la transmission de ces expériences vidéo nécessitent énormément de données. De l'autre côté, les appareils disponibles pour enregistrer, afficher et distribuer des vidéos de très hautes résolutions, tels que les smart-phones deviennent abordables pour les consommateurs. En conséquence, des gains en compression vidéo sont nécessaires pour permettre l'utilisation de ces technologies vidéo immersives.

A.1.1 Problématique

Pour répondre à la demande croissante de stockage et transmission vidéo sur Internet, le standard High Efficiency Video Coding (HEVC) a été normalisé au début de l'année 2013 [SBS14, SOHW12, Wie15]. Par rapport à la norme précédente AVC, le standard HEVC est capable de réduire par deux le débit d'une vidéo pour une qualité vidéo similaire [TWM+16, VVHH12]. Ce gain réduit l'énergie nécessaire à la transmission des vidéos. D'autre part, la complexité de calcul des encodeurs a été considérablement augmentée. Dans le cas du codage Intra, la complexité supplémentaire apportée par le HEVC est principalement due à la nouvelle structure de partitionnement en blocs de pixels appelé Coding Tree Unit (CTU) et au nombre de modes de prédiction Intra plus élevé.

La principale limitation des systèmes embarqués récents, notamment en termes de performances de calcul, vient de la limite énergétique des batteries. Une meilleure gestion des

systèmes embarqués et de leur consommation énergétique est donc cruciale pour prendre en charge de nouvelles fonctionnalités sans en altérer la fonctionnalité. La limitation énergétique est une contrainte majeure pour les applications d'image et de vidéo ; l'encodage et le décodage vidéo sont par exemple les algorithmes les plus énergivores des smartphones [CH10]. Une grande partie des systèmes sont susceptibles d'intégrer un encodeur HEVC à long terme et nécessiteront d'être *éconergétiques*. Par conséquent, représentant un sérieux défi pour les système temps réel embarqués, le but de ces travaux est de réduire et contrôler la consommation énergétique du processus d'encodage HEVC.

A.1.2 Plan

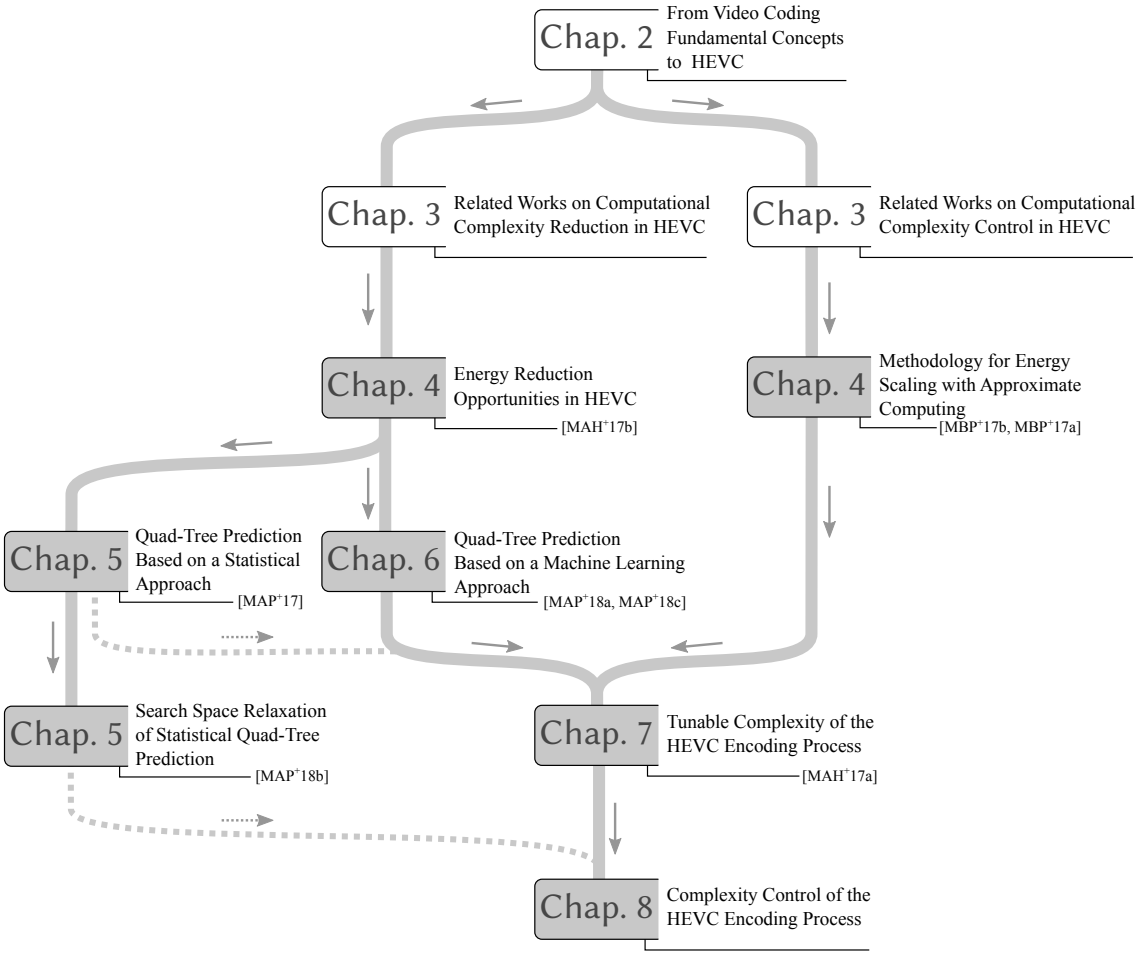


FIGURE A.1 – *Organisation des contributions de ce document chapitre par chapitre.*

Ce résumé reprend l'organisation principale des chapitres du corps de ce document ; la Figure 1.1 résume l'organisation des contributions de ce document chapitre par chapitre. La Section A.2 présente un aperçu du processus d'encodage d'image Intra par le standard HEVC et donne un aperçu des méthodes les plus récentes de réduction de complexité du processus d'encodage HEVC.

La Section A.3 présente deux techniques de prédiction de découpe CTU. La première méthode présentée dans la Section A.3.1 utilise la variance des pixels de l'image afin de prédire la découpe CTU avant d'avoir commencé le processus d'encodage. La seconde

méthode présentée dans la Section A.3.2 utilise l'intelligence artificielle afin de prédire la découpe d'un CTU.

La Section A.4 présente notre solution pour ajuster dynamiquement la complexité d'encodage HEVC d'une image. Dans un premier temps, la Section A.4.1 propose d'utiliser la prédiction de découpe CTU présentée dans la précédente section afin d'ajuster la complexité d'encodage à l'échelle de blocs CTU. Dans un second temps, la Section A.4.2 combine la solution présentée précédemment avec un allocateur de complexité afin d'ajuster la complexité d'encodage à l'échelle d'une image.

La Section A.5 présente finalement un système permettant de contrôler le temps d'encodage HEVC selon une consigne de temps d'exécution par image. Cette section rassemble les principales contributions présentées dans ce document et permet de montrer l'utilité et l'applicabilité de l'ensemble des travaux présentés dans ce document. Pour finir, la Section A.5 conclut ce résumé.

A.2 État de l'Art

A.2.1 Processus d'Encodage HEVC : Vue d'Ensemble

L'encodeur HEVC est basé sur un codeur vidéo hybride classique qui combine à la fois des prédictions Inter-image et Intra-image. La Figure A.2 présente une vue d'ensemble du processus d'encodage HEVC.

Pour l'encodage, chaque image est divisée en blocs de même taille appelés CTU. L'encodeur HEVC commence par prédire les blocs à partir de leur environnement. L'encodeur peut choisir de prédire le bloc à partir de bloc d'une autres images (inter-image prédiction) ou prédire le bloc à partir des blocs voisins au sein de la même image (intra-image prédiction). Après avoir calculé cette prédiction, l'encodeur calcule les résidus (erreurs de prédiction) en soustrayant l'image de la prédiction par celle de l'image d'origine. Les résidus sont ensuite transformés par une transformation spatiale linéaire, quantifiée et codée par entropie.

L'encodeur HEVC contient également la boucle de traitement du décodeur puisque l'image décodée est requise par l'encodeur pour réaliser les prédictions Intra et Inter-images. Cette partie décodeur est composée d'une quantification inverse et d'étapes de transformation inverse qui reconstruisent l'information résiduelle. Les résidus sont ensuite ajoutés à la prédiction pour générer l'image décodée. Enfin, la qualité visuelle de l'image reconstruite est améliorée à l'aide de deux filtrages : Sample Adaptive Offset (SAO) and Deblocking Filter (DBF).

Pour obtenir les meilleures performances entre compression et qualité, l'encodeur HEVC effectue une recherche exhaustive, appelée processus Rate-Distortion Optimization (RDO), testant tous les partitionnements possibles pour chaque bloc CTU. Durant le processus de RDO, un CTU est divisé en plus petits blocs, appelés Coding Block (CB), eux-mêmes pouvant être sous-divisés en plus petits CB. Dans le standard HEVC, la taille des CB peut être égale à 64×64, 32×32, 16×16, 8×8 ou bien 4×4 pixels. Chaque taille de blocs est prédite et le processus de RDO calcule le coût Rate-Distortion (RD) de ces blocs en fonction du taux de compression et de la qualité visuelle du bloc. Pour un CTU, le processus de RDO sélectionne ensuite la meilleure découpe, i.e. celle qui minimise le coût RD. En d'autres termes, le processus de RDO définit un espace de recherche (correspondant aux découpes testées) dans lequel il cherche à trouver la meilleure solution.

La Figure A.3 montre un exemple d'un CTU de taille 64×64 pixels divisé en plusieurs CB. Les blocs gris de la Figure A.3 représentent la découpe sélectionnée par le processus

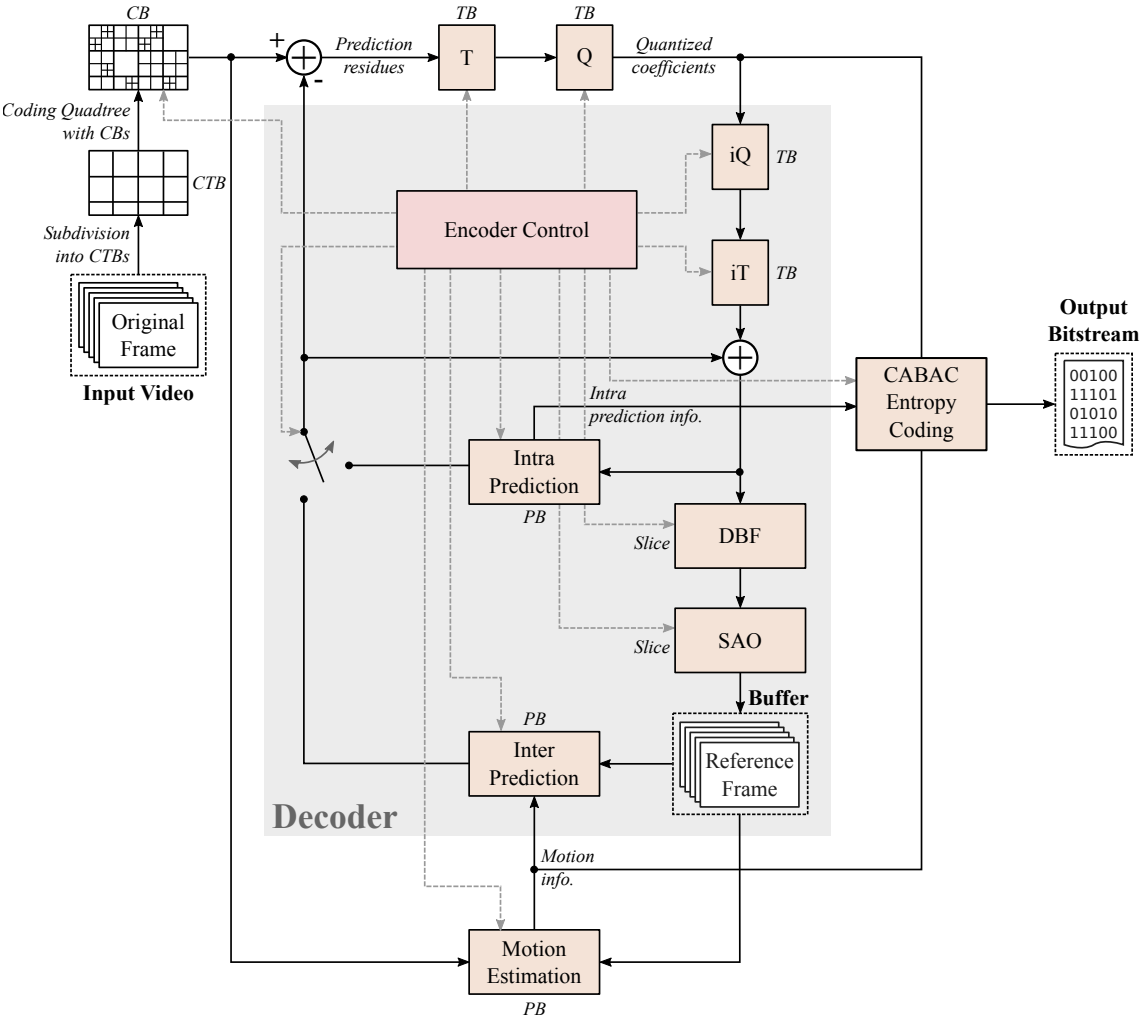


Figure A.2 – *Schéma-bloc d'un encodeur* Hevc.

de RDO après avoir testé toutes les découpes possibles. La Figure A.3 indique également la relation entre taille de CB et la profondeur de découpe.

