OPEN ACCESS

University of BRISTOL

This electronic thesis or dissertation has been downloaded from Explore Bristol Research, http://research-information.bristol.ac.uk

Author:
Fernandez Afonso, Mariana

Title:
Intelligent Resampling Methods for Video Compression

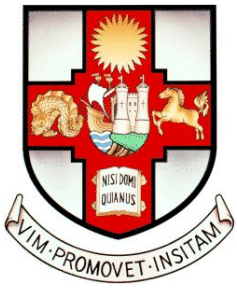
General rights
Access to the thesis is subject to the Creative Commons Attribution - NonCommercial-No Derivatives 4.0 International Public License. A copy of this may be found at https://creativecommons.org/licenses/by-nc-nd/4.0/legalcode This license sets out your rights and the restrictions that apply to your access to the thesis so it is important you read this before proceeding.

Take down policy
Some pages of this thesis may have been removed for copyright restrictions prior to having it been deposited in Explore Bristol Research. However, if you have discovered material within the thesis that you consider to be unlawful e.g. breaches of copyright (either yours or that of a third party) or any other law, including but not limited to those relating to patent, trademark, confidentiality, data protection, obscenity, defamation, libel, then please contact collections-metadata@bristol.ac.uk and include the following information in your message:

• Your contact details
• Bibliographic details for the item, including a URL
• An outline nature of the complaint

Your claim will be investigated and, where appropriate, the item in question will be removed from public view as soon as possible.

Intelligent Resampling Methods for Video Compression



Mariana Fernandez Afonso

Department of Electrical and Electronic Engineering
University of Bristol

A dissertation submitted to the University of Bristol in accordance with the requirements for award of the degree of Doctor of Philosophy in the Faculty of Engineering.

May 2019

Word Count: 38370

Acknowledgements

First and foremost I would like to thank my supervisor Professor David Bull for giving me the opportunity to pursue a PhD at the University of Bristol and as part of the most his amazing projects in video coding, the PROVISION ITN. I am also grateful for his continuous supervision, insightful discussions and for securing funding for the last year of my PhD. Also, a big thanks to my co-supervisor, Dr. Dimitris Agrafiotis, for the discussions and help during my first research years.

A very special thanks to my post-docs colleagues and friends Angeliki Katsenou and Fan Zhang, with whom I worked closely in several projects during the past years. I have definitely become a better researcher because of your guidance!

A big thanks to all my colleagues in the VI-Lab, including (by not limited to): Alex Mackin, Miltiadis Papadopoulus, Pui Anantrasirichai, Odysseas Pappas, Perla Mayo, Di Ma, Paul Hill, Igor Rizaev and Oktay Karakus. I will miss our lovely discussions, shared meals, VI-Lab events and, especially, the delicious office chocolates!

I am also grateful for my fellow PROVISION researchers and supervisors, spread around the world, producing top-end research. Our meetings were very insightful, packed with interesting discussions and always a lot of fun!

This thesis would not have been possible without the support from my parents and brother in the times where I most needed it. Thank you for the unconditional love and advice throughout my PhD journey.

Finally, I would like to thank my loving boyfriend, Sergio, who has been by my side throughout this entire journey. Thank you for spending hours helping me revise this thesis, and, especially, for always believing in me!

Abstract

Video compression is the core process that allows efficient storage and transmission of digital video. The main concept is to exploit redundancies in the video signal, with the goal of achieving the best possible trade-off between bitrate and quality. This very active research field has seen great improvements in the last 30 years, from the simple methods of the early 1990s, to more complex but highly optimized systems of modern compression formats. At the same time, improvements in consumer devices and the availability of faster Internet connections has allowed for the growing popularity of digital video. Nonetheless, with an ever-increasing demand for better quality and more immersive experiences from consumers, pressure for improved compression algorithms persists. Since current systems are already highly optimized, the challenge arises for innovative techniques to provide further enhancements.

This thesis explores novel intelligent resampling-based methods for future video compression systems, from texture synthesis to spatial resolution adaptation. Texture synthesis has been proposed previously for the coding of difficult scenes, specially those featuring dynamic textures. However, the classification of dynamic textures is usually too broad and not well understood. For this reason, an extensive analysis of encoding statistics is presented based on a new homogenous texture dataset. In addition, limitations of prior dynamic texture synthesis approaches are presented and a new method is proposed. Moreover, we explore another type of resampling approach, known as spatial resolution adaptation. Our proposed framework dynamically adapts the spatial resolution of the input video based on a prediction that uses low-level visual features and later reconstructs the original resolution video at the decoder. Several methods are studied to achieve improved quality of the reconstructed video, including the use of Convolutional Neural Networks (CNNs) trained on a large dataset of videos using both pixel-based loss functions and a perceptually-inspired framework.

Experimental results show that the proposed spatial resolution adaptation system achieves significant gains over HEVC using objective quality metrics, visual comparisons and subjective tests. In addition, an early version of the proposed approach was submitted to the Video Compression Technology Grand Challenge at ICIP 2017, and won first prize. For these reasons, we believe that this work provides a valuable contribution and offers significant potential for enhancing video coding systems.

Declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

Mariana Fernandez Afonso
February 2019

Table of contents

List of figures

List of tables

List of Symbols

The list below describes several symbols that will be later employed within the body of the document

List of Acronyms / Abbreviations

AV1	AOMedia Video 1
BVI	Bristol Vision Institute
CNN	Convolutional Neural Network
CU	Coding Unit
CTU	Coding Tree Unit
DCT	Discrete Cosine Transform
DTS	Dynamic Texture Synthesis
DFT	Discrete Fourier Transform
DMOS	Differential Mean Opinion Score
GAN	Generative Adversarial Network
GOP	Group of Pictures
HD	High Definition
HDR	High Dynamic Range
HEVC	High Efficiency Video Coding
HFR	High Frame Rate
HM	HEVC Test Model
HR	High Resolution
HVS	Human Visual System
JVET	Joint Video Exploration Team
LCC	Linear Correlation Coefficient
LR	Low Resolution
MSE	Mean Squared Error
MOS	Mean Opinion Score
MV	Motion Vector

NN	Neural Network
OF	Optical Flow
PC	Personal Computer
PSNR	Peak Signal-to-Noise Ratio
QP	Quantisation Parameter
SI	Spatial Information
SR	Super-Resolution
RD	Rate-Distortion
TI	Temporal Information
UHD	Ultra High Definition
VQA	Video Quality Assessment
VMAF	Video Multimethod Assessment Fusion
VVC	Versatile Video Coding

Chapter 1

Introduction

1.1 Motivation

Visual content represents an indispensable part of modern society. Most social networks rely predominantly on image and video sharing, while online video streaming platforms have become extremely popular in recent years. These services are consumed by millions of people worldwide and make digital multimedia one of the most prominent types of entertainment of the 21st century. The introduction of inexpensive recording and viewing devices, including smartphones, means that these experiences are accessible to almost everyone. It is projected that, by 2022, video streaming and downloads will exceed 82% of all consumer internet traffic [14]. In comparison with images and other multimedia signals, video contains a large amount of information. For example, a raw or uncompressed 15 second HD resolution (1920x1080 pixels) video clip requires 1.4 Gigabytes of storage space when encoded at typical parameters such as 8-bits per colour channel and 30fps. Therefore, it is clear it would be highly impractical to store and/or transmit an uncompressed tv-series episode or a full-length movie. For this reason, video compression or coding is a critical tool that aims to reduce the size of digital video by leveraging the intrinsic redundancies of the video signal. These typically include spatial, temporal and information-based redundancies. Recent advances in this field have led to compression ratios as high as 1000:1, as is the case for the HEVC (High Efficiency Video Compression) video standard and further research and development is ongoing.

At the same time, there is an increasing demand for more realistic and immersive video experiences with higher image quality, larger resolutions (up to 4K or 8K), higher frame rates (60 fps or even 120 fps) and brighter colours using High Dynamic Ranges (HDR). However, these extended video parameters significantly increase the size of video files and put additional pressure on current video coding systems to achieve even higher compression efficiencies. Accounting for these factors, it is clear that new developments in video compression are crucial and can have a high impact on modern society.

Over the years, several techniques have been proposed to improve video compression efficiency. For instance, texture synthesis, a common technique employed in computer graphics, which generates

synthetic textures for video games and other applications, has been exploited for video compression. Such frameworks, known as texture analysis and synthesis, attempt to replace the implicit coding of specific texture regions by a synthesised reconstruction of the content a the decoder side. Based on the assumption that the perceived quality would be similar to encoding the original texture, these schemes offer the potential to achieve improved coding efficiency compared to conventional methods.

Another active avenue of research towards realising compression improvements concerns the further exploitation of spatial redundancy. Large video formats, such as 4K (3840×2160 pixels), contain an increased amount of spatial redundancy, which is not yet optimally exploited by current video compression algorithms. One technique that has proven to be useful is to encode a low resolution version of the raw video, instead of coding at the original resolution. This technique requires the compressed video to be upsampled at the decoder, to recover the original resolution. This method is known as resolution adaptation. While this method can prove effective, its success greatly depends on the characteristics of the input video. For example, if a given video scene does not contain a high amount of spatial detail, encoding at a lower resolution will result in the same image quality while reducing the required transmission bitrate. Most prior work on this topic has shown that these methods are most beneficial at lower bitrates. For this reason, the challenge remains in designing reliable techniques to determine the optimal resolution at which to encode a given input video based on its content.

Video compression standards have evolved throughout the years, mostly through a succession of incremental improvements to the hybrid video coding model, which has persisted throughout generations of codecs. However, recent advances in machine learning, more specifically, deep neural networks might result in breakthrough innovations. These methods have shown massive potential in almost all vision-based tasks from image and video recognition to early machine learning-based image compression algorithms. Progress in this field is assisted by improvements in the computational capacity of parallel-enabled hardware, such as GPUs and the availability of large databases of visual content. All the reasons above make this area a worthy avenue of research towards improving video coding further.

1.2 Objectives and contributions

This thesis explores alternative approaches for video compression based on intelligent resampling techniques including texture synthesis and spatial resolution adaptation. It is important to first understand the behaviour and performance of current video coding algorithms, including their limitations, in order to focus on ways to provide meaningful improvements. In addition, as previously mentioned, the evolution of video coding has been focused on small incremental improvements to the long-established framework and, thus, in this work, we attempt to show that significant improvements can be achieved by applying more unconventional approaches.

The main contributions of the work presented herein are as follows:

- Development of an open-source dataset of homogeneous texture videos to assist the research of new algorithms for the coding of textured content.
- Analysis of videos with different textures through the extraction of encoding statistics from the latest video compression standard, HEVC, and redefinition of texture classes for video coding.
- Exploration and characterization of texture analysis and synthesis algorithms for these specific types of textures.
- Proposal of a new framework for spatial resolution adaptation using low level visual features extracted from uncompressed video content, which is able to reliably detect at which quantization level to switch between different spatial resolutions before encoding.
- Proposal of a new video coding framework using deep learning techniques for compressed video super-resolution.
- In-depth study of these deep learning architectures and training methodologies for the improvement of the reconstructed video quality with a focus on achieving higher objective and subjective quality, as measured by PSNR and subjective visual comparisons, respectively.

1.3 Thesis organization

The present thesis is divided into a total of 7 chapters which include the introduction as well as the conclusion and future work. Each technical chapter contains a section dedicated to describing the current state-of-the-art approaches relevant to the specific topic discussed. The outline of the thesis is as follows:

In **Chapter 2**, a brief background to the field of video compression is presented. We describe the main coding tools used by recent video codecs and present experimental results comparing the performance of a few modern video codecs, such as HEVC and AV1.

In **Chapter 3**, texture analysis and synthesis-based video coding techniques are introduced and an extensive analysis of the performance of HEVC on different texture types is conducted. The chapter concludes with an alternative method of synthesising structured dynamic textures is presented.

In **Chapter 4**, we introduce the topic of spatial resolution adaptation, which consists of dynamically encoding downsampled versions of the input video frames and upsampling to the high resolution at the decoder. We propose a simple but reliable framework designed for HEVC intra coding. Finally, experimental results based on PSNR and VMAF are reported.

In **Chapter 5**, the previous framework is improved by extending it to all frame types, improving the resolution prediction based on low level visual features and proposing the use of a convolutional neural network to reconstruct the full resolution video frames at the decoder.

In **Chapter 6**, an exploration of different methods of improving the quality of the reconstructed video frames using more advanced deep learning techniques is conducted. In addition, we also train a network designed to achieve perceptually oriented results using adversarial techniques.

Finally, in **Chapter 7**, conclusions and suggestions for future work are presented.

Chapter 2

Background

This chapter provides a review of relevant topics that will be explored further during the remainder of this thesis. First, an overview of video compression is presented, starting from the most important concepts of digital video to the basics of video coding. A brief overview of the history of standardization efforts and description of various modern video codecs, including HEVC and AV1, is then discussed. Lastly, since deep learning techniques are employed in the last chapters, a brief overview of the basic ideas behind deep learning is also provided.

2.1 Video compression review

A video is a sequence of consecutive images acquired by projecting a real-world scene in a 2-D plane using a video capturing sensor or by creating a sequence of artificially generated images (animation). Each individual image, known as a frame or picture, is displayed with a certain frequency defined by the frame rate, generally expressed in frames per second (fps) or hertz (Hz). Frame rates can range from 24 fps to 300 fps, depending on the application, with 24 frames per second being the most common frame rate used in the film industry [15]. A digital image is defined by a matrix of elements called pixels, each representing the brightness and colour of a specific discrete location in the 2-D plane. Pixels usually store three distinct values, which varies with the colour representation used. Commonly used colour formats include RGB, YUV (or YCbCr) [16]. The size of each image, known as the spatial resolution, is usually expressed in the format $W \times H$ pixels, where W and H correspond to the width and height, respectively. Some common spatial resolutions include 640×480 (SD), 1280×720 (HD 720) and 1920×1080 (HD 1080). Nowadays, the demand for higher resolutions, which include 3840×2160 (UHD), 4096×2160 (4K) and 7680×4320 (8K) is increasing. Another important video parameter is bit depth, which defines the number of bits per colour component. The most common value is 8 bits per channel, which corresponds to a total of 24 bits for each pixel. However, higher values up to 24 bits per channel are currently becoming increasingly popular, especially with the introduction of High Dynamic Range (HDR) video capture.

Depending on the parameters previously described and the length of the video, massive amounts of data might be required to represent a raw digital video, which makes it unfeasible for storage and transmission over the Internet. Video compression algorithms have been developed to address this challenge. These aim at reducing the amount of bits required for the video representation to allow for practical utilization. Video compression algorithms work by exploiting several forms of redundancy including spatial, temporal and statistical. They can either operate in a lossless manner, without loss of information or by applying a lossy step, known as quantization, which means that the output video will be different from the original. The goal of these methods is to maximize the visual quality for a given bitrate.

Video compression systems are composed of two main parts, an *encoder* and a *decoder*. The former is responsible for generating the compressed stream of bits (bitstream) from the input raw video file. The ratio between the bitrate of the compressed bitstream and the raw video file is known as the compression ratio. As the reverse process, the *decoder* is responsible for receiving a compressed bitstream as input and generating a raw displayable video file. Given that the bitstreams generated by the encoder need to be interpreted by the decoder, which is usually located in another device, these two systems need to be exactly compatible. For instance, for a video streaming service, encoding is performed at the data servers, while decoding is executed at the receiving device, such might be a television, a personal computer or even a mobile phone.

2.1.1 Brief history of video compression standards

Due to the compatibility requirements between encoders and decoders and to promote mass adoption of video compression systems (also known as codecs), standardization efforts are a necessity. Nonetheless, video compression standards only define the syntax of the bitstream that is interpreted by the decoder. As long as the encoder can produce a conforming bitstream, different encoder-specific optimization algorithms can be applied.

Figure 2.1 presents the timeline of the main video standards proposed throughout the years. It can be seen that since the late 1980s, several video compression standards have emerged. The first established digital video coding standard was H.261 [17], rectified by ITU-T Video Coding Experts Group (VCEG) in 1988. H.261 was also the first block-based hybrid codec and its ideas form the basis of the majority of modern video codecs. In 1993, another standardization body, ISO/IEC Moving Pictures Experts Group (MPEG) proposed its own standard, MPEG 1 [18]. In the next year, another standard intended for higher video quality was proposed as a collaboration between ITU-T VCEG and the ISO/IEC MPEG, called H.262 or MPEG-2 [19]. In the mid-2000s, the same partnership between ITU-T and MPEG developed the most successful video codec, still widely used as of the time of writing, H.264/MPEG-4 Advanced Video Coding (AVC) [20].

Even after the success of AVC, video coding standardization bodies, ITU-T and MPEG, continued to develop efforts towards improved technologies and, in 2013, introduced its successor, H.265/High Efficiency Video Coding (HEVC). HEVC includes several new tools and was shown to achieve the

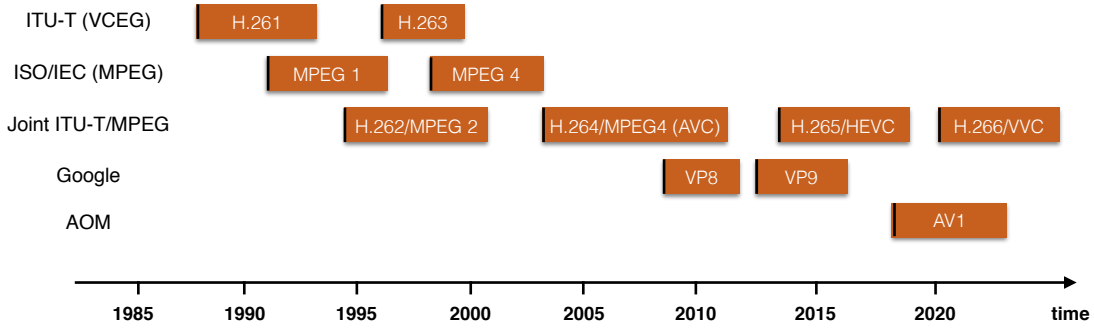


Figure. 2.1 Approximate timeline of video coding standardization efforts. The black bar represents the publication year. The standards coloured in orange have been widely adopted.

same perceptual quality using 50% of the bitrate compared to AVC [21]. In addition, in 2015, a new collaboration between ITU-T VCEG and ISO/IEC MPEG, the Joint Video Exploration Team and later in 2018, the Joint Video Experts Team (JVET) were founded with the goal to develop the new video coding standards. The successor to HEVC, entitled Versatile Video Coding (VVC) is planned to be finalized by 2020 [22].

Also in the late 2000s, other private companies began developing their own video coding formats, in an effort to move away from the royalty-based video codecs developed by ITU-T and MPEG. Google acquired On2 Technologies and open-sourced the video codec VP8 [23] in 2010, which was extended to VP9 [24] in 2012. At this point VP9 was mainly used by Google's YouTube platform as their main video codec. However, in 2015, with the same goal of pursuing royalty-free codecs, a number of major companies formed the Alliance for Open Media (AOM) [25]. Its founding members included Amazon, Cisco, Google, Intel and Netflix but since the formation, several other companies have joined. Their first video coding format was AOMedia Video 1 (AV1) [26], initially finalized in early 2018, but still under final developments as of the time of writing.

2.1.2 The hybrid video coding model

The main idea of video compression is to remove the existing spatial and temporal redundancy within and between video frames, respectively. Spatial redundancy comes from the correlation or similarity between adjacent pixels in a frame. For example, within a relatively homogeneous region, pixels values tend to be similar, thus, coding every pixel independently would be suboptimal. For this reason, video compression algorithms typically jointly encode a collection of adjacent pixels, known as blocks. Similarly, since a video is captured using a minimum frame-rate of 24 frames per second, successive frames generally contain very similar information. Therefore, an encoded frame can be used as a reference for subsequent frames after the application of a processed called motion compensation, which aims at estimating one or multiple blocks in the reference frame that matches the current block

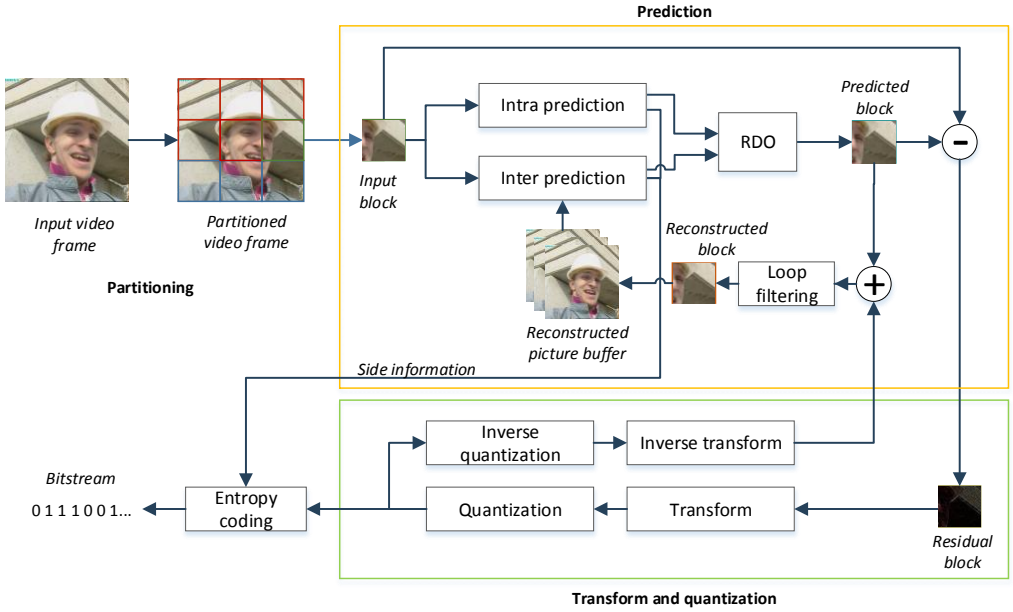


Figure. 2.2 Diagram representation of a basic hybrid video codec architecture.

to be encoded. By applying this technique, it is possible to send only a residual signal between the predicted and the original block, which generally requires fewer bits to encode.

As previously mentioned, the most successful video coding model, still employed in current video standards, is known as the hybrid video codec [17]. Figure 2.2 shows the basic structure of this model. The term hybrid comes from the fact that the encoder also contains a full decoder. The basic steps of this system are: partitioning, prediction, transform and quantization, and entropy coding. The process works as follows: each raw frame is recursively split into a number of partitions to be used for the next step, prediction. There are two types of prediction, intra and inter prediction. In the first, the information in a block is predicted from previously encoded spatially adjacent blocks. On the other hand, the latter applies motion compensation to achieve a prediction from previously decoded frames, referred to as reference frames. For the first frame in a video, only intra prediction is applied, since no temporal information is available from other frames. If this is not the case, each block can be either intra or inter predicted and the decision is based on a process called Rate-Distortion Optimization (RDO). The original block is then subtracted from the prediction to produce the residual, which is transformed by a linear spatial transform, quantized using a defined Quantization Parameter (QP), entropy coded and, finally, written to the bitstream. The reconstructed blocks, which are used for prediction, are obtained by applying inverse quantization and inverse transform. For this reason, the prediction/reconstruction process at the encoder is called the encoder loop. In addition to encoding the residual signal, the encoder also adds relevant side information required for decoding, such as the partitioning structure as well as motion vectors used for inter prediction. At the decoder side, the bitstream is interpreted and the process of obtaining the reconstructed frames is applied, which should

exactly match the frames obtained by the encoder loop. These then represent the output video signal to be displayed. For further reading on the basic concepts of video coding, please refer to [16].

2.1.3 Evaluation of coding performance

Video codecs are generally evaluated based on two key components: rate-distortion performance and complexity. The first measures the compression efficiency which, in video coding is a relationship between bitrate and visual quality. For the majority of encoders, this trade-off is controlled by the QP input value. Bitrate is typically measured in kilobits per second (Kbps) and is calculated using the following equation:

$$\text{bitrate} = \frac{\text{number of bits}}{\text{number of frames}} \cdot \text{frame rate} \tag{2.1}$$

Moreover, distortion or quality (the inverse of distortion) is evaluated using a quality metric that, in general, estimates how well the reconstructed video resembles the original video.

The combination of bitrate and quality at a given QP value is referred to as a rate-point or RD point. In order to fully evaluate the encoding performance, different rate-points at multiple bitrates and qualities are required and form a Rate-Distortion curve (RD curve). An example of two RD curves for two different codecs is illustrated in Figure 2.3. In this example, quality is evaluated using the Peak-Signal-to-Noise Ratio (PSNR), which will be described later. It can be seen that Codec 1 contains a rate-point at bitrate of 700 Kbps and a PSNR value of 40 dB, while Codec 2 also has a rate-point with the same bitrate but at a lower quality, 38 dB. Similarly, for the same quality of 40 dB, the video produced by Codec 1 requires 700 Kbps, while Codec 2 requires 900 Kbps. Therefore, by performing this analysis, it can be concluded that Codec 1 produces superior performance compared to Codec 2 in a rate-distortion sense for those rate-points. However, since this analysis needs to be performed for a range of desired bitrates (or qualities), a method named Bjntegaard-delta (BD) model [27] is employed. The idea is to calculate the average bitrate and quality differences between the RD curves. This model reports two values: the BD-PSNR and the BD-rate. The first measures the average PSNR difference for the same range of bitrates, whereas the latter estimates the average bitrate difference (in percentage) for the same quality values.

In order to estimate these BD measurements, the set of rate-points composed of bitrates (R) and quality (Q) pairs is fitted using a third-degree polynomial function in the bitrate logarithmic space, $r = \log R$. At least 4 rate-points for each codec are required for this type of fitting. To compute the BD-rate, the bitrate as a function of the quality, $\hat{r}(Q)$ is fitted as follows:

$$\hat{r}(Q) = a\,Q^3 + b\,Q^2 + c\,Q + d \tag{2.2}$$

Where a, b, c and d are the fitting parameters. The BD-rate is then estimated by the difference between the integrals of the fitted curves divided by the quality interval:

$$\Delta R = \mathbb{E}[Q_{max} - Q_{min}] \tag{2.3}$$

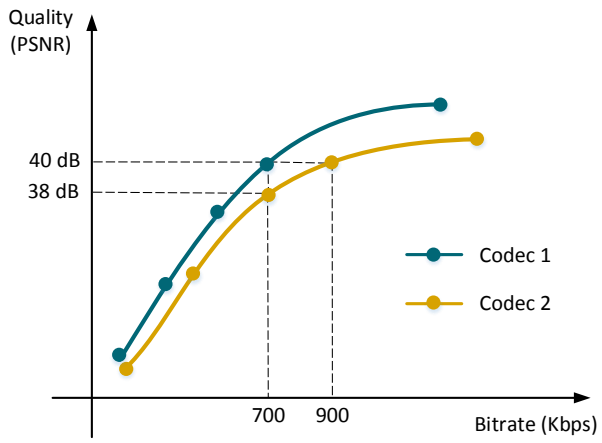


Figure. 2.3 Representation of a Rate-Quality (or RD-curve) of two different codecs for the same video sequence. Values for illustration purposes only.

Here, Q_{min} and Q_{max} correspond to the common maximum and minimum quality achieved by the rate-points of each codec. When computing the BD-quality, a similar process is used but, instead of fitting the rate as a function of the quality, the quality is fitted as a function of the rate and the average PSNR difference is estimated. If subjective quality metrics are used instead of objective quality metrics, BD measurements can be computed in a similar fashion using the methodology outlined in [28].

The complexity of the video codec is generally measured by the encoding and decoding times and memory consumption. This is based on the quantity and complexity of coding modes and tools but is also highly dependent on the encoder and decoder implementations. Standardization efforts are usually accompanied by a reference encoder and decoder, for example, HM is the HEVC reference software, while libaom is the reference AV1 software. Then, after the standard is finalized, production encoders and decoders are developed, which are typically orders of magnitude faster than the respective reference software but sometimes produce lower compression efficiency than the reference software.

2.1.3.1 Measuring video quality

The evaluation of video quality, as perceived by human observers, is an active research field in the video processing community. Having a reliable way to measure video quality that considers compression and other relevant distortions is extremely desirable since it can directly influence the behaviour and development of video encoding standards.

There are two main ways of evaluating the quality of a distorted video: objective metrics and subjective scores. Objective metrics are algorithms that attempt to measure the amount of distortion introduced in a video and can either be perceptually inspired or not. Those can be further categorized into full-reference, reduced-reference and no-reference, depending on the amount of

known information about the original video. In addition, some video quality metrics are actually image metrics computed on each individual frame and pooled or averaged to achieve a final score. On the other hand, subjective scores are obtained directly from experiments with human viewers observing the distorted content. Although subjective tests, if performed correctly, are generally considered to produce more reliable results, they require much more effort, time and resources (including subjects), compared to the objective metrics. For this reason, objective metrics are generally the main tool employed in practical situations, for example, during the development of new coding standards. However, results from subjective tests are extremely useful since they are used as ground truth for the development of perceptually-inspired objective metrics.

The simplest and most popular objective way to measure the quality of a distorted image or video is through the calculation of the Peak-Signal-to-Noise Ratio (PSNR). This full-reference metric is calculated using the Mean-Squared Error (MSE) between the original video frames, $\mathrm{I}_{ref}(x, y)$, and the distorted frames, $\mathrm{I}_{dist}(x, y)$. This error is computed by calculating the average squared pixel value differences at the same spatial positions between the frames, as shown by the following equation:

$$\mathrm{MSE} = \frac{1}{WH} \sum_{x=1}^{H} \sum_{y=1}^{W} \Big(\mathrm{I}_{ref}(x, y) - \mathrm{I}_{dist}(x, y) \Big)^2 \tag{2.4}$$

Where H and W are the height and width of each frame. PSNR is then computed by taking the logarithm of the inverse MSE and is reported in decibels (dB):

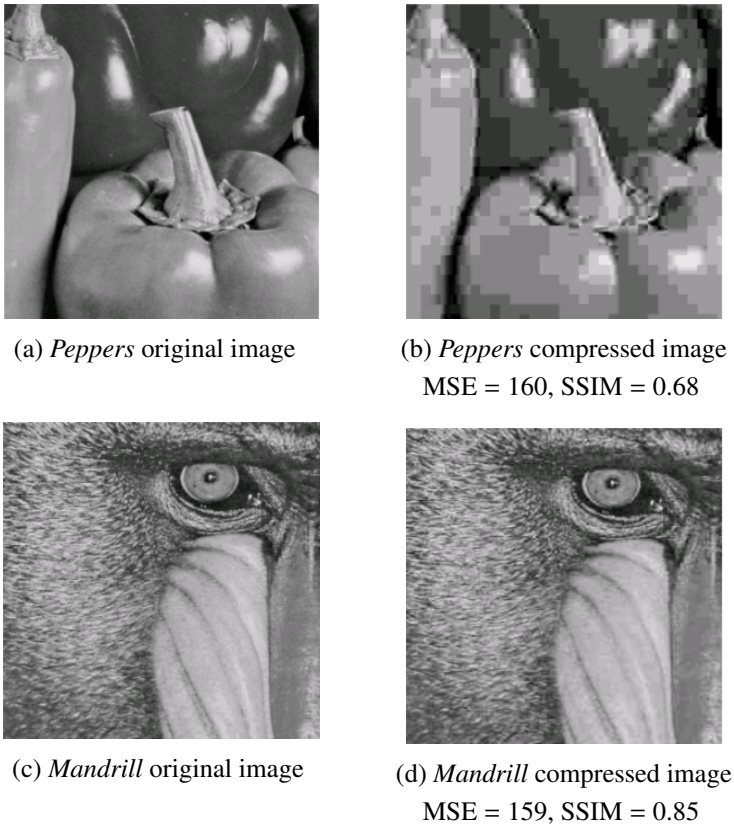
$$\mathrm{PSNR} = 10 \log_{10} \left(\frac{\mathrm{MAX}^2}{\mathrm{MSE}} \right) \tag{2.5}$$

Here, MAX is the highest allowed pixel value, which corresponds to 255 for 8 bit inputs and 1023 for 10 bit inputs.

MSE and PSNR values are calculated for individual frames only. Therefore, in order to obtain the final video metric, two possibilities exist, which are known as pooling strategies: to average the MSE values per frame and compute the final video PSNR or to average the PSNR value of individual frames. In this thesis, we employ the first approach when computing the PSNR of a test video.

Due to its low complexity, PSNR is still widely adopted in the video compression community, sometimes as the only objective metric used for the evaluation of video codecs. However, although simple and mathematically convenient in the context of optimization, it is well known that the correlation between PSNR and subjective scores is low [29]. This means, for example, that two videos with same PSNR value might have very different perceived qualities. For this reason, in order to overcome the limitations of PSNR, perceptual quality metrics, which aim at narrowing the gap between objective and subjective scores, have been developed. Among the multiple methods proposed, SSIM [30], VQM [31] and VMAF [32] are among the most popular and will be briefly described below.

SSIM is a popular image and video full-reference quality metric that quantifies perceptual degradation through a combination of three key image components: luminance (l), contrast (c) and structure (s). These are computed on blocks of $N \times N$ pixels of the reference image, x, and the distorted image, y. The final SSIM score is then given by $\text{SSIM}(x, y) = l(x, y)^{\alpha} \cdot c(x, y)^{\beta} \cdot s(x, y)^{\gamma}$, with the variables α, β and γ responsible for controlling the relative weight of each component, although usually set to 1. Spatial pooling of the SSIM index within a given image or frame is typically performed using the average of the SSIM index for each pixel, which can be calculated by applying a moving square or Gaussian window of sizes $N = 8$ or 11 pixels. In addition, when applied on videos, SSIM is generally computed on the luma components only and averaged for all frames to obtain a final score for the entire video. Experiments show that SSIM achieves higher correlation with subjective scores compared to PSNR for both images [30] and videos [1]. Figure 2.4 shows an example of two images compressed with JPEG, *Peppers* and *Mandrill*. Both have similar MSE values, however, the perceived quality is very different. *Peppers* contains heavy blocking artifacts while the artifacts are less visible in *Mandrill* due to the highly textured content. SSIM, on the other hand, gives more perceptually aligned scores, 0.68 to *Peppers* and 0.85 to *Mandrill*.



(a) *Peppers* original image

(b) *Peppers* compressed image
MSE = 160, SSIM = 0.68

(c) *Mandrill* original image

(d) *Mandrill* compressed image
MSE = 159, SSIM = 0.85

Figure. 2.4 Examples of MSE and SSIM values of two cropped (256×256) JPEG compressed images, *Peppers* and *Mandrill*. Although their MSE values are roughly the same, the visual quality of *Mandrill* is much higher, which is reflected by a higher SSIM value. Results extracted from [1].

Although SSIM is generally considered a more reliable estimate for image quality, it is not optimized for videos, since no temporal information between frames is considered. In order to overcome this limitation, techniques developed specifically to measure video quality have been proposed. One example is the Video Quality Metric (VQM) [31], standardized by the American National Standards Institute (ANSI). This perceptual video metric measures and combines the effects of several types of visual impairments including blurring, edge artifacts, motion jerkiness and distortions in the colour. It specifies several different modes, with the general mode employing a total of 7 features, combining spatial, temporal and chroma information. VQM is reported to provide superior performance compared to PSNR and SSIM in both the VQEG [33] and LIVE [34] video quality databases. However, it comes with significantly increased complexity [35].

