Universidade de Aveiro
2016

Departamento de Eletrónica,
Telecomunicações e Informática

Alexandre José Batista Santiago

HEVC Scalable Analysis: Performance and Bitrate Control

Análise do HEVC Escalável: Desempenho e Controlo de Débito

Universidade de Aveiro
2016

Departamento de Eletrónica,
Telecomunicações e Informática

Alexandre José Batista Santiago

HEVC Scalable Analysis: Performance and Bitrate Control

Análise do HEVC Escalável: Desempenho e Controlo de Débito

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Dr. António Navarro, Professor auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

o júri

presidente

Prof. Doutor Adão Silva
Professor Auxiliar do Dep. Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

Prof. Doutor Paulo Lourenço Nunes
Professor Auxiliar do Departamento de Ciências e Tecnologias de Informação do Instituto Superior de Ciências do Trabalho e da Empresa – Instituto Universitário de Lisboa

Prof. Doutor António Navarro
Professor Auxiliar do Dep. Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

agradecimentos

Em primeiro lugar a minha família pelo apoio dado ao longo da vida.

Ao professor António Navarro pela orientação, disponibilidade e interesse que sempre demonstrou ao longo da dissertação.

A todos os professores pelos ensinamentos dados durante a minha formação, desde a entrada na escola primária até á finalização do mestrado.

palavras-chave

Codificação de Vídeo, Codificação de Vídeo de Alta Eficiência (HEVC), Codificação de Vídeo Escalável de Alta Eficiência (SHVC), Controlo de Débito, Funções Débito-distorção, Modelo R- λ.

resumo

Esta dissertação apresenta um estudo da norma de codificação de vídeo de alta eficiência (HEVC) e a sua extensão para vídeo escalável, SHVC. A norma de vídeo SHVC proporciona um melhor desempenho quando codifica várias camadas em simultâneo do que quando se usa o codificador HEVC numa configuração *simulcast*. Ambos os codificadores de referência, tanto para a camada base como para a camada superior usam o mesmo modelo de controlo de débito, modelo R-λ, que foi otimizado para o HEVC. Nenhuma otimização de alocação de débito entre camadas foi até ao momento proposto para o modelo de testes (SHM 8) para a escalabilidade do HEVC (SHVC). Derivamos um novo modelo R-λ apropriado para a camada superior e para o caso de escalabilidade espacial, que conduziu a um ganho de BD-débito de 1,81% e de BD-PSNR de 0,025 em relação ao modelo de débito-distorção existente no SHM do SHVC. Todavia, mostrou-se também nesta dissertação que o proposto modelo de R-λ não deve ser usado na camada inferior (camada base) no SHVC e por conseguinte no HEVC.

keywords

Video Coding, High Efficiency Video Coding (HEVC), Scalable High efficiency Video Coding (SHVC), Rate Control, Rate-distortion Function, R-λ Model.

abstract

This dissertation provides a study of the High Efficiency Video Coding standard (HEVC) and its scalable extension, SHVC. The SHVC provides a better performance when encoding several layers simultaneously than using an HEVC encoder in a simulcast configuration. Both reference encoders, in the base layer and in the enhancement layer use the same rate control model, R-λ model, which was optimized for HEVC. No optimal bitrate partitioning amongst layers is proposed in scalable HEVC (SHVC) test model (SHM 8). We derived a new R-λ model for the enhancement layer and for the spatial case which led to a DB-rate gain of 1.81% and DB-PSNR gain of 0.025 in relation to the rate-distortion model of SHM-SHVC. Nevertheless, we also show in this dissertation that the proposed model of R-λ should not be used neither in the base layer nor in HEVC.

Contents

List of Figures

List of Tables

List of Acronyms

AMVP	Advanced Motion Vector Prediction
AVC	Advanced Video Coding
BD PSNR	Bjontegaard Delta PSNR
BD Rate	Bjontegaard Delta Bitrate
BL	Base Layer
BLA	Broken Link Access
Bpp	Bit per Pixel
BU	Basic Unit
CABAC	Context-bases Adaptive Binary Arithmetic Coding
CB	Coding Block
CD	Compact Disk
CGS	Color Gamut Scalability
CRA	Clean Random Access
CVS	Coding Video Sequence
CTB	Coding Tree Block
CTU	Coding Tree Unit
CU	Coding Unit
DPB	Decoded Picture Buffer
DVB	Digital Video Broadcasting
D-Q	Distortion-Quantization
EL	Enhance Layer
GOP	Group of Pictures
HD	High Definition
HEVC	High Efficiency Video Coding
HLS	High-Level Syntax
HRD	Hypothetical Reference Decoder
IDR	Instantaneous Decoding Refresh
ILP	Interlayer Prediction
ILR	Interlayer Reference
INBL	Independent Non-Base Layer
IRAP	Intra Random Access Point
JCT-VC	Joint Collaborative Team on Video Coding
LUT	Lookup Table
MAD	Mean Absolute Difference

MC	Motion Compensation
MFM	Motion Field Mapping
MPEG	Moving Picture Experts Group
MPM	Most Probable Mode
MPS	Most Probable Symbol
MTU	Maximum Transmit Unit
MV	Motion Vector
NAL	Network Abstraction Layer
NUH	NAL Unit Header
PB	Prediction Block
POC	Picture Order Count
PPS	Picture Parameter Set
PSNR	Peak Signal to Noise Ration
PU	Prediction Unit
QP	Quantization Parameter
RADL	Random Access Decodable Leading
RASL	Random Access Skipped Leading
RDO	Rate Distortion Optimization
RL	Reference Layer
RPS	Reference Picture Set
RQT	Residual Quad-Tree
R-D	Rate-Distortion
R-Q	Rate-Quantization
SAO	Sample Adaptive offset
SEI	Supplementary Enhancement Information
SHVC	Scalable High efficiency Video Coding
SNR	Signal to Noise Ratio
SPS	Sequence Parameter Set
STSA	Step-wise Temporal Sub-layer Access
SVC	Scalable Video Coding
TB	Transform Block
TMVP	Temporal Motion Vector Prediction
TSA	Temporal Sub-layer Access
TU	Transform Unit
VCEG	Video Coding Experts Group
VCL	Video Coding Layer

VPS	Video Parameter Set
VUI	Visual Usability Information
WPP	Wavefront Parallel Processing

1 Introduction

1.1 Video Coding

Before the advent of digital video technologies, video was stored as an analog signal on magnetic tapes. When the compact disc (CD) entered the market as a digital format replacement for analog audio, engineers saw the potential to also store video in digital format. The fact that large amount of storage and bandwidth was needed to record and convey raw video, led to the creation of tools which could reduce the amount of data used to represent the raw video. Since then, video signals have been the subject of considerable research. In the last fifteen years, the increase of bandwidth in the telecommunications allowed the growing availability of digital transmission links, which in turn led to a wide range of emerging applications such as digital TV/HDTV broadcasting, video-on-demand, video conferencing, video streaming and several more that have been developed. The most famous case of video application is YouTube with over a 1 billion users [1] and 500 hours of video being uploaded on YouTube databases every minute [2].

The need for international audiovisual standards emerged with the growing commercial interest in these applications and services. The standardization process facilitates equipment interoperability from different manufactures. When video is being broadcasted, its quality depends on the video encoding process and the allocated bandwidth. The encoding process is fundamental since it has an huge impact on the rate-distortion performance and also on the utilization of different resources such as processing power, transmission bandwidth and delay of streaming services.

Digital TV, one of the most popular digital video applications, is based on the success of H.262/MPEG-2 standard and the Digital Video Broadcasting (DVB) standard family [3] [4]. Following the success of MPEG-2, the ITU-T video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) developed the new video standard, H.264/MPEG-4 AVC, also known as Advanced Video Coding (AVC) [5]. H.264/MPEG-4 AVC has achieved considerable progress regarding coding efficiency, enhanced error robustness, increased flexibility and scope of applicability relative to its predecessors [6]. H.264/MPEG-4 AVC has been an enabling technology for digital video in almost every area that was not previously covered by H.262/MPEG-2 and has substantially displaced the older standard within its existing applications domains. Many applications, including broadcast of high definition (HD) TV signals over satellite, cable, and terrestrial transmission systems, video content acquisition and editing systems, camcorders, security applications, Internet and mobile network video, Blu-ray Discs, and real-time conversational applications such as video chat, video conferencing and telepresence systems amply use H.264/MPEG-4 AVC.

However, the increasing diversity of services, the growing popularity of HD video, and beyond-HD formats (e.g., 4k x 2k or 8k x4k resolution) are creating stronger needs for coding efficiency superior to H.264/MPEG-4 AVC capabilities [7]. Traffic caused by video applications targeting mobile devices, as well as the transmission needs for video-on-demand services, are imposing severe challenges on today networks, since the desire for higher quality and resolutions are being imposed by its users. As a result, MPEG and VCEG have formed a Joint Collaborative Team on Video Coding (JCT-VC) to develop a successor to H.264/MPEG-4 AVC. This standard is referred as High Efficiency Video Coding (HEVC) [8]. HEVC has been developed to address essentially all existing applications of H.264/MPEG-4 AVC and to achieve multiples goals, including coding efficiency, data loss resilience, as well as implementability using parallel processing architectures. Results show a reduction in bit rate requirements by half with similar subjective perceptual quality when compared with H.264/MPEG-4 AVC, at the expense of increased computational complexity. The first version of HEVC was completed in January 2013, and several revisions have been deployed since then.

Modern video transmission and storage systems using the Internet and mobiles networks are typically based on RTP/IP [9] for real-time services (conversational and streaming) and on computer file formats like mp4 or 3gp. Most RTP/IP access networks are typically characterized by a wide range of connections qualities and receiving devices. The varying connection quality is resulted from adaptive resource sharing mechanism of these networks addressing the time varying data throughput requirements of a varying number of users. The variety of devices with different capabilities ranging from cell phones with small screens and restricted processing power to high-end PCs with high-definition displays results from the continuous evolution of these endpoints.

Scalable video coding is a highly attractive solution to the problems posed by the characteristic of modern video transmission systems. The term scalability refers to the removal of parts of the video bitstream in order to adapt it to the various needs or preferences of end users as well as to varying terminal capabilities or network conditions.

1.2 Scalable Video Coding

Scalable video coding has been studied and standardized for more than 20 years. The prior international video coding standards H.262/MPEG-2 [10] and H.263 [11] already included several scalable related tools. However, the scalable profiles of those standards have rarely been used, since the characteristics of traditional video transmission systems did not support it, as well as the fact that the spatial and quality scalability features came along with a significant loss in coding efficiency. Another reason was a large increase in the decoder complexity as compared to the corresponding non-scalable profiles.

After a call for proposals by MPEG, for an efficient scalable video coding standard, and after the evaluation phase, in which several subjective tests for a variety of conditions were carried out and the proposals were carefully analyzed regarding their potential for a successful future standard, the scalable extension of H.264/MPEG-4 AVC proposed in [12] was selected as the starting point for the Scalable Video Coding (SVC) extension which was finalized as an amendment of H.264/MPEG-4 AVC.

SVC, the scalable extension of H.264/MPEG-4 AVC, enables a video sequence to be decoded fully or partially with variable quality, resolution and frame rate, in order to adapt to the available network bandwidth or application requirements. It uses a multi-layer scheme to provide spatial and quality scalability. The bitstream comprises the base layer which represents the basic content of the video sequence. From the base layer, the decoded video may have a low frame rate, low resolution or low quality [13]. When the bandwidth resources permit, one or more enhancement layers are transmitted to enhance the perceptual quality. The more bits are transmitted, the better the overall quality.

SVC has never been as widely adopted in products as the non-scalable coding standard, upon which the scalable extension has been based. Even for those applications that are well-suited to scalable coding, adoption in products have been limited. A key impediment to deployment of SVC has been the difficulty of implementation, and the significant implementation differences between scalable and non-scalable video coding standards. During the development of SHVC, the scalable extension of the HEVC standard by the JCT-VC [14], the lessons learned from the past attempts at scalable coding standardization were strongly considered. One of the major problems from the past, the implementation complexity, was minimized by enabling repurposing of multiple single-layer HEVC cores to achieve efficient scalable coding.

As such, SHVC became the first scalable video coding standard that is built upon high-level syntax only (HLS-only) scalable coding framework. Empowered by efficient inter-layer reference picture processing modules, SHVC achieves high scalable coding efficiency without requiring any block-level coding logic changes to the single-layer HEVC cores. Given that SHVC is a finished process, this dissertation will focus on the scalable extension of HEVC standard and in particular on the rate control module.

1.3 Video Coding and Rate Control

The objective of a video encoder is to generate the optimum perceptual video quality, or to minimize distortion, under a certain set of requirements such as channel bandwidth or storage limitations. General speaking, for a specific bit budget, the video encoder should optimally determine a set of the best quantization parameters by minimizing the value of the distortion, since the quantization parameter has a major role in the generation of bits and distortion. If a video sequence is encoded using all the different quantization parameters, then rate and quantization error can be obtained and it is possible to plot the rate-quantization (R-Q) or the distortion-quantization (D-Q) curves. R-Q and D-Q functions characterize the rate-distortion (R-D) behavior of the video encoding process.

There are two main approaches to solve the optimal bit allocation problem: Lagrange's optimization [15] and dynamic programming [16]. However, these methods have a high computational complexity due to the need to determine R-D characteristics of current and future video frames. Therefore, to obtain an estimation of the bit rate without having to implement the whole encoding process, mathematical models can estimate the bit rate or the quantization error. Multiples R-Q and D-Q functions have been reported in previous studies ([17] [18] [19] [20] [21]). Some of these schemes were adopted in standard-compliant video coders, such as, the test model for MPEG-2, TM-5 [17], the test model for H.263, TMN-8 [18] or the verification model for H.264/MPEG-4 AVC, VM-8 [21]. In HEVC a λ-domain rate control was proposed and later adopted by JCT-VC into the reference software of HEVC [22].

1.4 Motivations and Objectives

As discussed, the increase of higher video resolutions and video applications is requiring more efficient video coding standards, and therefore, HEVC was developed. Past scalable extensions were never widely adopted with the key reason being the significant implementation differences between scalable and non-scalable coding. With these differences being minimized in SHVC, the implementation of this tool can now be widely adopted in a variety of applications which make use of different spatial or temporal resolutions as well as video qualities. HEVC and his scalable extension, SHVC, are the key technologies for present and future video coding standards. One objective of this dissertation is the study of the scalable extension SHVC.

We start with a performance comparison between the SHVC and HEVC in a simulcast configuration. We compare three parameters, bitrate cost, encoding time and PSNR gain/loss in the enhancement layer (EL). Next, we investigate whether the rate-distortion model used for HEVC single-layer is appropriate for the EL in scalable encoder and propose a new R-D model. Finally, we implement the proposed R-D model and compare with the model used in the reference software in multi-layer encoding.

1.5 Outline of the Dissertation

The remainder of this dissertation is organized as follows:

Chapter 2 surveys the most recent digital video coding standard, HEVC, with detailed description of the new features.

Chapter 3 presents an overview to the scalable extension of HEVC, SHVC. This is the standard that provides the framework for our investigation.

Chapter 4 presents a brief review of the rate control algorithms used for HEVC single-layer. Since the rate control algorithms in HEVC, which are replicated into each layer in the scalable extension SHVC, we investigate if the R-D model is appropriate for the multi-layer case. In the final of this chapter we proposed a new R-D model.

Chapter 5 summarizes the results of the dissertation.

Finally, in Chapter 6, we draw some conclusions, and discuss future research work.

2 HEVC Standard: An Overview

2.1 Introduction

As previously stated, the H.265/HEVC is the new video coding standard. It was released as Recommendation ITU-T H.265, which is an extensive document that includes the normative content to which H.265 codecs should conform. As stated by ITU, in this document [23], the HEVC standard "was developed in response to the growing need for higher compression of moving pictures for various applications such as Internet streaming, communication, videoconferencing, digital storage media and television broadcasting. It is also designed to enable the use of the coded video representation in a flexible manner for a wide variety of network environments". Furthermore, data loss resilience, implementability using parallel processing architectures and ease of transport system integration have been taken into account.

2.2 HEVC High Level Syntax

The bitstream of HEVC consist of a sequence of data units which are called network abstraction layer (NAL) units, with numerous elements being inherited from the NAL unit of H.264/AVC [24]. NAL units carry parameter sets with high-level information concerning the entire video sequence or a subset of the pictures within it. Other NAL units carry coded samples in the shape of slices that belong to one of the various pictures types defined in HEVC. The picture types can indicate if the picture can be discarded without affecting the decodability of other pictures or can indicate positions in the bitstream where random access is possible. The slices contain information to manage decoded pictures, which indicate what previous pictures are to keep and in which order they are to be output. There are another NAL unit that contain optional supplementary enhancement information (SEI) that assist the decoding process or may assist in other ways such as providing hints about how best to display the video. All these elements that describe the structure of the bitstream or provide information are known as the "high-level syntax" part of HEVC [25].

2.2.1 The NAL Unit Header and the HEVC Bitstream

HEVC have two classes of NAL units, video coding layer (VCL) NAL units and non-VCL NAL units [25]. Each VCL NAL unit carries one slice segment of coded picture data while the non-VCL NAL unit carries control information related to multiple coded pictures. A coded picture along with the non-VCL NAL units that are associated with the coded picture, is called an HEVC access unit. There is no requirement that an access unit must contain any non-VCL NAL units, however it must consist of one or more VCL NAL units since each access unit contains a coded picture.

2.2.1.1 The NAL unit Header

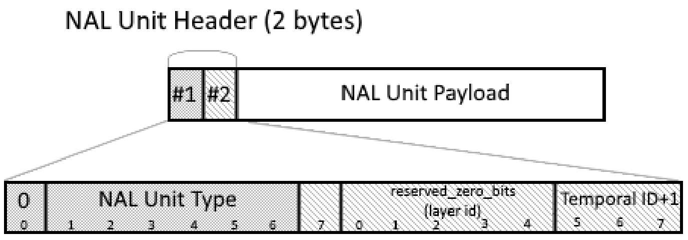

Figure 2.1 - HEVC NAL unit interface.

Figure 2.1 shows the 2 bytes structure of the NAL unit header. Both VCL and non-VCL NAL units start with this two byte NAL unit header that is designed to make it easy to parse the main properties of a NAL unit (what type it is, and what layer and temporal sub-layer it belongs to).

The first bit of the NAL unit header is always set to ‘0’ in order to prevent generating bit patterns that could be interpreted as MPEG-2 start codes in legacy systems. The next six bits hold the type of the NAL unit, identifying the type of data that is carried. This six bits grant 64 possible NAL unit type values, which are allocated equally between VCL and non-VCL NAL units. Table 2.1 list the NAL unit types and their associated meanings and type classes.

Table 2.1 - NAL unit types, meanings and type classes.

Type	Meaning	Class	Type	Meaning	Class
0, 1	Slice segment of ordinary trailing picture	VCL	32	Video parameter set (VPS)	non-VCL
2, 3	Slice segment of TSA picture	VCL	33	Sequence parameter set (SPS)	non-VCL
4, 5	Slice segment of STSA picture	VCL	34	Picture parameter set (PPS)	non-VCL
6, 7	Slice segment of RADL picture	VCL	35	Access unit delimiter	non-VCL
8, 9	Slice segment of RASL picture	VCL	36	End of sequence	non-VCL
10-15	Reserved for future use	VCL	37	End of bitstream	non-VCL
16-18	Slice segment of BLA picture	VCL	38	Filler data	non-VCL
19, 20	Slice segment of IDR picture	VCL	39, 40	SEI messages	non-VCL
21	Slice segment of CRA picture	VCL	41-47	Reserved for future use	non-VCL
22-31	Reserved for future use	VCL	48-63	Unspecified (available for system use)	non-VCL

The following six bits (last bit from first byte and first 5 bits of second byte) contains the layer identifier that indicates what layer the NAL unit belongs to, and is designed for scalable and layered extensions use.The last three bits of the NAL unit header contains the temporal identifier of the NAL unit, with seven possible values to represent and one value forbidden. Each access unit in HEVC belongs to one temporal sub-layer, as indicated by the temporal ID. Since every access unit belongs to one temporal sub-layer, all VCL NAL units belonging to the same access unit must have the same temporal ID indicated in their NAL unit headers.

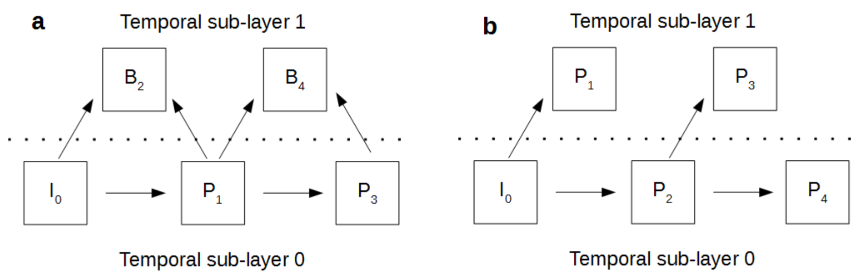

Figure 2.2 - Temporal sub-layer examples.

Pictures of any lower temporal sub-layer are forbidden, in the decoding process, from having dependencies on data sent for a higher temporal sub-layer. As shown in Figure 2.2, no pictures in the lower sub-layer reference any pictures in the higher sub-layer. This restriction allow the removal of higher sub-layers from the bitstream and consequently a decrease of pictures on it. This process is done by discarding all NAL units that have a temporal ID higher than the target temporal ID value, and it can allow rate adaptation in a network [25].

