FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Systematic Analysis of Super Hi-Vision Systems

Francisco Maria de Castro Rodrigues Vieira de Araújo

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Maria Teresa Magalhães da Silva Pinto de Andrade (PhD)

July 31, 2014

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES 2013/2014

A Dissertação intitulada

"Systematic Analysis of Super Hi-Vision Systems"

foi aprovada em provas realizadas em 23-07-2014

o júri

Presidente Professor Doutor Ricardo Santos Morla
Professor Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor André Ribeiro da Silva Marçal
Professor Auxiliar do Departamento de Matemática da Faculdade de Ciências da Universidade do Porto

Doutor Maria Teresa Magalhães da Silva Pinto de Andrade
Professora Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projeto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são corretamente citados.

Autor - Francisco Maria de Castro Rodrigues Vieira de Araújo

Faculdade de Engenharia da Universidade do Porto

Resumo

A *Ultra High Definition* (UHD) é uma tecnologia em desenvolvimento que começou a ser estudada pela NHK Science & Technology Research Laboratories, a organização de Radiodifusão pública japonesa, em 1995. Esta tecnologia consiste em dois formatos digitais: o 4K UHD ($3,840 \times 2,160$ pixeis) e 8K UHD ($7,680 \times 4,320$ pixeis), ou também chamado de *Super Hi-Vision* no Japão, ambos aprovados pela União de Telecomunicações Internacional. Estes formatos aparecem como a evolução natural aos sistemas atuais de Alta Definição e como uma tecnologia alternativa ao 3D. Para tentar marcar essa transição de tecnologia, deixou-se de categorizar a resolução da imagem de acordo com a sua resolução vertical, como 720p ou 1080p em *Alta Definição*(HD), para uma designação de resolução horizontal aproximada, dando origem aos termos 4K UHD e 8K UHD.

Hoje em dia o formato 4K começa a entrar nos mercados internacionais, com cada vez mais empresas a comercializar televisões a suportar essa resolução de pixeis. Contudo, dado que a quantidade de conteúdo disponível para 4K ainda é escasso, o volume de vendas é também bastante limitado, não tendo ainda ganho popularidade entre os consumidores. No entanto, a industria de cinema adotou oficialmente o 4K como o formato genérico para filmar, e preparam-se para vender *Blu-Ray* que suporta essa tecnologia, criando conteúdo para esses mesmos consumidores. O formato de 8K, apesar de já haver as normas e a tecnologia necessárias para a sua produção, ainda está em desenvolvimento. O grande desafio relativamente a este formato é precisamente devido à quantidade massiva de informação a ser processada por segundo, tendo 16 vezes mais pixeis que HD e 4 vezes mais que o formato 4K. Foi necessário então desenvolver um *codec* que fosse mais eficaz que o *codec AVC/H.264* usado em compressão atualmente. O resultado foi o desenvolvimento do *HEVC (High Efficiency Video Coding)*, que obtém o dobro do rácio de compressão do seu antecessor mantendo a mesma taxa de transferência de bits.

Atualmente, os laboratórios de pesquisa da NHK conseguem atingir débitos de transmissão de sinal 8K em tempo real, com um débito de *85 Mbps*. Isto leva a uma impressionante compressão com um rácio de 350:1, que ainda é insuficiente para transmitir o sinal através da rede IP. No entanto, a implementação deste sistema para a casa do assinante está prevista para os anos 2020 com o princípio de testes de transmissão via satélite em 2016, no Japão.

O objetivo desta dissertação é fazer uma análise sistemática sobre o estado atual do sistema de *Super Hi-Vision*, ou também chamado de modo 8K de *Ultra High Definition*, analisando desde as características principais do sinal de vídeo até à transmissão e apresentação de conteúdos, através de técnicas de pré-processamento e codificação. O *codec HEVC* também irá ser detalhado com pormenor de modo a observar as novas técnicas e métodos de compressão face a *codecs* anteriores. O objetivo é perceber como é que esta tecnologia está a ser desenvolvida, identificando os principais desafios a serem abordados, as suas limitações e sugerir maneiras de ultrapassa-los.

Abstract

Ultra High Definition (UHD) is a technology still under development, which was first investigated by NHK Science & Technology Research Laboratories, Japan's public Broadcasting Organization, in 1995. This technology consists in two digital formats, the 4K UHD ($3,840 \times 2,160$ pixels) and the 8K UHD ($7,680 \times 4,320$ pixels), or commonly known as *Super Hi-Vision* in Japan, approved by the International Telecommunication Union. These formats appear as the natural evolution to the current High Definition system and an alternative technology to 3D. To emphasize the leap in technology generation, the media categorization was changed from referring it according to the vertical resolution, such as 720p or 1080p in High Definition, to represent it by the approximate horizontal resolution, giving origin to the term 4K UHD and 8K UHD.

Nowadays, the 4K format is gradually entering the consumer market with increasingly more companies commercialising televisions that support that pixel resolution. However, given the amount of content available in 4K is still quite limited, the volume of sales is also low, having yet to gain popularity. Nevertheless, the film industry has officially adopted the 4K format as a recording standard, allowing the sales of *Blu-Ray* to support that technology, creating content for consumers to enjoy. The 8K format, although already possessing all the necessary technology and standards to enter into production, is still being optimized by a number of companies. The big challenge concerning this format is due precisely to the extremely high volume of data to process per second, as it has 16 times more pixels than HD and 4 times more than the 4K format. It was necessary to develop a more effective and efficient *codec* than the *AVC/H.264* widely used in compression nowadays. The result was the development of the *HEVC (High Efficiency Video Coding)*, which is said to have a double compression ratio when compared to its predecessor, allowing better video quality while using, virtually, the same bit rate.

NHK Science & Technology Research Laboratories have successfully encoded in real time, an 8K signal with a bit rate of *85 Mbps*. This leads to an impressive compression ratio of 350:1, which is still insufficient to transmit it through IP based networks. However, Japan is planning to implement this system in consumer homes around the year 2020, beginning with satellite transmission tests in 2016.

The objective of this dissertation is to preform a systematic analysis on the current status of the *Super Hi-Vision* system, also referred by 8K *Ultra High Definition*, from the main characteristics of the video signal, to the transmission and presentation approaches, through the pre-processing and encoding techniques. Within this context, the *codec HEVC* will also be thoroughly detailed with the intent of observing the new features and methods of compressing data comparing to with the previous generations of *codecs*. The objective is to understand how this technology is being developed, identifying the main challenges it has to address its limitations and to describe and suggest ways to overcome them.

Acknowledgement

I would like to thank my supervisor Prof. Maria Teresa Andrade of the Faculty of Engineering of University of Porto, for the patience, insightful advice and constructive feedback during the entire period of the research and development. I would also like to thank all my Professors from Faculty of Engineering of University of Porto who supported this journey throughout my years in University.

To all my friends, I would like to thank you for all your fellowship in difficult times and for your support in all my decisions made during all these years, even the crazy ones. I would specially like to thank my friend Diogo Barbosa, for the past 10 years of over complicated projects and for the help and patience in helping me develop test videos and examples used for this dissertations.

Last but not least, I would like to thank my parents and family for their unconditional support during all these years. Their help and patience taught me to exert myself in order to accomplish success in life.

Francisco Vieira de Araújo

"At the age old pond
a frog leaps into water
a deep resonance"

Matsuo Bashou

Contents

List of Figures

List of Tables

Abbreviations and Symbols

APSK	Amplitude and Phase Shift Keying
AVC	Advanced Video Coding
CABAC	Context Adaptive Binary Arithmetic Coding
CIE	Commission Internationale de l'Eclairage
CTB	Coding Tree Block
CTU	Coding Tree Unit
CB	Coding Block
CU	Coding Unit
DBF	De-blocking Filter
DCT	Discrete Cosine Transform
DST	Discrete Sine Transform
EBU	European Broadcasting Union
FFT	Fast Fourier Transform
FOV	Field Of View
HDTV	High Definition Television
HEVC	High Efficiency Video Coding
ISP	Internet Service Provider
ITU-R	International Telecommunication Union Radiocommunication Sector
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
LDPC	Low Density Parity Check
NAL	Network Abstract Layer
MIMO	Multiple Input Multiple Output
MPEG	Moving Picture Experts Group
NHK	Nippon Housou Kyoukai
OFDM	Orthogonal Frequency Division Multiplexing
PB	Prediction Block
PSK	Phase Shift Keying
PSNR	Peak Signal-to-Noise Ratio
PU	Prediction Unit
QPSK	Quaternary Amplitude and Phase Shift Keying
RGB	Colour space composed by red, green and blue values.
RTP	Real-time Transfer Protocol
RZ-DQPSK	Return-to-zero Differential Quadrature Phase Shift Keying

SFN	Single Frequency Network
SHV	Super Hi-Vision
SMPTE	Society of Motion Picture and Television Engineers
STC	Space Time Coding
TB	Transformation Block
TU	Transformation Unit
TV	Television
UHDTV	Ultra High Definition Television
VCEG	Video Coding Experts Group
WDM	Wavelength Division Multiplexing
WPP	Wavefront Parallel Processing
YC_BC_R	A process of encoding RGB composed by luma components (Y) and blue and red difference chroma components
YUV	Similar way to represent the luma (Y) and the chroma component (U and V) colour space

bps	Refers to the number of bits per second being transmitted
Mbps	Mega bits per second, or 10^6 bits per second
Mpx	Mega Pixels, or 10^6 pixels
Gbps	Giga bits per second, or 10^9 bits per second
Hz	Hertz or frequency of frames per second

Chapter 1

Introduction

This chapter provides a contextualization of the objectives of this dissertation and the work developed within, providing relevant background information to its reader. It also describes the problem this dissertation aims to address, the motivation behind the topic, the main goals to achieve towards the desired solution and the methodologies used towards achieving that solution. This chapter finalizes by describing the structure of this document, including a short description of each chapter.

1.1 Context

The need for higher quality video is a constant pursuit, specially given the rapid expansion of media content being delivered to the home users. The current generation of *High Definition Television* (HDTV) has been in the market for quite a few years, but it was only in the last decade that it started reaching the consumers with attractive prices. This was due to advances in compression technology, leading to better quality, together with improvements in transmission technology. Notably, the analogue to digital switching and the introduction of fibre optics has made possible to deliver HDTV content to the consumer's house. The combination of bigger and more affordable television sets, an optimized medium where information could travel with higher bit rates together with the development of better compression algorithms, resulted in the mass adoption of HDTV as a standard for consumer's image resolution. In parallel to that, the implementation of fibre optics allowed ISP [1] to provide home users with faster internet, which had a big impact on the content displayed all over the internet. Compared to standard definition, the largest spatio-temporal resolution HDTV could reach as much as five times the number of pixels displayed in a single frame.

The High Definition standard was defined with several formats, among which the *720p* containing $1,280 \times 720$ pixels which is approximately 0.9 Mpx [2], or the *1080p* with $1,920 \times 1,080$ pixels, delivering around a 2.1 Mpx per frame. Alternatively, another format was develop to allow lower bit rate and still having the same image quality as the *1080p* format. This new format was

[1] ISP refers to Internet Service Provider, companies who deliver internet to consumer houses provided they pay a fee.

[2] Mpx stands for Mega pixels, which refers to an image resolution with at least 10^6 pixels per frame.

defined as *1080i*, where *i* stands for *interlaced* as opposed to the *progressive* technique applied to the other formats, meaning it is able to show only the even image lines in one half-frame period alternating with the odd lines on the next half-frame period, recreating the full image by doubling the frame rate, without degrading the user's perceived quality. This format is usually either composed by $1,920 \times 1080$ pixels or 1440×1080 pixels, both being scanned using interlaced methods, thus presenting 540 lines during each half-frame period.

Another relevant aspect of consumer's influence begins with the film industry trying to revive the 3D [3] format which appeared initially around 1952 festered with the development and implementation of high definition standards. By using a combination of high definition image and 3D, film makers could record their films directly to 3D rather than a post-conversion, giving a more sense of realism and perception to the final output. The apex occurred in 2010 with the film *Avatar*, by James Cameron, which made the technology popular amongst other film makers and the general audience. Such trend made the TV industry focus their resources into creating televisions that supported both high definition and 3D technology. However, recent trends show that consumers are getting tired of this technology. This is due to the problems that were inherited from past iterations such as the need to use polarized glasses, limiting consumer's experience by the number of glasses they possess, and the health problems observed [11, 12] due to long time exposure.

With the decline use of 3D technology, the previous generation of picture resolution reached its maximum stage, with no possibility of improving. With today's encoding algorithms and fibre optics cables, the television manufacture industry decided to develop the next generation picture resolution standard called *Ultra High Definition* or *UHD*. Ultra High Definition is a technology still being developed, that began being researched by NHK Science & Technology Research Laboratories, Japan's public Broadcasting Organization, in the year 1995. This technology consists in two digital formats, the 4K UHD ($3,840 \times 2,160$ pixels) and the 8K UHD ($7,680 \times 4,320$ pixels), or commonly known as *Super Hi-Vision* in Japan, and was approved by the International Telecommunication Union. To emphasize the leap in technology generation, the media categorization was changed from referring it according to the vertical resolution, such as 720p or 1080p in High Definition, to describe it by the approximate horizontal resolution, the 4K UHD and 8K UHD.

1.2 Goals and Motivation

With the constant global consumer demand for better quality image content, television manufacturers are beginning to sell televisions that support the 4K UHD format, while the film industry has officially adopted the 4K format as a recording standard. However, due to the lack of media content such as films or television broadcast and due to the high price established per television equipment, it has not yet have a significant market penetration. Consumers are still buying high definition resolution televisions and waiting for the technology to mature. But the real problem

[3] 3D refers to a technique of delivering stereoscopic or three-dimensional images used in motion pictures to enhance illusion and depth perception.

behind the 4K UHD is not the lack of content available to home user but rather the huge bit rate it produces. Using the *H.264/MPEG-4 AVC* [4] to compress a 4K UHDTV sequence at 50 frames per second, the result is a whopping 16.5 Mbps bit rate sequence. It is a significant increase when compared to HD sequences, which requires around 6.5 Mbps for 720p and 9 Mbps for 1080p, at approximately the same objective quality level, measured through the PSNR[5] [13].

An uncompressed 4K UHD sequence, using the standards specification, can produce a maximum bit rate around 36 Gbps. The Super Hi-Vision format, having 4 times the spatial resolution of 4K UHD mode, generates a 144 Gbps bit rate sequence. Even when compressed with *H.264/MPEG-4 AVC*, the generated output bit rate is still extremely high to allow its delivery to the consumer's home through the available transmission channels. The need to have a better compression and transmission algorithm to apply to UHD signals, resulted in the development of the *HEVC* [6], which can achieve 50 % more data compression ratio when compared to *H.264/MPEG-4 AVC* at the same level of video quality [6]. NHK Science & Technology Research Laboratories have successfully encoded in real time an 8K signal with an output bit rate of *85 Mbps*, arriving to a compression ratio of 350:1, using *HEVC*. The compressed output was still far too big for today's transmission methods, even with dedicated fibre optics. Japan is planning to start broadcasting Super Hi-Vision signals using satellites on multiple channels [4].

The motivation for this dissertation is based on the problems Super Hi-Vision signal format inherits. Since an uncompressed signal can reach up to 144 Gbps and even with the current development of *HEVC*, the output bit rate is still too large to allow distributing the signal to the consumer's house using the available transmission facilities, forcing companies to update their equipment and transmission lines in order to support such technology. By improving the compression algorithm, broadcast companies can re-define and update their software, relaxing the channel bandwidth requirements while improving the image quality in relation to the current digital formats without the need to upgrade the equipment. The goal of this dissertation will not be to improve the HEVC algorithm, but rather to investigate different ways of applying its compression tools as well as possible pre-processing operations to apply, aiming at improving the algorithm's overall efficiency.

The fulfilment of such objective led us through an extensive analysis of the relevant state-of-the-art and consequently to the definition of the methodology to overcome to identified challenges. The first aspect that was dealt with, was obtaining the adequate source material, i.e, the video content with 8K spatial-temporal resolution. This was solved by assembling a video with the identical colour compression, bit depth used per colour and frame rate used by NHK Science & Technology Research Laboratories' experiment in collaboration with Mitsubishi Electric [4] [14], resulting in an accurate base line to compare results. Accordingly, the source material obtained was compressed using the HEVC reference software and the resulting PSNR was measured. Consequently a number of alternative approaches to apply the compression tools was investigated, together with

[4] H.264/MPEG-4 AVC is the current standard for recording, compressing and distribution of video content.

[5] PSNR, standing for Peak signal-to-noise ratio, is a logarithmic scaled ratio that represents the approximation of the human perception of the image quality.

[6] High Efficient Video Coding, currently being developed by ISO/IEC MPEG and ITU-T.

pre-processing operations to submit the signal prior to the compression. The work led to experiment with different alternatives of dividing each image of the video sequence, applying the compression tolls in parallel to each division and reconstruction back the complete images. A linear and non-linear approach to segment the video was developed. The linear method uses a blind approach where it slices the video using a fixed step, disregarding the content. The non-linear method uses an informed approach extracting the video's characteristics that provide relevant information. Each method was tested and the resulting PSNR and SSIM[7] was measured and compared to the basic compression mode. Additionally, other parameters were evaluated such as processing time as well as a subjective comparison. The results yield show it is possible to obtain improvements by conducting an "informed" partition of the image.

1.3 Document Structure

This dissertation is organized by six chapters and four appendix.

The Chapter 2 describes the State of the Art with all the relevant information about the Super Hi-Vision signal composition, audio, ecosystem and data compression.

The Chapter 3 demonstrates the problem this dissertation presents and how that same problem is tackled.

The Chapter 4 goes into detail and explain all aspects of the implemented algorithm.

The Chapter 5 displays the results produced by the proposed algorithm.

The Chapter 6 presents the conclusions of the developed work as well as some considerations regarding future development of the implemented algorithm.

The appendix A, B, C and D exhibits a multitude of images that portray to each implemented algorithm, in order to enhance and evaluate the results.

[7] SSIM stands for Structural similarity and represents an index that measures the similarity between two different images.

Chapter 2

State of the Art

This chapter provides a literature review of the relevant aspects regarding the ultra high definition TV, notably, the most recent standard specifications in terms of formats and compression tools, as well as existing services, equipments and experiences . As this technology is constantly evolving and being optimized, this dissertation will focus on the last published working progress standards. The chapter is organized by defining the standard parameters for ultra high definition signals and their transmission equipment, and the algorithm developed for encoding and decoding the signal.

2.1 Present state of Ultra High Definition TV

With the creation of new technologies, one needs to find a name that's both appealing and understandable for consumers. Sometimes the term "Ultra" in technology refers to something that is beyond the normal or that's an evolution of a previous generation technology. This naming technique helps consumers distinguish technologies and to immediately understand that it brings improvements in relation to its predecessors. The name Ultra High Definition Television, or UHDTV, is no exception. This self explanatory term shows that UHDTV is a natural evolution from the previous HDTV system. The figure 2.1 clearly illustrates that same evolution of the image resolution within the TV broadcasting scenario.

The formal use of the term Ultra High Definition refers to two digital formats called 4K, offering a 3840×2160 pixels spatial image resolution, and the 8K, which has 4 times more pixel resolution than the previous format accounting 7680×4320 pixels. Those resolutions covers virtually all of the human visual field of view (FOV) providing viewers with a better visual experience and a stronger sensation of reality. Those formats are referred as an approximation of their horizontal resolution, rather than the vertical resolution used by the previous generation HDTV, because 2160p and 4320p were not very intuitive numbers and would be misleading placing a "p" after, since there is only progressive scanning and not interlaced. The change to horizontal resolution naming also emphasises the transition to a newer, and consequently, better technology generation, making it appealing for consumers. 8K UHD term is also referred as *Super Hi-Vision* in Japan as the Hi-Vision term was used for HDTV, when they've started HD broadcast it in the mid 80's.

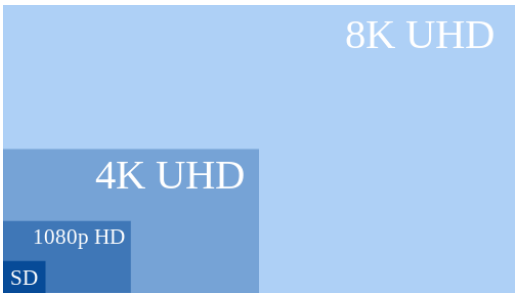


Figure 2.1: Comparison between UHDTV formats with the current and previous generation [1].

The existing standards specify parameters and constraints on spatial and temporal resolution[1], colour space, bit depth[2] and audio channel structure. And although they were developed to apply to both 4K and Super Hi-Vision formats, this dissertation will focus on the specifications and test results of the latter.

2.2 Standards

UHDTV is defined by two main standards, which were made to ensure maximum compatibility between equipment and signal composition from different Television manufacturers and media content producers. This section addresses them, giving a brief introduction regarding their content. However, their technical information will be detailed on the next sections when approaching the UHDTV's characteristics, as their content is extensive and, most of the time, has a deep interconnection.

2.2.1 ITU-R Recommendation BT. 2020

This technology started being researched by NHK Science & Technology Research Laboratories, which was defined and accepted by the *International Telecommunication Union* (ITU), a specialized agency of the United Nation, and published in August 23rd 2012 [8]. ITU is responsible for issues related to information and communication technologies, such as standardization of technology and global coordination of the radio-spectrum usage and satellite orbits. ITU is composed of three sectors that are able to manage different areas, namely the Radiocommunications Sector (ITU-R), Standardization Sector (ITU-T) and Development Sector (ITU-D). Each of those sectors are composed by several Study Groups that ensure and apply the sector's goals.

The *Recommendation BT 2020* is the UHDTV standard developed and proposed by ITU-R Study Group 6[3], Working Party 6C. This recommendation specifies the picture spatial and temporal resolution, system colorimetry, signal format and its digital representation.

[1] Spatial resolution refers to the width and height of the image and temporal resolution refers the number of images displayed per second.

[2] Number of different colours a given pixel can display.

[3] Also called by Broadcasting Service, responsible for radiocommunication broadcasting, including vision, sound, multimedia and data services principally intended for delivery to the general public.

2.2.2 SMPTE ST 2036

The *Standard 2036* is a suite of documents divided into multiple parts, developed by The Society of Motion Pictures and Television Engineers (SMPTE), an United States of America based engineering association that works in the motion imaging industry, creating standards in that area. This standard was developed to cover issues that were not defined by the Recommendation BT 2020, addressing areas such as image parameters values for program production (ST 2036-1), audio characteristics and channel mapping (ST 2036-2) and signal transmission using a single-link or multi-link data interface (ST 2036-3). This standard, however, is not yet completed as the ST 2036-3 only has a maximum frame rate capacity of 60 Hz, which is not fully compliant with the Recommendation BT 2020. Work is still undergoing to define interfaces capable of operating at 120 Hz.

2.3 Picture Parameters

The following discussion is the representation of the parameters developed for UHDTV, contemplated in both BT 2020 and ST 2036, addressing all the necessary information for displaying pictures.

2.3.1 Spatial and Temporal Characteristics

Parameter	**Values**
Picture aspect ratio	16:9
Pixel count Horizontal x Vertical	7680×4320
Sampling lattice	Orthogonal
Pixel aspect ratio	1:1 (square pixels)
Pixel addressing	Pixels are ordered from left to right in each row, and rows are ordered from top to bottom.

Table 2.1: The picture spatial characteristics as described in BT.2020 [8].

Parameter	**Values**
Frame frequency (Hz)	120, 60, 60/1.001, 50, 30, 30/1.001, 25, 24, 24/1.001
Scan mode	Progressive only

Table 2.2: The picture temporal characteristics as described in BT.2020 [8].

These characteristics, detailed in Table 2.2, will provide viewers with an increase feeling of reality and a more content immersion when used on screens with an diagonal size of at least 1.5

meters (60 inches) or bigger and for large screen (LSDI) presentations in theatres, halls and other venues such as sports venues or theme parks.

2.3.2 System Colorimetry

For backwards compatibility with HDTV systems, ST 2036-1 allows implementers to optionally adopt conventional reference primaries for UHDTV, which are consistent with Recommendation ITU-R BT.709 [4], as shown in Table 2.3 and Table 2.4. The colorimetry employed must be signalled on the interface to ensure the proper conversion.

Parameter	**Values**		
Opto-electronic transfer characteristics before non-linear pre-correction	Assumed linear		
Primary colours and reference white	**Chromaticity coordinates (CIE, 1931)**	**x**	**y**
	Red primary (R)	0.708	0.292
	Green primary (G)	0.170	0.797
	Blue primary (B)	0.131	0.046
	Reference white (D65)	0.3127	0.3290

Table 2.3: System colorimetry as described in BT.2020 and are consistent with ST 2036-1 values [8, 2].

Chromaticity coordinates (CIE, 1931)	**x**	**y**
Red primay (R)	0.640	0.330
Green primary (G)	0.300	0.600
Blue primary (B)	0.150	0.060
Reference white (D65)	0.3127	0.3290

Table 2.4: HDTV System colorimetry as described in BT.709 [9].

The supported colour gamut for each set of primary colour and reference white of both systems is illustrated in the diagram of Figure 2.2. It shows the current range of colours compared to the

[4] The Recommendation for HDTV system specifications.

new set of primaries proposed by BT 2020 and ST 2036-1, projected onto the CIE[5] 1931 RGB and CIE 1931 XYZ colour space.