A.2.2 Techniques de Réduction et Contrôle de Complexité d'Encodeur HEVC

Depuis que le standard Hevc a été défini, de nombreux travaux ont été publiés dans la littérature, présentant des méthodes pour réduire la complexité du processus d'encodage. La plupart de ces techniques visent à réduire la complexité due au processus exhaustif de RDO. Ces techniques peuvent être classées parmi les quatre grandes catégories suivantes :

- la découpe du CTU en CB,
- la sélection du mode de prédiction entre intra, inter ou SKIP,
- la sélection du mode de prédiction intra,
- la découpe des résidus.

La Figure A.4 montre le pourcentage de techniques de réduction de complexité pour chaque catégorie : la découpe du CTU en CB (QT) représentant 73%, la sélection du mode

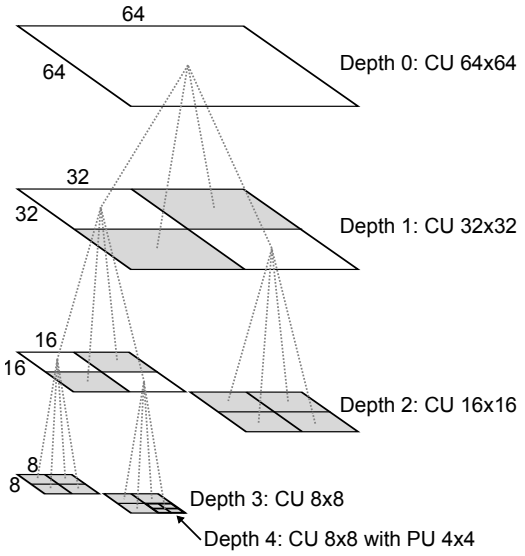


Figure A.3 – *Exemple de découpe CTU en CB.*

de prédiction entre intra, inter ou SKIP (PU) représentant 15%, la sélection du mode de prédiction intra (IM) représentant 29%, la découpe des résidus (RQT) représentant 15%.

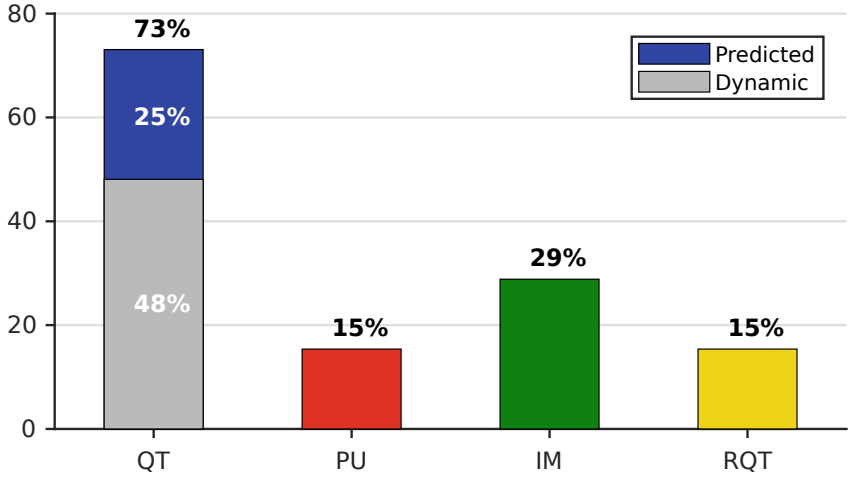


Figure A.4 – *Pourcentage de techniques de réduction de complexité par catégorie.*

Comme le montre la Figure A.4, la façon la plus courante pour réduire la complexité processus d'encodage HEVC est de simplifier la recherche de la découpe du CTU en CB. En effet, nous avons montré que cette partie du processus d'encodage est la plus complexe et nécessite 78% du temps d'encodage total [MAH+17b]. Pour la suite de ces travaux, il a été choisi de travailler sur la recherche de découpe CTU afin de réduire et contrôler la complexité du processus d'encodage.

A.3 Prédiction de Découpe en Blocs de Pixels (CTU)

Il a déjà été montré que le partitionnement d'un CTU pendant le processus de RDO est fortement lié à la complexité de la texture de l'image qui peut être statistiquement

représentée par la variance des blocs dans le cas de prédictions Intra [KSH13, BC15, WX16] comme illustré sur la Figure A.5.

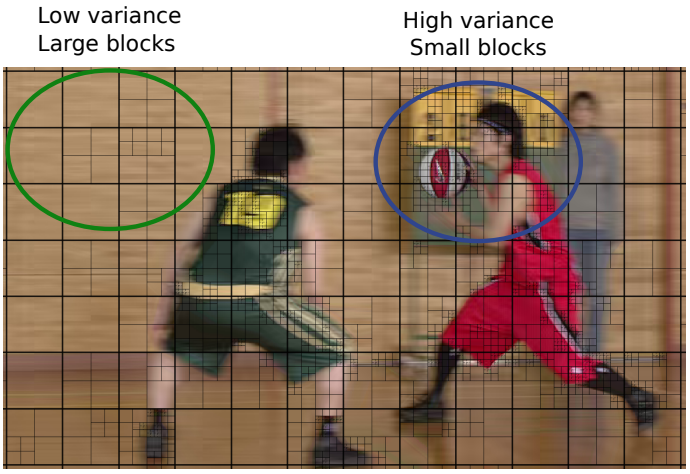


Figure A.5 – *Partitionnement de la 6e image de la séquence* BasketballDrive. *Le cercle vert (resp. bleu) montre que les régions ayant les variances les plus basses (resp. les plus hautes) ont tendance à être codées avec de plus grands CB (resp. plus petits).*

Une description de découpe CTU est nécessaire pour décrire explicitement la prédiction et pour pouvoir forcer l'encodeur à encoder une décomposition spécifique. La Figure A.6 illustre la représentation choisie pour décrire la découpe d'un CTU. Chaque CTU est représenté sous la forme d'une carte de profondeur de 8×8 éléments, appelée CTU Depth Map (CDM). Chaque élément de la matrice représente la profondeur d d'un bloc de 8×8 pixels du CTU. Une profondeur de 4 dans la CDM correspond à 4 CB de 4×4 pixels dans la décomposition du CTU.

3	3	2	2	1	1	1	1
3	3	2	2	1	1	1	1
2	2	3	3	1	1	1	1
2	2	3	4	1	1	1	1
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2

Figure A.6 – *Matrice* CDM *de la découpe CTU de la Figure A.3. Chaque élément de la matrice représente la profondeur d'un bloc de 8 par 8 pixels du CTU.*

Les sections suivantes présentent deux solutions proposées dans ce document pour prédire la découpe CTU avant le début du processus d'encodage. Le but de ces deux techniques est de remplir une CDM pour chaque CTU.

A.3.1 Prédiction de Découpe CTU Statistique

La première technique de prédiction de découpe CTU présentée dans ce document est basée sur une analyse statistique de la variance des pixels de l'image à encoder [MAP+17]. Pour cette étude, chaque CB de profondeur d est classé selon l'une des deux catégories suivantes :

Fusionné ou *Non Fusionné*. Un CB appartient à la population *Non Fusionné* lorsque le processus de RDO choisit d'encoder le CB à la profondeur actuelle d, tandis que lesCB appartient à la population *Fusionné* lorsque le processus de RDO choisit d'encoder le CB à une profondeur inférieure $d' > d$.

Utilisant la Cumulative Distribution Function (CDF) de la variance des pixels des CB de la population *Non Fusionné*, la méthode proposée calcule un seuil à partir duquel il y a plus de chance que le CB courant soit encodé en un CB de profondeur inférieur (encodé en un plus gros bloc). Les seuils étant variables en fonction du contenu de la vidéo, ils sont calculés à partir d'une image d'apprentissage, i.e. une image où le processus de RDO est appliqué de manière exhaustive afin d'obtenir la découpe optimale. De l'encodage de cette image sont extraits les découpes de CTU afin de calculer la moyenne et l'écart type des CB de la population *Non Fusionné*. Basés sur une modélisation hors-ligne, les seuils sont ensuite calculés pour chaque taille de CB à partir de ces deux moments.

La méthode propose ensuite d'utiliser ces seuils à travers un algorithme afin de prédire la découpe d'un CTU. L'algorithme part d'une découpe initiale la plus fine (Prediction Block (PB) de taille 4 par 4) en initialisant toute la CDM avec la valeur 4. Depuis la profondeur maximale ($d = 4$) jusqu'à la profondeur minimale ($d = 0$), il utilise les seuils afin de fusionner ou non les PB comme illustré sur la Figure A.7.

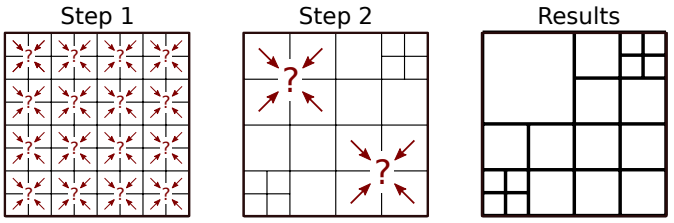


FIGURE A.7 – *Illustration de l'algorithme de prédiction de découpe CTU utilisé par la méthode statistique.*

L'algorithme est ascendant ; lorsque des PB n'ont pas été fusionnés à une profondeur inférieure, l'algorithme garde cette découpe et n'essaie plus de les fusionner. Cette méthode est capable de prédire correctement 49.89% des CTU en terme de surface d'image avec une erreur moyenne de 0.79% en terme de niveaux de profondeur.

Pour conclure, cette approche prouve qu'il est possible de prédire la découpe CTU à partir de la texture de l'image à encoder. Cependant, elle nécessite un apprentissage à la volée (pendant l'encodage) utilisant des images d'apprentissage.

A.3.2 Prédiction de Découpe CTU Basé sur l'Intelligence Artificielle

La deuxième technique de prédiction de découpe CTU présentée dans ce document utilise l'intelligence artificielle. Considérant que la variance des PB peut être utilisée pour prédire la découpe de CTU, nous avons réalisé une comparaison des caractéristiques extraites de l'état de l'art pour la prédiction de découpe CTU en utilisant l'intelligence artificielle [MAP+18a]. L'intelligence artificielle est un sous-domaine interdisciplinaire de l'informatique qui vise à remplacer les solutions conçues manuellement pour extraire des informations à partir de données saisies dans de nombreux domaines d'application.

Les caractéristiques les plus pertinentes et calculables en temps réel résultant de l'étude précédente sont utilisées comme attributs pour entraîner des classificateurs (classification supervisé C4.5) afin de prédire si un PB doit être fusionné ou non. La méthode utilise cette prédiction à travers un algorithme afin de prédire la découpe d'un CTU. L'algorithme part

d'une découpe initiale la plus fine (PB de taille 4 par 4) en initialisant toute la CDM avec la valeur 4. Depuis la profondeur maximale ($d = 4$) jusqu'à la profondeur minimale ($d = 0$), il utilise les prédictions des classificateurs afin de fusionner ou non les PB comme illustré sur la Figure A.8.

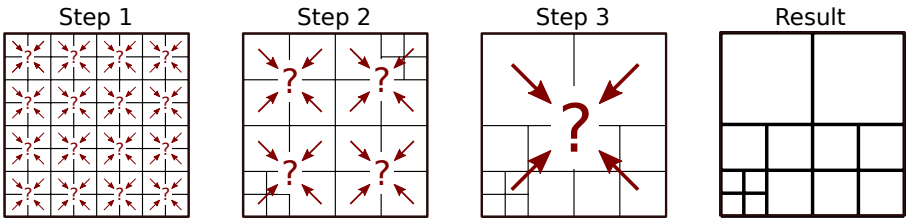


FIGURE A.8 – *Illustration de l'algorithme de prédiction de découpe CTU utilisé par la méthode basé sur l'intelligence artificielle.*

Pour cette méthode, l'algorithme est moins bridé et ne prend pas en compte les décisions prises au niveau de profondeurs inférieures. Cette méthode est capable de prédire correctement 52.63% des CTU en terme de surface d'image avec une erreur moyenne de 0.67% en terme de niveaux de profondeur.

Pour la prédiction de découpe CTU, les résultats expérimentaux montrent que la méthode basée sur l'intelligence artificielle obtient de meilleurs résultats en moyenne que la méthode statistique. De plus, la méthode basée sur l'intelligence artificielle ne nécessite pas d'image d'apprentissage pour prédire la découpe CTU. Ainsi, la prédiction de découpe CTU utilisant l'intelligence artificielle est utilisée comme point de départ dans les sections suivantes.

A.4 Encodeur HEVC à Complexité Ajustable

La Section A.3.2 a présenté une technique de prédiction de découpe CTU, nommée *Coarse* CDM *Solution Predictor* basée sur l'intelligence artificielle et illustrée par la Figure A.9. Pour un CTU donné, cette technique prédit une CTU Depth Map (CDM) avant le début du processus de RDO. Cette prédiction de découpe CTU est utilisée dans les sections suivantes comme base afin de contrôler la complexité du processus d'encodage Intra HEVC.

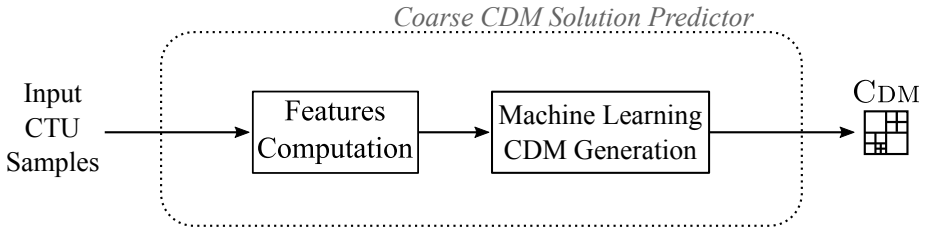


FIGURE A.9 – *Digramme blocs de prédiction de découpe CTU : "Coarse* CDM *Solution Predictor".*

A.4.1 Complexité Ajustable de la Recherche de Découpe CTU

Dans HEVC, l'exploration de tout l'espace de recherche par le processus de RDO pour trouver la meilleure découpe CTU consiste à tester toutes les tailles possibles de PB, c'est-à-dire tester tous les niveaux de profondeur de la décomposition CTU (voir Figure A.3).