2.2.1.2 VCL NAL Unit Types

Table 2.2 - The 32 HEVC VCL NAL unit types.

Trailing non-IRAP			
Non-TSA, non-STSA Trailing	0	TRAIL_N	Sub-layer non-reference
	1	TRAIL_R	Sub-layer reference
Temporal sub-layer access (TSA)	2	TSA_N	Sub-layer non-reference
	3	TSA_R	Sub-layer reference
Step-wise temporal sub-layer (STSA)	4	STSA_N	Sub-layer non-reference
	5	STSA_R	Sub-layer reference

Leading pictures			
Random access decodable leading (RADL)	6	RADL_N	Sub-layer non-reference
	7	RADL_R	Sub-layer reference
Random Access skipped leading (RASL)	8	RASL_N	Sub-layer non-reference
	9	RASL_R	Sub-layer reference
Intra random access point (IRAP) pictures			
Broken link access (BLA)	16	BLA_W_LP	May have leading pictures
	17	BLA_W_RADL	May have RADL leading
	18	BLA_N_LP	Without leading pictures
Instantaneous decoding refresh (IDR)	19	IDR_W_RADL	May have leading pictures
	20	IDR_N_LP	Without leading pictures
Clean random access (CRA)	21	CRA	May have leading pictures
Reserved			
Reserved non-IRAP	10-15	RSV	
Reserved IRAP	22-23	RSV	
Reserved non-IRAP	24-31	RSV	

Table 2.2 shows all 32 VCL NAL unit types and their NAL unit type values in the NAL unit header. Every VCL NAL unit of the same access unit shall have the same value of NAL unit type, and that value defines the type of access unit and its coded picture [26]. For instance, when all VCL NAL units of an access unit have NAL unit type with value 21, the access unit is called CRA access unit and the coded picture is called a CRA picture.

There are three basic classes of pictures in HEVC: intra random access point (IRAP) pictures, leading pictures and trailing pictures.

The IRAP picture type consist of NAL unit types with values between 16 and 23. Every IRAP picture must belong to temporal sub-layer 0 and be coded without using the content of any other pictures as reference data (i.e., using only intra coding techniques). The IRAP picture types are used to provide points in the bitstream where decoding can be started.

A bitstream must always start with an IRAP picture, but there can be many others IRAP pictures throughout the bitstream. IRAP pictures provide the possibility to tune in to a bitstream (e.g., switching from one TV channel to another), or seek the temporal position in video content (e.g., move the current play position in a video), or to seamlessly switch from one video stream to another in the compressed domain.

A leading and trailing picture are pictures that follows a particular IRAP picture in decoding order. In the case of leading picture, the output order precedes the IRAP picture, while the trailing picture follows the IRAP picture in both decoding and output order. Figure 2.3 shows examples of leading and trailing pictures.

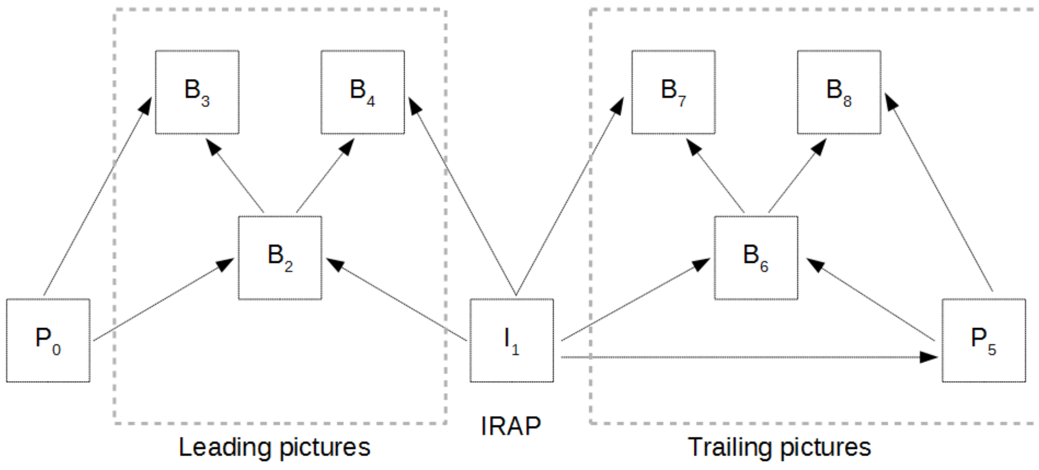

Figure 2.3 - Leading and trailing pictures.

Trailing pictures must use one of the trailing picture NAL unit types 0-5. Trailing pictures of a particular IRAP picture are not allowed to depend on any leading or trailing pictures of previous IRAP pictures. Instead, they can only depend on the associated IRAP picture and other trailing pictures of the same IRAP picture. All leading pictures of an IRAP picture must precede, in decoding order, all trailing pictures that are associated with the same IRAP picture. This means that the decoding order of associated pictures is always: (1) the IRAP picture, (2) the associated leading pictures and (3) the associated trailing pictures [24].

2.2.1.2.1 Ordinary Trailing (TRAIL), Temporal Sub-layer Access (TSA) and Step-wise Temporal Sub-layer Access (STSA) Pictures

Trailing pictures can belong to any temporal sub-layer. They can reference the associated IRAP picture and other trailing pictures associated with the same IRAP picture, but they cannot reference leading pictures. They also cannot be output after the next IRAP picture in decoding order is output [25].

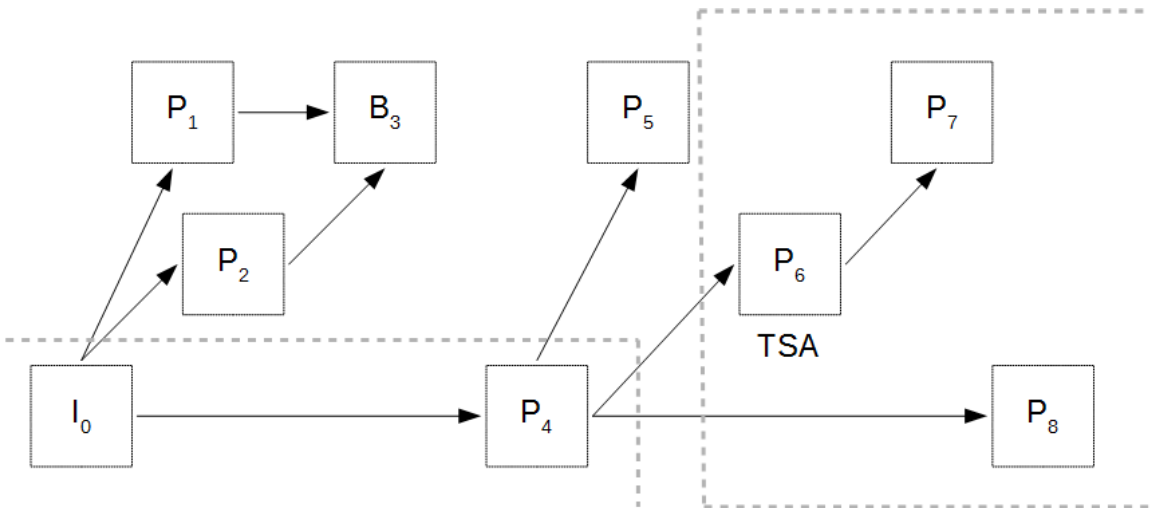

Figure 2.4 - TSA example.

A TSA picture is a trailing picture that indicates a temporal sub-layer switching point. It can only be used for a picture if it is guaranteed that no picture that precedes the TSA picture in decoding order with a temporal ID that is greater than or equal to the temporal ID of the TSA picture itself is used for prediction of the TSA picture or any subsequent pictures in the same or higher temporal sub-layer as the TSA picture [25]. For example, picture P_6 in Figure 2.4 can use the TSA picture type since only previous pictures in temporal sub-layer 0 are used for prediction of the TSA picture itself and subsequent pictures in decoding order.

The STSA picture type is similar to the TSA picture type, but it only guarantees that the STSA picture itself and pictures of the same temporal ID as the STSA picture that follow it in decoding order do not reference pictures of the same temporal ID that precede the STSA picture in decoding order. The STSA pictures can therefore be used to mark positions in the bitstream where it is possible to switch to the sub-layer with the same temporal ID as the STSA picture [24].

One example of an STSA picture in Figure 2.4 is picture P_2. This picture cannot be a TSA picture since B_3 references P_1. However, picture P_2 can be an STSA picture because P_2 does not reference any picture of sub-layer 1, nor does any sub-layer 1 picture that follows P_2 in decoding order reference any sub-layer 1 picture that precedes P_2 in decoding order.

TSA and STSA pictures must have a temporal ID higher than 0.

2.2.1.2.2 Random Access Decodable Leading (RADL) and Random Access Skipped Leading (RASL) pictures

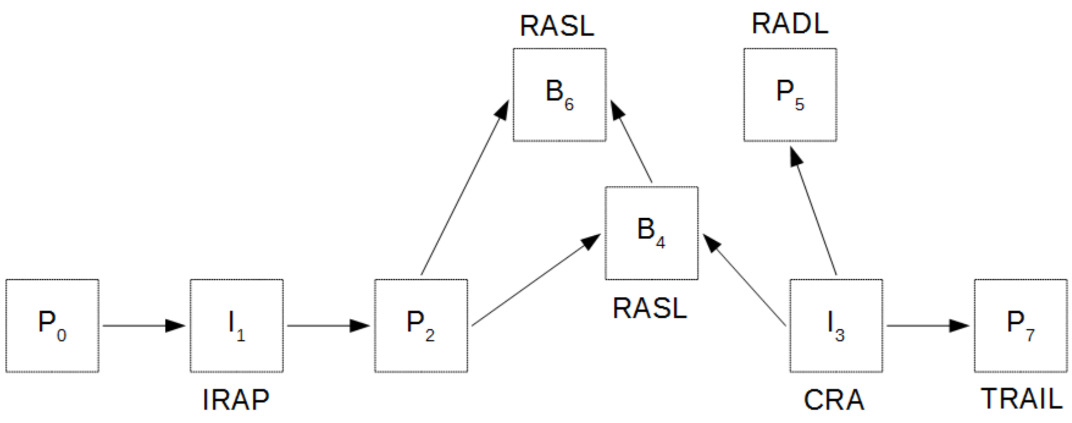

Figure 2.5 - RADL and RASL pictures.

Leading pictures shall be signaled using either the RADL or RASL NAL unit type. RADL and RASL pictures can belong to any temporal sub-layer, but they cannot be referenced by any trailing picture.

A RADL picture is a leading picture that can be decodable when a random access is performed at the related IRAP picture. Consequently, RADL pictures can only reference the related IRAP picture and other RADL pictures of the same IRAP picture.

A RASL picture is a leading picture that may not be decodable when random access is performed from the related IRAP picture. Figure 2.5 shows two RASL pictures which are both non decodable since pictures P_2 precedes the CRA picture in decoding order. Because of its position in decoding order, a decoder that performs random access at the position of the CRA picture cannot decode the P_2 picture, and therefore cannot decode these RASL pictures and will discard them. The use of RASL type for decodable leading pictures is not forbidden, such as the RADL picture in Figure 2.5, but is recommended to use RADL type when possible to be more network friendly [25]. Only other RASL pictures are allowed to be dependent on a RSAL picture, which means that any picture who depend on a RASL picture must also be a RASL picture. RADL and RASL pictures can be mixed in decoding order, but not in output order, since RASL pictures need to precede RADL pictures in output order.

2.2.1.2.3 Broken Link Access (BLA), Instantaneous Decoding Refresh (IDR) and Clean Random Access (CRA) pictures

When a part of a bitstream starting from a CRA picture is included in another bitstream, the RASL pictures associated with the CRA picture are not present in the combined bitstream. To make such splicing operation straightforward, the NAL unit type of the CRA picture can be altered to indicate that is a BLA picture. The RASL picture associated with a BLA picture are not properly decodable, therefore should not be outputted.

The IDR picture is an intra picture that thoroughly refresh the decoding process and starts a new coding video sequence (CVS). The presence of an IDR picture indicates that no subsequent picture in the bitstream will require reference to pictures prior to the picture that it contains in order to be decoded.

A CRA picture is an intra picture that, in contrast to an IDR picture, does not refresh the decoder and does not begin a new CVS. This allow leading pictures of the CRA picture to depend upon pictures that precede the CRA picture in decoding order. Consequently, bitstream relying on CRA pictures for random access normally have an increase of coding efficiency.

2.2.2 Parameter Sets

Parameter sets in HEVC are an inheritance from the concept used in H.264/AVC [27] with some modifications and additions. The introduction of parameters sets in H.264/AVC was a response to the problematic effects of a loss of the sequence header and picture header. The loss of the first packet of a picture, which carries not only the first picture segment data but also the picture header, can lead to a defective reconstructed picture of current and following pictures. During the design of H.264/AVC, it was concluded that the vulnerability of a picture header wasn't a transport problem, but rather an architectural issue of the video codec, and therefore the parameter set concept was introduced to solve this issue [24].

2.2.2.1 Video Parameter Set (VPS)

The VPS is a new parameter set introduced in HEVC to convey information that is applicable to multiple layers as well as sub-layers. H.264/AVC did not contain a similar parameter set, which led to complexity and overhead for H.264/AVC scalable video coding and multiview video coding extensions. The introduction of the VPS fixed these shortcoming and allowed a clean and extensible high-level design of multilayer codecs. The VPS transmit information includes:

1) common syntax elements shared by multiple layers or operations points, in order to avoid unnecessary duplications;
2) essential information of operation points needed for session negotiation, including, e.g., profile and level;
3) other operation point specific information, which doesn't belong to one SPS, e.g., hypothetical reference decoder (HRD) parameters for layers or sub-layers;

The decomposition of essential information of each operation point does not required a variable length coding, which for networking elements allow a decrease in the workload.Some information between the VPS and the SPS belonging to the (same) layer can be duplicated, which allow a version 1 decoder to disregard the VPS NAL unit and still have available all information required to decode the bitstream.

2.2.2.2 Sequence Parameter Set (SPS)

The SPS contain information which applies to an entire coded video sequence. All pictures in the same CVS shall use the same SPS. The SPS content can be subdivided into six categories:

1) a self-reference (its own ID);
2) decoder operation point related information, e.g., profile, picture size, number sub-layers;
3) enabling flags for certain tools within a profile, and associated coding tool parameters in case the tool is enabled;
4) information restricting the flexibility of structures and transform coefficient coding;
5) temporal scalability control (similar to H.264/SVC);
6) visual usability information (VUI), which includes HRD information;

2.2.2.3 Picture Parameter Set (PPS)

The PPS contains information that may change from picture to picture, however, multiple pictures may refer to the same PPS, even if they have different slice coding types (I, P and B).
The PPS includes information comparable to what was part of the PPS in H.264/AVC, including:

1) a self-reference;
2) initial picture control information such as initial quantization parameter (QP), a number of flags indicating the use of, or presence of, certain tools or control information in the slice header;
3) tiling information;

2.2.2.4 Slice Header

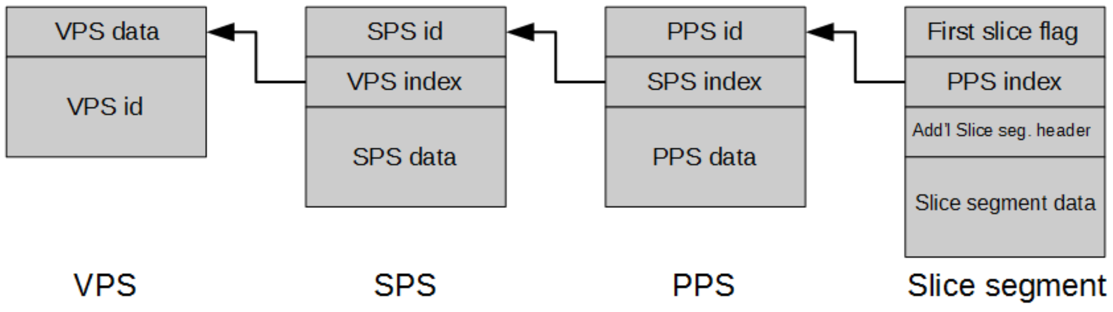

Figure 2.6 - Parameter set referencing hierarchy.

The slice header contains information that can change from slice to slice, and also contain picture related information that is only relevant for a certain slice or picture. The size of the slice header may be larger than the PPS, mainly when there are tile or wavefront entry point offset in the slice header and RPS, prediction weights or reference picture list modification are explicitly signaled [26]. Activation of parameter sets resembles the H.264/AVC [24]. To identify for a given slice the active parameter set at each level of the parameter set type hierarchy, each slice header contains a PPS identifier which references a particular PPS. The PPS, in turn, contain an identifier that references a particular SPS and the SPS contain an identifier that references a particular VPS. Figure 2.6 illustrates this referencing hierarchy.

The parameter sets of a given type (PPS, SPS and VPS) are kept in tables, whose maximum size is specified by the numbering range of the parameter set Ids. This implementation strategy permit that the parameter set activation can be as simple as accessing the PPS tables based on information in the slice header, copying the information found into the relevant decoder data structure, and following the reference in the SPS to the relevant VPS. This operation is lightweight since, in the worst case, it only need to be done once per picture. Parameter set NAL units do not contain parsing dependencies, since they are self-contained and don't require context derived from another NAL unit for parsing. Even though this cost a few more bits, the handling of the reception of a parameter set NAL unit, regardless of its type, is straightforward.

Each type of parameter set contains an extension mechanism, which permit extending the parameter set with backward compatibility and without creating a parsing dependency to the profile/level information carried in the VPS and SPS in futures versions of HEVC.

2.2.3 Reference Picture Set (RPS)

The RPS concept for reference picture management is relatively distinct from the reference picture management of its precedents. Instead of signaling relative changes to the decoded picture buffer (DPB), the status of the DPB is signaled in each slice.

The reconstructed pictures are managed in the DPB in order to be used for reference. Pictures in the DPB can be tagged as "used for short-term reference", "used for long-term reference" or "unused for reference". When a picture is tagged as "unused for reference" it can't be used anymore for prediction, and when it isn't needed anymore for output it can be removed from the DPB [24].

For each particular slice, a complete set of the reference pictures that are used by the current picture or any subsequent picture must be provided. Therefore, a complete set of all pictures must be kept in the DPB for use by the current or future picture is signaled. With the RPS concept, no information from earlier pictures in decoding order is needed to maintain the correct status of reference pictures in the DPB [28].

Table 2.3 - RPS example.

Picture	**RPS** (reference picture used by current picture)
I_0	-
P_1	$\{I_0,1\}$
B_2	$\{I_0,1\}, \{P_1,1\}$
B_3	$\{I_0,1\}, \{P_1,1\}, \{B_2,1\}$
B_4	$\{P_1,1\}, \{B_2,1\}$

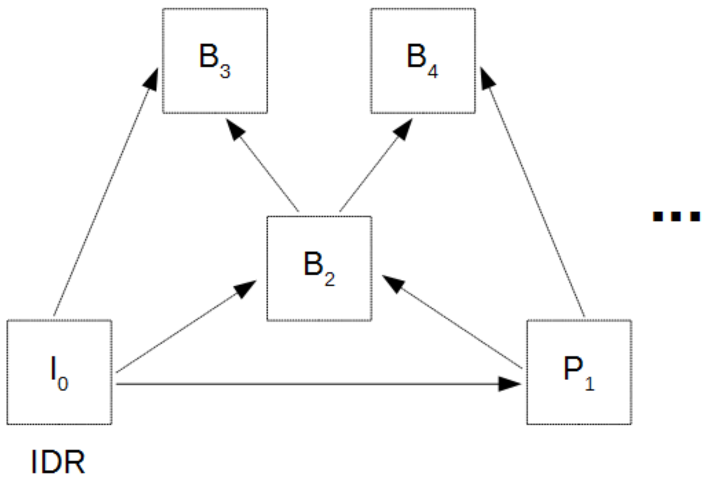

Figure 2.7 - Coding structure for RPS example.

Figure 2.7 shows an example of a coding structure, where the RPS for its pictures are presented in Table 2.3. The first picture in decoding order is an IDR picture, I_0, where no RPS is signaled since an IDR picture resets the codec which includes turning all pictures in the DPB into non-reference pictures. Since the RPS of an IDR picture is empty, there is no RPS syntax signaled for IDR pictures. The second picture in decoding order, P_1, uses I_0 for reference. It shall therefore include I_0 in its RPS. Picture B_2 uses both I_0 and P_1 for reference so they both are included in the RPS of B_2. Picture B_3 uses I_0 and B_2 for reference so they are included in the RPS of B_3. But also P_1 must be included since this picture will be used for reference for future pictures. Finally, picture B_4 will use B_2 and P_1 for prediction. Since the RPS of B_4 does not list I_0, I_0 will be marked as "unused for reference" and therefore I_0 cannot be used for reference by B_4 or by any picture that follows B_4 in decoding order.

2.2.4 Supplemental Enhancement Information (SEI)

SEI message mechanism enables the encoder to provide metadata that can be used for various purposes, such as picture output timing and displaying, as well as loss detection and concealment. SEI messages are an option in the bitstream, and does not interfere with the correct decoding. SEI messages are carried within the SEI NAL units. In H.264/AVC, all SEI messages are considered to be a prefix type, which would require that the SEI message have to precede all VCL NAL units of an access unit. HEVC introduces the concept of a suffix SEI message, which follows a VCL NAL unit of an access unit. Table 2.4 list all the HEVC SEI messages, indicating whether they are prefix or suffix type and a brief description. The motive to use SEI messages in HEVC is to enable that supplemental data is interpreted equally in different systems that make use of HEVC.