As shown in the figure 2.2 bellow, UHDTV colour gamut includes 75.8% of the CIE 1931 colour space and is distributed more evenly, whereas HDTV colour gamut covers around 35.9% of the total colour. This results in an unprecedented viewing experience in various environments, giving a extended sense of reality to viewers [15].

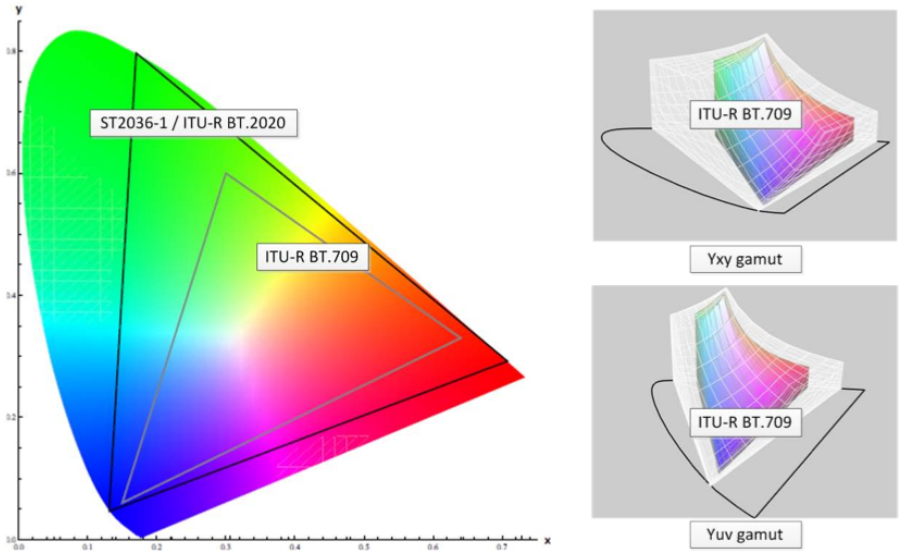


Figure 2.2: UHDTV supported colour gamut in both xy and uv, comparing to HDTV current colour gamut [2].

2.3.3 Colour Space Conversion

As previously noted, SMPTE ST 2036-1 and BT.2020 specify a common expanded colour space for 4K UHD and Super Hi-Vision formats. ST 2036-1 gives the option of supporting the BT.709 colour space to be used, considering the legacy of the current media content. This way it will always assure retro compatibility with any previous generation device without interfering with UHDTV colour space. However, in the future when the media content will be produced and post-produced for UHDTV systems, it will contain its colour space, so it is necessary to be backwards compatible, allowing colour conversion to and from legacy BT.709 and BT.601 colour spaces to comply with legacy work flows. This issue is still being currently debated and is undergoing further work, as there are not automatic gamut mapping system capable of delivering acceptable conversions [2].

[5] CIE is a French abbreviation, standing for International Commission on Illumination.

Parameter	Values
Signal format	$R'G'B'$ or $Y'C'_BC'_R$
Non-linear transfer function	$E' = \begin{cases} 4.5E, & 0 \leq \mathrm{E} < \beta \\ \alpha E^{0.45} - (\alpha - 1), & \beta \leq \mathrm{E} \leq 1 \end{cases}$ where E is the voltage normalized by the reference white level and proportional to the implicit light intensity that would be detected with a reference camera colour channel R,G, B; E' is the resulting non-linear signal of E. $\alpha = 1.099$ and $\beta = 0.018$ for 10 bit system $\alpha = 1.0993$ and $\beta = 0.0181$ for 12 bit system

Table 2.5: UHDTV signal format [8].

2.3.4 Sampling and bit-depth

Super Hi-Vision systems can employ either $R'G'B'$, $Y'C'_BC'_R$ or $Y'_CC'_{BC}C'_{RC}$ coded signal component, as detailed in Table 2.5, and may be sampled as 4:4:4, 4:2:2 or 4:2:0. This format shows how the sampling lettuce is being proceeded, displaying the luma component (light intensity) ratio and the two chroma (colour) sample ratios. 4:4:4 system refers to a situation where no sub-sampling is done, meaning each chroma component has the same number of horizontal samples as the luma, providing the best colour quality. The 4:2:2 system is a format where the horizontal chroma component is decimated by half in regard to the luma component. 4:2:0 system provides the most compression rate as the horizontal chroma component is sub-sampled by a factor of two and there is no vertical chroma component.

Each component can be sampled at a bit-depth of 10 bits (1024-bit colour) or 12 bits (4096-bit colour)[8], providing a picture with a maximum of 1.07 billion and 68.7 billion different colours, respectively. This displays a slight improvement when comparing to the current HDTV generation, having each component only being sampled with a 8 bit or 10 bit depth [9].

2.4 Audio

With the enhanced visual experience of the Super Hi-Vision, there was an opportunity to expand the audio quality providing viewers with a better, more immersed experience, while improving the overall quality of audio. The combination of audio and video should provide an unique experience to the viewer, thus marking the dawn of the new generation television format.

The current HDTV technology is capped because it uses the *Dolby AC 3* codec that supports 5.1 audio channels. To improve the audio characteristics, ST 2036-2 proposes a system that is

capable of supporting up to 22.2 multichannel audio[2], competing with the film industry current audio technology of 7.1 audio channel or even with recent 9.1 or 11.1 "immersive audio" generation, marketed for film theatres. ST 2036-2 also assures that the current 5.1 and stereo audio configurations will continue to be delivered, proposing a backward compatible system while improving the overall audio quality, providing an harmonious experience alongside UHDTV visual experience.

2.4.1 Carriage and Delivery Methods

Audio reproduction in a UHDTV system has an uncompress signal with a sample rate of 48 kHz or 96 kHz at 24 bit, as specified in SMPTE ST 2036-2, or higher resolution. In order to achieve such high interoperability, standards applied to carriage and delivery of immersive audio play a critical part in UHDTV systems. Although there is not a current standard common file format, various companies are doing effort to consolidate and develop the *BWF (Broadcast Wave Format)*, which is currently being revised by ITU-R and EBU[6]. SMPTE has currently assigned a team, TC-25CSS, whose goal is to study and implement this technology in film theatres. By using audio objects to relay the position of the audio in relation to the environment, this process was a key element in providing a basis for the common audio format to be distributed to the home users. Audio objects can also be controlled independently, for instance the voice level or background noise, allowing home listeners to adjust each volume individually, providing an extra assistance to people with hearing issues.

2.4.2 Audio Compression

The ideal audio compression used in high bandwidth distributions, such as UHDTV, should be a lossless data compression allowing the audio to be reconstructed from the compressed data, without losing any of its property. DTS Master Audio (5.1 audio channels) and Dolby True-HD (7.1 audio channels) currently carry immersive audio objects using a mathematical losseless compression algorithm. Another way to compress audio is to use a audio lossless compression format, which is basically a lossy algorithm[7], but it uses a psycho acoustic modal[8] by removing frequencies outside of a certain range, allowing a seemliness audio quality. Usually this type of algorithms have a lower bit-rate when compared to lossless algorithms.

The Moving Picture Experts Group (MPEG), are currently developing a new multimedia container that is able to compress data audio more effectively than the current widely used MPEG-4. Further detail will explained in section 2.7.

[6]EBU stands for European Broadcast Union, which is responsible for the cooperation and implementation of new telecommunication technology with EBU Members, assuring a smooth transition.

[7]Lossy refers to the fact that it losses information duration compression and it may not be recovered afterwards.

[8]Psycho Acoustic is a mathematical model of a representation for the Human ear frequency range.

2.4.3 Speakers and Sound Placement

The contemplated model to deliver the 3D Audio in a home environment includes a 22.2 multichannel audio system composed by an upper layer with 9 speakers spread evenly in a 3 by 3 fashion, a middle layer with 10 speakers, a lower layer with 3 speakers next to the television and finishing with 2 arrays of loudspeakers beside the screen, as displayed in the figure 2.3 below.

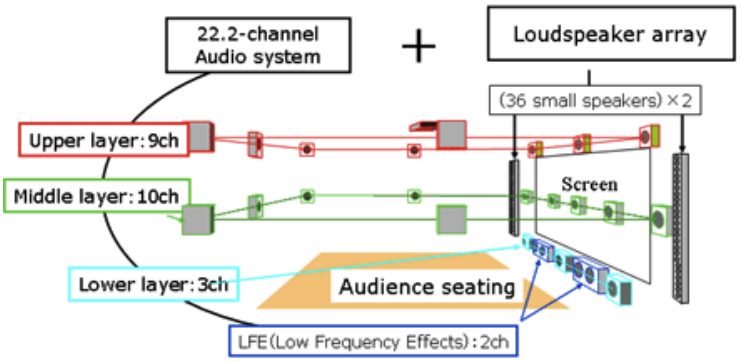


Figure 2.3: Proposed setting of speakers to achieve the 22.2 audio system[3].

The problem with this configuration, although very efficient, is that it is very unlikely that most homes have enough space to place the rather large addition of speakers correctly, in order to achieve the desired immersive sound. NHK is currently working on a new method of delivering the sound by simplifying the initial configuration and place the speakers embedded into the television itself, as exemplified in the figure 2.4. The technology developed provide a real time conversion of the 22.2 multichannel sound into several signals, which then are transmitted to the embedded loudspeaker arrays, acting as multiple speaks with multi-directional sound properties. This configuration will likely be more expensive but it will be convenient for the consumer, as there is no need for extra space and to physically install the 22 loudspeakers in the recommended fashion.

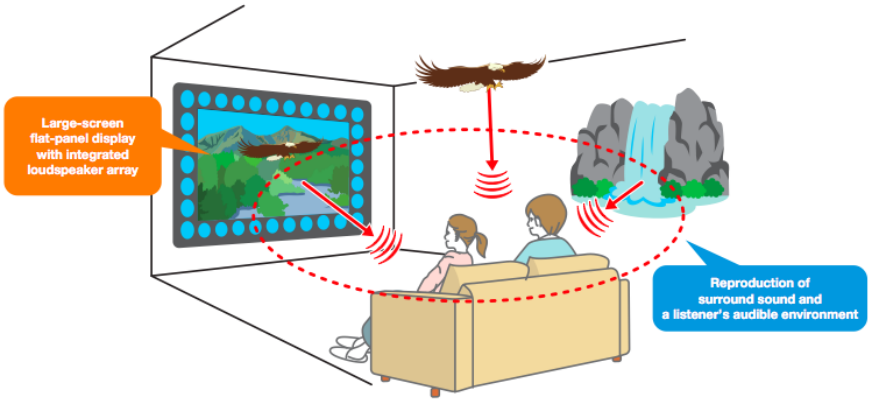


Figure 2.4: NHK Engineering System proposal for the Super Hi-Vision display with embedded audio loudspeakers [4].

2.5 Viewing Conditions

2.5.1 Field of View

With the Super Hi-Vision properties being designed to have the best viewing experience to the audience, with the superb screen resolution and 3-dimensional audio technology, one of the bottlenecks[9] of this television format generation is the field of view. The human eye have a 180 degree forward-facing horizontal field of view and UHDTV falls short of that number, as shown in Table 2.6. But it is not the intent to create a full surrounding view like virtual reality technology, but to provide with the maximum field of view where many people can share the same experience.

2.5.2 Angular resolution

When considering the "design viewing distance", one has to take into account the angular resolution. This resolution gives us the pixel per unit visual angle[10] ratio, which implies a one pixel per arc-minute[11] measurement. According to *Recommendation ITU-R 1127* [16], the relative distance to the picture at which the picture quality of each system always falls into the perfect evaluation range, where pixels cannot be distinguished and has the most quality, is the goal of designing the viewing distance. This relative distance to the picture height is an alternative expression of FOV, for the same television resolution system. The expression is described as being:

$$\tan(\frac{\theta}{2}) = \frac{r}{2 \times n} \Longleftrightarrow \tan(\frac{\theta}{2}) = \frac{r}{2 \times d \times b} [10]$$

where θ is the FOV displayed in angles, r is the aspect ratio and the n is the relationship between the design viewing distance d and picture height b.

Recommendation ITU-R BT.1845 defines the optimal viewing distance as the distance at which the pixel count per visual angle of one minute is one. It lists the optimal viewing distances relative to the screen height and the optimal field of view for image systems with various pixel counts, as shown in Table 2.6.

These two Recommendations suggest that the picture quality of an image system having viewing conditions in which the angular resolution is one pixel per one arc-minute falls into the perfect evaluation range.

[9] Expression representing the weakest link in a technology, reducing the overall quality.
[10] Angular resolution describes the ability of the eye to distinguish small details in an visual object.
[11] In a visual acuity system, a 20/20 vision equals one arc-minute.

Image system ($h \times v$)	**Reference**	**Aspect ratio** ($a : b$)	**Pixel aspect ratio** (r)	**Optimal horiz. viewing angle** (θ)	**Optimal viewing distance** (d)
720×483	Rec. ITU-R BT.601	4:3	0.88	11^o	7 H
640×480	VGA	4:3	1	11^o	7 H
720×576	Rec. ITU-R BT.601	4:3	1.07	13^o	6 H
1024×768	XGA	4:3	1	17^o	4.4 H
1280×720	Rec. ITU-R BT.1543	16:9	1	21^o	4.8 H
1400×1050	SXGA+	4:3	1	23^o	3.1 H
1920×1080	Rec. ITU-R BT.709	16:9	1	32^o	3.1 H
3840×2160	Rec. ITU-R BT.1769	16:9	1	58^o	1.5 H
7680×4320	Rec. ITU-R BT.1769	16:9	1	96^o	0.75 H

Table 2.6: Optimal horizontal viewing angle and optimal viewing distance in screen height (H) for various digital image systems [10].

As shown by this table, the recommended viewing for a Super Hi-Vision TV is a 96 degree horizontal field of view. Comparing with the human eye 180 degree FOV, Super Hi-Vision should occupy 53% of the viewing field, which is substantially superior to the 18% of a 1080 HDTV.

2.5.3 Viewing Distance

The viewing distance is referring to the recommended distance one should sit related to the screen, in order to maximize its experience. Too close and the pixel grid is visible while too far and the quality of the image deteriorates. As indicated in Table 2.6, by increasing the image resolution, the viewing distance decreases and it is directly co-related with the height of the picture. This means that by providing the same pixel count and increasing the size of the television, since the pixel density (dpi) is the same, to achieve the desired horizontal field of view, the viewing distance will have to increase as well.

Studies have shown that a typical distance from the television screen in a living room is around 2.1 meters [5], and has stayed roughly the same throughout the television generations. One can extrapolate the size of the television to achieve that optimal viewing distance in relation with the recommended diagonal screen size, as shown in Table 2.7.

4K TV screen size increases in a small proportion compared to the recommended size, however this means that the current screen height would cover large part of the living room wall. This could be considered an enhanced viewing condition. The same cannot be concluded for the Super Hi-Vision TV. The recommended size is already significantly bigger compared to a typical living room

Image system ($h \times v$)	**Recommended diagonal screen size**[5, p. 6]	**Optimal viewing distance**	**Vertical screen size of the recommended diagonal**	**Vertical screen size with a viewing distance 2.1 meters**
3840×2160	2.5 meters (100-inches)	1.9 meters	1.27 meters	1.4 meters
7680×4320	5 meters (200-inches)	1.88 meters	2.5 meters	3 meters

Table 2.7: Recommended screen size, contrasting with a typical viewing distance.

height. The need to comply with the 2.1 meter average viewing distance recommendation, would require the existence of even larger dimensions television sets, comparing to the large screens that are already being used for HDTV. That would lead to dimensions that would not be appropriate for common living rooms. However, Super Hi-Vision system may instead find applications for television presentations to the public in theatres, auditoriums, theme parks and other public venues.

2.6 Digital Ecosystem

Although standardization efforts have already begun, there is not yet a full suite of specifications for the complete Super Hi-Vision ecosystem. Accordingly, this subsection will focus on the studies being conducted by *NHK Science & Technology Research Laboratories*, since they are the most advanced company in this field and provide the most diverse research material.

2.6.1 System Model

Figure 2.5 illustrates an example of a possible ecosystem for the Super Hi-Vision system, demonstrating from the capture of the UHDTV signal to the display on the television sets in the consumer's home, as well as the interaction with previous generation systems.

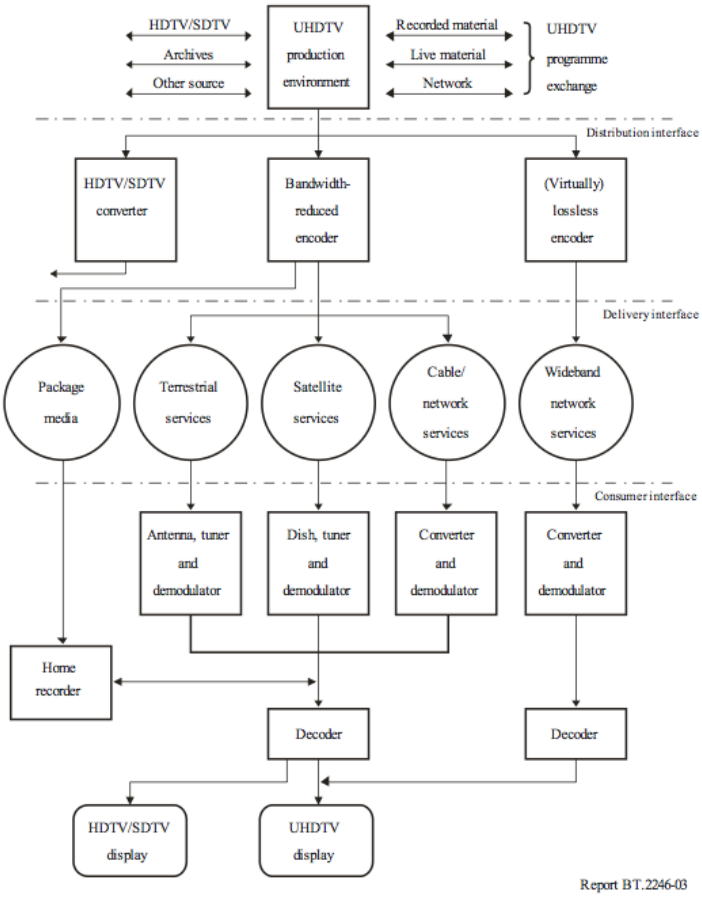


Figure 2.5: A possible model for Super Hi-Vision system [5].

2.6.2 Satellite Broadcasting

The current generation of broadcasting technology falls far from the high demands of transmitting and receiving the Super Hi-Vision compressed signals, given the huge bit rates involved. With

that intent, NHK is developing a new satellite technology with the aim of start testing broadcast emission in 2016. Assuming that the 12 GHz band satellites currently used for broadcasting information today will be borrowed for this purpose, NHK proposes a new large-capacity transmission technology. This technology can transmit Super Hi-Vision by a single satellite transponder by using a transmitter and receiver that comply with the "transmission system for advanced wideband digital satellite broadcasting" (ARIB STD-B44) [4]. The wide band digital broadcast uses a special method of signal modulation scheme called 16APSK[12], alongside the other modulation schemes already implemented today, which enables even more information to be transmitted. This scheme make the flow of data being broadcasted with existing satellites more efficient by a factor of 1.8 times [4].

NHK is also developing a wideband satellite transmission technology that will allow broadcasting of Super Hi-Vision signal on multiple channels. This technology features a transmitter and receiver which can use up to a 69 MHz bandwidth, reaching twice as much bandwidth of existing 12 GHz band satellites. This is achieved by doubling the bandwidth used per channel and satellite radiated power, which enables the possibility of transmitting information at a rate of *139 Mbps* using 8PSK[13] modulation or *186 Mbps* with the new 16APSK modulation scheme, as exemplified in Figure 2.6. NHK also developed a wideband modulator and demodulator with a 300 MHz bandwidth to be used in the 21 GHz band Satellite broadcasting, by dividing the two channels in the assigned 600 MHz range in the radio-spectrum, also illustrated in Figure 2.6. The advantages of using such wide spectrum is the possibility of transmitting QPSK[14] signals at a rate of approximately *370 Mbps* [4].

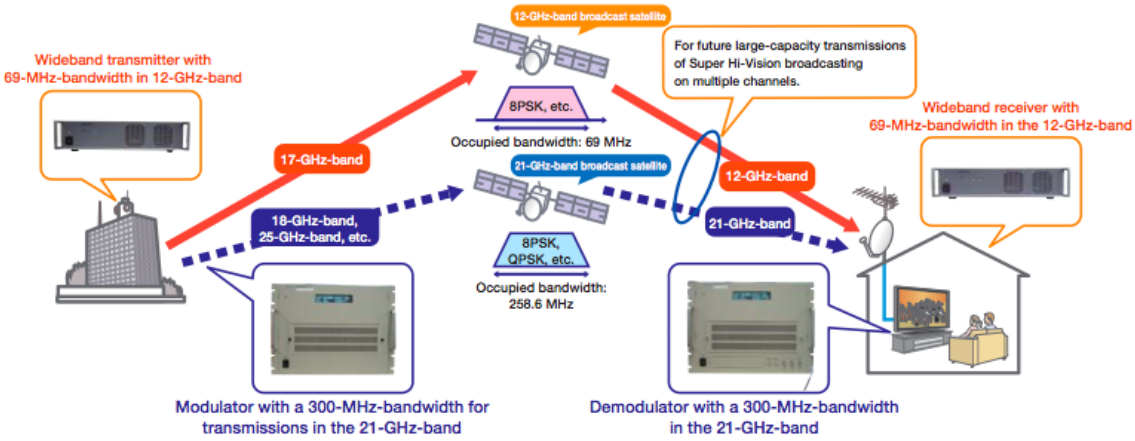


Figure 2.6: Visual concept of broadcasting in the $12-GHz$ and $21-GHz$-bands[4].

[12] 16APSK (Amplitude and Phase Shift Keying): A modulation system that can simultaneously transmit 4 bits of information by applying 16 amplitude and phase shifts to the carrier wave for transmission.

[13] 8PSK (8-Ary Phase Shift Keying): A modulation scheme that simultaneously transmits 3 bits of information by using carrier waves with eight different phases with 45-degree spacing.

[14] QPSK (Quaternary Amplitude and Phase Shift Keying): A modulation scheme that simultaneously transmits 2 bits of information by using carrier waves with four different phases with 90-degree spacing.

2.6.3 Terrestrial Broadcasting

NHK is currently conducting efforts to provide terrestrial broadcasting of Super Hi-Vision compressed signal, transmitted in a single channel by using the LDPC[15] code as the error-correcting code and experimental dual-polarized MIMO[16] and applying the "ultra multi-level" OFDM[17] transmission equipment with a 32k-point Fast Fourier Transform (FFT) [17].

In the conventional terrestrial digital broadcasting system, the Single Frequency Network (SFN) covers the service area with multiple transmitting sites operating at the same frequency, thus being efficient in the use of the radio-spectrum. To improve the transmission characteristics NHK is conducting transmission tests using a new SFN scheme in which the STC[18] method is applied to the transmission signals of adjacent transmitters, instead of a single large antenna. However, this technology is still being researched and has yet to tested.

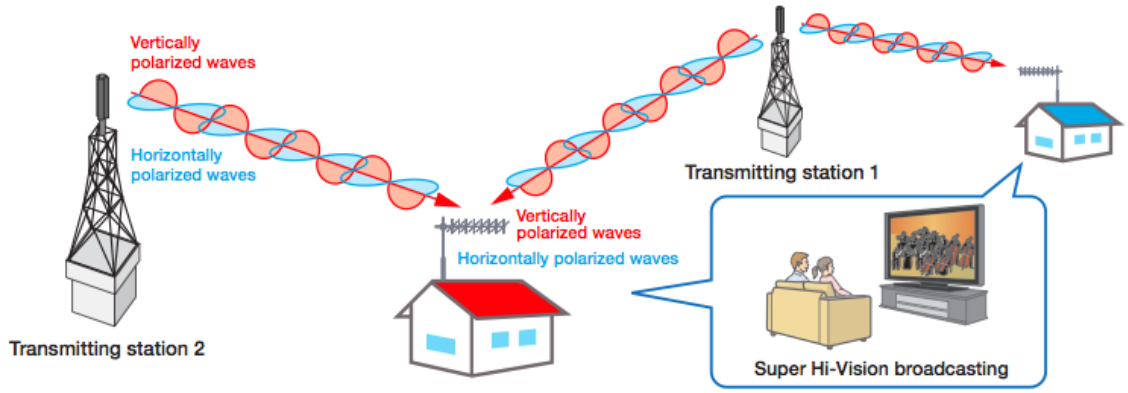


Figure 2.7: Terrestrial transmission of Super Hi-Vision by using STC in SFN [4].

2.6.4 Long Haul Optical Transmission

NHK is currently developing a long haul transmission system using optical fibre for transmitting Super Hi-Vision media content from a relay location to a broadcast station. The system is capable of converting a 72 Gbps[19] uncompressed Super Hi-Vision, which is roughly equivalent to a 64 high-definition serial digital interface (HD-SDI) signals, into two distinct 43 Gbps signals. Reed-Solomon (255, 239) error correction algorithm is then applied to each individual signal and

[15] LDPC (Low Density Parity Check): A linear error correcting code that makes it possible to obtain characteristics close to the Shannon limit. It uses a sparse parity check matrix.