Pour limiter et contrôler la complexité du processus d'encodage HEVC, la solution proposée est de tester un nombre sélectionné de niveaux de profondeur autour de la CDM prédite.

La Figure A.10 illustre la solution proposée, nommée *Tunable Complexity CTU Encoding*, pour limiter et piloter la complexité du processus d'encodage HEVC.

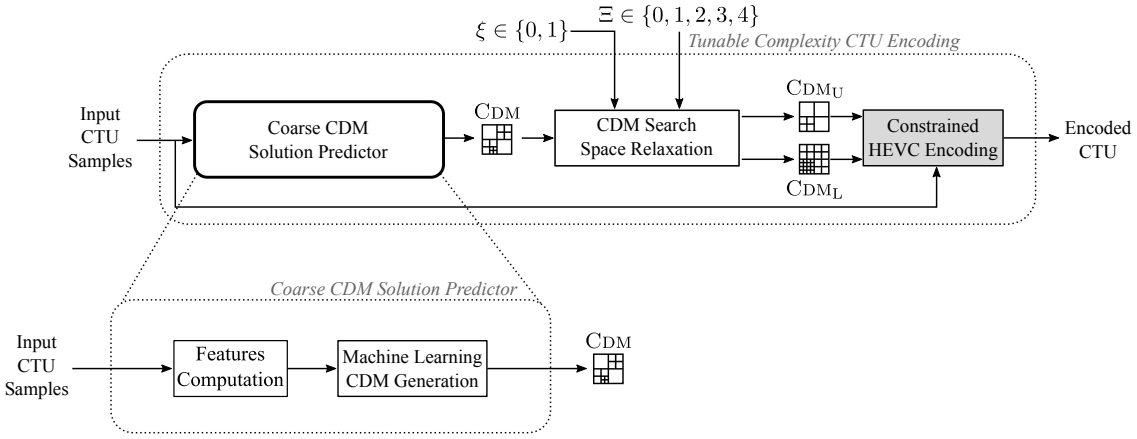


FIGURE A.10 – *Digramme blocs de l'algorithme de génération d'intervalle de découpe CTU : "Tunable Complexity CTU Encoding"*

Le système décrit par la Figure A.10 prend en entrée les pixels d'un CTU et utilise la méthode de prédiction de découpe CTU présentée dans la Section A.3.2 (*Coarse CDM Solution Predictor*) pour prédire la CDM associée au CTU en entrée. La CDM est alors utilisée pour générer un intervalle de deux CDM : une CDM_U supérieure et une CDM_L inférieure, écartées d'un nombre de niveaux de profondeur choisi. Enfin, l'encodeur HEVC est forcé d'appliquer uniquement le processus de RDO dans l'intervalle défini par CDM_L et CDM_U.

L'algorithme utilisé pour générer CDM_L et CDM_U, nommé CDM *Search Space Relaxation*, est illustré par la Figure A.11.

L'algorithme CDM *Search Space Relaxation*, illustré dans la Figure A.11, prend une CDM et un paramètre Ξ en entrées et génère $\text{CDM}_L(\Xi)$ et $\text{CDM}_U(\Xi)$. Le paramètre Ξ définit le nombre de niveaux de profondeur entre $\text{CDM}_L(\Xi)$ et $\text{CDM}_U(\Xi)$; la découpe d'un CTU étant sur 5 niveaux de profondeur (de $d = 0$ à $d = 4$), Ξ est inclus dans l'intervalle $\Xi \in \{0, 1, 2, 3, 4\}$. Lorsque $\Xi = 0$, un seul niveau de profondeur, correspondant à une CDM en entré, est testé alors que lorsque $\Xi = 4$, tous les niveaux de profondeur (de $d = 0$ à $d = 4$) sont testés.

Pour ajouter un niveau de profondeur autour de la CDM en entré, i.e quand $\Xi = 1$, l'algorithme CDM *Search Space Relaxation* se divise en deux étapes :

1. La première étape appliquée par l'algorithme pour générer $\text{CDM}_U(\Xi)$ fusionne tous les groupes de quatre PB voisins (dans l'ordre du Z-scan) avec la même profondeur de la CDM en entrée. Le nombre de PB fusionnés par l'algorithme dépend de la CDM en entrée et est imprévisible. Les blocs fusionnés sont représentés en rouge dans Figure A.11.

2. La deuxième étape appliquée par l'algorithme pour générer $\text{CDM}_L(\Xi, \xi)$ divise en 4 sous PB tous les PB qui n'ont pas été fusionnés pendant l'étape 1 de l'algorithme. Les blocs divisés sont représentés en bleu dans la Figure A.11. La distance en niveaux de profondeur obtenue entre CDM_L et CDM_U est exactement égale à 1. Lorsqu'il est

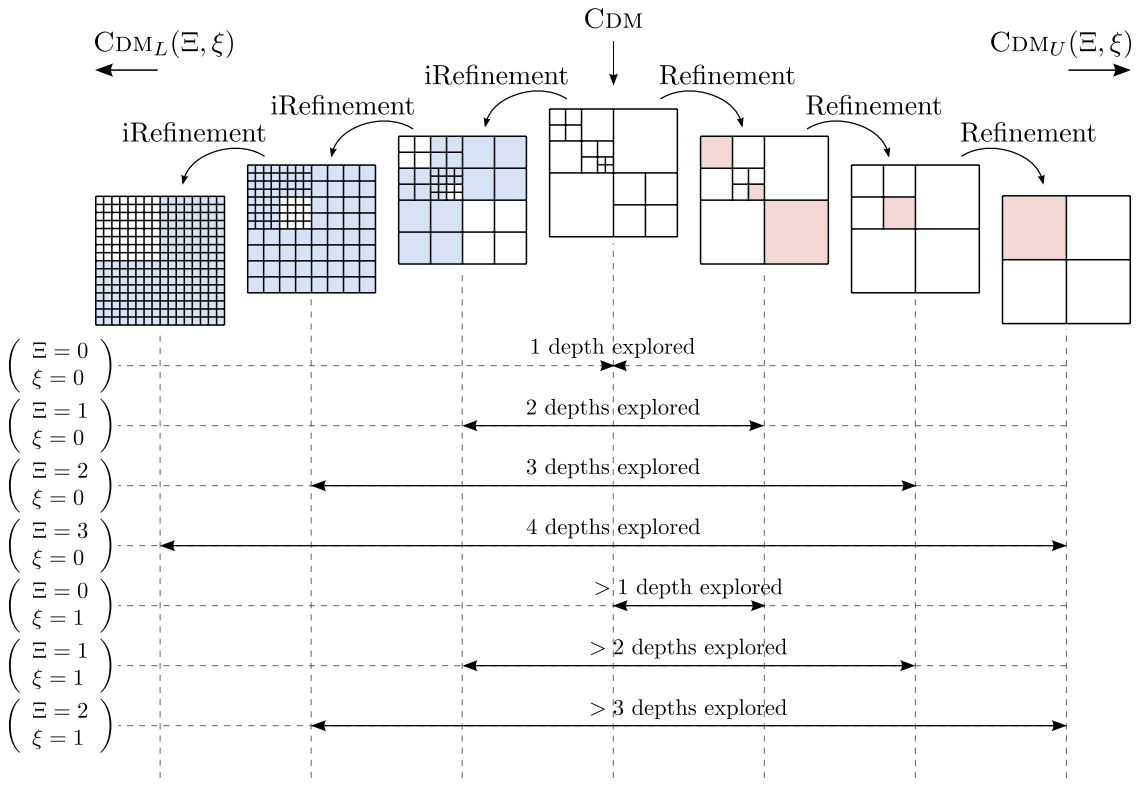


FIGURE A.11 – *Illustration de l'algorithme "CDM Search Space Relaxation". Les blocs fusionnés et divisés sont représentés respectivement en rouge et en bleu. Le nombre de niveaux de profondeur testé lorsque le processus de découpe CTU est contraint est indiqué selon le couple de paramètres* (Ξ,ξ).

contraint entre CDM$_L(1)$ et CDM$_U(1)$), le processus d'encodage teste exactement 2 niveaux de profondeur répartis autour de la CDM d'entrée.

Pour ajouter plus d'un niveau de profondeur, les étapes 1 et 2 sont répétées sur CDM$_U(\Xi)$ et CDM$_L(\Xi)$, respectivement, comme illustré dans Figure A.11. Les deux passes de l'algorithme génèrent un intervalle entre CDM$_L$ et CDM$_U$ avec un nombre prévisible et exact de niveaux de profondeur répartis autour de la CDM d'entrée.

Pour conclure, cette section présente l'algorithme CDM *Search Space Relaxation*, utilisé pour étendre l'espace de recherche autour de la CDM prédite par l'algorithme *Coarse* CDM *Solution Predictor*, décrit à la Section A.3.2. Cependant, le nombre de niveaux de profondeur défini par le paramètre Ξ est limité à une valeur entière comprise entre 0 et 4.

A.4.2 Complexité Ajustable de l'Encodeur Image par Image

Pour permettre de limiter et piloter la complexité d'encodage plus finement, cette section présente une méthode pour répartir la complexité, appelée Constrain the Docile CTU (CDC), à l'intérieur d'une image tout en minimisant la perte de performance RD [MAH+17a].

Dans le cas où différentes contraintes de réduction de complexité d'encodage, plus ou moins fortes, doivent être appliquées à l'intérieur d'une image, le répartiteur de complexité CDC propose d'appliquer les contraintes les plus fortes sur les CTU qui obtiennent les coûts RD les plus faibles. Cependant, sans effectuer l'encodage complet de l'image, les

coûts RD ne sont pas connus. En se basant sur une étude prouvant la corrélation entre les coûts RD des images successives d'une séquence vidéo, la méthode propose d'utiliser les coûts RD de l'image précédente pour répartir la contrainte. En utilisant le répartiteur de complexité CDC, il est maintenant possible de contraindre l'encodage d'une image avec un nombre décimal de niveaux de profondeur. Ce système, nommé *Tunable Complexity Frame Encoding*, est illustré par la Figure A.12.

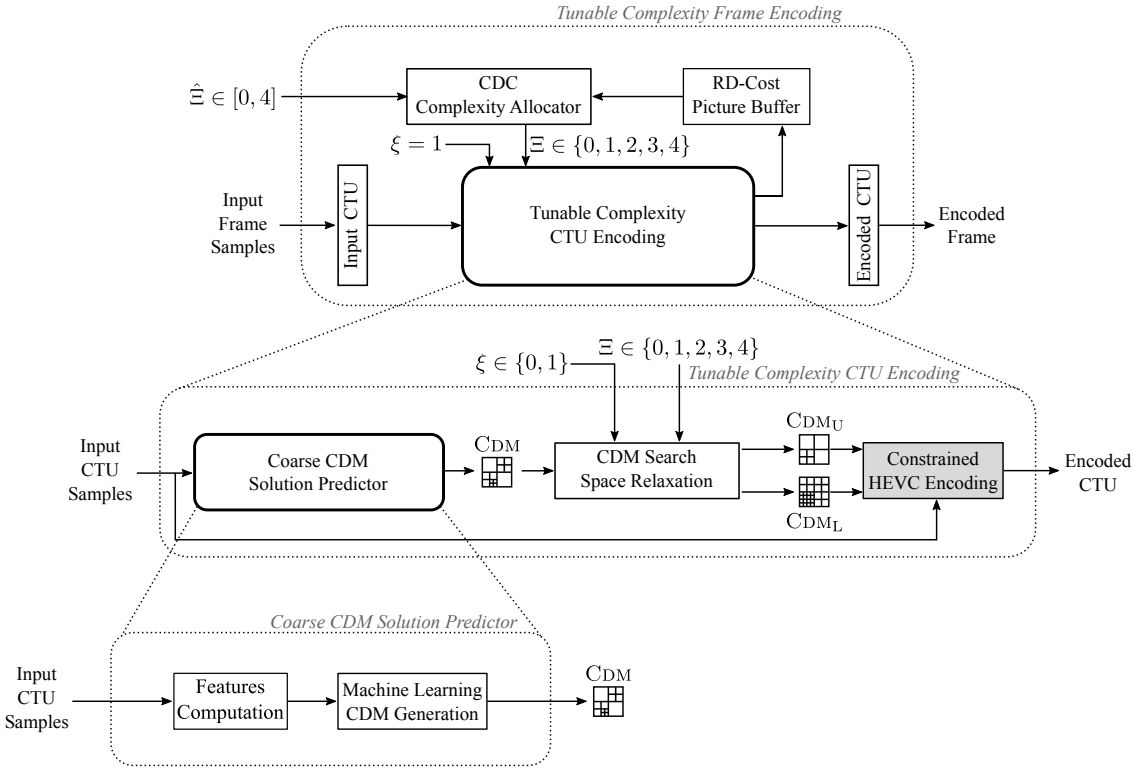


Figure A.12 – *Schéma bloc permettant de limiter et piloter la complexité d'encodage plus finement : "Tunable Complexity Frame Encoding".*

Le système décrit par la Figure A.12 prend une image et un paramètres $\hat{\Xi}$ en entrée et contraint le processus d'encodage à ne tester que le nombre décimal de niveaux de profondeur $\hat{\Xi}$ pour encoder l'image d'entrée. Le répartiteur de complexité CDC divise le nombre décimal $\hat{\Xi}$ en deux parties : la partie entière et la partie fractionnaire. La partie entière de $\hat{\Xi}$ définit le nombre minimum de niveaux de profondeur $\lfloor\hat{\Xi}\rfloor$ testés pour encoder tous les CTU de l'image d'entrée. La partie décimale de $\hat{\Xi}$ définit le nombre de CTU qui teste un niveau de profondeur supplémentaire $\lceil\hat{\Xi}\rceil$. Le répartiteur de complexité CDC utilise les coûts RD de l'image précédente stockée dans un tampon pour sélectionner quels CTU teste $\lfloor\hat{\Xi}\rfloor$ ou $\lceil\hat{\Xi}\rceil$ niveaux de profondeur. Les CTU ayant obtenu les coûts RD les plus élevés à l'image précédente sont contraint à tester $\lceil\hat{\Xi}\rceil$ niveaux de profondeur, le reste des CTU sont contraint à tester $\lfloor\hat{\Xi}\rfloor$ niveaux de profondeur. Le répartiteur de complexité CDC transmet la valeur appropriée du paramètre Ξ au *Tunable Complexity CTU Encoding* décrit à la Section A.4.1 pour contraindre la complexité du CTU courant.