More recently, a video quality metric based on machine learning, the Video Multimethod Assessment Fusion (VMAF) [32], was proposed. VMAF combines the benefits of two image quality metrics, Visual Information Fidelity (VIF) [36] and Detail Loss Metric (DLM) [37], with the addition of a simple temporal information feature, the frame difference. VIF is a popular image quality metric based on the computation of information fidelity loss and is calculated at multiple spatial scales (by applying scaling). In VMAF, each of the scales computed in VIF are used as independent features. On the other hand, DLM is a metric that measures the loss of useful detail affecting the visibility of the contents of an image. This feature is based on a Wavelet-domain decomposition [16]. The final feature is the absolute temporal difference pixel-wise between adjacent frames. The idea is that the amount of temporal information influences the ability of the HVS to perceive visual distortions. These features are combined using a machine-learning based algorithm, more specifically a Support Vector Machine Regressor (SVMr), which estimates the optimal weights for each of the elemental features using subjective scores collected using a database of distorted videos as the reference. Experiments reported in [32] show that VMAF is able to provide superior performance compared to PSNR, SSIM and VQM for the VQEG database and competitive performance for the LIVE video database. Since the first model was proposed, further improvements related to the utilization of the frame difference information were presented in [38].

2.1.4 Modern video codecs

The following sections will introduce different algorithms and techniques employed in modern video coding standards, more specifically, in HEVC, VP9, AV1 and ITU-T and MPEG's efforts beyond HEVC. A larger focus will be given to HEVC, since it is the anchor codec used throughout this thesis.

2.1.4.1 H.265/HEVC

High Efficiency Video coding (H.265/HEVC) is a video coding standard proposed by a joint collaboration between ITU-T and MPEG standardization bodies, JCT-VC, and is the successor of AVC. Due to the addition and modification of a number of coding tools, its compression efficiency is significantly higher than AVC. HEVC was designed with a focus on high resolution content and

allows inputs of up to 8K resolution, 8192×4320. It also contains parallel processing tools for efficient hardware decoder implementations. A brief description of the coding tools in HEVC organized by categories is presented below. When relevant, differences with respect to the coding tools in AVC are also mentioned. Further details can be found in [21].

- **Coding structure:** Similar to AVC, HEVC allows for a flexible coding and prediction structure. The highest encoding level is a slice, which can correspond to an entire frame or a portion of a frame. Since the most common is for a slice to correspond to a full frame, we will refer to slice as pictures. Three types of pictures are defined in HEVC, key or Intra (I), Predictive (P) and Bi-predictive (B) pictures. Intra pictures only allow intra prediction, which means that only spatial redundancies within the same picture are exploited. Therefore, these types of pictures can be decoded independently of any other previously decoded pictures and, therefore, provide random access points and stop error propagation. On the other hand, P and B pictures allow prediction from other previously decoded pictures, known as inter prediction. While B pictures permit prediction from both previous and future pictures (in the display order), P frames can only contain unidirectional prediction.

  The Group of Pictures (GOP) defines the prediction structure used by the encoder. The distance between key pictures (I or P) is generally referred by the GOP size and the distance between I pictures is known as the intra period. For instance, IBBBPBBBI defines a GOP of size 4 and an intra period of 8 pictures. In addition, the access configuration defines the type of GOP used for different applications. All intra (AI) is a configuration where only I pictures are allowed and is generally only used for high quality post-production applications. For Random Access (RA), the decoding and display order of some pictures is different and all slice types are allowed. This configuration provides improved compression efficiency compared to the others, especially when employing hierarchical B pictures, in which prediction using past and future references is exploited by utilization of a hierarchical GOP structure. An example is illustrated in Figure 2.5. Compared to a non-hierarchical prediction structure, the use of hierarchical B pictures, especially with larger GOP sizes, was shown to significantly increase the coding performance [39]. Finally, low delay configuration allows for I and P slices only and the decoding and display order is the same, with no reordering needed. This type is mainly used for video conferencing and other applications where reduced delay times are critical.

- **Partitioning:** In AVC, the input video frames were initially split into fixed blocks of size 16×16 luma samples, known as macroblocks. In HEVC, the macroblock was replaced by the Coding Tree Unit (CTU), which allows for more flexible partitioning. The maximum CTU size is also larger, up to 64×64 samples, which was shown to improve the compression efficiency, especially for high resolution videos. The CTUs are then partitioned into smaller blocks, Coding Units (CU), using a quad-tree structure. The minimum CU size is 8×8 pixels, which corresponds to 4 splits considering a CTU size of 64 square samples.

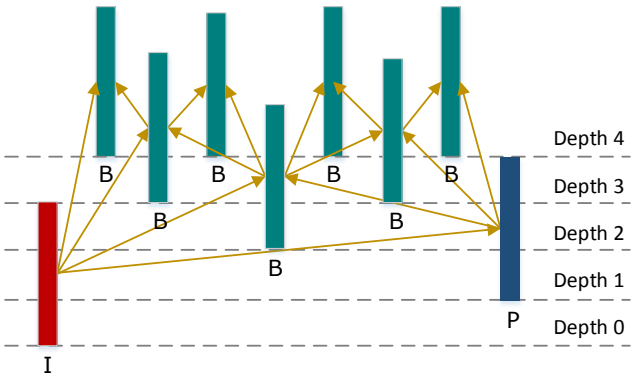


Figure. 2.5 Representation of a hierarchical prediction structure with GOP size of 8.

- **Intra picture prediction:** Intra prediction attempts to predict the samples of the block to encode using previously decoded samples in the edges of the current block. Three types of intra prediction strategies are defined in HEVC: DC, planar and directional prediction. In DC mode, all samples in the block are predicted as a constant average value of the reference samples. Planar mode consists of a surface fitting from the reference samples. Directional prediction interpolate and projects the reference samples in a specific direction to obtain the predicted samples. For example, using the vertical direction, the samples above the current block are copied to the corresponding positions to form the predicted block. Compared to AVC, HEVC increased the number of intra prediction modes compared to AVC, from 8 to 33 directional, which accounts for significant gains at the expense of the added complexity in testing more modes.

- **Inter picture prediction:** Inter prediction is performed by achieving an estimation of the temporal motion of each block to encode using previously decoded frames, using both forward and backward prediction (past or future frames). The motion estimation process derives a motion vector and is achieved using block matching algorithms at an interpolated reference frame, in order to achieve more accurate estimation. HEVC uses quarter-sample precision for Motion Vectors (MVs). In addition, HEVC uses a merge mode, in which the motion information including motion vectors and reference indexes of current coding block are copied from its spatially or temporally neighbouring blocks that have already been decoded. This reduces the amount of signalling information required for transmission. In addition, if no residual is coded after prediction, it is referred to as skip mode (a skip flag is enabled).

- **Transform and quantization:** After a residual signal has been obtained from either the intra or inter prediction processes, it is subject to a separable 2-D linear transform. The transform is essential to the encoding process since it provides energy compaction, in most cases, requiring fewer bits to encode than the original residual signal and also providing a separation between frequency regions, which allows better lossy perceptual coding. For inter predicted residuals, an approximation of the Discrete Cosine Transform (DCT) is used, whereas, for 4×4 intra

predicted blocks, there is also a possibility of applying the Discrete Sine Transform (DST). There is also an option to skip the transform step and encode the samples directly. Transform Units, TU, can vary in size, ranging from 4×4 to 32×32 pixels. After the application of the transform, the coefficients are subject to quantization using pre-defined quantization matrices. The quantization step is controlled by the Quantization Parameter (QP), which ranges from 0 to 51. An increase of the QP by 6 values is equivalent to double the quantization step. Unlike all processes until this point, which were lossless (no information was lost), quantization cannot be exactly inverted and, therefore, provides lossy compression. In order to increase visual quality, high frequency coefficients are more heavy quantized compared to low frequency coefficients. This is because a loss in high frequencies has a lower impact to the overall visual quality [40].

- **Entropy coding:** The quantized coefficients need to be written into the bitstream. In order to reduce the information redundancy that exists, a technique called entropy coding is applied, which is common to any data compression task. Entropy coding is a lossless process and aims at representing a stream of information with the least amount of bits possible while guaranteeing that it is exactly recoverable at the decoder. Similarly to AVC, HEVC uses an advanced technique known as Context Adaptive Binary Arithmetic Coding (CABAC), which greatly improves the coding performance compared to previous algorithms. CABAC is an arithmetic coding technique adapted to the specific use case of video coding. It encodes binary symbols to keep the complexity low and due to the local correlation that exists between symbols in video coding, it uses a local context to better model the probabilities. HEVC uses an improved CABAC model, which increase its throughput speed and provides higher compression performance.

In addition to the main standard, several extensions to address a more specific range of applications were also developed. These extensions address the coding of videos with higher bit depths, support for 4:2:2 and 4:4:4 colour sampling, lossless encoding, screen content coding (graphics and other types of non-natural video) and 3-D video. Another type of extension is scalable video coding which corresponds to the embedding multiple encoded videos at different bitrates into the same bitstream.

2.1.4.2 VP9

VP9 is a video codec developed by Google and released in December 2012 as part of the WebM project [41]. Its main advantage compared to AVC and HEVC is that it is open-source and royalty-free, which means that video content producers, distributors and chip-makers are not required to pay royalties for deploying it. Upon its initial release, VP9 was adopted as the main encoder for Youtube, but has since been supported by other video streaming platforms such as Netflix and is available in a number of browsers including Chrome, Mozilla, Firefox and Opera. The following provides a brief overview of the technical aspects of VP9, including the main differences compared to HEVC.

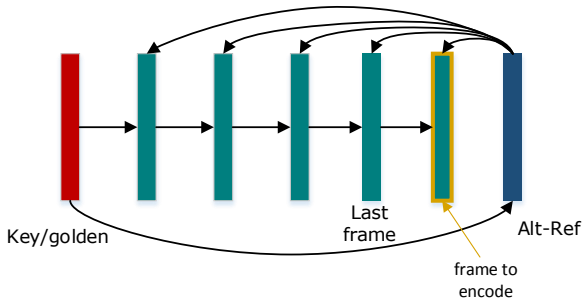


Figure. 2.6 Representation of the coding structure of VP9. The highlighted frame is the next to be encoded. Its references are the key frame, the last encoded frame and the Alt-Ref frame.

Similarly to HEVC, in order to boost the compression efficiency for higher spatial resolutions (above 1920×1080), VP9 has introduced larger prediction blocks up to 64×64, which are called Super-Blocks (SB). These blocks are recursively split with the smallest allowed block size of 4×4. At each level, three possible splits are supported: horizontal, vertical or both (into 4 sub-blocks), with the horizontal and vertical splits being terminal, in a sense that further partitioning is not allowed.

VP9 supports 10 intra prediction modes including DC, TM (True-Motion prediction), and 8 directional predictions in block sizes that vary from 32×32 to 4×4 pixels. The total number of modes is significantly lower than HEVC, which allows 35 different intra modes. Similarly to HEVC, the DC mode prediction is obtained by averaging the adjacent reference pixel values. On the other hand, TM mode is a simple alternative to the planar mode in HEVC.

In terms of inter prediction, motion compensation is performed using one or two of up to three past or future reference frames in the picture buffer. If two frames are used to predict a single block, it is defined as compound prediction. Unlike HEVC, VP9 allows for multiple options for sub-pixel motion interpolation filters: regular, sharp or smooth, which produce different results in the appearance of edges.

In VP9, the coding structure is significantly different compared to previous encoders. Each frame can reference three types of pictures: an intra-only, known as key or golden pictures, the last or previous frame, and an Alternative Reference frame (Alt-Ref), which is a frame encoded purely to be used as a reference and is never displayed. A representation of the coding structure of VP9 is presented in Figure 2.6.

2.1.4.3 AV1

AV1 was developed by a collective effort from the members of the Alliance for Open Media (AOM) [25] and combines coding tools from VP9, Daala [42] and Thor [43]. The goal of AOM was to develop an alternative royalty-free video format, with competitive performance against HEVC. In April 2018, the bitstream specification was finalized, but optimizations and fixes are still ongoing, as of the time of writing.

AV1 has introduced several new tools compared to VP9 with reported bitrate savings around 30% compared for the same video quality [44]. The following will present a few of the key difference between both codecs, which contribute to its overall coding performance. A more comprehensive overview of its coding tools can be found in [44].

In terms of the block partitioning, AV1 provides a more flexible structure compared to VP9 in which the highest block level, the Super-Block (SB), was increased to a size of 128×128 pixels. This block can be recursively partitioned using a 10-way structure, including asymmetric partitioning, down to a minimum size of 4×4. In addition, a wedge mode that allows for non-rectangular shaped predictions is introduced. The wedges are decided based on a codebook and allow for different intra or inter predictions on each part, adapting better to object boundaries.

AV1 has introduced several new intra prediction techniques. First, it enhanced the directional prediction by the use of an extended number of intra mode directions, 56 compared to 10 of VP9, allowing for even finer prediction granularity. It also includes a Paeth and a Smooth prediction. The Paeth predictor replaces the TM predictor in VP9 and uses three pixels from the neighbouring reference pixels, from the top, left and top-left. There are three options for the smooth predictor: vertical, horizontal or combined, which use a quadratic interpolation technique in the vertical, horizontal or the average directions, respectively. Other intra coding tools in AV1 include the recursive-filtering-based intra, the colour palette prediction, chroma from luma and intra block copy.

Inter prediction is also enhanced in AV1. The number of reference pictures for each frame is increased from 3 in VP9 to 7 in AV1 with the introduction of a number of new reference frame types to allow for a more flexible hierarchical prediction structure, similar to what is employed in HEVC. It also introduced a number of additional inter coding tools including Overlapped Block Motion Compensation (OBMC) for the refinement the prediction at block boundaries, a number of global motion compensation techniques with perspective and affine motion models and advanced compound predictions.

AV1 also contains better loop restoration filters, more specifically, the Constrained Directional Enhancement Filter (CDEF) which is a edge-preserving filter applied after the deblocking filter that estimates the edge directions and applies a non-separable non-linear filtering technique. After this filter, other restoration techniques based on Wiener and self-guided filters can also be applied to enhance the quality of the reconstructed block further.

In addition to all these new coding tools, AV1 also includes some optional unconventional techniques such as a frame super-resolution tool which downsamples the video frames in-loop at the encoder side, in the horizontal direction only, and reconstructs the full resolution using a linear filter followed by a loop restoration filter. Another tool is film grain synthesis, which estimates the film grain at the encoder using a parametric model, removes it from the encoded video and then synthesises the missing film grain at the decoder.

2.1.4.4 JEM

Even though HEVC has achieved impressive performance compared to its predecessors, the need for higher compression efficiencies is still required, especially considering the demand for higher spatial resolutions, qualities, frame rates and bit depths. For this reason, efforts are currently underway for a new coding standard from the ITU-T and MPEG standardization bodies. The aim is to achieve from significantly higher compression efficiency compared to what is realized by HEVC.

The first step in this standardization process was to develop an experimental test model, the JVET Exploration Model (JEM), containing a set of newly proposed video coding tools. This test model has achieved significant performance improvements compared to HEVC test model with an increase in encoder and decoder complexity [45]. Next, we present a few novel coding tools included in JEM. The document [45] provides a detailed overview of the main coding tools included.

In terms of the block partitioning, JEM introduces a more flexible partitioning structure called Quadtree plus Binary Tree (QTBT) with larger maximum CTU sizes of up to 128×128. This approach allows two types of splitting: a quadtree (QT) that splits the blocks into 4 equal parts, and a binary tree (BT) with either horizontal or vertical splits. Therefore, both square and rectangular CU blocks are supported. In addition, unlike in VP9, horizontal or vertical splits are not final. An example of the partitioning of a CTU using QTBT can be found in Figure 2.7. The value of 1 denotes a horizontal split while a vertical split is represented by a 0.

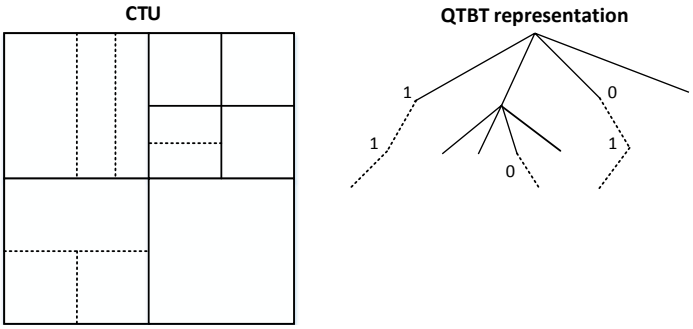


Figure. 2.7 Example of a partitioning of a CTU using the QTBT structure.

In JEM, the number of intra modes is increased from 33 directional modes to 65, with DC and planar modes the same as in HEVC. In order to limit the complexity, the search for the optimal intra coding mode uses a list of the 6 Most Probable Modes (MPM). These are generally chosen from neighbouring blocks. Instead of using a two-tap linear interpolation filter for the prediction of the intra block, a four-tap filter is applied in JEM. Correlations between luma and chroma components are also exploited by the use of a Cross-Component Linear Model (CCLM), which attempts to predict chroma intra modes based on the optimal luma modes.

Several inter prediction tools have also been included in JEM. First, the motion vector precision is increased to 1/16 pel (pixel distance) for the luma component compared to 1/4 employed in HEVC. JEM allows to split a current CU into smaller sub-CU and deriving more accurate motion information

for each using two proposed methods: Advanced Temporal Motion Vector Prediction (ATMVP) and Spatial-Temporal Motion Vector Prediction (STMVP). To handle non-translational motion, JEM adds a simplified affine motion prediction mode. Illumination changes between the reference and current CU blocks are minimized by a method called Local Illumination Compensation (LIC), which adjusts the illumination using a linear model, whose parameters are estimated using a least-squares approach.

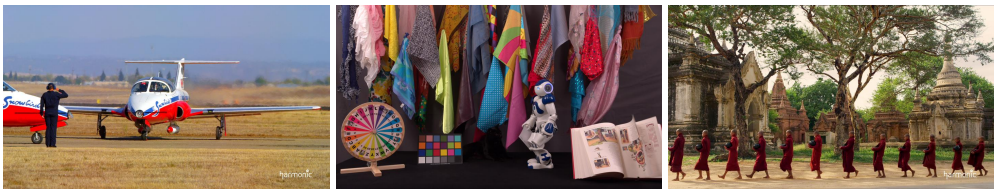
JEM introduces the concept of Adaptive Multiple Transform (AMT), which consists of applying several types of transforms from the DCT and DST families in addition to the conventional DCT-II and DST-VII used in HEVC. Additionally, a Mode-Dependent Non-Separable Secondary Transforms (MDNSST) is proposed. This transform is applied to the coefficients obtained from the primary transform and aims at further decorrelating the residual signal. Given that complexity of non-separable transforms are much higher than separable transforms (such as the primary transforms), their application is restricted to the low frequency coefficients in either 4×4 or 8×8 block regions.

In addition to the existing deblocking and Sample Adaptive Offset (SAO) filters in HEVC, two new in-loop filters are proposed in JEM: a bilateral and an Adaptive Loop Filter (ALF). The order of application is bilateral - SAO - ALF - deblocking. The bilateral filter is a non-linear filter that smooths and reduces the noise while preserving edges. It applies a weighted average of the values of neighbouring pixels. On the other hand, the ALF is designed to minimize the MSE between the reconstructed and original samples using a Wiener-based adaptive filter. For the luma component, one filter is chosen among 25 options and is based on the direction and activity of the local gradients. The idea of this filter is to align the directionality of the decoded block with the original block.

2.1.4.5 VVC

Following the high savings achieved by JEM, in October 2017, JVET announced a Call for Proposals (CfP) for the next video coding standard [22]. In 2018, this future coding standard was named Versatile Video Coding (VVC) and is expected to be finalized in 2020. The committee is targeting bitrate savings compared to HEVC in the range of 30-50% range for the same visual quality with an expected increase in encoding complexity. VVC is being designed as a flexible coding standard, applicable for multiple types of video content including but not limited to SDR and HDR, computer generated content, 360-degree videos and light-field data [46]. Although the final set of tools has not been finalized yet, it is expected that the final standard will include tools from HEVC, JEM and also newly proposed tools from the last CfP.

As of the time of writing, a few tools that have been included in the latest VVC test model, VTM 3.0 [46], that are likely be part of the final standard include: larger CTU sizes up to 128 × 128, a multi-type tree partitioning structure, which is a quad-binary-ternary tree allowing more flexible splits, the increased number of directional intra mode to 65, wide-angle intra prediction, affine motion compensation, spatio-temporal block motion copy, adaptive loop filter (ALF) and Multiple Transform Selection (MTS). Additionally, multiple neural network-based video coding tools have been proposed during the CfP, including ViSTRA [47]. Although promising results were shown and further study of



(a) AirAcrobatic - Harmonic (b) CatRobot - JVET (c) Myanmar - Harmonic

(d) CalmingWater - BVI-TEX (e) ToddlerFountain - Netflix (f) LampLeaves - BVI-TEX

(g) DaylightRoad - JVET (h) RedRock - Harmonic (i) RollerCoaster - Netflix

Figure. 2.8 Test sequences used for the codec comparison experiments.

these tools is ongoing, as of the time of writing, due to the higher complexity associated with these methods, it is likely that they will not be included in VVC.

2.1.5 Performance comparison between HEVC, JEM, VP9 and AV1

Due to the high number of codecs proposed in recent years, especially with the introduction of royalty-free alternatives to the ITU-T/MPEG standards, several works reporting coding performance comparisons between HEVC, VP9 and AV1 have been recently published. However, their conclusions are often contradictory, mainly due to differences in parameters or reference software versions. Therefore, we present an independent experiment comparing relatively recent versions of the reference software for HEVC, VP9 and AV1 [1]. In addition, we also include JEM, as an indicator of what might be the performance of the future codec, VVC. In these tests, we focus on coding performance only and ignore the encoding complexity due to the fact that for some of the codecs, especially AV1, the figures are changing rapidly with frequent optimization improvements.

For these experiments, 9 sequences from a number of video databases, including Harmonic Inc [48], Netflix Chimera [49], BVI texture database [13] and the JVET CTC [50], were considered. All sequences are HD resolution (1920×1080 pixels), 10 bits per channel and contain 4:2:0 chroma subsampling. The frame rate of all sequences is 60 fps except for CatRobot, which is 50 fps. A sample frame of each sequence and their respective sources can be found in Figure 2.8.

[1]This work has been done in collaboration with Dr. Fan Zhang, Dr. Angeliki Katsenou and Prof. David Bull at the University of Bristol.

Table 2.1 Specification of the video codecs used and their specific encoding parameters. Trivial encoding parameters such as width, height and frame-rates are not shown.

Codec	Software version	Specific encoding parameters
HEVC	HM 16.18	-c encoder_randomaccess_main10.cfg –InputBitDepth=10 –IntraPeriod=64
JEM	JEM 7.0	-c encoder_randomaccess_main10.cfg –InputBitDepth=10 –IntraPeriod=64
VP9	WebM Project VP9 Encoder - libvpx - commit from May 31, 2018	–passes=2 –i420 –profile=2 –cpu-used=1 –input-bit-depth=10 –kf-max-dist=64 –kf-min-dist=64 –arnr-maxframes=7 –arnr-strength=5 –lag-in-frames=16 –aq-mode=0 –bias-pct=100 –minsection-pct=1 –maxsection-pct=10000 –end-usage=q –min-q=0 –max-q=63 –auto-alt-ref=1 –max-gf-interval=14 –min-gf-interval=4 –frame-parallel=0 –threads=1 –tile-columns=0 –ivf
AV1	AOMedia Project AV1 Encoder - libaom - commit from May 30, 2018	–passes=2 –i420 –profile=0 –cpu-used=1 –input-bit-depth=10 –kf-max-dist=64 –kf-min-dist=64 –arnr-maxframes=7 –arnr-strength=5 –lag-in-frames=16 –aq-mode=0 –bias-pct=100 –minsection-pct=1 –maxsection-pct=10000 –end-usage=q –min-q=0 –max-q=63 –auto-alt-ref=1 –max-gf-interval=16 –min-gf-interval=4 –frame-parallel=0 –threads=1 –tile-columns=0 –ivf

We compare four modern video codecs using their reference implementations, HEVC (HM 16.18), JEM 7.0, VP9 (libvpx) and AV1 (libaom). The versions of VP9 and AV1 source code used are from May 31 and May 30 2018, respectively. This is after the bitstream freeze of AV1 had been completed (which means that no more changes to the bitstream structure are allowed). More information about the parameter settings of each codec can be found in Table 2.1. The parameters were chosen in order to provide test conditions, as most as possible aligned with the JVET Common Test Conditions (CTC) [50]. The intra period was set to 64 frames and the GOP size to 16 pictures. Each sequence was encoded 4 QP values. For HM and JEM, the QP range varied from 20 to 37, while for VP9 and AV1, the CQ value (the parameter that controls the quantization level) was set to values from 40 to 63.

The objective quality metrics, PSNR and VMAF (described in Section 2.1.3) were calculated for all sequences and rate-points and average savings between video codecs tested is presented in Table 2.2. The left table shows BD-rate savings based on PSNR, while the right used VMAF scores. It can be seen from the PSNR results that JEM outperforms all other codecs, with over 25% BD-rate savings compared to HEVC and more than 20% to AV1. Compared to VP9, AV1 improves coding efficiency by almost 30%. It is also slightly better than HEVC, with savings of 3.8%. When considering VMAF, the numbers are slightly different, for example, showing AV1 savings over HEVC of 5%.

The highlights are that although not all tools in JEM will be part of the future VVC standard, its coding performance compared to HM is very promising. In addition, even though the royalty-free codec, AV1 cannot achieve the same efficiency shown by JEM, its improvements in relation to VP9 are impressive and might be a good starting point for the future of open codecs.

Table 2.2 Average BD-rate savings (%) between the test codecs (vertical axis) and the anchor codecs (horizontal axis). *Left:* BD-rate based on PSNR; *Right:* BD-rate based on VMAF.

Codecs	HM	JEM	libvpx	libaom
HM	-	35.5	-24.9	4.4
JEM	-25.3	-	-43.4	-21.2
libvpx	37.4	83.9	-	39.0
libaom	-3.8	29.0	-27.6	-

Codecs	HM	JEM	libvpx	libaom
HM	-	40.7	-22.5	5.7
JEM	-27.6	-	-43.7	-22.6
libvpx	30.0	84.6	-	36.7
libaom	-5.0	30.7	-26.3	-

2.2 Deep learning overview

This section will give a brief overview of some important concepts of deep learning, a subset of machine learning, which is an important tool applied in Chapters 5 and 6 of this work. Machine learning is the scientific field that studies computational algorithms and models that learn to perform specific tasks based on previous data. These techniques are currently applied in many aspects of modern society with applications in a variety of areas, for instance, recommendation systems, search engines, digital assistants and digital photography. Machine learning has become a popular field for multidisciplinary research with rapid innovations and direct application into real-world products.

In the past, the development of machine learning algorithms required specialized domain knowledge and carefully designed features to extract meaningful information from raw data that could be used by the models. However, in recent years, due to access to more powerful computational equipment and the availability of very large datasets, a sub-classes of machine learning, known as neural networks, has become increasingly popular [51]. Neural networks are a circuit of individual elements, called neurons, structured in layers, that apply a linear function of trainable weights and biases to the input. When combined with non-linear activation functions, multiple layers, large amounts of data and a differential loss function (or cost function), these networks are able to learn complex functions with minimal domain-specific knowledge. The network parameters are learned by applying an algorithm known as Stochastic Gradient Descent (SGD) [52], which solves the minimization of the cost function by propagating the gradients backward from the output layer all the way to the input layer. Each parameter update is performed after being processed by a defined number of training samples, given by the batch size.

The term deep learning refers to the use of neural networks with an increased number of layers (depth), which tend to be more powerful than more shallow networks. Deep learning algorithms have overcome handcrafted machine learning methods in almost a variety of tasks including most computer vision applications. In addition, for some specific tasks such as visual recognition [53] or strategy games [54], these methods are even able to outperform human subjects.

2.2.1 Convolutional Neural Networks

Computer vision, which is the discipline concerned with designing computer systems to understand and learn from visual representations, such as images, videos or other types of multi-dimensional

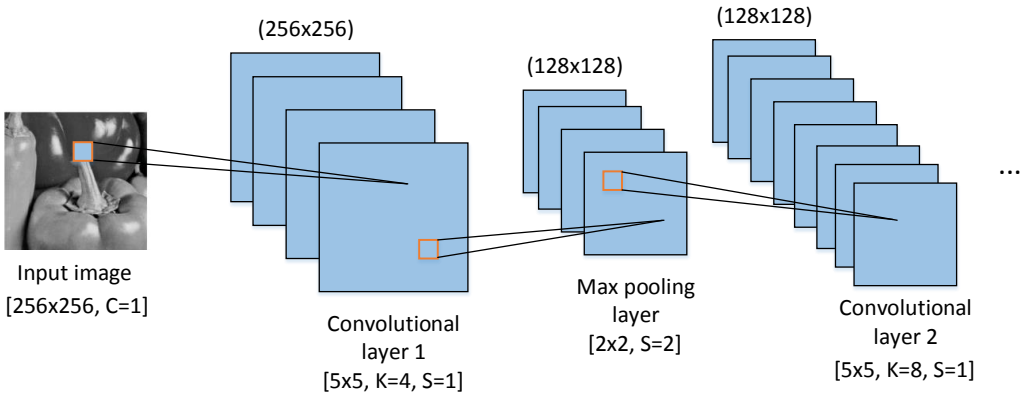


Figure. 2.9 Representation of the first layers of a convolutional neural network. C represents the number of colour channels, K the number of filters (depth) and S the stride.

data, has seen great improvements with advances in deep learning. For instance, in recent years, the accuracy of an image classification challenge on a very large labelled database, named ImageNet [55], has dramatically improved from 16% in 2012 [56] to 2.3% in 2017 [57]. One particular type of deep learning network, Convolutional Neural Network (CNN or ConvNet) has been widely adopted by the computer vision community. CNNs are composed of a number of stacked convolutional layers and, optionally, pooling layers. Each convolutional layer contains multiple learnable filters or kernels of a defined size (e.g. 3×3 or 5×5) that are applied successively to the outputs of previous layers. Pooling layers (e.g max pooling layers), on the other hand, locally combine the results of these convolutions within neighbouring regions, reducing the spatial dimensionality of the representations and creating invariance to translational shifts. In addition, each convolution or pooling is applied on a block that is shifted by a fixed number of positions, controlled by the stride.

Figure 2.9 presents an example of the first 3 layers of a CNN that processes a greyscale image (number of colour channels = 1) of resolution 256×256 pixels. The first convolutional layer contains 4 filters of size 5×5 with stride 1 (S=1). These are applied to the entire image and results in 4 filtered representations containing 256×256 coefficients. Next, a max pooling layer with stride 2 keeps only the maximum value of a 2×2 neighbourhood, reducing the representations to 128×128. Finally, a second convolutional layer is applied on the output of the pooling layer.

2.2.2 Activation functions

In neural networks, including CNNs, non-linearities, also called activation functions, are employed. These introduce non-linear behaviour to this otherwise linear system which is one of the key elements for the learning of complex functions. Several types of activation functions have been developed over the years and the main idea is to control when a given neuron is "active" or "inactive" given the input value. The sigmoid function is defined by $y = \frac{1}{1+e^{-x}}$, where x is the input and y the output. Its shape is similar to a step function, meaning that for negative values, the output tends to 0 and for positive values, it tends to values close to 1. However, sigmoid functions have a drawback in the

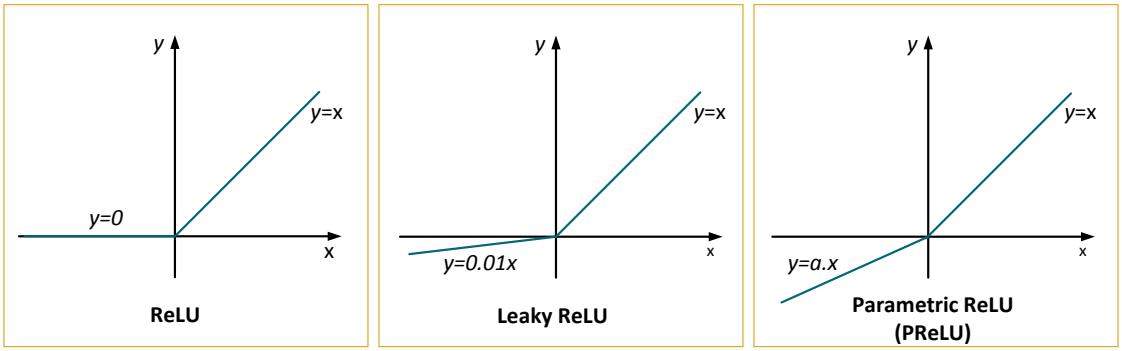


Figure. 2.10 Examples of different ReLU activation functions.

fact that they allow for the "vanishing gradients" problem [52], which stops or decreases the rate of learning for some neurons. In order to overcome this and other issues, other activations functions have been proposed, including the Rectified Linear Unit (ReLU) [52] and its variations. Figure 2.10 presents three types of ReLU functions: the conventional ReLU, the leaky ReLU [58] and finally the parametric ReLU (PReLU) [59]. The use of a small slope for negative input values on the Leaky ReLU and the parametric ReLU promotes better gradient behaviour compared to the conventional ReLU.

2.2.3 Training methodologies

When training a deep neural network, several aspects need to be taken into account. These include the training dataset and hyper-parameter choice. A few recommendations, grouped in different categories, are summarized bellow [60–62]:

- **Training dataset**: the use of a large and varied training dataset and a separate test or validation set is essential to avoid overfitting the network, which means that it will not generalize well to new samples. In addition, to further improve generalization, data augmentation techniques using rotations, translations, illumination variations and horizontal or vertical flips of the training samples are recommended. Finally, the raw input data is usually normalized by calculating and removing the mean value and optionally dividing by the standard deviation.

- **Hyper-parameters:** several parameters that control the training of neural networks need to be carefully selected considering the domain and application. One important parameter is the learning rate, which controls the change of the weights and biases of the neurons or convolutional filters on every step. The higher the learning rate, the quicker the model learns. However, if this parameter is set too high, the training can become unstable. For this reason, several values are generally tested, in an iterative fashion, until the ideal one is found. Another parameter to consider is the batch size, which refers to the number of samples per step. Although larger batch sizes promote better conversion due to the use of more samples for each step,

smaller batch sizes have been found to increase the models' performance and generalization ability [63].

- **Optimization algorithms:** Gradient descent [61] is the most popular optimization technique applied for neural networks. Its goal is to minimize the differentiable loss function by updating the learnable parameters of the network using an algorithm called backpropagation [61]. Although effective, conventional gradient descent is very slow because it requires processing the entire dataset before performing a single parameter update. Therefore, variations that apply the algorithm for each training sample or "mini-batch" have been proposed. These are called Stochastic Gradient Descent (SGD) and mini-batch gradient descent, respectively. Further improvements to these algorithms have been proposed in the literature and include the use of Momentum and different learning rates for different parameters. Currently, the most widely employed optimization strategy is the ADAM algorithm [64], which combines the benefits of two previously proposed strategies: Adaptive Gradient Algorithm (AdaGrad) [65] and Root Mean Square Propagation (RMSProp) [66].