[16] MIMO (Multiple-Input Multiple-Output): A wireless transmission system using multiple antennas for transmission and for reception.

[17] OFDM (Orthogonal Frequency Division Multiplexing): A digital modulation method that is used by conventional terrestrial digital broadcasting.

[18] STC (Space Time Coding): A method of encoding information temporally and spatially, then distributing the encoded signals to multiple transmission antennas.

[19] More information on data compression in Section 2.7.

posteriorly converted into RZ-DQPSK[20] optical signals of different wavelengths and transmitted over a single optical fibre using a wavelength division multiplexing (WDM) technique. This method ensures that the signal is less inclined to be affected by distortion on the transmission path and any errors caused by noise can be corrected at the reception side, which means the signal can be transmitted stably, without losing any of its content.

With existing systems, it is necessary to install optical amplifiers along the transmission line to compensate for attenuation of the optical signal power due to long-distance transmission. Pump light sources for Raman amplification[21] will be implemented in both relay site and broadcast station location, making the optical signal amplified by feeding the pump lights from both sites into the optical fibre itself. This means the transmission system does not require any optical amplifier on the transmission path and, thereby, simplifies the configuration, operation and maintenance of optical transmission system, while improving the overall quality.

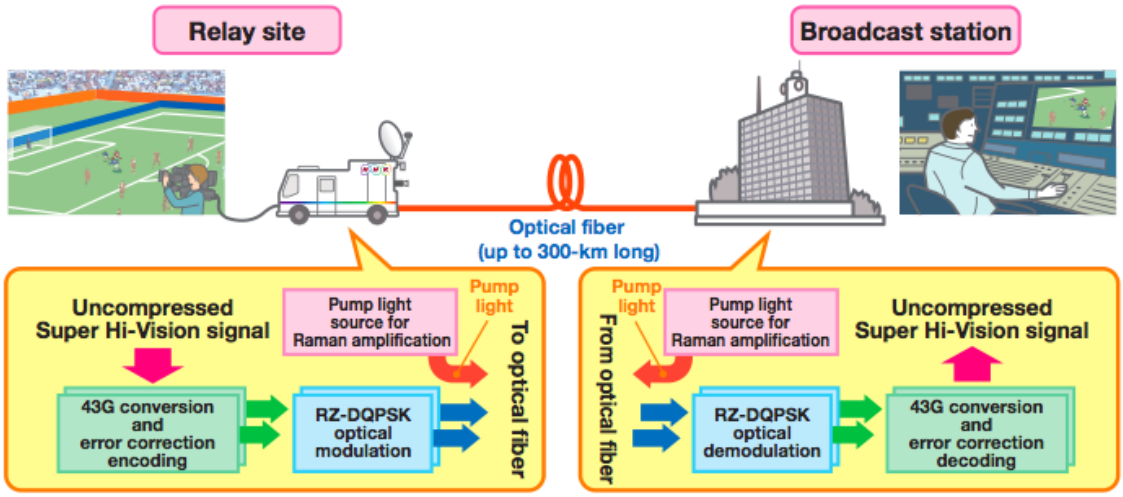


Figure 2.8: Image of transmission of uncompressed Super Hi-Vision material to the broadcasting station [4].

2.7 Data Compression

As described in Section 2.3, the requirements for Super Hi-Vision will be the highest resolution format concerning 2D video, given that the human visual system cannot distinguish higher resolutions. The high requirements imposed on Super Hi-Vision leads to a generation of extremely high bit rate streams. An uncompressed Super Hi-Vision signal is far too large to be transmitted using current network technology and transmission channels, as discussed in section 2.6.

[20] RZ-DQPSK: Return-to-Zero Differential Quadrature Phase Shift Keying.

[21] Pump light source for Raman amplification: Device for generating a pump light to amplify an optical signal within a wavelength about 100 nm longer than that of a normal pump light.

A good way to get an approximation of the bit rate generated by an uncompressed video signal is to use a simple mathematical formula:

$$\text{bit rate} = \text{width} \times \text{height} \times \text{frame rate} \times \text{bit depth per colour} \times \frac{Y'}{4} + \frac{C_b}{4} + \frac{C_r}{4} \text{ of chroma subsampling}$$

Using that formula we can extrapolate some values and see how much bit rate is generated by an uncompressed 7680×4320 Super Hi-Vision signal, complying with the Recommendation ITU-R BT.2020 specifications. One should also take into consideration that all UHDTV image scanning are progressive, unlike HDTV that supported interlaced methods, meaning each frame displays the full image. Since interlaced formats were susceptible to sudden movements or fast passed motion pictures, UHDTV standards decided to consider just progressive image scanning.

Frame Rate (Hz)	**Bit depth per colour** (bit)	**Chroma Sub sampling** ($Y'CbCr$)	**Bit rate** ($Gbps$)
60	10	4:2:0	29.86
60	10	4:4:4	59.72
60	12	4:2:0	35.83
60	12	4:4:4	71.66
120	10	4:2:0	59.71
120	10	4:4:4	119.44
120	12	4:2:0	71.66
120	12	4:4:4	143.32

Table 2.8: Uncompressed Super Hi-Vision signals with different characteristics.

As seen by the Table 2.8, a Super Hi-Vision using the maximum admissible values for its resolution parameters, achieves a colossal bit rate of 143.32 Gbps. Comparing with the 4K UHD format and using the same characteristics, the signal achieves a bit rate of 35.83 Gbps[22], which is $\frac{1}{4}$ of the bit rate produced by a Super Hi-Vision system.

The current HDTV generation mostly uses the MPEG-4 as the media container, aggregating both audio and video compression and other meta-data information. H.264/MPEG-4 part 10 or AVC is the video compression format used in MPEG-4 and was considered to be one of the key elements to provide HDTV to the mass market using the Internet or broadcasting channels, due to the compression rates it achieves. AVC made it possible to transmit high quality resolution videos exploiting the previous generation medium, as copper cables or even telephone lines, or taking advantage of fibre optics to transmit even more information. However the AVC is limited to a frame rate of 60 Hz and, although recent additions to the algorithm were made to support 4K UHD mode regarding bit depth and extended colour gamut included in Recommendation ITU-R BT.2020, AVC simply cannot obtain sufficiently high compression rates to enable the transmission of a Super Hi-Vision signal to home users. Accordingly, improvements were needed in the field of both video compression as well as transmission. The following sub-sections describe the developments that were recently accomplished in that direction by the international community.

[22] $3840 \times 2160 \times 12 \times 120 \times 3 = 35.83 Gbps$

2.7.1 ISO/IEC MPEG-H

Given the limitations of MPEG-4 to fully support the Super Hi-Vision format, the *Moving Pictures expert Group* (MPEG) of ISO/IEC decided launch an initiative to develop a suitable standard. Such initiative resulted in the MPEG-H suite of specifications, formally known as *ISO/IEC 23008 - High efficiency coding and media delivery in heterogeneous environments*. MPEG-H, which is still under development, is composed by 8 parts so far, covering areas regarding new and more efficient methods to transport information, higher compression rate algorithms, 3D audio delivery and other meta-data components. Although MPEG-H is intended to be used within UHDTV environments, it is also possible to use it with previous generation TV formats.

2.7.2 Part 1: MPEG Media Transport

MMT or MPEG Media Transport is a digital media container that can transfer multimedia content using the all-Internet Protocol (All-IP) network. This container supports Ultra HD video content, 3D video content, interactive content, user generated content, applications that support multi-device presentation, subtitles, picture-in-picture video and multiple audio tracks.

NHK Science & Technology Research Laboratories, in cooperation with ISO/IEC MPEG, has developed an experimental equipment based on MMT that can use both broadband networks and broadcasting systems together. The means of harmonizing the media transport schemes facilitates the synchronization of content to Super Hi-Vision devices, across each combination of broadband and broadcasting systems. It also allows easy presentation of video and audio signals that are designed for other types of devices such as tablets, smart-phones, computers, etc.

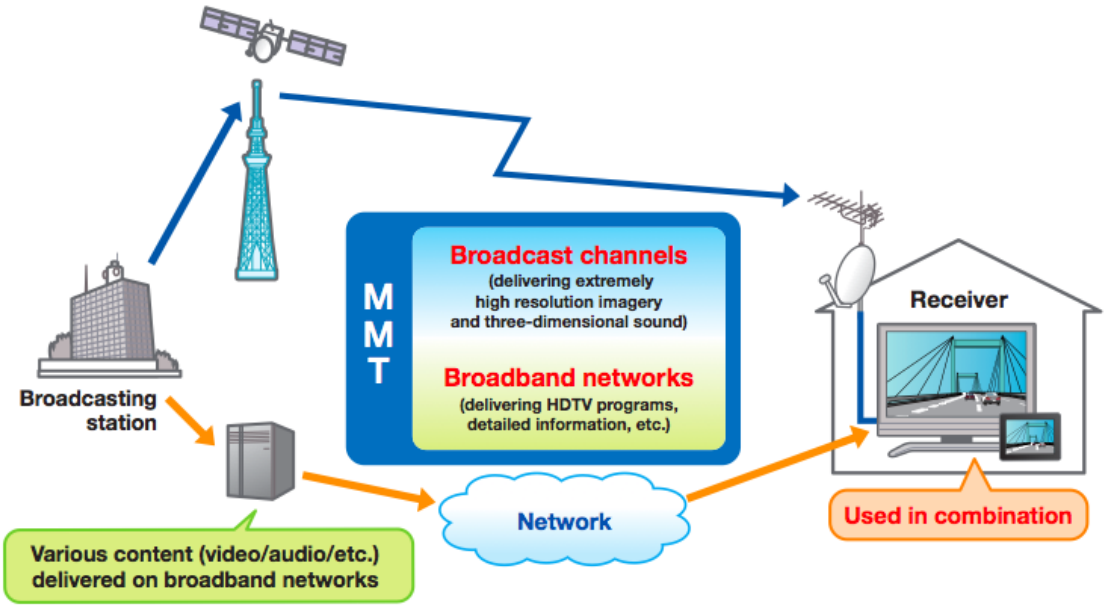


Figure 2.9: Combination of broadcasting and broadband networks [4].

2.7.3 Part 2: High Efficiency Video Coding

The High Efficiency Video Coding (HEVC) standard is the most recent joint video project of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group, working together in a partnership known as the Joint Collaborative Team on Video Coding (JCT-VC). In ISO/IEC, the HEVC standard will become MPEG-H Part 2 (ISO/IEC 23008-2) and in ITU-T it is likely to become *ITU-T Recommendation H.265*.

2.7.3.1 HEVC Standard

The video coding layer of HEVC is based on the same inter and intra-picture prediction with a 2-D transform coding approach as all video compression standards used since H.261. The Figure 2.10 illustrates the block diagram used to create a bit stream in HEVC, which is similar to the H.264 with the exception that an optional extra filtering process named Sample Adaptive Offset (SAO) was added after the in-loop deblocking filter. This filter helps reconstruct the original signal amplitudes by recurring to a look-up table. Another minor difference from the H.264 is the entropy coding block which provide a new version of CABAC, optimized for parallel processing.

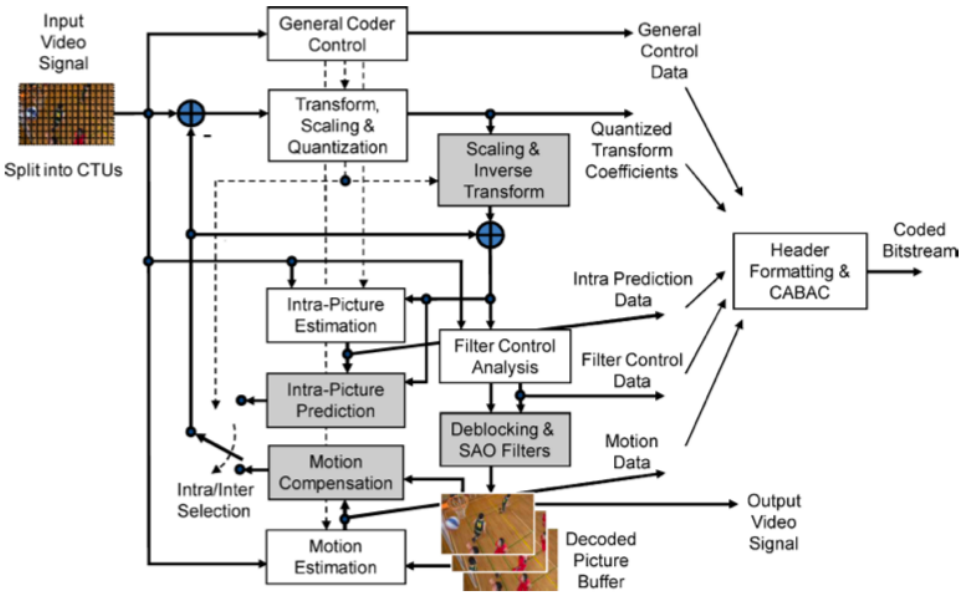


Figure 2.10: The HEVC video encoding model. [6]

The High Efficiency Video Coding introduces a new coding structure called Coding Tree Unit (CTU), discarding the macroblock structure used in H.264/AVC. Each block size is selected by the encoder and can vary from 16x16, 32x32 or 64x64, which differs from the traditional 16x16 macroblock. CTUs consist of an assemble of luma and chroma Coding Tree Block (CTB) and an associated Syntax Element block. In a general 4:2:0 colour sampling case, one CTU is composed by one luma CTB and two chroma CTBs. Luma CTB have the same size LxL of the CTU, while the chroma CTBs have L/2 x L/2.

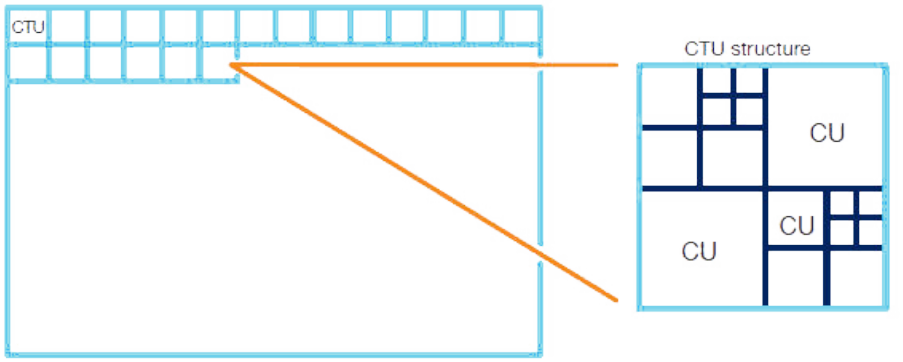


Figure 2.11: Coding Tree Unit Structure.

Each CTB can be partitioned into smaller blocks using a tree structure and quadtree-like signalling syntax[23], called Coding Units (CUs), as illustrated in the Figure 2.11, which in turn consists of a luma and chroma Coding Blocks (CBs). Since the quadtree syntax has the associated CTU as the root, and that tree specifies the position and size of each individual CU it contains, meaning the maximum supported size a CU can have is the CTU's L x L size and the minimum size is 8x8. CU can also be partitioned further into Prediction Units (PUs) and Transform Units (TUs), each containing their own trees, illustrated by the Figure 2.12.

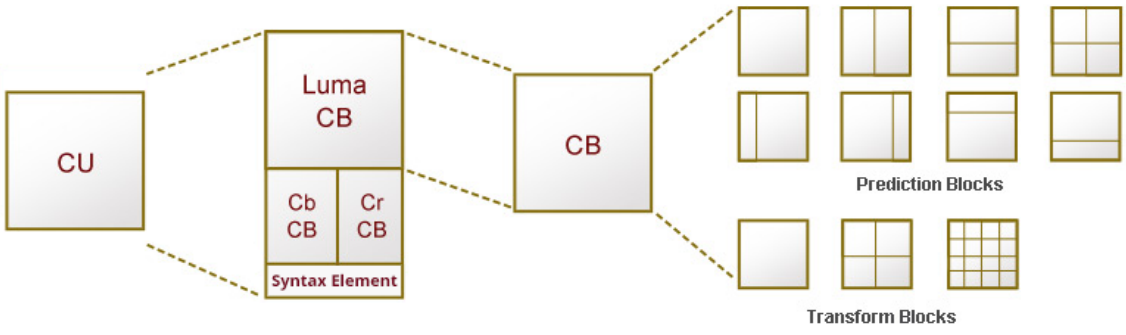


Figure 2.12: Coding Tree Structure and division.

[23] Quadtree refers to the arrangement of four tree leaf within one tree node.

The PUs has its root at the CU level, which is where it is decided whether to code the area using a Intra-picture (spatial) or Inter-picture (temporal) prediction. Each PU can be further partitioned in size into a luma and chroma Prediction Blocks (PBs) and an associated prediction syntax. Depending on the prediction-type chosen, PB size can vary from 64x64 into a smaller block size of 4x4. When the prediction is signalled as Intra-picture, a CB can be split into one PB (M x M) or into four equal-sized PB quadrants (M/2 x M/2), each having their own intra prediction mode. Intra prediction supports 33 directional modes, a planar mode and a DC mode. Due to this increased number of modes, when coding luma PBs, HEVC considers the three most probable modes (MPMs) and when coding chroma HEVC can select five modes, including one to match the same mode as the luma prediction. The selected mode can also be chosen based on previous decoded neighbouring PBs which can follow the TU tree exactly, although the actual prediction process operates separately. HEVC then applies a smoothing filter adapted to the directionality chosen, the discontinuity detected and the block size. To ensure the boundaries aren't discontinued between PBs, a filter is also applied. When the prediction is signalled as inter-picture, HEVC can split the CB into one, two or four PBs, thus having a total of eight different sizes to split, as seen on the figure 2.12. This includes the M x M and M/2 x M/2 size used in intra, with the addition of M/2 x M or M x M/2 sizes or even M/4 x M (on the left hand size) and M/4 x M (on the right hand size), and M x M/4 for up and down side. Each inter-picture predicted PB is assigned one or two motion vector (unidirectional or bidirectional), and reference picture indices. HEVC includes a Advanced Motion Vector Prediction (AMVP) mode that can gather several most probable candidates based on neighbour PBs, where the process is preformed once per motion vector in a unidirectional PU or twice for a bidirectional PU. Another mode that can be used is the merge mode that can derive the information it needs from spatial or temporal neighbouring blocks. Each mode builds a list of candidate motion vectors, and then selects one of them using an index coded on the bitstream. For the motion compensation, HEVC specifies motion vectors in 1/4 sample precision, but uses an 8-tap filter for luma PB and a 4-tap 1/8 precision filter for chroma PB. HEVC also supports weighted prediction for both unidirectional and bidirectional PUs. However those weights are always explicitly transmitted in the slice header, as opposed to the AVC.

The TUs is where the prediction residual is coded using block transforms, storing both transformation and quantization coefficients. Similar to PU, the TU has a tree structure containing the CU level as the root. Each TU is further split into luma and chroma Transform Blocks (TBs). The luma and chroma CB may be identical to the luma and chroma TB or it may be partitioned into smaller TBs. TB are composed by squares of 4x4, 8x8, 16x16 and 32x32 sizes, which is then applied a discrete cosine transform (DCT). Since there is no 64x64 transform, a 64x64 CU must be split at least once into four 32x32 TUs. Regarding the specific case of a 4x4 luma TB, since there can't be a 2x2 chroma TB, the 8x8 CU will be split into four 4x4 luma TB and two 4x4 chroma TB. In this particular case, a discrete sine transform (DST) is applied instead of DCT, as it proved to be more effective due to the statistical properties the TB inherits. The HEVC then uses URQ scheme for quantization, controlled by a quantization parameter (QP), which is defined from a 0 to 51 range, having the step increased by 6. TUs' coefficients are coded into the bitstream

in different manner compared to the H.264/AVC. The coefficients are grouped into a 4x4 fashion and then scanned diagonally (down and left) using their last xy position until it reaches the DC coefficient.

Deblocking filter (DBF) in HEVC operate within the inter-picture prediction loop and is performed on the edges of a 8x8 grid only, for both luma and chroma samples, reducing the blocking artifacts created in the quantization process. The vertical edges of the picture are de-blocked first, followed by the all horizontal edges. The filter acts similar to the H.264/AVC but the boundary strengths only support three levels: 2, 1 and 0. Because of the 8-pixel separation between edges, they do not depend on each other enabling a highly parallelized implementation. After this filter is done, an optional filtering process called Sample Adaptive Offset (SAO) can be applied. This process is done per each CTU and operates once per each pixel, within the CTU, by adding an offset value to it based on values in the look-up tables transmitted by the encoder. This allows a refinement of the reconstructed signal and thus closely match the source image, allowing a better predicted image to be used as a reference picture in the inter prediction loop.

Entropy coding done in HEVC is very similar to the H.264/AVC with the exception of only using the Context Adaptive Binary Arithmetic Coding (CABAC) entropy coding. The core algorithm of CABAC is the same used in H.264/AVC but with a few minor improvements, such as less context state variables are used and the initialization process used was simplified. The bypass-encoded bins are carefully grouped together as much as possible, making it possible to decode more than one bypass bin at a time, thus improving the speed by using the parallel-processing architecture's tools.

HEVC is designed to take advance of paralleling-processing by enabling a multi-threaded decoder to decode a single picture with threads using two tools called Tiles and Wavefront. Tiles tools enable the picture to be divided into rectangular grids of CTUs (tiles) each containing an integer number of independently decodable CTUs that are within the tiles. This allows for motion vector prediction and intra-picture prediction not to be performed across tile boundaries, acting as if each tile is a separate picture. The Wavefront Parallel Processing (WPP) tool enables each CTU row across a slice to be decoded using their own thread. After a particular thread has started decoding the second CTU of that particular row, the context models of the entropy coder are transferred to the row below and the next thread starts decoding the next row, and so on, ensuring that there aren't cross-thread prediction dependency issues. WPP is not meant to be used in combination with the tiles tool and it has been proven that provides a better compression compared to tiles.

2.7.3.2 Comparing HEVC to AVC

Since macroblocks are a basic unit used in processing the motion estimation, which is the most demanding component in the processing part, the use of a quadtree fixes this issue due to the easy access and decision of information. It has been reported that HEVC can achieve 50 % more compression in a binary tree searching algorithm than the H.264 counterpart, which means that using the same bandwidth, HEVC can insert more information, and thus improving the overall quality

of the video[6, 18]. Additional work is also planned to extend the standard to support several additional application scenarios, including extended-range uses with enhanced precision and colour format support (currently just supports 4:2:0), scalable video coding, and 3-D/stereo/multi-view video coding.

When used with lower resolution signals, such as those of previous generation systems, HEVC is also able to deliver significant improvements in the compression/quality ratio when compared to H.264. Some companies, such as NTT[24], are doing efforts to apply this compression algorithm into its video services, eliminating communication delays and drop-outs due to the lack of processing speed, therefore enabling smooth and real-time video.

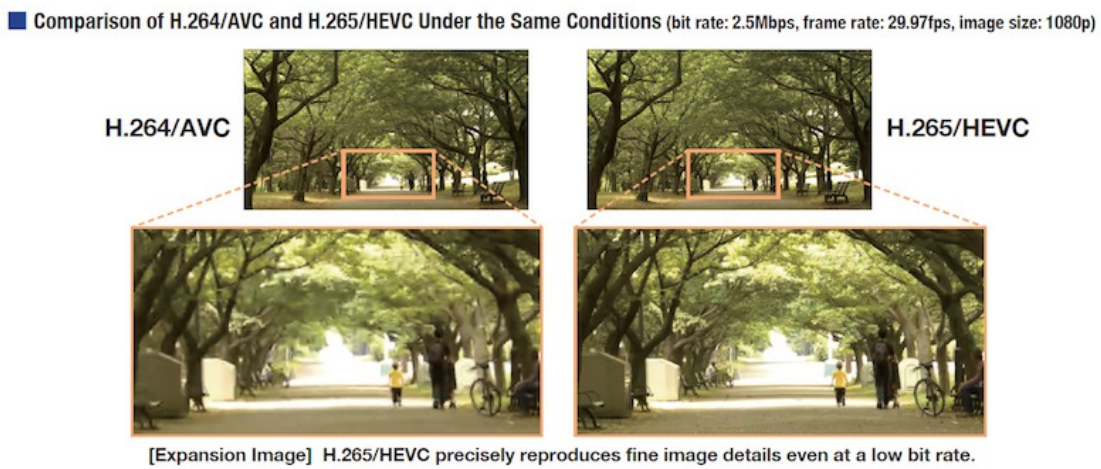


Figure 2.13: Example of HEVC to H.264 using the same characteristics [7].

2.7.4 Part 3: 3-Dimensional Audio

This part of the container is still being researched and developed, but it is already being prepared for the Super Hi-Vision audio specification. 3-D Audio is referred as a standard for producing a "3-Dimensional Audio", in which there are many loudspeakers presented in the environment. Currently there are some problems that are being investigated, such as the need to automatically adapt audio program material to the target number of loudspeakers in a given consumer's environment and to consider that Super Hi-Vision places the consumer relatively close to the screen, having to enhance the sound source to provide the sense of realism.

[24]NTT refers to Nippon Telegraph and Telephone Corporation, a Japanese telecommunications company.