Table 2.4 - HEVC SEI message.

SEI message	Type	Description
Buffering period	Prefix	Provides parameters for HRD initialization
Picture timing	Prefix	Provides HRD parameters and interlaced picture indication
Pan scan rectangle	Prefix	Provides conformance cropping window parameters, to indicate when output pictures are smaller than decoded pictures
Filler payload	Prefix/Suffix	Carries unused data, to enable encoder to achieve desired bit rate
User data registered	Prefix/Suffix	Carries user-specific data, with type registered through registration authority
User data unregistered	Prefix/Suffix	Carries user-specific data, not registered
Recovery point	Prefix	Indicates first picture with acceptable quality after non-IRAP random access
Scene info	Prefix	Description of scene and scene transition
Picture snapshot	Prefix	Indicates picture intended for use as still-image snapshot of the video
Progressive refinement segment start	Prefix	Indicates start of a sequence of coded pictures to progressively improve quality of a particular picture
Progressive refinement segment end	Prefix	Indicates end of a sequence of coded pictures to progressively improve quality of a particular picture
Film grain characteristics	Prefix	Describes a parameterized model for film grain synthesis
Post filter hint	Prefix/Suffix	Provides coefficients of a post filter
Tone mapping info	Prefix	Provides remapping information of the colour samples of the output pictures for customization to particular display environments
Frame packing arrangement	Prefix	Indicates that the output picture contains multiple distinct spatially packed frames, and the particular arrangement used
Display orientation	Prefix	Indicates to decoder to rotate or flip the output picture prior to display
Structure of pictures info	Prefix	Provides series pattern information of coded picture types
Decoded picture hash	Suffix	Provides a hash for each colour component of the decoded picture, to assist decoder to detect mismatch with encoder
Active parameter sets	Prefix	Indicates the active VPS and SPS
Decoding unit info	Prefix	Provides HRD parameters for sub-AU decoding units
Temporal sub layer zero index	Prefix	Provides information to assist decoder to detect missing coded pictures
Scalable nesting	Prefix	Associates other SEI messages with bitstream subsets
Region refresh info	Prefix	Indicates if slice segments belong to a refreshed region of the picture
Reserved	Prefix/Suffix	For future extensions

2.3 Block Partitioning Structure in HEVC

HEVC differentiate between blocks and units. While the blocks address a particular area in a sample array (e.g. luma Y), units include the collocated blocks of all encoded color components (Y, Cb, Cr) as well as all syntax elements and prediction data that is associated to the blocks (e.g. motion vectors).

The base entities are the Coding Tree Block (CTB) and the corresponding Coding Tree Unit (CTU). The CTU contains the CTBs of the color components and forms a complete entity in the bitstream syntax. A CTB is the root of a quadtree partitioning into Coding Blocks (CB). A CB is partitioned into one or more Prediction Blocks (PB) and forms the root of a quadtree partitioning into Transform Blocks (TB). A corresponding set of units is specified which comprise the block and the respective syntax structure, each. Accordingly, a Coding Unit (CU) contains the Prediction Units (PU) and the tree-structure set of Transform Unit (TU). While the PU contains the joint prediction information for all color components, a TU contains a separate residual syntax for each color component. The location and size of the CBs, PBs and TBs of luma component applies to the corresponding CU, PU and TU [26] [29].

2.3.1 Coding Tree Unit

A CTU represents the basic processing unit in HEVC and is in that regard similar to the concept of a macroblock from previous video coding standards. The CTU consist of a luma CTB and the corresponding chroma CTBs and syntax elements. In main profile, the minimum and maximum sizes of the CTU are specified in the SPS among the sizes of 8x8, 16x16, 32x32 and 64x64. The support of large sizes up to 64x64 provides a better coding efficiency, but increase the encoder/decoder delay, the memory requirements, and the computational complexity of the encoder process.

2.3.2 Coding Unit

A CTU can be split into multiple coding units (CU) of variable sizes to adapt to various local characteristics. For that reason, each CTU contains a quadtree syntax, which specifies its subdivision into CUs. Similarly to a CTU, a CU consist of a square block of luma samples, the two corresponding blocks of chroma samples, and the syntax associated with these sample blocks. The luma and chroma sample arrays that are contained in a CU are referred to as coding blocks (CB).

Let the size of the CTU be 2Nx2N, where N is one of the values of 32, 16 or 8, the CTU can be a single CU or can be split into four smaller units of equal sizes of NxN, which are nodes of coding tree. If the units are leaf nodes of coding tree, the units become CUs, otherwise it can be split once more into four smaller units when the split size is equal or larger than the minimum CU size that is specified in the SPS. This rendering results in a recursive structure specified by a coding tree.

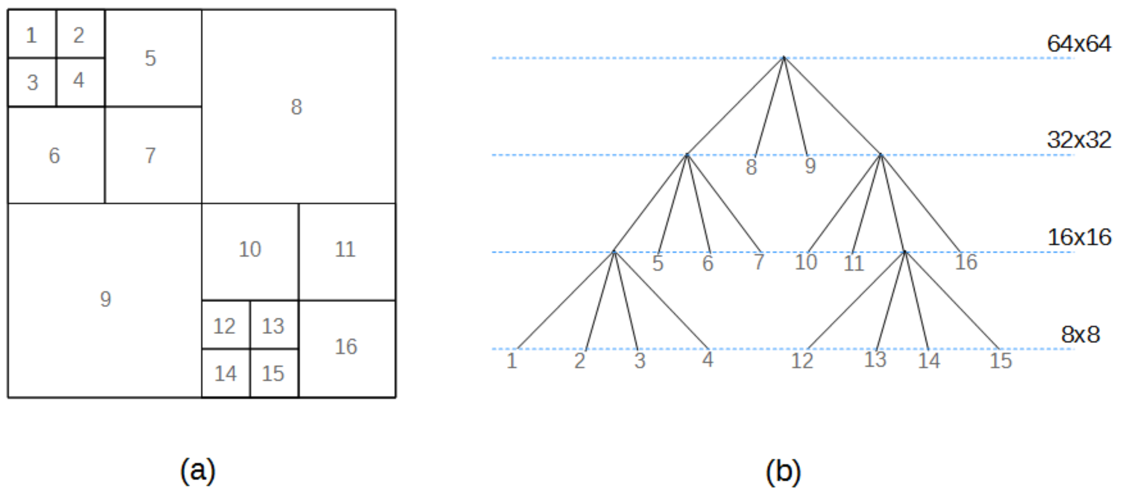

Figure 2.8 - Example of CTU partitioning and processing order. (a) CTU partitioning. (b) Corresponding coding tree structure.

Figure 2.8 exemplifies the CTU partitioning and the processing order of CUs when the size of CTU is 64x64 and the minimum CU size is 8x8. Figure 2.8 (a) represent the CTU being split into 16 CUs with different sizes and positions and Figure 2.8 (b) the corresponding coding tree structure of the CTU partitioning. This processing order of CUs can be interpreted as a depth-first traversing in the coding tree structure [30].

This flexible and recursive process offer several benefits. The first benefit is the support of CU sizes greater than the conventional 16x16 size, i.e., a large CU can represent a homogeneous region by using a smaller number of symbols than the case of using several small blocks. Another benefit is the support of arbitrary sizes of CTU which enables the codec to be promptly optimized for various content applications and devices. By choosing an appropriate size of CTU and maximum hierarchical depth, the hierarchical block partitioning structure can be optimized to the target application. Finally, by eliminating the distinction between macroblock and sub-macroblock and using only CU, the multilevel hierarchical quadtree structure can be detailed in a very simple way.

2.3.3 Prediction Unit

One or more PUs are specified for each CU, which is a leaf node of coding tree. Attached to each CU, the PU works as a basic representative block that shares the prediction information. A CU can be split into one, two or four PUs according to the PU splitting type. The PU can only be split once. HEVC define two splitting shapes for the intra coded CU and eight splitting shapes for inter coded CU.

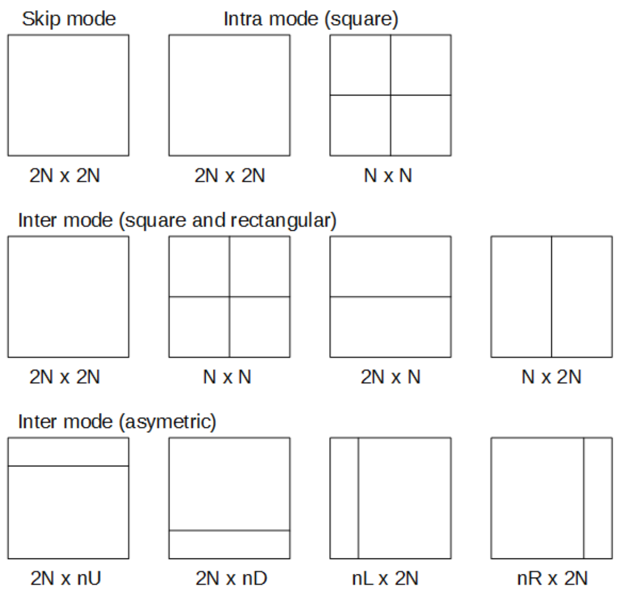

Figure 2.9 - Illustration of PU splitting types in HEVC.

Each CU in HEVC can be classified into three categories: skipped CU, inter coded CU and intra coded CU. An inter coded CU uses motion compensation scheme for the prediction of the current block, while an intra coded CU uses neighboring reconstructed samples for the prediction. A skipped CU is a special form of inter coded CU where both the motion vector difference and the residual energy are equal to zero. For each category, PU splitting type is specified differently as shown in Figure 2.9. Some PU splitting types are restricted to specific situations (PART_NxN and asymmetric shapes), since in some cases they can lead to an increase in the complexity.

2.3.4 Transform Unit

Similar with the PU, one or more transform units (TUs) are specified for the CU. HEVC permit a residual block to be split into multiple units recursively to form another quad-tree which is similar to the coding tree for the CU. The TU is a representative block that have residual or transform coefficients for applying the integer transform and quantization. For each TU, one integer transform that have the same size of the TU is applied to obtain residual coefficients, and in turn these coefficients are transmitted to the decoder after quantization on a TU basis.

After obtaining the residual block by prediction process based on PU splitting type, it is split into multiple TUs according to a quad-tree structure called residual quad-tree (RQT). For each TU, an integer transform is applied for each leaf node of the quad-tree. Similar to the coding tree, RQT is also structure by successive signaling of the syntax element, split_transform_flag, in a recursive mode [29].

When the TU size is equal to the CU size, the transform is applied to the residual block covering the whole CU regardless of the PU splitting type (this case exist only for inter coded CU).

The maximum depth of transform tree is closely related to the encoding complexity. To provide the flexibility on this feature, HEVC specifies two syntax elements in the SPS which control the maximum depth of transform tree for intra coded CU and inter coded CU, respectively.

2.3.5 Slice Partitioning

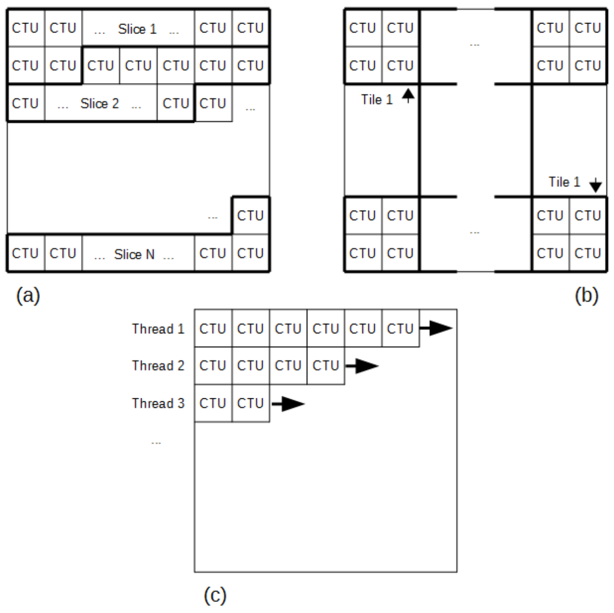

Figure 2.10 - Subdivision of a picture into (a) slices and (b) tiles. (c) Illustration of wavefront parallel processing.

A slice is a data structure of sequences of CTUs that is independent from other slices of the same picture, in other words, a slice is independently decoded in terms of entropy, predictive and residual coding. A picture may be split into one or several slices, as shown in Figure 2.10 (a), being each slice comprised of payload data and a slice header which contains information to decode the slice data, such as the address of the first CTU in the slice or slice type. Each slice can be coded using different coding types as follows:

1. I slices CUs are coded using only intrapicture prediction;
2. P slice CUs can also be coded using interpicture prediction with at most one motion-compensated prediction signal per PB;
3. B slice CUs can also be coded using interpicture prediction with at most two motion-compensated prediction signals per PB.

The concept of slices root from different motivations. The most important being packetization. Normally, during transport, a packet comprises a single slice, which means the loss of a packet translates into the loss of a single slice. The ability to partition a picture into several smaller slices allows for increased error robustness and resynchronization in the likely event of packet losses. Restricting the maximum bit size of a slice also permits compliance to IP networks maximum transmit unit (MTU) which limits the payload data per packet. Lastly, the usage of independently decoded slices enables parallel processing of those slices reducing encoding/decoding times greatly [26] [23].

HEVC also defines tiles, which are self-contained and independently decodable rectangular regions of the picture. The main purpose of tiles is to support the use of parallel processing architectures for encoding and decoding. The header information, by being contained in the same slice, can be shared between multiples tiles. On the other hand, a single tile may contain multiple slices. A tile consist of a rectangular arranged group of CTUs (typically containing about the same number of CTUs) as shown in Figure 2.10 (b).

Finally, with wavefront parallel processing (WPP) a slice is divided into rows of CTUs, where each of this rows can represent a different thread for parallel processing. In order to deduce the entropy context models for posterior rows, a delay of two CTUs is imposed between threads. WPP provides, in certain cases, better compression results than tiles avoiding some artifacts introduced by the use of tiles. An example is shown in Figure 2.10 (c). WPP is not allowed to be used in combination with tile.

2.4 Intrapicture Prediction

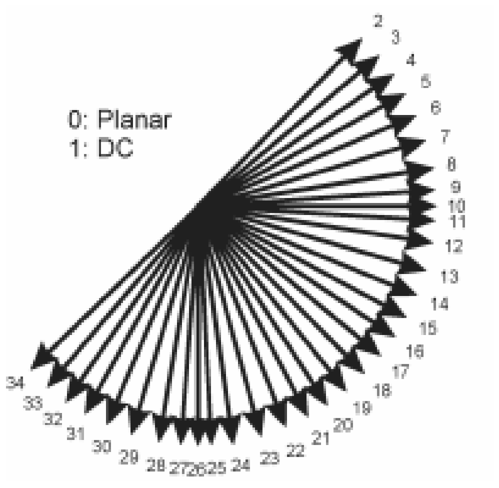

Figure 2.11 - Modes and directional orientations for intrapicture prediction [23].

Intrapicture prediction operates according to the TB size, and the prediction signal is formed using the previously decoded boundary samples from spatially neighboring TBs. There are four effective intra prediction block sizes with sizes from 4x4 up to 32x32, and each supporting 33 distinct prediction directions. The possible prediction directions are shown in Figure 2.11. Planar prediction and DC prediction can also be used Intrapicture predictive coding can uphold all slice types and support various coding methods referred as intra angular, intra planar and intra DC [31].

2.4.1 Intra Angular Prediction

The intrapicture prediction of HEVC works in the spatial domain. Compared with the eight prediction directions of AVC, HEVC supports 33 prediction directions, designated as Intra_Angular[k], where k is a mode number from 2 up to 34 [32]. The intra prediction process is performed by extrapolating samples from the projected reference sample location according to a given directionality. In order to simplify the need for sample-by-sample switching between reference row and column buffers, all sample locations within one prediction block are projected to a single reference row or column depending on the directionality of the selected prediction mode. For angular modes with range between 2 and 17, the samples located in the above row are projected as additional samples located in the left column. For angular mode with range between 18 and 34, the samples located at the left column are projected as samples located in the above row. The prediction process of the intra angular modes in HEVC is uniform across all block sizes and prediction directions, whereas in AVC, block sizes of different sizes use different methods.

2.4.2 Intra Planar and Intra DC Prediction

Besides the intra angular prediction, HEVC support two alternative prediction methods, intra planar and intra DC prediction (similar to same modes used in AVC). While intra DC prediction uses an average value of reference samples for the prediction, intra planar prediction uses average values of two linear predictions using four corner reference to prevent discontinuities along the block boundaries. Also, intra planar prediction mode is supported with all block sizes.

2.4.3 Reference Sample Smoothing

The reference samples used for intrapicture prediction are sometimes filtered by a three-tap [1 2 1]/4 smoothing filter, similar to what is used for 8x8 intrapicture prediction in AVC. For 32x32 blocks, all angular modes except horizontal (mode 10) and vertical (mode 26) use a filtered reference. In 16x16 blocks, the reference samples are filtered for most directions except the near-horizontal (modes 9, 10, 11) and near-vertical (modes 25, 26, 27) directions. The intra planar mode also uses the smoothing filter when the block size is 8x8 or larger.

DC and angular prediction modes 10 or 26 (exactly horizontal or exactly vertical) may introduce discontinuities along the block boundaries. To remove this problem, the first prediction row and column are filtered in the case of DC prediction with a two-tap finite response filter. Similarly, the first prediction column for exactly vertical prediction and the first prediction row for exactly horizontal prediction are filtered using a gradient-based smoothing.

2.4.4 Mode Coding

Due to the large number of intra prediction modes, HEVC defines three most probable modes (MPM) when coding the luma intrapicture prediction [25]. When the first two MPM are not equal, the third MPM is set to planar mode, DC mode or angular mode 26, according to which of these modes, in this order, is not a duplicate of one of the first two modes. When the first two MPM are the same, and they are not angular modes, the three MPM are set equal to planar mode, DC mode and angular mode 26, respectively. When the first two MPM are the same and the first mode has an angular mode value, the second and third MPM modes are chosen as the two angular prediction modes that are closest to the angle of the first. Figure 2.12 summarizes the derivation process for the three MPM.

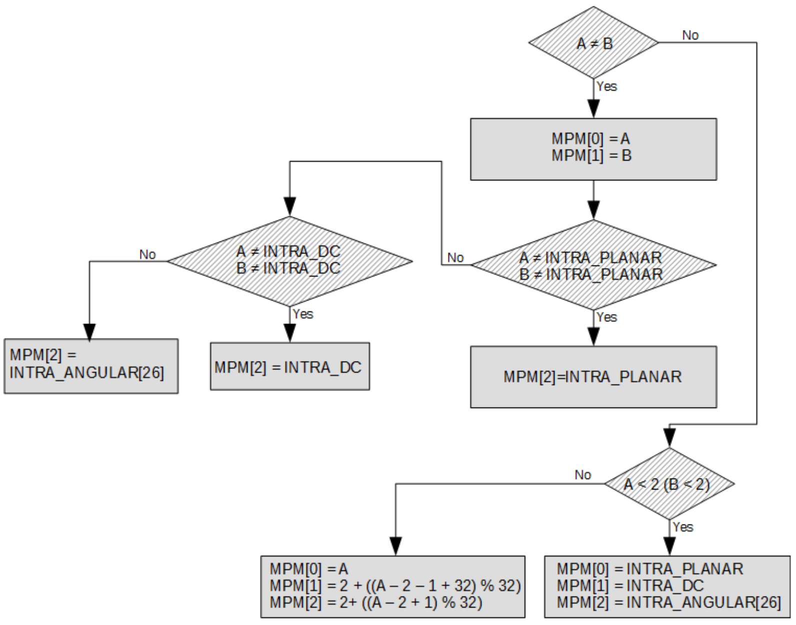

Figure 2.12 - Derivation process for the three most probable modes.

For chroma intrapicture prediction, HEVC allows the encoder to select one of five modes: planar mode, angular mode 26, angular mode 10, DC mode and derived mode. The derived mode specifies that the chroma prediction uses the same angular direction as the luma prediction.

2.5 Interpicture Prediction

HEVC does not modify the already interpicture prediction design from previous video coding standards. Nevertheless, HEVC made improvements on all the mechanics of the inter prediction process. The motion vector prediction was enhanced with advanced motion vector prediction (AMVP) that infers the most probable candidates based on data from neighboring PBs and allows the inheritance of motion vectors from adjacent spatial or temporal PBs. The inter prediction block merging technique was simplified by inferring all motion data from already decoded blocks. Quarter sample precision is used for the motion vectors, and 7/8-tap filters are used for luma interpolation and 4-tap filters for chroma improving filtering for high frequencies. Weighted prediction was also simplified [25] [26].

2.5.1 Fractional Sample Interpolation

The samples of the PB for an intrapicture predicted CB are acquire from those of a corresponding block region in the reference picture index, which is at a position displaced by the horizontal and vertical components of the motion vector. HEVC support motion vector with quarter-pixel accuracy for the luma samples. For chroma samples, the motion vector accuracy is determined according to the chroma sampling format, which for 4:2:0 sampling results in one-eighth pixel accuracy for the chroma samples.