- **Weight initialization:** In order to promote better learning and avoid problems such as vanishing gradients or exploding gradients, especially for deeper networks, weights and biases of neural networks need to be properly initialized prior to training. Random initializations have been used in the past, however, other heuristic algorithms have been proposed, which work better with specific activation functions, such as ReLU. One of the most popular is the He initialization [59], which initializes the weights by sampling from a Gaussian distribution and multiplying by a value that depends on the number of inputs to that layer.

- **Regularization:** In order to avoid overfitting, apart from increasing the number of training samples, regularization techniques are usually employed. These include an adjustment of the loss function by the addition of an L1 or L2 regularization term [61], penalizing large weights. Another very different technique is dropout [67], which modifies the network by randomly and temporarily deactivating parts of the network during training.

2.3 Summary

This chapter provided an introduction to video compression including the differences between modern video coding standards. In addition, the basic concepts of deep learning were presented. All of these are key concepts in the remaining chapters of this thesis.

Digital video, in its raw form, requires a high amount of data, which is impractical for storage and transmission. For this reason, video compression standards are crucial systems that aim at compacting the information by exploiting different types of redundancies in the data. The ultimate goal is to optimize the rate-distortion trade-off, reducing the video bitrate, without significant compromising the output video quality.

Throughout the years, multiple video coding standards have been proposed, mostly led by highly organized standardization bodies. After the introduction of the hybrid video codec, the main coding structure has seen small incremental improvements with each generation of video codecs. Nonetheless, each new codec has provided significantly improvements in compression efficiency, with the introduction of more effective coding blocks and tools. Recently, the introduction of royalty-free video codecs has led to a higher variety of competing video codecs. Still, with the demand for higher video quality and immersive content, video compression remains an active area of research and development.

Machine learning techniques have seen great adoption and have provided with breakthrough performance in a variety of computer vision fields, especially after the development of deep learning. For this reason, it would be expected that the application of these techniques for video coding would have the potential to provide even higher compression gains than achieved by conventional methods. Chapters 5 and 6 of this thesis will describe a novel video compression scheme that employs these learning techniques. For this reason, this chapter included a brief introduction of deep learning, including the concepts of convolutional neural networks, activation functions and relevant training techniques.

Chapter 3

Texture analysis and synthesis

Textures form a predominant part of the natural world and include content such as grass, trees, rock surfaces and ocean waves. For this reason, they are an important aspect of digital videos and images. Although the definition of texture is not consensual, it is generally defined as a region with a large number of elements called *textels*, which are micro-objects placed in an appropriate way forming a particular texture [68]. Textures can then be characterized by the randomness of this placing, referred to as regularity, which can range from regular to stochastic. A regular texture is formed by a well defined tiling organised into strong periodic patterns, while a stochastic texture exhibits more random arrangements. Figure 3.1 presents examples of non-textured content (structures), and textured content with varying levels of regularity.

A subset of the work presented in this chapter has been published in [69].

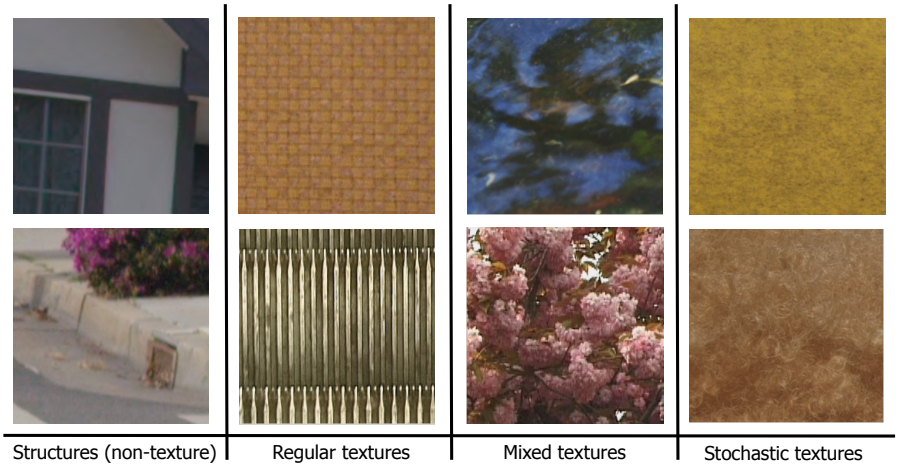


Figure. 3.1 Examples of structures and textures with different characteristics from regular to stochastic.

Textured content can either be rigid (static textures) or non-rigid (dynamic textures) depending on the motion type. Dynamic textures exhibit similar spatial properties to static textures, but evolve in their appearance over time governed by underlying physical processes [70]. Examples of dynamic

textures include ocean waves, waterfall, fire, leaves blowing in the wind, grass fields and smoke. However, due to camera motion or lighting changes, the representation of static textures can also change over time. Figure 3.2 shows sample frames of a static texture, a rock wall, and a dynamic texture, waves, obtained from the BVI-Texture database [4]. This open dataset contains a number of high definition (1920×1080) static and dynamic texture sequences.

(a) Static texture (b) Dynamic texture

Figure. 3.2 Sample frame of static and dynamic texture videos: (a) *BricksTiling* is a titling shot of a rock wall, (b) *CalmingWater* is a shot of a wavy water surface.

3.1 Review of texture analysis and synthesis

3.1.1 Texture analysis

Within image and video processing, texture analysis comprises several tasks, such as texture classification, segmentation, characterization and synthesis. These techniques have found applications in a number of fields including medical imaging, remote sensing and image/video compression [68].

Typically, four types of approaches have been used to analyse textures: statistical methods, spectral methods, model-based methods and structural methods [68, 71]. Statistical-based methods extract features or statistics from the texture region. Examples of features include the Gray-Level Co-occurrence Matrix (GLCM) [72] or geometrical features such as edges. Fourier transform, sub-band decomposition using Wavelets or Gabor filters are often used for texture analysis as spectral methods [68]. Model-based methods attempt to find an appropriate representation of the texture by estimating a set of parameters which define its appearance. Autoregressive (AR) models, Markov Random Fields (MRF) or fractal models are examples of popular approaches for texture modelling. Lastly, structural methods are based on the notion that textures can be considered as two-dimensional periodical patterns of a set of primitives and their properties and placement rules are used to characterize the texture.

3.1.2 Texture synthesis

Texture synthesis is the process of generating a new texture based on a sample texture, in a way that when perceived by a human observer, appears to be generated by the same underlying process [73]. These techniques can be applied to both static and dynamic textures. In the following sections, a brief

description of previous work in image (Section 3.1.2.1) and video (Section 3.1.2.2) texture synthesis is presented. In addition, their application in compression is also discussed.

3.1.2.1 Image texture synthesis

Texture synthesis has been extensively studied in the field of computer graphics for a variety of applications, including the generation realistic textures for the rendering of complex or large scenes [74, 75], to fill empty texture regions, known as image in-painting [76] and for image compression [77, 2, 78]. In the latter case, higher compression efficiency can be achieved by replacing the explicit encoding of textured regions with synthesised textures.

Texture synthesis techniques can be split into two main categories: parametric and non-parametric. The first uses a set of parameters to describe the sample texture and to subsequently generate new textures. This class of algorithms has been proposed in many works, for instance, in [79], which characterize a texture by means of Wavelet-based features and their statistics. On the other hand, non-parametric methods synthesise new textures by generating either one pixel or a patch at a time from example pixels or patches from the input texture [80, 74]. These methods are known as non-parametric due to the fact that no parameters are estimated. One popular example of non-parametric texture synthesis is the *Graphcut* method proposed in [75].

Application of texture synthesis to image compression As mentioned above, texture synthesis has been applied to image compression [2, 78]. The idea behind these methods is to leverage the visual masking effect of texture content on the Human Visual System (HVS) by replacing the coding of textured regions with visually similar synthesised content. These methods are known as image compression-by-synthesis. In [2], the authors proposed a synthesis scheme integrated with JPEG [77]. Images are first segmented into regions using a wavelet-based image segmentation approach and homogeneous texture regions, which contain the same texture process, are selected. Those regions are then synthesised using *Graphcut* [75]. Figure 3.3 presents an example of a reconstructed image compressed by this method compared to conventional JPEG coding. Although both images have similar visual quality, the synthesized version consumes fewer bits for encoding. However, if one looks closely, differences can be spotted, for instance, on the ground, where the resulting texture looks plausible but does not contain the exact texture pattern as the original.

3.1.2.2 Video texture synthesis

With regards to video texture synthesis, achieving perceptually reliable results is a more complex problem. The aim of video texture synthesis is to synthesize an infinite number of video frames based on a limited sample of the input texture. Some of the approaches aim to find smooth transitions from one part of the video to another in a way that no relevant temporal boundary are perceivable [81]. A similar idea has also been proposed by the *Graphcut* authors [75] which extend their technique to videos by considering the same approach but in a 3-D space. Finally, [82] proposes that a dynamic

Figure. 3.3 Example of synthesis results for the Goldhill image. Left: synthesised image at 1.99 bpp; Right: JPEG image at 2.30 bpp. From [2].

texture sequence can be represented by a dynamic linear model (DLM), more specifically, a first order Autoregressive Moving-Average (ARMA) process with a white zero-mean Gaussian input.

Application of texture synthesis to video coding Texture synthesis has also been applied to video compression with a similar general idea as for the image compression application [83–85, 3]. Figure 3.4 shows a diagram of the most important modules of a video compression scheme using texture synthesis. In general, these approaches add two new modules to the video compression scheme: texture analysis and texture synthesis. The first aims at identifying regions in the video that appear to be good candidates for synthesis and therefore do not need to be coded. This process results in side information that is transmitted to the decoder, which in turn, is responsible for applying synthesis techniques to reconstruct the missing regions. If this process can be applied in a way that the reconstructed frames maintain their perceptual quality, usually quantified by a Video Quality Assessment (VQA) module, significant bitrate savings can be achieved. Table 3.1 presents the characteristics of a number of previous works on texture synthesis for video coding. Moreover, a detailed review of this topic can be found in [86].

Within the analysis module, texture regions are usually identified and separated from non-textured regions. This is achieved by means of segmentation and texture classification. The authors of [84] propose to segment and classify textures based on the extraction of GLCM and Gabor filter features, while [85] uses Steerable Pyramids to identify texture regions, but is limited to still image compression. In [3], a Wavelet-based approach (using DT-CWT [87]) is used to segment homogeneous textured regions which are then classified into non-textured, static textures and dynamic textures based on Wavelet and motion vectors-based statistics. Figure 3.5 shows an example of the segmentation and classification results of [3].

Several static and dynamic texture synthesis approaches have been proposed for video compression throughout the years. A Markov Random Field (MRF) model is used in [85] to synthesize similar static textures at the decoder while [83, 84, 3] propose to conventionally encode the key frames and

Table 3.1 Summary of previous work in video compression using texture analysis and synthesis based on the techniques used for segmentation and classification, static texture synthesis and dynamic texture synthesis

Ref.	**Segmentation and Classification**	**Static Textures**	**Dynamic Textures**
[83]	Generic homogeneous segments detector	Warping using 8-parameters Perspective model	3-D graphcut synthesis
[84]	Segmentation and classification using texture features including Gray Level Co-occurrence Matrix (GLCM) and Gabor filters	Warping using 8-parameters Perspective model and foreground/background global motion compensation model	N/A
[85]	Classification of homogeneous textures using Steerable Pyramids	Modelled as Gaussian Markov Random Fields for image coding only	Two modified ARMA models after camera motion compensation which are either added to the reference picture buffer or used directly for prediction
[3]	Region-based segmentation using dual-tree complex wavelet transform and Classification (DT-CWT) in non-textured, static and dynamic textures based on DT-CWT and motion characteristics	Warping using 8-parameters Perspective model	Modified ARMA model after warping applied locally to a GOP

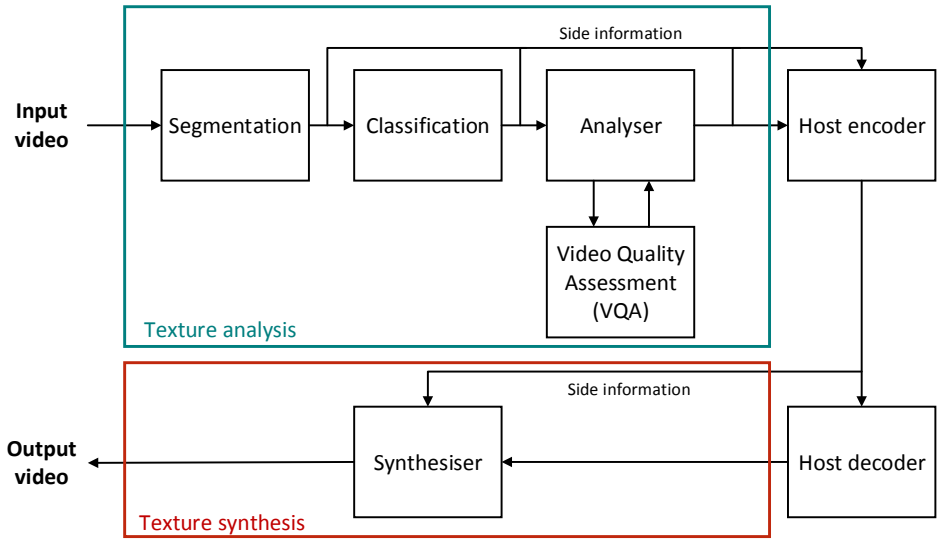


Figure. 3.4 Diagram of the general structure of texture analysis and synthesis-based video compression frameworks.

apply warping to the other frames using an 8-parameter perspective motion model. In addition, in [3], dynamic textures in bidirectionally (B) predicted frames are skipped at the encoder and synthesised at the decoder using a dynamic textures synthesis algorithm. The most popular algorithm for synthesising dynamic textures within a compression framework is to use modified versions of the ARMA model, restricted to a single GOP [85, 3].

Moreover, within a synthesis-based coding framework, in-loop VQA modules are generally necessary in order to assess the quality of the reconstructed textures. The video metrics used are generally perceptually inspired since conventional metrics such as PSNR are unable to accurately correlate with subjective scores on synthesised content [3]. For this reason, the authors of [3] propose

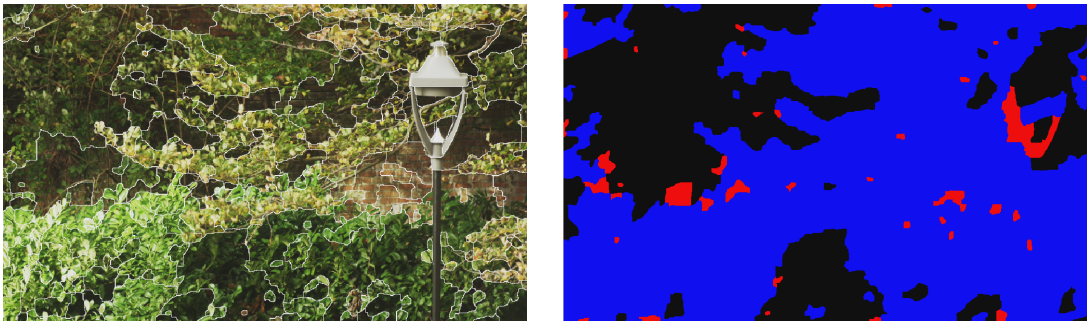


Figure. 3.5 Result of the segmentation (left) and classification (right) processes proposed by [3] for the HD resolution sequence LampLeaves from the BVI texture database [4]. Each region is classified into three classes: non-textured (black), static textures (red) and dynamic textures (blue).

a VQA methodology for synthesised content based on a perceptual Artifact-based Video Metric (AVM), which estimates blurring distortion, similarity distortion, and visible edge artifacts. Later on, they propose an updated perceptual metric called PVM [35].

3.1.2.3 Limitations of texture synthesis for video compression

Texture synthesis for video compression has been an active area of research for the last decade but has not achieved good adoption in standardized video codecs. We have identified a number of reasons for this:

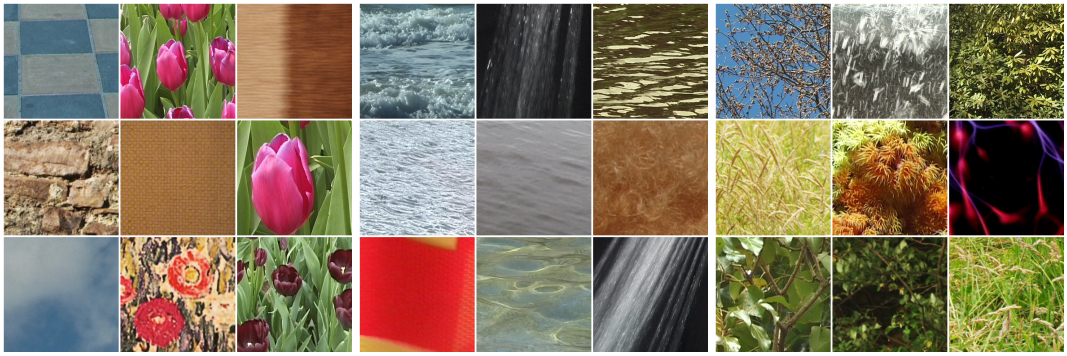
- The majority of reported texture synthesis methods has used H.264 as the host codec. However, the most recent video coding standards, such as HEVC, have improved the compression of bi-directional (B) pictures and therefore, less savings are achieved by skipping regions and warping or synthesizing dynamic texture content than for H.264.
- Current video quality metrics, including AVM and PVM, are still not reliable enough for the estimation of video quality, especially considering synthesised content, which is confirmed by the predominance of PSNR within the video compression community.
- Dynamic texture synthesis algorithms were developed especially for stochastic textures and produce artifacts when the texture is more regular. This will be addressed in Section 3.4.

3.2 HomTex: Homogeneous Texture dataset

As mentioned in the previous section, texture is a fundamental part of visual content and has been exploited for image and video compression. Therefore, in order to further explore this topic, a database of video textures is required. However, most video databases contain scenes with mixed content, including textures and structures (or objects). For these reasons, we have proposed a new dataset containing only homogeneous textures - the **Hom**ogeneous Video **Tex**ture Dataset (HomTex). In this context, homogeneity means that the textures are spatially and temporally consistent, and thus no spatial or temporal segmentation is required.

HomTex comprises 120 sequences with a spatial resolution of 256×256 pixels and contains a variety of texture types. Most of the sequences were obtained by cropping patches from two existing datasets: DynTex [88] and the BVI Texture dataset [13]. DynTex contains a large variety of dynamic texture content and has been extensively used by the research community as a benchmark for dynamic texture classification and synthesis. The BVI texture dataset has been recently developed and contains high quality static and dynamic textures at HD resolution.

In most works, a limited number of texture categories is usually considered for texture analysis and synthesis purposes, more specifically, static and dynamic textures. However, we believe that these categories do not reflect the amount of variability found in visual textures. Therefore, in addition to the uncompressed video content, HomTex also contains manual annotations made by experts on three



(a) Static (b) Dynamic continuous (c) Dynamic discrete

Figure. 3.6 Sample frames from example HomTex sequences classified into different categories based on their dynamics and structure.

different visual characteristics: dynamics, structure and granularity. A brief description of each is presented below:

- **Dynamics:** whether the physical appearance of the texture changes with time (dynamic) or the texture is rigid and the only motion present is due to a moving camera or a moving object (static).
- **Structure:** whether the nature of the texture is one of a continuous deformable media (continuous) or the texture is composed of a collection of structured discernible parts (discrete) [88].
- **Granularity:** related to the size of the smallest recognizable object or texture primitive observed. A texture with smaller size primitives has a high granularity level, while a texture with larger size primitives has a lower granularity level [89].

An important distinction is between sequences that are dynamic but contain different types of structure, discrete and continuous. This classification of dynamic textures has been mentioned previously in [88] but no justification or evidence was shown. In Section 3.3.2.3, we propose to split video textures into three categories based on encoding statistics, namely, static, continuous dynamic and discrete dynamic.

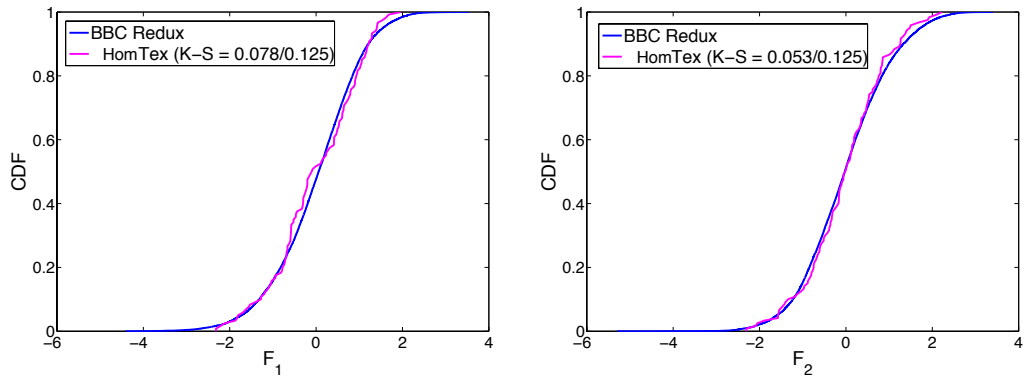
The number of sequences grouped according to these characteristics is presented in Table 3.2. It can be seen that the majority of the sequences, 95, represent dynamic textures, while 25 sequences are static textures. The full annotation of each sequence in the HomTex database can be found in Appendix A. Moreover, Figure 3.6 shows the first frame of a number of HomTex sequences split into the three major categories defined.

In addition, we also wanted to understand if the characteristics of HomTex are representative of the ones found in real world video content. For this reason, an analysis was performed using the methodology of [90], which offers a way of measuring how well a given dataset reflects the

Table 3.2 Annotation of the HomTex based on three characteristics.

Dynamics	Structure	Granularity	Num. of sequences	Total
static	continuous	high	0	25
		medium	1	
		low	4	
	discrete	high	2	
		medium	6	
		low	12	
dynamic	continuous	high	4	45
		medium	18	
		low	23	
	discrete	high	9	50
		medium	32	
		low	9	

characteristics of another dataset by evaluating the distributions of low level visual features between different datasets. This work has studied the features of a large-scale database containing thousands of modern broadcast video content, BBC Redux [91]. Following this methodology, several features including spatial information, motion vectors, colour information and contrast were extracted from each database and were transformed into 5 orthogonal factors using Principal Component Analysis (PCA). The frequency distributions for the most relevant factors, Naturalness and Movement, were then compared with that of BBC Redux. Figure 3.7 shows the Cumulative Distribution Functions (CDF) distributions of both datasets.



(a) Factor 1 - Naturalness

(b) Factor 2 - Movement

Figure. 3.7 Distribution comparison between BBC Redux and HomTex.

We then apply the two-sample Kolmogorov-Smirnov test [92] (also shown in Figure 3.7). The values obtained indicate that the hypothesis that both distributions could come from the same continuous distribution is valid. This is an important finding because it indicates that the conclusions drawn from this database might be generalized to a larger database of real word video since it contains a similar distribution of low level visual characteristics.

The HomTex database is publicly available for the research community and can be downloaded from its webpage [1]. At the time of writing, it has been used in several publications [93–95] in the field of texture analysis and synthesis for video coding.

3.3 Texture analysis based on HEVC encoding statistics

In this section, a study of different video texture properties based on encoding statistics extracted from the HEVC reference software, HM [96], is presented. The purpose of this work is twofold: (i) to achieve a higher understanding of how modern video codecs perform for different texture types and; (ii) to achieve a better characterization of textures that can aid future research in this field. The statistics exploited include mode selection, partitioning information, motion vectors and bitrate allocation. For this study, we use the HomTex dataset, proposed and described in Section 3.2.

While modern video codecs such as HEVC already consider some perceptual characteristics of the HVS, increased gains might be achieved by further exploiting texture masking effects. As mentioned previously, in the context of video compression, textures are typically categorized into two different classes: static and dynamic [13, 3, 85, 84, 83]. However, the classification of dynamic textures is not always consistent and includes a large range of diverse content such as water, leaves, smoke and trees. This classification might not be optimal when trying to apply and optimize coding strategies, such as texture synthesis [3, 85, 84, 83], which to the best of the author's knowledge, have treated all dynamic textures equally. This motivates us to examine this topic more closely and to propose a more robust classification that is justified and validated using statistics from HEVC encoding.

We propose to classify video textures into three major categories for video coding purposes, more specifically: static textures, continuous dynamic textures and discrete dynamic textures. Multiple works concerning the classification of dynamic textures have been recently published [97, 98]. However, their aim has been to classify textures based on semantics, which is not always useful for coding. The distinction between continuous and discrete dynamic textures was made in [88] when annotating the DynTex database, but no data was presented to support this classification. In this work, we present a detailed analysis of different texture types with respect to a large number of HEVC encoding statistics. The homogeneous video texture dataset, HomTex, is used which allows the statistics to be directly associated to a single texture without other interfering content. A number of papers have been recently published which include analysis of encoding statistics [99–102], but the number of statistics examined was very limited and for different applications. Therefore, we believe this work introduces a new perspective into the subject of texture classification for video coding.

3.3.1 Encoding statistics

HEVC encoding statistics were extracted using the test model version HM 16.2. All sequences from the HomTex dataset were encoded using the Main profile using three configurations: Random Access,

[1] https://data.bris.ac.uk/data/dataset/1h2kpxmxdhccf1gbi2pmvga6qp

Low Delay and All Intra and 5 base QP values {22, 25, 27, 32 and 37}. A total of 39 statistics were obtained from the encoding process at the Coding Tree Unit (CTU) level. These were then post-processed to obtain features per sequence, for various QP values and frame types (I, B and P). For the purpose of this work, P frames are defined as using only past frames for reference and B frames as using both past and future frames.

Table 3.3 shows a summary of all the statistics extracted grouped into different categories along with a short description of each. For the measure of correlation between the original and the residual frames, the 2-D Pearson product-moment correlation coefficient was used, considering only the luminance (Y) component of the frames.

3.3.2 Results and analysis

The analysis performed and the results obtained are presented in four sub-sections. Section 3.3.2.1 describes the observations made through examination of the distribution of the statistics of the different classes. Section 3.3.2.2 analyses the effect that the different levels of granularity have on the encoding behaviour. Section 3.3.2.3 presents a clustering analysis that was employed with the aim of finding the inherent structure of the data. Finally, Section 3.3.2.4 makes some suggestions for areas of further work that have the potential to offer compression performance improvements.

3.3.2.1 Effect of dynamics and structure on encoding behaviour

We split the extracted data into the three classes mentioned previously (static, continuous dynamic and discrete dynamic) and computed texture-class specific distributions for each of the extracted statistics. This helps in identifying class related patterns in the behaviour of the encoder. Figure 3.8 depicts the distributions of a subset of the statistics for the B frames of the Random Access configuration, using a QP value of 25. The results show a clear distinction between the different texture types in terms of the encoding statistics. Additionally, a number of observations made are described below.

The percentage of prediction modes (intra, inter, skip or merge) vary significantly for different texture types. As expected, static textures are associated with a high percentage of skip mode, with a median of around 70% of the blocks. This is because static textures mainly contain global motion (camera panning, tiling) which is easily compensated by block prediction algorithms in HEVC. On the other hand, continuous dynamic textures are mostly encoded using intra mode, which means that motion compensation fails to produce an accurate prediction of the frames based on previously encoded frames. Finally, discrete dynamic textures contain a high percentage of skip, merge and inter modes, in contrast to continuous dynamic textures.

The Rate-Distortion (RD) performance of the encoder also varies with the texture type. As would be expected, static textures require a much smaller number of bits (avgBits) compared to dynamic textures and exhibit lower distortion (avgDist) for the same QP. An interesting finding is that discrete dynamic textures, on average, require a higher number of bits per pixel compared to continuous dynamic textures and result in higher distortion.

Table 3.3 Statistics extracted from HM during the encoding process.

Category	Statistics	Description
Prediction modes	intra (%) stdIntra Skip (%) stdSkip merge (%) stdMerge inter (%) stdInter	Percentage and standard deviation of the prediction modes. Those are extracted for each partition and scaled according to the size of the partition and averaged over all CTUs and frames within a sequence.
Reference indexes	ref0 (%) ref1 (%) ref2 (%) ref3 (%)	Percentage usage of different pictures from the reference picture list, scaled by the size of the partition.
Partitioning	avgPart stdPart	Average and standard deviation of the number of partitions per CTU
Bits	avgBits stdBits	Average and standard deviation of bits per pixel (bpp)
Distortion	avgDist stdDist	Average and standard deviation of the Sum of Absolute Differences (SAD) per pixel which is the distortion used within the RDO.
Bit allocation	bitsModeSignal (%) bitsPart (%) bitsIntraDir (%) bitsMergeIdx (%) bitsMotionPred (%) bitsResidual (%) bitsOthers (%)	Percentage of bits spent to encode mode selection, partitioning, intra modes, merge indexes, residual coding and others
Residual Statistics	avgMSEresi stdMSEresi avgMSERecError stdMSERecError avgCorrResi stdCorrResi avgCorrCodedResi stdCorrCodedResi	Average and standard deviation of the Mean Square Error (MSE) of the residual, the MSE of the reconstructed frame, correlation between original and residual frames, correlation between residual and coded residual
Intra mode	DCIntra PlanarIntra avgIntraDir stdIntraDir	Percentage of Intra predicted partitions that use DC and planar mode (scaled by size). Average and standard deviation of the Intra mode direction
Motion Vectors	avgLengthMV stdDistMV	Average length of the motion vectors. Standard deviation of the distribution of motion vector's directions

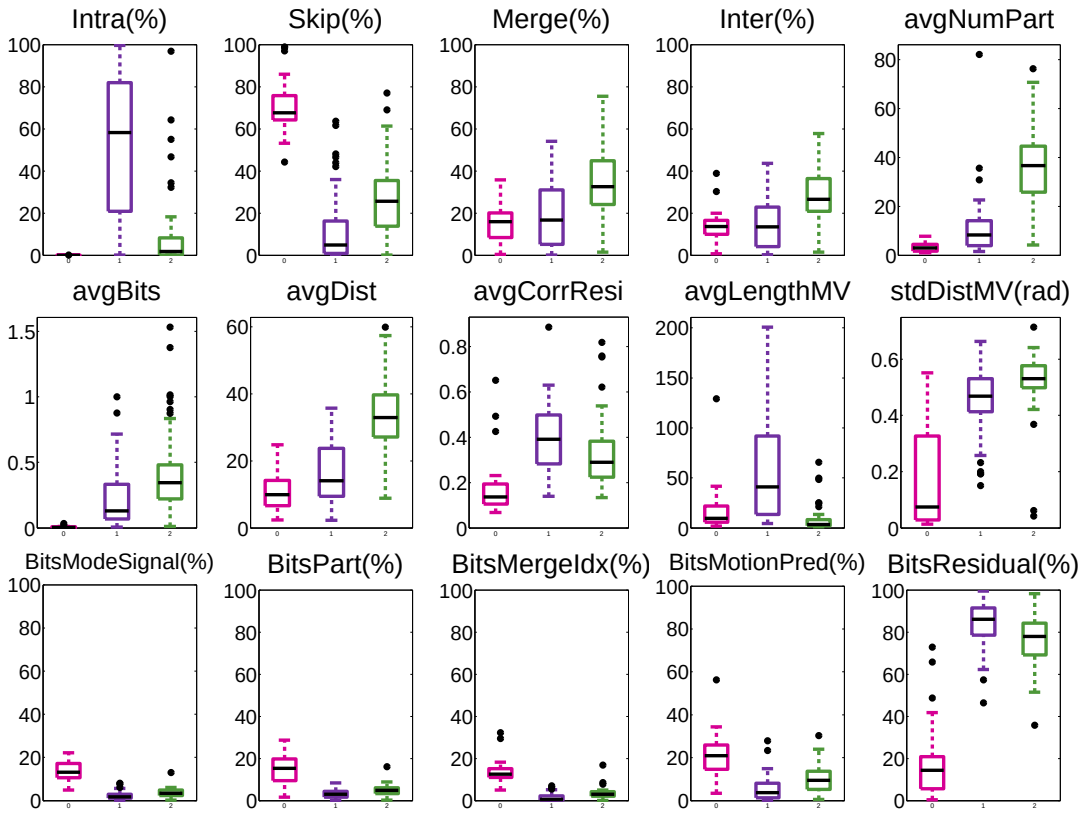


Figure. 3.8 Distribution of several encoding statistics for the static (magenta), continuous dynamic (purple) and discrete dynamic (green) textures. These results were obtained using Random Access configuration and a QP value of 25.

The number of CTU partitions also varies with texture type, with the lowest number of partitions being observed for static textures with a median of 4 partitions per CTU, and the highest for discrete dynamic textures with a median of 35 partitions per CTU. As for continuous dynamic textures, they require fewer partitions compared to discrete dynamic textures but more than static textures.

In terms of the allocation of bits, for both types of dynamic texture the majority of bits are spent on coding the residual signal (more than 80% of the total bits spent). This means that the bits used for coding additional information, such as motion vectors and mode signalling, are less relevant to the final bitrate of the encoded bitstream. In addition, dynamic textures exhibit high distortion and high correlation between the original frame and the residual since a worse prediction is usually obtained. In contrast, for static textures, the bit allocation is more evenly distributed among the signalling data, motion vector data and residual signal.

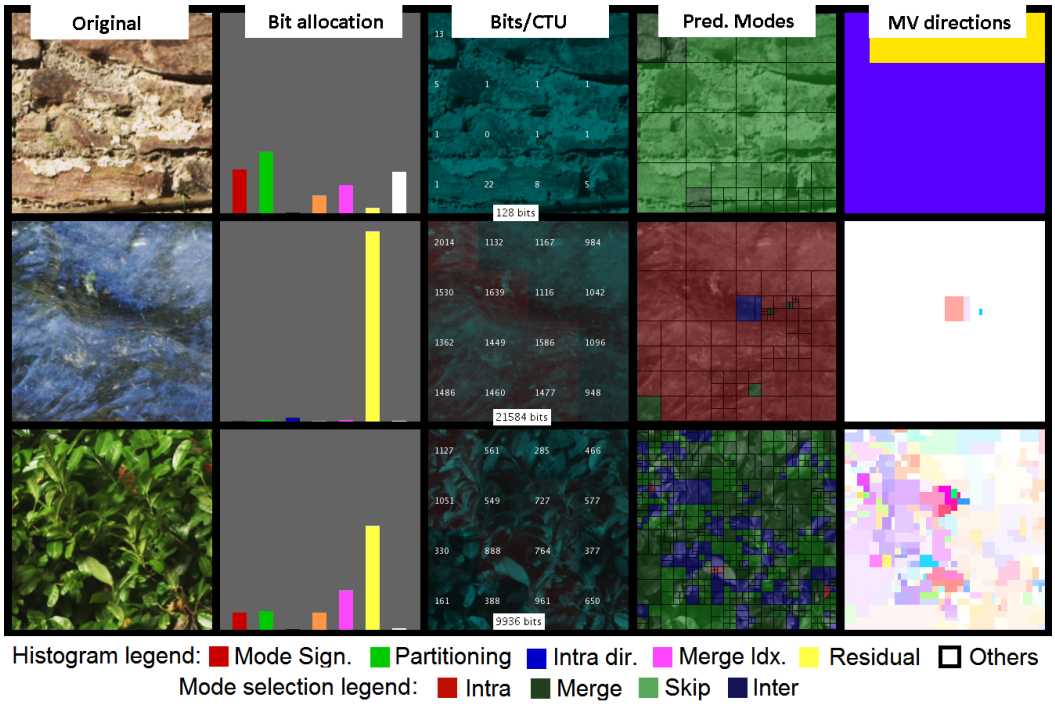
Finally, motion vectors also show different characteristics depending on the texture type. Static textures are associated with motion vectors with a small magnitude (length) and high directional consistency. Discrete dynamic textures also typically contain small magnitude motion vectors but with high directional irregularity. This is understandable since discrete dynamic textures contain structures with local motion characteristics that might be predicted by different motion vectors. In contrast, continuous dynamic textures are associated with large magnitude motion vectors with slightly less directional irregularity.