Chapter 3

Proposed algorithm to improve Super Hi-Vision encoded video

Although this dissertation focus more on a systematic analysis of the Super Hi-Vision, there are some intrinsic problems with the encoding of a Super Hi-Vision video that can be explored academically. This chapter will be dedicated to describe the problems and the methods used to tackle those problems.

3.1 Objectives and Functionalities

The problem with Super Hi-Vision system is the intrinsic high bit rate associated with its uncompressed signal. As stated in 2.7, a Super Hi-Vision signal can reach bit rates in the range of 144 Gbps if uncompressed. With the joint video project of the ITU-T VCEG and the ISO/IEC MPEG working on HEVC, NHK in collaboration with Mitsubishi Electric has developed their own real time encoder for Super Hi-Vision (Working draft 4 compliant)[4], compressing a 30 Gbps[1] signal to a 85 Mbps, achieving a compression ratio of 350:1 [14]. The attained result comes by dividing the screen into 17 horizontal stripes, with a resolution of 7680×256, being each stripe then encoded in parallel by different processors. Data is exchange between them to enable taking profit of redundancies and thus achieve better compression efficiency. Parallelising the encoding process, enables to achieve lower processing times without requiring extremely sophisticated processors.

This impressive compression ratio, delivering an 85 Mbps real-time encoded bitstream, is sufficient to allow the transmission of a super Hi-Vision signal using one satellite transponder. However, it is not yet sufficient to deliver it to home users, due to limitations both in the home equipment as well as in the network connections. Accordingly, there is still the need to study approaches that might be able to improve the efficiency of the recently developed compression algorithm, or to launch a new initiative to formulate a new compression paradigm that could deliver the desired levels of compression. Given that the emergent HEVC algorithm was the result of efforts jointly made by the most prominent researchers in the area around the world, the latter

[1] Super Hi-Vision running at 60 Hz with 10 bit colour depth and a 4:2:0 chroma sub sampling, as shown in Table 2.8

would likely prove to be a useless effort. Within this context, the main goal established for this dissertation was to investigate new approaches for using more efficiently the already developed tools, or to devise pre-processing operations that could help the algorithm to take wiser decisions and thus achieve better results in terms of quality-compression ratio. Although it had been initially decided that the work would also include conducting network simulations to evaluate the effects of network conditions on the quality of the encoded video, this objective was dropped during the preparation of this dissertation. This was due to the rather high complexity already involved in the pursuit of the main objective. This decision has allowed to dedicate more attention to the main goal, which eventually led to the obtention of better results. In fact, the results regarding the use of pre-processing techniques to increase the efficiency of the HEVC coding algorithm were so promising that it was decided to prepare and submit a scientific article to an international conference or journal.

The developed algorithm is based on the NHK's approach to partition the Super Hi-Vision video into several different slices but instead of always applying an horizontal slicing, it also applies a vertical slicing or a non-linear slicing method. By doing so, it can reduce the inter-dependency spatial correlation of each slice and better fit the motion of the video, originating a far better quality and compressed encoded video compared. The idea was to prove that always applying an horizontal slicing as NHK demonstrated, may not be the best approach, as some videos might have vertical or horizontal displacements, thus creating more inter-dependency between each slice strip as it would be ignoring the spatial information of the frame. Another method was developed to verify if the efficiency, or quality-compression relationship, of the encoder was based on the stripe size. Those dimensions were varied by changing the step size from 256 to 512.

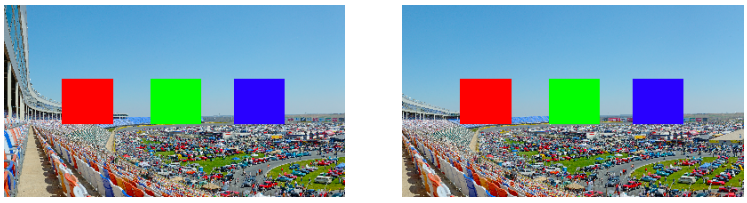
As such, the algorithm can be divided into two types of slicing: **linear** and **non-linear** slicing. A linear slicing will always have the same fixed size regardless of its content, while non-linear slicing takes into consideration the spatial activity of the image to partition it into variable size regions. It thus relies on a pre-processing operation that obtains indications concerning the type of division to apply to each picture on the test data. The next sections explain those methods in more detail.

To test the different type of slicing, four videos were created, each one with its different type of movement. The Super Hi-Vision videos were created by applying in the background a static 70+ Mpx photo and then moving the camera to simulate movement. The type of movements chosen were horizontal, vertical, diagonal and zoom-in. To simulate several environments as well, each video contains a different background photo, allowing for a more ample testing regarding the slice versus the spatial information it contains. Each video is composed by 7680×4320 spatial resolution, $60Hz$ frame rate, no chroma sub-sampling and 8 bit colour per channel, which is non compliant with the *ITU-R Recommendation BT. 2020* but it was the maximum colour option in the video editor, saving them in a raw AVI extension. To limit the time processing of the algorithm, the duration of the video was chosen to be 0.5 seconds, which implies a 30 frame video, having a file size of 2.98 GB[2] plus the metadata associated with each file. The sound channel in each video

[2] $7680 \times 4320 \times 60 \times 8 \times 3 \times 0.5$ = 23.887,872,000 Gb or 2.985,984,000 GB

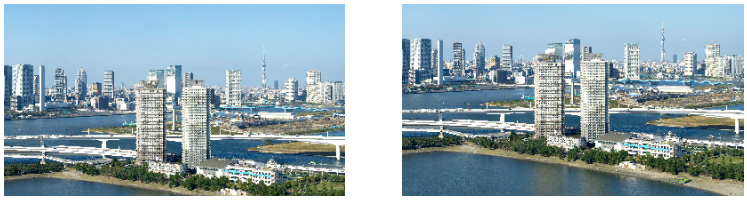
is omitted as it is not the focus of this dissertation.

Each video will be referred, henceforward, by their respective movement, in order to clearly distinguish its content. To illustrate the content of each video, the first and last (30th) frame were capture and displayed in the following scaled figures.



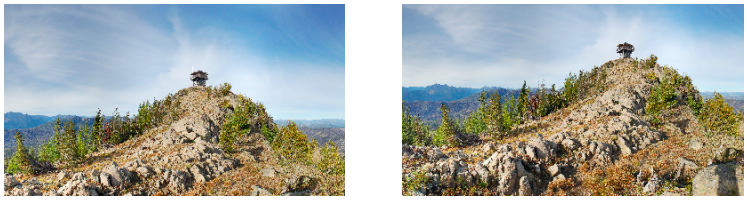
(a) First Frame (b) Last frame

Figure 3.1: Original horizontal-moving video.



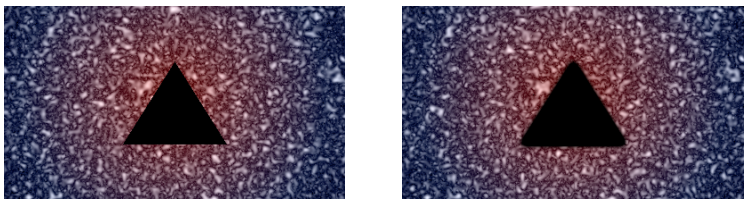
(a) First Frame (b) Last frame

Figure 3.2: Original vertical-moving video.



(a) First Frame (b) Last frame

Figure 3.3: Original diagonal-moving video.



(a) First Frame (b) Last frame

Figure 3.4: Original zoom-in video.

3.2 Linear Slicing

The linear slicing method was develop to observe the inter-relationship between the direction of the slices versus the displacement of the video. This linear slicing method, as previously explained, is based on the method used by NHK, which involves dividing the video into 17 horizontal 7680×256 stripes and then encoding them individually. Since the displacement can be vertical, another type of slicing was tested, which consists in dividing the video into 30 vertical 256×4320 stripes. Those proposed techniques used in the linear slicing method, illustrated in Figure 3.5, were tested on several videos with different types of displacement.

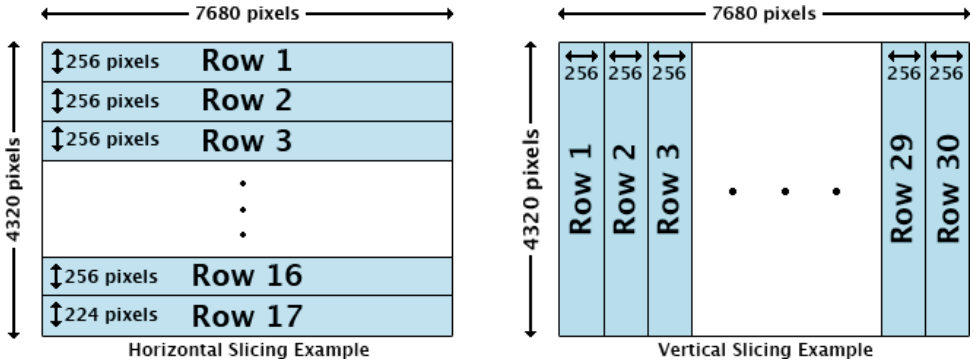


Figure 3.5: Horizontal and Vertical slicing methods with a 256 step.

The next step was to verify if increasing the step size would have any impact in the quality, compression rate and processing time of the encoded video, since there was a big disparity of values between horizontal and vertical slicing facing their respective moving videos. It was expected that an increase in the stripes dimension, represented in Figure 3.6, could deliver better quality and higher compression ratio, since it would allow to better take profit of spatial redundancy.

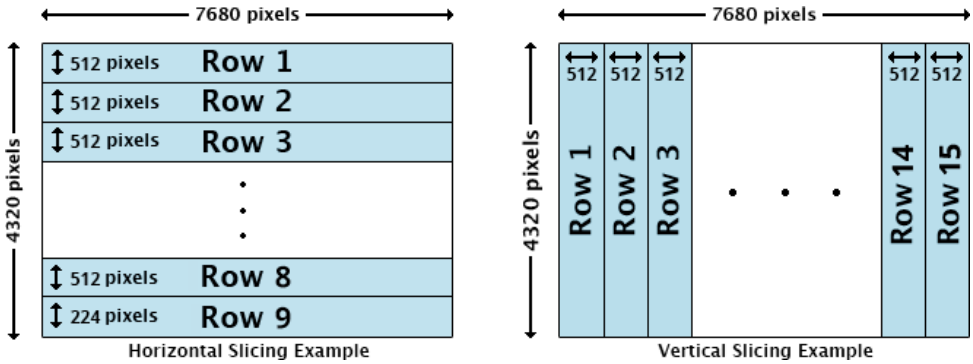


Figure 3.6: Horizontal and Vertical slicing methods with a 512 step.

Whilst the disparity on the obtained efficiency seemed to decrease when applying different step sizes leading to larger stripes dimensions, the quality gains were marginal, comparing to the

original slicing situation with 256 step. Accordingly, it was decided that such minimal gains did not justify continuing the experiments by varying the step size.

Our assumption is that the use of horizontal stripes versus vertical stripes should be dependent on the type of movement found in the video. Accordingly, having proved that the coding algorithm worked correctly when processing in parallel either horizontal or vertical stripes, the next step would be to devise a method that could wisely decide between the two. That method should thus be able to detect the direction of the movement in the video. Given the high volume of data associated to each Super Hi-Vision video, the analysis of every pixel of every frame in this pre-processing step, would compromise the efficiency of the complete encoding operation. For that reason, we have devised a fast algorithm that uses only a small part of the video data to detect the movement. The algorithm creates five windows of 8 rows by 8 columns, one on each corner of the target frame and one in the middle. The search is then done using a custom *Fast Search Algorithm* (FSA) that uses the *Mean Absolute Difference* (MAD) inside a 24-by-24 window on the anchor frame to find the position of the 8-by-8 target window with the least errors. This generates a 3-by-3 matrix containing the MAD values representing the errors on all nine directions of that window.

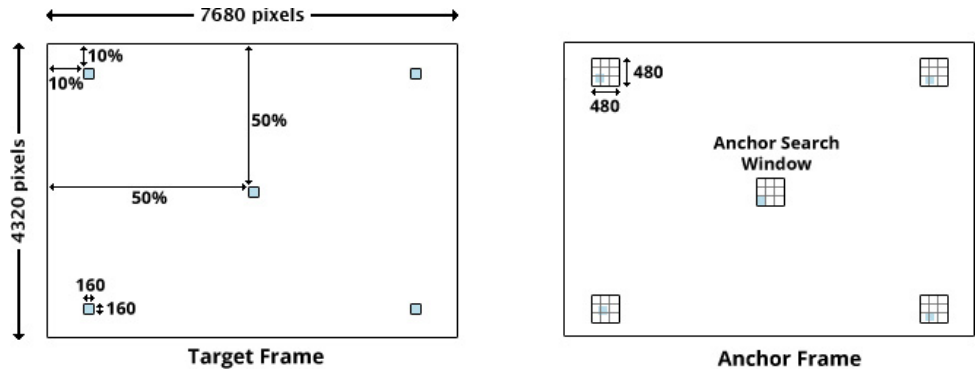


Figure 3.7: Example of the FSA searching the blue squares inside the anchor window.

This method is then performed on all five sub-sampled windows resulting in a 3-by-3-by-5 matrix containing the MAD values, illustrated on the figure 3.7. The orientation of the video is chosen by adding the values on each position inside of all the five matrix and then picking the cell with the lowest value. The location of the value represents the general orientation of the video, as exemplified on the figure 3.8.

A custom Fast Search Algorithm was chosen instead of an Exhaustive Search Algorithm, because during the process of developing the algorithm, the latter proved to take substantially more time to process, just to generate a more precise orientation matrix. The result given by using the custom Fast Search Algorithm proved to be a rather accurate approximation of the exact orientation matrix, using just a 1/5 of the time.

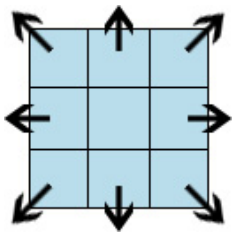


Figure 3.8: The final matrix showing which position corresponds to which direction the video is having.

3.3 Non-linear Slicing

The non-linear slicing method was developed to provide an alternative and possibly more efficient slicing method to the linear one, taking also into consideration the information about the spatial activity of the image. This method consists into reading the luma components of several frames from the video and applying a Sobel Filter to get the frame's horizontal and vertical kernels. The spatial information magnitude of each frame is given according to this formula:

$$SI_r = \sqrt{s_h^2 + s_v^2} \tag{3.1}$$

where s_h and s_v represents the horizontal and vertical Sobel kernels and r the selected frame. Those values represent the magnitude of each pixel characterizing the spatial information. By reading that information on several frames, not only are we visualizing the image complexity spatially but also in a temporal fashion, guaranteeing that a possible movement of the complexity across the frames is recorded. The next step is to get the spatial information mean to use as a base line to distinguish what represents an area it high or low detail. This mean can be obtained mathematically as:

$$SI_{mean} = \frac{1}{P}\sum SI_r \tag{3.2}$$

where P stands for the number of pixels that compose the frame [19]. After obtaining the image complexity in both space and time, and using the spatial information mean as a baseline, the algorithm then divides the resulting matrix into a 4-by-4 matrix. Each cell inside the matrix represents a 1920×1080 sub-sample, which is then calculated the average spatial information inside that sample. One demonstration of a resulting matrix can be given as:

$$\begin{bmatrix} 11.245 & 15.221 & 36.865 & 41.333 \\ 7.955 & 10.071 & 20.453 & 23.112 \\ 180.210 & 123.231 & 101.563 & 190.239 \\ 230.317 & 108.773 & 240.622 & 163.440 \end{bmatrix} \tag{3.3}$$

To create stripes, thus setting boundaries between image regions, the algorithm aggregates values if they meet the following conditions: 1) their mean value is lower than the overall mean value and they differ less than a determined threshold; 2) their value is larger than the average. To complement the example 3.3 above, if the spatial information mean was given as *99.238* then the resulting slicing scheme would be represented as:

$$\begin{bmatrix} 1 & 1 & 3 & 3 \\ 1 & 1 & 4 & 4 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix} \tag{3.4}$$

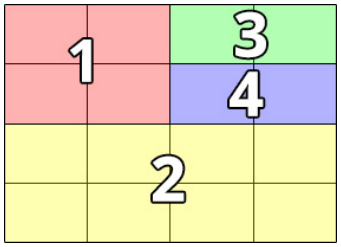


Figure 3.9: The slicing representation of the matrix generated.

The numbers inside the resulting matrix represent the slice number and how they will be partitioned to create a non-linear sliced video, represented in the figure 3.9. Each slice is then encoded individually and reconstructed, exactly as in the linear slicing method.

Chapter 4

Implementation

This chapter describes in detail the tools, methods and the options that were taken throughout the process, in order to develop the video slicing techniques described in chapter 3.

4.1 Program implementation

Before starting the implementation of the proposed algorithm, two programming languages were considered in order to test functionalities and performance times: *OpenCV*[1] and *MATLAB*. Since the scope of this dissertation is based on a large amount of matrix based operations, *OpenCV* revealed to be quite slower when loading the 2.98 GB Super Hi-Vision video and doing simple mathematical operations, as opposed to *MATLAB*. The implementation chosen was the high-level language *MATLAB* using *vectorization*[2] coding techniques, whenever possible, to improve performance. All the processing times displayed during the development, execution and results have as base the computational power of the machine it is running. To have an accurate measurement, every time the program was executed, the same conditions were met. This implied that all non essential program was closed having just *MATLAB* opened, always running with an environment temperature of around 19 Celsius. It is to note that, although aiming for fasting processes and code optimization that reduces the overall processing time, all the tests had a duration of at least one hour. However, having in mind that without those optimizations and fast processes, the code had durations of a minimum two hours.

Regarding the structure, the program has one main file that runs the algorithm called *main.m*, which in turn executes several functions to detect the video direction and slice it using a linear or non-linear approach. The main function has one mandatory argument, the file name, and other two optional argument that allows bypassing the detection of movement and slice size, which in turn are useful to debug or force non optimal situations. To accurately measure the processing time, at the end of each step time the execution time, in seconds, is displayed.

[1] OpenCV is the shortening for Open Source Computer Vision and is a library of programming functions for real time computer vision.

[2] Vectorization is an inline process of implementing loop-based algorithms and scalar-oriented values in MATLAB.

The heart of the program is the **x265**[3], an open-source project and free application library for encoding video streams into H.265/High Efficiency Video Coding (HEVC) format which is under the *GNU GPL 2 license*[4], allowing each slice to be encoded using the HEVC standards. This external program is made from C++ programming language and is executed through a system command in *MATLAB*.

Another external program is used to play the raw YUV slices or the encoded x265 files, called **ffplay**[5]. This is simple media player without any graphical user interface that can be executed through a bash command, and is based on the FFmpeg libraries and the SDL library. This tool was mainly used to debug raw YUV slices using the following command in a bash shell:

```
$ ffplay file.yuv -video_size 7680x256 -x 7680
```

where the *-video_size WxH* argument is only specified on YUV files since they do not contain any metadata and the *-x W* argument forces the window to have the full Super Hi-Vision width without resizing to the screen's maximum width. To play the x265 encoded file the command would be the same except there isn't any need to specify the video size as it is already included in the metadata. This tool was fundamental in observing the subjective quality of the each sliced video or the full reconstruction.

All of the source code used in this dissertation is well documented and it is located on my homepage[6], as well information on how to compile the x265 and ffplay on different operating systems.

4.2 Accessing the video

MATLAB has a class that reads video containers and loads them to memory called *VideoReader*. This class provides all the metadata information, such as the length of the video in seconds, the number of frames it contains and the video size, amongst other useful information. The frame accessing is done through the *read* method, which returns a height-by-width-by-3 sized matrix containing the numerical values corresponding to the red, green and blue component.

4.3 imageSlice Object

To aid with the slicing process and to make the algorithm more agile, an object-oriented class named *imageSlice* was created. The class has as attributes the number of rows and columns, colour depth, orientation (0 for horizontal or 1 for vertical) and finally the row-by-columns-by-depth original matrix sliced from the image. This class thus lays the foundations as each object consists of one slice from a specific frame and it is stored in memory, which is perfect to access and iterate through an array of imageSlices while performing mathematical operations.

[3] http://x265.org

[4] GNU General Public License is a license widely used in free software that states that everyone is permitted to copy, modify and distribute verbatim copies of the content, as long as it is free.

[5] https://www.ffmpeg.org/

[6] http://paginas.fe.up.pt/~ee05068/university/documentation

4.4 Directionality detection

One of the first steps after executing the main program is doing a directionality detection, as specified in the previous chapter. This functionality allows the program to decide of how it should slice, or not, the current video. Although not implemented in this dissertation, it is prepared to repeat the process every 30 frames, allowing a precise slicing in longer and more real videos. As stated in the figure 3.7, the algorithm chooses the first frame as the anchor and the third frame of that set, as the target frame. The third frame was chosen because the second frame in some cases, did not display movement as the video had a slow displacement and the fourth frame originated results which the minimum error value was quite high. Those values could improve greatly if used a *Exhaustive Search Algorithm* instead the custom *Fast Search Algorithm* but the processing time would greatly increase. Also having in mind the processing speed, since this algorithm works with the luma component of the frame and the *VideoReader* class reads frames in RGB format, converting a 7680×4320 RGB image to YUV took roughly 55 seconds each but decimating it by 20 times took just 1.5 seconds. The decimating process removes samples of the signal, which is equivalent to use a lower sampling rate that cannot satisfy the Nyquist limit. This means the image loses quality and details due to that aliasing introduced, so a Gaussian Smoothing[7] filter was applied to remove the image sharpness and restore it to an approximation before decimating, thus reducing the error margin from the searching algorithm, illustrated on the figure 4.1.

(a) Decimated anchor frame without the filter from the horizontal video.

(b) Decimated anchor frame with the filter from the horizontal video.

Figure 4.1: Comparing non-filtered vs filtered decimated frame.

Currently this functionality returns if the video should be encoded without slicing or if it should be sliced vertically, horizontally or, in the case the minimum value is bigger than a threshold, non-linear. This is decided by reading the matrix that results from the custom *Fast Search Algorithm* as displayed in Figure 3.8. The threshold was decided by reading the values of all four types of videos and observing that a minimum error bigger than 20 with horizontal or vertical slicing generated videos with a low quality and compression rate. Since there is not a diagonal slicing method, when any diagonal directionality is detected, the algorithm opts to use a non-linear approach.

[7] This filter acts as an anti-aliasing filter, smoothing it based on statistical properties of the image.

4.5 Linear slicing algorithm implementation

Linear slicing can be done using an horizontal or vertical approach, and although the techniques result in a different orientation slice, the process more or less identical. After identifying the direction of the video, a function is evoked that slices the video according to the predetermined orientation. The process begins by reading frame by frame and in each iteration generating an array of imageSlice objects, containing all the slices with the specific step size of that frame. The end result is an array that contains slices-by-frames, which is a perfect format to access both a specific slice across all frames or all the slices in one specific frame.

4.6 Non-linear slicing algorithm implementation

The process to generate non-linear slices diverges a bit from the linear counterpart. The algorithm has two distinct phases: reading the spatial activity and slicing accordingly. The first phase, as described in the section 3.3, starts by reading the first, middle and last frame of the video. Then it decimates each by 20 and converting from RGB colour space to YUV, which in turn a Sobel Filter is applied to the luma component of the selected frames to get the horizontal and vertical kernels. This method was used because the end result by reading all the frames and keeping the original size would result, again, in a very similar output but processing for an average of 47 minutes comparing with the 3.8 second processing time of the selected method. After obtaining the 4-by-4 spatial activity matrix, exemplified on the matrix 3.3, the steps used to agglomerate values together to form a slice are the following, illustrated in Figure 4.2.

1. Creates a loop and select the first top left non-zero value as anchor value.
2. In case the anchor value is **below** the spatial information mean:
   (a) Iterates rows first until it reaches a value that is below the anchor value minus an offset or above the anchor value plus an offset.
   (b) Iterates columns until it reaches a value that is below the anchor value minus an offset or above the anchor value plus an offset.
   (c) After having the bottom and right limit, it replaces the values inside by a negative value representing the slice number.
   (d) It jumps back to the beginning to search for a new non-negative anchor value.
3. In case the anchor value is **above** the spatial information mean:
   (a) Iterates through the rows first until it reaches a value below the spatial information mean minus an offset.
   (b) Iterates the columns until it reaches a value a value below the spatial information mean minus an offset.
   (c) Replaces the values inside the area with the negative value of the slice number.
   (d) Jumps back to the beginning searching for a new non-negative anchor value.

4. Exist the loop if there is not a positive value inside the matrix and inverts the values, resulting in a matrix as displayed in 3.4.

The reason why the rows are first searched is because the *MATLAB* function **find** searches rows first and then columns, resulting in the first top left value that satisfies the condition. This has the effect that non-linear slices tend to be horizontal rather than vertical if similar values are near each other but are broken in the middle by an higher or lower value. Regarding the offsets, it was chosen an offset of 30% for anchor values below the spatial information mean and an different offset of 20% to be used when the anchor value is above the mean. Those offset were chosen to be different because the spatial information mean is usually a high value and 30% could result in an area with low detail being agglomerated with an higher detailed slice. The same principle applies to the offset value below the mean, because those values are usually very small and 30% is enough to verify if they are similar. All those values were obtained through trial and error, with a very limited data set, as well as the fact that the algorithm just agglomerates all values above the spatial information mean. This is because if they were split and formed different slices, there would be a high chance of breaking that area's inter-dependency, resulting in a video with less quality and compression rate, since the spatial information mean represents the borderline where all details are located.