The luma fractional sample interpolation in HEVC uses a symmetric 8-tap filter for the half-sample positions and a 7-tap filter for the quarter-sample positions. HEVC uses a single consistent separable interpolation process for generating all fractional positions without intermediate rounding operations, which improves accuracy and simplifies the architecture of the fractional sample interpolation. The filter tap values of the interpolation filtering were partially derived from DCT basis function equations.

In Figure 2.13, the positions with upper-case letters, Ai,j, express the available luma samples at integer sample locations, whereas the positions with lower-case letters express samples at non-integer sample locations, which need to be generated by interpolation.

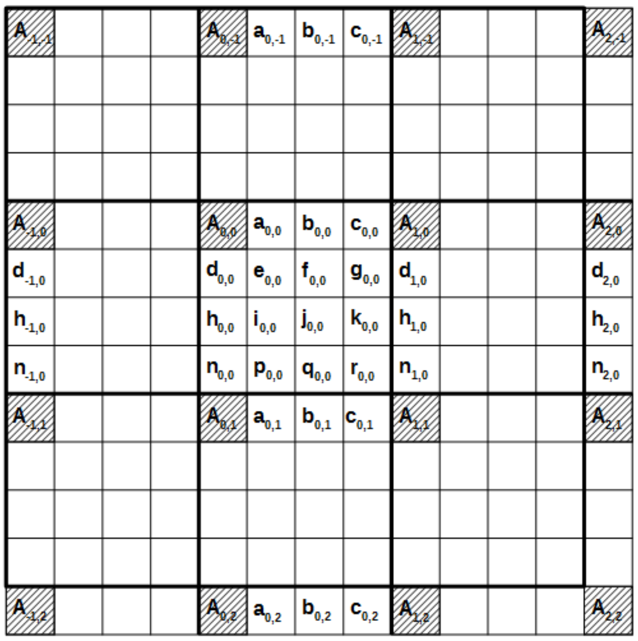
$A_{-1,-1}$ $A_{0,-1}$ $a_{0,-1}$ $b_{0,-1}$ $c_{0,-1}$ $A_{1,-1}$ $A_{2,-1}$
$A_{-1,0}$ $A_{0,0}$ $a_{0,0}$ $b_{0,0}$ $c_{0,0}$ $A_{1,0}$ $A_{2,0}$
$d_{-1,0}$ $d_{0,0}$ $e_{0,0}$ $f_{0,0}$ $g_{0,0}$ $d_{1,0}$ $d_{2,0}$
$h_{-1,0}$ $h_{0,0}$ $i_{0,0}$ $j_{0,0}$ $k_{0,0}$ $h_{1,0}$ $h_{2,0}$
$n_{-1,0}$ $n_{0,0}$ $p_{0,0}$ $q_{0,0}$ $r_{0,0}$ $n_{1,0}$ $n_{2,0}$
$A_{-1,1}$ $A_{0,1}$ $a_{0,1}$ $b_{0,1}$ $c_{0,1}$ $A_{1,1}$ $A_{2,1}$
$A_{-1,2}$ $A_{0,2}$ $a_{0,2}$ $b_{0,2}$ $c_{0,2}$ $A_{1,2}$ $A_{2,2}$

Figure 2.13 - Integer and fractional sample positions for luma interpolation.

The samples labelled $a_{0,j}$, $b_{0,j}$, $d_{0,0}$, $h_{0,0}$, and $n_{0,0}$ are generated from the samples $A_{i,j}$ by applying the 8-tap filter for half-sample positions and the 7-tap filter for the quarter-sample positions as follows:

$$a_{0,j} = \left(\sum\nolimits_{i=-3..3} A_{i,j}\, qfilter[i]\right) \gg (B-8)$$
$$b_{0,j} = \left(\sum\nolimits_{i=-3..4} A_{i,j}\, hfilter[i]\right) \gg (B-8)$$
$$c_{0,j} = \left(\sum\nolimits_{i=-2..4} A_{i,j}\, qfilter[1-i]\right) \gg (B-8)$$
$$d_{0,0} = \left(\sum\nolimits_{i=-3..3} A_{i,j}\, qfilter[j]\right) \gg (B-8)$$
$$h_{0,0} = \left(\sum\nolimits_{i=-3..4} A_{i,j}\, hfilter[j]\right) \gg (B-8)$$
$$n_{0,0} = \left(\sum\nolimits_{i=-2..4} A_{i,j}\, qfilter[1-j]\right) \gg (B-8)$$

the constant $B \geq 8$ represent the bit depth of the reference samples, and the filter coefficient values are specified in Table 2.5. In these formulas >> denotes an arithmetic right shift operation.

Table 2.5 - Filter coefficients for luma fractional sample interpolation.

Index i	-3	-2	-1	0	1	2	3	4
hfilter[i]	-1	4	-11	40	40	-11	4	1
qfilter[i]	-1	4	-10	58	17	-5	1	0

The samples labelled $e_{0,0}$, $f_{0,0}$, $g_{0,0}$, $i_{0,0}$, $j_{0,0}$, $k_{0,0}$, $p_{0,0}$, $q_{0,0}$, and $r_{0,0}$ can be generated by applying the corresponding filters to samples located at vertically adjacent $a_{0,j}$, $b_{0,j}$, and $c_{0,j}$, positions as follows:

$$e_{0,0} = \left(\sum\nolimits_{v=-3..3} a_{0,v}\, qfilter[v]\right) \gg 6$$
$$f_{0,0} = \left(\sum\nolimits_{i=-3..3} b_{0,v}\, qfilter[v]\right) \gg 6$$
$$g_{0,0} = \left(\sum\nolimits_{i=-3..3} c_{0,v}\, qfilter[v]\right) \gg 6$$
$$i_{0,0} = \left(\sum\nolimits_{i=-3..4} a_{0,v}\, hfilter[v]\right) \gg 6$$
$$j_{0,0} = \left(\sum\nolimits_{i=-3..4} b_{0,v}\, hfilter[v]\right) \gg 6$$
$$k_{0,0} = \left(\sum\nolimits_{i=-3..4} c_{0,v}\, hfilter[v]\right) \gg 6$$
$$p_{0,0} = \left(\sum\nolimits_{i=-2..4} a_{0,v}\, qfilter[1-v]\right) \gg 6$$
$$q_{0,0} = \left(\sum\nolimits_{i=-2..4} b_{0,v}\, qfilter[1-v]\right) \gg 6$$
$$r_{0,0} = \left(\sum\nolimits_{i=-2..4} c_{0,v}\, qfilter[1-v]\right) \gg 6$$

the interpolation filtering is separable when B is equal to 8, so the same values could be computed in this case by applying the vertical filtering before the horizontal filtering.

It is at this point in the process that weighted prediction is applied when selected by the encoder. HEVC only applies explicit weighted prediction by scaling and offsetting the prediction with values sent explicitly by the encoder. The bit depth of the prediction is then adjusted to the original bit depth of the reference samples. In case of uniprediction, the interpolated prediction value is rounded, right-shifted and clipped to have the original bit depth. In the case of biprediction, the interpolated prediction values from two PBs are added first, then rounded, right-shifted and clipped.

The fractional sample interpolation process for the chroma components is similar to the one for the luma component, except that it uses a 4-tap filter with the fractional accuracy depending on the chroma format [25] [26] [23].

2.5.2 Merge Mode

Motion information typically consist of one or two reference pictures indices, the horizontal and vertical motion vector displacement values and an identification of which reference picture list is associated with each index in the case of prediction regions in B slices. HEVC includes a merge mode to derive the motion information from spatially or temporally neighboring blocks.

The merge mode is conceptually analogous to the direct and skip modes in AVC, but with two important differences. First, it transmits index information to pick one out of several available candidates, in a way referred to as motion vector competition scheme. It also explicitly identifies the reference picture list and reference picture index, while the direct mode assumes that these have some predefined values.

The set of possible candidates in the merge mode consist of spatial neighbor candidates, temporal candidate and generated candidates. Figure 2.14 illustrate the positions of five spatial candidates. For each candidate position, the availability is checked according to the order {a_1, b_1, b_0, a_0, b_2}. If the block located at the position is intrapicture predicted or the position is outside of the current slice or tile, it is considered as unavailable.

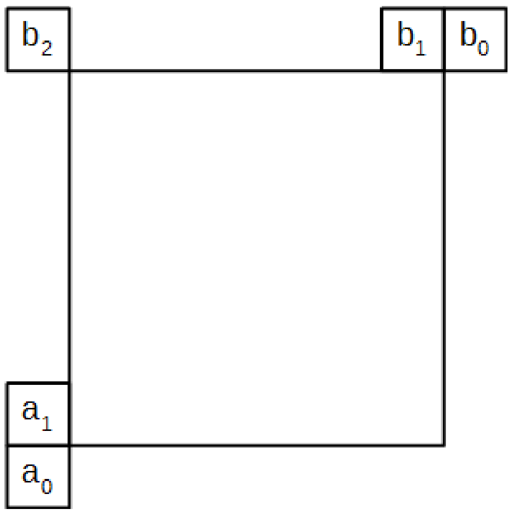

Figure 2.14 - Positions of spatial candidates of motion information.

After validating the spatial candidates, if the candidate position for the current PU would refer to the first within the same CU, the position is excluded, as the same merge could be achieved by a CU splitting into prediction partitions. Moreover, any redundant entries where candidates have exactly the same motion information are also excluded.

For the temporal candidate, the right bottom position just outside of the collocated PU of the reference picture is used if it is available, if not the center position is used instead.

The maximum number of merge candidates C is specified in the slice header. If the number of merge candidates found is larger than C, only the first C-1 spatial candidates and the temporal candidate are retained. If the number of merge candidates identified is less than C, additional candidates are generated until the number is equal to C.

For B slices, additional merge candidates are generated by choosing two existing candidates according to a predefined order for reference picture list 0 and list 1. For example, the first generated candidate uses the first merge candidate for list 0 and the second merge candidate for list 1.

When the slice is a P slice or the number of merge candidates is still less than C, zero motion vectors associated with reference indices from zero to the number of reference pictures minus one are used to fill any remaining entries in the merge candidate list.

In HEVC, the skip mode is treated as a special case of the merge mode when all coded block flags are equal to zero. In this specific case, only a skip flag and the corresponding merge index are transmitted to the decoded [25] [23].

2.5.3 Advanced Motion Vector Prediction (AMVP) for Nonmerge Mode

When an interpicture-predicted CB is not coded in the skip or merge modes, the motion vector is differentially coded using a motion vector predictor. Similar to the merge mode, HEVC allows the encoder to choose the motion vector predictor among multiple predictor candidates.

Only two spatial motion candidates are chosen according to the availability among five candidates in Figure 2.14. The first spatial motion candidate is chosen from the set of left position {a_0, a_1} and the second one from the set of above positions [b_0, b_1, b_2} according to their availabilities, while keeping the searching order as indicated in the two sets.

HEVC only allows a much lower number of candidates to be used in the motion vector prediction process for the non-merge case. Besides, the encoder need to perform motion estimation, which is one of the most computationally expensive operations in the encoder, and by allowing a small number of candidates reduce the complexity.

When the reference index of the neighboring PU is not equal to that of the current PU, a scaled version of the motion vector is used. The neighboring motion vector is scaled according to the temporal distances between the current picture and the reference pictures indicated by the reference indices of the neighboring PU and the current PU. When two spatial candidates have the same motion vector components, one redundant spatial candidate is excluded.

When the number of motion vector predictors is not equal to two and the use of temporal motion vector prediction (TMVP) is not explicitly disable, the TMVP candidate is included. Finally, a zero motion vector is included repeatedly until the number of motion vector prediction candidates is equal to two, which guarantees that the number of motion vector predictor is two. Therefore, only a coded flag is necessary to identify which motion vector prediction is used in the case of non-merge mode [25] [23].

2.5.4 Transform, Scaling and Quantization

HEVC uses transform coding of the prediction error residual in a similar manner as in prior standards. The residual block is partitioned into multiple square TBs. The supported transform block sizes are 4x4, 8x8, 16x16 and 32x32.

Core Transform

Two-dimensional transforms are computed by applying 1-D transforms in the horizontal and vertical directions. The elements of the core transform matrices were derived by approximating scaled DCT basis functions. For simplicity, only one integer matrix for the length of 32 points is specified, and subsampled versions are used for other sizes. For example, the matrix for length-16 transform is as shown in the equation below [23] [33].

$$H = \begin{bmatrix}
64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\
90 & 87 & 80 & 70 & 57 & 43 & 25 & 9 & -9 & -25 & -43 & -57 & -70 & -80 & -87 & 90 \\
89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 & -89 & -75 & -50 & -18 & 18 & 50 & 75 & 89 \\
87 & 57 & 9 & -43 & -80 & -90 & -70 & -25 & 25 & 70 & 90 & 80 & 43 & -9 & -57 & -87 \\
83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 & 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\
80 & 9 & -70 & -87 & -25 & 57 & 90 & 43 & -43 & -90 & -57 & 25 & 87 & 70 & -9 & -80 \\
75 & -18 & -89 & -50 & 50 & 89 & 18 & -75 & -75 & 18 & 89 & 50 & -50 & -89 & -18 & 75 \\
70 & -43 & -87 & 9 & 90 & 25 & -80 & -57 & 57 & 80 & -25 & -90 & -9 & 87 & 43 & -70 \\
64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\
57 & -80 & -25 & 90 & -9 & -87 & 43 & 70 & -70 & -43 & 87 & 9 & -90 & 25 & 80 & -57 \\
50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 & -50 & 89 & -18 & -75 & 75 & 18 & -89 & 50 \\
43 & -90 & 57 & 25 & -87 & 70 & 9 & -80 & 80 & -9 & -70 & 87 & -25 & -57 & 90 & -43 \\
36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 & 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\
25 & -70 & 90 & -80 & 43 & 9 & -57 & 87 & -87 & 57 & -9 & -43 & 80 & -90 & 70 & -25 \\
18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 & -18 & 50 & -75 & 89 & -89 & 75 & -50 & 18 \\
9 & -25 & 43 & -57 & 70 & -80 & 87 & -90 & 90 & -87 & 80 & -70 & 57 & -43 & 25 & -9
\end{bmatrix}$$

The values of the entries in the matrix were selected to have key symmetry properties that enable fast partially factored implementations with far fewer mathematical operations than an ordinary matrix multiplication. For the transform block size of 4x4, an alternative integer transform derived from a DST is applied to the luma residual blocks for intrapicture prediction modes, with the transform matrix

$$H = \begin{bmatrix}
29 & 55 & 74 & 84 \\
74 & 74 & 0 & -74 \\
84 & -29 & -74 & 55 \\
55 & -84 & 74 & -29
\end{bmatrix}$$

The basis functions of the DST better fit the statistical property that the residual amplitudes tend to increase as the distance from the boundary samples that are used for prediction becomes larger. In terms of complexity, the 4x4 DST-style transform is not much more computationally demanding than the 4x4 DCT-style transform. The usage of the DST type of transform is restricted to only 4x4 luma transform blocks.

For quantization, HEVC uses essentially the same uniform reconstruction quantization (URQ) scheme controlled by a quantization parameter (QP) as in AVC. The range of the QP values is defined from 0 to 51. Quantization scaling matrices are also supported.

To reduce the memory needed to store frequency-specific scaling values, only quantization matrices of sizes 4x4 and 8x8 are used. For the larger transformations of 16x16 and 32x32 sizes, an 8x8 scaling matrix is sent and is applied by sharing values within 2x2 and 4x4 coefficient groups in frequency subspaces [23] [33].

2.6 Entropy Coding

Context-Bases Adaptive Binary Arithmetic Coding (CABAC) is a form of entropy coding used in AVC. Entropy coding is a form of lossless compression used at the last stage of video encoding and the first stage of video decoding, after the video has been reduced to a series of syntax elements. CABAC involves three main functions, binarization, context modeling and arithmetic coding. Binarization maps the syntax elements to binary symbols (bins). Context modeling estimates the probability of the bins. Lastly, arithmetic coding compresses the bins to bits based on the estimated probability [34].

Several different binarization process are used in HEVC including k-th order truncated Rice (TRk), k-th order Exp-Golomb (Egk) and fixed-length (FL) binarization. The binarization process is selected based on the type of syntax element. In some cases, binarization also depends on the value of a previously processed syntax elements or slice parameters that indicate if certain modes are enabled.

Context modeling provides an accurate probability estimate required to achieve high coding efficiency. Accordingly, it is highly adaptive and different context models can be used for different bins and the probability of that context model is updated based on the values of the previously coded bins while bins with similar distributions often share the same context model. The context model for each bin can be selected based on the type of syntax element, bin position in syntax element, luma or chroma, neighboring information. The probability models are stored as 7 bits entries, 6 bits for the probability state and 1 bit for the most probable symbol (MPS), in a context memory and addressed using the context index computed by the context selection logic. HEVC uses the same probability update method as AVC, however the context selection logic has been modified to improve throughput [23].

Arithmetic coding is based on recursive interval division, and HEVC make use of the same arithmetic coding used in H.264/AVC. A range, with an initial value of 0 to 1, is divided into two subintervals based on the probability of the bin. The encoded bits provide an offset that, when converted to a binary fraction, selects one of the two subintervals, which indicates the value of the decoded bin. After every decoded bin, the range is updated to equal the selected subinterval, and the interval division process repeats itself. Arithmetic coding can be done using estimated probability (context coded) or assuming equal probability of 0.5 (bypass coded) [26].

2.7 In-Loop Filters

The in-loop filters are applied in the encoding and decoding loops, after the inverse quantization and before saving the picture in the DPB. The HEVC standard specifies two in-loop filters, a deblocking filter and a sample adaptive offset (SAO). The deblocking filter is applied first and it attenuates discontinuities at the prediction and transform block boundaries. The SAO is applied to the output of the deblocking filter and it improves the quality of the decoded picture by attenuating ringing artifacts and changes in sample intensity of some areas of a picture.

In the HEVC, the deblocking filter is only applied to the boundaries of the PU and TU, which rely on a 8x8 samples grid for both luma and chroma. The strength of the deblocking filter is controlled by the values of several syntax elements similar to the scheme in H.264/AVC, but it only uses tree strengths rather than five [35].

The reconstructed samples are processed by the SAO module immediately after being filtered by the deblocking filter. The deblocked samples are subsequently modified by adding an offset value whose magnitude rely on a set of SAO parameters (type, four offset values and band position/edge class). These SAO parameters are encoded in the bitstream for each CTU and can have different values for the luma and the two chroma components of each CTU [36].

3 Overview of SHVC

3.1 Introduction

In the last section, we studied the HEVC algorithms. As our research is aligned to SHVC, we describe all SHVC tools in this section. During the design of the high-level syntax (HLS) of the first version of HEVC, several features were planned according to minimize changes to support the future scalable extensions and other extensions. The second version of HEVC introduces the HLS common to SHVC, MV-HEVC and future layered extensions.

Scalable video coding offers a mechanism for coding video into multiple layers, where each layer is characterized using the same video with different quality representation. The base layer (BL) represent the lowest quality representation while one or more enhancement layers (EL), which provide improved video quality, can be coded by referencing lower layers. Decoding a subset of layers of a scalable coded video bitstream results in video with a lower quality but still satisfactory than would result if the full bitstream were decoded. This permit a smooth degradation compared to non-scalable video bitstreams, where reduction in bitrate usually causes severe drops in video quality.

There are several types of video scalability, namely, temporal, spatial, quality, color gamut and bit depth. Temporal scalability allow the reduction of the frame rate of the video, spatial scalability allow the reduction of the spatial resolution of the video, quality scalability provides the reduction of the SNR, color gamut scalability allow the conversion of color gamut for services compatible with legacy devices and bit depth allow the reduction of the bit depth of the video [37] [38].

The use of scalable video coding allows applications to benefit from the adaptability of the bitstream according to the requirements of the decoder and network conditions. Error resiliency is another important feature of scalable video coding, since errors leading to loss of EL data cause much less degradation of video quality than errors leading to loss of non-scalable data.

3.2 SHVC Architecture and Scalability Features

In this section, we introduce SHVC architecture and what are its relationships with non-scalable HEVC. Figure 3.1 (a) illustrate the architecture of a two layer SHVC encoder. The scalable encoder consist of two encoders, one for each layer. In spatial scalable coding, the input video is downsampled and conveyed to the BL encoder, while the input video of the original size proceed to the EL encoder. In quality scalable coding, both encoders use the same input video. The BL encoder conforms to a single-layer video coding standard, so that backwards compatibility with single-layer coding is satisfied. The EL encoder includes additional coding features. Both encoders streams are multiplexed to form the scalable bitstream [39].

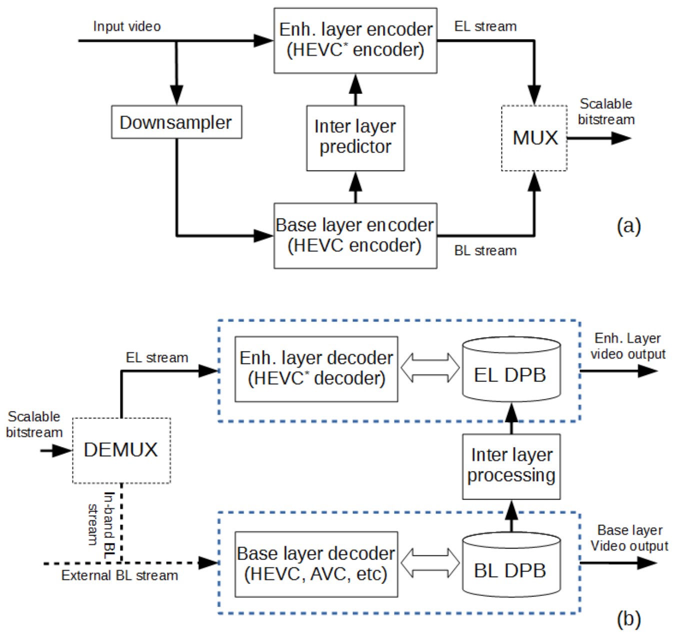

Figure 3.1 - SHVC (a) encoder and (b) decoder architectures with two layers.