To aid in the analysis of the statistics and to allow for easier and more intuitive comparison between different sequences, a visualization tool was implemented. This tool generates a visual representation of a subset of the statistics for each video frame. It contains the original frames, a histogram of bit allocation, the bits spent per CTU, a colour coded mode prediction decision per PU, a colour coded representation of the magnitude and direction of the motion vectors (according to [103]), the predicted frame, the reconstructed frame, the residual and the coded residual.

Figure 3.9 presents a screenshot of the visualization tool for three representative sequences of different texture class: *BricksTiling* (static), *CalmingWater* (continuous dynamic) and *LampLeaves* (discrete dynamic). These sequences were encoded using the same configuration described for the results in Figure 3.8. Several observations reported above can also be seen in the visualization. For example, the dominance of skip mode for static textures and intra mode for continuous dynamic textures, the high partitioning of discrete dynamic textures and the high distortion and residual for both types of dynamic textures. In addition, it can be seen that, in comparison with the continuous dynamic texture, the visual quality of the predicted frame of the discrete dynamic texture is higher. This is due to the use of intra mode for the continuous dynamic texture, creating the block effect shown which is a consequence of the failure of the motion prediction technique.

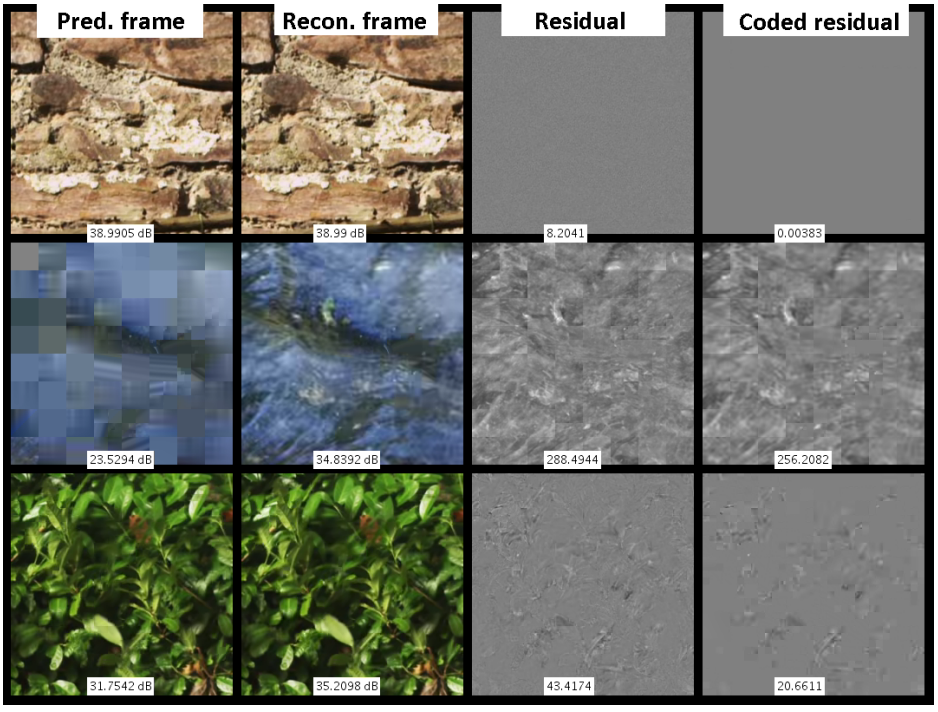
3.3.2.2 Effect of granularity on the encoding behaviour

In order to determine the influence of texture granularity on the encoding statistics, we have investigated the changes between two different granularity versions of the same sequence. In total,



(a)

Original frame, bit allocation histogram (legend below image), bits per CTU, prediction modes and partitioning and color-coded motion vector directions.



(b) Continuation; Predicted frame, reconstructed frame, residual (original - predicted) and coded residual (original - reconstructed).

Figure. 3.9 Visualization tool of a subset of the encoding statistics for three different sequences: BricksTiling (top row), CalmingWater (middle row) and LampLeaves (bottom row).

HomTex contains 40 pairs of the same sequences with different granularities, 8 static, 15 continuous dynamic and 17 discrete dynamic. Figure 3.10 shows bar plots comparing the statistics of low granularity sequences and the respective high granularity versions. The plots show a subset of the B frame statistics and are also divided by texture type. It can be seen that, as expected, increasing the granularity increases the average number of partitions (avgNumPart) and decreases the average length of motion vectors (avgLengthMV). This is due to the lower size of the discernible objects within the texture, which requires more partitions but also reduces the length of the motion (per pixel). Another interesting observation is that for higher granularities, fewer blocks are predicted using intra mode for dynamic textures (the number of intra blocks for static textures is insignificant for both granularities), resulting from more efficient motion estimation. Also, for dynamic textures, skip mode is selected more often. Nonetheless, the bits used to code the high granularity textures (avgBits) does not decrease, because of a higher distortion (avgDist).

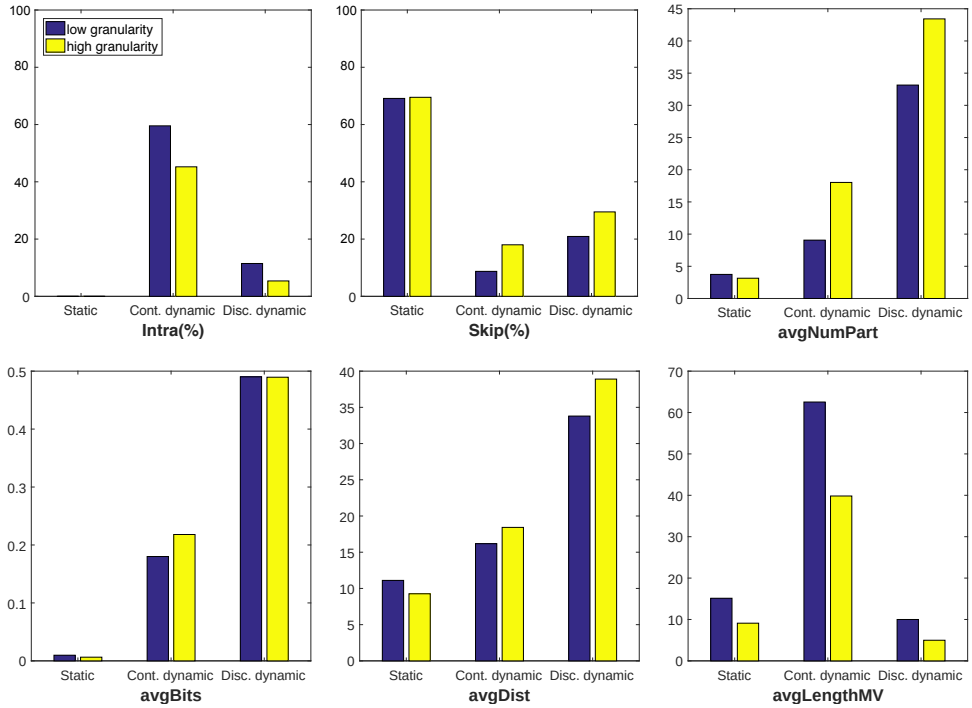
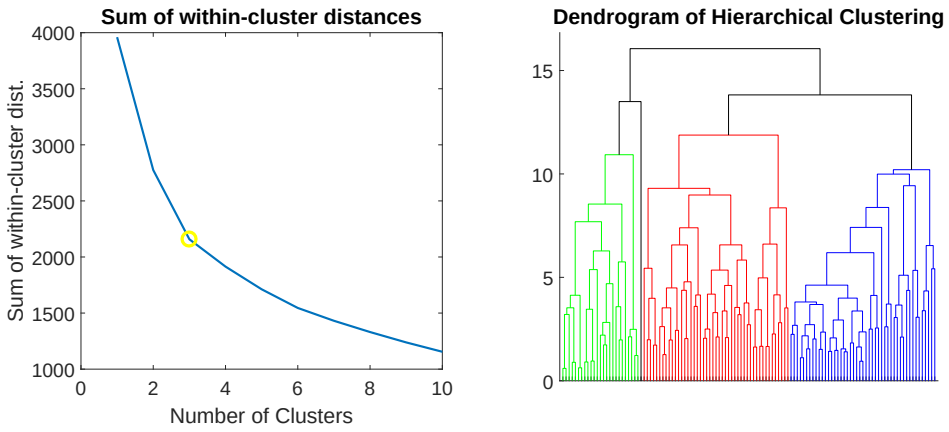


Figure. 3.10 Comparison of the average value of a subset of the statistics obtained from pairs of sequences with different granularities and different texture types for B frames, Random Access configuration and QP 25.

3.3.2.3 Validation of class descriptions through clustering

According to previous observations, there are significant differences between the encoding statistics of static, continuous dynamic and discrete dynamic textures. However, this result alone is insufficient to validate the hypothesis that these three classes represent a good classification based on encoding statistics. For this reason, an unsupervised learning technique, clustering, was applied to all the



(a) (b)

Figure. 3.11 Clustering analysis results on the encoding statistics for Random Access B frames, QP 25: (a) Sum of the within-cluster distances depending on the number of clusters; (b) Result of hierarchical clustering analysis.

extracted statistics to understand the underlying structure of the data. It is important to note that in clustering analysis, the sequences are not labelled with a class. This ensures that the clusters are obtained based on the features only.

Three widely used clustering algorithms were employed, K-Means, Hierarchical Agglomerative clustering (using complete linkage) and Spectral clustering [104]. Additionally, Principal Component Analysis (PCA) [104] was also applied for feature extraction, followed by K-Means using the 5 first principal components (80% of the total variance). The input features were 33 statistics of the Random Access B frames at a QP value of 25. Other settings were also tested, and similar findings were found. Since the majority of algorithms require the number of clusters to be pre-determined, a simple analysis was conducted using a plot of the sum of within-cluster distances for different number of clusters when applying K-means clustering, shown in Figure 3.11(a). The number of clusters is often decided by the "elbow" of this plot [104], which in this case, hints at the existence of three clusters. Moreover, Figure 3.11(b) shows the dendrogram of hierarchical clustering, which works by iteratively adding together the data points that are closer in the data space. It can be seen from the result that mainly three clusters were formed, corresponding roughly to the three manually annotated classes.

The clustering results obtained from applying the different algorithms were evaluated using four popular cluster validity indices: one internal index, the average Silhouette index and three external indexes, Purity, Normalized Mutual Information (NMI) and the Adjusted Rand Index (ARI) [104]. The internal index is based on the distances between and within points of the same cluster while external indexes use class labels. The latter methods used the manual class annotations mentioned in previous sections with a value of 0 and 1 representing a random and a perfect clustering, respectively. The clustering performance is presented in Table 3.4 and indicates that using three clusters is a good representation of the data, since most indices have high values. Additionally, it can be noted that the performance is consistent among the different algorithms employed.

Table 3.4 Clustering performance of several clustering algorithms on all the statistics extracted from HM for a QP of 25 and Random Access configuration.

Clustering method	Silhouette	Purity	NMI	ARI
K-Means	0.45	0.84	0.56	0.57
Hierarchical Clustering	0.36	0.80	0.49	0.48
Spectral Clustering	0.44	0.84	0.57	0.56
PCA + K-Means	0.45	0.84	0.56	0.57

Given the large number of features, visualizing the data is challenging. Therefore, we have opted for a representation based on an undirected graph where each sequence is a node, connected to the n nearest neighbour sequences according to the euclidean distance on the feature space. Several values of n were tested and in the end a value of 15 was chosen to provide optimal visualization. This representation is depicted in Figure 3.12, where static, continuous dynamic and discrete dynamic textures according to the manual labelling are distinguished by the colours magenta, purple and green, respectively.

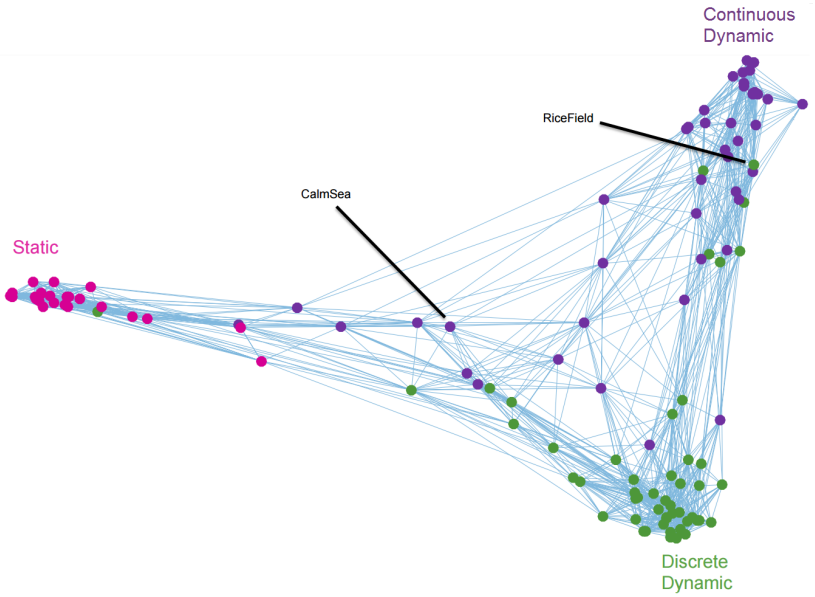


Figure. 3.12 Undirected graph representation of the sequences.

By observing this representation, it is possible to identify three distinct clusters in addition to a number of nodes that do not seem to belong to any of the clusters. These nodes correspond to textures that do not fit the average characteristics of any cluster and could be considered as outliers. An example is the *CalmSea* sequence, which depicts water flowing slowly. Due to the semantic content, water, this sequence is considered to be a dynamic texture. However, since it contains low motion characteristics, motion compensation works better which brings it closer to a static texture, especially for some of the statistics such as *avgBits* and *Skip (%)*. Additionally, for some sequences, the manual class annotations do not agree with the ones of its nearest neighbours. One example is the *RiceField*

sequence, in which, although discrete in nature, its encoding statistics resemble a continuous dynamic texture due to its fine structure and flow-like motion patterns.

3.3.2.4 Suggested areas of improvements in texture coding

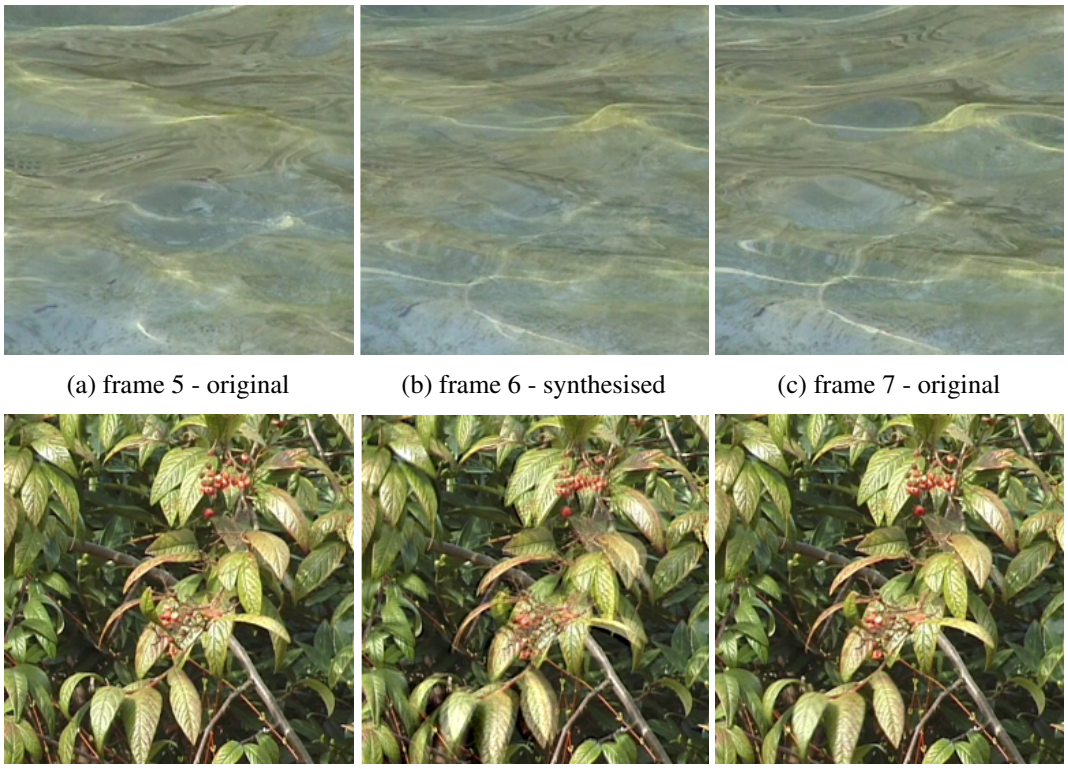
Based on the conducted analysis and the results presented in the previous sections, it is clear that dynamic textures represent a challenging problem for the compression scheme of HEVC. In the case of discrete dynamic textures, better local motion estimation could lead to more accurate predictions, reducing the residual signal. In contrast, for continuous dynamic textures, local motion compensation techniques might fail due to the random nature of this texture type. For these, approaches such as texture synthesis might prove to be helpful by allowing the reconstruction of those textures at the decoder. Furthermore, given that the majority of bits are spent on residual coding for dynamic textures, future work could focus on the development of texture-adaptive residual coding techniques. Finally, for static textures, the compression efficiency of HEVC is higher when compared with dynamic textures, suggesting that the current coding framework is well-suited for this texture type.

3.4 Exploration of texture synthesis using motion models

As mentioned before, previous studies of dynamic texture synthesis for video coding have limited their scope to the development of algorithms for the synthesis of static and dynamic textures. However, Section 3.3, we have exposed the large variability among video content that is usually defined as dynamic texture and, thus, proposed a better classification with differentiation between discrete dynamic textures and continuous dynamic textures. In this section, we analyse the performance of these synthesis approaches on different texture types and explore an alternative to synthesis by means of motion interpolation, more specifically, optical flow.

Figure 3.13 shows three successive frames from a continuous dynamic texture, *GreenWater-scaled* and a discrete dynamic texture *BushesFruits-scaled* from HomTex dataset. The first and third frames correspond to the original frames while the middle frame was obtained by applying the Dynamic Texture Synthesis (DTS) approach proposed by [3]. It can be seen that, for the continuous dynamic texture case, the synthesised result looks plausible. On the other hand, for the discrete texture, the synthesized result contains a high amount of blur and shadow-like artifacts where there is motion of the underlying structures in the scene. This is a common pattern we have identified when applying DTS to discrete dynamic textures.

To overcome this problem, an alternative synthesis approach that uses optical flow [105] is suggested here. Instead of modelling the dynamic texture as a stochastic process, we propose to interpolate the frame by using motion compensation from dense optical flow estimation. The accuracy of the interpolation will depend on how accurately these motion models can capture the local motion characteristics of different dynamic textures.



(a) frame 5 - original (b) frame 6 - synthesised (c) frame 7 - original

(a) frame 17 - original (b) frame 18 - synthesised (c) frame 19 - original

Figure. 3.13 Synthesis results for *GreenWater-scaled* (top) and *BushesFruits-scaled* (bottom) from HomTex using the dynamic texture synthesis approach of [3]. Only the middle frame is synthesised.

The goal of optical flow estimation is to estimate the motion field, which is characterized by the displacement of each pixel in a frame. Several algorithms for optical flow estimation have been proposed since the original formulation proposed in [105]. These different algorithms can be classified into: gradient-based, correlation-based, frequency-based and energy-based methods [106, 107].

Classic gradient-based methods rely on two assumptions, brightness constancy and spatial smoothness [105]. The first assumes that, given a small temporal interval, pixel intensities are translated from one frame to the next, while the latter assumes that neighbouring pixels share the same velocity. Even though these properties rarely hold exactly for real videos, they have worked well in practice. Nonetheless, this formulation is not robust for all content types and motion characteristics, and therefore, progressive additions or innovations have been proposed including the use of course-to-fine estimation to help with large displacements, texture decomposition and high-order filter constancy to reduce the influence of lighting changes [108].

Since discrete dynamic textures are characterized by small structured objects which exhibit random local motion, in order to synthesise a frame of this content type, local motion displacements need to be estimated. Among different optical flow algorithms, the work of [109] is especially suited since it was developed specifically for the estimation of local motion of non-rigid structures. This

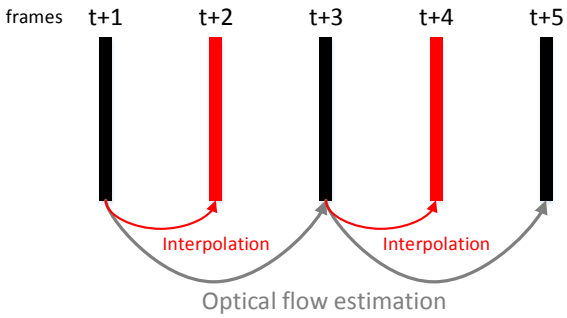


Figure. 3.14 Illustration of how the optical flow estimation is used to perform the interpolation of middle frames.

approach was originally proposed for medical image registration, which aims at spatially aligning two or more images, and is based on the relationship between phase shits of the Dual-Tree Complex Wavelet Transform (DT-CWT) coefficients and dense motion fields. The proposed algorithm uses an iterative process that tries to determine the spatial shift of the image to be registered in order to match the phases of the complex wavelet coefficients of both images. The relationship between the phase shift of the coefficients, θ_d, and the motion field, $\vec{u}(\mathbf{x})$, is given by:

$$\frac{\partial \theta_d}{\partial t} = \nabla_{\mathbf{x}} \theta_d \cdot \vec{u}(\mathbf{x}) \tag{3.1}$$

where $\nabla_{\mathbf{x}} \theta_d = [(\partial \theta_d / \partial x), (\partial \theta_d / \partial z), (\partial \theta_d / \partial z)]$ represents the gradient at $\mathbf{x}$ for sub-band d in the directions x, y and z.

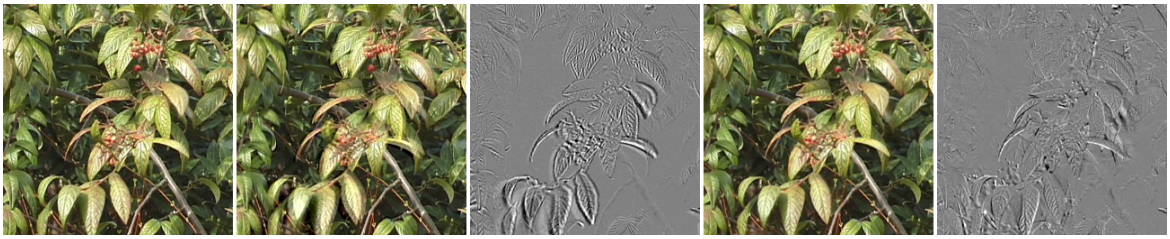
Instead of using this algorithm for image registration, we apply this to image interpolation. For this purpose, we estimate the motion fields between a given frame at time t and the frame at time $t + 2$ and then interpolate the middle frame $t + 1$ by shifting the image at time t by half the amplitude of the motion field. This concept is illustrated in Figure 3.14.

Figure 3.15 shows three examples of synthesised or interpolated frames, along with the residuals between the original (left) and interpolated frames (middle) by each method. It can be seen that DTS produces blurring and shadow artifacts whereas a much sharper image is produced by optical flow interpolation. This can be confirmed by the lower energy residuals obtained.

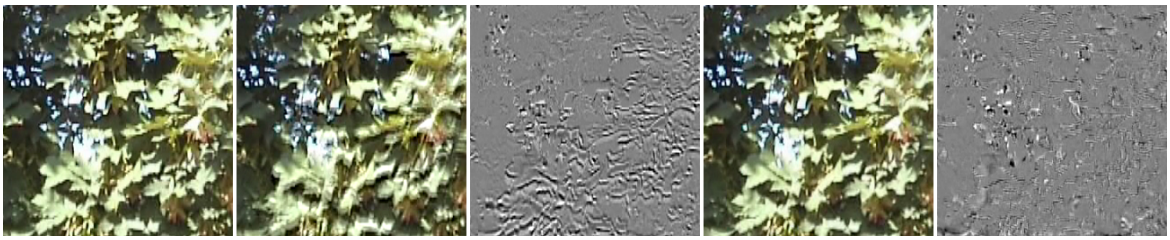
We have compared the performance of the dynamic texture synthesis (DTS) approach of [3] and the optical flow method for all the discrete dynamic texture sequences of HomTex. Based on the average PSNR values of the synthesized frames, for 20 out of 50 discrete dynamic texture sequences, higher quality is achieved by using optical flow interpolation compared to DTS with gains up to 3.3 dBs (on average). Table B.1 in the Appendix B shows the comprehensive results for all sequences. Moreover, based on these results, it is clear that this method does not perform well for every content. We have identified several reasons why this is the case: (i) some sequences contain high amplitude motion, which is not well handled by the optical flow algorithm; (ii) occlusions often occur on the

edges of the video frames, which are not well handled by the algorithm since it warps one frame using the estimated motion field to obtain the interpolated frame; (iii) for some sequences, successive frames are uncorrelated which means that the same structures are not exactly present among different frames, making it impractical to apply motion models.

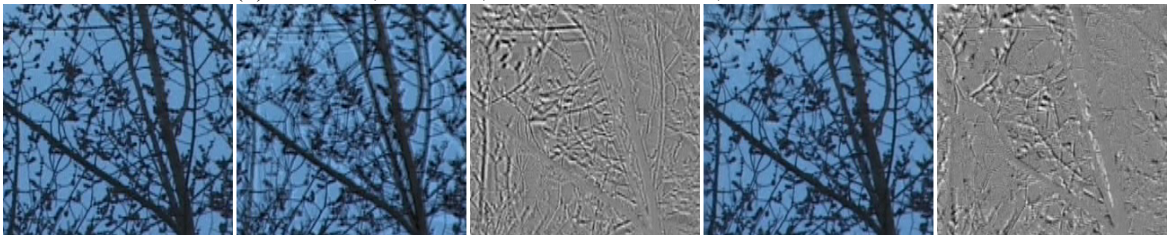
Since this method fails to produce better quality reconstructed frames in all cases, it could be used in conjunction with DTS for future video coding schemes. More specifically, there would be two options to reconstruct a block classified as dynamic texture. At the VQA module, the synthesis/interpolation model that produces the highest reconstruction quality based on a given quality metric would be selected. We believe this approach has the ability to improve the reconstruction quality for some types of content, especially discrete dynamic textures, without affecting the ones for which this method is not suitable, mostly continuous dynamic textures, and can be a future topic of research.



(b) BushesFruit_scaled, frame 18, PSNR DTS: 20.6 dB, PSNR OF: 23.7 dB



(b) DarkTree, frame 14, PSNR DTS: 22.2 dB, PSNR OF: 24.2 dB



(c) LeavesRotating, frame 10, PSNR DTS: 19.5 dB, PSNR OF: 23.4 dB

Figure. 3.15 Comparison between dynamic texture synthesis (DTS) of [3] and the proposed optical flow method (OF) for three different discrete dynamic textures sequences of HomTex. From left to right: original frame, frame predicted using DTS, residual between original and DTS prediction, frame predicted using OF, residual between original and DTS prediction.

3.5 Discussion

This chapter was concerned with the applications of texture analysis and texture synthesis for video compression. Our contributions included the development of an openly available homogeneous texture database, an analysis of textures types based on encoding statistics of HEVC and the exploration of motion models for texture synthesis or interpolation.

We present an extensive and detailed study of different video texture types from the perspective of the encoding statistics of HEVC. Since textures represent a crucial part of video content, it is important to understand the shortcomings of modern video codecs, such as HEVC, for these content types helping to create a foundation for the development of new tools for the next generation of video compression standards. An unsupervised learning technique has been applied to the data collected and reveals a clear structure, with a separation between static, continuous dynamic textures and discrete dynamic textures.

Following this analysis, we explore different alternatives to texture synthesis proposed in the past, more specifically, using dense optical flow algorithms. We have found that these work well for some types of content, usually the ones classified as discrete dynamic textures. The shortcomings of this approach were also analysed and some suggestions for future incorporation of these techniques for video coding algorithms were described.

Chapter 4

Introduction to spatial resolution adaptation

As high resolution and immersive formats (HD, 4K and 8K) become increasingly popular, video compression standards struggle to achieve high compression gains while keeping the complexity manageable. Throughout the years, several different strategies have been proposed in order to improve rate-distortion performance of video coding standards including texture synthesis approaches, covered in Chapter 3. However, these approaches are restricted to textured content, and therefore, have a narrow scope for real word video. In this chapter, we consider another method, known as resolution adaptation, in which the idea is to encode a lower spatial resolution input video and then reconstruct the full resolution at the decoder. These types of techniques work by balancing the resampling and compression distortions to produce a better rate-distortion trade-off. Previous work on this topic obtained significant gains, especially at lower bitrates [110–115]. This approach contains similarities to texture synthesis, given that information needs to be reconstructed at the decoder.

The subset of the work presented in this chapter has been published in [116].

4.1 Review of spatial resolution adaptation frameworks

Adaptive resolution approaches can be classified based on the decisions made on three different aspects: the level at which it is applied, the adaptation algorithm and the scaling method used. Resolution adaptation can be employed at various stages within the encoding process. In [111, 110, 115], downsampling is added as a new mode at the macroblock or CTU levels depending on the host codec used. The RDO process is modified to include a new resolution adaptation mode which is compared against other modes at the original resolution. Alternatively, in [117, 113, 112, 114] downsampling is performed as a pre-processing step prior to encoding, and applied on inter frames only [112], at the GOP level [117] and at the sequence level [113, 114].

The methods used for downsampling and upsampling are crucial for providing high quality reconstruction of the video content. Simple interpolation filters, such as bicubic, are commonly

used due to their high efficiency [112, 117], while more computationally intensive super-resolution techniques [112, 114, 115] are able to provide improved reconstruction quality.

Previous work on this topic most often assumed that resolution adaptation is only applicable at lower bitrates [112, 114]. This assumption neglects the content-dependent nature of the input video, which limits the applicability of these methods to video compression. Other works attempt to overcome this problem by adaptively changing the resolution depending on the rate-distortion (R-D) operating conditions. In [117, 113], the optimal resolution is determined by minimizing the distortion at a given bitrate by decomposing the overall distortion into a downsampling distortion and a coding distortion. This assumes that both distortions are independent, which according to [117] is a reasonable consideration. However, modelling the rate-distortion performance at different bitrates requires pre-encodings or modelling of the encoder which is a difficult task and usually incurs high computational complexity. Finally, adaptive resolution methods have also been applied in the context of Scalable Video Coding (SVC) [118] as well as rate control [119]. Table 4.1 summarizes the most important characteristics of a number of previous works on this topic.

Table 4.1 Summary of previous works in the field of resolution adaptation for video coding.

Ref.	**Anchor codec**	**Downsampling level**	**Adaptation algorithm**	**Scaling method**
[110]	MPEG-2	Macroblock (16x16 pixels)	RD performance at the macroblock level is used to determine the best mode	Algorithm for scaling images in the compressed domain, DCT
[111]	H.264	Macroblock (16x16 pixels)	RD performance at the macroblock level is used to determine the best mode	Custom filter design
[112]	H.264	Frame (inter-frames only)	N/A	Example-based super-resolution algorithm
[117]	H.264	GOP	Model for downsampling distortion and coding distortion	Bicubic interpolation
[113]	H.264	Sequence	Model for downsampling distortion and coding distortion	Filter design
[114]	H.264 and H.265 (HEVC)	Sequence	N/A	Custom, low complexity, super-resolution algorithm
[115]	H.265, HEVC	CTU	RD performance at the macroblock level is used to determine the best mode	CNN-based super-resolution algorithm

As discussed before, resolution adaptation has been studied in past related work. However, for most works, older codecs such as H.264 were mainly used as the host codecs and only sequences of lower spatial resolutions were considered. Therefore, we believe its full potential has not been fully achieved, especially for high definition video content and modern video codecs such as HEVC. The main challenges include reliably switching between spatial resolutions and developing algorithms for accurately upsampling the video after decoding while keeping a low complexity of the overall system.

4.2 Proposed adaptive resolution framework

In this section, we propose an efficient spatial resolution adaptation approach for intra coding. Our system dynamically applies downsampling to frames exhibiting low spatial detail delivering improved rate distortion performance, together with a reduction in computational complexity of the encoding process. An experimental investigation into the relationship between the QP threshold, which determines when to encode lower resolution frames, and the distortion obtained after downsampling and upsampling allows the development of a fast, optimal resolution prediction model.

The proposed adaptive resolution coding scheme is illustrated in Figure 4.1, and includes additional steps at both the encoder and the decoder. First, each frame of the original (uncompressed) input video is downsampled to a lower resolution (I_d) and then upsampled to its original resolution (I_u). The distortion, as measured by the PSNR, of the upsampled frame with respect to the original resolution is then computed. Scaling is performed by applying Lanczos filtering for downsampling and upsampling of the video frames. A discussion on scaling filters can be found in Section 4.2.1. A single downsampling ratio or factor of 2 is considered in this work. This ratio has been shown in previous work [120] to generate better performance for video coding compared to other downsampling ratios.