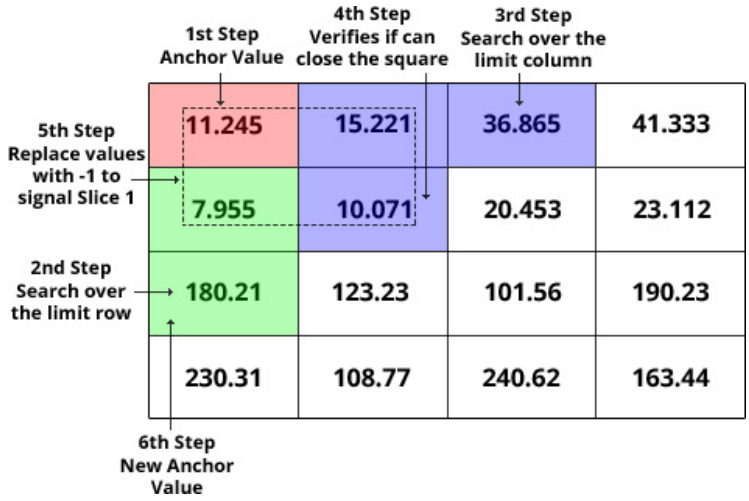


Figure 4.2: Example of one iteration of the searching algorithm.

After completing that first step, a file is saved containing a index matrix with the slicing boundaries so it can be used to reconstruct it posteriorly. Another possibility would be to numerate the slices by using a quadtree like scheme that, when reconstructing, the algorithm could mathematically find the right slice and bind together. However creating an external file containing that scheme information proved to be just as efficient as the quadtree structure.

The second step is to slice the frames in the correct way using the scheme matrix that was generated in the first step. The algorithm iterates through all the frames and, within each frame, reads the boundaries of each slice and cuts according to it. The result is a frame-by-slice array

exactly like the linear slicing method. Since all slices are *imageSlice* type object, it means that process of encoding them will not vary from slicing method.

4.7 Encoding Process

The encoding process is the most important step in the algorithm as it encode the slices individually and generates .265 files compliant with the HEVC standards. As explained in the section 4.1, an external C++ program is used to encode the slices, called *x265*. This open source program is being develop by the same volunteers who develop the *x264* project, the popular open source H.264/MPEG-4 AVC encoder, thus being a reliable program to get accurate results. However, a few tests revealed that encoding raw RGB coloured AVI files resulted in failed encoded files. This meant that every file had to be converted into the YUV colour space with 4:2:0 sub-sampling before encoding. Since the *x265* is running externally from *MATLAB*, it also meant the slices had to be physically saved in the hard drive, so that *MATLAB* could execute a system call, passing the file name as argument. The arguments used to encode each slice are described in following table 4.1.

Command	**Parameter**	**Description from the x265 Evaluator's Guide**
–preset	slower	Sets parameters to preselected values, trading off compression efficiency against encoding speed. These parameters are applied before all other input parameters, thus having the possibility of overriding them. **Values:** ultrafast, veryfast, faster, fast, medium **(Default)**, slow, slower, veryslow, placebo.
–input-res	<size of the slice>	Source picture size [w x h], auto-detected if Y4M;
–colorprim	bt2020	Set what color primitives for converting to RGB. **Values:** bt709, bt470m, bt470bg, smpte170m smpte240m, film, bt2020, undef **(Default)**.
–fps	60	Source frame rate; auto-detected if Y4M;
–psnr		Calculate and report Peak Signal to Noise Ratio.
–ssim		Calculate and report Structural Similarity values
–no-progress		Disable per-frame encoding progress reporting.
–log	1	Hides all INFO tag information regarding the file read, displaying only the WARNING and ERROR messages. **Values:** 0:ERROR; 1:WARNING; 2:INFO**(Default)**; 3:DEBUG; 4:FULL -1:NONE
–csv	<name of the file>	Writes encoding results to a comma separated value log file. Creates the file if it doesn't already exist, else adds one line per run.
–recon	<name of the file>	Re-constructed image YUV or Y4M output file name of the encoded file.
-o	<name of the file>	Bitstream output file name. There's no defined file extension so the extension chosen was 265.

Table 4.1: The commands used in x265 and their respective descriptions.

One example of executing the x265 through a system call can be :

```
$ x265 slice1.yuv --preset slower --input-res 7680x256 --colorprim bt2020
--fps 60 --psnr --ssim --no-progress --log 1 --csv output.csv
-o slice1.265 --recon reconstructed.yuv
```

4.8 Reconstruction of the video

The reconstruction algorithm was proven to be the most concept design challenging part of the algorithm. The first approach was done by trying to concatenate all the different encoded slices into one single file. That proved to be very hard because it would require to edit the x265 program to have access to an inter-slice metadata, and by forking that project, it would require to know everything just to make the algorithm work. Another approach was to read the byte stream format of the file and add the information directly, but that gave a lot of problems because even detecting the RTP[8] header with the respective payload and detecting each NAL Unit header and data, some slices had type B-frames while others only had type P-frames. Even if those problems were conquered, the fact that adding information to the I-frame NAL unit, since they are coded using CABAC, it would imply the entropy would change, so it would require to decode the CABAC first, add information and code again. The final approach proved to be the simplest in concept and implementation. Instead of focusing on the encoded files, since the scope of this dissertation is to verify if slicing the video helps enhance the quality, compression rates and coding speed, the focus was shifted to the output YUV file from the encoded slice. This YUV file has the exact same content and quality as it is the result from decoding the encoded file. This approach is much easier to manipulate those files because each frame is composed of one raw luma frame, one frame of the U chroma component and V chroma component.

Having the approach decided, the implementation of the algorithm consisted on three distinct methods: one for each direction in the linear slicing and one for the non-linear slicing. To compare the subjective quality of each reconstructed file, the first frame was saved into a JPG image so it could be properly studied and compared.

4.8.1 Horizontal slicing reconstruction

The horizontal reconstruction is the most simple and fastest of all the reconstruction methods. Since the slices are already horizontal, the script loads the current frame from all sliced files and first writes the Y component from all slices into a YUV file, followed by the U and V component. The chroma component have half the rows since the slices were converted from RGB to YUV with 4:2:0 sub-sampling. The process is repeated until it writes all the 30 frames.

[8]RTP stands for Real-time Transfer Protocol and it is a standard way to format audio and video packages to be delivered over IP networks.

4.8.2 Vertical slicing reconstruction

The vertical reconstruction is more computational heavier than the horizontal one, due to the fact that in each row, there are more than one slice. This requires the algorithm to pre-load all the slices for a specific frame and starting to write the Y component row by row, writing slice by slice per row. This produces a loop within a loop, which in turn repeats for the U and V component. This type of approach is the safest but can be computationally heavier, as the algorithm complexity is increased.

4.8.3 Non-Linear slicing reconstruction

The non-linear reconstruction is the most complex as there can be many slices per row, each one with a different size, which can make the process harder to detect if the slices in a row are beginning or finishing or both. The first step is to read the index matrix generated, as detailed in the section 4.6, that contains all the information on how the slices are positioned. Knowing that each slice has a minimum size of 1920×1080, which represents one cell of the index matrix, a 4-by-4 offset matrix is created having in consideration how the slices extend outside of minimum rows and column. Only one offset matrix is created as their size does not change over time. One example would be if the algorithm is writing the second row from figure 3.9, that would imply that it was writing the middle of slice 1 and the beginning of the slice 4.

The reconstruction it self works in a similar fashion as the vertical method. The algorithm pre-loads all the slices from the current frame and starts to iterate the index matrix's rows, verifying how many slices exist in that specific row. After obtaining that information, the algorithm iterates row-by-row until it reaches 1080, corresponding to the height of one cell inside the index matrix. It then iterates through the list of slices in that specific row, adding an offset if necessary. This process is again repeated for the U and V component. The end result is a 3 nested loop cycle for each colour component, adding to the algorithm complexity which, in turn, makes longer computational times.

Chapter 5

Results

This chapter is dedicated exclusively to the demonstration of the final results achieved from the implementation of the algorithm discussed in the previous chapter. The following results achieved from the different types of videos are obtained from the CSV output file generated by the x265 external encoding program, which can be found on my homepage[1] along with the resulting screen shots from each reconstructed image. To fully understand the differences between each method, one should focus on the **PSNR** of the luma and chroma component and the **SSIM** of the luma component.

The *PSNR* (Peak Signal-to-Noise Ratio) is a process to measure the quality of an encoded image by taking into account the artefacts created by the compression, giving the ratio between the original image and the compressed one. *PSNR* values are usually expressed on a logarithmic decibel scale which, in the case of a 8-bit per channel video, can range from $50dB$ representing a very good quality video to $25dB$ which represents a poor quality video.

The *SSIM* (Structural similarity) Y is another process that measures the similarity between the encoded image and the original image. This process was designed to provide more information about the quality of the image as traditional methods such as *PSNR*, which provides the ratio of perceived errors within the image rather than similarity with the uncompressed image. The *SSIM* result is a decimal index value ranging between -1 and 1, where 1 can only be achieved if both images are identical. Usually *SSIM* index values above 0.98 are perceived as identical by the naked eye, providing the same subjective quality analysis as the original uncompressed image.

All of this information was gathered in a table for each type of video, displaying all the encoding methods used on the slices comparing to the normal method of encoding without any slicing. One relevant issue is, as explained in 4.8 from the previous chapter, the reconstruction was done on a YUV level and not by using the encoded x265 files. The column containing the encoded file size was actually based on the total bit rate produced by each slice and then multiplied by the video's duration of 0.5 seconds and divided by 8 to convert from bit to byte. Those values represent the compressed raw-data and do not include any metadata. Another value worth mentioning is the

[1] http://paginas.fe.up.pt/~ee05068/university/documentation

FPS (Frames Per Second), which represents the average rate at which the frames were encoded, thus being directly correlated with the processing time.

Appendix A, B, C and D contains all graphics and slicing boundaries images relative to each type of video, so it would not overload this chapter.

5.1 Horizontal-moving video

Slicing Method	**Total Processing time** (s)	**Average FPS** (FPS)	**Average Global PSNR** (dB)	**Average SSIM Y**	**Total Bit rate** (kb/s)	**File size** (KB)
No Slice	3118.17	0.010000	40.336	0.975326	18794.08	1174.630
Horizontal	1875.36	0.277647	43.445	0.974030	17820.28	1113.767
Horizontal 512	2013.47	0.147778	43.063	0.972718	17785.23	1111.577
Vertical	3811.36	0.238667	36.935	0.957024	47364.59	2960.287
Vertical 512	2976.28	0.155333	38.141	0.964787	33777.99	2111.124
Non-linear	2694.22	0.116000	46.449	0.982046	17108.97	1069.310

Figure 5.1: Results for different methods used in an horizontal-moving 3.1 video.

As shown in the table 5.1, applying an horizontal slicing to a horizontal-moving video yields far better results than applying a vertical slicing method. This difference is caused due to the inter-dependency of the spatial activity within the image being severed, resulting in a video with much less PSNR and SSIM Y. Figures A.1 and A.3 clearly displays that the upper half of the image is composed of an uniform colour while the bottom part has the most detail. Severing that spatial activity's correlation in a vertical fashion causes an overall encoding quality to be far less than the horizontal counterpart. However, increasing the step size on the vertical method, we can clearly see an improvement but still not nearly as good as the horizontal slicing method's results. Increasing the step size as well on the horizontal method seems almost irrelevant, as the total processing time increases and yields very few improvements on the quality and compression rate.

The normal way to encode the file without slicing is usually the best option to encode a file as it uses a variable bit rate to encode the video, taking into mind the spatial and temporal correlations. This results in a output file with a good combination of PSNR and SSIM Y index values and a perfect compression rate. In this case, the compression is around **99.92%** comparing to the 1.495 GB original 4:2:0 sub-sampled YUV file. However, having the horizontal slicing method resulted in an increase in the video quality and compression rate, the best method was the non-linear slicing method. By concatenating 1920×1080 sized slices together, considering their spatial activity over time, the areas with the least details were agglomerated, as portrayed in the figure A.5. This

lead to a better compression rate chosen by the encoder as less bits are required to encode more information.

Regarding the reconstruction process, although most of the time the assembled video do not leave traces that it was sliced when viewed at the original size, some errors do occur. Those errors usually form when slices containing large areas with uniform colours also include small but detailed objects. The figure 5.2 illustrate one example of those types of errors, with an antenna located with the sky as background, while comparing that same area when used a normal non slicing method to a horizontal and non-linear slicing method. The area exemplified is located in the right side on the transition between slices 9 and 10, in the horizontal 256 slicing method, or between the transition between slices 3 and 5 from the non-linear method. As illustrated by the figure, the no slicing and non-linear method look quite similar, although the tip of the antenna looks sharper on the latter, while the horizontal blurs half of the antenna located on the slice 9. The bit rate used for that slice was substantially less than slice 10, as seen in the graphic in the figure A.1, thus causing those errors. Figure 5.3 shows the same picture but with a 100% contrast and with a sharpening filter. In this situation, the areas where the slices meet are clearly observable in both slicing methods. It becomes easy to perceive this way that the tip of the antenna is slightly shifted to the left on the non-slicing method, which is where the slices 3 and 5 meet.

Figure 5.2: Different slicing method exemplified in an area.

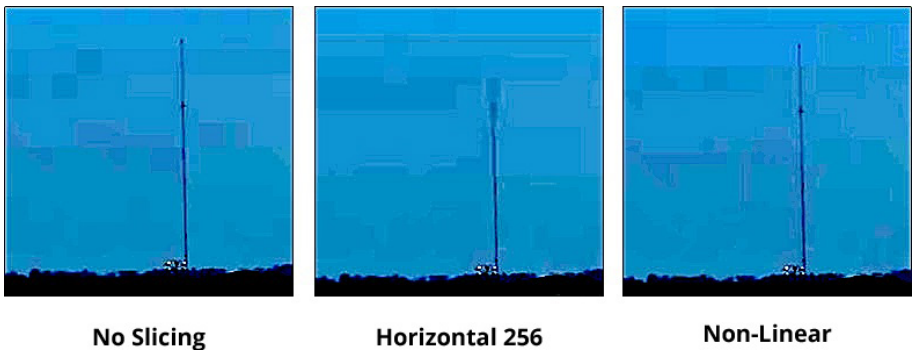


Figure 5.3: Different slicing method exemplified in an area with 100 % contrast.

5.1.1 Directionality Detection Algorithm

The directionality detection algorithm used to verify the type of displacement used in the video originated the following results:

$$\begin{bmatrix} 13.4844 & 15.9375 & 4.6406 \\ 12.2969 & 10.8750 & \mathbf{4.0625} \\ 75.0625 & 67.4688 & 60.5625 \end{bmatrix} \tag{5.1}$$

which verifies the correct orientation of the video by having its lowest value placed on the middle right cell, representing an horizontal movement to the right.

5.1.2 Spatial Information Algorithm

The algorithm used to verify the spatial activity across all the frames, regarding the horizontal-moving video, yield as result a threshold of **103.4381**, corresponding to the spatial information mean calculated using the formula presented in 3.2. Using the mean to agglomerate the slices in a 4-by-4 matrix resulted in the following:

$$\begin{bmatrix} 19.0030 & 4.9063 & 4.5548 & 4.3771 \\ 46.8049 & 12.9404 & 15.7376 & 15.0426 \\ 192.2533 & 135.2892 & 164.7322 & 173.2942 \\ 165.0702 & 244.2425 & 229.7141 & 227.0466 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 4 & 4 & 4 \\ 2 & 5 & 5 & 5 \\ 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 \end{bmatrix} \tag{5.2}$$

which represents the video being partitioned using five non-linear slices, corresponding to the figure A.5 illustrated in the Appendix A. This further emphasises that agglomerating low details areas and high detailed areas originate a far better encoded video than using a normal encoding system without any pre-sorting algorithm.

5.2 Vertical-moving video

As observed in Table 5.4, the situation proved correct comparing to the theory that applying a slicing method that follows the displacement of the video provides a better quality and compression rate video as opposed to its perpendicular method. In this particular case, applying a vertical slicing method, did not yield better results than the non slicing method. Nevertheless the method with a 512 step originated a $10KB$ more compressed file, but the trade back was the slight less quality, as displayed by the PSNR and SSIM Y values. This indicates that even if the displacement is vertical, there are always some horizontal correlation in the spatial activity which when split reduces the overall quality of the video, as seen in the figure B.4. However, besides having similar

Slicing Method	Total Processing time (s)	Average FPS (FPS)	Average Global PSNR (dB)	Average SSIM Y	Total Bit rate (kb/s)	File size (KB)
No Slice	3534.72	0.010000	37.393	0.952224	31594.75	1974.672
Horizontal	5133.57	0.106471	37.003	0.913169	48991.49	3061.968
Horizontal 512	3838.76	0.072222	37.634	0.926538	40699.75	2543.734
Vertical	3201.07	0.290000	37.153	0.943153	32694.16	2043.385
Vertical 512	3015.25	0.152667	37.188	0.943968	31439.19	1964.949
Non-linear	3938.73	0.090000	40.678	0.943912	33278.48	2079.905

Figure 5.4: Results for different methods used in an vertical-moving 3.2 video.

PSNR and SSIM Y, the vertical slicing method with 512 step was around 8 minutes faster than not slicing, which in a target file with a larger duration, would imply substantial a time-saving gains. The same table also displays that increasing the step in the opposite slicing method originates a better PSNR and SSIM Y but still not enough compared to the simple 256 vertical slicing method.

Focusing on the non slicing method, we can see a big difference in the quality and compression rate compared to the horizontal-moving video. This is due to the fact that, although similar to the horizontal-moving video where it has an upper low detailed area and a lower high detailed area, the vertical-moving video has substantially more detail spread in the middle to lower part of the video, as seem in the graph in the figure B.1 and B.2. Since the video has a vertical down displacement, the image in the background will shift up. This displays more information in the bottom and the high detailed area located in the middle will shift up, removing parts of the lower detailed areas, such as the sky. This conjugation of factors reduce the overall quality of the video and its compression rate, but it still originated **a 99.88%** compression rate compared to the 1.465GB YUV file.

However, if we focus on the non-linear slicing method, we can see that the PSNR and SSIM increased but the compression rate is still slightly higher. The result expresses the correlation of the spatial activity that also shifts in a temporal way. This is because the slices are organized in a way that compensates the displacement of the spatial activity during the 30 frames of the video, as seen in B.5.

5.2.1 Directionality Detection Algorithm

The directionality detection algorithm used to verify the type of displacement used in the vertical-moving video originated the following results:

$$\begin{bmatrix} 79.8750 & 51.9375 & 51.9844 \\ 52.4531 & 37.2344 & 53.5469 \\ 65.3750 & \mathbf{16.4844} & 63.2969 \end{bmatrix} \tag{5.3}$$

which verifies the correct orientation of the video by having its lowest value placed on the middle bottom cell that represents an vertical movement down. The high minimum value in this case, is still below the 20 threshold so it confirms the best option for a linear slicing method.

5.2.2 Spatial Information Algorithm

The algorithm used to verify the spatial activity across all the frames, regarding the vertical-moving video, yield as result a threshold of **146.2247**, corresponding to the spatial information mean calculated using the formula presented in 3.2. Using the mean to agglomerate the slices in a 4-by-4 matrix resulted in the following:

$$\begin{bmatrix} 19.3238 & 11.0339 & 10.8418 & 10.3370 \\ 200.5248 & 189.6603 & 187.8599 & 168.7717 \\ 195.8138 & 273.3634 & 258.7083 & 214.9036 \\ 94.6004 & 125.6495 & 193.8538 & 184.3490 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 4 & 4 & 4 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 3 & 5 & 5 & 5 \end{bmatrix} \tag{5.4}$$

which represents the video being partitioned using five non-linear slices corresponding to the figure B.5 displayed in the Appendix B. This demonstrates how the high detailed areas are spread in the middle and bottom of the video, having a clear contrast to the location of the sky on the upper part of the video.

5.3 Diagonal-moving video

The table 5.5 below shows the results of all the five methods applied to the diagonal-moving video. As is demonstrated, encoding the video without any slicing technique yields an mediocre results in quality and compression rate, which comparing to the previous tests, is significantly lower. However encoding the video with linear slicing techniques results in two distinct situations, where horizontal method apparently has better PSNR but less SSIM Y than the non slicing version and the vertical method results in a low quality encoding output, although both methods produce a similar file sized output.

Slicing Method	Total Processing time (s)	Average FPS (FPS)	Average Global PSNR (dB)	Average SSIM Y	Total Bit rate (kb/s)	File size (KB)
No Slice	4019.75	0.010000	34.606	0.957770	37177.98	2323.624
Horizontal	6352.59	0.091176	35.826	0.919258	70502.19	4406.387
Horizontal 512	5268.16	0.055556	35.882	0.927687	52195.14	3262.196
Vertical	5335.66	0.170000	31.981	0.922065	73762.58	4610.161
Vertical 512	4941.77	0.092667	32.857	0.935776	53799.88	3362.492
Non-linear	4895.94	0.085000	41.251	0.968017	39527.83	2470.489

Figure 5.5: Results for different methods used in an diagonal-moving 3.3 video.

The directionality and movement speed used on this video provided a challenge, as using linear slicing method would result in losing the spatial information by breaking into slices that were too small to harbour the movement within those same slices. However encoding the video without slicing creates the same situation detailed in the last section 5.2, where the spatial activity is shifted up and comparing to the first frame, the last frame has an increased detail. Applying a non-linear slicing method reduces that effect by removing the inter-dependency between areas throughout the frames thus making them more independent. The results, as seen in the table, clearly highlight that this method generates a substantial superior video quality than the non slicing method but a similar compression rate.

5.3.1 Directionality Detection Algorithm

The directionality detection algorithm used to verify the type of displacement used in the diagonal-moving video originated the following results:

$$\begin{bmatrix} 55.3906 & 47.4062 & 68.5312 \\ 68.2656 & 31.7500 & 42.0781 \\ 83.2969 & \mathbf{24.8594} & 52.1562 \end{bmatrix} \tag{5.5}$$

where the lowest value is located on the middle bottom cell and is above the 20 threshold limit. Since the developed custom *Fast Search Algorithm* can only search in nine directions, as displayed in figure 3.8, and the angle which the video was moving was not in a perfect 45 degrees, it meant it would fall under the vertical bottom direction. The threshold was added to prevent a linear slicing if the minimum error was too big, as proven from the encoding results from either horizontal or vertical slicing methods.

5.3.2 Spatial Information Algorithm

The algorithm used to verify the spatial activity across all the frames, regarding the diagonal-moving video, yield as result a threshold of **117.3387**, corresponding to the spatial information mean calculated using the formula presented in 3.2. Using the mean to agglomerate the slices in a 4-by-4 matrix resulted in the following:

$$\begin{bmatrix} 13.6023 & 10.3311 & 17.8807 & 12.2299 \\ 21.9464 & 39.6727 & 131.7245 & 36.5074 \\ 119.1020 & 197.1174 & 232.9168 & 149.9669 \\ 239.0337 & 232.1362 & 232.1158 & 191.1356 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 1 & 5 & 7 \\ 2 & 4 & 6 & 8 \\ 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 \end{bmatrix} \tag{5.6}$$

which represents the video being partitioned using eight non-linear slices corresponding to the figure C.5, displayed in the Appendix C. The graph illustrates a clear change in processing time and PSNR from the slice number three where all the detail is concentrated, to the slices number five and seven which represent the area with the lowest details.

5.4 Zoom-in video

Slicing Method	**Total Processing time** (s)	**Average FPS** (FPS)	**Average Global PSNR** (dB)	**Average SSIM Y**	**Total Bit rate** (kb/s)	**File size** (KB)
No Slice	5669.03	0.010000	46.447	0.980707	12516.98	782.311
Horizontal	4959.60	0.106471	46.395	0.980235	12899.41	806.213
Horizontal 512	4925.45	0.057778	46.426	0.980582	12755.67	797.229
Vertical	4787.93	0.195667	46.645	0.979860	15270.53	954.408
Vertical 512	4552.69	0.102000	46.685	0.979984	14008.85	875.553
Non-linear	5234.32	0.048000	46.647	0.981000	12917.13	807.321

Figure 5.6: Results for different methods used in an zoom-in 3.4 video.

As shown in the table 5.6, all the different methods produce very similar outputs, which contrasts with the previous diagonal-moving video test. The non slicing method originates an encoded file with a compression rate of **99.95%** compared to the $1.495GB$ sub-sampled YUV file. As displayed in the figure 3.4, this video does not provide much variation through the time, having the centre triangle growing while the background gets slightly darker. This subtle change which is neither horizontal nor vertical, reflects on the encoded values resulting on quite similar objective

quality and compressing rates. The same concept applies to the non-linear method where the result, although slightly higher than the remaining methods, is very similar and uniform in a global perspective.