A.5 Contrôle du Temps d'Encodage d'Encodeur HEVC

A.5.1 Présentation du Système Complet

Le but principal de cette section est d'utiliser le système de *Tunable Complexity Frame Encoding* présenté dans la section précédente pour construire un système de contrôle, en temps réel, de la complexité d'encodage. Comme ce système de contrôle sert de vitrine pour démontrer l'utilité et l'applicabilité des principales contributions de ce document, nous avons choisi de concevoir un système de contrôle capable de cibler un temps d'encodage par image.

Le système de *Tunable Complexity Frame Encoding* peut être modélisé par un retard unitaire de gain K_h comme illustré par la Figure A.13. Entre le temps de sortie du processus d'encodage et la sortie du système y, une perturbation w est ajoutée, qui correspond au l'approximation de la mesure, aux perturbations matérielles et logicielles ou à la variation temporelle de complexité d'encodage due au contenu de la séquence vidéo en entrée.

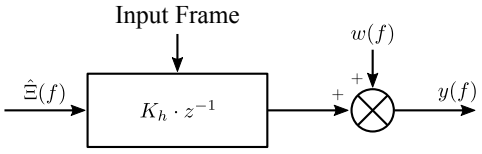


Figure A.13 – *Modélisation du système par un retard unitaire de gain K_h.*

Le gain K_h est modélisé en fonction de la commande $\hat{\Xi}$, la résolution de la vidéo et la valeur du paramètre d'encodage Quantization Parameter (QP). Afin d'asservir le système, un régulateur PI est implémenté comme système de contrôle. Les gains K_p et K_i correspondant à l'action proportionnelle et intégrale, respectivement, sont réglés afin de maximiser la précision et la stabilité du système.

Pour finir la Figure A.14 représente l'ensemble du système, écrit de manière hiérarchique, permettant de contrôler le temps d'encodage image par image selon une cible en seconde donnée.

Au plus haut niveau, le contrôleur Proportional–Integral (PI) prend en entrée le temps d'encodage ciblé par image et le temps d'encodage de l'image précédente pour définir la bonne valeur du paramètre $\hat{\Xi}$ transmise au *Tunable Complexity Frame Encoding* système. Le paramètre $\hat{\Xi} \in [0, 4]$ correspond au nombre de niveaux de profondeur testés pendant le processus de RDO pour l'image courante. Le paramètre $\hat{\Xi}$ permet de réduire la complexité du processus d'encodage Intra pour l'image courante.

Le *Tunable Complexity Frame Encoding* système, détaillé dans Section A.4.2, reçoit l'image courante à encoder et le paramètre $\hat{\Xi}$ et contraint le processus d'encodage à tester uniquement le nombre décimal des niveaux de profondeur $\hat{\Xi}$ pour l'image courante. Le nombre décimal de niveaux de profondeur $\hat{\Xi}$ est envoyé au répartiteur de complexité CDC pour définir le paramètre $\Xi \in \{0, 1, 2, 3, 4\}$ approprié transmis au *Tunable Complexity CTU Encoding* système pour chaque CTU.

Le répartiteur de complexité CDC divise le nombre décimal $\hat{\Xi}$ en deux parties : la partie entière et la partie décimale. La partie entière de $\hat{\Xi}$ définit le nombre minimum de niveaux de profondeur $\lfloor \hat{\Xi} \rfloor$ testé par tous les CTU de l'image d'entrée. La partie décimale de $\hat{\Xi}$ définit le nombre de CTU qui testent un niveau de profondeur supplémentaire $\lceil \hat{\Xi} \rceil$. Le répartiteur de complexité CDC utilise les RD-cost de l'image précédente stockés dans

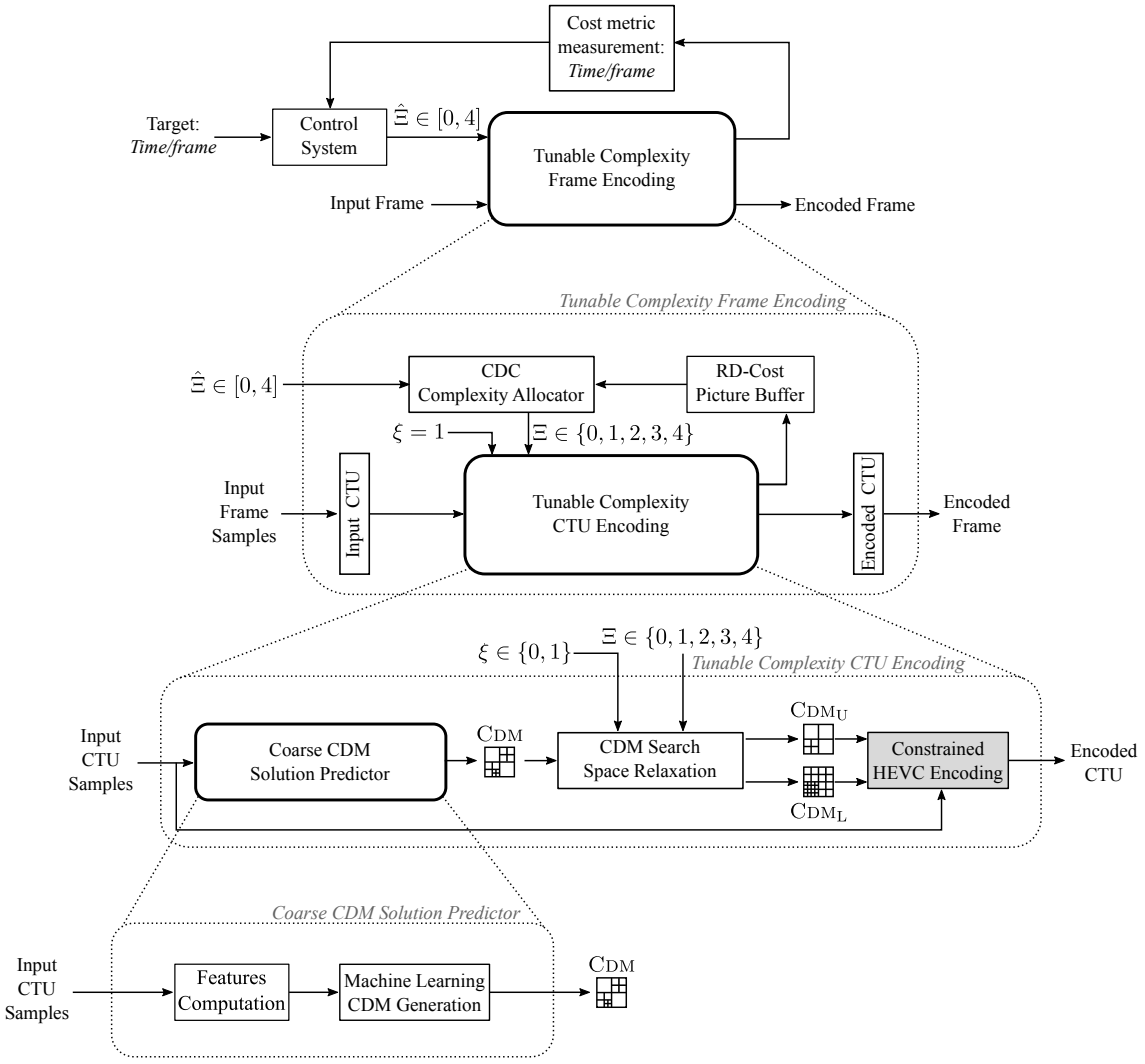


Figure A.14 – *Vue d'ensemble du système de contrôle du temps d'encodage image par image d'un encodeur* Hevc.

un tampon pour sélectionner quel CTU test $\lfloor\hat{\Xi}\rfloor$ ou $\lceil\hat{\Xi}\rceil$ niveaux de profondeur selon le protocole défini dans Section A.4.2.

Le *Tunable Complexity CTU Encoding* système reçoit le nombre entier de niveaux de profondeur Ξ et les pixels du CTU courant à encoder. *Coarse* Cdm *Solution Predictor*, décrit à la Section A.3.2, est utilisé pour prédire la découpe du CTU courant en générant la Cdm associée. L'algorithme Cdm *Search Space Relaxation*, détaillé dans Section A.4.1, reçoit la Cdm générée et le nombre de niveaux de profondeur Ξ pour calculer l'intervalle de niveaux de profondeur formé par Cdm_U et Cdm_L. Enfin, l'encodeur Hevc est forcé d'appliquer uniquement le processus de RDO entre l'intervalle défini par Cdm_L et Cdm_U.

A.5.2 Résultats Expérimentaux

Pour comparer et évaluer la performance des outils d'encodage, les métriques les plus utilisées sont appelées les métrique de Bjøntegaard [Bjo01]. Basés sur le Peak Signal-to-Noise Ratio (Psnr), les métriques de Bjøntegaard mesurent la différence moyenne entre deux courbes RD interpolées par quatre points. Le Bjøntegaard Delta Bit Rate (BD-BR) in-

Table A.1 – *Temps d'encodage obtenue, erreur, BD-BR et BD-Psnr moyen en fonction du temps d'encodage ciblé.*

Temps d'encodage ciblé (%)	Temps d'encodage obtenu (%)	Erreur (%)	BD-BR (%)	BD-PSNR (dB)
90	90.87	0.87	0.06	-0.00
80	81.24	1.24	0.10	-0.00
70	71.22	1.22	0.22	-0.01
60	61.20	1.20	0.70	-0.03
50	51.40	1.40	1.78	-0.09
40	43.28	3.28	3.20	-0.15

dique la différence moyenne de débit binaire en pourcentage pour deux encodages de même qualité : qualité mesuré par le Psnr. De même, le Bjøntegaard Delta Psnr (BD-Psnr) mesure la différence moyenne de Psnr en décibels (dB) pour deux encodages différents produisant le même débit binaire.

La Table A.1 présente les résultats en termes de BD-BR et BD-Psnr en fonction du temps d'encodage normalisé pour des cibles de temps d'encodage comprise entre 100% et 40% par pas de 10%. Ces résultats sont la moyenne des résultats obtenus sur 18 séquences vidéo.

Les résultats de la Table A.1 montrent que notre système de contrôle du temps d'encodage d'encodeur Hevc Intra permet de contrôler le temps d'encodage avec une erreur variant de 0,87% à 3,20% (avec une moyenne de 1,53%) selon le temps d'encodage ciblé pour une dégradation comprise entre 0,06% à 3,20% de BD-BR.

A.6 Conclusion

Ce document propose un ensemble d'études et de méthodes dont l'objectif final est d'ajuster et de contrôler la complexité et donc la consommation énergétique du processus d'encodage Hevc. Deux méthodes de prédiction de découpe CTU sont proposées : la première basée sur une approche *Statistique* de la variance de l'image et la seconde utilisant l'*Intelligence Artificielle* à travers des classificateurs.

À partir de cette prédiction, une solution générique est présentée afin d' ajuster dynamiquement la complexité d'encodage Hevc d'une image. Cette solution étend l'espace de recherche autour de la prédiction de la découpe de CTU en générant un intervalle de niveaux de profondeur dans lequel l'encodageHevc est limité. Couplé un une méthode d'allocation de complexité dans l'image, la solution est capable d'ajuster la complexité du processus d'encodage tout en minimisant les dégradations en termes de compression et qualité.

Enfin, un système de contrôle temps réel ajustant dynamiquement le processus d'encodage pour maintenir sa complexité sous un certain seuil est proposé. Ce système rassemble les principales contributions présentées dans ce document et permet de démontrer l'utilité et l'applicabilité l'ensemble des travaux présentés dans ce document.

APPENDIX B

Video Sequences

This appendix presents the video sequences used in the experiments of this document. A set of 18 sequences defined by Joint Collaborative Team on Video Coding (JCT-VC) in the Common Test Conditions (CTC) [Bos13] is selected for the experiments, as summarized in Table B.1.

Table B.1 – *Test sequences*

Class	Sequence name	Resolution	Nb frame	Frame rate	Source
A	Traffic	2560×1600	150	30	CTC
A	PeopleOnStreet	2560×1600	150	30	CTC
B	Kimono	1920×1080	240	24	CTC
B	ParkScene	1920×1080	240	24	CTC
B	Cactus	1920×1080	500	50	CTC
B	BQTerrace	1920×1080	600	60	CTC
B	BasketballDrive	1920×1080	500	50	CTC
C	RaceHorses	832×480	300	30	CTC
C	BQMall	832×480	600	60	CTC
C	PartyScene	832×480	500	50	CTC
C	BasketballDrill	832×480	500	50	CTC
D	RaceHorses	416×240	300	30	CTC
D	BQSquare	416×240	600	60	CTC
D	BlowingBubbles	416×240	500	50	CTC
D	BasketballPass	416×240	500	50	CTC
E	FourPeople	1280×720	600	60	CTC
E	Johnny	1280×720	600	60	CTC
E	KristenAndSara	1280×720	600	60	CTC

Two sequences from class A, 5 sequences from class B, 4 sequences from class C, 4 sequences from class D and 3 sequences from class E. Each class corresponds to a specific resolution. The sequences of class E are considered to represent conversational applications.