Figure 3.1 (b) illustrate the architecture of a two layer SHVC decoder based on the reference index framework. When the BL is embedded within the SHVC bitstream, the input bitstream is demultiplexed into two separate layers. The BL bitstream is sent to the BL decoder and the EL bitstream is sent to the EL decoder. The BL decoder is an HEVC decoder, while the EL decoder (denoted as HEVC*) only differs from BL decoder at the High-Level syntax (HLS). To achieve efficient interlayer prediction (ILP), interlayer processing is applied to the reconstructed BL pictures retrieved from the decoded picture buffer (DPB) of the BL. Afterward, the processed pictures are put into the DPB of the EL and used as interlayer reference pictures for predictive coding of the EL.

The SHVC standard support the use of a BL bitstream coded using a non-HEVC single layer codec (e.g., H.264/AVC). After decoding, the reconstructed BL pictures are provided to the SHVC decoder, along with some information associated with the BL pictures. The remaining SHVC decoding operations follow the same procedures as the case with the embedded BL bitstream [37] [38] [39].

The reference-index-based SHVC architecture in Figure 3.1 (b) provides several design benefits. Firstly, since the ref_idx_syntax element is already present in the single layer HEVC standard at the PU level, referencing an interlayer reference picture can be done in a transparent manner at the block level. Any necessary changes to the EL decoder are only done at the slice header and above, and for this reason the SHVC architecture is often referred to as the HLS-only framework. Secondly, to achieve ILP, the only BL information that the EL needs to access is the reconstructed pictures present in the DPB of the BL, which includes the reconstructed texture samples and BL motion information. Since the DPB of the BL is provided as an open interface in a single-layer codec implementation, the scalable codec architecture does not require changes at all to the BL codec, allowing the BL codec to operate as a black box. Operating the BL codec as a black box allow the scalable system to easily support non-HEVC codecs to be used in the BL. Finally, the scalable system represents an architecture design harmonized with that of the Multiview extensions of HEVC, and that fact allow in the future a possible unification between the two extensions [37] [38].

3.3 Common Multilayer High-Level Syntax

The purpose of the common multilayer design is to enable maximum flexibility and future extensibility, allowing new definition of combinations of different types of scalability, even though the current specification does not define profiles enabling such combinations. An example of a future definition is the combination of spatial scalability and multiview scalability.

HLS refers to the syntax elements at the slice layer and above. The multilayer HLS design changes made for SHVC are applicable to the NAL unit header, VPS, SPS, PPS, slice header and some SEI messages.

3.3.1 Multilayer Design Concept and Definitions

A coded picture refers to a coded image in a single layer and an access unit contains one or more coded pictures, each from a different layer, that are associated with the same instant of output time. The POC (Picture Order Counter) represents the relative order output of pictures and is also used for reference picture indication. The derived POC for all pictures in an access unit is equal. Figure 3.2 shows an example of a two layer spatial scalable bitstream, containing a lower resolution BL and a higher resolution EL, with two temporal sub-layers. Each solid rectangle represent a picture and each row of pictures represent a layer [38]. A column of pictures represents an access unit. In SHVC, the decoding can start in an access unit if the picture at the lower layer of the bitstream is an IRAP (Intra Random Access Point). In Figure 3.2, SHVC can start the decoding process in the highlighted access unit, shown below, if only picture A is an IDR, while in the previous standard, SVC, both picture A and B were needed to be an IDR.

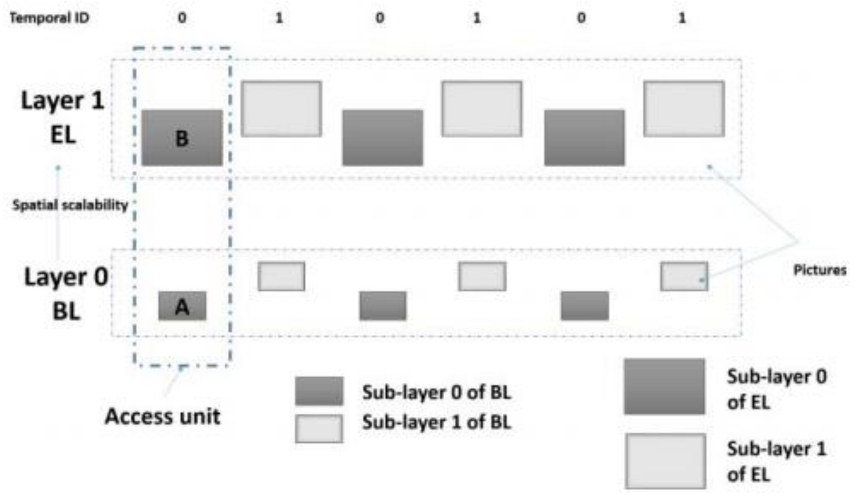

Figure 3.2 - Example of a bitstream representing layer, sublayer, picture and access unit [38].

With the flexibility to maintain POC derivation and reference picture selection, the concept of POC reset was introduced [40]. POC reset involves the reset of the derived POC value of the current picture and decrement of the POC of previous pictures for reference selection.

HEVC bring new concepts to express the conformance of multilayer bitstreams. A layer set is defined as a set of layers that forms a decodable sub-bitstream. More than one layer set may be specified for a bitstream corresponding to one or more sub-bitstreams. The example in Figure 3.2 shows a layer set containing two layers: Layer 0 (BL) and Layer 1 (EL). A layer set containing only Layer 0 (BL) can be specified, however, a layer set containing only Layer 1 (EL) cannot be specified because Layer 1 (EL) requires Layer 0 (BL) for decoding.

An operation point is a sub-bitstream representing a layer set at a potentially lower frame rate, for which a maximum temporal identity (ID) value is specified. Temporal ID value identifies temporal prediction restrictions among pictures such that a picture with a particular temporal ID value cannot use a picture with a higher temporal ID for reference. Using the example of Figure 3.2, an operation point could be specified that includes only temporal sublayer 0 pictures of Layers 0 and 1, which would include all the pictures that are shaded in dark grey, but an operation point that included temporal sublayer 0 pictures of Layer 0 and temporal sublayer 1 pictures of Layer 1 cannot be specified.

An output layer set is defined as a layer set with an associated set of target output layers. The target output layers of a layer set specify which layers the decoder will output. For the layer set in Figure 3.2, Layer 1 (EL) could be set as the only target output layer [38].

3.3.2 Syntax

To represent the layer ID of the layer contained in the NAL unit, the NAL unit header (NUH) use the syntax element nuh_layer_id. Every NAL units of a particular layer of a bitstream shall have the same nuh_layer_id value. When ILP of an EL is done using the reference layer (RL), the nuh_layer_id value of the RL need to be lower than the nuh_layer_id of the EL. For the BL to be backward compatible, the nuh_layer_id value must be 0. The syntax allow the possibility of using up to 63 layers in the bitstream. No new NAL unit types were introduced in SHVC since the types defined in HEVC can be used directly [41].

In SHVC, the parameters sets SPS and PPS can be shared by sequences and pictures, respectively, across layers, i.e., pictures from two different layers can refer to the same SPS or PPS. In the case of the parameter set VPS, the syntax elements that it contains can be applied across layers in a multilayer bitstream, which is useful for describing the overall bitstream characteristics for session negotiation. The VPS describes the number of layers and the dependency relationship between the layers, the representation format of the layers, DPB sizes and other information related to defining the conformance of the bitstream, which include layer sets, output layer sets, profile, tier level and timing related parameters.

The dependency relationship in the VPS extension describes the types of scalability and ILP used for each layer, with the value of ScalabilityId specifying the type of multilayer scalability used. DependencyId is used for spatial and/or SNR scalability, ViewOrderIdx for view scalability and AuxId for auxiliary pictures. Additionally, an indication is given as to which ILP tools can be used for coding a layer using its RL, including interlayer motion vector prediction and interlayer sample prediction.

3.4 Interlayer Reference Picture

In Figure 3.1, when any parameter between the RL and EL, including spatial resolution, bit depth and color gamut, is different, SHVC interlayer processing is applied to the RL pictures to form interlayer reference (ILR) pictures. Let us remind that RL can be BL and in this particular case EL is the upper layer of BL. Usually, the lower layer is called RL since it may not be the lowest layer, BL.

Interlayer processing in SHVC includes three modules, texture resampling, motion field resampling and color mapping. In SHVC, an EL picture can use multiple interlayer references for Interlayer Prediction (ILP), but to limit the complexity of interlayer processing, only one ILR picture can require resampling whenever an EL picture is decoded. Since interlayer references only occur within the current access unit, the ILR pictures do not need to be sorted after the current EL picture is decoded.

3.4.1 Texture Resampling

When an EL picture and its RL picture have different spatial resolutions and/or sample bit depths, texture resampling is applied to the RL picture to form an ILR picture that has the same resolution and bit depth as the EL picture. For each sample in an ILR picture, collocated fractional sample position is first identified, and then the resampling filter is applied to form the sample value in the ILR picture.

Concerning the collocated sample position determination, SHVC supports generalized spatial scalability, including arbitrary spatial resolution ratio, interlayer cropping mode and flexible resampling phase. The collocated sample position is calculated in units of 1/16 and based on the spatial ration, cropping parameters and resampling phase parameters.

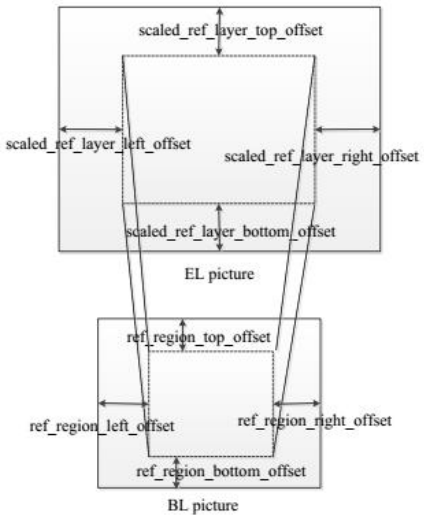
scaled_ref_layer_top_offset
scaled_ref_layer_left_offset
scaled_ref_layer_right_offset
scaled_ref_layer_bottom_offset
EL picture
ref_region_top_offset
ref_region_left_offset
ref_region_right_offset
ref_region_bottom_offset
BL picture

Figure 3.3 - Cropping parameters [38].

Given the location of a sample (luma or chroma) in the ILR picture, (xEL, yEL), the collocated RL sample location (x16RL, y16RL), in units of 1/16 RL samples, is derived as follows:

$$x16_{RL} = \left((x_{EL} - offsetX_{EL}) \times horS - \frac{phaseX \times horS + 8}{16} + 2^{11} \right) \gg (12 + offsetX_{RL}) \qquad (3.1)$$

$$y16_{RL} = \left((y_{EL} - offsetY_{EL}) \times verS - \frac{phaseY \times verS + 8}{16} + 2^{11} \right) \gg (12 + offsetY_{RL}) \qquad (3.2)$$

where horS and verS are the horizontal and vertical inverse scaling factors between the EL and RL in a 16 bit fixed point precision, phaseX and phaseY are the signaled horizontal and vertical resampling phases, $offsetX_{EL}$ and $offsetY_{EL}$ are the left and top scaled RL offsets, $offsetX_{RL}$ and $offsetY_{RL}$ are the left and top reference region offsets, and >> represents the arithmetic right shift. Observing Figure 3.3, $offsetX_{EL}$ and $offsetY_{EL}$ correspond to scaled_ref_layer_left_offset and scaled_ref_layer_top_offset, $offsetX_{RL}$ and $offsetY_{RL}$ correspond to ref_region_left_offset and ref_region_top_offset.

To support a flexible spatial relationship between two layers, SHVC uses two sets of cropping parameters, reference region offsets and scaled RL offsets, which are signaled inside the PPS extensions to enable efficient signaling of cropping, padding and region of interest extraction at the picture level. In Figure 3.3, each sets of cropping parameters contains four individual parameters to specify the number of luma samples to be padded or cropped on the four sides (top, bottom, left and right). The reference region offsets are defined for the RL picture while the scaled RL offsets are defined for the EL picture, and both are used to specify cropping or padding operations of the EL picture, and to calculate the values of horS and verS at equations (3.1) and (3.2) [37] [38].

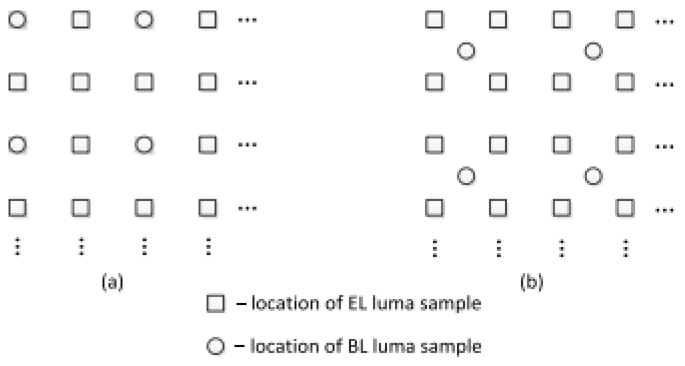
(a)
(b)
□ – location of EL luma sample
○ – location of BL luma sample

Figure 3.4 - Relative sampling grid position between the EL and the RL pictures for 2x spatial ratio. (a) SHVC default positions. (b) SVC default positions.

SHVC also supports flexible resampling phase adjustment between two layers. The PPS extension uses four syntax elements, phase_hor_luma, phase_ver_luma, phase_hor_chroma and phase_ver_chroma, to specify the horizontal and vertical resampling phase offsets of the luma and chroma components between the EL and RL, with phase offsets defined in units of 1/16 EL samples.

SHVC assumes default resampling phases, but its flexible phase adjustment capability gives the encoder the flexibility to choose different downsampling filters and specify corresponding phase adjustment parameters to be used in the resampling process. By default, the encoder downsampling filter assume that the encoder aligned the luma samples of the EL and RL pictures at the top-left corner, as shown in Figure 3.4 (a). Depending on the downsampling filters used by the encoder to generate the pictures in the RL, the relative phase relation between the RL and the EL can be different from the default. Figure 3.4 (b) shows the relative downsampling filter phase used in SVC [37].

Furthermore, SHVC can use the flexible resampling phase adjustment to support efficient interlaced-field-to-progressive-frame scalability, in which the RL pictures are interlaced field pictures, and the EL pictures are frame pictures. An additional application of the flexible resampling phase adjustment feature is chroma format scalability, where EL and RL use different chroma formats, e.g., RL is 4:2:0 and EL is 4:2:2 or 4:4:4 chorma format.

Now, let us focus our attention to the resampling interpolation process. The resampling interpolation filters used in SHVC were designed using the same principles as those used to design the HEVC motion compensation (MC) interpolation filters. While in the HEVC MC interpolation filters were defined only for the 1/4 pixel positions for luma and 1/8 pixel position for chroma, SHVC defines filter coefficients for the remaining 1/16 pixel position phases. The filter tap lengths were kept unchanged. Table 3.1 and Table 3.2 show the filter coefficients of the first eight phases, since the remaining eight are symmetric [37].

Table 3.1 - Luma upsampling Interpolation filter.

Phase	Interpolation filter coefficients							
1/16	0	1	-3	63	4	-2	1	0
2/16	-1	2	-5	62	8	-3	1	0
3/16	-1	3	-8	60	13	-4	1	0
4/16	-1	4	-10	58	17	-5	1	0
5/16	-1	4	-11	52	26	-8	3	-1
6/16	-1	3	-9	47	31	-10	4	-1
7/16	-1	4	-11	45	34	-10	4	-1
8/16	-1	4	-11	40	40	-11	4	-1

Table 3.2 - Chroma upsampling interpolation filter.

Phase	Interpolation filter coefficients			
1/16	-2	62	4	0
2/16	-2	58	10	-2
3/16	-4	56	14	-2
4/16	-4	54	16	-2
5/16	-6	52	20	-2
6/16	-6	46	28	-4
7/16	-4	42	30	-4
8/16	-4	36	36	-4

The resampling process is applied first in the horizontal direction and then in the vertical direction. If the sample bit depth in the RL is less than eight, a right shift operation is applied to the output of the horizontal resampling filters to assure that the intermediate data do not exceed a 16 bit dynamic range.

SHVC supports bit depth scalability, used when the sample bit depths of EL and RL pictures differ. When this occur, the bit depth difference between the EL and the RL is absorbed into the right shift after the vertical resampling filtering, such that the resampled ILR picture has the same bit depth as that of the EL picture.

3.4.2 Motion Field Resampling

Since SHVC uses the reference index-based scalable architecture, the ILR picture can be selected as the collocated picture for temporal motion vector prediction (TMVP) derivation. This allows interlayer motion vector (MV) prediction to be carried out without changes at the block level coding process. When the SNR scalability is employed, the motion parameters of an RL picture can be used directly for TMVP derivation, however, when EL and RL pictures have distinct spatial resolutions or use cropping parameters, motion parameters used in TMVP derivation need to be processed. That is, the collocated block needs to be identified by taking into account the spatial ratio and cropping parameters, and the MVs of the collocated block need to be scaled. When this processing is necessary, an interlayer motion field mapping (MFM) method is employed to generate the motion field of the ILR picture.

In HEVC, the PU is the block unit used that contains the motion information. To save memory required to store motion parameters of the reference pictures, PU level motion information is subsampled and stored into units of 16x16 blocks, and this process is denominated compressed motion field. TMVP is then carried out using the compressed motion field of the collocated picture. For backward compatibility considerations, SHVC MFM uses the compressed motion field from the RL picture. Additionally, the resampled motion field of the ILR picture is also generated and stored in units of 16x16 blocks. For each 16x16 block of the ILR picture, the motion parameters are derived based on those of its collocated block in the RL picture. The collocated RL 16x16 blocks are identified by taking into account the spatial scaling factors and any cropping parameters between the two layers.

Once the collocated RL 16x16 block is identified, the prediction mode and reference picture indices of the 16x16 block in the ILR picture are set equal to those of the collocated RL block, and the MVs of the current 16x16 block are generated by scaling the MVs of the collocated block based on the spatial scaling factor between the EL and the BL. When cropping parameters shown in Figure 3.3 are present, the collocated RL 16x16 block may be outside of the RL picture, and in this situation the motion parameters of the current 16x16 block in the ILR picture are marked as unavailable.

In addition to motion parameters of the collocated picture, the TMVP derivation process also needs certain high-level information of the collocated picture, including POC, slice type, and reference picture lists of each slice of the collocated picture. The POC values of the EL picture and RL picture are aligned, which means that the ILR simply use the POC value of the RL picture. One single virtual slice is generated for each ILR picture, then the slice type and reference picture lists of the ILR slice are copied from those of the first slice of the corresponding RL picture. In case the corresponding RL picture is coded with multiple slices and the slice type and/or reference picture list for any two of those slices are different, SHVC bitstream conformance requires interlayer motion prediction from an ILR picture to be disabled.

Finally, we should refer that despite the MFM is used for TMVP, the encoder strategies can make use of the reference layer motion vectors as the initial vector in the motion estimation search of the enhancement layer. No standardization of these strategies gives some freedom to the encoder designer. The reader interested in these strategies adopted in SVC can read the paper [42].

3.4.3 Color Mapping

Color gamut scalability (CGS) denominates the use of scalability when the RL and EL have distinct color gamuts, normally with the EL having a wider color gamut than the RL, and with SHVC applying a color mapping process to improve the coding efficiency. A 3D lookup table (LUT) based color mapping is used to generate texture samples in the ILR picture by converting samples in the RL picture from the RL color space to the EL color space [43]. When spatial scalability and CGS are used in combination, both upsampling and color mapping are required to generate the ILR picture. In this situation, to reduce the computational complexity, color mapping is applied on the lower resolution picture before the upsampling [44].

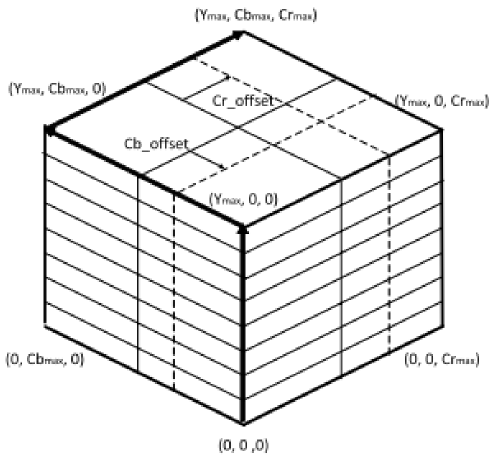

Figure 3.5 - 3D colour space partitions [43].

As shown in Figure 3.5, the 3D LUT-based color mapping in SHVC splits the input 3D YCbCr color space into up to 8x2x2 cuboid partitions. In the Y dimension, up to eight uniform partitions are used. In the Cb and Cr dimensions, up to two non-uniform partitions are used. When non-uniform partitioning is used for Cb and Cr, offsets relative to uniform partitioning labelled Cb_offset and Cr_offset in Figure 3.5, are signaled to specify the adaptive Cb and Cr partition thresholds. Since the chroma samples are usually unevenly distributed, allowing non-uniform partitions can deliver better sample classification compared with uniform partitions.