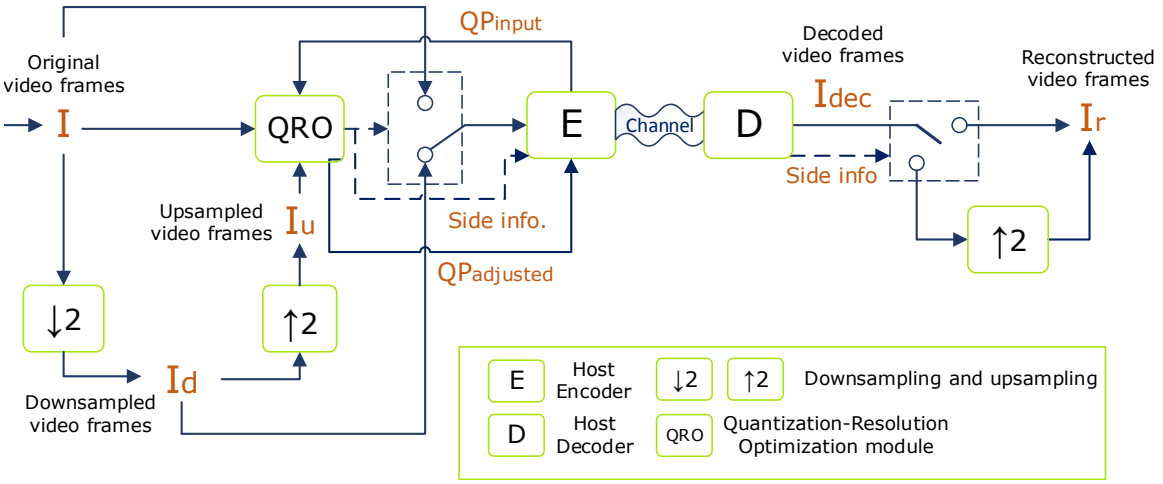


Figure. 4.1 Diagram of the proposed adaptive resolution framework.

The PSNR value, along with the input base QP, are used by the Quantization-Resolution Optimization (QRO) module, which is responsible for determining the optimal resolution to encode the input video frame. If the lower resolution is selected, the downsampled frame (I_d) is sent to the host encoder (E), and a flag bit is added to the bitstream, for each frame, to signal the resolution decision. In addition, the QP used to encode the low resolution frame is adjusted from the input base QP to reflect the resolution change. On the other hand, if the QRO makes the decision not to apply resampling, the original resolution frame is encoded and the QP remains unchanged. Encoding the video frames at a lower resolution frames requires less bitrate than encoding at the original resolution. However, the quality of the decoded frame is reduced due to the resampling process.

At the decoder (D), the resolution decision is extracted from the bitstream. If the decision was to encode a low resolution frame, the decoded frame is upsampled using the same scaling method applied at the encoder to reconstruct the full resolution frame (I_r). This produces a frame with the same resolution as the input video frame.

This framework allows for different resolution decisions across temporally adjacent video frames e.g one frame can be encoded at a lower resolution, while the next frame can be encoded with the original resolution. Since this approach is proposed for intra coding only, which means each frame is encoded independently, having frames with different resolutions does not require any additional modification to the video coding framework. The only side information necessary is the frame-level flag bit, which corresponds to a negligible overhead compared to the size of the encoded frames.

The performance of the proposed coding scheme depends on two key modules: (i) the algorithm employed for image scaling, and (ii) the QRO module. These are described in detail in the following sections.

4.2.1 Image scaling overview

Image scaling refers to the spatial resizing of a digital image. Two types of scaling can be applied: downscaling (or downsampling) and upscaling (or upsampling). Downsampling refers to the process of decreasing the image resolution in one or both dimensions using a defined sampling factor. In contrast, upsampling increases the spatial resolution of the image by interpolating the missing information. Note that the scaling factors can take fractional values (e.g. 1.5×). Downsampling generally introduces a loss of high frequency information since it reduces the number of pixels of the original image. For this reason, spatial aliasing, which leads to undesirable artifacts, can occur [121]. Therefore, an anti-aliasing filter, that aims at removing selected high frequency components before downsampling is generally employed. Upsampling can be realised using various interpolation filters or kernels or through super-resolution algorithms. The latter generally produces more accurate reconstructions at the expense of higher computational complexity. In this section, a brief overview of popular upsampling filters is presented, while super-resolution algorithms will be addressed in Chapter 5.

Several linear filters have been developed for the scaling of digital images, including: nearest neighbour, bilinear, bicubic and Lanczos. These are all linear separable filters [121] meaning that they apply linear combinations of adjacent pixels and are applied separately to each dimension in a 2-D image. Given that a digital image is a discrete point-sampling of a continuous function, the idea of interpolation is to approximate the continuous function and sample from it to determine the intermediate values. Figure 4.2 illustrates an example of a 1-D interpolation of grey level values using three interpolation filters: nearest neighbour, linear and cubic.

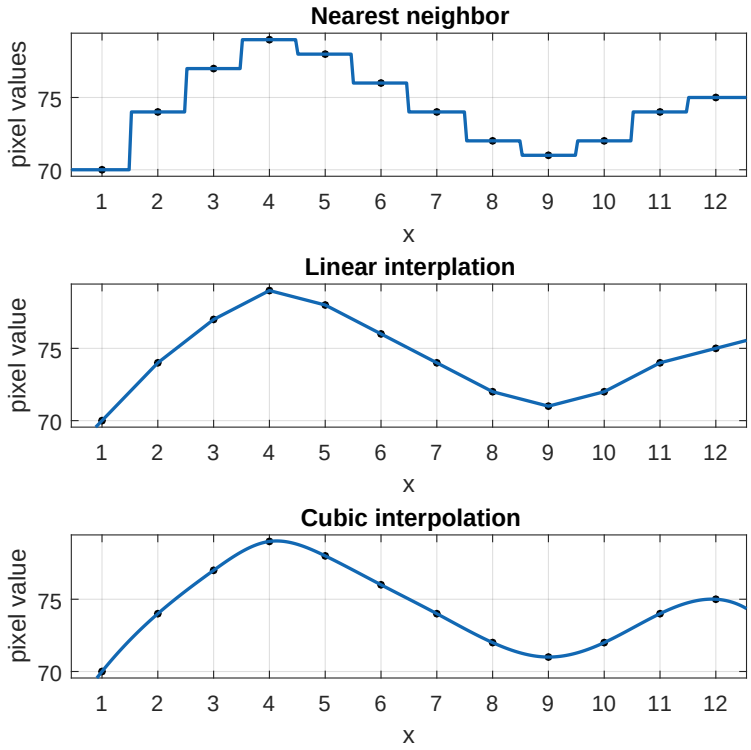


Figure. 4.2 Illustration of 1-D nearest neighbour, linear and cubic interpolations.

Nearest neighbour is the simplest form of interpolation and consists of assigning to the interpolated pixel the value of the closest pixel from the original image. Bilinear interpolation uses a weighted average of the closest 2 pixels to linearly interpolate each missing pixel. In contrast, bicubic interpolation uses a cubic approximation considering the neighbouring 4 pixels at each dimension when calculating the unknown pixel value. Bicubic interpolation has been a popular choice for rescaling digital images.

Another filter, the Lanczos resampling filter, uses the Lanczos kernel [122], which is a sinc function windowed by a horizontally stretched sinc function. The Lanczos filter ($L_(x)$) is given by:

$$L(x) = \begin{cases} \text{sinc}(x)\text{sinc}(\frac{x}{a}), & \text{if } -a < x < a \\ 0, & \text{otherwise.} \end{cases} \tag{4.1}$$

where parameter a determines the size of the kernel and typical values correspond to 2 or 3. The Lanczos filter is a finitely supported approximation of the sinc filter, which is the theoretical optimal interpolation filter [121]. In our experiments, we have empirically found that $a = 3$ provides the best reconstruction performance (measured by PSNR) for image resampling. In the remaining sections and chapters of this thesis, we denote Lanczos filter with kernel size 3 as simply Lanczos3.

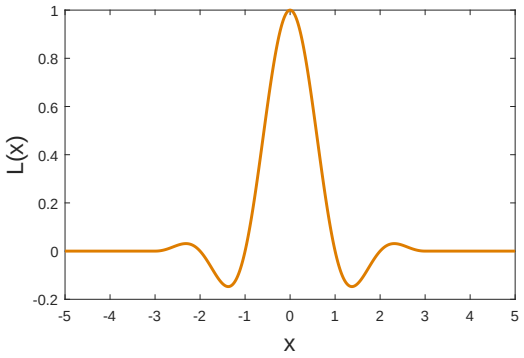


Figure. 4.3 Lanczos filter function for $a = 3$.

In order to illustrate and compare the reconstruction performance of the filtering methods previously described, Figure 4.4 shows a patch of a video frame from the *BQTerrace* sequence [10], which was downsampled and upsampled using a downsampling ratio of 2 and the following methods: bilinear, bicubic and Lanczos3. The original patch is also included for reference. In addition, the PSNR value between the reconstructed and the original patches is also presented below each image. It is clear from the figures that the worst performing filter is bilinear, which produces a blurry result, which is a consequence of the averaging of the closest pixel values. In contrast, Lanczos3 provides the best reconstruction quality with sharper edges.

In order to further assess the reconstruction quality of the image scaling methods described (bilinear, bicubic and Lanczos3), their performance was evaluated on a dataset of 12 HD (1920×1080) test sequences, listed in Table 4.3. For each method, the same filter was used for both upsampling and downsampling with a ratio of 2. No compression was employed and the PSNR (Y channel only) between the original and the downsampled and upsampled video was computed. The result of this comparison is presented in Table 4.2.

In addition to providing the reconstructed quality based on PSNR, the complexity of the methods was also evaluated and is shown in Table 4.2. The complexity figures are estimated based on their average execution time of the combination of downsampling and upsampling, obtained on an Intel Core i7-4790 CPU@3.60GHz PC. Additionally, in order to compare the relative time of the scaling algorithms with the time to encode a frame using intra mode, the encoding time of the HEVC reference codec (HM 16.2, All Intra configuration [10]) is also presented. This value was obtained by averaging the encoding execution time of the first frame of each sequence using several QP values {22, 27, 32,

(a) Original (b) Bilinear [PSNR = 22.3 dB]

(c) Bicubic [PSNR = 24.2 dB] (d) Lanczos3 [PSNR = 25.0 dB]

Figure. 4.4 Visual quality comparison of a patch of the first frame of the BQTerrace sequence interpolated by three different methods: (b) bilinear, (c) bicubic and (d) Lanczos3. The original is shown in (a).

37 and 42}. The goal of this analysis is to understand if the addition of scaling algorithms at the encoder would compromise the overall encoding time.

Table 4.2 Average PSNR and relative execution time of the three interpolation methods tested for 12 HD test sequences.

	Bilinear	**Bicubic**	**Lanczos3**	**HEVC Intra Coding**
PSNR (dB)	29.5	31.7	32.5	-
Relative time	1.0	1.36	1.72	1025

It can be observed from Table 4.2 that the Lanczos3 filter offers the best reconstruction accuracy among of all approaches tested, with an average PSNR value of 32.5 dB, which is a 0.8 dB PSNR gain over bicubic. In addition, the complexity of the Lanczos3 filter is only 26% greater than bicubic. When compared to HEVC intra coding, the resampling time of all methods is several orders of magnitude lower, which means that no significant overhead would be introduced by adding this rescaling step at the encoder.

Based on the previous analysis, it was determined that Lanczos3 provides a good trade-off between reconstruction quality and complexity. Therefore, we propose to apply Lanczos3 as the scaling filter for both downsampling and upsampling in our proposed resolution adaptation framework for intra coding. This filter will be used both at the encoder and decoder.

4.2.2 Predicting the optimal resolution decisions

As discussed earlier, the QRO module is responsible for determining which frames to downsample based on the resampling quality of the input video frames. Previous works [114, 110] have reported that there is generally a bitrate/QP value at which the R-D performance of encoding lower resolution frames and upsampling at the decoder just outperforms encoding full-size frames. This value is referred to as the QP threshold.

To this end, in order to understand how the QP threshold varies with the video content, an experiment using 50 randomly selected sequences from the HomTex dataset [69] was conducted. This dataset, presented in Chapter 3, contains a variety of textures which are an important component of natural videos. In this experiment, each sequence was encoded using HEVC reference software, HM 16.2, All Intra configuration (Main profile), at the original resolution and at half the original resolution. The reconstructed videos were then upsampled back to the original resolution and the video quality was evaluated using PSNR. As mentioned in the previous section, the Lanczos3 filter was used to perform the scaling of the video frames. In order to account for variability and to accurately estimate the QP thresholds, a large number of QP values was tested, more specifically all integer values from 20 to 50. For each sequence, 10 equally sampled frames were encoded equating a total of 500 frames. Finally, the RD curves of the original and lower resolution frames were plotted and the QP thresholds estimated based on the identification of the crossover points between the curves.

For instance, Figure 4.5 presents the RD curves obtained for a frame of the *CarpetPanAverage* and the *ReflectionWater* sequences from HomTex. Both show that, for low QP values, the RD peformance is higher when encoding the original resolution frame, while for high QP values, encoding at a lower resolution is beneficial. However, the QP value at the crossover, the QP threshold, is different between sequences. For *CarpetPanAverage*, the QP threshold is around 28 and for *ReflectionWater*, it is roughly 36. By observing the RD curves for all sequences in the training dataset, it is observed that the value of the QP thresholds are highly content dependent. Nevertheless, if these thresholds can be reliably estimated, significant compression gains could be achieved compared to encoding at a fixed resolution, as seen by the curves.

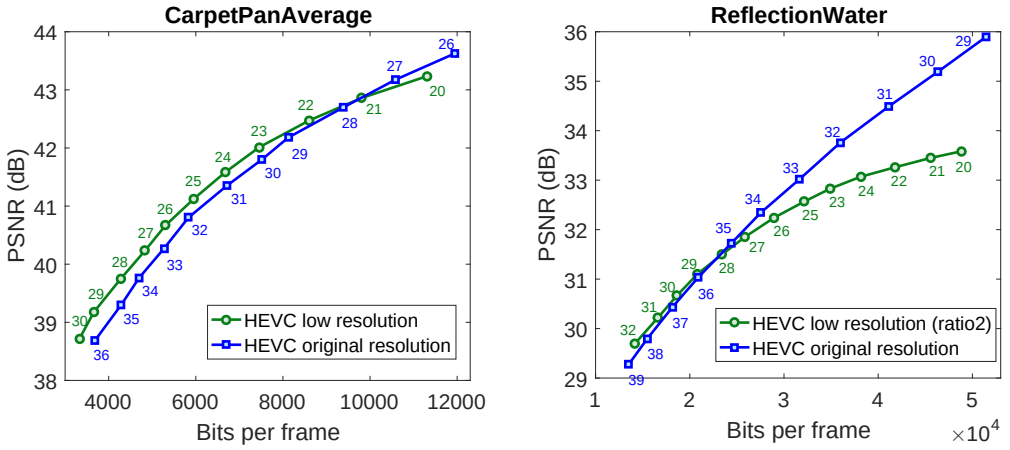


Figure. 4.5 RD curves obtained by encoding the original and lower spatial resolution versions of a frame of the *CarpetPanAverage* (left) and ReflectionWater (right) sequences from HomTex using HEVC intra coding.

By analysing the data collected from the experiments, it was observed that there is a high correlation between the QP thresholds and the resampling PSNR, the PSNR value between the uncompressed frame scaled to the lower resolution and back up to the original resolution and the original frame. Figure 4.6 shows a scatter plot of this correlation where every data point represents a different frame (from a total of 500 frames). The Pearson Correlation Coefficient (PCC), which is a measure of correlation and ranges from 0 to 1 is 0.86 (absolute value) for this data. Following the shape of the data points, an exponential fitting was applied (green curve in Figure 4.6). The fitted curve is represented by the following equation:

$$\mathrm{QP}_{\mathrm{thres}}(\mathrm{PSNR}_r) = 10^{\alpha+\beta q} + K \tag{4.2}$$

where $\mathrm{QP}_{\mathrm{thres}}$ is the QP threshold, PSNR_r is the resampling PSNR, expressed in dB, $\alpha = 2.05$ and $\beta = -0.014$ are the fitted parameters and K is an optional constant. The goodness of fit, evaluated by the Mean Absolute Error is 3.46 QP values, which means that the average difference between the fitted QP threshold is and the "true" QP thresholds is 3.46. Although not very small, we consider this

error to be acceptable for our experiments, especially considering that this is a very simple estimation based on a single feature, the $PSNR_r$.

In this work, in order to minimize the error for cases in which the fitted QP threshold is lower than the "true" QP threshold, which means that resolution adaptation would be applied when it is not beneficial, $K = 2$ was used (red curve in Figure 4.6). This offset is able to reduce these cases by approximately 30%, while still maintaining an overall accurate model.

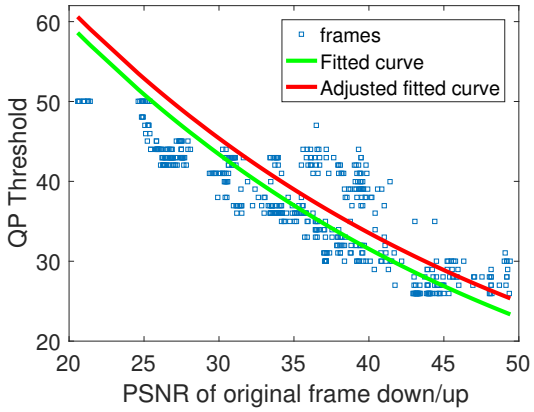


Figure. 4.6 Relationship between the PSNR of the downsampled frame and the QP threshold.

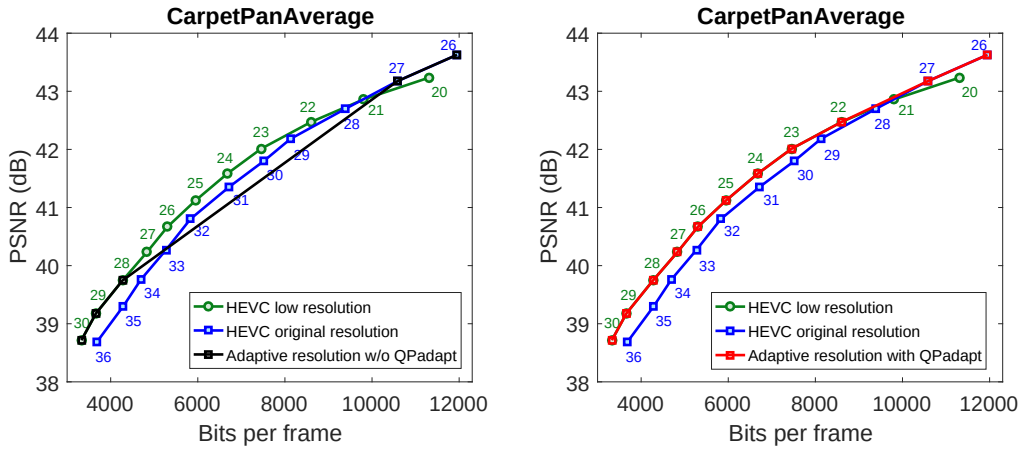
Based on this fitting model and assuming that it generalizes for videos with other spatial resolutions, it is possible to obtain a prediction of the QP_{thres} for a given frame based on the value of $PSNR_r$ and the fitted curve (Equation 4.2). This assumption will be tested in Section 4.2.3, where the proposed coding scheme is applied to general HD and UHD resolution test videos.

Combining the above points, the QRO module is able to determine the optimal downsampling ratio based on the predicted QP threshold and the input base QP, given by the following equation:

$$\text{Optimal downsampling ratio} = \begin{cases} 1, & \text{if } QP_{input} < QP_{thres} \\ 2, & \text{if } QP_{input} \geq QP_{thres}. \end{cases} \tag{4.3}$$

Once the value of QP_{thres} is exceeded, and low resolution video frames are encoded, the bitrate is significantly reduced. For this reason, in order to achieve a similar bitrate as would have been achieved by encoding full resolution video frames, the QP value for which to encode the low resolution frames is adjusted. Based on an analysis performed on the training dataset, we have found that, at the crossover points, the difference between the input base QP of the original resolution and the lower resolutions is approximately 6 (QP units). This agrees with the analysis in [123]. Therefore, based on the input QP value, QP_{input}, the adjusted $QP_{adjusted}$, used to encode the low resolution video frames, is given by:

$$QP_{adjusted} = \begin{cases} QP_{input}, & \text{if } QP_{input} < QP_{thres} \\ QP_{input} - 6, & \text{if } QP_{input} \geq QP_{thres}. \end{cases} \tag{4.4}$$



(a) (b)

Figure. 4.7 Illustration of the QP adjustment process: (a) RD curves obtained by encoding the original, the lower spatial resolution versions of a frame of the *CarperPanAverage* (left) and ReflectionWater (b) sequences from HomTex using HEVC intra coding.

This adjustment guarantees a smooth transition between encoding at the original resolution and the lower resolution. This effect is illustrated in Figure 4.7 which shows RD performance of a frame of *CarpetPanAverage* when applying the proposed resolution adaptation algorithm with and without the QP adaptation. It can be seen that after QP 27, low resolution frames are encoded, which significantly decreases the bitrate, creating a bitrate gap between QP 27 and QP 28 (Figure 4.7 (b)). This gap can be minimized with the proposed QP adjustment (Figure 4.7 (a)).

4.2.3 Experimental results

The proposed resolution adaptation approach has been integrated into the HEVC reference codec (HM 16.2), and was fully tested using the All-Intra configuration (main profile) [10] on the first 300 frames (when available) of 12 HD and 3 UHD test sequences. Commonly used QP values from 22 to 42 with a interval of 5 were tested [10].

All test sequences were obtained from public video databases, namely, the BVI-HFR database [9], the HEVC Common Test Conditions (CTC) test set [10], the Tampere 4K test dataset [11], and the Derf's Test Media Collection [12]. Table 4.3 lists all the test sequences and their sources, while Figure 4.8 shows an example frame of each test sequence. This particular set of test sequences was selected to provide content with varying levels of spatial detail, which could be found in real world applications, ranging from very textured to more homogeneous or blurry content. These were classified into three groups according to our measure of spatial resampling distortion, the resampling PSNR, $PSNR_{res}$: low (Group I), medium (Group II) and high distortion (Group III). This classification can also be found in Table 4.3.

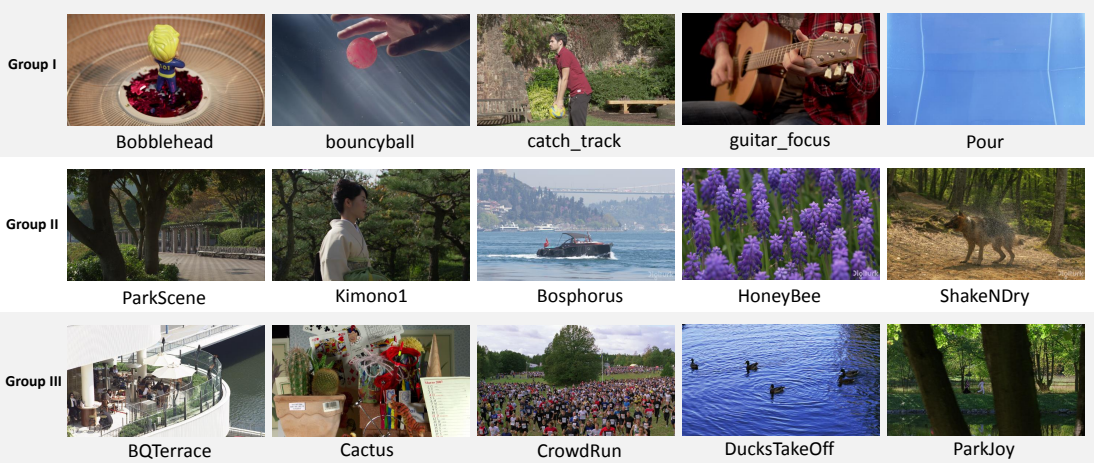


Figure. 4.8 Sample frame from the 15 test sequences used for this experiment.

Another important aspect to note is that there is no overlap between the dataset used to train the QP threshold prediction model (QRO) and this test dataset. This allows us to test the generalization of the proposed coding scheme.

The compression performance of the proposed spatial resolution adaptation approach was compared with HEVC encoding at a fixed resolution (the anchor). The results, based on BD-rate measurements [124] across all frames, are shown in the last column of Table 4.3. It can be observed that the proposed method always provides better or nearly equivalent performance against the anchor for all the sequences tested. On average, the proposed scheme achieves bitrate savings of 3.9% over the whole bitrate range (all QP values tested). As expected, this improvement is more significant on sequences with lower spatial detail, 7.9 % for group I, than on those with higher spatial detail, 0.03 % for group III. It was also observed that higher bitrate savings are usually achieved for lower bitrates, where it is more likely that the QP would exceed the predicted QP thresholds and, therefore, lower resolution frames would be encoded.

Additionally, Table 4.3 also presents the encoding time reduction of the proposed framework. Since applying resolution adapation allows the encoding of low resolution frames, when beneficial, encoding execution time is significantly reduce compared to fixed resolution encoding. These reported savings were computed as the complexity reduction averaged across the 5 rate-points (at different QP values). For frames encoded at lower spatial resolution, the encoding time is reduced by an average of 53%. However, at rate-points where the input QP values are close to the predicted QP thresholds, some frames might be encoded at a lower resolution while others are encoded at the original resolution. It was also found that the added complexity introduced by applying the scaling process is insignificant compared to the overall encoding time, so no overhead is incurred by performing the scaling at the encoder.

Finally, Table 4.3 also reports the predicted QP thresholds as computed by the proposed RQO model. For the sequences with lower resampling distortion (group I), the predicted threshold QP value

Table 4.3 Results of the proposed adaptive resolution coding scheme against HEVC using several test sequences divided into three groups (G). BVI-H: Bristol Vision Institute High Frame Rate dataset [9], HEVC: test sequences from the Commom Test Conditions [10], T4K: Tampere 4K test sequences dataset [11], Xiph: Derf's Test Media Collection [12], Res.: Spatial resolution, $PSNR_{res}$: Resampling PSNR, QP thres.: Predicted QP threshold, T: Total execution time difference, R: BD-Rate.

G	Sequence	Dataset	Res.	$PSNR_{res}$	QP thres.	T (%)	R (%)
I	bobblehead	BVI-H	HD	51.7	23.7	-56.0	-12.5
	bouncyball	BVI-H	HD	48.4	26.2	-57.4	-7.1
	catch_track	BVI-H	HD	46.5	27.7	-45.5	-3.7
	guitar_focus	BVI-H	HD	50.3	24.7	-55.3	-7.2
	pour	BVI-H	HD	25.5	49.3	-57.2	-7.8
	Average			**49.2**	**25.5**	**-54.3**	**-7.6**
II	ParkScene	HEVC	HD	36.3	37.5	-17.2	-0.5
	Kimono1	HEVC	HD	42.4	31.2	-32.6	-2.4
	Bosphorus	T4K	UHD	45.9	28.1	-31.0	-5.3
	HoneyBee	T4K	UHD	41.4	32.2	-30.4	-5.3
	ShakeNDry	T4K	UHD	41.5	32.1	-28.8	-6.9
	Average			**41.5**	**30.8**	**-28.0**	**-4.1**
III	BQTerrace	HEVC	HD	33.2	41.3	-8.2	0.1
	Cactus	HEVC	HD	34.9	39.2	-10.4	0.0
	CrowdRun	Xiph	HD	30.8	44.3	-3.9	0.1
	DucksTakeOff	Xiph	HD	30.9	44.1	-7.2	-0.3
	ParkJoy	Xiph	HD	29.1	46.7	-0.3	0.1
	Average			**31.8**	**43.1**	**-6.0**	**-0.03**
	Overall average			**40.8**	**33.1**	**-29.4**	**-3.9**

is on average 25.5, while for the test sequences with higher spatial detail (group III), the average QP threshold is 43.1. This indicates that the benefit of the proposed resolution adaptation approach is higher for video content exhibiting less spatial detail.

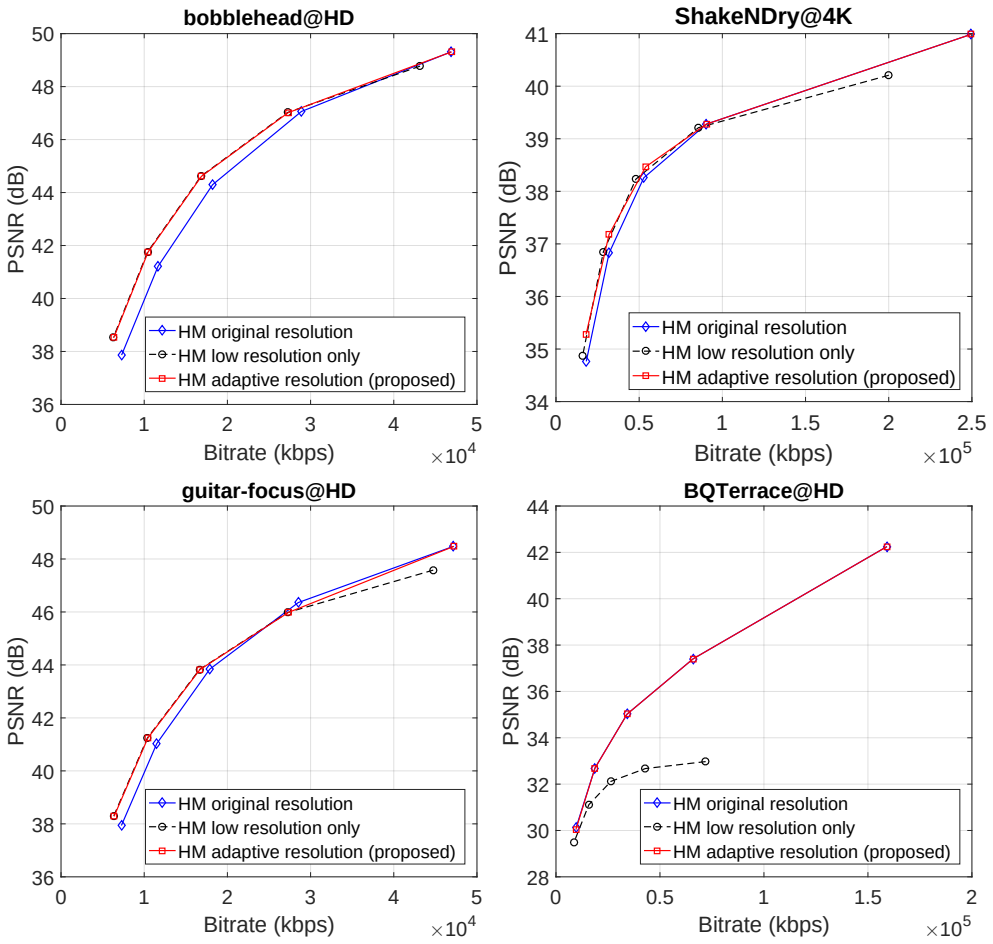


Figure. 4.9 R-D curves of the proposed coding scheme and the anchor, HEVC, for four sequences tested.

Figure 4.9 presents the RD performance of the proposed method compared to the HEVC anchor for a subset of the test sequences. The figures also include the performance of HEVC when encoding at fixed lower resolution (using a downsampling ratio of 2 for all frames). For instance, for *bobblehead* (Group I), both approaches have equal performance for the first QP value, QP = 22. This is because the predicted QP threshold is higher (23.7) than the input QP and therefore, no resolution change is made to the video frames. However, for higher QP values, the proposed approach significantly outperforms HEVC. A similar pattern is shown in *ShakeNDry* (Group II), except that for this sequence, the QP threshold was predicted as a higher value. In the case of *guitar_focus* (Group I), the proposed adaptive resolution provides a slightly lower RD performance compared to HEVC for the second QP value, QP = 27, which reflects an inaccurate prediction of the QP threshold. Nonetheless, for the

whole range of QP values, significant gains are still achieved (see Table 4.3). Lastly, for the sequence *BQTerrace* (Group III), the two R-D curves overlap for all QP values. This suggests that the prediction model is able to make conservative decisions at higher bitrates and sequences with high spatial detail.

By analysing the difference between the fixed low resolution encoding (dashed curve) and the proposed method, it can be seen that the proposed coding scheme is able to reliably switch between resolutions in order to achieve the optimal reconstruction quality. Our approach has shown to be not only applicable at lower bitrates, but also at higher bitrates due to the use of an adaptation method.

4.3 Discussion

This chapter has presented a resolution adaptation approach for HEVC intra coding, which adaptively enables the downsampling of input frames prior to encoding and reconstructs to the original resolution after decoding. The determination of the optimal resolution is based on the observation of the relationship between the quality after downsampling/upsampling and the threshold QP at which it becomes beneficial to encode lower resolution frames. This approach shares similar characteristics with a texture analysis and synthesis coding framework, discussed in Chapter 3, due to the fact that information is discarded at the encoder and reconstructed at the decoder side using synthesis/interpolation algorithms. When integrated with HEVC intra coding, the proposed scheme achieves promising BD-rate savings, on average 3.9% compared to fixed resolution encoding. In addition, the proposed coding scheme provides a reduction in computational complexity at the encoder side by an average of 29%.

The high correlation between the downsampling quality and the QP thresholds has only been found for intra coding, as mentioned in Section 4.2.2. Therefore, the scope of this work is limited. Chapter 5 will explore resolution adaptation for the more general case of video coding using temporal prediction. Moreover, it will also explore the use of more advanced scaling algorithms to improve the overall video quality.

Chapter 5

High quality spatial resolution adaptation using super-resolution

In the last chapter, we presented introductory work on spatial resolution adaptation for video coding. A low complexity framework was presented which is able to provide rate-distortion gains by dynamically adapting the resolution of the input video frames prior to encoding and reconstructing the full resolution at the decoder. In this chapter, we present several improvements and extensions to this work including the use of a super-resolution technique based on deep learning to provide higher upscaling quality. We first present some background and related work in the field of super-resolution and on the use of machine-learning techniques (neural networks and deep learning) for video coding. We then describe our proposed modified coding scheme and present detailed experimental results to validate its performance.

A portion of the work presented in this chapter has been published in [125] and [47].

5.1 Background and related work

5.1.1 Review of super-resolution algorithms

Super-resolution is the process of obtaining one or more high resolution (HR) images from one or more low resolution (LR) images. This topic has been studied extensively for several decades and has found applications in a variety of fields, including satellite and aerial imaging, medical image analysis, facial and textual image improvement and compressed image/video enhancement [126]. For example, consider the images in Figure 5.1. The left image is the LR version and is composed of a collection of four sub-images. The goal of super-resolution is to obtain an HR image (right) from the LR sample. In this example, the upsampling factor is 4, which means that the HR image contains 16 times as many pixels compared to the LR image. Due to the fact that the same LR version could have been obtained from several HR images, super-resolution is an ill-posed inverse problem, allowing for multiple plausible solutions.

Figure. 5.1 Example of super-resolution using an upscaling factor of 4 times (4x).

Depending on the number of LR images used to recover one HR image, super-resolution algorithms are classified as single-image super-resolution, if only one LR image is provided, or multiple-image super-resolution, if several LR images are used [126]. Typically, multiple LR images are captured from the same scene using different angles or positions, which helps to constrain the inverse problem and achieve a more accurate solution.

The most popular single-image super-resolution algorithms generally employ a machine-learning-based techniques to hallucinate the missing high frequency information using the relationship between pairs of LR and HR images obtained from a training dataset. Several techniques have been proposed based on dictionary learning [127, 128, 130], self-similarity [129] and, more recently Convolutional Neural Networks (CNN) [5, 6].

Dictionary-based approaches use external image databases of HR and LR examples to train a dictionary consisting of image patches that form an efficient representation of natural images. The reconstruction of the HR image is performed by subdividing the test LR image into overlapping patches and searching for the nearest neighbours of each patch in the trained dictionary. This classic technique was later improved by applying techniques such as neighbour embedding [128, 130] and sparse coding [127].