This set of sequences is composed by video sequences that strongly differ from one another in terms of frame rate, motion, texture and spatial resolution.

To illustrate the video sequence content properties, Figures B.1, B.2, B.3, B.4, B.5, B.6, B.7, B.8, B.9, B.10, B.11, B.12, B.13, B.14, B.15, B.16, B.17, B.18 and B.19 show one frame of each sequence whose are plotted with the same size regardless their original resolutions. Figures B.3 and B.4 show the frames before and after the cut scene of Kimono video sequence, respectively.

Figure B.1 – *100^{th} frame of Traffic video sequence (*2560×1600*).*

Figure B.2 – *100^{th} frame of PeopleOnStreet video sequence (*2560×1600*).*

Figure B.3 – *139th frame of Kimono video sequence (*1920 × 1080*), just before the cut scene.*

Figure B.4 – *140th frame of Kimono video sequence (*1920 × 1080*), just after the cut scene.*

Figure B.5 – *100th frame of ParkScene video sequence (*1920 × 1080*).*

Figure B.6 – *100^{th} frame of Cactus video sequence (*1920×1080*).*

Figure B.7 – *100^{th} frame of BQTerrace video sequence (*1920×1080*).*

Figure B.8 – *150^{th} frame of BasketballDrive video sequence (*1920×1080*).*

Figure B.9 – *100^{th} frame of RaceHorses video sequence (*832×480*).*

Figure B.10 – *100^{th} frame of BQMall video sequence (*832×480*).*

Figure B.11 – *100^{th} frame of PartyScene video sequence (*832×480*).*

Figure B.12 – *100^{th} frame of BasketballDrill video sequence* (832×480).

Figure B.13 – *100^{th} frame of RaceHorses video sequence* (416×240).

Figure B.14 – *100^{th} frame of BQSquare video sequence* (416×240).

Figure B.15 – *100^{th} frame of BlowingBubbles video sequence (416×240).*

Figure B.16 – *100^{th} frame of BasketballPass video sequence (416×240).*

Figure B.17 – *100^{th} frame of FourPeople video sequence (1280×720).*

Figure B.18 – *100^{th} frame of Johnny video sequence* (1280×720).

Figure B.19 – *100^{th} frame of KristenAndSara video sequence* (1280×720).

APPENDIX C

Encoding Time Control System of HEVC Intra Encoder Results

Chapter 8 has presented the *Encoding Time Control System* which uses PI-based feedback control to reduce and control the encoding time in second by frame under a specific target. The control system command the *Tunable Complexity Frame Encoding* scheme detailed in Chapter 7 to scale the complexity of the encoding process by adjusting the quad-tree partitioning process.

This appendix presents the detailed performance of the *Encoding Time Control System* of HEVC Intra encoder summarized by Figure 8.8 in Chapter 8. The following table gives the experimental results obtained for the proposed encoding time control system on a real time HEVC encoder. The experimental set-up has been detailed in Section 4.3.1. The evaluation of the performance is realized on the 18 video sequences presented in Table B.1 for targets ranging from 90% to 40%. In the experiments presented in this appendix, the target times in second are based on the reference full encoding time of the video sequences according to the QP value. For each sequence and QP value, the full encoding time is divided by the number of frames of the video sequences and the targets are computed by weighting the obtained averaged time per frame by the target coefficient. All results of the following sections include the time overhead due to the entire encoding time control system scheme. Table C.1 details the results by sequence according to the target times.

Table C.1 – *Real Normalized Encoding Time (NET), NET Error, BD-BR and BD-PSNR averaged by sequence according to the target NET.*

Target NET (%)		Real NET(%)	NET Error (%)	BD-BR (%)	BD-PSNR (dB)
	90	90.95	0.95	0.05	-0.00
Traffic	80	80.99	0.99	0.06	-0.00
Class A	70	70.93	0.93	0.10	-0.01
2560x1600	60	61.11	1.11	0.60	-0.03
150 frames	50	51.22	1.22	1.83	-0.10
	40	42.03	2.03	3.73	-0.20
	90	90.90	0.90	0.02	-0.00
PeopleOnStreet	80	81.00	1.00	0.02	-0.00
Class A	70	70.93	0.93	0.03	-0.00
2560x1600	60	61.08	1.08	0.35	-0.02
150 frames	50	51.20	1.20	1.62	-0.09
	40	42.78	2.78	3.45	-0.20
	90	91.34	1.34	0.59	-0.02
Kimono	80	81.40	1.40	1.04	-0.04
Class B	70	71.46	1.46	1.31	-0.05
1920x1080	60	61.53	1.53	2.47	-0.08
240 frames	50	51.62	1.62	4.47	-0.15
	40	42.17	2.17	8.42	-0.28
	90	91.04	1.04	0.04	-0.00
ParkScene	80	81.10	1.10	0.08	-0.00
Class B	70	71.10	1.10	0.15	-0.01
1920x1080	60	61.16	1.16	0.69	-0.03
240 frames	50	51.23	1.23	1.49	-0.07
	40	41.73	1.73	3.38	-0.15
	90	90.68	0.68	0.02	-0.00
BasketballDrive	80	81.53	1.53	0.02	-0.00
Class B	70	71.61	1.61	0.19	-0.01
1920x1080	60	61.61	1.61	0.67	-0.02
500 frames	50	51.64	1.64	1.85	-0.06
	40	43.04	3.04	3.92	-0.11
	90	91.22	1.22	0.03	-0.00
Cactus	80	81.22	1.22	0.07	-0.00
Class B	70	71.25	1.25	0.16	-0.01
1920x1080	60	61.26	1.26	0.60	-0.02
500 frames	50	51.30	1.30	1.42	-0.05
	40	41.86	1.86	3.16	-0.11
	90	89.69	-0.31	0.01	-0.00
BQTerrace	80	81.09	1.09	0.01	-0.00
Class B	70	71.25	1.25	0.07	-0.00
1920x1080	60	61.26	1.26	0.21	-0.02
600 frames	50	51.28	1.28	0.79	-0.06
	40	43.40	3.40	1.60	-0.11

Table C.1 – *(continued)*

Target NET (%)		Real NET(%)	NET Error (%)	BD-BR (%)	BD-PSNR (dB)
	90	89.29	-0.71	0.04	-0.00
RaceHorses	80	80.80	0.80	0.10	-0.01
Class C	70	70.93	0.93	0.22	-0.01
832x480	60	60.95	0.95	0.70	-0.04
300 frames	50	51.25	1.25	1.48	-0.09
	40	42.69	2.69	2.36	-0.13
	90	90.87	0.87	0.00	-0.00
PartyScene	80	80.92	0.92	0.00	-0.00
Class C	70	70.94	0.94	0.02	-0.00
832x480	60	60.90	0.90	0.14	-0.01
500 frames	50	50.91	0.91	0.45	-0.03
	40	42.92	2.92	0.99	-0.07
	90	91.26	1.26	0.01	-0.00
BasketballDrill	80	81.28	1.28	0.05	-0.00
Class C	70	71.27	1.27	0.19	-0.01
832x480	60	61.23	1.23	1.16	-0.06
500 frames	50	51.27	1.27	2.08	-0.10
	40	41.50	1.50	3.89	-0.19
	90	90.60	0.60	0.02	-0.00
BQMall	80	81.27	1.27	0.03	-0.00
Class C	70	71.31	1.31	0.13	-0.01
832x480	60	61.29	1.29	0.59	-0.03
600 frames	50	51.32	1.32	1.50	-0.08
	40	42.97	2.97	2.81	-0.15
	90	90.71	0.71	0.01	-0.00
RaceHorses	80	81.00	1.00	0.01	-0.00
Class D	70	70.72	0.72	0.05	-0.00
416x240	60	60.52	0.52	0.28	-0.02
300 frames	50	51.10	1.10	0.84	-0.05
	40	42.45	2.45	1.57	-0.10
	90	90.81	0.81	0.00	0.00
BQSquare	80	81.12	1.12	0.00	-0.00
Class D	70	71.02	1.02	0.02	-0.00
416x240	60	60.74	0.74	0.13	-0.01
600 frames	50	51.34	1.34	0.43	-0.03
	40	44.60	4.60	0.89	-0.07
	90	90.55	0.55	0.00	-0.00
BlowingBubbles	80	81.15	1.15	0.01	-0.00
Class D	70	70.99	0.99	0.14	-0.01
416x240	60	60.93	0.93	0.36	-0.02
500 frames	50	51.59	1.59	0.77	-0.05
	40	46.16	6.16	1.09	-0.07

Table C.1 – *(continued)*

Target NET (%)		Real NET(%)	NET Error (%)	BD-BR (%)	BD-PSNR (dB)
	90	90.44	0.44	0.01	-0.00
BasketballPass	80	81.18	1.18	0.01	-0.00
Class D	70	71.07	1.07	0.12	-0.01
416x240	60	61.21	1.21	0.34	-0.02
500 frames	50	51.38	1.38	1.00	-0.06
	40	43.75	3.75	1.81	-0.11
	90	91.52	1.52	0.00	-0.00
FourPeople	80	81.54	1.54	0.02	-0.00
Class E	70	71.46	1.46	0.05	-0.00
1280x720	60	61.45	1.45	0.51	-0.03
600 frames	50	51.59	1.59	2.49	-0.14
	40	43.89	3.89	4.42	-0.25
	90	91.92	1.92	0.19	-0.01
Johnny	80	81.96	1.96	0.20	-0.01
Class E	70	72.01	2.01	0.72	-0.03
1280x720	60	61.76	1.76	1.70	-0.07
600 frames	50	52.03	2.03	4.35	-0.18
	40	46.10	6.10	5.46	-0.22
	90	91.81	1.81	0.01	-0.00
KristenAndSara	80	81.84	1.84	0.01	-0.00
Class E	70	71.80	1.80	0.26	-0.01
1280x720	60	61.61	1.61	1.03	-0.05
600 frames	50	51.89	1.89	3.16	-0.16
	40	44.92	4.92	4.56	-0.23

List of Figures

List of Tables

Personal Publications

[MAH+17a] Alexandre Mercat*, Florian Arrestier, Wassim Hamidouche, Maxime Pelcat, and Daniel Menard. Constrain the docile ctus: an in-frame complexity allocator for hevc intra encoders. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 1163–1167. IEEE, 2017.

[MAH+17b] Alexandre Mercat*, Florian Arrestier, Wassim Hamidouche, Maxime Pelcat, and Daniel Menard. Energy reduction opportunities in an hevc real-time encoder. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 1158–1162. IEEE, 2017.

[MAP+17] Alexandre Mercat*, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Prediction of quad-tree partitioning for budgeted energy hevc encoding. In *Signal Processing Systems (SiPS), 2017 IEEE International Workshop on*, pages 1–6. IEEE, 2017.

[MAP+18a] Alexandre Mercat*, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Machine learning based choice of characteristics for the one-shot determination of the hevc intra coding tree. In *2018 Picture Coding Symposium (PCS)*, pages 263–267. IEEE, 2018.

[MAP+18b] Alexandre Mercat*, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. On predicting the hevc intra quad-tree partitioning with tunable energy and rate-distortion. *Journal of Real-Time Image Processing*, pages 1–14, 2018.

[MAP+18c] Alexandre Mercat*, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Probabilistic approach versus machine learning for one-shot quad-tree prediction in an intra hevc encoder. Accepted for publication in *Journal of Signal Processing Systems (JSPS)*, 2018.

[MBP+17a] Alexandre Mercat*, Justine Bonnot, Maxime Pelcat, Karol Desnos, Wassim Hamidouche, and Daniel Menard. Smart search space reduction for approximate computing: A low energy hevc encoder case study. *Journal of Systems Architecture*, 80:56–67, 2017.

*Publications related to the research work described in this document.

[MBP+17b] Alexandre Mercat*, Justine Bonnot, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Exploiting computation skip to reduce energy consumption by approximate computing, an hevc encoder case study. In *Proceedings of the Conference on Design, Automation & Test in Europe*, pages 494–499. European Design and Automation Association, 2017.

[MHPM16] Alexandre Mercat*, Wassim Hamidouche, Maxime Pelcat, and Daniel Menard. Estimating encoding complexity of a real-time embedded software hevc codec. In *Design and Architectures for Signal and Image Processing (DASIP), 2016 Conference on*, pages 26–33. IEEE, 2016.

[MNMZ14] Alexandre Mercat, Jean-François Nezan, Daniel Menard, and Jinglin Zhang. Implementation of a stereo matching algorithm onto a manycore embedded system. In *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*, pages 1296–1299. IEEE, 2014.

[NMDG16] Jean-François Nezan, Alexandre Mercat, Patrice Delmas, and Georgy Gimelfarb. Optimized belief propagation algorithm onto embedded multi and manycore systems for stereo matching. In *Parallel, Distributed, and Network-Based Processing (PDP), 2016 24th Euromicro International Conference on*, pages 332–336. IEEE, 2016.

[NMMP17] Erwan Nogues, Daniel Ménard, Alexandre Mercat, and Maxime Pelcat. On learning the energy model of an mpsoc for convex optimization. In *Proceedings of the Computing Frontiers Conference*, pages 385–390. ACM, 2017.

[NPMM16] Erwan Nogues, Maxime Pelcat, Daniel Menard, and Alexandre Mercat. Energy efficient scheduling of real time signal processing applications through combined dvfs and dpm. In *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pages 622–626. IEEE, 2016.

[PMD+17] Maxime Pelcat, Alexandre Mercat, Karol Desnos, Luca Maggiani, Yanzhou Liu, Julien Heulot, Jean-François Nezan, Wassim Hamidouche, Daniel Ménard, and Shuvra S Bhattacharyya. Reproducible evaluation of system efficiency with a model of architecture: From theory to practice. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017.