For each cuboid partition, the cross-color linear operation shown in (3.3) is used to perform color mapping.

$$\begin{bmatrix} y'_E \\ u'_E \\ v'_E \end{bmatrix} = \begin{bmatrix} a_y & b_y & c_y \\ a_u & b_u & c_u \\ a_v & b_v & c_v \end{bmatrix} \times \begin{bmatrix} y_R \\ u_R \\ v_R \end{bmatrix} + \begin{bmatrix} o_y \\ o_u \\ o_v \end{bmatrix} \tag{3.3}$$

where (y_R, u_R, v_R) is the input RL sample of the three color components; a_i, b_i, c_i and o_i, with i=y, u, or v, are the mapping coefficients and offsets of the cross-color linear model; and (y'_E, u'_E, v'_E) is the output sample of the three color components in the EL color gamut. For every cuboid partition, cross-color linear model parameters (a_i, b_i, c_i and o_i, with i=y, u, or v) are derived at the encoder and signaled in the PPS extensions.

Because the color mapping uses a cross-color linear model, it is essential that each color component in the input sample is aligned in terms of their sampling grid positions. To improve the precision of the color mapping process, the sampling grid positions of different color components are aligned before the cross-color linear model in (3.3) is applied. To apply color mapping to a given luma sample, the intermediate chroma sample values corresponding to the luma sampling grid positions are first calculated using a 2-tap or a 4-tap linear filter using the spatially neighboring chroma samples. The aligned intermediate chroma sample values are then used in the cross-color linear model. Similarly, to apply color mapping to a given chroma sample, the intermediate luma sample value corresponding to the chroma sampling grid positions is first calculated with a 2-tap or 4-tap linear filter using spatial neighboring luma samples, then the cross-color linear model is applied with the aligned luma sample value [37].

3.5 Final Remarks

In this Section, we described SHVC tools with focus on Interlayer prediction. Therefore, interpolation filters for 1/16 pixel resolution were shown. The BL motion field is also available for the enhancement layer. We are now ready to move to a new chapter, that of Rate-control where we propose one rate-distortion model.

4 Rate-Distortion Algorithms

4.1 Introduction

In the previous two chapters, we studied all main tools to encode video in HEVC and in SHVC, respectively. HEVC tools are them applied to every layer of SHVC with the addition of interlayer prediction. In this chapter, we will describe the rate control algorithms in HEVC which is replicated into each layer in SHVC. The independent rate control strategy in each layer was the main motivation of the research work in this dissertation.

4.2 Rate Control in HEVC

Rate control normally enables the encoder to select among a fixed and discrete set of coding parameters to achieve the target bitrate, constrained to a particular distortion/quality. One of these most frequently used parameters is QP.

In the design of rate control algorithms, Rate-Distortion (R-D) performance is one of the fundamental considerations. The rate control algorithm should accurately achieve the target bitrate and minimize the distortion. The rate control problem can be formulated as: minimize the distortion D, subject to a target total number of bits R_t.

$$\{Para\}_{opt} = \arg min_{\{Para\}} D \quad s.t. \quad R \leq R_t \tag{4.1}$$

where {Para} is the coding parameter set, which include mode, motion, QP, etc. Using the Lagrange multiplier method, the constrained optimization problem can be converted into unconstrained optimization problem as expressed in (4.2).

$$\{Para\}_{opt} = \arg min_{\{Para\}} \ (D + \lambda R) \tag{4.2}$$

where λ is the Lagrange multiplier, which indicates the slope of the R-D curve. D + λR is usually called R-D cost and Figure 4.1 represents a typical R-D curve of a video sequence [45].

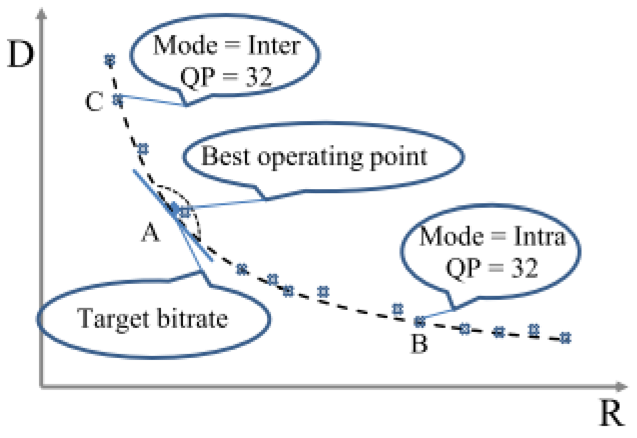

Figure 4.1 - A typical operational R-D curve [22].

A new rate distortion model R-λ has been inserted in the HEVC reference software HM which is explained in the following section.

4.3 R-λ Domain Rate Control

Several types of R-D model have been proposed to characterize the relationship between R and D. In [22], two typical R-D models were used to characterize the HEVC R-D relationship, Exponential function (4.3) and Hyperbolic function (4.4). It was concluded that for the HEVC, the Hyperbolic model was better than the Exponential model in characterizing the R-D relationship.

$$D(R) = C.e^{-KR} \tag{4.3}$$

$$D(R) = C.R^{-K} \tag{4.4}$$

C and K are model parameters related to the characteristics of the video source and in both models they are different. It is known that λ is the slope of R-D curve, which can be expressed as

$$\lambda = -\frac{\partial D}{\partial R} = C.K.R^{-K-1} \triangleq \alpha R^{\beta} \tag{4.5}$$

where α and β are parameters associated to the video source. If equation (4.5) is expressed in λ domain, we obtain

$$R = \left(\frac{\lambda}{\alpha}\right)^{\frac{1}{\beta}} \triangleq \alpha_1.\lambda^{\beta_1} \tag{4.6}$$

where α_1 is equal to $\left(\frac{1}{\alpha}\right)^{\frac{1}{\beta}}$ and β_1 is equal to $\frac{1}{\beta}$. The equation (4.6) demonstrates that the bitrate R is determined by λ value and [22] verify the model in (4.6).

4.4 High-level Bit Allocation

Before the above rate-distortion model can be applied to each CU, the total available number of bits is fully distributed per image like in the past with other video coding standards, MPEG-2 and MPEG4-AVC. Bit allocation can be implemented on three levels, group of pictures (GOP) level bit allocation, picture level bit allocation and basic unit (BU) level bit allocation.

Bits allocation for the first picture

Some coding parameters for the first picture, such as QP, in some rate control applications can be specified before the encoding. When it happens, the encoder does not need to define the target bits for that picture. However, when such parameter is not specified, it is impossible for the encoder to accurately estimate the number of bits for the first picture, since he has no prior knowledge on the video sequence. However, encoding the first picture without using the exact target bits is not a problem, since the performance of the rate control is measure over a long period and the encoder has a lot of changes to adjust the bitrate. In [22] a simple way was used to determine the target bits for the first picture:

$$R_{PicAvg} = \frac{R_{tar}}{FR} \tag{4.7}$$

where R_{PicAvg} stands for average bits per picture, R_{tar} the target bitrate of the sequence and FR the frame rate.

GOP Level Bit Allocation

To know the precisely target bits for every GOP, multiple pass encoding need to be performed. However, for practical applications is undesirable. One of the strategies used is, if the previous GOPs cost more or fewer bits than the target bits, then the current GOP should cost fewer or more bits accordingly. This strategy is used in the rate control algorithm in [22], and is calculated by

$$Target_{GOP} = \frac{R_{PicAvg} \times (N_{coded} + SW) - R_{coded}}{SW} \times N_{GOP} \tag{4.8}$$

Where $Target_{GOP}$ stands for target bits of a GOP, N_{coded} number of the coded picture, SW the size of sliding windows and N_{GOP} the number of pictures in a GOP.

Picture Level Bit Allocation

The picture level target bit in [22], which aims to assign the leftover bits to the remaining pictures according to the weight of each picture, is calculated as

$$Target_{Pic} = \frac{Target_{GOP} - Coded_{GOP}}{\sum_{\{AllNotCodedPicture\}} w_{pic}} \times w_{PicCurr} \tag{4.9}$$

where $Target_{Pic}$ stands as target bits of a picture, $Coded_{GOP}$ number of bits coded in the current GOP, w_{pic} weight of picture level bit allocation and $w_{PicCurr}$ weight of picture level bit allocation for current picture.

Basic Unit Level Bit Allocation

BU level bit allocation is analogous to picture level bit allocation, which aims to allocate the leftover bits to the remaining Bus according to the weight of each BU, and is calculated as

$$Target_{BU} = \frac{Target_{Pic} - Bit_H - Coded_{Pic}}{\sum_{\{AllNotCodedBUs\}} w_{BU}} \times w_{BUCurr} \tag{4.10}$$

where $Target_{BU}$ stands as target number of bits of a BU, Bit_H is the estimated header bits (obtained by averaging the header bits of the previous pictures of the same level), $Coded_{Pic}$ number of bits coded in the current picture, W_{BU} weight of BU level bit allocation and w_{BUCurr} weight of BU level bit allocation for current BU. In this case, W_{BU} is not a predetermined value as w_{Pic}, and is calculated based in estimated mean absolute difference (MAD). Mad is acquired from the prediction error of the collocated BU in the previously coded picture belonging to the same picture level and is calculated as

$$MAD_{BU} = \frac{1}{N_{pixels}} \sum_{\{AllPixelsInBU\}} \left| P_{org} - P_{pred} \right| \tag{4.11}$$

where N_{pixels} stands as the number of pixels in the current BU, P_{org} is the pixel value of the original signal and P_{pred} is the pixel value of the predicted signal. Finally, W_{BU} is obtained by

$$W_{BU} = MAD_{BU}^2. \tag{4.12}$$

In HEVC, the basic unit is the CU. Once W_{BU} is determined, the rate-distortion described in Section 4.3 is used.

4.5 Rate Control Coding

Knowing only the target bits or bits per pixel (bpp) is insufficient to start encoding the video sequence. Others parameters, at CU level, have to be determined such as the λ value and QP value, among others parameters.

λ Determination and Updating

It is straightforward to determine λ according to the target bitrate R by equation (4.5), however, the problem is how to determine the parameters α and β, since different video sequences could have different values. To adapt the different characteristics of the various video sequences, [22] designed the following algorithms.

$$\lambda = \alpha . bpp^{\beta} \tag{4.13}$$

Equation (4.13) is used to derive λ value from target R for a picture or BU, but the model may have different α and β values for different cases. So the algorithm initialize α and β with the values 3.2003 and -1.367, respectively. The initial values of α and β are not decisive for rate control coding, as different sequences can have different values and the values will keep updating with the encoding proceeding.

After encoding one BU or one picture, the real bpp cost (bpp_{real}) and real applied λ value (λ_{real}) are used to update α and β of the model by equations (4.15) and (4.16).

$$\lambda_{comp} = \alpha_{old} bpp_{real}^{\beta_{old}} \tag{4.14}$$

$$\alpha_{new} = \alpha_{old} + \delta_{\alpha} \times \left(\ln \lambda_{real} - \ln \lambda_{comp}\right) \times \alpha_{old} \tag{4.15}$$

$$\beta_{new} = \beta_{old} + \delta_{\beta} \times \left(\ln \lambda_{real} - \ln \lambda_{comp}\right) \times \ln bpp_{real} \tag{4.16}$$

δ_{α} and δ_{β} are predefined values set to 0.1 and 0.05 respectively and λ_{comp} is the computed value from (4.13) with real encoded bpp.

RDO coding

When λ value is set, all coding parameters can be determined by exhaustive rate distortion optimization (RDO) search. However, instead of exhaustive search to find the optimal QP for a given λ, the QP value is determined by (4.17) proposed in [46].

$$QP = c_1 \times \ln(\lambda) + c_2 \tag{4.17}$$

where c_1 and c_2 are predefined values set to 4.2005 and 13.7122 respectively. The determined QP value should be rounded to the nearest integer.

To keep the quality of coded video consistent, both λ and QP should not change considerably. For picture level, λ and QP should satisfy (4.18) and (4.19) [47]. And for BU level, λ and QP should satisfy (4.20) to (4.23) [47].

$$\lambda_{C\ PicEst} = Clip\left(\lambda_{L\ PicAvg} \times 2^{-\frac{2}{3}}, \lambda_{L\ PicAvg} \times 2^{\frac{2}{3}}, \lambda_{C\ PicEst}\right) \tag{4.18}$$

$$QP_{C\ PicEst} = Clip(QP_{L\ PicAvg} - 2, QP_{L\ PicAvg} + 2, QP_{C\ PicEst}) \tag{4.19}$$

$$\lambda_{C\ BU\ Est} = Clip\left(\lambda_{L\ BU} \times 2^{-\frac{1}{3}}, \lambda_{L\ BU} \times 2^{\frac{1}{3}}, \lambda_{C\ BU\ Est}\right) \tag{4.20}$$

$$QP_{C\ BU\ Est} = Clip(QP_{L\ BU} - 1, QP_{L\ BU} + 1, QP_{C\ BU\ Est}) \tag{4.21}$$

$$\lambda_{C\ BU\ Est} = Clip\left(\lambda_{C\ PicEst} \times 2^{-\frac{2}{3}}, \lambda_{C\ PicEst} \times 2^{\frac{2}{3}}, \lambda_{C\ BU\ Est}\right) \tag{4.22}$$

$$QP_{C\ BU\ Est} = Clip(QP_{C\ PicEst} - 2, QP_{C\ PicEst} + 2, QP_{C\ BU\ Est}). \tag{4.23}$$

The meaning of the notations used in the previous equations are shown in Table 4.1. The function Clip means the third parameter is limited between the first and second parameter.

Table 4.1 - Meaning of the notations used in equations (4.18) to (4.23).

Notation	**Meaning**
$\lambda_{C\ PicEst}$	Estimated λ of the current picture
$\lambda_{L\ PicAvg}$	Average λ used in the last picture belonging to the same level
$QP_{C\ PicEst}$	Estimated QP of the current picture
$QP_{L\ PicAvg}$	Average QP used in the last picture belonging to the same level
$\lambda_{C\ BU\ Est}$	Estimated λ of the current BU
$\lambda_{L\ BU}$	λ of the previous BU
$QP_{C\ BU\ Est}$	Estimated QP of the current BU
$QP_{L\ BU}$	QP of the previous BU

4.6 Proposed Rate-distortion Model

The R-D model proposed [48] in this dissertation for the enhancement layers for the spatial scalability case include a logarithmic function and is given by

$$D(R) = -C.\ln R + K \tag{4.24}$$

where C and K are model parameters related to the characteristics of the video source. It is known that λ is the slope of R-D curve, which can be expressed as the derivative of (4.24),

$$\lambda = -\frac{\partial D}{\partial R} = C.R^{-1} \triangleq \alpha R^{-1} \tag{4.25}$$

where α is a parameter associated to the video source. If equation (4.25) is expressed in λ domain, we obtain

$$R = \alpha.\lambda^{-1}. \tag{4.26}$$

The equation (4.26) demonstrates that the bitrate R is determined by λ value and we will show later that our R-D model surpass λ-R model included in SHM.

5 Experimental Results

5.1 Introduction

In this dissertation, we focused our research on two layers. We started by comparing SHVC with HEVC. Afterward, we implemented the proposed R-D model described in Section 4.6.

For the encoding of the video sequences we used three encoding configurations, Low Delay P, Low Delay and Random Access. In the Low Delay P configuration the first picture is coded as an I picture and the following pictures are coded as P pictures. For the Low Delay configuration, the process is similar to the Low Delay P configuration, with the difference that the subsequent pictures are now encoded as B pictures. Both this configurations have low coding delay (coding order is equal to output order). Figure 5.1 shows an example of these prediction structure.

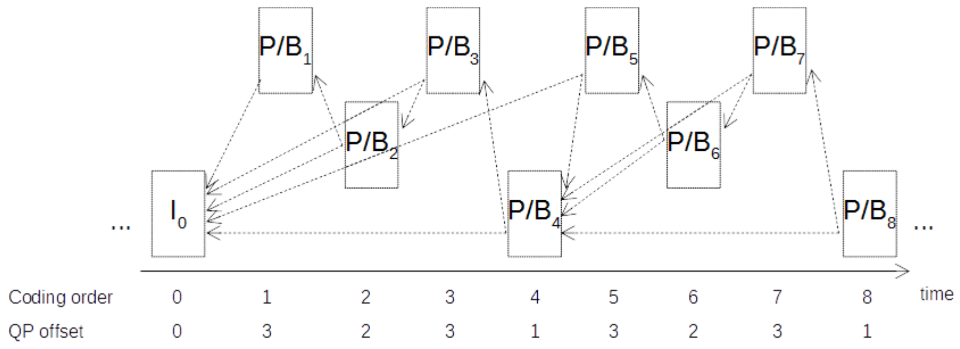

Figure 5.1 - Prediction structure of Low Delay P and B configurations.

In the case of Random Access configuration, a hierarchical B structure is used [49]. The bi-directional hierarchical prediction structure achieve a higher coding efficiency than the other configurations. However, coding order and output order of the pictures differ and thereby induce a coding delay. I pictures are inserted periodically to control possible error propagation. Figure 5.2 shows an example of this prediction structure.

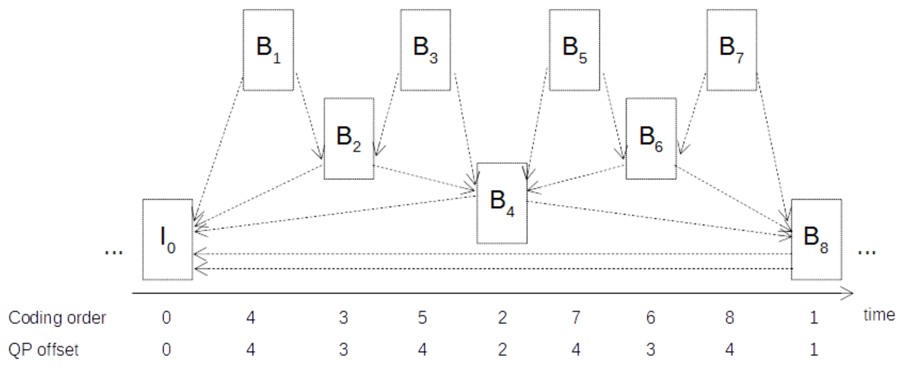

Figure 5.2 - Prediction structure of the Random Access configuration.

The video sequences used in single layer encoding are from class B (1080p) and E (720p). For multi-layer (SHVC), the video sequences classes used are class B (BL with 720p and EL with 1080p).

Finally, to evaluate the overall results between reference software and our proposed model, Bjontegaard Delta PSNR (BD PSNR) and Bitrate (DB Rate) [50] was used. The BD metric allow to compute the average gain in PSNR and average percentage of bitrate reduction between two rate-distortion curves.

5.2 SHVC Performance vs HEVC performance

The experiment was performed using SHM 8.0 version [51] of the SHVC reference software, and the HM 16.6 [52] version of the HEVC reference software, to evaluate the coding performance of SHVC for two-layer spatial scalability. Three types of comparisons were made, bitrate reduction, encoding time reduction and EL distortion (PSNR) gain. The HEVC simulcast codes the BL and EL independently, and the total EL+BL bits and encoding time is used for comparison with the SHVC standard. Afterward, we compare the EL distortion between SHVC and HEVC simulcast (there is no need to compare de distortion in the BL, since we realized in Section 3.2 that the BL in SHVC is encoded has if it was a HEVC encoder).

A spatial scalability of 1.5x was performed following the SHVC common test conditions [53] with two video sequences [54], shown in Table 5.1, being used in the experiment. Fixed QP was used for each sequence, with five bit rate points being tested: five BL QPs (20, 24, 28, 32, 36) combined with five EL QPs (20, 24, 28, 32, 36). Only one coding configuration, low delay with p pictures, was used. Table 5.2 shows the coding performance between HEVC in a simulcast configuration and SHVC.

Table 5.1 - Spatial Scalability Test Sequences.

Sequence Name	Resolution	Frame rate	Chroma Sampling
Crowd Run	1920x1080	50	4:2:0
	1280x720	50	4:2:0
Old Town Cross	1920x1080	50	4:2:0
	1280x720	50	4:2:0

Observing Figure 5.3, in average, the encoding time has a reduction of one quarter when using SHVC instead of a simulcast configuration with HEVC. Although HEVC simulcast could mitigate and reduce the encoding time, it would require encoding two sequences at the same time, which would require the double of computational resources than used with SHVC.

The use of SHVC leads to a reduction of the bit rate overall, however the reduction percentage depends on the value of the ΔQP used as we can see in Figure 5.4. When the ΔQP is negative, QP of BL is higher than the QP of the EL, the reduction is smaller. However, when the ΔQP is positive (QP of BL is lower than the QP of the EL) the reduction is bigger, with the higher reductions occurring when the ΔQPs is between 4 and 8.