In order to avoid the use of external training databases, which often need to contain an extensive number of training examples, self-similarity-based super-resolution algorithms were developed. These exploit the fact that patches in natural images tend to re-occur within and across different spatial resolution scales of the same image. In [129], this idea is extended by allowing geometric variations in the patch search.

Recently, inspired by the developments in the field of neural networks and deep learning, a CNN-based super-resolution method, SRCNN [5], was proposed to learn the mapping between LR and HR images. The proposed network is trained end-to-end using the LR image patches as the inputs to the CNN and to the corresponding HR patches as ground truth. Before passing through the network, the LR images are first upsampled using bicubic interpolation. The training is achieved by minimising

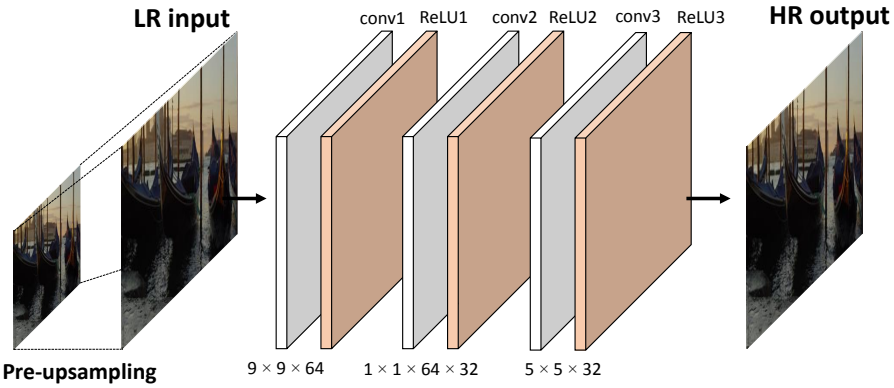


Figure. 5.2 Illustration of a network architecture of SRCNN [5].

the MSE between the output of the CNN and the original ground-truth HR image, which corresponds to the ℓ_2 loss. Stochastic Gradient Descent (SGD) and standard back-propagation is used to train the network's parameters. The network architecture of SRCNN is illustrated in Figure 5.2. A brief review of deep learning concepts can be found in Section 2.2.

Since the publication of SRCNN, several networks based on deep learning have been proposed for super-resolution. One work that achieved outstanding performance, Very Deep Super-Resolution (VDSR) [6] proposed a different network architecture inspired by advances in the field of image classification. It attempts to overcome some of the limitations of SRCNN, such as: (i) the use of small receptive fields (the region in the input image that influences the output); (ii) slow convergence; and (iii) the use of a single upsampling ratio for each trained network. VDSR, on the other hand, uses larger receptive fields, applies residual learning to boost the learning speed and accuracy by predicting the residual image instead of the HR image directly, and finally, defines a single convolutional network for multiple upsampling ratios. In addition, the number of convolutional layers is extended from 5 to 20, each followed by ReLu layers. Similarly to the case of SRCNN, ℓ_2 loss is minimised over the training database. The performance obtained by this network exceeded previous state-of-the-art methods in terms of reconstruction quality. Figure 5.3 presents the network architecture of VDSR.

5.1.2 Review of machine-learning-based video coding

Machine learning approaches have also recently gained traction with the image and video compression communities. Several methods have been proposed for a number of applications including compressed video super-resolution [131, 115, 132], compression artifact removal [133, 134], loop filtering [135] and intra coding [136].

In [131], super-resolution is exploited for compressed video content, using the same network architecture of SRCNN. Each video is first downsampled and then compressed using H.264 with different compression levels and the network is trained to reconstruct similar video frames as the high resolution versions. Before training on compressed videos, the network is pre-trained using a large

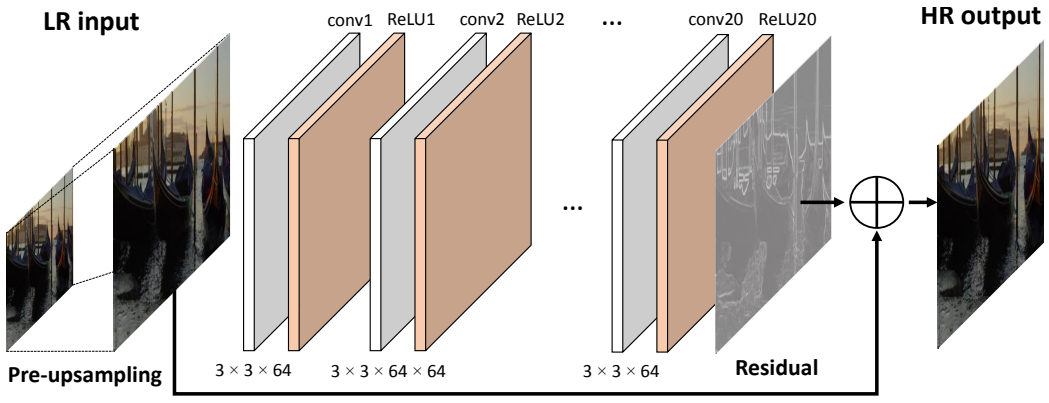


Figure. 5.3 Illustration of a network architecture of VDSR [6].

number of LR and HR image pairs from ImageNet [55]. Consecutive frames are used to reconstruct the same video frame, after the application of motion compensation. This approach achieves good reconstruction quality but uses a limited number of training and testing videos and its goal is not to enhance video compression efficiency.

In an attempt to integrate super-resolution using CNN into an adaptive resolution framework for video coding, [115] proposes a new mode which features downsampling of the CTU prior to encoding and upsampling after decoding. This mode is tested inside the encoding loop and rate-distortion optimisation (RDO) is used to access its benefits. Results show bitrate savings, especially for UHD content. This method has been applied to intra frames only and additional overhead is needed to signal the extra down/upsampling mode in the bitstream. In addition, the complexity is very high due to the added mode in the RDO process. Following a similar idea, in [132], a CNN-based resampling method at the CTU level, named CNN-BARC, is proposed. Again, this method is only applied to intra frames. One of the differences between this work and [115] is that a CNN is also trained to perform downsampling of the input CTUs.

Deep learning has also recently been applied to image denoising and compression artifacts reduction. One approach, DnCNN [134], proposes a feed-forward denoising CNN using residual learning and batch normalization. A single network has been trained to handle several general image denoising tasks such as Gaussian denoising, single image super-resolution and JPEG [77] image deblocking. Following this idea, in [5], a new CNN architecture for JPEG compression artifacts removal, AR-CNN [133], is proposed. Results show that the CNN is able to successfully reduce several types of compression artifacts including blocking.

The application of deep learning to image and video artifact reduction can be useful for the design of loop filters within the video compression systems. In [132], the loop filter in JEM [45] is replaced by a trained CNN and is applicable to all frame types. This network, called CNNLF, is applied after the deblocking filter and before the SAO filter [21]. Different parameters are trained for 3 levels of the quantization step size (controlled by the QP value), and the model with the closest QP compared

to the QP of the current frame is selected. The decision to use CNNLF is controlled by the RDO and is applied at the CTU level.

Finally, learning-based approaches have also been proposed to perform intra coding. In [136], two Neural Networks (NN) are used to predict the intra prediction samples and the optimal mode, based on conditional probabilities. The mode predicted controls the parameters of the second NN and needs to be signalled in the bitstream. The first NN contains 3 hidden layers, while the second contains only 1 hidden layer. This approach is able to achieve performance gains compared to conventional intra coding approaches.

5.2 Proposed framework

5.2.1 Overview

As introduced in Chapter 4, spatial resolution adaptation is a promising technique for video coding which can help improve compression efficiency. However, the approach proposed previously had a number of limitations and was developed primarily to achieve a low complexity solution. In the following sections, we address these limitations by presenting a number of improvements and extensions which significantly increase the applicability and performance of the approach. More specifically, the main differences are listed below:

- Resolution adaptation is applied to both intra frames and motion compensated frames, increasing the applicability of the proposed approach;
- A deep learning technique is used to produce high quality spatial reconstruction at the decoder;
- A large number of high resolution training sequences are used for the training;
- The number of downsampling ratios extended;
- Both spatial and temporal features are used to characterize the relationship between the video content and their respective QP threshold.

Instead of restricting the spatial resolution adaptation to intra frames only, we extend the previous work to include motion compensated frames using Random Access configuration in HEVC. Therefore, instead of making decisions for each input video frame, the proposed framework considers an entire shot.

Given the massive success of machine learning approaches for image reconstruction tasks, as covered in Section 5.1, we proposed to employ a CNN to perform the scaling of the decoded images. This CNN is trained using compressed video frames at different quantization levels. Compared to using linear filters, the CNN models helps to minimize both scaling and compression artifacts introduced by the encoding framework.

Our proposed strategy is based on two modules that require offline training: the Quantization Resolution Optimization (QRO) module, which is responsible for predicting the QP thresholds based on the content of the input video frames, and the deep learning model that reconstructs full resolution frames at the decoder. Therefore, in order to increase the generalization of the models trained, we have collected a large and varied dataset of training video sequences from openly available sources. This dataset consists of a total of 100 UHD resolution sequences. In addition to assisting the training process, this large dataset of helps us understand relationships between content and the performance of encoding videos at multiple resolutions, which is a topic that will be explored in more detail in Section 5.2.3.

The downsampling ratio controls the amount of scaling introduced in a spatial resolution adaptation framework. The majority of previous work consider only one ratio and this can limit the possible gains of such methods. In this work, we extend the number of downsampling ratios used in Chapter 4 from 1 to 2. This decision was based on the analysis of the performance of encoding the training sequences at multiple spatial resolutions. However, the addition of new downsampling ratios affects also the prediction of the QP thresholds since the number of QP thresholds needs to be equal to the number of downsampling ratios.

In our previous work, a single feature was used to predict the QP threshold, the resampling PSNR. This feature measures the distortion introduced by the downsampling/upsampling processes and was shown to have a strong correlation with the QP threshold. However, when applying the spatial adaptation to a more general use case, using motion compensated frames, we have found that both spatial and temporal features are required for an accurate prediction. Therefore, in this chapter, we propose a new prediction model that combines several spatio-temporal features.

5.2.2 Coding scheme

The proposed coding scheme, illustrated in Figure 5.4, can be used in conjunction with any host video codec. In the present work, HEVC reference software, HM 16.18, was used. The first module, the Quantization-Resolution Optimization (QRO), receives the original video frames and frames that were previously downsampled and upsampled using two different downsampling ratios, 1.5 and 2. Based on those, low-level spatio-temporal features are extracted. Next, a machine learning model, trained offline, predicts the optimal resolution of the input video frames among three possible choices: full resolution and downsampling ratios 1.5 and 2. This decision depends on the low-level features and on the input base Quantization Parameter (QP) from the host encoder. Signalling is performed using two bit flag, SR_{flag}, is used to represent the three different possible optimal resolutions, more specifically 0, 1 and 2 for full resolution and downsampling ratios of 1.5 and 2, respectively. The resolution adapted video frames are then sent to the host encoder. This module also produces an adjusted QP value and resolution flag bits which are added to the bitstream. The QRO module is described in detail in Section 5.3.

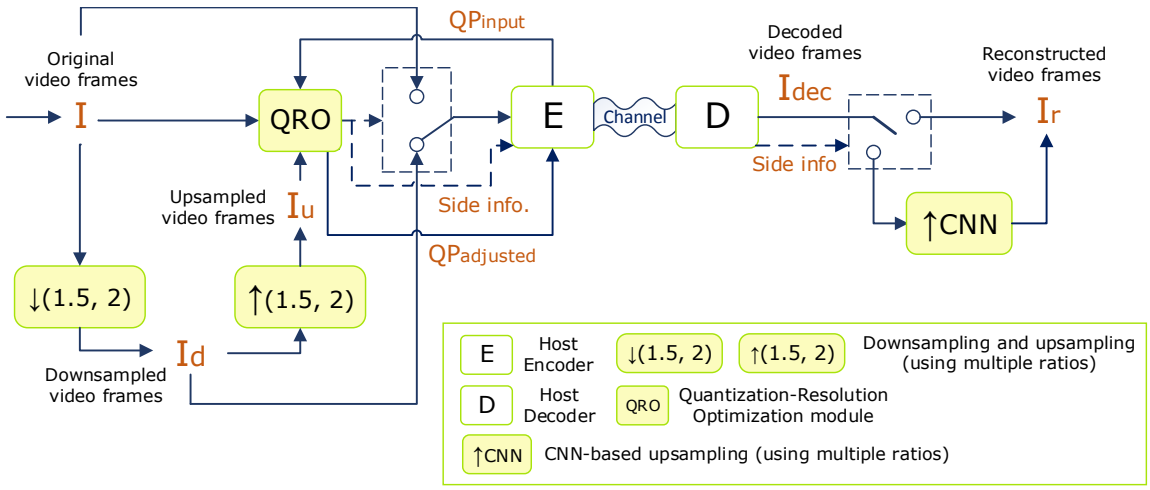


Figure. 5.4 Diagram of the proposed adaptive resolution framework. The modules highlighted have changed compared to the one proposed in Chapter 4.

Finally, the host decoder reconstructs the optimal resolution video frames, which are then sent to the CNN super-resolution module responsible for performing the scaling of the frames to full resolution. Moreover, multiple CNN models are employed for different downsampling ratios and base QP value ranges.

In the current version of our proposed coding framework, adaptation decisions are performed on a shot-level, since video characteristics are usually similar within the same shot. However, other options are also possible and would most likely provide higher compression efficiency. These would include performing the adaptation for each intra period or GOP. However, at the moment, in HEVC, having frames with multiple resolutions is not possible. For this reason, a look-ahead approach which would split the encoding into multiple intra period-size chunks and encode each using different resolution decisions would be required.

5.2.3 Training sequences

As mentioned above, we have used a large database of 100 openly available UHD sequences for the training process. The sequences were obtained from a variety of sources, more specifically, Netflix Chimera [49], Ultra Video Group [11], Harmonic Inc [48], SJTU [137] and AWS Elemental [138]. All the sequences are UHD resolution, contain 4:2:0 chroma subsampling and have been temporally cropped to the first 64 frames. The use of a reduced number of frames helps to lower the total encoding time required for all training sequences. In addition, each sequence contains a single scene (without scene cuts) and the majority of the test sequences have a frame rate of 60 fps and bit depth of 10 bits per sample. Further detailed specifications of all training sequences can be found in the Appendix C.1. To illustrate the variety of video content found in our training dataset, Figure 5.5 shows a sample frame of each training sequence.

Figure. 5.5 Sample frame of each of the 100 training sequences used for this work.

5.2.3.1 Content Characterization

In order to characterize the content of the training dataset in a way that can be compared with other video databases, low-level features that represent the amount of spatial information (SI) and the amount of temporal information (TI) or Motion Vectors (MV) are usually employed [139, 140]. The literature proposes several different ways to compute these features. In this work, we have calculated two popular pairs of features, namely SI/TI defined in the ITU recommendation [140] and SI/MV defined in [139]. Due to the fact that two definitions of SI are used, we have named them SI_1 and SI_2, respectively.

SI is a feature that estimates the edge density in a video. Videos that are sharper and contain fine details or textured content are expected to have a higher SI value. SI is usually calculated by convolving the luminance channel of each video frame at time t using a Sobel filter (S_x and S_y), calculating the edge magnitude at each pixel value at each spatial position (x, y), and pooling over all frames in the video. According to [140], SI is computed as follows:

$$S_{mag} = \sqrt{S_h^2(x,y) + S_w^2(x,y)} \tag{5.1}$$

$$SI_1 = \max_t\{\mathrm{std}_{x,y}\{S_{mag}\}\} \tag{5.2}$$

where the standard deviation of the edge magnitude is calculated for each frame and the maximum among all frames is used as the SI measure.

In [139], SI is computed differently. Instead of using the standard deviation, it uses the average edge magnitude for each frame and averages over all frames, as shown by the following equation:

$$\mathrm{SI}_2 = \sum_t \sqrt{\frac{L}{1080}} \sqrt{\sum_{x,y} \frac{S_{mag}}{P}} \tag{5.3}$$

where P is the number of total number of pixels in a frame and L is the number of lines (equivalent to the height) of the video. The first term in the equation is intended to provide a normalization of the SI for multiple resolutions.

TI, on the other hand, measures the amount of motion energy in a video. Scenes with fast motion characteristics tend to have a higher TI value, while more static videos have lower values of this metric. One way to estimate the motion energy is to calculate the pixel-wise intensity differences between adjacent frames in time, known as the frame different (FD). In [140], TI is calculated using the standard deviation of the FD within a frame and the maximum among all frames. This calculation is shown in the following equations:

$$\mathrm{FD}_t = |F(i,j)_{t+1} - F(i,j)_t| \tag{5.4}$$

$$\mathrm{TI} = \max_t\{\mathrm{std}_{(x,y)}\{\mathrm{FD}_t\}\} \tag{5.5}$$

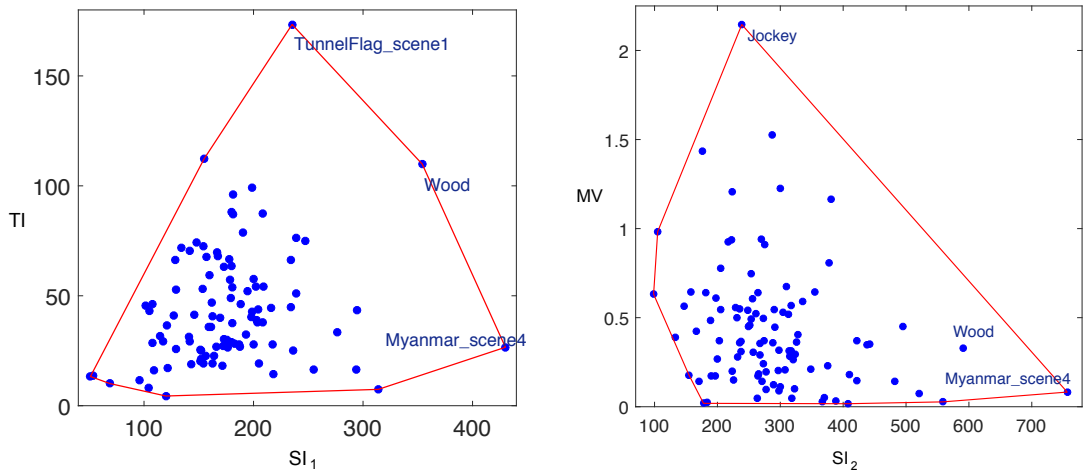
However, this feature is correlated with the amount of spatial information and texture characteristics [139]. Therefore, in [139], an alternative TI feature, based on the calculation of block-based Motion Vectors (MV), is proposed. In our experiments, we have used the Matlab implementation of the Three-step block matching algorithm [141] with 64×64 blocks, with a maximum displacement of 64 pixels. After the computation of the motion vectors for each pair of consecutive video frames ($\mathbf{v}$), the MV features are calculated by averaging and calculating the root of the squared magnitude of the motion vectors ($\mathbf{v}_{smag}$) for each frame and finally averaging the results for all frames, as follows:

$$\mathbf{v}_{smag} = \mathbf{v}_h^2(x,y) + \mathbf{v}_w^2(x,y) \tag{5.6}$$

$$\mathrm{MV} = \frac{f}{L} \sqrt{\sum_{x,y,t} \frac{\mathrm{v}_{smag}}{M}} \tag{5.7}$$

where M is the total number of motion vectors across frames and f corresponds to the frame rate of the input video. The first term provides a normalization of the feature across different resolutions and frame rates.

We have computed these four features (SI_1, SI_2, TI and MV) for all training video sequences and the results of SI/TI based on [140] and SI/MV based on [139] can be found in Figure 5.6. Both show reasonable coverage of the spatial and temporal information space. However, for some of the sequences highlighted in the figures, the feature values deviate significantly from the dataset average



(a) SI/TI according to [140]

(b) SI/MV according to [139]

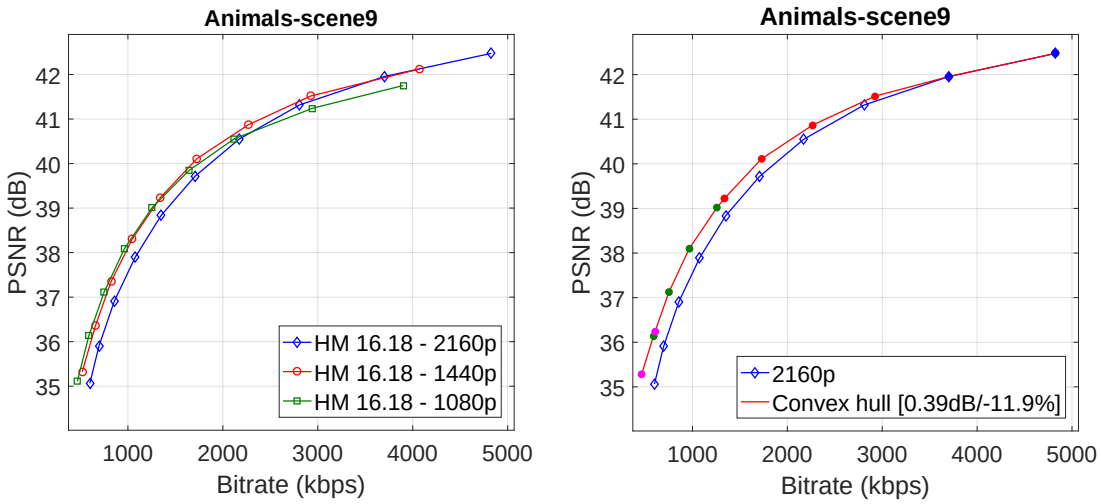
Figure. 5.6 Characterization of the training sequences using spatial information (SI), temporal information (TI) and Motion Vectors (MV).

and thus could be considered outliers. There is an inconsistency between the distribution of temporal information characterized by TI and MV. For example, *TunnelFlag_scene1* is the sequence with the highest TI value by far, but is not the sequence with the highest MV, which is *Jockey*. Another observation is that there is a lack of sequences with high values of SI and TI or MV. This is because, typically, when the motion in a scene is high, this is accompanied by motion blur which decreases the sharpness, leading to a lower amount of spatial information.

5.2.4 Rate-distortion performance across spatial resolutions

Given this dataset, the training process proceeds as follows. First, the input video sequences are downsampled from their original spatial resolution, UHD, to multiple lower resolutions. We have experimented with 3 lower resolutions, 1440p (2560x1440), 1080p (1920x1080) and 720p (1280x720), which correspond to downsampling ratios of 1.5, 2 and 3, respectively. After obtaining the original and low resolution uncompressed videos, they are encoded using HEVC test model HM 16.18 (the latest available software version at the time of experimentation) using multiple QP values, ranging from 21 to 45. After decoding, the decoded videos are upsampled to the original resolution and quality metrics, such as PSNR, are computed with respect to the original video. As discussed in Chapter 4, Lanczos3 has been chosen as the method for down/upsampling.

Consider the RD curves in Figure 5.7 (a), which compares the performance across the first 3 spatial resolutions for the training sequence *Animals-scene9*. It can be observed that encoding at lower resolutions often provides RD gains compared to encoding at the original resolution. More specifically, encoding at 1440p is beneficial for bitrates lower than 3500 Kbps, while encoding at an even lower resolution, 1080p, starts being beneficial for lower bitrates, around 1000 Kbps. Overall,



(a) RD curves for multiple resolutions

(b) Convex hull for multiple resolutions

Figure. 5.7 RD performance (based on PSNR) for the sequence *Animals-scene9* encoded using HM 16.18 with multiple input resolutions. Scaling was achieved using Lanczos3.

for this sequence, over the entire bitrate range, the BD-rates between the low resolution curves and the original resolution curve are 10.4%, 11.0% and 1.7% for 1440p, 1080p and 720p resolutions, respectively. Appendix C.2 presents detailed RD performance information across resolutions for all training sequences.

The convex hull, defined by the smallest convex set that contains all rate-points, which in this problem identifies the optimal rate-distortion encodes (using different parameters including QP and resolution) was also computed for the same sequence and is shown in Figure 5.7 (b). The colours of the points in the convex hull correspond ones used represent each resolution in Figure 5.7 (a). This plot shows more clearly the variations of optimal resolutions at different bitrates and the potential rate-distortion benefits of combining multiple downsampling ratios to achieve higher performances in a resolution adaptation framework.

This multi-resolution analysis was conducted for all sequences in the training dataset. Table 5.1 presents the average, best and worst performance of encoding at each spatial resolution and the combined convex hull. According to these figures, encoding at 1440p achieves, on average, BD-rate savings of 4.4% while the other resolutions provide average losses. However, these values represent the savings across the entire bitrate range, and therefore, does not reflect local savings achieved at a certain bitrate range. By combining the optimal resolution on the convex hull, BD-rate savings of 9.4% are obtained, on average, for all training sequences, which, once more, confirms the potential of a multi-resolution system. Moreover, when comparing the maximum and minimum savings for a single resolution, it can be seen that encoding at 720p achieves the highest savings of 31.7% but also the largest loss of 337.1%. This means that encoding at this resolution can be highly beneficial in a rate-distortion sense, but the benefits are content dependent. On the other hand, the other resolutions

Table 5.1 RD performance statistics of encoding at different spatial resolutions compared to encoding at the original resolution (2160p), based on PSNR, for the training sequences.

	Average		**Max savings**		**Min savings**		
Res.	BD-rate [%]	BD-PSNR [dB]	BD-rate [%]	BD-PSNR [dB]	BD-rate [%]	BD-PSNR [dB]	**Percentage optimal** [%]
1440p	-4.44	0.06	-21.53	0.62	22.11	-1.56	30.7
1080p	2.10	-0.28	-28.34	0.83	54.13	-2.98	25.9
720p	46.32	-1.54	-31.65	1.09	337.98	-4.92	16.8
Convex Hull	-9.41	0.27	-29.03	0.95	-0.26	0.00	-

show lower maximum savings but also smaller losses, which is advantageous for a practical system which relies on predictions of the optimal resolutions, prior to encoding.

Finally, in order to quantify the individual benefit of each resolution, the last column of Table 5.1 shows percentage of points in the convex hulls that were encoded with the given resolution. On average, 30.7% of the encodes in the convex hulls were encoded at 1440p. This figure decreases to 25.9% for 1080p and 16.8% for 720p. Therefore, from this analysis, the most beneficial resolution for an adaptive system is 1440p, which corresponds to a downsampling ratio of 1.5. In addition, for this ratio, savings can be achieved at lower bitrates. Following this conclusion, our proposed resolution adaptation framework, described in this chapter considers the downsampling ratios 1.5 and 2, compared to the work presented in Chapter 4, which employed a single ratio of 2.

5.3 Quantization-Resolution Optimization

In the previous section, we have presented the training sequences, their performance across resolutions and the decision to employ two different downsampling ratios to improve the performance of our adaptive resolution coding scheme. In this section, we present an improved Quantization-Resolution Optimization (QRO) module, which builds on the same idea as the method presented in Chapter 4 with several important modifications. Since two downsampling ratios are used two different QP thresholds need to be predicted.

To illustrate this, Figure 5.8 shows the same RD curves in of the sequence *Animals-scene9* as presented in Figure 5.7 with the addition of the QP values used to encode the original resolution rate-points, shown in blue. It can be observed that the RD curve which corresponds to encoding at 1440p starts to achieve higher PSNR values compared to the original above QP = 27 and the 1080p points start to achieve higher PSNR values compared to the 1440p points after QP = 35. These QP values correspond to the QP thresholds at ratios 1.5 (QPt1.5) and 2 (QPt2), respectively. Therefore, the aim of the proposed QRO module is to accurately predict these thresholds for any given input video.

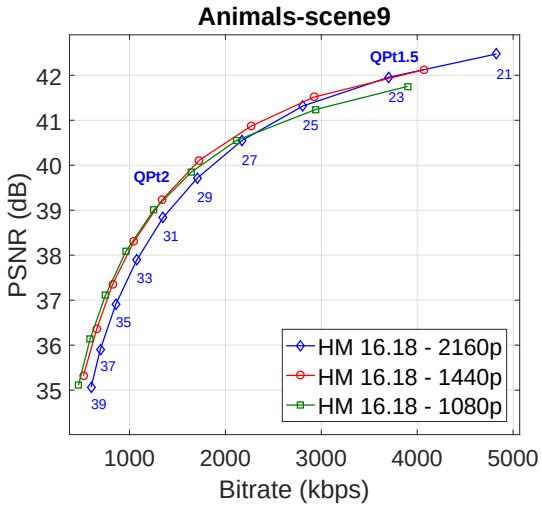


Figure. 5.8 RD curves of the sequence *Animals-scene9* encoded using 3 different resolutions. QPt1.5 and QPt2 refer to the QP threshold at downsampling ratio 1.5 and 2, respectively.

We propose to predict the QP thresholds based a machine-learning regression model trained offline using low-level visual features extracted from the uncompressed video frames as inputs. The decision to use uncompressed video features derives from the fact that no pre-encoding or multiple passes are required and this can easily be integrated with existing video codecs. Based on these features and the respective ground truth QP thresholds, the QRO model predicts the QP thresholds, which, in turn, leads to the prediction of the optimal resolution to encode the input video frames. A detailed description of the low-level features used and the prediction process are presented in the following sections.

5.3.1 Low level features

For the prediction of the QP thresholds, four features are used. These were selected from a large set of video features using a cross-validated forward feature selection process [142]. Among the features tested were: GLCM statistics [72], pixel-based frame difference and Normalized Cross Correlation (NCC) [143]. Two spatial and two spatio-temporal features were selected. The spatial features consist of the resampling PSNR, $PSNR_r$, and a spatial resolution-dependent quality metric, SRQM, whereas the temporal features consist of statistics of the Temporal Coherence (TC), which measures temporal dependencies between frames. These features are described below.

5.3.1.1 Spatial features

The resampling PSNR, $PSNR_r$, is computed by calculating the PSNR between the original full resolution frames (HR) and the frames downsampled and subsequently upsampled to the original resolution (LR). Resampling is performed using Lanczos3 filter. This feature estimates the distortion caused by the loss of high frequency information when resampling the video frames. This feature

is identical to the one presented in Section 4.2.2, since two downsampling ratios are used, two resampling PSNR features are computed, one for each ratio.

SRQM SRQM (Spatial Resolution Quality Metric), originally proposed in [144], is a full-reference quality metric designed specifically to assess spatial resampling distortions in videos. It attempts to evaluate the video quality between a low resolution video (LR) and its original resolution (HR). A 2-D Haar Discrete Wavelet Transform (DWT) [16] is applied to both the HR (I^{HR}) and LR (I^{LR}) video frames, decomposing the image into 4 subbands (B), the low frequency, DC (B_{LL}), and three high frequency subbands: horizontal (B_{LH}), vertical (B_{HL}) and diagonal (B_{HH}). These are computed according to the following equations:

$$\begin{aligned} \{B_{LL}^{HR}, B_{HL}^{HR}, B_{LH}^{HR}, B_{HH}^{HR}\} &= \mathrm{DWT}(I^{HR}) \\ \{B_{LL}^{LR}, B_{HL}^{LR}, B_{LH}^{LR}, B_{HH}^{LR}\} &= \mathrm{DWT}(I^{LR}) \end{aligned} \tag{5.8}$$

Next, the absolute difference between the corresponding high frequency subbands of HR and LR frames ($\triangle$B) is calculated and averaged:

$$\triangle B = \frac{1}{3}\{|B_{HL}^{HR} - B_{HL}^{LR}| + |B_{LH}^{HR} - B_{LH}^{LR}| + |B_{HH}^{HR} - B_{HH}^{LR}|\} \tag{5.9}$$

Given that spatial distortions are not uniformly distributed within a frame, spatial pooling is employed by dividing the subband coefficients differences into K non-overlapping blocks and accounting for local distortions. For this work, we defined the size of each block as 32×32 due to all training sequences being UHD resolution as in [144]. The Average Spatial Distortion (ASD) for each block with index k is then calculated by averaging the coefficient differences within a block and obtaining the ASD for blocks $k = 1, 2, ..., K$:

$$\mathrm{ASD}(k) = \sum_{\forall (x,y) \in \mathrm{Block_k}} \frac{\triangle B(x,y)}{M} \tag{5.10}$$

where M is the number of pixels in a block.

The final distortion measure is calculated based on the maximum value among all blocks within a frame:

$$D_{SRQM} = \max\{\mathrm{ASD}(1), \mathrm{ASD}(2), ..., \mathrm{ASD}(K)\} \tag{5.11}$$

The final SRQM quality score is calculated using the average distortion for each frame, converted to decibels:

$$\mathrm{SRQM} = 20 \cdot log_{10}({D_{SRQM}}^{-1}) \tag{5.12}$$

Again, two SRQM features are produced for each input frame according to the two downsampling ratios considered.

Although both the PSNR_r and SRQM provide an estimate of the quality of spatial resampled video frames, we have found both features to be useful in combination for predicting the QP thresholds.

5.3.1.2 Spatio-temporal features

We estimate the Temporal Coherence (TC) between two successive video frames [145] as a spatio-temporal feature. TC estimates how well the current frame corresponds to the subsequent frame at each spatial frequency, f, by measuring the magnitude-squared coherence between the Power Spectral Density of adjacent frames in time. The Power Spectral Density (PSD) is computed using the Fast Fourier Transform (FFT) for each dimension (horizontal and vertical). For each dimension TC takes values between 0 and 1 and can be calculated using the following equation:

$$\text{TC}_{dim}(f) = \frac{|P_{I_t I_{t+1}}(f)|^2}{P_{I_t I_t}(f) P_{I_{t+1} I_{t+1}}(f)} \tag{5.13}$$

where $P_{I_t I_{t+1}}$ is the cross-spectral density between the two frames and $P_{I_t I_t}$ is the auto-spectral density of the frame at t.

Two statistics of the calculated TC are used as the low-level visual features. The first, TC_{mean} is the sum of the average horizontal (TC_h) and vertical (TC_v) TC measures:

$$\text{TC}_{\text{mean}} = \frac{1}{\text{N}_h} \sum_{\forall f \in \text{F}_h} \text{TC}_h(f) + \frac{1}{\text{N}_v} \sum_{\forall f \in \text{F}_v} \text{TC}_v(f) \tag{5.14}$$

where F_h, F_v, N_h, N_v are the horizontal and vertical frequency ranges and number of frequencies, respectively.

The second feature, the skewness of TC, TC_{sk}, is calculated as the sum of the skewness measure of each dimension:

$$\text{TC}_{\text{sk}} = \text{skewness}\{\text{TC}_h(f)\} + \text{skewness}\{\text{TC}_v(f)\} \tag{5.15}$$

We have found TC to be more helpful at estimating the QP thresholds between resolutions than other temporal measures such as the pixel-wise frame difference.

5.3.2 QP threshold prediction

The features were calculated and averaged for all frames of the set of training sequences. In addition, target QP thresholds were estimated by analysing the crossover points between the RD curves obtained by encoding all training sequences with the three different resolutions mentioned (2160p, 1440p and 1080p). All plots can be found in the Appendix C.2. One observation is that our estimated ground truth QP thresholds can differ slightly from the true values, given that they were based on fittings of the RD curves and a limited number of RD points.