Bibliography

[ABIAFC+17] Achraf Ait-Beni-Ifit, Othmane Alaoui-Fdili, Patrick Corlay, François-Xavier Coudoux, and Driss Aboutajdine. Profiling and Modelling of HEVC Intra Video Encoder's Energy Consumption for Next Generation WVSNS. In Amr El Abbadi and Benoît Garbinato, editors, *Networked Systems*, volume 10299, pages 472–482. Springer International Publishing, Cham, 2017. 156

[ACZ+15] J An, YW Chen, K Zhang, H Huang, YW Huang, and S Lei. Block partitioning structure for next generation video coding. In *https://www.itu.int/md/T13-SG16-C-0966/en*, 2015. 157

[AKP13] Sangsoo Ahn, Munchurl Kim, and Seongmo Park. Fast decision of CU partitioning based on SAO parameter, motion and PU/TU split information for HEVC. In *Picture Coding Symposium (PCS), 2013*, pages 113–116. IEEE, 2013. 31, 34

[ALK15] Sangsoo Ahn, Bumshik Lee, and Munchurl Kim. A Novel Fast CU Encoding Scheme Based on Spatiotemporal Encoding Parameters for HEVC Inter Coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(3):422–435, March 2015. 31, 34

[Ast95] Karl Johan Astrom. *PID controllers: theory, design, and tuning*. 1995. 134

[BC15] Biao Min and Ray C. C. Cheung. A Fast CU Size Decision Algorithm for the HEVC Intra Encoder. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(5):892–896, May 2015. 33, 35, 63, 64, 79, 80, 86, 87, 107, 164

[BDFM16] Gisle Bjontegaard, Thomas Davies, Arild Fuldseth, and Steinar Midtskogen. The Thor Video Codec. In *2016 Data Compression Conference (DCC)*, pages 476–485, Snowbird, UT, USA, March 2016. IEEE. 12

[BDM98] Luca Benini and Giovanni De Micheli. *Dynamic Power Management*. Springer US, Boston, MA, 1998. 48

[Bjo01] G. Bjontegaard. Calculation of average PSNR differences between RD-Curves. Austin, Texas, 2001. 15, 171

[Bos13] F. Bossen. Common HM test conditions and software reference configurations. In *JCTVC-L1100*, Geneva, Switzerland, 2013. 56, 77, 116, 173

[BPM+15] S. G. Blasi, E. Peixoto, B. Macchiavello, E. M. Hung, I. Zupancic, and E. Izquierdo. Context adaptive mode sorting for fast HEVC mode decision. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 1478–1482. IEEE, 2015. 35, 36

[Bro05] Len Brown. Acpi in linux. *Proceedings of the Linux Symposium*, 51, 2005. 57

[BWX11] Jim Bankoski, Paul Wilkins, and Yaowu Xu. Technical overview of VP8, an open source video codec for the web. In *2011 IEEE International Conference on Multimedia and Expo*, pages 1–6, Barcelona, Spain, July 2011. IEEE. 12

[BZIP15] Saverio G. Blasi, Ivan Zupancic, Ebroul Izquierdo, and Eduardo Peixoto. Fast HEVC coding using reverse CU visiting. In *Picture Coding Symposium (PCS), 2015*, pages 50–54. IEEE, 2015. 36

[CAA+12] Guilherme Correa, Pedro Assuncao, Luciano Agostini, Luis da Silva Cruz, and others. Motion compensated tree depth limitation for complexity control of HEVC encoding. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 217–220. IEEE, 2012. 43, 44

[CAA+13] Guilherme Correa, Pedro Assuncao, Luciano Agostini, Luis da Silva Cruz, and others. Complexity control of HEVC through quadtree depth estimation. In *EUROCON, 2013 IEEE*, pages 81–86. IEEE, 2013. 43, 44, 156

[CAAC11] Guilherme Corrêa, Pedro Assuncao, Luciano Agostini, and Luis A. Cruz. Complexity control of high efficiency video encoders for power-constrained devices. *Consumer Electronics, IEEE Transactions on*, 57(4):1866–1874, 2011. 43, 44

[CAAC13] G. Correa, P. Assuncao, L. Agostini, and L. A. D. S. Cruz. Coding Tree Depth Estimation for Complexity Reduction of HEVC. In *DCC*, pages 43–52. IEEE, March 2013. 43, 44

[CAAC16] Guilherme Correa, Pedro Assuncao, Luciano Agostini, and Luis A. da Silva Cruz. *Complexity-Aware High Efficiency Video Coding*. Springer International Publishing, Cham, 2016. 43, 114

[CAAdSC15] Guilherme Correa, Pedro Assuncao, Luciano Agostini, and Luis da Silva Cruz. Pareto-Based Method for High Efficiency Video Coding with Limited Encoding Time. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1, 2015. 43, 44, 150, 192

[CAAdSC16] Guilherme Correa, Pedro Assuncao, Luciano Agostini, and Luis A. da Silva Cruz. Complexity scalability for real-time HEVC encoders. *Journal of Real-Time Image Processing*, 12(1):107–122, June 2016. 43, 44, 114, 150, 192

[CAdSC+12] Guilherme Correa, Pedro Assuncao, Luis da Silva Cruz, Luciano Agostini, and others. Dynamic tree-depth adjustment for low power HEVC encoders. In *Electronics, Circuits and Systems (ICECS), 2012 19th IEEE International Conference on*, pages 564–567. IEEE, 2012. 43, 44

[CAdSC+13a] Guilherme Correa, Pedro Assuncao, Luis da Silva Cruz, Luciano Agostini, and others. Computational complexity control for HEVC based on coding tree spatio-temporal correlation. In *Electronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on*, pages 937–940. IEEE, 2013. 43, 44

[CAdSC+13b] Guilherme Correa, Pedro Assuncao, Luis da Silva Cruz, Luciano Agostini, and others. Constrained encoding structures for computational complexity scalability in HEVC. In *Picture Coding Symposium (PCS), 2013*, pages 297–300. IEEE, 2013. 43, 44, 114, 150, 192

[CAdSC+14] Guilherme Correa, Pedro Assuncao, Luis da Silva Cruz, Luciano Agostini, and others. Classification-based early termination for coding tree structure decision in HEVC. In *Electronics, Circuits and Systems (ICECS), 2014 21st IEEE International Conference on*, pages 239–242. IEEE, 2014. 32, 34, 43

[CAdSCA15] Guilherme Correa, Pedro Assuncao, Luis A. da Silva Cruz, and Luciano Agostini. Encoding time control system for HEVC based on Rate-Distortion-Complexity analysis. In *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, pages 1114–1117. IEEE, 2015. 43, 44, 150, 192

[CAVAdSC15] Guilherme Correa, Pedro A. Assuncao, Luciano Volcan Agostini, and Luis A. da Silva Cruz. Fast HEVC Encoding Decisions Using Data Mining. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(4):660–673, April 2015. 32, 43

[CC13] Pai-Tse Chiang and Tian Sheuan Chang. Fast zero block detection and early CU termination for HEVC video coding. In *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, pages 1640–1643. IEEE, 2013. 38, 39

[CCI88] CCITT. Recommendation H.120 : Codecs for Videoconferencing Using Primary Digital Group Transmission. 1988. 9

[CGL82] T. F. Chan, G. H. Golub, and R. J. LeVeque. Updating Formulae and a Pairwise Algorithm for Variances Computing Sample. In *COMPSTAT 1982 5th Symposium held at Toulouse 1982*, page 30. Springer Science & Business Media, 1982. 70

[CH10] Aaron Carroll and Gernot Heiser. An analysis of power consumption in a smartphone. In *USENIX annual technical conference*, volume 14, pages 21–21, Boston, MA, 2010. 4, 160

[CIS16] CISCO. Global_2021_forecast_highlights. In *https://www.cisco.com/c/m/en_us/solutions/service-provider/vni-forecast-highlights.html*, page 6, 2016. 44

[CIS17] CISCO. The Zettabyte Era: Trends and Analysis. In *https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html*, page 32, 2017. 3, 4

[CJ12] Kiho Choi and Euee S. Jang. Early TU decision method for fast video encoding in high efficiency video coding. *Electronics Letters*, 48(12):689, 2012. 38, 39

[CK13] Seunghyun Cho and Munchurl Kim. Fast CU Splitting and Pruning for Suboptimal CU Partitioning in HEVC Intra Coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(9):1555–1564, September 2013. 31, 34

[CMR+14] Vinay Kumar Chippa, Debabrata Mohapatra, Kaushik Roy, Srimat T. Chakradhar, and Anand Raghunathan. Scalable Effort Hardware Design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(9):2004–2016, September 2014. 49

[CNP12] Michele Belotti Cassa, Matteo Naccari, and Fernando Pereira. Fast rate distortion optimization for the emerging HEVC standard. In *Picture Coding Symposium (PCS), 2012*, pages 493–496. IEEE, 2012. 31, 34, 63, 87

[CPS+13] Gaoxing Chen, Zhenyu Pei, Lei Sun, Zhenyu Liu, and Takeshi Ikenaga. Fast intra prediction for HEVC based on pixel gradient statistics and mode refinement. In *Signal and Information Processing (ChinaSIP), 2013 IEEE China Summit & International Conference on*, pages 514–517. IEEE, 2013. 36, 38

[CZHD18] Fang-Yi Chao, Lu Zhang, Wassim Hamidouche, and Olivier Deforges. SALGAN360: VISUAL SALIENCY PREDICTION ON 360 DEGREE IMAGES WITH GENERATIVE ADVERSARIAL NETWORKS. page 4, 2018. 156

[DMW15] Fanyi Duanmu, Zhan Ma, and Yao Wang. Fast CU partition decision using machine learning for screen content compression. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 4972–4976. IEEE, 2015. 35, 79, 86, 87, 107

[Dom12] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, October 2012. 85

[dSAC12] Thaísa L. da Silva, Luciano V. Agostini, and Luis A. Cruz. Fast HEVC intra prediction mode decision based on EDGE direction information. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1214–1218. IEEE, 2012. 36, 38

[DXJ+16] Xin Deng, Mai Xu, Lai Jiang, Xiaoyan Sun, and Zulin Wang. Subjective-Driven Complexity Control Approach for HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):91–106, January 2016. 42, 44, 150, 192

[EAD+12] Rotem Efraim, Naveh Alon, Rajwan Doron, Ananthakrishnan Avinash, and Weissmann Eliezer. Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge. *IEEE Computer Society*, 32(2):20–27, April 2012. 57

[FDBB+18] D.G. Fernández, A.A. Del Barrio, G. Botella, C. García, M. Prieto, and R. Hermida. Complexity reduction in the HEVC/H265 standard based on smooth region classification. *Digital Signal Processing*, 73:24–39, February 2018. 32, 34, 37, 38

[FDBBG18] D. G. Fernández, A. A. Del Barrio, G. Botella, and C. García. Fast and effective CU size decision based on spatial and temporal homogeneity detection. *Multimedia Tools and Applications*, 77(5):5907–5927, March 2018. 32, 34

[FDZX16] Lei Feng, Ming Dai, Chun-lei Zhao, and Jing-ying Xiong. Fast prediction unit selection method for HEVC intra prediction based on salient regions. *Optoelectronics Letters*, 12(4):316–320, July 2016. 33, 34, 35, 63, 79, 107

[Fou01] Xiph.Org Foundation. Theora Specification. In *https://www.theora.org/doc/Theora.pdf*, 2001. 12

[FWRR11] Christian Feller, Juergen Wuenschmann, Thorsten Roll, and Albrecht Rothermel. The VP8 video codec - overview and comparison to H.264/AVC. In *2011 IEEE International Conference on Consumer Electronics -Berlin (ICCE-Berlin)*, pages 57–61, Berlin, Germany, September 2011. IEEE. 12

[GKJ+14] Kalyan Goswami, Byung-Gyu Kim, Dongsan Jun, Soon-Heung Jung, and Jin Soo Choi. Early Coding Unit-Splitting Termination Algorithm for High Efficiency Video Coding (HEVC). *ETRI Journal*, 36(3):407–417, June 2014. 31, 34

[GL] RyeongHee Gweon and Yung-Lyul Lee. N-Level Quantization in HEVC. page 5. 25

[GSK+13] Mateus Grellert, Muhammad Shafique, Karim Khan, Muhammad Usman, Luciano Agostini, Julio CB Mattos, and Jörg Henkel. An adaptive workload management scheme for HEVC encoding. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 1850–1854. IEEE, 2013. 42, 44, 150, 192

[GZS+17] Mateus Grellert, Bruno Zatt, Muhammad Shafique, Sergio Bampi, and Jörg Henkel. Complexity control of HEVC encoders targeting real-time constraints. *Journal of Real-Time Image Processing*, 13(1):5–24, March 2017. 42, 44

[HFH+09] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009. 85

[HOPP13] Wassim Hamidouche, Christian Olivier, Yannis Pousset, and Clency Perrine. Optimal resource allocation for Medium Grain Scalable video transmission over MIMO channels. *Journal of Visual Communication and Image Representation*, 24(3):373–387, April 2013. 50, 51

[HRGs+16] Christian Herglotz, Rafael Rosales, Michael Gla\s s, Jürgen Teich, and André Kaup. Multi-objective design space exploration for the optimization of the HEVC mode decision process. In *Picture Coding Symposium (PCS), 2016*, pages 1–5. IEEE, 2016. 36

[HSI+15] Daniel Hackenberg, Robert Schone, Thomas Ilsche, Daniel Molka, Joseph Schuchart, and Robin Geyer. An Energy Efficiency Feature Survey of the Intel Haswell Processor. In *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*, pages 896–904. IEEE, May 2015. 57

[HZ14] Hao Zhang and Zhan Ma. Fast Intra Mode Decision for High Efficiency Video Coding (HEVC). *IEEE Transactions on Circuits and Systems for Video Technology*, 24(4):660–668, April 2014. 31, 37, 38