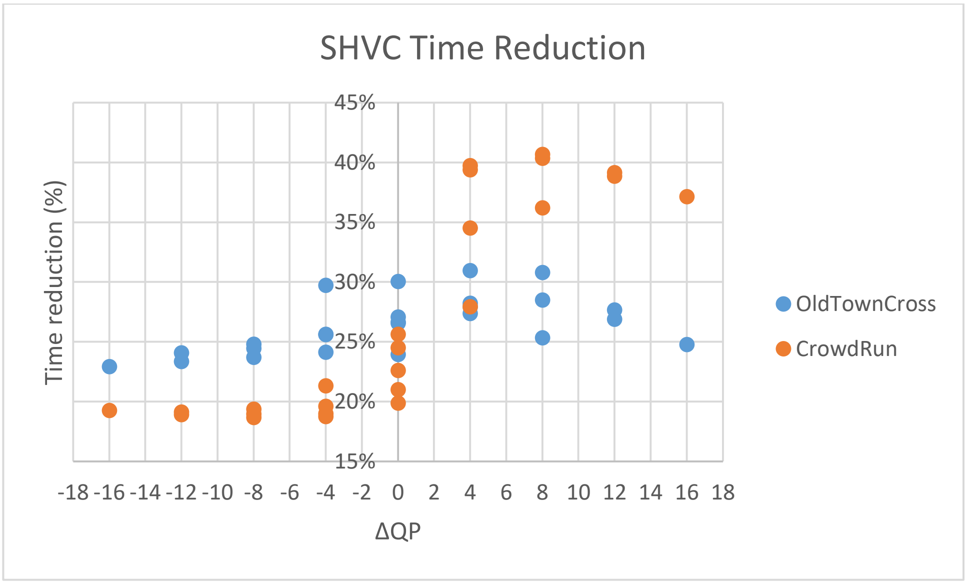

Figure 5.3 – Dependence of the QP between layers in the time reduction.

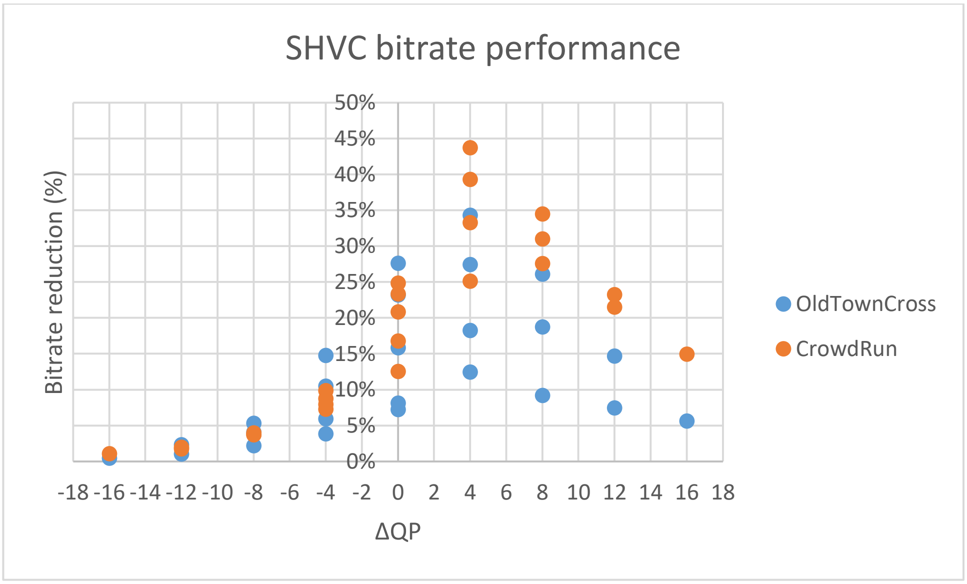

Figure 5.4 – Dependence of the QP between layers in the bitrate reduction.

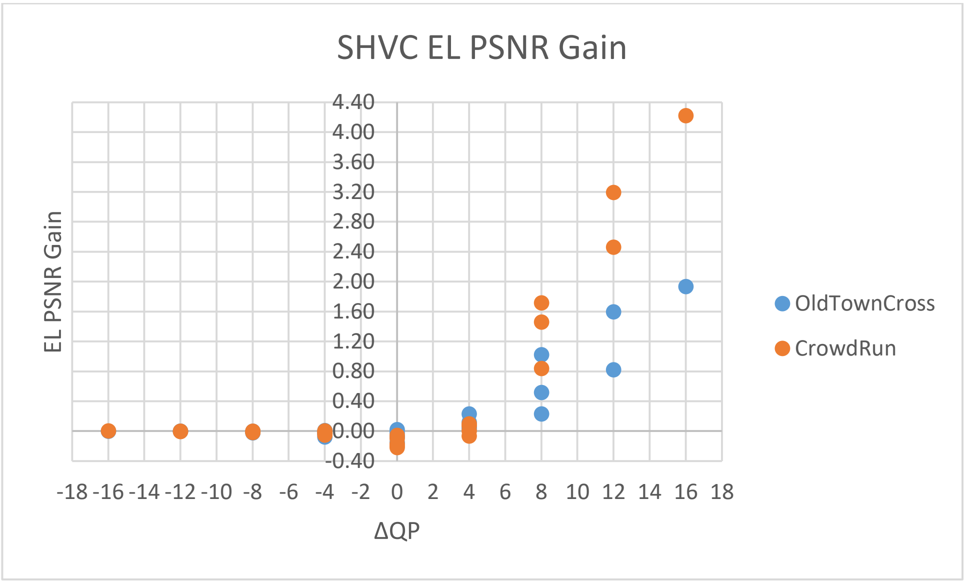

Figure 5.5 – Depende of the QP between layers in the EL PSNR.

The SHVC EL video, normally, does not lose much quality in comparison with HEVC EL video. Observing Table 5.2 the loss in quality never increase more than 0.1 dB for the case of the "Old Town Cross" video sequence and 0.2 dB for the case of the "Crowd Run" video sequence. However, the gains always occur when the ΔQP is positive, with the increase of the ΔQP implying higher gains, as we can observe in Figure 5.5. We can also verify that when the EL QP is equal to 32, we always got the higher reduction time.

Table 5.2 - Coding performance between SHVC and HEVC.

		Old Town Cross			***Crowd Run***		
QP BL	**QP EL**	**Bitrate reduction**	**Time reduction**	**PSNR EL gain (dB)**	**Bitrate reduction**	**Time reduction**	**PSNR EL gain (dB)**
20	20	7.2%	23.9%	-0.074	12.5%	19.9%	-0.058
20	24	12.4%	27.4%	0.061	25.1%	27.9%	-0.069
20	28	9.2%	25.3%	0.230	27.6%	36.2%	0.837
20	32	7.4%	26.9%	0.821	21.5%	38.9%	2.458
20	36	5.6%	24.8%	1.935	15.0%	37.1%	4.221
24	20	3.8%	24.1%	-0.034	7.3%	18.7%	0.001
24	24	8.1%	26.6%	0.017	16.8%	21.0%	-0.102
24	28	18.2%	27.9%	0.037	33.3%	34.5%	0.001
24	32	18.7%	30.8%	0.517	31.0%	40.3%	1.456
24	36	14.7%	27.7%	1.595	23.2%	39.1%	3.193
28	20	2.2%	23.7%	-0.017	3.7%	18.7%	-0.002
28	24	5.9%	25.6%	0.005	7.9%	19.0%	-0.009
28	28	15.8%	26.6%	-0.023	20.8%	22.6%	-0.175
28	32	27.4%	30.9%	0.116	39.3%	39.7%	0.094

28	36	26.1%	28.5%	1.021	34.5%	40.7%	1.712
32	20	1.0%	23.3%	-0.004	2.0%	18.9%	-0.005
32	24	4.1%	24.4%	-0.002	3.7%	19.0%	-0.007
32	28	10.5%	25.6%	-0.031	8.7%	19.6%	-0.025
32	32	23.2%	30.0%	-0.078	23.3%	24.5%	-0.200
32	36	34.3%	28.2%	0.227	43.7%	39.4%	0.068
36	20	0.4%	22.9%	-0.002	1.0%	19.2%	0.003
36	24	2.3%	24.1%	-0.001	1.7%	19.1%	-0.007
36	28	5.3%	24.8%	-0.026	4.0%	19.3%	-0.021
36	32	14.7%	29.7%	-0.081	9.9%	21.3%	-0.053
36	36	27.6%	27.1%	-0.155	24.8%	25.6%	-0.221

It is clear, as explained in Section 3, that the interlayer prediction achieves a bit reduction and time reduction of 28.7% and 33.6% without any significant loss in quality [55].

5.3 R-D model for the EL

To verify if our R-D model overcomes the R-D model used in HEVC for the EL in multi-layer coding, we fixed the QP value of the BL and varied the EL QP values. After obtaining the bitrate R and distortion D of the EL, we then fitted the R-D curve according to three different models, Hyperbolic and Logarithmic. R is expressed in bits per pixel (bpp) and D is expressed in mean square error (MSE) of luma component. To measure how well these models fitted in the experimental observations, we used correlation coefficient. Several tests were performed for different QPs values in the BL. The configurations used were Low Delay P, Low Delay and Random Access. The video sequences used are shown in Table 5.3.

Table 5.3 - Spatial Scalability Test Sequences for Rate Control..

Sequence Name	**Resolution**	**Frame rate**	**Chroma Sampling**
Crowd Run	1920x1080	50	4:2:0
	1280x720	50	4:2:0
Old Town Cross	1920x1080	50	4:2:0
	1280x720	50	4:2:0
kimono	1920x1080	24	4:2:0
	1280x720	24	4:2:0

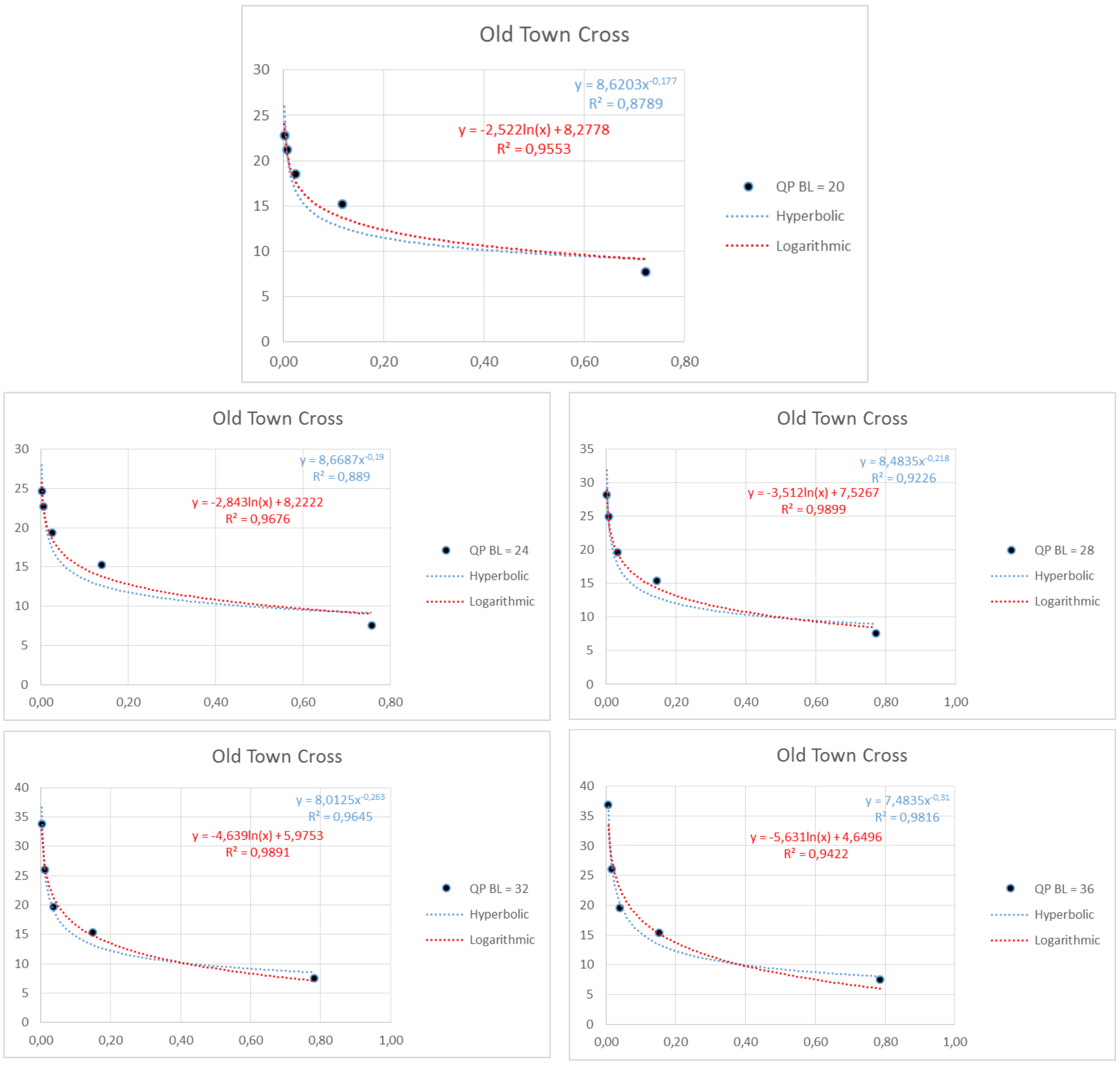

Figure 5.6 – R-D curve fitting according to the three different models for EL.

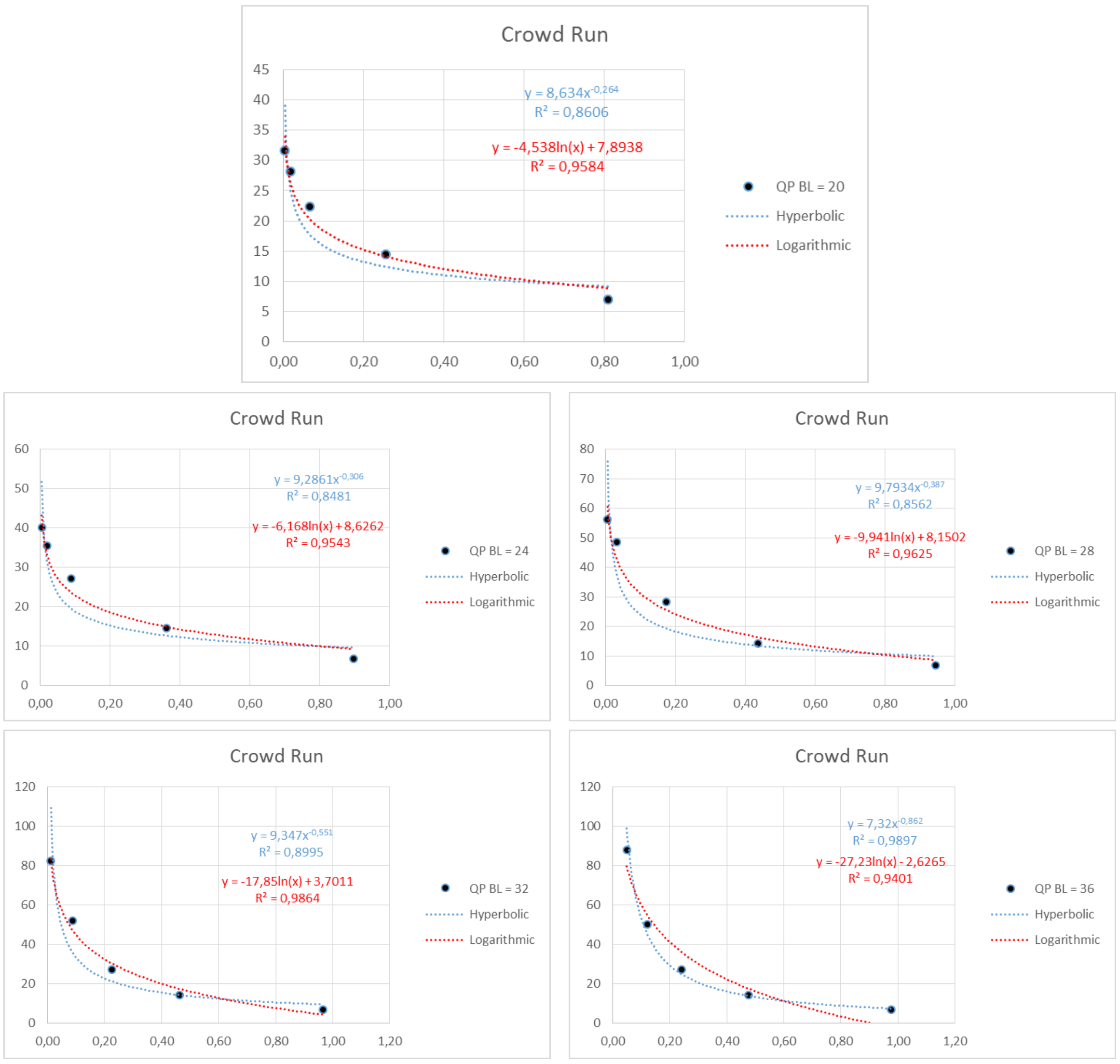

Figure 5.7 – R-D curve fitting according to the three different models for EL.

Analyzing Figure 5.6 and Figure 5.7, none of the models has a perfect fitting. The correlation coefficient in the hyperbolic model used in HEVC, is irregular along the tests. For the logarithmic model, it exhibit a more regular and higher correlation coefficients than the hyperbolic model. From the two fitting curves, we can conclude that none of the models is a perfect fit. However, the logarithmic model is far better characterizing the R-D relationship than the hyperbolic model.

We extended our research into the BL, and observing Figure 5.8 we verified that the hyperbolic function is indeed the best model to characterize the R-D relationship for the BL. However, as we have previously verified, the hyperbolic model used in HEVC is not appropriate for the EL in SHVC.

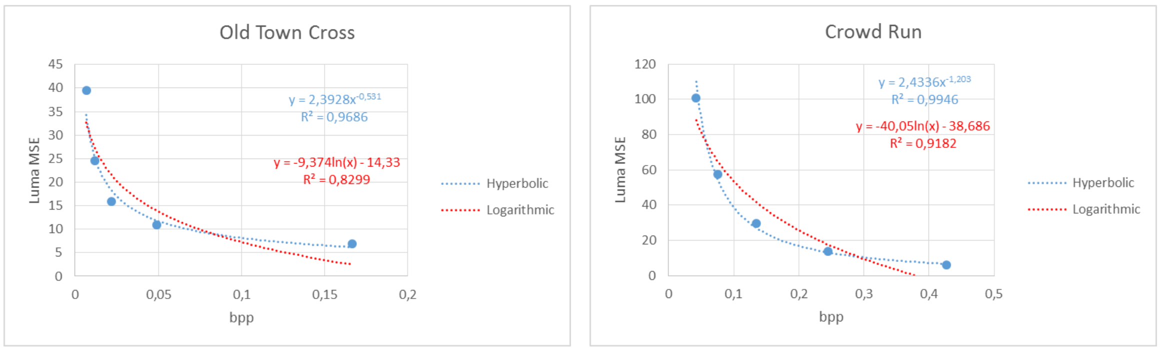

Figure 5.8 – R-D curve fitting for BL.

Taking in account what we determined before, we implemented our model from Section 4.6 in the SHVC reference software. For that, we changed the equation used to derive λ to the following proposed new equation:

$$\lambda = \alpha . bpp^{-1} \tag{5.1}$$

Table 5.4 to Table 5.9 show the results obtained with our proposed R-D model.

Table 5.4 - Experimental Results on Rate Control, Low Delay P coding structure without hierarchical configuration (bitrate in Kbps, PSNR in dB).

Sequence	Target bitrate	SHM (no Hierarchical)			Proposed RC (no Hierarchical)			BD PSNR	BD Rate
		Bitrate	PSNR	Bitrate error	Bitrate	PSNR	Bitrate error		
OldTownCross	2500	2500.28	34.75	0.01%	2502.74	34.78	0.11%	-0.0063	0.601%
	5000	5005.10	35.36	0.10%	5000.42	35.38	0.01%		
	10000	10005.74	35.98	0.06%	10006.12	35.96	0.06%		
	15000	15005.60	36.38	0.04%	15006.40	36.30	0.04%		
	20000	20004.62	36.67	0.02%	20005.14	36.60	0.03%		
CrowdRun	2500	2501.16	30.98	0.05%	2507.38	31.05	0.30%	0.0369	-2.645%
	5000	5001.56	31.48	0.03%	5019.70	31.52	0.39%		
	10000	10009.20	32.36	0.09%	10009.86	32.37	0.10%		
	15000	15007.46	33.11	0.05%	15010.78	33.18	0.07%		
	20000	20007.02	33.85	0.04%	20011.02	33.93	0.06%		
Kimono	2500	2501.70	40.06	0.07%	2501.63	40.05	0.07%	0.0170	-0.992%
	5000	5004.29	41.35	0.09%	5003.75	41.38	0.08%		
	10000	10007.98	42.58	0.08%	10008.75	42.60	0.09%		
	15000	15009.76	43.19	0.07%	15009.06	43.20	0.06%		
	20000	20008.54	43.63	0.04%	20009.04	43.64	0.05%		

Table 5.5 - Experimental Results on Rate Control, Low Delay P coding structure with hierarchical configuration (bitrate in Kbps, PSNR in dB).