To understand how the selected visual features correlate with the estimated QP thresholds, Figure 5.9 shows the relationship between SRQM and TC_{mean} with the QP thresholds for ratio 2 and 1.5,

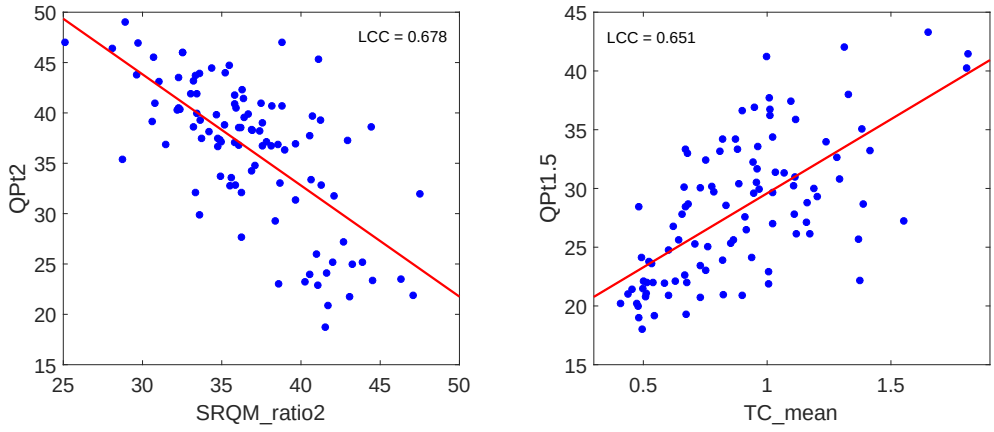


Figure. 5.9 Relationship between selected low-level features and the QP thresholds: (a) SRQM (ratio 2) vs QP threshold (ratio 2); (b) TC_{mean} vs QP threshold (ratio 1.5). The red curve is a linear fit on the data, for illustration purposes only.

respectively, for each training sequence. From the distribution, it can be seen that SRQM has a negative correlation with the QP thresholds. In general, the higher the SRQM score, the lower the QP thresholds. This means that the scaling artifacts introduced have a lower impact on the video quality, leading to a lower QP threshold. On the other hand, as a general trend, the higher the TC_{mean} feature, the higher the QP threshold, which suggests that spatial resampling is more appropriate in videos with complex motion characteristics.

After obtaining the full set of features and ground truth QP thresholds, two Support Vector Machine regressor (SVM_r) [142] models were trained, one for each downsampling ratio. The process of predicting the QP thresholds based on the visual features is illustrated in Figure 5.10.

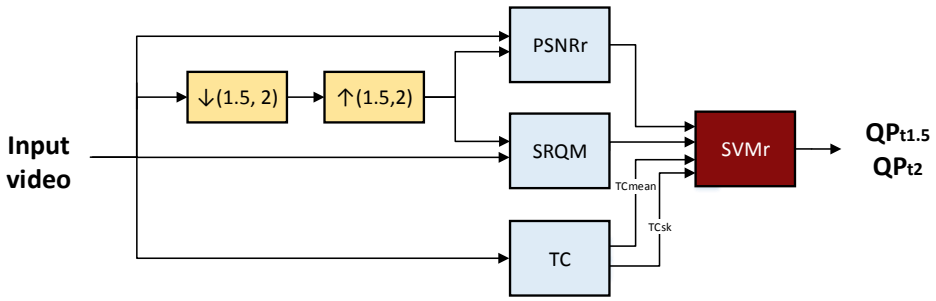


Figure. 5.10 Diagram representing the proposed QRO process of predicting the QP thresholds from visual features.

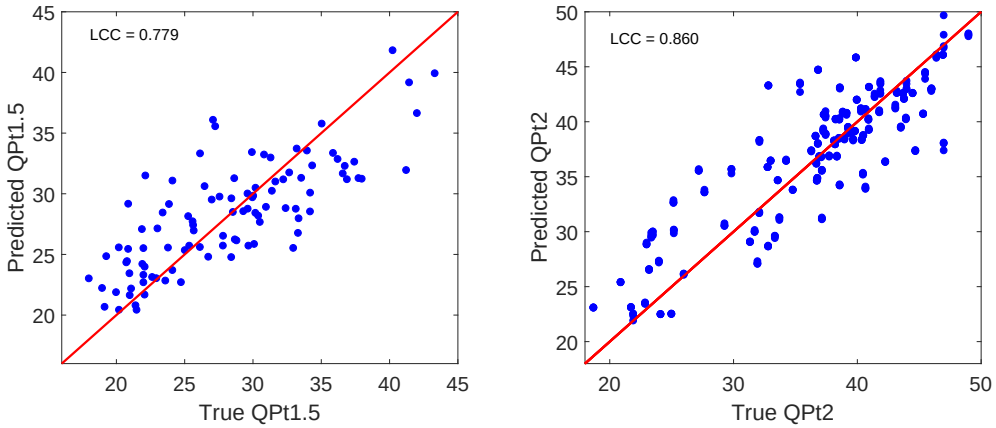
A 5-fold cross-validation using the training data was conducted to train the two SVM regressors. The accuracy of the regression analysis can be found in Table 5.2. Moreover, Figure 5.11 shows the

relationship between the ground truth QP thresholds and the cross-validated predicted thresholds for each sequence in the training dataset.

Table 5.2 Goodness of fit of the QP thresholds prediction models.

Model	**RMSE**	**MAE**
QP threshold - ratio 1.5	3.77	2.97
QP threshold - ratio 2	3.60	2.69

On average, the difference between the predicted QP threshold values and the ground truth values is 2.97 for ratio 1.5 and 2.69 for ratio 2, in QP units. Given these values, there is still room for improvement on the accuracy of the predictions. However, we note that the prediction model was trained on videos that were upsampled using the Lanczos3 filter, not our proposed CNN and are also based on QP thresholds using PSNR to assess the visual quality. Therefore, we believe these predictions can be considered conservative. As we will show in Section 5.5, our CNN can significantly improve video quality after decoding.



(a) (b)

Figure. 5.11 QP threshold ground truth vs predicted.

5.3.3 QP adjustment

Similarly to the approach proposed in Chapter 4, the final step in the QRO process is to adapt the base QP of the host encoder when spatial resolution adaptation is performed. The goal is to achieve similar bitrates as otherwise would be achieved by encoding the original resolution video frames. Since the low resolution video produces lower bitrates, in order to achieve the same bitrate, a lower QP value should be used. Based on an analysis of the training dataset, we found that at the crossover points, the difference between the base QP of the original resolution and the lower resolutions is approximately 3 and 6 (QP units) for ratios 1.5 and 2, respectively. Therefore, based on the input QP value, QP_{input},

and the optimal ratio given by the SR_{flag} (see Section 5.2), the adjusted $QP_{adj.}$, used to encode the video frames is given by:

$$QP_{adj.} = \begin{cases} QP_{input}, & \text{if } SR_{flag} = 0 \\ QP_{input} - 3, & \text{if } SR_{flag} = 1 \\ QP_{input} - 6, & \text{if } SR_{flag} = 2. \end{cases} \tag{5.16}$$

5.4 CNN for compressed super-resolution

Inspired by developments in machine learning and deep learning for image reconstruction, we propose to employ a CNN-based single-image super-resolution method at the decoder side for the upsampling process. Due to its impressive performance compared to other approaches, we have based our CNN on the architecture of VDSR [6]. This network was previously described in Section 5.1.1 and shown in Figure 5.3. It consists of a very deep network architecture with a cascade of 20 convolutional layers, each layer containing 64 filters of size 3×3 followed by Rectified Linear Units (ReLU). Moreover, Bicubic interpolation is used to pre-upsample the low resolution input to the high resolution before being processed by the convolutional layers. Instead of using Bicubic, we apply Lanczos3 for pre-upsampling given that it is the same filter used for downsampling at the encoder and was shown to provide superior performance (Section 4.2.1). To reduce the complexity of the system, the CNN is applied to the Y channel of the reconstructed video frames, while the chroma channels are upsampled using Lanczos3 only. This is justified by the fact that the Human Visual System is more sensitive to details in intensity compared to colour [16], and therefore the benefit of processing the chroma channels would be limited.

5.4.1 Training methodology

We train the CNN parameters using videos that were downsampled and compressed using HEVC (HM 16.18). This methodology is illustrated in Figure 5.12 and is similar to the process applied for obtaining the RD information for training the QRO model in Section 5.3. Hence, the video frames used for the training of the CNN have both scaling and HEVC compression artifacts. This is different from the standard super-resolution problem, where no compression is introduced.

The training sequences described in Section 5.2.3, were downsampled to two different resolutions, 1440p and 1080p, using downsampling ratios (r) of 1.5 and 2, respectively. These videos are then encoded using two base QP values: a "low" QP range and a "high" QP range, more specifically, {27,37} and {25,35} for ratios 1.5 and 2, respectively. These were chosen based on the typical range of QP values used in the Common Test Conditions (CTC) for JVET [10]. The use of two quantization ranges provides a distinction between videos with different levels of compression artifacts, and is intended to provide improved performance, having also been applied in previous works (see Section 5.1). This procedure generates a total of 4 sets of training videos, which will be used to guide the learning of the different CNN models.

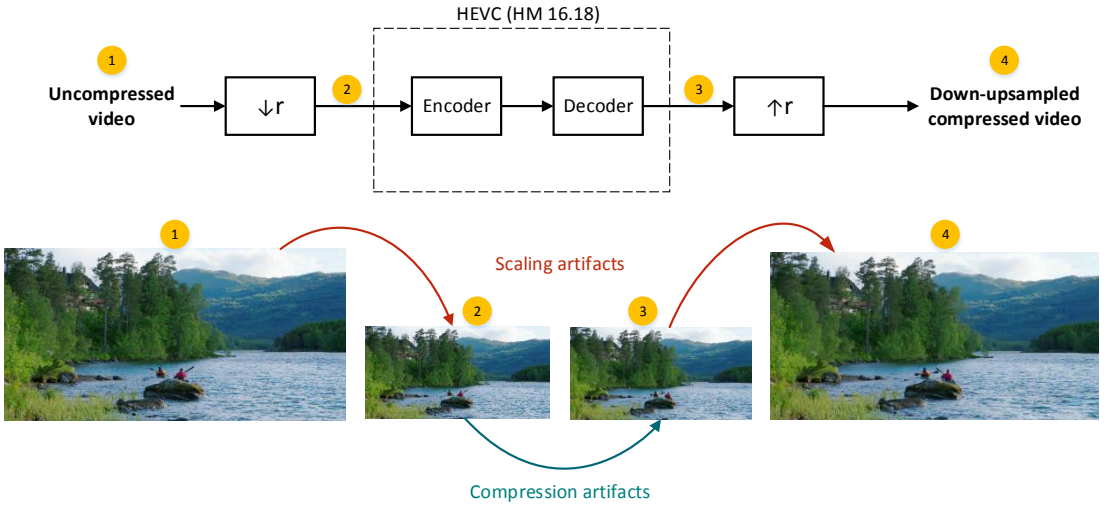


Figure. 5.12 Illustration of the process of obtaining the training video frames for the proposed CNN for reconstruction.

The proposed CNN is trained through the use of mini-batches of input and output image blocks. In order to limit the training time, only 8 frames from each training sequence are used. Each frame is split into 128 randomly selected blocks of size 41×41 pixels. Moreover, to enhance the generalization ability of the model, data augmentation consisting of 90 degree rotations is also applied to each block. As a final step, the pixels are normalized to the range 0 to 1, by dividing by the maximum allowed pixel value (1023 for 10 bit videos). This process is performed for both the input video frames, which were subject to the scaling and compression processes, and the original frames. Each input block corresponds to a co-located output block from the uncompressed video. The size of each block, 41×41, is equivalent to the receptive field of this CNN architecture, which by definition is the local window in the input image that influences the results of one pixel in the output image. In total, approximately 400 thousand blocks are used to train each of our 4 CNN models.

After obtaining the data, Tensorflow version 1.2 [146] was the deep learning framework used to conduct the training of the network's parameters. In order to speed up the training process, which takes approximately 1 day per model, a NVIDIA 1080ti Graphical Processing Unit (GPU) was used. In addition, the following hyper-parameters were employed: total of 150 epochs (entire iterations on the training dataset), batch size of 64, Adam optimizer [64] with the parameters beta1=0.9 and beta2=0.999, learning rate of 1e-4 for 100 epochs and then 1e-5 for the remaining 50 epochs. The parameters were learned by minimizing the ℓ_2 loss over each mini-batch, which is equivalent to minimizing the MSE (or maximizing the PSNR) between the reconstructed block, $f(x)$ and the original uncompressed block, y as expressed by the following equation:

$$\text{Loss} = \ell_2(x) = \frac{1}{2} \cdot \|y - f(x)\|^2 \tag{5.17}$$

We have found that minimizing this loss improves the overall quality by reducing the artifacts introduced by rescaling and compression and creating an overall sharper frames. In addition, perceptual quality is also visibly improved by the application of the CNN (Section 5.5.3).

5.4.2 Deployment

The trained CNN model parameters (weights and biases) need to be stored at the decoder. For each CNN, a total of approximately 400000 parameters are required, which corresponds to nearly 1 MB of storage space, to be stored at the decoder side. The overall operation of the proposed decoding process is illustrated in Figure 5.13. After decoding the video frames, if resampling was performed by the encoder, signalled by a resampling flag, SR_{flag}, the frames are first upsampled to the final resolution using Lanczos3 and the luminance (Y) channel is further processed. The first step is to normalization the input values to achieve a range from 0 to 1 (the same used for training). Then, the frames are split into non-overlapping blocks of size 128×128. This process is employed in order to reduce the GPU memory required, which would be impractical for large spatial resolutions. The blocks contain an overlap of 4 pixels, in then averaged to provide a smooth transition. Depending on the value of the SR_{flag} and the base QP used by the encoder, one of the 4 pre-trained CNN models is employed. Finally, the frames are inverse normalized to the correct bit depth and the colour components are concatenated to form the final reconstructed video frames.

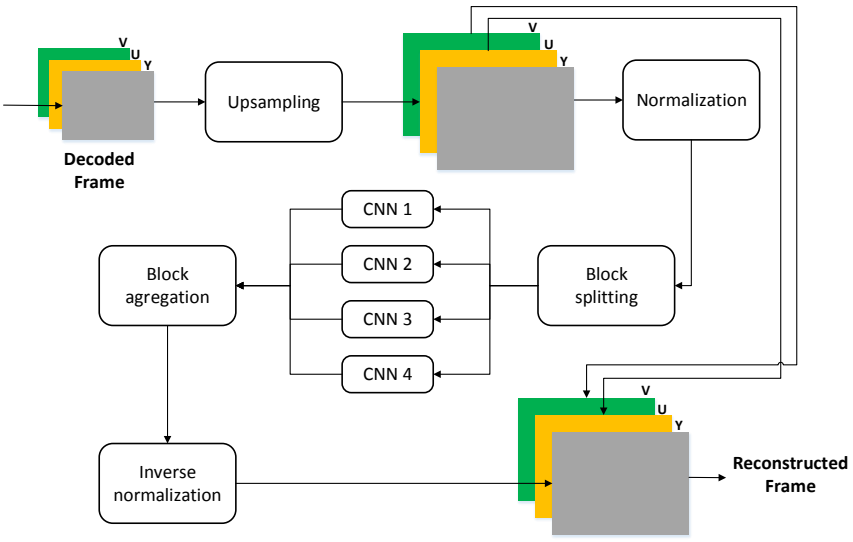


Figure. 5.13 Diagram of the proposed decoder methodology.

As mentioned above, the CNN model to use is selected based on the downsampling ratio employed at the encoder (predicted by the QRO and represented by the SR_{flag}) and the base QP used for encoding.

This is selected using the following criteria:

$$\text{CNN}_{\text{param.}} = \begin{cases} model1, & \text{if } \text{SR}_{\text{flag}} = 1 \text{ and } \text{QP}_{\text{input}} < 32 \\ model2, & \text{if } \text{SR}_{\text{flag}} = 1 \text{ and } \text{QP}_{\text{input}} \geq 32 \\ model3, & \text{if } \text{SR}_{\text{flag}} = 2 \text{ and } \text{QP}_{\text{input}} < 30 \\ model4, & \text{if } \text{SR}_{\text{flag}} = 2 \text{ and } \text{QP}_{\text{input}} \geq 30. \end{cases} \tag{5.18}$$

where 32 and 30 correspond to the mid-point between the base QP values used for the training of the models with downsampling ratios of 1.5 and 2, respectively. Furthermore, since all blocks can be processed independently, parallelization of the CNN processing is possible, which could significantly minimize the complexity impact of this approach at the decoder.

5.5 Experimental results

5.5.1 Test sequences

An evaluation of the proposed adaptive resolution framework was conducted using 14 test sequences from several databases, including BVI texture database [13], BBC R&D [147], MPEG DASH dataset [148], Netflix ElFluente [149] and Netflix Chimera [49]. Moreover, 3 of the test sequences are commonly used in the evaluation for future video codecs at JVET [50]. All sequences are UHD resolution, are composed of 300 frames, are either at 60 or 50 fps, are 10 bits per sample and use 4:2:0 chroma subsampling. Further characterization of each individual test sequence including respective sources can be found in the Appendix D. Furthermore, an example frame of each of the sequences can be found in Figure 5.14. It is important to note that none of these sequences are included in the set of training sequences used to train the QRO module or the parameters of the CNN models.

Figure. 5.14 Sample frame from each of the test sequences.

5.5.2 Rate-distortion performance

HEVC reference software HM 16.18 was used as the host codec and anchor for the experimental tests conducted on the proposed approach. The test sequences were encoded using 4 rate-points at the QP values {27,32,37,42} using Random Access configuration and Main10 profile. All other encoding parameters followed the Common Test Conditions [10]. The quality of the reconstructed videos was evaluated both PSNR and VMAF [32].

Table 5.3 summarizes the experimental results, measured in BD-rate and BD-quality, obtained for the proposed method against the anchor for the set of test sequences. In addition, the predicted QP thresholds at ratio of 1.5 ($QP_{t1.5}$) and 2 (QP_{t2}), are also presented.

Table 5.3 Experimental results of the proposed spatial resolution adaptation approach compared to the fixed-resolution HEVC HM 16.18 anchor for the 14 test sequences.

Sequence	**Pred.** $QP_{t1.5}$	**Pred.** QP_{t2}	**BD-rate (PSNR) [%]**	**BD-PSNR [dB]**	**BD-rate (VMAF) [%]**	**BD-VMAF**
S01: CalmingWater	20.5	27.6	-14.8	0.67	-16.1	5.0
S02: CatRobot	27.5	36.6	-13.2	0.41	-15.5	3.0
S03: CentralLineCrossing	27.1	36.2	-7.4	0.22	-12.8	2.1
S04: Chimera-ep08	23.9	32.4	-19.8	0.68	-24.0	4.4
S05: Chimera-ep12	26.9	37.5	-13.6	0.55	-12.4	2.4
S06: Chimera-ep17	22.7	31.7	-12.8	0.48	-15.5	3.1
S07: Chimera-ep20	26.3	34.0	-10.9	0.46	-19.1	3.4
S08: DaylightRoad	25.6	32.8	-10.5	0.23	-16.9	3.1
S09: ElFuente-ep33	20.8	22.8	-22.0	0.79	-24.7	5.3
S10: FoodMarket	26.2	35.4	-11.5	0.47	-16.7	3.6
S11: LampLeaves	28.2	38.3	-10.9	0.51	-9.4	2.4
S12: Manege	32.7	41.96	2.7	-0.1	-4.6	0.7
S13: NingyoPompons	27.7	37.5	-13.5	0.65	-14.9	3.8
S14: PetiBato	27.9	37.1	-3.6	0.07	-12.1	1.9
Average	26.0	34.4	-11.5	0.43	-15.3	3.1

On average, the proposed approach produces BD-rate savings of 11.5% based on PSNR and 15.3% based on VMAF for the test dataset. The sequences with the best performance are *S09: ElFuente-ep33* and *S04: Chimera-ep08* with significant BD-rate savings of 22.0% and 19.8% based on PSNR and 24.7% and 24.0% based on VMAF, respectively. These sequences contain high motion and a blurred background, which are beneficial characteristics for our spatial resolution adaptation approach. On

the other hand, for one test sequence, *Manege*, our method produces a slight decrease in BD-rate based on PSNR of 2.7% but produces gains of 4.6% when considering VMAF as the quality metric. This is a sequence with low motion and a high amount of spatial details, which makes it less likely to benefit from our approach.

It can also be seen that the QP thresholds predicted range from 20.5 to 32.7 for ratio 1.5 and 27.6 to 41.96 for ratio 2. However, an interesting observation is that the sequence with the lowest QP thresholds, which is *CalmingWater* is not the same as the sequence with the highest savings, *ElFuente-ep33*. Overall, the QP thresholds predicted for ratios 1.5 and 2 are 26.0 and 34.7 respectively, which means that, on average, for the tested sequences, it is beneficial to perform downsampling to 1440p before encoding from a QP value of 26 and to 1080p from QP 34.7, given an input sequence with UHD (2160p) resolution.

RD curves based on PSNR and VMAF for all test sequences are presented in Figure 5.15 and 5.16, respectively. In addition, aiming at assessing the independent benefit of the CNN models, the performance of the proposed method without the use of a CNN at the decoder is also presented. In this case, the same prediction results are used but only Lanczos3 is employed for upsampling. This is represented by the black dashed RD curve in each plot. BD-rate and BD-quality metrics against the anchor (blue curve) were also computed and can be found next to the legend of each curve and represent savings. Finally, for each rate-point, the resolutions used are represented by the coloured markers on the black curve and follows the colour code: magenta for ratio 1.5 and green for ratio 2.

Savings were achieved for almost all rate points across a wide range of bitrates for most sequences. Out of 56 rate-points (4 rate-points / sequence $\times$ 14 sequences), only 7 were encoded at the original resolution. Downsampling with a ratio 1.5 was performed for 23 rate-points, mainly for the ones with higher bitrates and the remaining 26, including all rate-points with the lowest bitrate, were encoded with downsampling ratio 2. Apart from the first two rate-points of *DaylightRoad2* and last two of *Manege*, the spatial downsampled rate-points generate better performance compared to the anchor, which reflect accurate predictions by the QRO module. Moreover, these results show a clear benefit of the addition of the downsampling ratio 1.5, which provides bitrate savings in higher bitrate ranges, increasing the flexibility of our proposed approach. Finally, it can be seen that the application of the CNN contributes favourably to our method by providing significant BD-rate savings on top of the resolution adaptation, 6.4% based on PSNR.

5.5.3 Visual quality evaluation

In order to access the differences in visual quality provided by the use of the CNN for reconstruction of the full resolution frames, a visual quality comparison is shown in Figure 5.17, and presents reconstructed patches from two of the test sequences using Lanczos3 for upsampling and using the CNN at similar bitrates. The example patch corresponds to the lowest bitrate rate-points of *CatRobot1* and *LampLeaves* (see Figures 5.15 and 5.16). It can be seen that reconstructed using Lanczos3

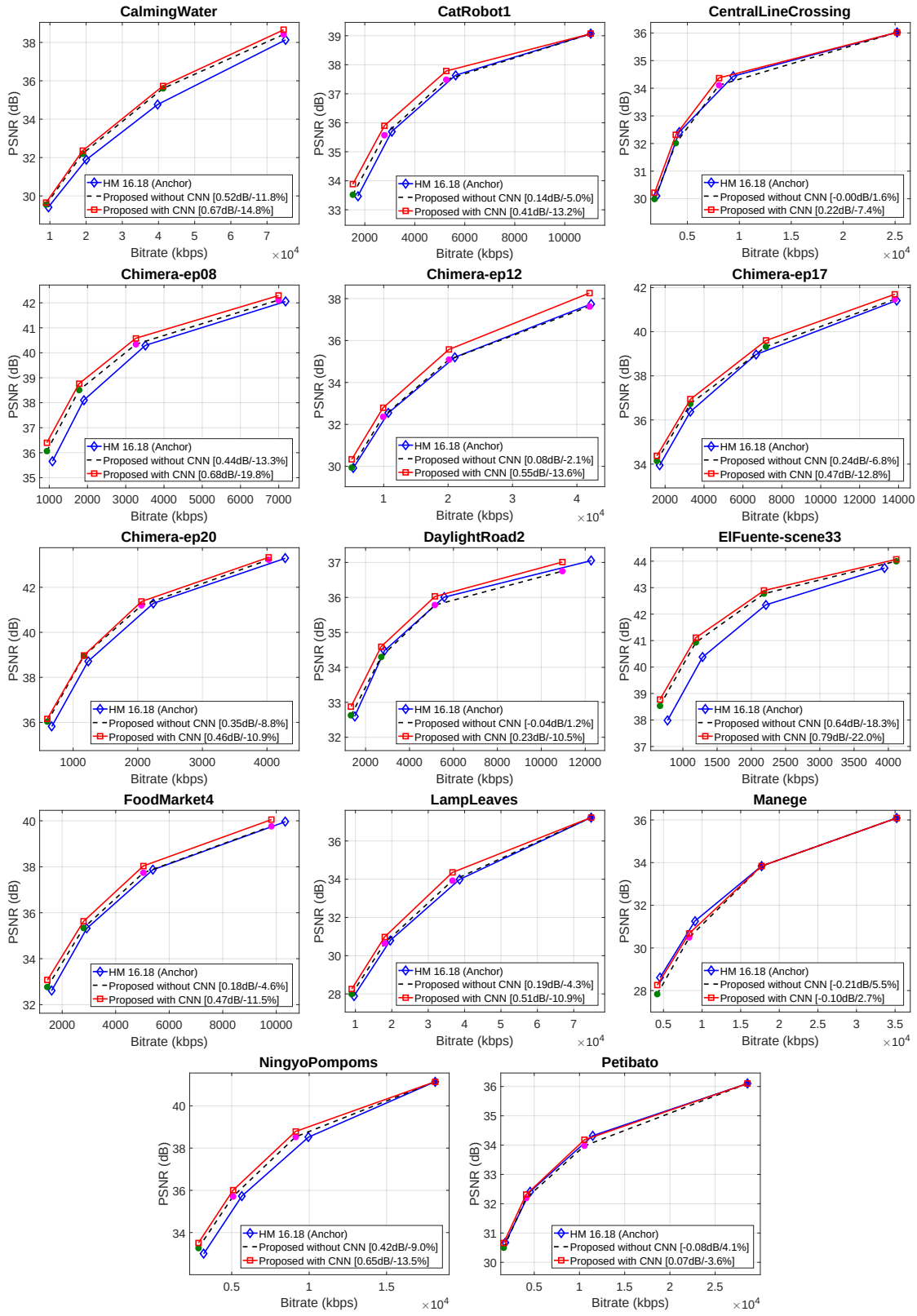


Figure. 5.15 Rate-quality curves (based on PSNR) comparing the anchor, HM 16.18, and the proposed method with and without the CNN. The downsampling ratios selected by the QRO are represented by different colors: 1.5 (magenta) and 2 (green).

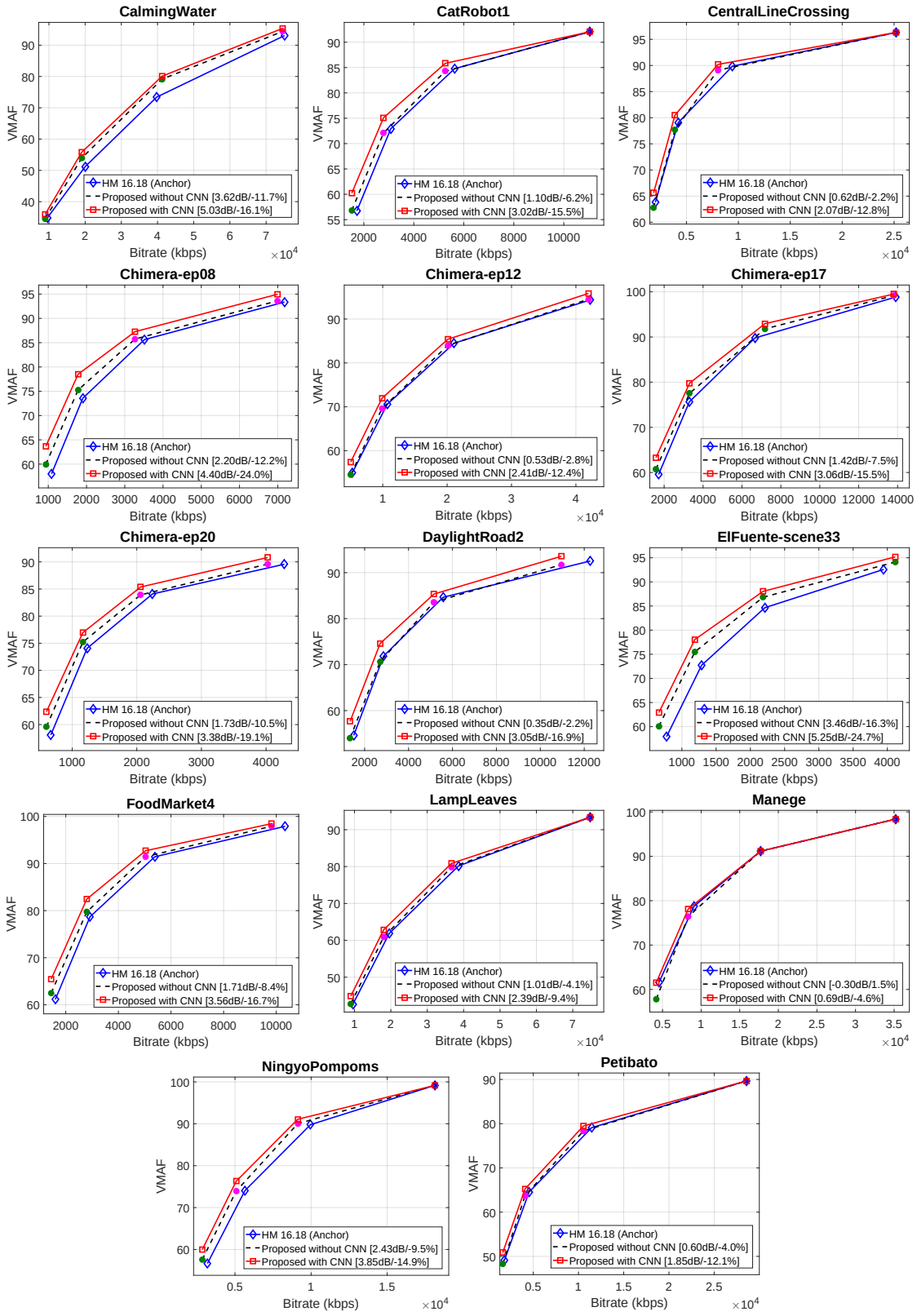


Figure. 5.16 Rate-quality curves (based on VMAF) comparing the anchor, HM 16.18, and the proposed method with and without the CNN. The downsampling ratios selected by the QRO are represented by different colors: 1.5 (magenta) and 2 (green).

(a) CatRobot1 @ 1500 Kbps - L3

(b) CatRobot1 @ 1500 Kbps - CNN

(a) LampLeaves @ 18000 Kbps - L3

(b) LampLeaves @ 18000 Kbps - CNN

Figure. 5.17 Visual comparison between different upsampling methods, Lanczos3 (L3) and the proposed CNN. The patches are 512×512 pixels and were extracted from the 150th frame of CatRobot and LampLeaves at the lowest quality rate-point. Best viewed in the digital version.

contains compression artifacts especially around edges, while in the one reconstructed by the CNN, these are visibly reduced. In addition, the representation of edges is also improved by the CNN.

Further visual comparisons of the proposed approach against the anchor HEVC encoded at the original resolution, at similar bitrates are shown in Figure 5.18 for the lowest quality rate-points of the sequences *CatRobot1*, *DaylightRoad2* and *Chimera-ep08*. These results show a clear improvement in perceptual quality confirming the savings achieved using objective quality metrics.

5.5.4 Complexity analysis

The proposed spatial resolution approach has been mainly designed with the goal of achieving improved compression efficiency. Nonetheless, it is also relevant to understand the impact on the encoder and decoder complexities, in order to evaluate how this would be suited as a practical video codec technique.

Since the proposed approach allows the adaptive encoding and decoding of low resolution video frames, the overall encoding time is reduced, even considering the time required to compute the low-

(a) CatRobot1 @ 1500 Kbps

(b) DaylightRoad2 @ 1300 Kbps

(c) Chimera-ep08 @ 950 Kbps

Figure. 5.18 Comparison between the anchor HM 16.18 (left-up and middle) and the proposed method (left-down and right) for three test sequences, CatRobot1, DaylightRoad2 and Chimera-ep08 at similar bitrates (the lowest bitrate rate-point in Figure 5.15). The patches are 512×512 pixels and were extracted from the 150th frames of each reconstructed sequence. Best viewed in the digital version.

level visual features, which is approximately 3 seconds per frame. However, although low resolution frames are decoded, the overall decoding time is significantly increased by the application of the CNN to reconstruct the full resolution frames. For the test set and configurations used, the average encoding time was decreased, on average, by 51%. As for the decoding time, this was increased 5.3 times, on average. However, the decoding time figure is based on the use of a recent GPU for the CNN processing. More specifically, these experiments were conducted using a PC with an Intel Core i7 4770K CPU @ 3.5 GHz processor and a NVIDIA 1080ti GPU for the CNN inference, which takes approximately 1.3 seconds per UHD frame. If a CPU was used instead, the decoding time would be increased by a factor of approximately 100 times.

Based on the complexity analysis results presented above, we believe that the proposed framework would be impractical for current production encoders and decoders. However, upcoming advances in dedicated hardware chips for deep learning [150], are likely to result in a wide availability of deep learning-enabled devices including mobile devices, PCs and televisions. For this reason, we believe this approach would be feasible for in the near future for the next generation of video codecs. It is also important to note that no complexity optimizations were considered in the design of our proposed CNN, and therefore, there is still room for improvement in that sense.

5.5.5 Comparison with other methods

As described in Sections 4.1 and 5.1.1, spatial resolution adaptation has been previously explored for video coding applications. The work presented in [123] is especially relevant given that it also employs a CNN for the upsampling of the decoded samples. Unlike ours, the resolution adaptation is applied within the encoding loop, at the CTU level. This leads to a higher flexibility in deciding which CTUs would benefit from being encoded at a lower resolution. This can be useful when the same picture contains regions with different spatial characteristics. In addition, no prediction stage is necessary since the resolution decision is made by the RDO process. However, this approach requires extra signalling of the resolution decisions for each CTU. In addition, even if all CTUs are encoded at the lower resolution, the number of CTUs remains the same and, therefore, there is a signalling overhead compared to encoding the entire frame using a reduced spatial resolution. This effect is relevant since the cost of transmitting the additional side-band information is higher at lower bitrates. Another disadvantage is the significant increase in encoding time caused by the additional mode decision steps and the use of a CNN within the encoder loop.