[HZLB13] Han Huang, Yao Zhao, Chunyu Lin, and Huihui Bai. Fast bottom-up pruning for HEVC intraframe coding. In *Visual Communications and Image Processing (VCIP), 2013*, pages 1–5. IEEE, 2013. 31, 34

[ISO93] ISO/IEC. International Standard 11172-2: Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5mbit/s – Part 2: Video. 1993. 11

[ISO95] ISO/IEC. International Standard 14496-2: Coding of Audio-Visual Objects – Part 2: Visual. 1995. 11

[ISO99] ISO/IEC. International Standard 13818-2: Generic Coding of Moving Pictures and Associated Audio Information – Part2: Video. 1999. 11

[ISO03] ISO/IEC. International Standard 14496-10: Advanced Video Coding. 2003. 12

[ISO13] ISO/IEC. International Standard 23008-2: High Efficiency Video Coding. 2013. 12

[IT93] ITU-T. Recommendation H.261: Video Codec for Audiovisual Services at p×64 kbits. 1993. 10, 12

[IT95] ITU-T. Recommendation H.263: Video Coding for Low Bit Rate Communication. 1995. 11

[IT99] ITU-T. Recommendation H.262: Transmission of Non-Telephone Signals. 1999. 11

[IT03] ITU-T. Recommendation H.264: Advanced Video Coding. 2003. 12

[IT13] ITU-T. Recommendation H.265: High Efficiency Video Coding. 2013. 12

[ITU99] ITU. Recommandation ITU-T P.910. Subjective video quality assessment methods for multimedia applications. Geneva, 1999. 85

[JASY17] Zhipeng Jin, Ping An, Liquan Shen, and Chao Yang. CNN oriented fast QTBT partition algorithm for JVET intra coding. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4, St. Petersburg, FL, December 2017. IEEE. 157

[JMC12] Wei Jiang, Hanjie Ma, and Yaowu Chen. Gradient based fast mode decision algorithm for intra prediction in HEVC. In *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, pages 1836–1840. IEEE, 2012. 36, 38

[JMMEDdM16] Amaya Jimenez-Moreno, Eduardo Martinez-Enriquez, and Fernando Diaz-de Maria. Complexity Control Based on a Fast Coding Unit Decision Method in the HEVC Video Coding Standard. *IEEE Transactions on Multimedia*, 18(4):563–575, April 2016. 43, 44, 150, 192

[JPG04] Ravindra Jejurikar, Cristiano Pereira, and Rajesh Gupta. Leakage aware dynamic voltage scaling for real-time embedded systems. page 275. ACM Press, 2004. 48

[JV16] JCT-VC. HEVC reference software. https://hevc.hhi.fraunhofer.de/, 2016. 21, 26

[KJSS15] Kyungmin Lim, Jaeho Lee, Seongwan Kim, and Sangyoun Lee. Fast PU Skip and Split Termination Algorithm for HEVC Intra Prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(8):1335–1346, August 2015. 31, 34

[KKS+18] Takafumi Katayama, Kazuki Kuroda, Wen Shi, Tian Song, and Takashi Shimamoto. Low-complexity intra coding algorithm based on convolutional neural network for HEVC. In *2018 International Conference on Information and Computer Technologies (ICICT)*, pages 115–118, DeKalb, IL, March 2018. IEEE. 155

[KL51] Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951. 86

[KL07] Hari Kalva and Jae-Beom Lee. The VC-1 Video Coding Standard. *IEEE Multimedia*, 14(4):88–91, October 2007. 12

[KP11] Philipp Klaus Krause and Ilia Polian. Adaptive voltage over-scaling for resilient applications. In *2011 Design, Automation & Test in Europe*, pages 1–6. IEEE, 2011. 49

[KP16] Hyo-Song Kim and Rae-Hong Park. Fast CU Partitioning Algorithm for HEVC Using an Online-Learning-Based Bayesian Decision Rule. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):130–138, January 2016. 32, 34

[KSH13] Muhammad Usman Karim Khan, Muhammad Shafique, and Jörg Henkel. An adaptive complexity reduction scheme with fast prediction unit decision for HEVC intra encoding. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 1578–1582. IEEE, 2013. 33, 35, 63, 64, 79, 86, 107, 164

[KVL+15] Ari Koivula, Marko Viitanen, Ari Lemmetti, Jarno Vanne, and Timo D. Hämäläinen. Performance evaluation of Kvazaar HEVC intra encoder on Xeon Phi many-core processor. In *Signal and Information Processing (GlobalSIP), 2015 IEEE Global Conference on*, pages 1250–1254. IEEE, 2015. 56

[KVV+15] Ari Koivula, Marko Viitanen, Jarno Vanne, Timo D. Hamalainen, and Laurent Fasnacht. Parallelization of Kvazaar HEVC intra encoder for multi-core processors. In *Signal Processing Systems (SiPS), 2015 IEEE Workshop on*, pages 1–6. IEEE, 2015. 56

[KYC08] Marta Karczewicz, Yan Ye, and Insuk Chong. Rate distortion optimized quantization. In *VCEG-AH21*, Antalya Turkey, 2008. 25

[LdSdSCAo13] Thaisa Leal da Silva, Luis da Silva Cruz, Luciano V. Agostini, and others. HEVC intra mode decision acceleration based on tree depth levels relationship. In *Picture Coding Symposium (PCS), 2013*, pages 277–280. IEEE, 2013. 36, 38

[LJ17a] Dokyung Lee and Jechang Jeong. Fast CU size decision algorithm using machine learning for HEVC intra coding. *Signal Processing: Image Communication*, December 2017. 33, 34

[LJ17b] Dokyung Lee and Jechang Jeong. Fast intra coding unit decision for high efficiency video coding based on statistical information. *Signal Processing: Image Communication*, 55:121–129, July 2017. 32, 34

[LJHC18] Ting-Lan Lin, Hui-Yu Jiang, Jing-Ya Huang, and Pao-Chi Chang. Fast intra coding unit partition decision in H.266/FVC based on spatial features. *Journal of Real-Time Image Processing*, July 2018. 157

[LLW+17] Xingang Liu, Yinbo Liu, Peicheng Wang, Chin-Feng Lai, and Han-Chieh Chao. An Adaptive Mode Decision Algorithm Based on Video Texture Characteristics for HEVC Intra Prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(8):1737–1748, August 2017. 36, 37, 38

[LNC96] J.T. Ludwig, S.H Nawab, and A.P. Chandrakasan. Low-power digital filtering using approximate processing. *IEEE Journal of Solid-State Circuits*, 31(3):395–400, 1996. 49

[LTLHTY12] Hui Li Tan, Fengjiao Liu, Yih Han Tan, and Chuohao Yeo. ON FAST CODING TREE BLOCK AND MODE DECISION FOR HIGH-EFFICIENCY VIDEO CODING (HEVC). 2012. 31, 34, 35, 36

[LZZ14] Liquan Shen, Zhaoyang Zhang, and Zhi Liu. Adaptive Inter-Mode Decision for HEVC Jointly Utilizing Inter-Level and Spatiotemporal Correlations. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(10):1709–1722, October 2014. 35, 36

[MAH+17a] Alexandre Mercat, Florian Arrestier, Wassim Hamidouche, Maxime Pelcat, and Daniel Menard. Constrain the Docile CTUs: an In-Frame Complexity Allocator for HEVC Intra Encoders. In *Acoustics, Speech and Sig-*

nal Processing (ICASSP), 2017 IEEE International Conference on. IEEE, 2017. 6, 168

[MAH+17b] Alexandre Mercat, Florian Arrestier, Wassim Hamidouche, Maxime Pelcat, and Daniel Menard. Energy Reduction Opportunities in an HEVC Real-Time Encoder. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 1158–1162. IEEE, 2017. 5, 30, 63, 163

[MAP+17] Alexandre Mercat, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Prediction of Quad-Tree Partitioning for Budgeted Energy HEVC Encoding. In *Signal Processing Systems (SiPS), 2017 IEEE Workshop on*, pages 1–6. IEEE, 2017. 6, 86, 164

[MAP+18a] Alexandre Mercat, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Machine Learning Based Choice of Characteristics for the One-Shot Determination of the HEVC Intra Coding Tree. In *2018 Picture Coding Symposium (PCS)*, pages 263–267. IEEE, 2018. 6, 165

[MAP+18b] Alexandre Mercat, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. On predicting the HEVC intra quad-tree partitioning with tunable energy and rate-distortion. *Journal of Real-Time Image Processing*, pages 1–14, July 2018. 6

[MAP+18c] Alexandre Mercat, Florian Arrestier, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Probabilistic Approach versus Machine Learning for One-Shot Quad-Tree Prediction in an Intra HEVC Encoder. *Journal of Signal Processing Systems*, 2018. 6

[MBG+13] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. The latest open-source video codec VP9 - An overview and preliminary results. In *2013 Picture Coding Symposium (PCS)*, pages 390–393, San Jose, CA, USA, December 2013. IEEE. 12

[MBP+17a] Alexandre Mercat, Justine Bonnot, Maxime Pelcat, Karol Desnos, Wassim Hamidouche, and Daniel Menard. Smart Search Space Reduction for Approximate Computing: a Low Energy HEVC Encoder Case Study. *Journal of Systems Architecture*, September 2017. 5, 76

[MBP+17b] Alexandre Mercat, Justine Bonnot, Maxime Pelcat, Wassim Hamidouche, and Daniel Menard. Exploiting Computation Skip to Reduce Energy Consumption by Approximate Computing, an HEVC Encoder Case Study. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, 2017. 5

[MCRR11] Debabrata Mohapatra, Vinay K. Chippa, Anand Raghunathan, and Kaushik Roy. Design of voltage-scalable meta-functions for approximate computing. In *2011 Design, Automation & Test in Europe*, pages 1–6. IEEE, 2011. 49

[MDVPO90] Peter Macken, Marc Degrauwe, Mark Van Paemel, and Henri Oguey. A voltage reduction technique for digital systems. page 29, February 1990. 48

[Mul17] MulticoreWare. x265 HEVC Encoder / H.265 Video Codec. http://x265.org/, 2017. 26

[NMD+09] Bich-yen Nguyen, Carlos Mazure, Daniel Delprat, Cecile Aulnette, Nicolas Daval, François Andrieu, and Olivier Faynot. Overview of FDSOI technology from substrate to device. *Semiconductor Device Research Symposium, 2009. ISDRS '09. International*, pages 1–2, 2009. 48

[NMP16] Erwan Nogues, Daniel Menard, and Maxime Pelcat. Algorithmic-level Approximate Computing Applied to Energy Efficient HEVC Decoding. *IEEE Transactions on Emerging Topics in Computing*, pages 1–1, 2016. 50

[noa] Joint Collaborative Team on Video Coding Reference Software, ver. HM. 42

[NPMM16] Erwan Nogues, Maxime Pelcat, Daniel Menard, and Alexandre Mercat. Energy Efficient Scheduling of Real Time Signal Processing Applications through Combined DVFS and DPM. In *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pages 622–626, Heraklion, Crete, Greece, February 2016. IEEE. 109

[OR98] Antonio Ortega and Kannan Ramchandran. Rate-Distortion Methods for Image and Video Compression. *IEEE Signal Prcess*, pages 560–576, 1998. 114

[OSS+12] Jens-Rainer Ohm, Gary J. Sullivan, Heiko Schwarz, Thiow Keng Tan, and Thomas Wiegand. Comparison of the Coding Efficiency of Video Coding Standards—Including High Efficiency Video Coding (HEVC). *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1669–1684, December 2012. 12

[PCL16] Kuan-Kai Peng, Jui-Chiu Chiang, and Wen-Nung Lie. Low Complexity Depth Intra Coding Combining Fast Intra Mode and Fast CU Size Decision in 3d-HEVC. pages 1126–1130. IEEE, 2016. 33, 63, 86

[PMP+16] Wagner Penny, Italo Machado, Marcelo Porto, Luciano Agostini, and Bruno Zatt. Pareto-based energy control for the HEVC encoder. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 814–818. IEEE, 2016. 42, 44

[Qua14] Qualcomm. Snapdragon 810 processor product brief. In *https://www.qualcomm.com/documents/snapdragon-810-processor-product-brief*, 2014. 26

[Qui14] John Ross Quinlan. *C4. 5: Programs for machine learning*. Elsevier, 2014. 90

[RCAFE+14] Damián Ruiz-Coll, Velibor Adzic, Gerardo Fernández-Escribano, Hari Kalva, José Luis Martínez, and Pedro Cuenca. Fast partitioning algorithm for HEVC Intra frame coding using machine learning. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 4112–4116. IEEE, 2014. 34, 35, 79, 86, 87, 107

[RFEA+15] Damián Ruiz, Gerardo Fernández-Escribano, Velibor Adzic, Hari Kalva, José Luis Martínez, and Pedro Cuenca. Fast CU partitioning algorithm for HEVC intra coding using data mining. *Multimedia Tools and Applications*, pages 861–894, November 2015. 34, 35, 79, 86, 87, 107

[Ric04] Iain E. Richardson. *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia*. John Wiley & Sons, 2004. 14

[RSNP12] Lakshminarayanan Renganarayana, Vijayalakshmi Srinivasan, Ravi Nair, and Daniel Prener. Programming with relaxed synchronization. In *Proceedings of the 2012 ACM workshop on Relaxing synchronization for multicore and manycore scalability*, pages 41–50. ACM, 2012. 48, 49, 185

[San16] Sandvine. 2016 - Global Internet Phenomena - Latin America & North America. In *https://www.sandvine.com/hubfs/downloads/archive/2016-global-internet-phenomena-report-latin-america-and-north-america.pdf*, page 15, 2016. 3

[SBS14] Vivienne Sze, Madhukar Budagavi, and Gary J. Sullivan, editors. *High Efficiency Video Coding (HEVC)*. Integrated Circuits and Systems. Springer International Publishing, Cham, 2014. 3, 24, 26, 159