Sequence	Target bitrate	SHM (Hierarchical)			Proposed RC (Hierarchical)			BD PSNR	BD Rate
		Bitrate	PSNR	Bitrate error	Bitrate	PSNR	Bitrate error		
OldTownCross	2500	2506.66	34.61	0.27%	2448.24	34.66	-2.07%	0.0226	2.598%
	5000	4999.72	35.22	-0.01%	5005.04	35.19	0.10%		
	10000	10006.68	35.78	0.07%	10007.04	35.84	0.07%		
	15000	15007.52	36.19	0.05%	15006.88	36.26	0.05%		
	20000	20007.94	36.54	0.04%	20007.02	36.57	0.04%		
CrowdRun	2500	2504.60	31.03	0.18%	2502.78	31.12	0.11%	0.0549	-3.727%
	5000	5006.00	31.55	0.12%	5004.36	31.62	0.09%		
	10000	10007.48	32.42	0.07%	10002.90	32.45	0.03%		
	15000	15008.58	33.20	0.06%	15002.16	33.21	0.01%		
	20000	20009.18	33.86	0.05%	20002.58	33.93	0.01%		
Kimono	2500	2501.88	39.84	0.08%	2499.93	39.83	0.00%	0.0132	-0.714%
	5000	5003.96	41.12	0.08%	5001.96	41.12	0.04%		
	10000	10008.36	42.46	0.08%	10007.50	42.50	0.08%		
	15000	15008.42	43.10	0.06%	15009.37	43.12	0.06%		
	20000	20008.93	43.58	0.04%	20009.25	43.59	0.05%		

Table 5.6 - Experimental Results on Rate Control, Low Delay coding structure without hierarchical configuration (bitrate in Kbps, PSNR in dB).

Sequence	Target bitrate	SHM (no Hierarchical)			Proposed RC (no Hierarchical)			BD PSNR	BD Rate
		Bitrate	PSNR	Bitrate error	Bitrate	PSNR	Bitrate error		
OldTownCross	2500	2502.52	34.81	0.10%	2501.00	34.85	0.04%	0.0080	-0.949%
	5000	4999.04	35.44	-0.02%	4999.76	35.47	0.00%		
	10000	10007.04	36.11	0.07%	10005.66	36.12	0.06%		
	15000	15007.82	36.54	0.05%	15006.64	36.47	0.04%		
	20000	20006.90	36.85	0.03%	20006.24	36.77	0.03%		
CrowdRun	2500	2501.34	31.01	0.05%	2507.14	31.08	0.29%	0.0509	-3.519%
	5000	5006.62	31.52	0.13%	5013.04	31.56	0.26%		
	10000	10007.16	32.41	0.07%	10009.04	32.44	0.09%		
	15000	15008.28	33.19	0.06%	15011.30	33.27	0.08%		
	20000	20006.76	33.93	0.03%	20011.18	34.03	0.06%		
Kimono	2500	2500.60	40.24	0.02%	2500.17	40.21	0.01%	0.0160	-0.958%
	5000	5002.61	41.58	0.05%	5002.35	41.61	0.05%		
	10000	10005.98	42.79	0.06%	10007.21	42.81	0.07%		
	15000	15008.95	43.37	0.06%	15009.10	43.38	0.06%		
	20000	20009.42	43.81	0.05%	20008.78	43.81	0.04%		

Table 5.7 - Experimental Results on Rate Control, Low Delay coding structure with hierarchical configuration (bitrate in Kbps, PSNR in dB).

Sequence	Target bitrate	SHM (Hierarchical)			Proposed RC (Hierarchical)			BD PSNR	BD Rate
		Bitrate	PSNR	Bitrate error	Bitrate	PSNR	Bitrate error		
OldTownCross	2500	2508.50	34.65	0.34%	2469.72	34.75	-1.21%	0.0172	-1.852%
	5000	5009.96	35.30	0.20%	5004.40	35.26	0.09%		
	10000	10008.24	35.91	0.08%	10009.36	35.96	0.09%		
	15000	15000.84	36.35	0.01%	15008.18	36.41	0.05%		
	20000	20008.62	36.71	0.04%	20009.02	36.76	0.05%		
CrowdRun	2500	2504.38	31.05	0.18%	2502.60	31.15	0.10%	0.0532	-3.525%
	5000	5007.40	31.58	0.15%	5004.98	31.65	0.10%		
	10000	10008.52	32.48	0.09%	10003.38	32.51	0.03%		
	15000	15007.60	33.28	0.05%	15002.00	33.29	0.01%		
	20000	20008.52	33.95	0.04%	20014.88	34.02	0.07%		
Kimono	2500	2501.44	40.00	0.06%	2499.05	40.00	-0.04%	0.0197	-1.066%
	5000	5001.70	41.31	0.03%	4996.62	41.32	-0.07%		
	10000	10005.21	42.65	0.05%	10005.84	42.70	0.06%		
	15000	15008.36	43.28	0.06%	15008.02	43.31	0.05%		
	20000	20004.03	43.74	0.02%	20004.25	43.77	0.02%		

Table 5.8 - Experimental Results on Rate Control, Random Access coding structure without hierarchical configuration (bitrate in Kbps, PSNR in dB).

Sequence	Target bitrate	SHM (no Hierarchical)			Proposed RC (no Hierarchical)			BD PSNR	BD Rate
		Bitrate	PSNR	Bitrate error	Bitrate	PSNR	Bitrate error		
OldTownCross	2500	2501.52	34.69	0.06%	2501.94	35.01	0.08%	0.0487	-2.744%
	5000	5002.06	35.45	0.04%	5006.82	35.56	0.14%		
	10000	10010.52	36.13	0.11%	10005.64	36.03	0.06%		
	15000	15018.52	36.50	0.12%	15009.50	36.44	0.06%		
	20000	20062.80	36.76	0.31%	20116.50	36.80	0.58%		
CrowdRun	2500	2513.26	31.15	0.53%	2703.46	31.27	8.14%	-0.1614	12.944%
	5000	5345.04	31.82	6.90%	5001.46	31.65	0.03%		
	10000	10401.62	32.82	4.02%	9558.02	32.39	-4.42%		
	15000	15493.14	33.48	3.29%	14379.38	33.10	-4.14%		
	20000	20558.16	34.13	2.79%	19172.24	33.75	-4.14%		
Kimono	2500	2509.10	40.37	0.36%	2500.71	40.42	0.03%	0.0609	-3.662%
	5000	5006.54	41.62	0.13%	5002.78	41.69	0.06%		
	10000	10009.48	42.70	0.09%	10004.79	42.78	0.05%		
	15000	15009.29	43.29	0.06%	15010.80	43.33	0.07%		
	20000	20003.76	43.75	0.02%	20011.16	43.77	0.06%		

Table 5.9 - Experimental Results on Rate Control, Random Access coding structure with hierarchical configuration (bitrate in Kbps, PSNR in dB).

Sequence	Target bitrate	SHM (Hierarchical)			Proposed RC (Hierarchical)			BD PSNR	BD Rate
		Bitrate	PSNR	Bitrate error	Bitrate	PSNR	Bitrate error		
OldTownCross	2500	2500.82	34.67	0.03%	2424.48	35.05	-3.02%	0.2201	-18.358%
	5000	5115.86	35.25	2.32%	5001.36	35.62	0.03%		
	10000	10006.20	36.11	0.06%	10004.40	36.16	0.04%		
	15000	15004.28	36.51	0.03%	15004.38	36.49	0.03%		
	20000	20007.42	36.79	0.04%	20003.86	36.78	0.02%		
CrowdRun	2500	2500.84	31.20	0.03%	2500.30	31.25	0.01%	-0.2234	16.432%
	5000	5003.04	31.83	0.06%	4953.54	31.64	-0.93%		
	10000	10064.54	32.78	0.65%	9434.46	32.38	-5.66%		
	15000	15160.28	33.67	1.07%	15050.06	33.33	0.33%		
	20000	20164.28	34.40	0.82%	20301.12	33.96	1.51%		
Kimono	2500	2507.28	40.16	0.29%	2500.34	40.22	0.01%	0.0671	-3.949%
	5000	5019.96	41.47	0.40%	5001.33	41.55	0.03%		
	10000	10010.51	42.60	0.11%	10007.72	42.67	0.08%		
	15000	15004.06	43.20	0.03%	15003.74	43.22	0.02%		
	20000	20000.51	43.68	0.00%	20000.67	43.67	0.00%		

From Table 5.4 to Table 5.9, we obtained the following averages BD PSNR and bitrate errors summarized in Table 5.10. We can observe that our R-D model for Low Delay P, Low Delay and Random Access with hierarchical mode configurations obtained a better PSNR than the model used in the reference software, whereas without hierarchical mode only the Random Access configuration did not performed better than the reference software. In all configurations, with exception of Random Access without hierarchical mode, our proposed model achieves a reduction in the bitrate. Nevertheless, if all BD PSNR and BD-Rate values are summed, the result is still a gain in PSNR and a reduction in the bitrate. Concerning the bitrate error (the difference to the bitrate target), our model obtained a lower bitrate error with Hierarchical mode. When the Hierarchical mode was inactive, Random Access was the only configuration who performed better than the reference software.

We can then conclude that our R-λ model (4.24) outperforms the model used in the reference software of SHVC for the upper layer.

Table 5.10 - Average values of BD-PSNR, BD-Rate and Bitrate error.

	No Hierarchical			Hierarchical		
	Low Delay P	Low Delay	Random Access	Low Delay P	Low Delay	Random Access
BD PSNR	0.016	0.025	-0.0173	0.03	0.03	0.021
BD Rate	-1.01%	-1.81%	2.18%	-0.61%	-2.15%	-1.96%
SHM bitrate error	0.06%	0.06%	1.26%	0.08%	0.09%	0.40%
Proposed RC bitrate error	0.10%	0.08%	-0.22%	-0.09%	-0.04%	-0.50%

6 Conclusion and Future Work

In this dissertation, we start with analyzing the new video encoder HEVC. HEVC represent a number of improvements in video coding technology. Its video coding layer design is based on conventional block-based motion-compensated hybrid video coding concepts, but with some important modifications relative to prior standards. Previous tests to this dissertation show that the features of the new design provide approximately a 50% bitrate savings for equivalent perceptual quality relative to the performance of prior standards.

Next we analyzed the scalable extension of HEVC, SHVC. In contrast to previous scalable coding standard SVC, the EL codec in SHVC can be built by repurposing existing single-layer HEVC codec cores. SHVC supports conventional scalability features such as temporal, spatial, and SNR scalability, as well new scalability features such as hybrid codec, bit depth, and CGS. In the analysis of the results obtained, we showed that for a spatial scalability of 1.5x there is a reduction of 33.6% of the encoding time compared to a HEVC simulcast configuration. We also verified that for a QP equal to 32 in the EL, we always got the higher time reduction. About the bitrate cost, in overall there is a reduction of 28.7%, however the reduction cost is dependable of the values of the QP used in the BL and EL. If the ΔQP is negative, the bitrate reduction is smaller. On the other hand, if the ΔQP is positive, the bitrate reduction is bigger. For ΔQP between 4 and 8 we obtained higher time and bitrate reduction. We also concluded that the distortion of the EL only suffer a minor degradation compared to the EL of the HEVC simulcast configuration. SHVC is an intricate and complex standard and the analysis in this work can be taken as a point of entry to analyze the performance of SHVC with more than two layers.

As video consumption continues to quickly grow over heterogeneous networks and on a wide range of devices, and as new video formats such as UHD, high dynamic range, and wide color gamut emerge, scalability could prove to be attractive to many video applications. Therefore, we analyze the λ-domain rate control adopted in HEVC for the case when is used in the scalable extension, SHVC. The R-D model used to characterize the relationship between R and λ for the case of single-layer is the hyperbolic model. However, we proved that the model used for single-layer is not the best model for the EL. We determined that the logarithmic model characterize the relationship between R and λ far better for the EL. Ours results demonstrated that for the Low Delay's configuration and Random Access configuration with hierarchical mode, our model performed better than the reference software, while without hierarchical mode only the Low Delay's configuration performed better than the reference software. Concerning the future work to improve our R-D model, we suggest determining the percentage of CUs predicted from the lower layer, since it may justify why our R-D model is worse for Random Access with no hierarchical configuration.

7 References

[1] Youtube, “YouTube Statistics,” March 2015. [Online].

[2] Reelseo, “500 hours of video uploaded to Youtube every minute,” November 2015. [Online].

[3] U. Reimers, Digital Video Broadcasting (DVB) - The International Standard for Digital Television, New York: Springer, 2004.

[4] G. Faria, J. Henriksson, E. Stare and P. Talmola, “DVB-H: Digital broadcast services to handheld devices,” *Proceeding of IEEE,* vol. 94, 2006.

[5] ITU-T and ISO JTC 1, “Advanced Video Coding for Generic Audiovisual Services,” [Online]. Available: http://www.itu.int/rec/T-REC-H.264.

[6] A. Luthra, G. Sullivan and T. Wiegand, “Special Issue on the H.264/AVC Video Coding Standard,” *IEEE Trans. Circuits Systems Video Tech.,* vol. 13, 2007.

[7] MPEG, “Vision, applications and requirements for high efficiency video coding (HEVC),” in *ISO/IEC/JYC1/SC29/WG11 N11872*, Daegu, South Korea, 2007.

[8] B. Bross, W. Han, G. Sullivan, J. Ohm and T. Wiegand, *High Efficiency Video Coding (HEVC) Text Specification Draft 9, document JCTVC-k1003,* ITU/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), 2012.

[9] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, “RTP: A transport protocol for real-time applications,” *RFC 1889,* 1996.

[10] *Generic Coding of Moving Pictures and Associated Audio Information - Part2: Video,* ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG-2 Video), ITU-T and ISO/IEC JTC 1, 1994.

[11] *Video Coding for Low Bit Rate communication,* ITU-T Rec. H.263, ITU-T, 2000.

[12] H. Schwartz, T. Hinz, H. Kirchhoffer, D. Marpe and T. Wiegand, *Technical Description of the HHI Proposal for SVC CE1,* ISO/IEC JTC1/SC 29/WG 1, Doc. M11244, 2004.

[13] H. Schwarz, D. Marpe and T. Wiegand, “Overview of the scalable video coding extension of the H.264/AVC standard,” *IEEE Trans. Circuits Syst. Video Tech.,* vol. 17, pp. 1103-1120, 2007.

[14] J. Chen, J. Boyce, Y. Ye, M. Hannuksela, G. Sullivan and Y. Wang, “High efficiency video coding (HEVC) scalable extension Draft 7,” in *Joint Collaborative Team on Video Coding, JCTVC-R1008 v7*, Sapporo, 2014.

[15] H. Everett, “Generalized Lagrange multiplier method for solving problems of optimum allocation of resource,” *Oper. Res.,* vol. 11, pp. 399-417, 1963.

[16] R. E. Bellman, *Dynamic Programming,* Princeton University, Princeton, 1957.

[17] ISO/IEC-JTC1/SC29/WG11, *MPEG Video Test Model 5 (TM-5),* document MPEG93/457, 1993.

[18] ITU-T SG16/Q15, *Video Codec Test Model, Near-term, Version 8 (TMN8),* Portland: Document Q15-A-59, 1997.

[19] J. Ribas-Corbera and S. Lei, “Rate control in DCT video coding for low-delay communication,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 9, pp. 172-185, 1999.

[20] T. Chiang and Y. Zhang, “A new rate control scheme using quadratic rate distortion model,” *IEEE Trans. Circuits Syst. Video Tech.,* vol. 7, pp. 246-250, 1997.

[21] Video Group, *Text of ISO/IEC 14496-2 MEPG-4 Video VM-Version 8.0,* Sweden: ISO/IEC/JTC1/SC29/WG11 Coding of Moving Pictures and Associated Audio MPEG 97/W1796, 1997.

[22] B. Li, H. Li, L. Li and J. Zhang, “λ Domain Rate Control Algorithm for High Efficiency Video Coding,” *IEEE Transactions on Image Processing,* vol. 23, pp. 3841-3854, 2014.

[23] G. J. Sullivan, J.-R. Ohm, W.-J. Han and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard,” *IEEE Trans. Circ. Sys. Video Tech.,* vol. 22, pp. 1649-1668, 2012.

[24] R. Sjöberg, T. Chen, A. Fujibayashi, M. Hannuksela, J. Samuelsson, T. Tan, Y.-K. Wang and S. Wenger, “Overview of HEVC High-Level Syntax and Reference Picture Management,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, pp. 1858-1870, 2012.

[25] V. Sze, M. Budagavi and G. Sullivan, High Effiency Video Coding (HEVC): Algorithms and Architectures, Springer, 2014.

[26] M. Wien, High Efficiency Video Coding: coding tools and specifications, Springer, 2015.

[27] S. Wenger, “H.264/AVC over IP,” *IEEE Trans.Circuits Syst. Video Technol.,* vol. 13, pp. 645-656, 2003.

[28] R. Sjöberg and J. Samuelsson, “Absolute Signaling of Reference Pictures,” in *JCTVC-F493*, Turin, 2011.

[29] I.-k. Kim, J. Min, T. Lee, W.-J. Han and J. H. Park, “Block Partitioning Structure in the HEVC Standard,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, pp. 1697-1706, 2012.

[30] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, “Introduction to Algorithms,” *Cambridge, MA: MIT Press,* 2001.

[31] J. Lainema, F. Bossen, W.-J. Han and J. Min, “Intra Coding of the HEVC Standard,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1792 - 1801, 2012.

[32] J. Lainema and K. Ugur, “Angular Intra Prediction in High Efficiency Video Coding (HEVC),” in *Multimedia Signal Processing (MMSP), 2011 IEEE 13th International Workshop on*, Hangzhou, 17-19 Oct. 2011.

[33] ISO/IEC JTC1/SC29/WG11 N14970, *High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Improved,* Strasbourg, 2014.

[34] V. Sze and M. Budagavi, “High Throughput CABAC Entropy Coding in HEVC,” *IEEE Trans. Circ. Syst. Video Techn.,* vol. 22, pp. 1178-1791, 2012.

[35] A. Norkin, G. Bjotegaard, A. Fuldseth, M. Narroschke, M. Ikeda, K. Andersson, M. Zhou and G. V. d. Auwera, "HEVC deblocking filter," *Circuits and Systems for Video Technology,* vol. 22, pp. 1746-1754, 2012.

[36] C. M. Fu, E. Alshina, Y. W. Huang, C. Y. Chen, C. Y. Tsai, C. W. Hsu, S. M. Lei, J. H. Park and W. J. Han, "Sample adaptive offset in the HEVC standard," *Circuits and Systems for Video Technology,* vol. 22, pp. 1755-1464, 2012.

[37] ISO/IEC JTC1/SC29/WG11 N 14971, *Scalable HEVC (SHVC) Test Model 8 (SHM 8),* Strasbourg, 2014.

[38] J. M. Boyce, Y. Ye, J. Chen and A. K. Ramasubramonian, "Overview of SHVC: Scalable Extensions of the High Efficiency Video Coding Standard," *IEEE Trans. Circ. Syst. Video Techn.,* vol. 26, pp. 20-34, 2016.

[39] P. Helle, H. Lakshman, M. Siekmann, J. Stegemann, T. Hinz, H. Schwraz, D. Marpe and T. Wiegand, "A scalable video coding extension of HEVC," in *Data Compression Conference*, Berlin, 2013.

[40] G. J. Sullivan, "JCT-VC AHG Report: Multi-layer Picture Order Count Derivation (AHG10)," document JCTVC-P0010, San Jose, 2014.

[41] J. Boyce, S. Wenger, W. Jang, D. Hong, Y.-K. Wang and Y. Chen, "High Level Syntax Hooks for Future Extensions," JCTVC-H0388, San Jose, 2012.

[42] R.-J. Wang, J.-T. Fang, Y.-T. Jiang and P.-C. Chang, "Quantization-Distortion Models for Interlayer Predictions in H.264/SVC Spatial Scalability," *IEEE TRANSACTIONS ON BROADCASTING,* vol. 60, no. 2, pp. 413-419, 2014.

[43] P. Bordes, P. Andrivon and R. Zakizadeh, "AHG14: Color Gamut Scalable Video Coding Using eD LUT," document JCTVC-M0197, Incheon, 2013.

[44] Y. He and J. Dong, "SCEI: Combined Bit Depth and Color Gamut Conversion With 3D LUT for SHVC Color Gamut Scalability," document JCTVC-P0186, San Jose, 2014.

[45] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Process. Mag.,* vol. 15, pp. 23-50, 1998.

[46] B. Li, J. Xu, D. Zhang and H. LI, "QP refinement according to Lagrange multiplier for high efficiency video coding," *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS),* pp. 477-480, May 2013.

[47] H. Wang and S. Kwong, "Rate-distortion optimization of rate control for H.264 with adaptive initial quantization parameter determination," *IEEE Trans. Circuits Syst. Video Technol,* vol. 18, pp. 140-144, January 2008.

[48] A. Santiago and A. Navarro, "A new λ-R model for the Scalable High Efficiency Video Coding standard," *submitted to Electronics Letters.*

[49] H. Schwarz, D. Marpe and T. Wiegand, "Hierarchical B pictures," in *Document JVT-P014*, Pozna, 2005.

[50] G. Bjontegaard, “Calculation of Average PSNR Differences between RD Curves,” *ITU-T Q6/SG16 (VCEG),* 2001.

[51] "SHM-8.0," [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_SHVCSoftware/.

[52] “HM-16.6,” [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/.

[53] V. Seregin and Y. He, “Common SHM test conditions and software reference configurations,” in *Joint Collaborative Team on Video Coding, JCTVC-Q1009*, Valencia, 2014.

[54] “Xiph.org Video Test Media,” [Online]. Available: https://media.xiph.org/video/derf/. [Accessed 2015].

[55] A. Santiago and A. Navarro, “Optimal Bitrate for the Scalable High Efficiency Video Coding (SHVC),” in *approved to CLME2017*.

[56] J. Ugur, A. Alshin, E. Alshina, F. Bossen, W. Han, J. Park and J. Lainema, “Motion compensated prediction and interpolation filter design in H.265/HEVC,” *IEEE J Sel Top Sinal Process,* pp. 946-956.