5.4.1 Directionality Detection Algorithm

The directionality detection algorithm used to verify the type of displacement used in the zoom-in video originated the following results:

$$\begin{bmatrix} 26.0156 & 42.4062 & 19.7969 \\ 27.5938 & \mathbf{1.0938} & 28.4531 \\ 23.3594 & 33.8281 & 10.2188 \end{bmatrix} \tag{5.7}$$

which has a rather low minimum error value in the centre of the matrix, clearly emphasising the orientation of the video is centre-wise.

5.4.2 Spatial Information Algorithm

The algorithm used to verify the spatial activity across all the frames, regarding the zoom-in video, yield as result a threshold of **137.2124**, corresponding to the spatial information mean calculated using the formula presented in 3.2. Using the mean to agglomerate the slices in a 4-by-4 matrix resulted in the following:

$$\begin{bmatrix} 138.7981 & 155.1666 & 149.3767 & 143.1667 \\ 138.6554 & 133.1716 & 127.2327 & 149.9620 \\ 143.2250 & 92.9969 & 92.8914 & 140.1760 \\ 143.7845 & 150.4336 & 157.8363 & 138.5246 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 2 & 2 & 2 \\ 1 & 2 & 2 & 2 \\ 1 & 3 & 3 & 5 \\ 1 & 4 & 4 & 4 \end{bmatrix} \tag{5.8}$$

which represents the video being partitioned using five non-linear slices corresponding to the figure D.5 displayed in the Appendix D. The partition of the matrix illustrates that the detail of this video is located on the edge of the video, while the triangle in the centre corresponds to the lowest detailed area, thus the gain in PSNR and SSIM Y portrayed in the graphic D.5a.

Chapter 6

Conclusion and Future work

According to the results described previously in Chapter 5, the use of the linear slicing method is able to deliver better quality-compression ratios when combined with the correct slicing orientation. However, when analysing on a global perspective and considering only the optimization of the quality-compression ratio, it is not as efficient as the straightforward encoding method, which does not slice the video. This is due to the fact that the algorithm is ignoring the spatial information and slicing in a pre-determined area, regardless of its content as displayed in the figure 5.2. Nevertheless, the straightforward non-slicing method does not profit from the possibility of parallelizing the processing tasks, which either increases the overall encoding time of the video or requires extremely powerful machines. When comparing the 256 to the 512 step slice either with a horizontal or vertical orientation, the latter slicing step always proved to originate better results, as less spatial information correlation is broken. Those results reinforce the idea that the method used by NHK, in collaboration with Mitsubishi Electric, of always slicing horizontally with a fixed 256 step, when implementing it in a real-time encoding system, is not always the best approach [14].

The non-linear slicing method always yielded positive results comparing to the non slicing method, exceeding it by a large margin in some cases. This method proved to be far superior than the linear approach, as it takes the shift of spatial activity across the frames into consideration and slices accordingly. The algorithm could still be improved to produce better results, which could imply a further division of the 4-by-4 1920×1080 area to a 8-by-8 960×540 area, covering smaller areas and isolating the lower detailed areas more effectively. The intrinsic problem of this method, however, is the fact that it has to check the spatial activity every few frames to accurately measure how the displacement will affect it. This would imply that in a real-time encoding environment the system would have to buffer a few frames to determine how the slicing would proceed. The non-linear reconstruction itself also has a heavier computational algorithm than when reconstructing a linear sliced video. Since this method was developed to test if the quality and compression rate would improve compared to a linear method or a non slicing method, a real-time encoding environment was discarded, as it would be harder to implement.

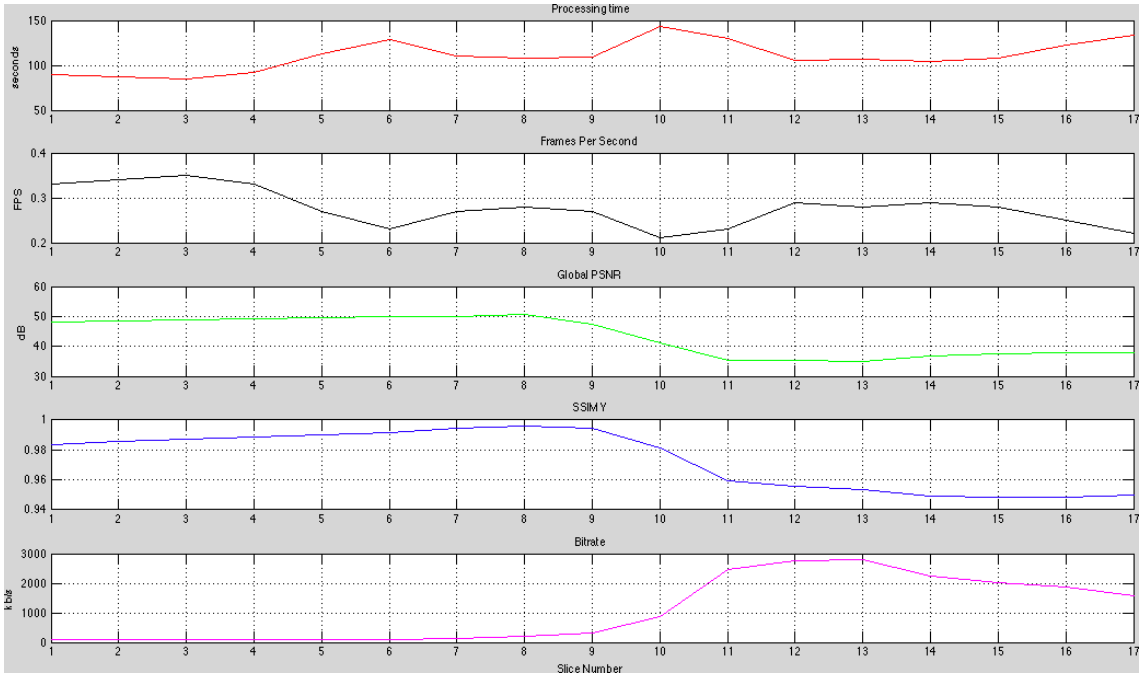
Taking advantage of slicing, one future implementation of this system could be to implement

a distributed system that allowed further parallel processing, reducing the overall encoding time. This system could have dedicated servers that would only encode slices when requested and one central hub that would request those slices and would begin to reconstruct the file as they were arriving. By using these types of distributed systems, the computational power could be shifted, allowing this linear or non-linear slicing method to be used in a real-time encoding environment. This implementation could also mean the HEVC external encoding program used would be independent from the system, which would allow it to be updated when new versions were released. This could increase the overall quality and compression rate without much alterations in the slicing algorithm.

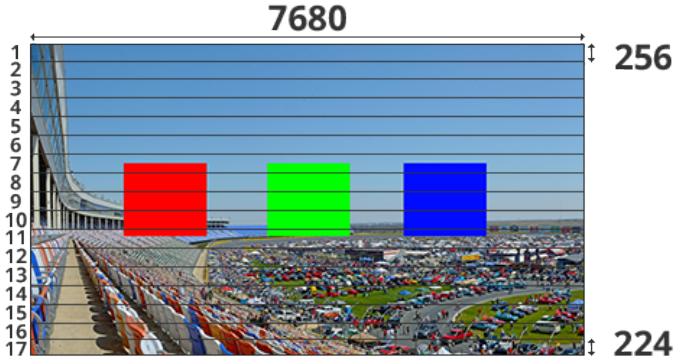
Appendix A

Horizontal-Moving Slicing Images

In this annex, the images from the horizontal-moving video regarding the different slicing methods and their respective graphs will be illustrated. The images portrayed were all resized to improve readability. The graphics displayed show a detailed analysis of each slice in the video and the image exhibiting the slicing boundaries represents the respective first frame. The sequence of the methods represented in this annex is: Horizontal 256 (A.1), Horizontal 512 (A.2), Vertical 256 (A.3), Vertical 512 (A.4) and Non-linear (A.5).

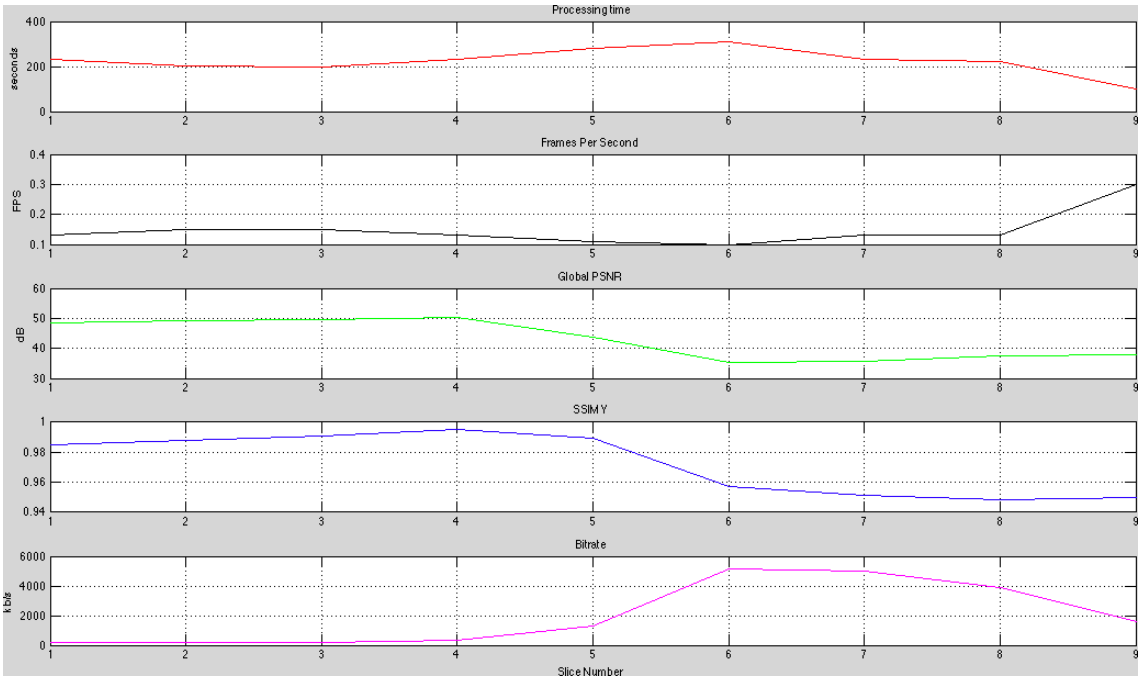


(a) Horizontal Slicing Graph

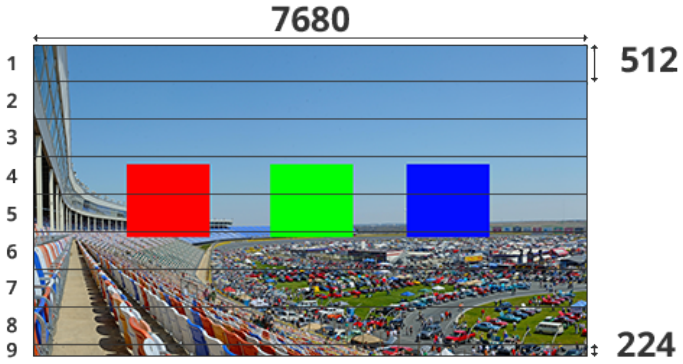


(b) Horizontal Slicing Boundaries

Figure A.1: Horizontal Slicing with 256 step.

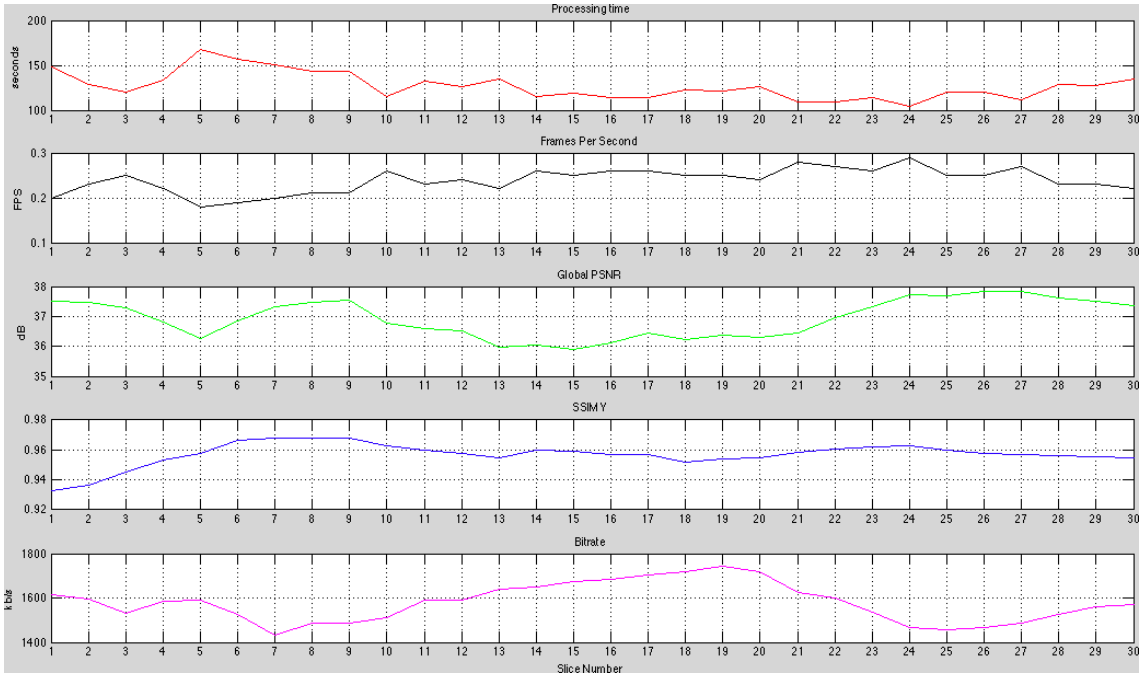


(a) Horizontal Slicing Graph

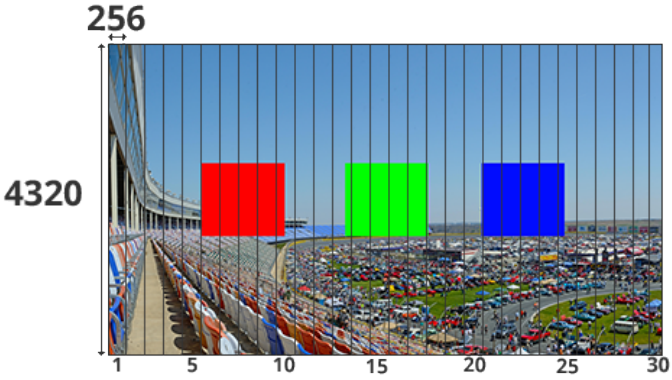


(b) Horizontal Slicing Boundaries

Figure A.2: Horizontal Slicing with 512 step.

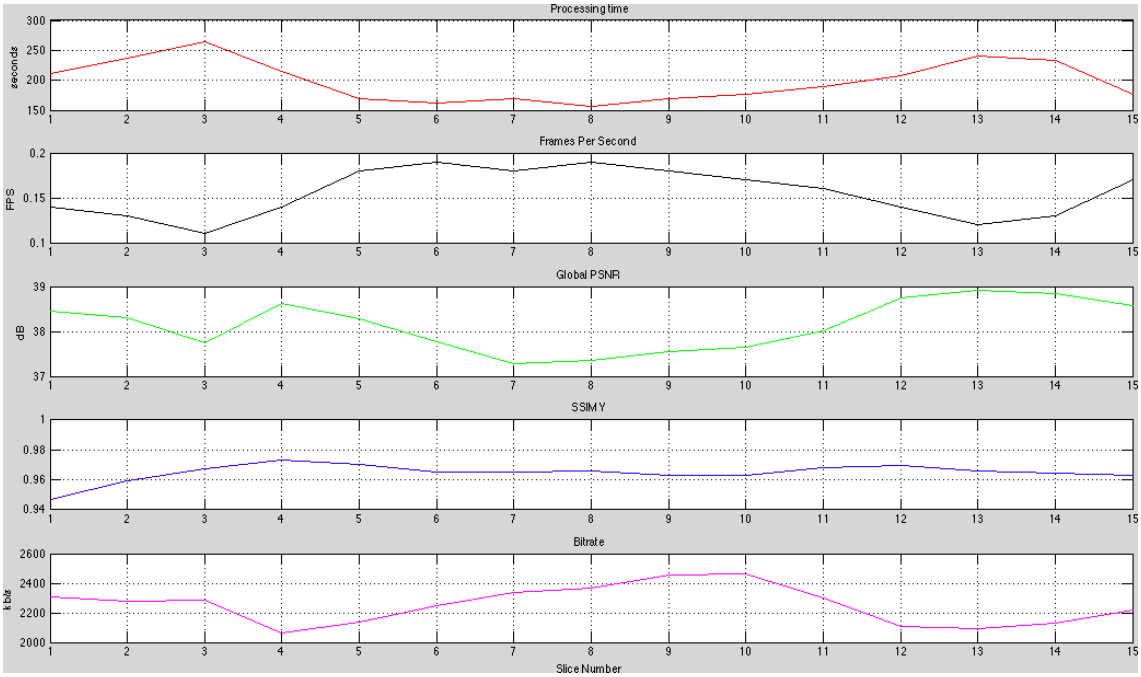


(a) Vertical Slicing Graph

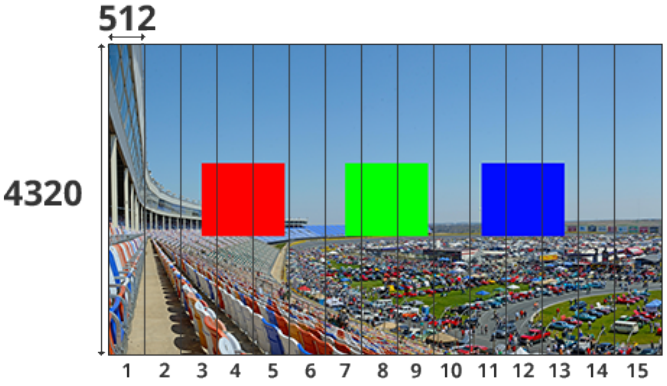


(b) Vertical Slicing Boundaries

Figure A.3: Vertical Slicing with 256 step.

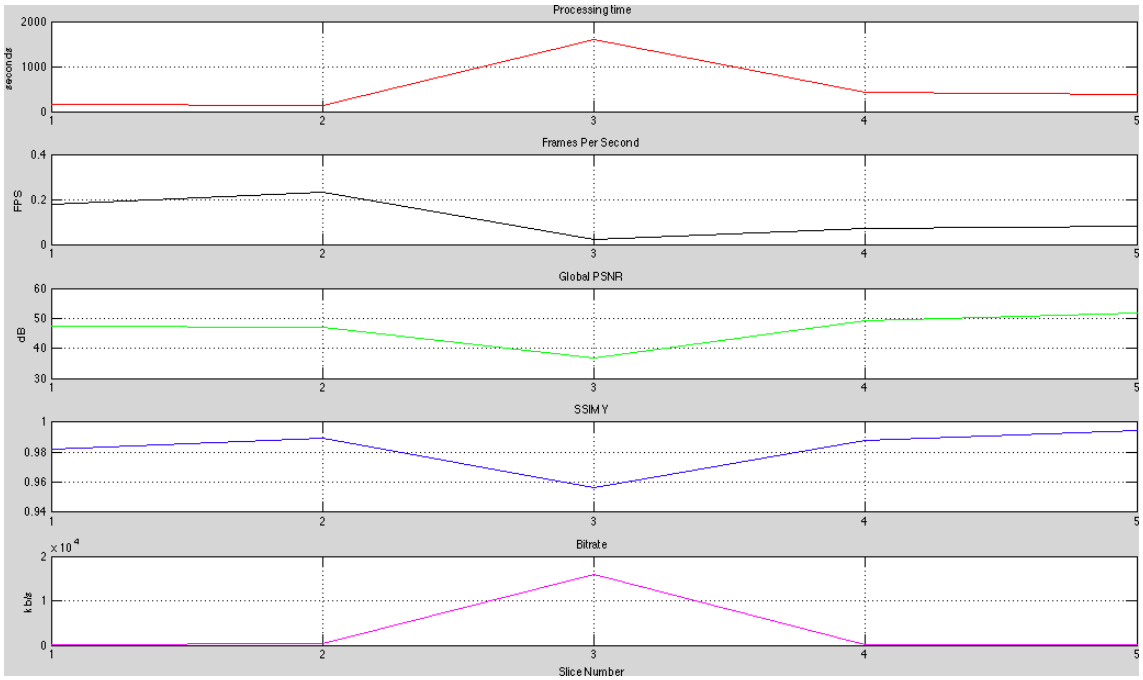


(a) Vertical Slicing Graph

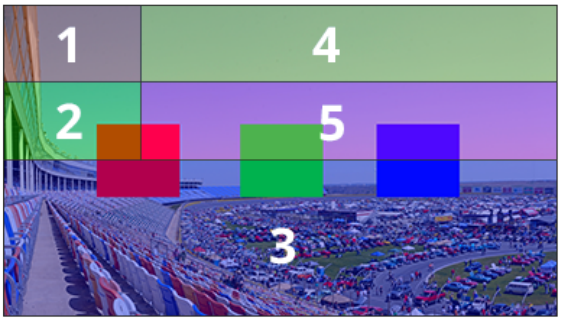


(b) Vertical Slicing Boundaries

Figure A.4: Vertical Slicing with 512 step.



(a) Non-Linear Slicing Graph



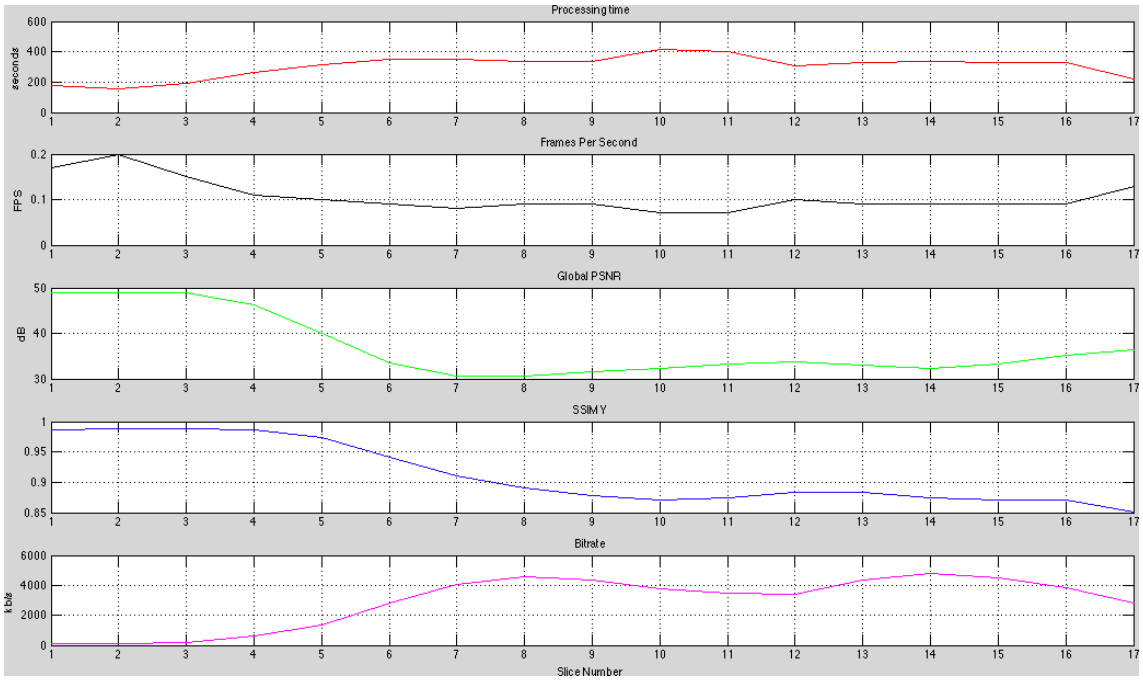
(b) Non-Linear Slicing Boundaries

Figure A.5: Non-Linear Slicing

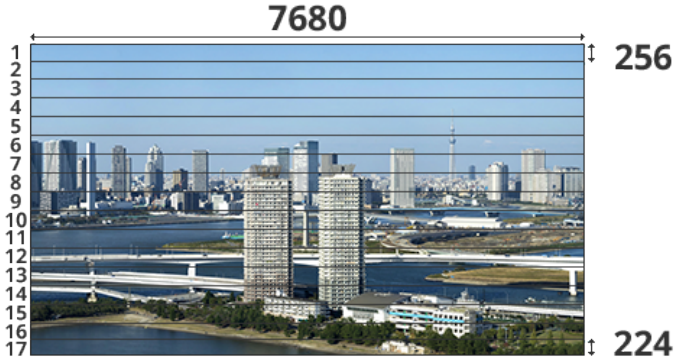
Appendix B

Vertical-Moving Slicing Images

In this annex, the images from the vertical-moving video regarding the different slicing methods and their respective graphs will be illustrated.The images portrayed were all resized to improve readability. The graphics displayed show a detailed analysis of each slice in the video and the image exhibiting the slicing boundaries represents the respective first frame. The sequence of the methods represented in this annex is: Horizontal 256 (B.1), Horizontal 512 (B.2), Vertical 256 (B.3), Vertical 512 (B.4) and Non-linear (B.5).