[SDMHR11] Stelios Sidiroglou-Douskos, Sasa Misailovic, Henry Hoffmann, and Martin Rinard. Managing performance vs. accuracy trade-offs with loop perforation. In *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, pages 124–134. ACM, 2011. 50

[SFB+08] Angada B. Sachid, Roswald Francis, Maryam Shojaei Baghini, Dinesh K. Sharma, Karl-Heinz Bach, Reinhard Mahnkopf, and V. Ramgopal Rao. Sub-20 nm gate length FinFET design: Can high-k spacers make a difference? In *2008 IEEE International Electron Devices Meeting*, pages 1–4. IEEE, 2008. 48

[SGZ13] Yunyu Shi, Zhiyong Gao, and Xiaoyun Zhang. Early TU Split Termination in HEVC Based on Quasi-Zero-Block. Atlantis Press, 2013. 38, 39

[SHDP17] Naty Sidaty, Wassim Hamidouche, Olivier Deforges, and Pierrick Philippe. Emerging video coding performance: 4k quality monitoring. In *2017 Ninth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–3, Erfurt, Germany, May 2017. IEEE. 156

[SLZ+13] Liquan Shen, Zhi Liu, Xinpeng Zhang, Wenqiang Zhao, and Zhaoyang Zhang. An Effective CU Size Decision Method for HEVC Encoders. *IEEE Transactions on Multimedia*, 15(2):465–470, February 2013. 31, 33, 34

[SOHW12] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, December 2012. 3, 20, 24, 26, 159

[Sul18] Gary J. Sullivan. Video Coding Standards Progress Report: Joint Video Experts Team Launches the Versatile Video Coding Project. *SMPTE Motion Imaging Journal*, 127:94–98, 2018. 12, 156

[SW98] Gary J. Sullivan and Thomas Wiegand. Rate-Distortion Optimization for Video Compression. IEEE Signal Process(15):74–90, 1998. 16, 114

[SWFL15] Xiwu Shang, Guozhong Wang, Tao Fan, and Yan Li. Fast CU size decision and PU mode decision algorithm in HEVC intra coding. pages 1593–1597. IEEE, September 2015. 34, 35, 37, 38

[SY13] Xiaolin Shen and Lu Yu. CU splitting early termination based on weighted SVM. *EURASIP Journal on Image and Video Processing*, 2013(1):4, 2013. 32, 34, 86

[SZA13] Liquan Shen, Zhaoyang Zhang, and Ping An. Fast CU size decision and mode decision algorithm for HEVC intra coding. *IEEE Transactions on Consumer Electronics*, 59(1):207–213, 2013. 33, 35, 63, 79, 80, 87, 107

[SZZ+15] Liquan Shen, Zhaoyang Zhang, Xinpeng Zhang, Ping An, and Zhi Liu. Fast TU size decision algorithm for HEVC encoders using Bayesian theorem detection. *Signal Processing: Image Communication*, 32:121–128, March 2015. 38, 39

[TK15] Junaid Tariq and Sam Kwong. Hybrid Fast Intra Mode Decision and Early Termination of Prediction Unit (PU) Splitting for HEVC. pages 1782–1786. IEEE, October 2015. 31, 34, 37, 38

[TWLY07] An-Chao Tsai, Jhing-Fa Wang, Wei-Guang Lin, and Jar-Ferr Yang. A Simple and Robust Direction Detection Algorithm for Fast H.264 Intra Prediction. pages 1587–1590. IEEE, July 2007. 37

[TWM+16] Thiow Keng Tan, Rajitha Weerakkody, Marta Mrak, Naeem Ramzan, Vittorio Baroncini, Jens-Rainer Ohm, and Gary J. Sullivan. Video Quality Evaluation Methodology and Verification Testing of HEVC Compression Performance. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):76–90, January 2016. 12, 29, 159

[Ult17] UltraVideoGroup. Kvazaar HEVC Encoder. http://ultravideo.cs.tut.fi/#encoder, 2017. 26, 155

[UMF13] Anna Ukhanova, Simone Milani, and Soren Forchhammer. Game-theoretic rate-distortion-complexity optimization for HEVC. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 1995–1999. IEEE, 2013. 42, 44

[Van17] Vantrix. F265 Open Source HEVC/H.265 Project. http://vantrix.com/f-265-2/, 2017. 26

[VKL+15] Marko Viitanen, Ari Koivula, Ari Lemmetti, Jarno Vanne, and Timo D. Hamalainen. Kvazaar HEVC encoder for efficient intra coding. In *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, pages 1662–1665. IEEE, 2015. 56, 63

[VTE+16] Jean-Marc Valin, Timothy B. Terriberry, Nathan E. Egge, Thomas Daede, Yushin Cho, Christopher Montgomery, and Michael Bebenita. Daala: Building a next-generation video codec from unconventional technology. In *2016 IEEE 18th International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6, Montreal, QC, Canada, September 2016. IEEE. 12

[VVH14] Jarno Vanne, Marko Viitanen, and Timo D. Hamalainen. Efficient Mode Decision Schemes for HEVC Inter Prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(9):1579–1593, September 2014. 35, 36

[VVHH12] Jarno Vanne, Marko Viitanen, Timo D. Hamalainen, and Antti Hallapuro. Comparative Rate-Distortion-Complexity Analysis of HEVC and AVC Video Codecs. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1885–1898, December 2012. 29, 159

[WBSS04] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 14

[Wie15] Mathias Wien. *High Efficiency Video Coding*. Signals and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015. 3, 24, 26, 159

[WSBL03] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, July 2003. 56, 105, 114

[WWW16] Xiuzhe Wu, Hanli Wang, and Zhihua Wei. Bayesian rule based fast TU depth decision algorithm for high efficiency video coding. pages 1–4. IEEE, November 2016. 38, 39

[WWZ+17] Zhao Wang, Shiqi Wang, Jian Zhang, Shanshe Wang, and Siwei Ma. Effective Quadtree Plus Binary Tree Block Partition Decision for Future Video Coding. In *2017 Data Compression Conference (DCC)*, pages 23–32, Snowbird, UT, USA, April 2017. IEEE. 157

[WX16] Xuanjing Wang and Yonglin Xue. Fast HEVC intra coding algorithm based on Otsu's method and gradient. In *Broadband Multimedia Systems and Broadcasting (BMSB), 2016 IEEE International Symposium on*, pages 1–5. IEEE, 2016. 33, 34, 35, 36, 38, 63, 64, 164

[XLM+14] Jian Xiong, Hongliang Li, Fanman Meng, Shuyuan Zhu, Qingbo Wu, and Bing Zeng. MRF-Based Fast HEVC Inter CU Decision With the Variance of Absolute Differences. *IEEE Transactions on Multimedia*, 16(8):2141–2153, December 2014. 31, 34

[XLWM14] Jian Xiong, Hongliang Li, Qingbo Wu, and Fanman Meng. A Fast HEVC Inter CU Selection Method Based on Pyramid Motion Divergence. *IEEE Transactions on Multimedia*, 16(2):559–564, February 2014. 31, 34

[YG17] Mingyuan Yang and Christos Grecos. Fast intra encoding decisions for high efficiency video coding standard. *Journal of Real-Time Image Processing*, 13(4):797–806, December 2017. 34, 36, 37, 38, 39

[YJJ10] Piao Yinji, Min Junghye, and Chen Jiangle. Encoder improvement of unified intra prediction. Guangzhou, July 2010. 21

[YKX+15] Yun Zhang, Sam Kwong, Xu Wang, Hui Yuan, Zhaoqing Pan, and Long Xu. Machine Learning-Based Coding Unit Depth Decisions for Flexible Complexity Allocation in High Efficiency Video Coding. *IEEE Transactions on Image Processing*, 24(7):2225–2238, July 2015. 33, 34

[YLL16] Yingbiao Yao, Xiaojuan Li, and Yu Lu. Fast intra mode decision algorithm for HEVC based on dominant edge assent distribution. *Multimedia Tools and Applications*, 75(4):1963–1981, February 2016. 36, 37, 38

[ZLL15] Jinlei Zhang, Bin Li, and Houqiang Li. An Efficient Fast Mode Decision Method for Inter Prediction in HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(8):1502–1515, 2015. 36, 79, 80, 87, 107

[ZOS15] Wenjun Zhao, Takao Onoye, and Tian Song. Hierarchical Structure-Based Fast Mode Decision for H.265/HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(10):1651–1664, October 2015. 34, 35, 36

[ZSZG17] Tao Zhang, Ming-Ting Sun, Debin Zhao, and Wen Gao. Fast Intra-Mode and CU Size Decision for HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(8):1714–1726, August 2017. 37, 38

[ZWK13] Tiesong Zhao, Zhou Wang, and Sam Kwong. Flexible Mode Selection and Complexity Allocation in High Efficiency Video Coding. *IEEE Journal of Selected Topics in Signal Processing*, 7(6):1135–1144, December 2013. 42, 44

[ZZK+17] Linwei Zhu, Yun Zhang, Sam Kwong, Xu Wang, and Tiesong Zhao. Fuzzy SVM-Based Coding Unit Decision in HEVC. *IEEE Transactions on Broadcasting*, pages 1–14, 2017. 33, 34

[ZZL17] Mengmeng Zhang, Xiaojun Zhai, and Zhi Liu. Fast and adaptive mode decision and CU partition early termination algorithm for intra-prediction in HEVC. *EURASIP Journal on Image and Video Processing*, 2017(1), December 2017. 33, 34, 36, 37, 38

[ZZMZ11] Liang Zhao, Li Zhang, Siwei Ma, and Debin Zhao. Fast mode decision algorithm for intra prediction in HEVC. In *Visual Communications and Image Processing (VCIP), 2011 IEEE*, pages 1–4. IEEE, 2011. 21

INSA de RENNES
Service des Formations

AVIS DU JURY SUR LA REPRODUCTION DE LA THESE SOUTENUE

Titre de la thèse:
Complexity Control for Low-Power HEVC Encoding

Nom Prénom de l'auteur : MERCAT ALEXANDRE

Membres du jury :
- Monsieur THIESSE Jean-Marc
- Monsieur Cagnazzo Marco
- Monsieur MENARD Daniel
- Monsieur HAMIDOUCHE Wassim
- Monsieur COUDOUX François-Xavier
- Madame MOKRAOUI Anissa
- Monsieur PELCAT Maxime
- Monsieur DUFAUX Frédéric

Président du jury : Mme Anissa MOKRAOUI

Date de la soutenance : 07 Décembre 2018

Reproduction de la these soutenue

☒ Thèse pouvant être reproduite en l'état
☐ Thèse pouvant être reproduite après corrections suggérées

Fait à Rennes, le 07 Décembre 2018

Signature du président de jury

Le Directeur,

M'hamed DRISSI

INSTITUT NATIONAL des SCIENCES APPLIQUÉES - RENNES

Titre : Contrôle de la Complexité pour l'Encodage HEVC Basse Consommation d'Energie

Mots clés : compression vidéo, HEVC, réduction de complexité, contrôle de complexité, intelligence artificielle

Résumé : L'Internet des objets (IoT) est devenu une réalité et ses applications pressenties vont fortement augmenter la demande de vidéo mobile. En conséquence, les systèmes montent en complexité algorithmique et le portage du codage vidéo sur plates-formes embarquées devient problématique. Les nouveaux contenus vidéo 4K et 360°, venant avec des résolutions spatiales (8K, 16K) et temporelles (120 images/seconde) élevées compliquent encore le problème. Il est donc nécessaire de réduire l'empreinte des nouveaux codecs tels que HEVC tout en préservant les performances en compression et en qualité d'image de ces codecs. La performance énergétique limitée des batteries des systèmes embarqués pousse à proposer de nouvelles méthodes pour ajuster et contrôler la complexité et l'énergie des codecs HEVC.

Ce document propose un ensemble d'études dont l'objectif est d'ajuster et de contrôler la complexité et donc la consommation énergétique de l'encodeur HEVC. Deux méthodes de prédiction de découpe de CTU sont proposées : la première basée sur une approche statistique utilisant la variance de l'image et la seconde utilisant l'intelligence artificielle. À partir de cette prédiction, une méthode est proposée pour ajuster la complexité de l'encodage HEVC. Cette solution étend l'espace de recherche autour de la prédiction et alloue la complexité dans l'image afin de minimiser les dégradations en termes de compression et de qualité. Enfin, un système de contrôle temps réel de la complexité d'encodage est proposé. Il démontre l'applicabilité des contributions de ce document en maintenant la complexité d'encodage proche d'une consigne.

Title : Complexity Control for Low-Power HEVC Encoding

Keywords : video coding, HEVC, complexity reduction, complexity control, machine learning

Abstract : The Internet of Things (IoT) is now a reality. Forthcoming applications will boost mobile video demand to an unprecedented level. The induced increase in computational complexity is a challenge when executing in real-time new video coding standards on embedded platforms, limited in computing, memory, and energy. New 4K UHD and 360-degree video contents coming with high spatial (8K, 16K) and temporal (120fps) resolutions further complicate the problem. In this context, codecs such as HEVC (High Efficiency Video Coding) must be worked on to reduce their complexity while preserving the bitrate and image quality. The bounded energy density of embedded systems batteries requires designers to propose new methods scaling and controlling the complexity and energy consumption of HEVC codecs.

This document presents a set of studies aiming at scaling and controlling the complexity, and therefore the energy consumption, of HEVC Intra encoding. Two methods of quad-tree partitioning prediction in "one-shot" are proposed: one based on variance-aware statistic approach and one based on Machine Learning using data-mining classifiers. From the obtained prediction, a generic tunable complexity scheme of HEVC encoding is introduced. It expands the search space around the original partitioning prediction and allocates complexity in a frame while minimizing performance loss in terms of bitrate and visual quality. Finally, a real-time control system is created that dynamically manages the encoding process to keep the encoding complexity under a specific target. It demonstrates the applicability of the major contributions of this document.