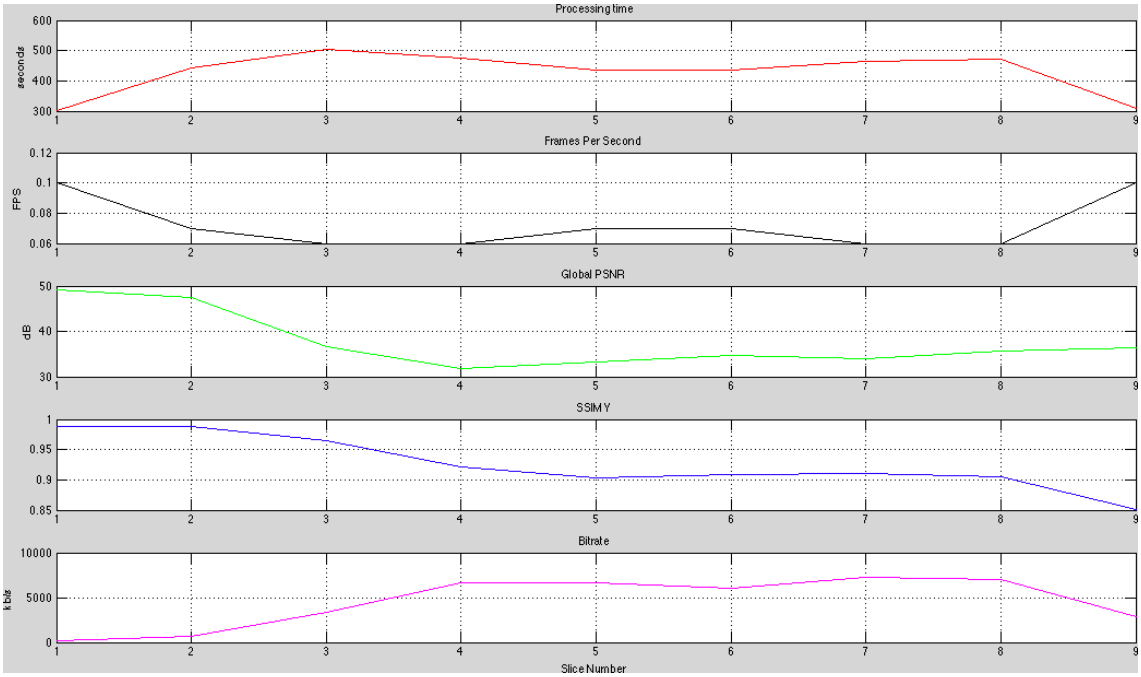


(a) Horizontal Slicing Graph

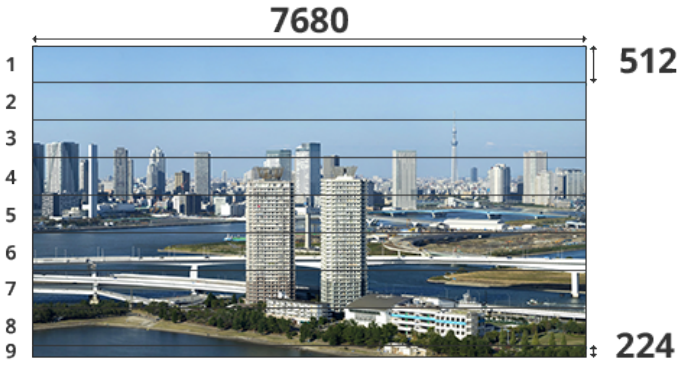


(b) Horizontal Slicing Boundaries

Figure B.1: Horizontal Slicing with 256 step.

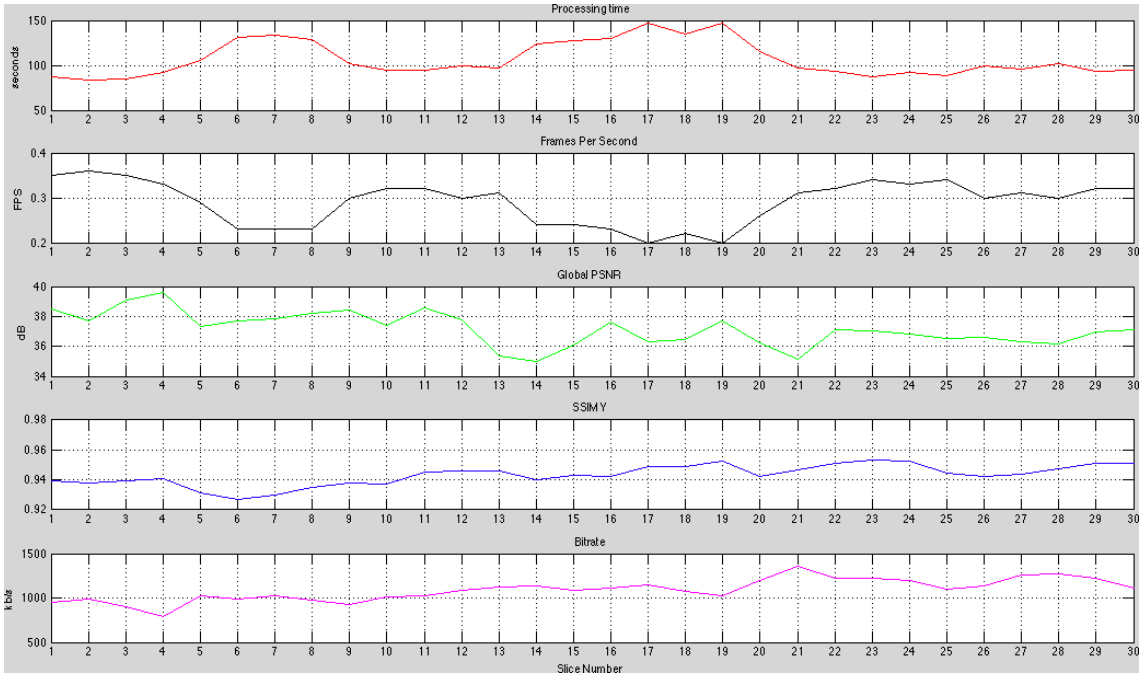


(a) Horizontal Slicing Graph

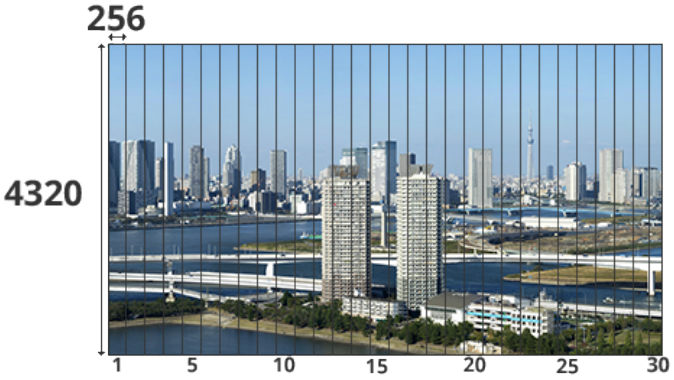


(b) Horizontal Slicing Boundaries

Figure B.2: Horizontal Slicing with 512 step.

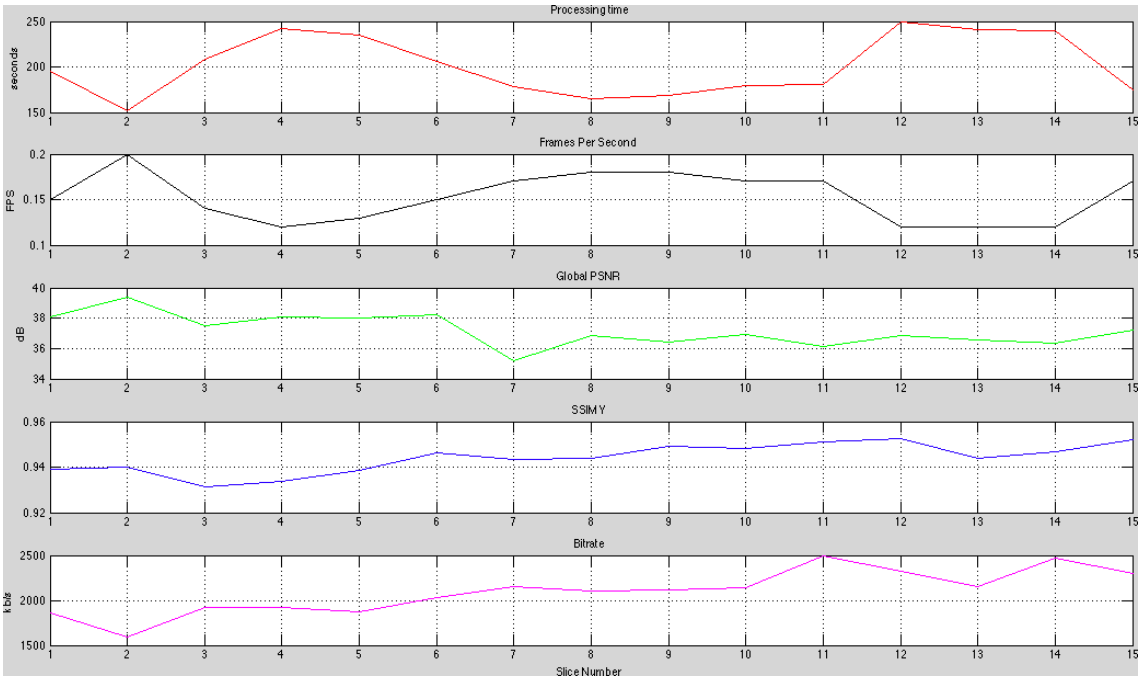


(a) Vertical Slicing Graph

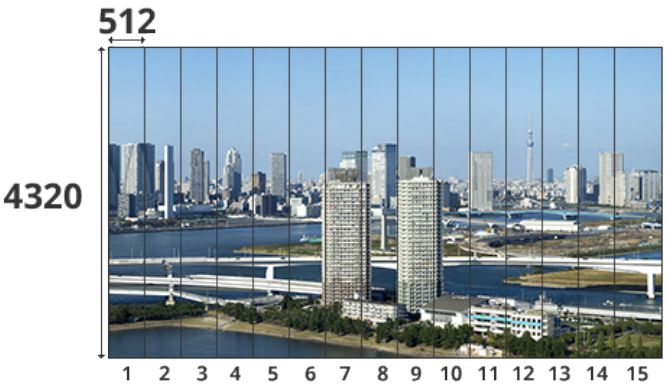


(b) Vertical Slicing Boundaries

Figure B.3: Vertical Slicing with 256 step.

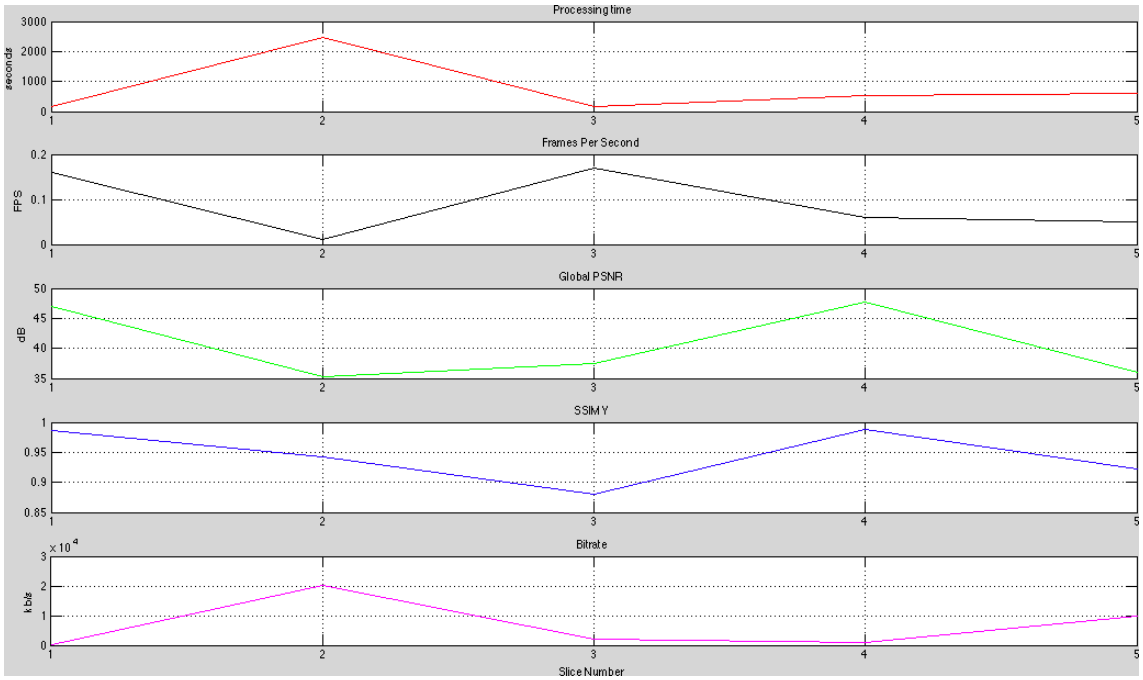


(a) Vertical Slicing Graph



(b) Vertical Slicing Boundaries

Figure B.4: Vertical Slicing with 512 step.



(a) Non-Linear Slicing Graph

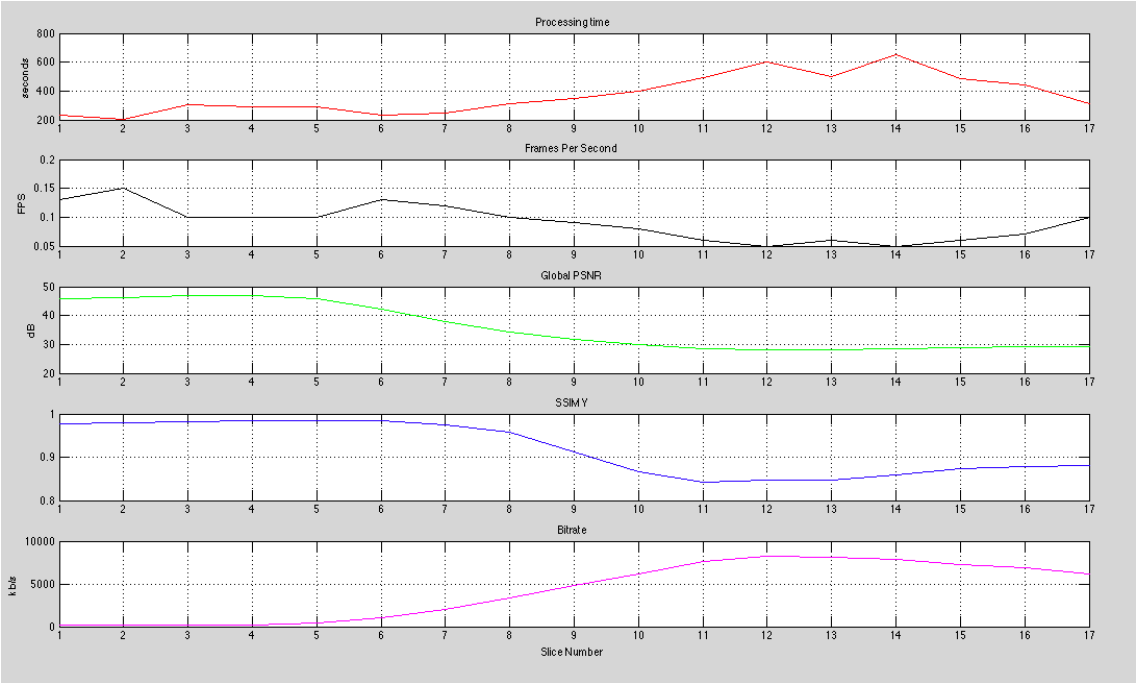
(b) Non-Linear Slicing Boundaries

Figure B.5: Non-Linear slicing

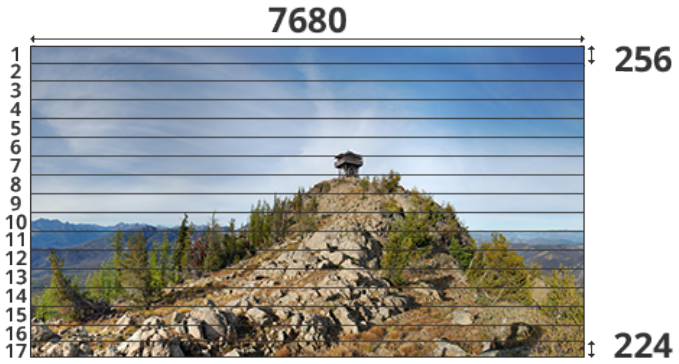
Appendix C

Diagonal-Moving Slicing Images

In this annex, the images from the diagonal-moving video regarding the different slicing methods and their respective graphs will be illustrated. The images portrayed were all resized to improve readability. The graphics displayed show a detailed analysis of each slice in the video and the image exhibiting the slicing boundaries represents the respective first frame. The sequence of the methods represented in this annex is: Horizontal 256 (C.1), Horizontal 512 (C.2), Vertical 256 (C.3), Vertical 512 (C.4) and Non-linear (C.5).

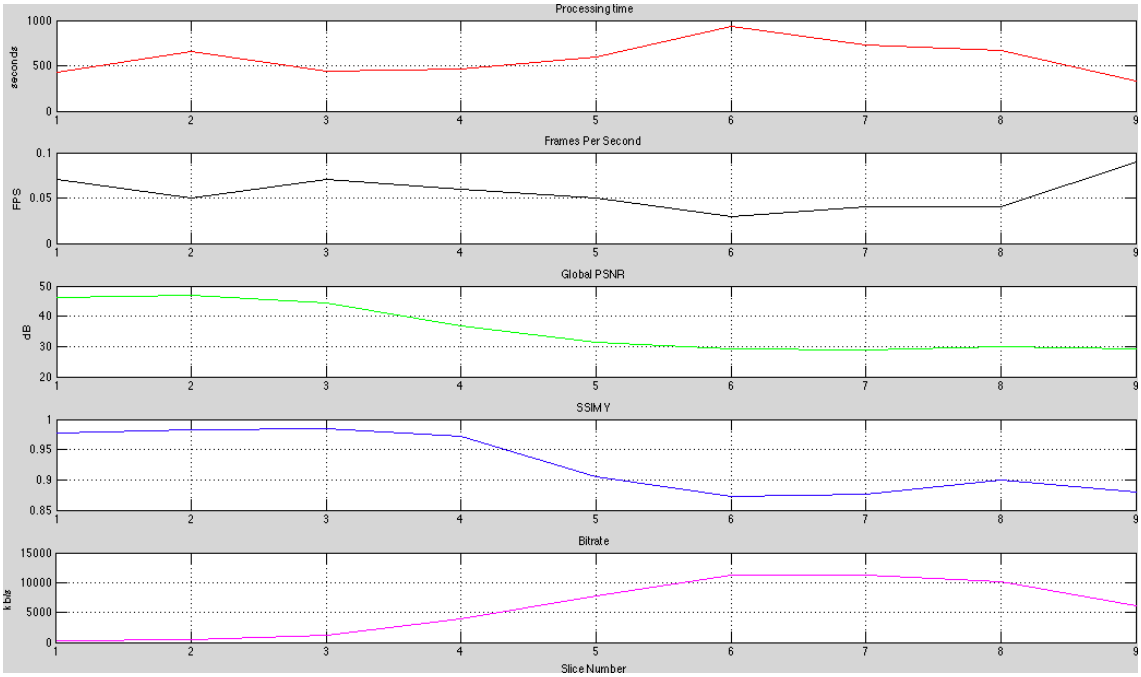


(a) Horizontal Slicing Graph

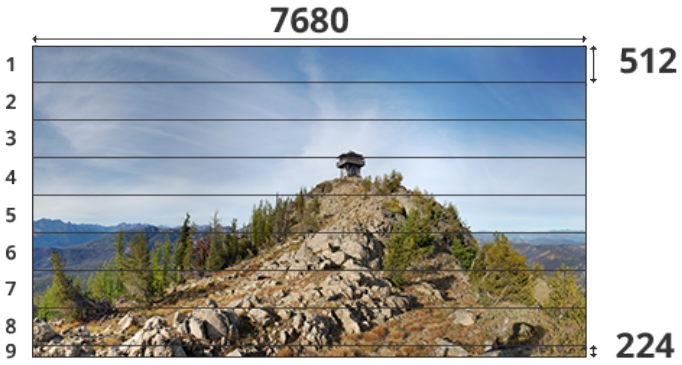


(b) Horizontal Slicing Boundaries

Figure C.1: Horizontal Slicing with 256 step.

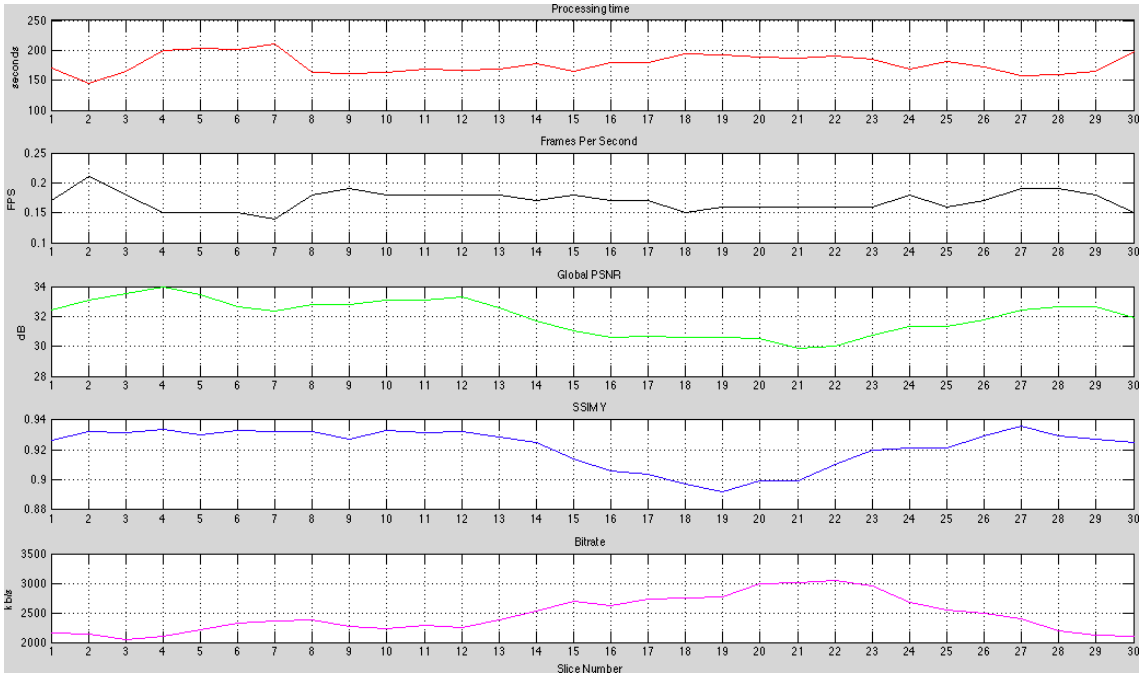


(a) Horizontal Slicing Graph

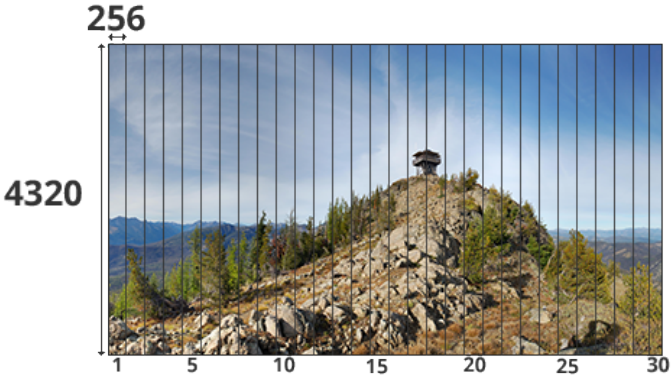


(b) Horizontal Slicing Boundaries

Figure C.2: Horizontal Slicing with 512 step.

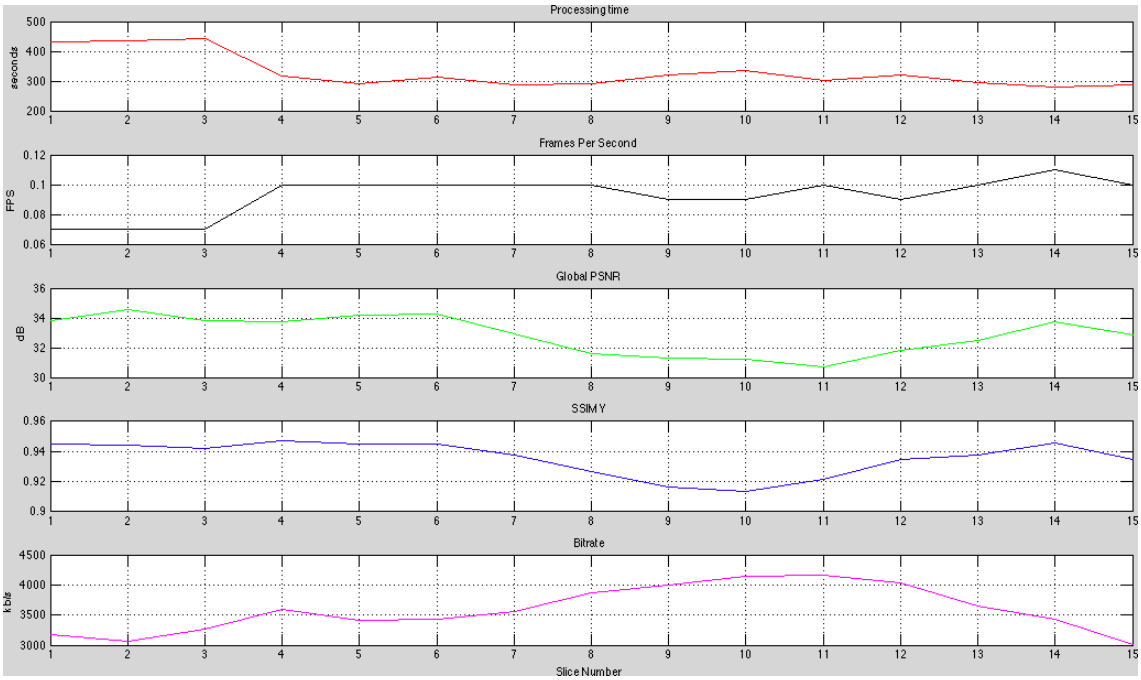


(a) Vertical Slicing Graph

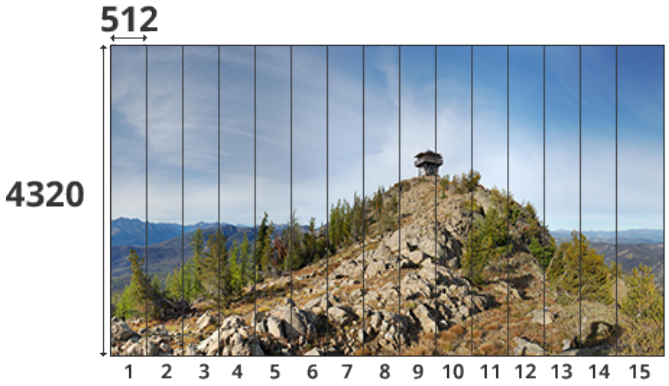


(b) Vertical Slicing Boundaries

Figure C.3: Vertical Slicing with 256 step.

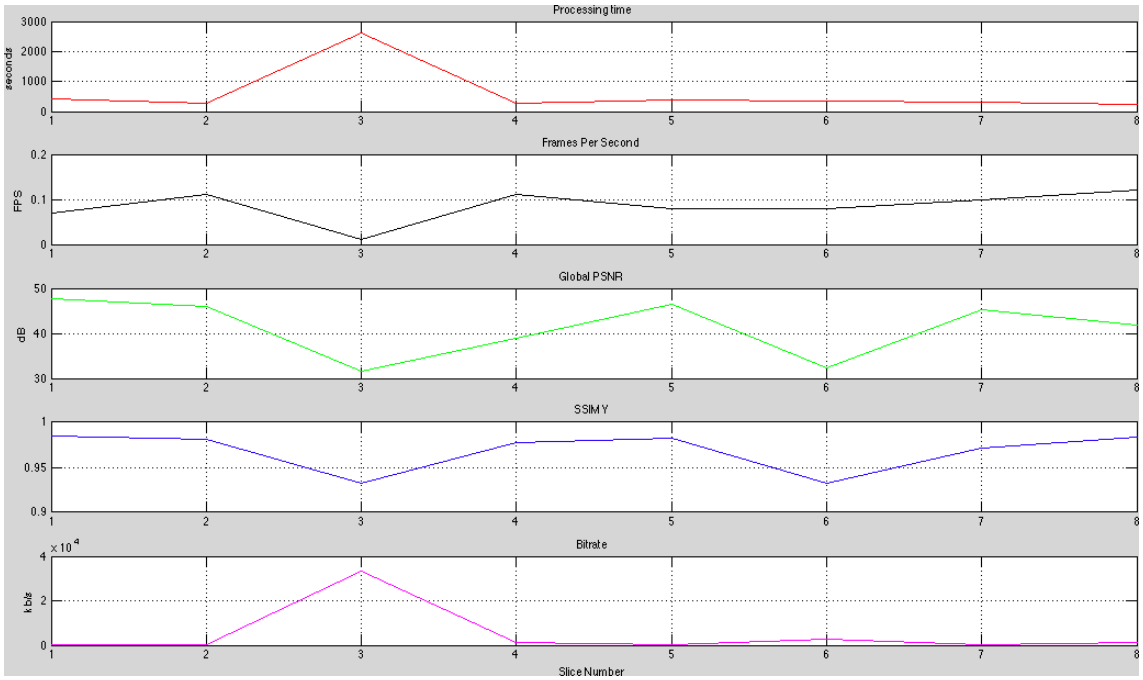


(a) Vertical Slicing Graph



(b) Vertical Slicing Boundaries

Figure C.4: Vertical Slicing with 512 step.



(a) Non-Linear Slicing Graph

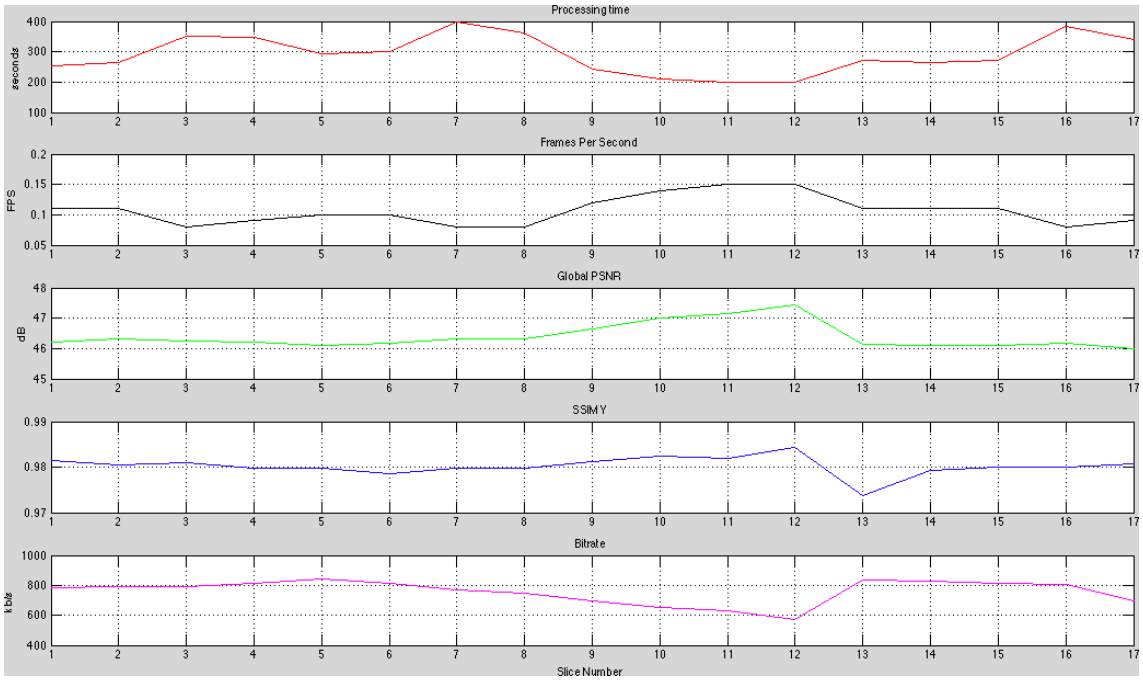
(b) Non-Linear Slicing Boundaries

Figure C.5: Non-Linear Slicing

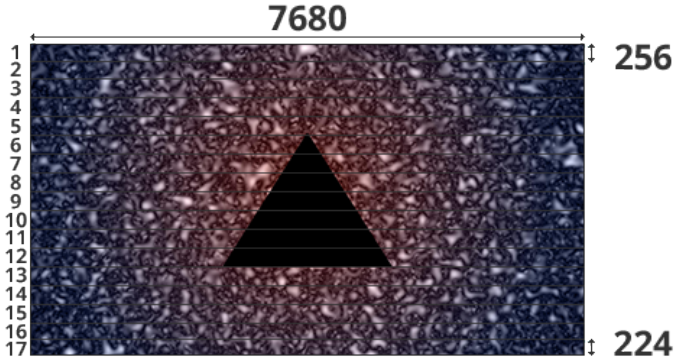
Appendix D

Zoom-In Slicing Images

In this annex, the images from the zoom-in video regarding the different slicing methods and their respective graphs will be illustrated. The images portrayed were all resized to improve readability. The graphics displayed show a detailed analysis of each slice in the video and the image exhibiting the slicing boundaries represents the respective first frame. The sequence of the methods represented in this annex is: Horizontal 256 (D.1), Horizontal 512 (D.2), Vertical 256 (D.3), Vertical 512 (D.4) and Non-linear (D.5).

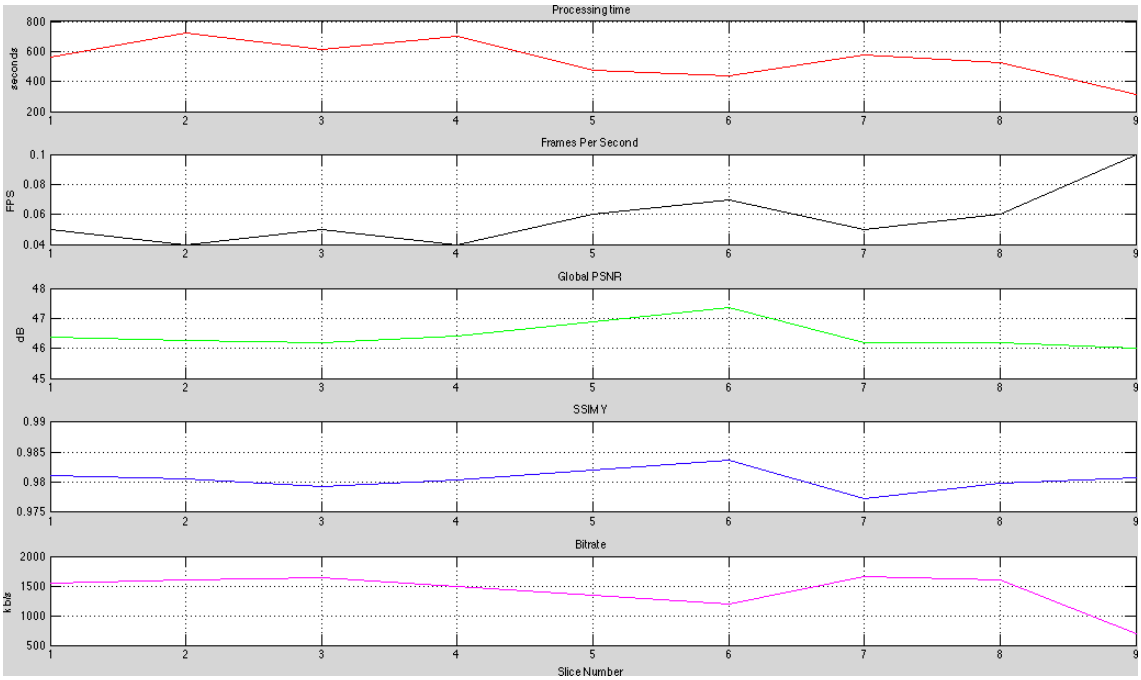


(a) Horizontal Slicing Graph

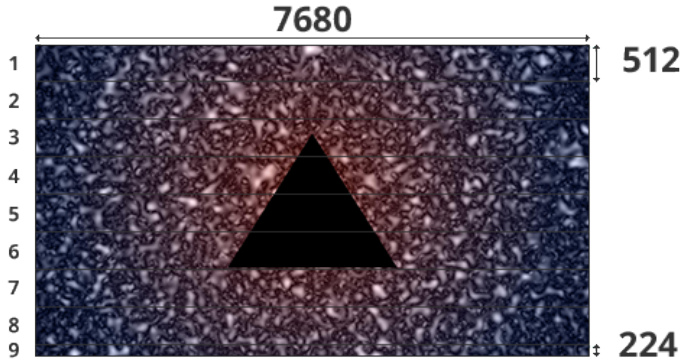


(b) Horizontal Slicing Boundaries

Figure D.1: Horizontal Slicing with 256 step.

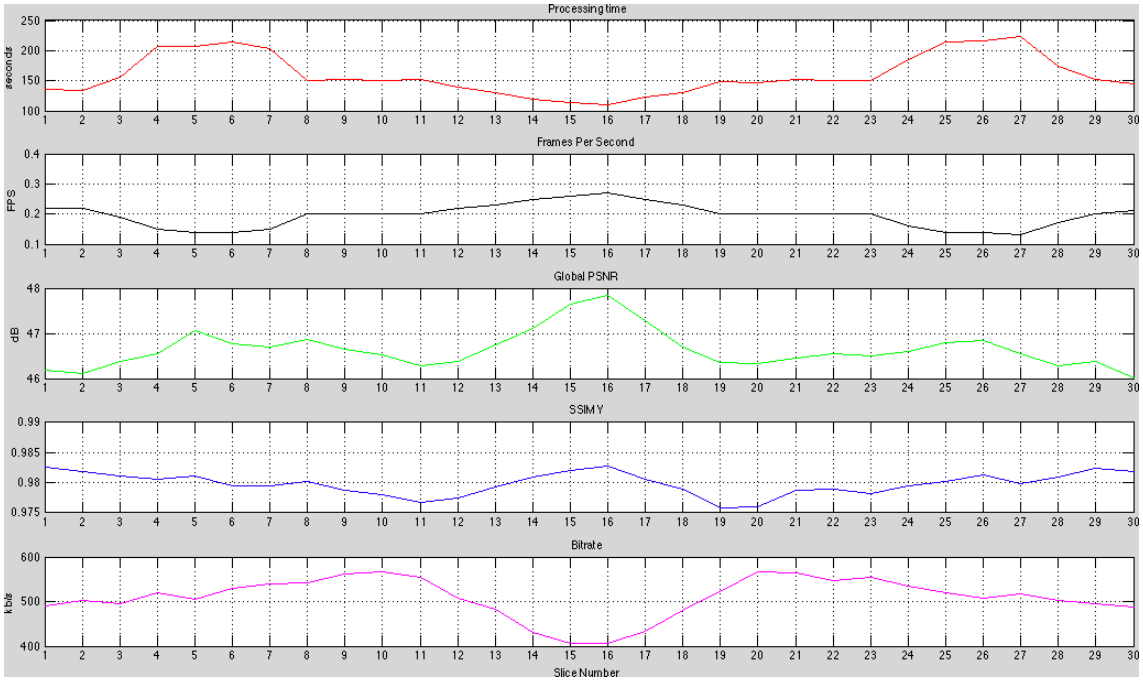


(a) Horizontal Slicing Graph

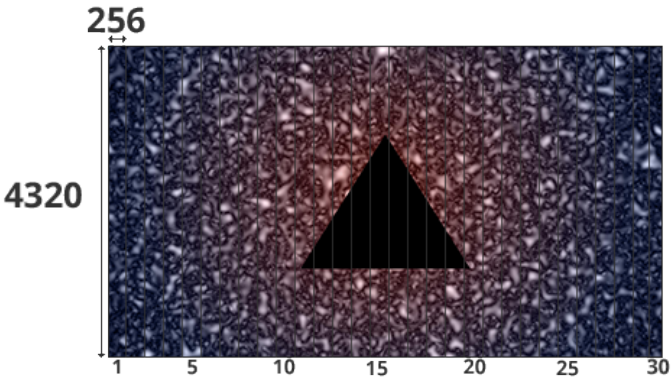


(b) Horizontal Slicing Boundaries

Figure D.2: Horizontal Slicing with 512 step.

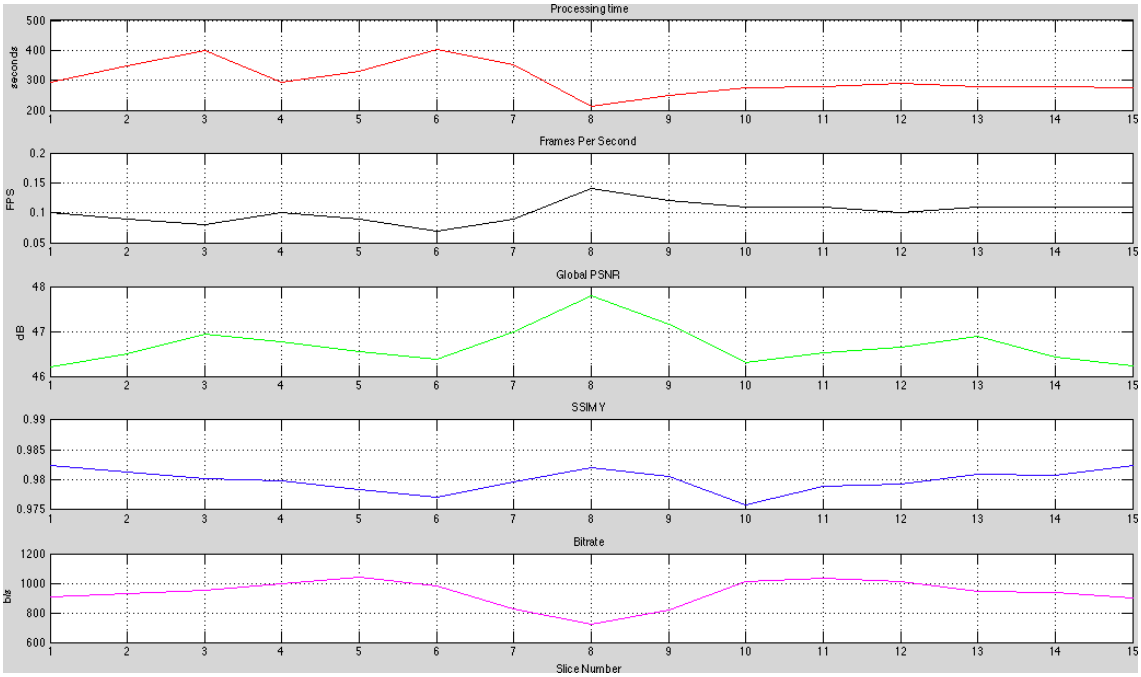


(a) Vertical Slicing Graph

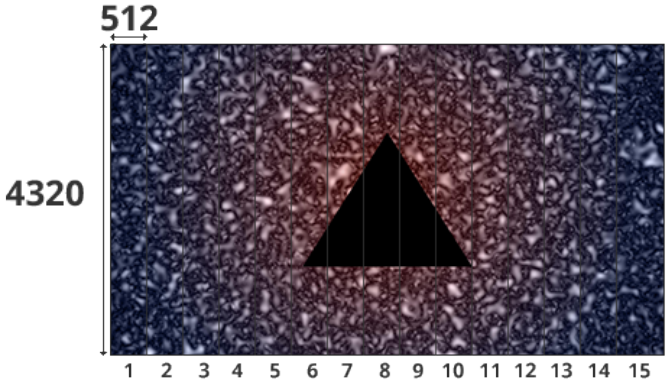


(b) Vertical Slicing Boundaries

Figure D.3: Vertical Slicing with 256 step.

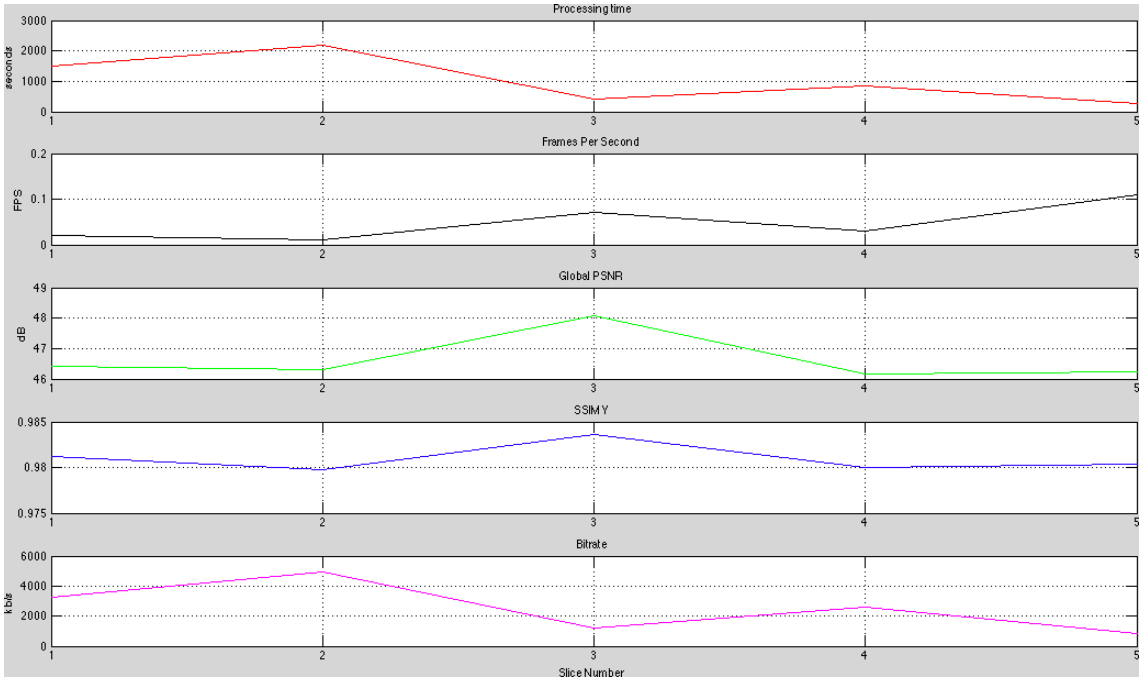


(a) Vertical Slicing Graph

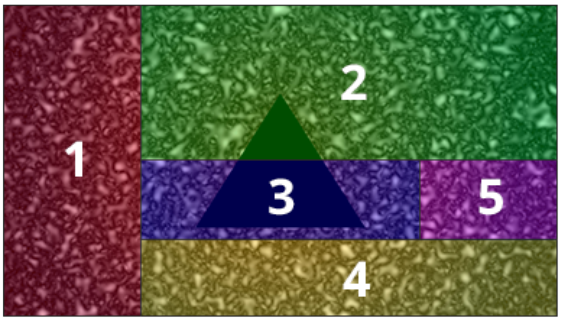


(b) Vertical Slicing Boundaries

Figure D.4: Vertical Slicing with 512 step.



(a) Non-Linear Slicing Graph



(b) Non-Linear Slicing Boundaries

Figure D.5: Non-Linear Slicing

References

[1] 8K resolution - Wikipedia. http://en.wikipedia.org/wiki/8K_resolution.

[2] SMPTE. Initial Report of the UHDTV Ecosystem Study Group. Technical report, Society of Moving Picture and Television Engineers, 2013.

[3] Neowin. ITU approves 8K ultra HDTV specification. http://www.neowin.net/news/itu-approves-8k-ultra-hdtv-specification.

[4] NHK. NHK Open House 2013 Exhibition. http://www.nhk.or.jp/strl/open2013/tenji/pdf/open2013_siryo_e.pdf.

[5] ITU-R BT.2246-2. The present state of ultra-high definition television. Technical report, International Telecommunication Union, 2012.

[6] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, December 2012.

[7] HEVC Software CODEC HEVC-1000 SDK | NTT-AT. http://www.ntt-at.com/product/rfs_hevc_sdk/.

[8] RECOMMENDATION ITU-R BT.2020. Parameter values for ultra-high definition television systems for production and international programme exchange. Technical report, International Telecommunication Union, 2012.

[9] RECOMMENDATION ITU-R BT.709. Parameter values for the HDTV standards for production and international programme exchange. Technical report, International Telecommunication Union, 2002.

[10] RECOMMENDATION ITU-R BT.1845. Guidelines on metrics to be used when tailoring television programmes to broadcasting applications at various image quality levels, display sizes and aspect ratios. Technical report, International Telecommunication Union, 2010.

[11] 3-D TV is Officially Dead (For Now) and This is Why it Failed - IEEE Spectrum. http://spectrum.ieee.org/tech-talk/consumer-electronics/audiovideo/3d-tv-is-officially-dead-for-now-and-this-is-why-it-failed, 2014.

[12] Health effects of 3D - Wikipedia. http://en.wikipedia.org/wiki/Health_effects_of_3D.

[13] Pierre Larbier. 4K DELIVERY TO THE HOME. http://ateme.com/IMG/pdf/4k_delivery_to_the_home_-_pierre_larbier_-_ateme.pdf, 2012.

[14] 8K Ultra HD compact camera and H.265 encoder developed by NHK with UHD trial broadcasts slated for 2016 - DigInfo TV. http://www.diginfo.tv/v/13-0043-r-en.php.

[15] "Super Hi-Vision" as Next-Generation Television and Its Video Parameters. http://informationdisplay.org/IDArchive/2012/NovemberDecember/FrontlineTechnologySuperHiVisionasNextGen.aspx.

[16] RECOMMENDATION ITU-R BT.1127. Relative quality requirements of television broadcast systems. Technical report, International Telecommunication Union, 1994.

[17] NHK. Long-distance transmission test is successfully achieved. http://www.nhk.or.jp/pr/english/press/pdf/20140203.pdf, 2014.

[18] Elemental Technologies. HEVC DEMYSTIFIED - A Primer on the H.265 Video Codec. Technical report, 2013.

[19] Honghai Yu and Stefan Winkler. Image complexity and spatial information. In *2013 Fifth International Workshop on Quality of Multimedia Experience (QoMEX)*, pages 1–6. IEEE, July 2013.