In regards to compression efficiency, in [123], BD-rate savings of –5.6% based on PSNR-Y are reported, which is lower than we have found using our method. However, this performance figure cannot be directly compared to ours since, in addition to the use of different UHD test sequences, their experiments are based on an older HM version, HM 12.1, and the method was only designed and applied to Intra pictures.

5.6 Grand Challenge in Video Compression Technology at ICIP 2017

A previous version of the proposed adaptive resampling framework was submitted to the Grand Challenge in Video Compression Technology at the International Conference in Image Processing (ICIP), held in September 2017. The goal of this challenge was to receive innovative video coding contributions which outperformed the current video coding standards. They received several contributions, some accompanied by technical papers to the conference. The evaluation was conducted using both objective metrics such as PSNR and VMAF, and independent subjective testing, conducted by members of the organization committee the challenge.

In addition to adapting the spatial resolution of the input video, our proposed model also included temporal adaptation. However, since this is out of scope for the present thesis, in this section, we will focus exclusively on the spatial resolution part and its respective results. This contribution has won 1st place at the challenge and has been published as a letter in the IEEE Transactions on Circuits and Systems for Video Technology (CSVT) [47].

5.6.1 Proposed approach

Similarly to the model presented in Sections 5.3 and 5.4, the spatial resolution decision module attempts to predict the QP thresholds by employing a learning-based approach based on the use of low-level spatio-temporal features extracted from uncompressed video frames. In this version, four features are used: one spatial, the resampling PSNR, and three temporal features based on the Normalized Cross-Correlation (NCC) [145] and Temporal Coherence (TC). In addition, unlike the method proposed previously, only a single downsampling ratio of 2 is considered. The feature vector, $\mathbf{K}$, is composed as follows:

$$\mathbf{K} = [\mathrm{PSNR_r}, \mathrm{NCC_{skewness}}, \mathrm{TC_{kurtosis}}, \mathrm{TC_{skewness}}] \tag{5.19}$$

where $\mathrm{PSNR_r}$ is the resampling PSNR, $\mathrm{NCC_{skewness}}$ is the skewness of the NCC, $\mathrm{TC_{kurtosis}}$ is the kurtosis of the TC and $\mathrm{TC_{skewness}}$ is the skewness of the TC.

Unlike the model described in Section 5.3, which used a SVM regressor, the fitting parameters of this model were obtained by a simple linear regression, and new samples can be predicted using the following equation:

$$\mathrm{QP_{thres}} = \mathbf{W} \cdot \mathbf{K}' + \beta \tag{5.20}$$

where $\mathbf{W}$ and β are the fitting parameter and are given by $\mathbf{W} = [-0.62, 1.94, -0.58, -3.87]$ and $\beta = 63.7$. The training dataset consisted of a total of 57 UHD sequences from the Harmonic Inc dataset [48] and the Root Mean Square Error (RMSE) of the fit on the training data is 3.26 (QP values).

Similarly to the work presented earlier in this chapter, we proposed to train an end-to-end CNN to be employed at the decoder for the scaling of the decoded videos. The CNN architecture was also

Table 5.4 Experimental results measured by BD-rate and BD-quality using PSNR, VMAF and subjective MOS scores comparing the proposed method and the anchor HEVC reference software HM 16.14.

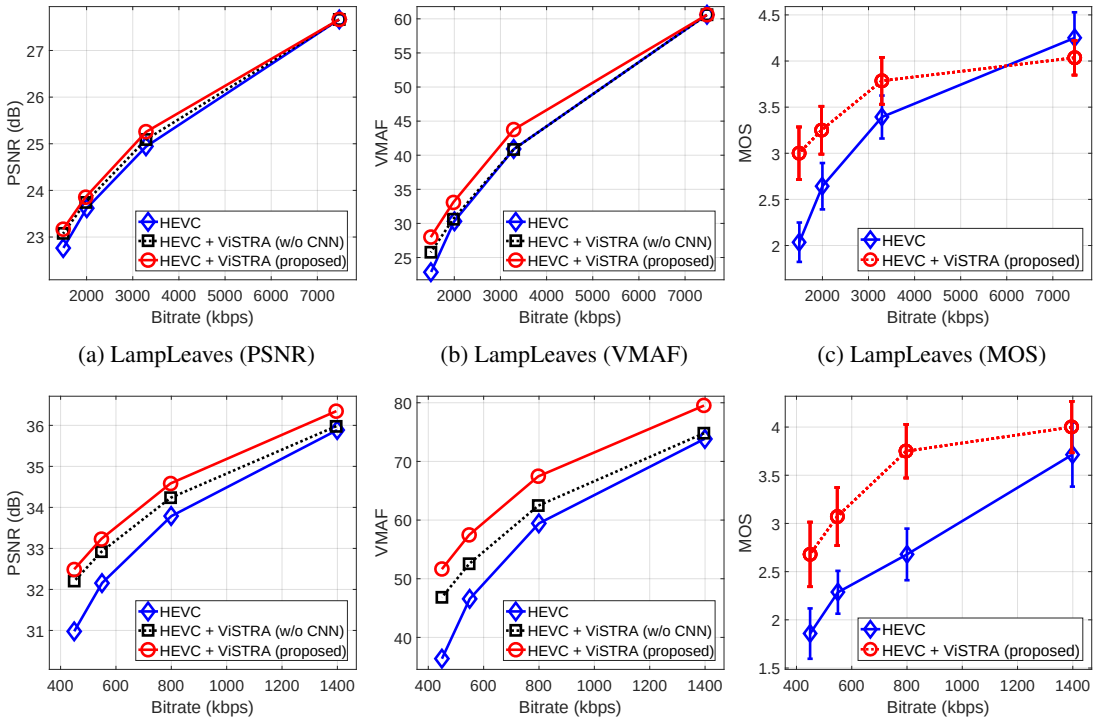
Sequence	BD-rate (PSNR) [%]	BD-PSNR [dB]	BD-rate (VMAF) [%]	BD-VMAF	BD-rate (MOS) [%]	BD-MOS
CalmingWater	-12.7	0.33	-19.2	4.3	-18.1	0.26
DropsOnWater	-1.4	0.13	-11.2	2.8	-41.4	0.49
LampLeaves	-9.9	0.28	-11.9	2.8	-26.4	0.32
TreeWills	-17.5	0.52	-23.5	5.8	-29.6	0.50
Avg. HD	**-10.4**	**0.32**	**-16.5**	**3.9**	**-28.9**	**0.40**
CatRobot	-19.6	0.87	-26.3	8.9	-44.0	0.81
DaylightRoad	-14.9	0.51	-21.5	6.8	-34.8	0.66
FoodMarketPopcorn	-16.9	0.80	-24.4	8.2	-25.4	0.42
FoodMarketSteam	-20.9	1.04	-29.7	9.4	-46.5	0.77
ParkRunning	-16.4	0.49	-23.1	5.7	-36.0	0.44
Avg. UHD	**-17.7**	**0.74**	**-25.0**	**7.8**	**-37.3**	**0.62**
Avg. total	**-14.5**	**0.55**	**-21.2**	**6.1**	**-33.6**	**0.52**

based on VDSR but was trained on the same 57 Harmonic Inc training sequences used for training of the prediction model, encoded using an earlier version of the HEVC test model, HM 16.14. The other training parameters are similar to the presented in Section 5.4. However, instead of using Tensorflow, training was performed using the deep learning framework CAFFE [151].

5.6.2 Objective and subjective results

The Grand Challenge test dataset is composed of 9 test sequences, 4 HD, 1920×1080, and 5 UHD cropped to 2560×1600 pixels. Some sequences were part of the JVET test set [50] and others of the BVI Texture database [13]. Four target rate-points were provided for each sequence together with the respective HEVC (HM 16.14) anchor bitstreams. These rate-points were selected in order to provide low quality anchors with QP values ranging from 35 to 49, which is higher than the ones used in the previous experimental results in Section 5.5.2.

Independent subjective tests were performed by the organizers using the anchor, HM 16.14, and the received submissions using a single-stimulus methodology with a total of 28 participants conforming to the home environment conditions outlined in BT.500-13 [152]. Results using BD-rate and BD-quality measurements on objective metrics and subjective scores comparing the proposed approach with the anchor can be found in Table 5.4. Additionally, Figure 5.19 presents the rate-quality curves for two of the test sequences, *LampLeaves* and *CatRobot*. The BD-MOS values were computed by the procedure outlined in [28].



(a) LampLeaves (PSNR) (b) LampLeaves (VMAF) (c) LampLeaves (MOS)

(d) CatRobot (PSNR) (e) CatRobot (VMAF) (f) CatRobot (MOS)

Figure. 5.19 Rate-quality curves using PSNR, VMAF and subjective (MOS) scores of the proposed method with and without the CNN, and the anchor, HEVC (HM 16.14). Subjective tests (MOS) were performed using the proposed method with the CNN only. The error bars represent a 95% confidence interval.

Results presented show that the proposed method achieves significant improvements compared to HEVC with average BD-rate gains of 14.5% using PSNR and 21.2% using VMAF. In general, VMAF savings are always more pronounced which reflects a good subjective performance of this approach since VMAF correlates better with subjective scores [153]. Finally, subjective tests performed by the committee confirm the perceptual quality gains of the proposed spatial adaptation framework, with an average BD-MOS of 0.52.

The gains can also be observed in the rate-quality curves in Figure 5.19, which also show the comparison between the proposed model with and without the CNN. In the results of the subjective tests, for many rate-points, especially for lower bitrates, there is a clear perceptual advantage in applying the proposed approach, even considering the confidence interval.

The results also show that our method achieves higher gains for the cropped UHD compared to the HD test sequences, 17.7% BD-rate (PSNR) for the cropped UHD compared to 10.4% for the HD sequences. This can be justified by the increased spatial redundancy in higher resolution video content, which means that relevant less information is lost with the resampling process applied.

Compared to the experimental results of the proposed framework presented in Section 5.5, these show slightly higher savings due to the fact that that higher QP values were evaluated, which is beneficial for our adaptive resolution model.

5.7 Discussion

In this chapter, we have presented a framework for video compression which dynamically adapts the spatial resolution of input video frames using multiple downsampling ratios and reconstructs the full resolution video frames at the decoder using a deep learning based model. This approach shows significant BD-rate savings and perceptual gains compared to HEVC encoding at a fixed resolution and it is not restricted to low bitrate scenarios.

A large scale video training dataset consisting of 100 UHD video sequences collected from a variety of open-source video databases is used to train the Quantization-Resolution Optimization module and the super-resolution CNN. We believe this is one of the factors that contributes to our good performance given that it allows the models to learn from a diverse set of content.

By extracting spatio-temporal low-level visual features from the input video frames, we are able to predict a QP threshold which determines at each input QP value, the optimal resolution to encode the video. This approach allows the improvement of the rate-distortion characteristics of the encoded video without the need for pre-encoding and it is content adaptive.

Compared to using a linear filters, such as Lanczos3, to upsample the video frames at the decoder, the use of the CNN, although requiring much more computational complexity is able to improve the reconstruction quality measured by both pixel-based distortion metrics, perceptual quality metrics and visual quality comparisons.

Overall, our spatial resolution adaptation approach achieves BD-rate gains of 11.5 % based on PSNR and 15.3 % based on VMAF, for the sequences tested. Although these results are promising, there is room for improvement in several aspects including improving the accuracy of the QRO and developing more advanced deep learning techniques for image reconstruction.

Finally, the results of a previous version of our approach which was submitted to the Grand Challenge in Video Coding Technology at ICIP 2017 and won the first prize, was also presented in this chapter. In addition to objective metrics, the organizers of this challenge also performed subjective tests, which confirmed the perceptual benefits of our framework.

Chapter 6

Advanced and perceptually-inspired super-resolution for compressed videos

As seen in Chapter 5, the addition of a CNN for the super-resolution of video frames as part of the proposed resolution adaptation framework contributes to an improved reconstruction quality, measured by objective metrics. In addition, visual results presented show that this technique also enhances the visual quality by minimizing blurring and compression artifacts. Nevertheless, a simple network architecture and training methodology were used. In addition, no direct perceptual optimizations were considered when designing the system. For this reason, we have observed that, for some types of content, the generated super-resolution results are overly smooth and lack realistic texture detail. In the current chapter, we explore a variety of modifications to our proposed network in order to address these issues, inspired by latest advances in the area of single-image super-resolution. Several important concepts discussed in this chapter are introduced in Section 2.2.

6.1 Review of advanced super-resolution algorithms

Single-image super-resolution is currently a very active research topic in the field of computer vision, with a variety of proposed work in recent years [5, 6, 154, 7, 155, 8]. These works are focussed on three main goals: complexity reduction, aiming at achieving real-time performance; quality improvement measured by objective pixel-based distortion metrics, such as PSNR and; perceptual quality improvement, with the goal of producing more realistic and natural-looking images. The following sections provide an overview of the most important recent advances within these categories.

6.1.1 Complexity optimization

Improved performance in image reconstruction and other computer vision tasks have mainly been achieved by an increase in the total number of learned parameters of the proposed neural network architectures, more specifically, from the addition of more layers, increased number of filters and

parametric activation functions. This significantly affects the computational complexity for both training and inference (deployment). For this reason, a number of works have focused on reducing the computational complexity of these methods with the goal of achieving real-time speed without significantly affecting the reconstruction quality.

As described in Section 5.1, both SRCNN [5] and VDSR [6] employ pre-upsampling of the input image using bicubic interpolation before the image is processed by the network. This means that the all the processing at the convolutional layers is performed on images features with the same resolution as the High Resolution (HR) image. Given that the complexity of the network is proportional to the input feature size, this has a significant impact on the time required to apply the super-resolution operation. For example, when considering a network with 3 convolutional layers (with zero padding), the computational complexity, in big-O notation, can be calculated as follows:

$$O\{(f_1^2 \cdot n_1 + n_1 \cdot f_2^2 \cdot n_2 + n_2 \cdot f_3^2) \cdot w_{HR} * h_{HR})\}, \tag{6.1}$$

where f_i is the filter size and n_i is the number of filters of each layer i, and w_{HR} and h_{HR} are the width and height of the HR image. In addition to computational complexity, these operations also require considerable amounts of memory, which are also proportional to the input size.

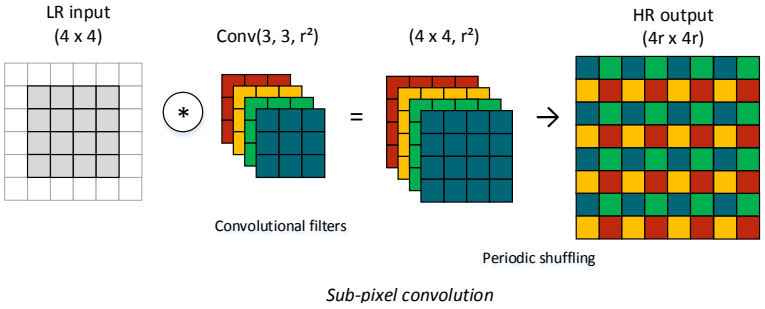


Figure. 6.1 Illustration of the sub-pixel convolutional operation by upsampling an input of size 3×3 using a ratio $r = 2$ to 6×6.

In order to reduce the number of computations and memory required, [156, 154] propose to process Low Resolution (LR) image features throughout the network and upsample them towards the end of the network rather than processing HR image features. According to Equation 6.1, for an upsampling ratio of 2, processing LR features would decrease the complexity by 4 times. This upsampling operation can be achieved by means of deconvolution layers (also known as transposed convolution or sub-pixel convolution) [157] or nearest-neighbour upsampling layers [158]. These types of layers aim at learning the mapping between the LR features and the HR features and can, therefore, be used to replace interpolation filters. In a convolution layer, filters are convolved with the input features using a stride k resulting in an output of size $1/k$ times of the input. In contrast, in a deconvolution layer, the output will be k times the input.

Figure 6.1 presents an illustration of an efficient implementation of deconvolution, know as sub-pixel convolution, proposed in [154]. In this example, the aim is to upsample the input feature matrix of size 4×4 using a ratio of $r = 2$ obtaining then an output of size 8×8. This is achieved by convolving the input with r^2 filters of size 3×3 (for example) with zero-padding and obtaining 4 feature maps which can, then, be shuffled to generate a single output of size $4r \times 4r = 8 \times 8$. By using these upsampling techniques, the authors of [156, 154] propose efficient networks that are able to achieve real-time performance without compromising quality compared to SRCNN.

6.1.2 Performance optimizations

At the same time, other works have focussed on optimizing the quality of the super-resolved images beyond the results obtained by SRCNN and VDSR using recently proposed developments in deep learning. In [7], Super-Resolution Residual Network (SRResNet) is presented. In addition to replacing bicubic interpolation of the input image by sub-pixel convolution, as described above, this network applies the concepts of residual blocks [159], skip-connections [160], batch normalization [161] and parametric ReLUs (PReLU) [59].

Deep residual networks, introduced in ResNet [159], provide a robust framework for training deeper networks and have become a milestone in the field. They help reduce the *vanishing gradient problem* [61], which is when the gradient at earlier layers become infinitely small due to multiple multiplications during back-propagation. This effect is usually found when training very deep networks consisting of a large number of stacked layers. Instead, Residual Networks (ResNet) have introduced the concept of residual blocks, which are units containing several convolutional layers, activation functions and skip connections providing a direct link between the input and the output of the block. This connection removes the need to learn the identity function. In the case of image reconstruction tasks, this is especially relevant since it helps to propagate low level features extracted from earlier layers to the output, as networks become deeper and, as a result, features become more sparse. A representation of a residual block is presented in Figure 6.2. Here, $\mathbf{x}$ is the input and RB($\mathbf{x}$) is the output of the residual block, which corresponds to the equivalent function produced the layers, F($\mathbf{x}$), added to the input.

Batch normalization (BN) [161] is a technique to stabilize the learning and improve the performance of deep neural networks. It normalizes the outputs of the previous layer by subtracting the batch mean and dividing by the batch standard deviation. This is similar to the normalization of the input features when those inputs contain different scales and variances, but it is employed at each layer. Given the batch mean, λ_B, and standard deviation, σ_B, of each feature, x_i, the normalized features, $\hat{x}_i$ are:

$$\hat{x}_i = \frac{x_i - \lambda_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{6.2}$$

where ϵ is a small constant added for numerical stability. In addition, to provide more flexibility to the network, BN also adds two trainable parameters, β and γ. The output of the batch normalization

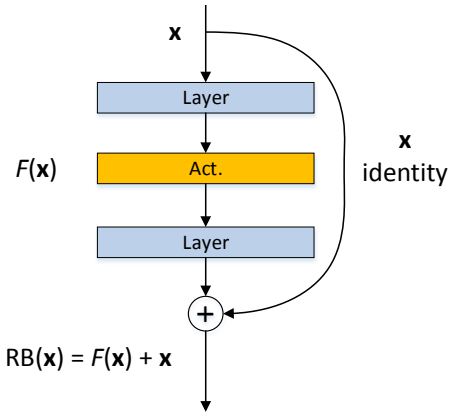


Figure. 6.2 Example of a residual block. The variable **x** represents the input, RB(**x**) is the output of the residual block and Act. is an activation function.

layer is then calculated as follows:

$$\mathrm{BN}_{\gamma,\beta}(x_i) = \gamma \hat{x}_i + \beta \tag{6.3}$$

This allows the network to learn the optimal values of γ and β and it is even possible to reverse the normalization of the features if this is the optimal case. During inference, instead of normalizing by the average of the mini-batch, the statistics on the entire training dataset are used.

The network structure of SRResNet is illustrated in Figure 6.3. It starts with a convolutional layer followed by a succession of 16 residual blocks, each containing two convolutional layers, BN layers and PReLU functions. After that, a single convolutional and a BN layers are employed. This architecture contains a skip connection between the output of the first activation layer and this last BN layer. All operations are performed on the LR image with zero padding and stride equal to 1 in order to preserve the size of the feature maps. Finally, upsampling is then conducted by the sub-pixel convolutional blocks at the end of the network. Compared to previously proposed approaches, this network provides state-of-the-art PSNR reconstruction quality on the conventional super-resolution test datasets.

Building on the work of SRResNet, an improved network, Enhanced Deep Super-Resolution (EDSR), is proposed in [155] and was the winner of the NTIRE 2017 super-resolution challenge. The first modification, compared to SRResNet is to remove all BN layers. The argument is that the normalization of the features by the BN reduces the flexibility of the trained models. In addition to improving performance, this modification reduces the required memory complexity. This allows the number of network layers to be increased, from 16 in SRResNet to 32 in EDSR and the number of filters to increase from 64 to 256. Furthermore, a technique known as residual scaling is applied. This consists of applying a multiplication by a small constant (e.g. 0.1) to the skip connections within a residual block.

Finally, also in [155], a multiscale network, MDSR, able to jointly reconstruct images using three upsampling ratios is proposed. The input consists of a single LR image while the output is three HR

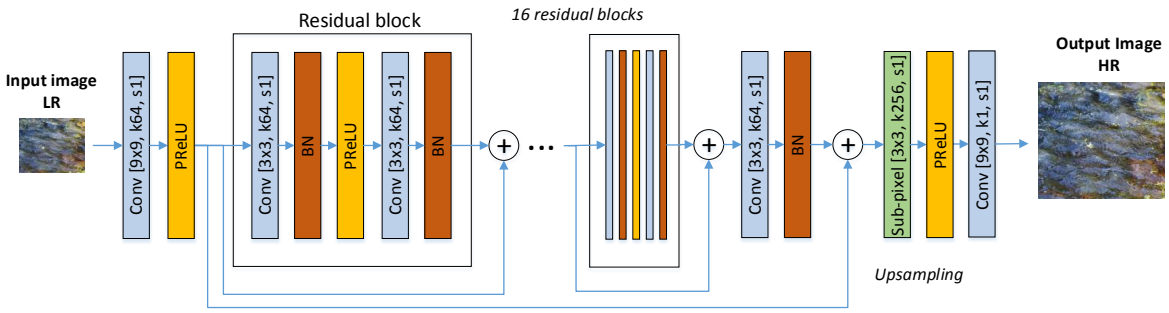


Figure. 6.3 Illustration of the network structure of SRResNet [7].

images at multiple resolutions. The advantage is that if multiple upsampling ratios are desired, only one network needs to be trained and employed instead of three separate ones.

6.1.3 Perceptually-inspired super-resolution

Although the previously described CNN-based super-resolution methods produce much higher quality images compared to previous interpolation approaches, there is still a large perceptual quality gap between the original and the super-resolved HR images. Often, the HR solutions obtained lack high frequency information, are overly smooth and have a poor representation of textured content. According to [162, 163], this problem is caused by the use of ℓ_1 or ℓ_2 loss functions, which promote results that are pixel-wise averages of possible solutions. In addition, it is well-known that PSNR does not correlate well with subjective image quality [29]

To overcome this issue, approaches based on perceptually-inspired loss functions have been proposed [162–164]. More specifically, in [164], the authors explore alternative loss functions based on the well-know perceptual quality metrics, SSIM [30] and MS-SSIM [165]. The limitations of ℓ_2 loss are presented, a small performance improvement is achieved by switching to ℓ_1 (sum of absolute differences). Finally, they propose a novel loss function, using a combination of MS-SSIM and ℓ_1, that provides improved quality accessed by objective metrics and visual inspection.

Using a different approach, in [162] and [163], a loss function based on high level features from a pre-trained deep network is proposed as a replacement for pixel-wise distortions. More specifically, the loss is calculated as the Euclidean distance in the feature space of a specific layer of a pre-trained CNN, VGG19 [166]. VGG19 is a popular network used for image classification tasks and had been pre-trained on the ImageNet dataset [167] (a dataset that contains millions of labelled images). To capture high-level concepts, the distance is typically computed using one of the layers of the network. The authors argue that a loss function based on high-level statistics will correlate better with perceptual quality compared to a loss based on pixel-wise differences. A low distance in this loss function means that the semantic content of the images is similar. Experimental results on several image reconstruction tasks show that this loss function provides more perceptually convincing results

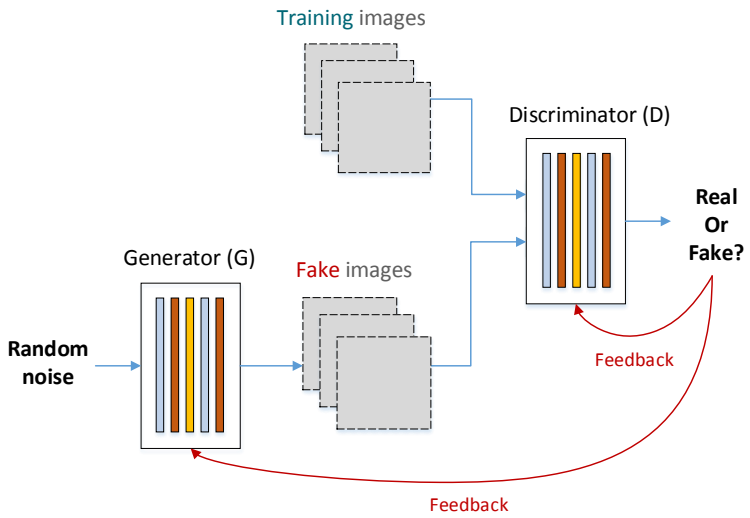


Figure. 6.4 Illustration of an Generative Adversarial Network (GAN) framework.

compared to models trained to optimize pixel metrics, although some grid-like artifacts are introduced [162, 163].

The next milestone for perceptual-based super-resolution resulted from the introduction of Generative Adversarial Networks (GAN) [168]. In this technique, a generative network, G, is trained to capture the data distribution of the training dataset at the same time as a discriminator network, D, is trained to distinguish between *real* samples from the training set and *fake* samples generated by the generator. This framework corresponds to a min-max game in which the goal is to train both networks to "fool" each other until their performances are strong and an equilibrium is achieved. GANs have been applied successfully for several computer vision tasks, including the generation of photo-realistic images [168] and image reconstruction [169, 7]. For the latter, the adversarial training is used to guide the learning of the model which, as a result, is able to reconstruct more realistic images.

Figure 6.4 presents an illustration of the structure of a GAN framework. For image generation tasks, the input of the generator is a noise sample that is used to produce an image which resembles images from the training dataset (without being an exact copy on a sample from the training dataset). F each input mini-batch, the generator produces "fake" samples which are then evaluated by the discriminator. At the same time, the discriminator also receives "true" images directly from the training dataset and is trained to distinguish between those two sets of images. The discriminator is then updated and provides feedback to improve the generator's ability to produce images that look real.

The loss functions used to train the generator and the discriminator are formulated using cross-entropy [61]. Given the probability of a sample image x being real, represented by $D(x)$, the discriminator aims at maximizing the cross-entropy of this probability for real images, from the training distribution p_{data} and minimizing this for fake images generated by the generator, $G(z)$ from

the random distribution p_z. Therefore, the objective function of the discriminator can be represented as follows:

$$\max_D V(D) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{6.4}$$

The generator, on the other hand, is trained to produce images that appear to be real to the discriminator, minimizing the second part of Equation 6.4, as follows:

$$\min_G V(G) = \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{6.5}$$

As proposed in [168], in order to improve the gradient behaviour, an alternative formulation of the generator loss is commonly used:

$$\min_G V(G) = \mathbb{E}_{z\sim p_z(z)}[-\log(D(G(z)))] \tag{6.6}$$

EnhanceNet [169] and SRGAN [7] are examples of two single super-resolution approaches that apply adversarial learning to recover HR images from LR versions. In both works, adversarial training is used in combination with a perceptual loss and a pixel-based loss (ℓ_2). This combination was shown to provide the best texture accuracy and, at the same time, avoid the generation of additional artifacts.

In SRGAN, the overall generator loss function, l_G corresponds to a weighted sum of the three losses:

$$l_G = w_1 \cdot \ell_2 + w_2 \cdot l_{VGG/i.j} + w_3 \cdot l_{Adv.} \tag{6.7}$$

where $l_{VGG/i.j}$ is the perceptual feature loss, described above, calculated at the j-th convolution before the i-th maxpooling layer of the pre-trained VGG19 network, and $l_{Adv.}$ is the adversarial loss as formulated in Equation 6.6. The weights used by SRGAN are $w_1 = 1$, $w_2 = 1/12.75$ and $w_3 = 10^{-3}$.

The generator of SRGAN follows the same network architecture of SRResNet, described in Section 6.1.2, whereas the generator in EnhanceNet contains only 10 residual blocks, uses nearest neighbour layers to perform the upsampling and applies residual learning by adding the network output to the upsampled image using bicubic interpolation.

Both approaches are able to produce high quality images even considering high upsampling ratios (e.g. 4×). Although the PSNR values between the original and the generated HR images is lower compared to other networks optimized for pixel-based distortions, such as SRResNet, the resulting images are sharper, preserve much higher frequencies and textures, and were shown to be more visually pleasing to human critics based on subjective experiments conducted. Out of the two techniques, SRGAN has been shown to produce fewer artifacts compared to EnhanceNet, which tends to generate over-textured images.

Recently, a modified version of SRGAN, Enhanced SRGAN (ESRGAN), presented in [8], further improves the visual quality of the super-resolved images. Four modifications to key components are introduced: Residual blocks are replaced by the Residual-in-Residual Dense Blocks (RRDB) in the generator network; BN is removed from the network;the conventional GAN framework is replaced

by a relativistic GAN [170]; and improvements to the perceptual loss are explored by extracting the features before the activation function, instead of after. The proposed model is able to produce more natural looking images with improved texture recovery compared to SRGAN. This approach won the first place at the PIRM2018 super-resolution challenge [171], held in conjunction with ECCV in 2018. The metric used to access the visual quality in this challenge was a Perceptual Index (PI) computed from two non-reference quality metrics, Ma's [172] score and NIQE [173]. A lower PI represents better perceptual quality.

Figure 6.5 shows a visual comparison of the super-resolution results obtained using bicubic interpolation and a variety of CNN-based techniques: EDSR, EnhanceNet, SRGAN and ESRGAN. These results were obtained using an upsampling ratio of 4. Sample images from two popular test datasets, BSD 100 and Set 14, are considered. These results were extracted from [8]. In addition to the PSNR value of each image, the figure also contains PI, mentioned above. It can be seen that the PSNR of the reconstructed images is the highest for EDSR, which was trained using pixel-based loss. However, their generated images lack details and are overly-smooth. On the other hand, the methods in the second row, based on a GAN framework, are able to produce more realistic and natural results, although some artifacts that are not present in the original HR image are introduced. These are also the methods with the lowest PI (which indicates higher perceptual quality).

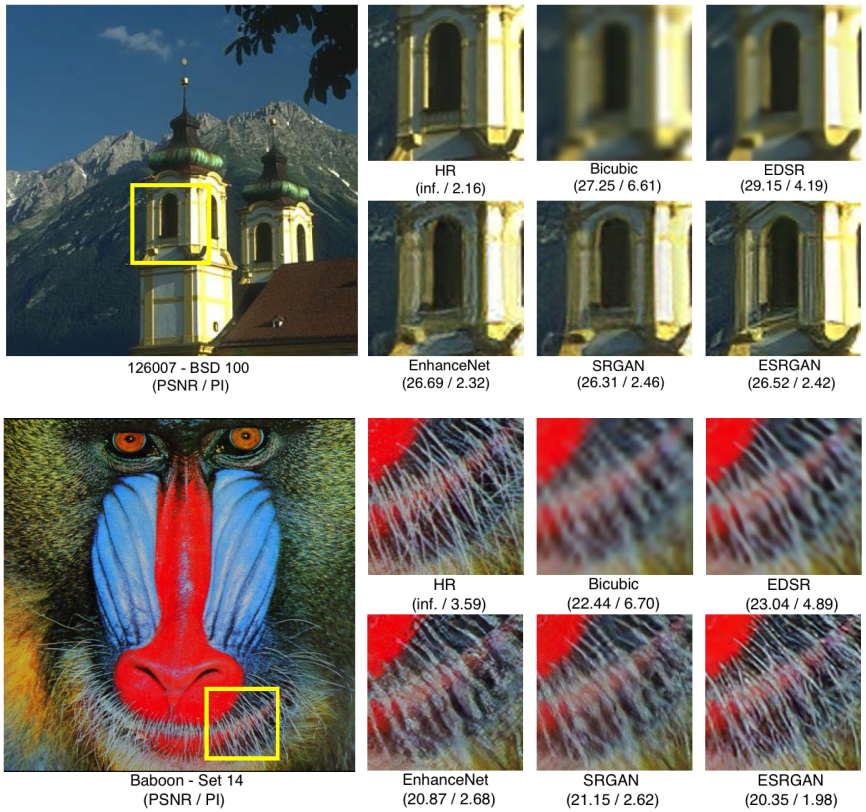


Figure. 6.5 Visual comparison of the results of a variety of single-image super-resolution approaches for two sequences from BSD100 and Set14 (upscaling ratio is 4×). Results from [8].

6.2 Proposed improvements for compressed super-resolution

The following sections explore modifications to the compressed super-resolution CNN employed within our resolution adaptation framework, proposed in Chapter 5. We aim at improving the video quality based on two different types of assessment: pixel-wise distortion metrics, such as PSNR (or MSE), and perceptual quality using visual inspection and perceptually-inspired quality metrics. The target is to produce natural-looking videos with realistic details and textures. Depending on the desired application of the proposed video compression framework, different trained networks can be employed.

6.2.1 Distortion-based optimizations

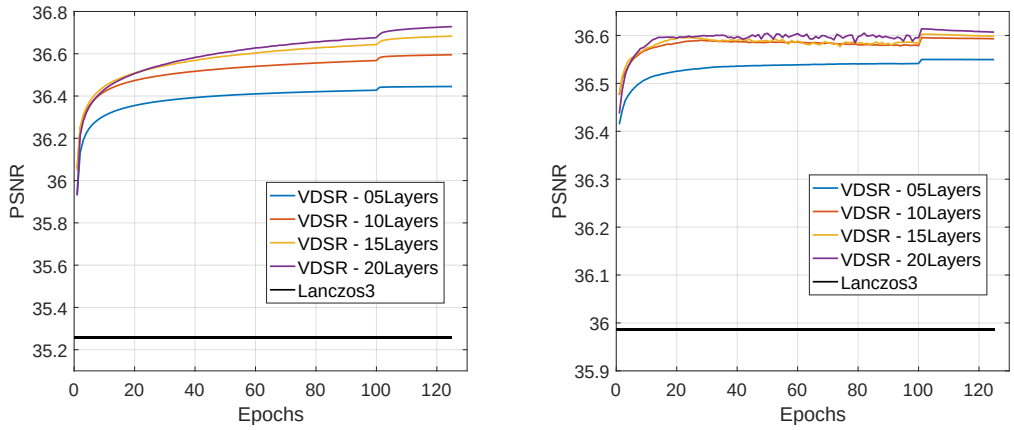
6.2.1.1 Number of layers

The network structure of VDSR contains 20 convolutional layers. This number was found by the authors to be optimal for their architecture and for the task of single-image super-resolution. However, considering the problem addressed in this work is compressed images, the optimal number of layers might differ. In addition, as seen in Section 6.1.1, the complexity of the network is directly proportional to the number of convolutional layers and therefore, it might be beneficial to limit the size of the network. In this section, experiment with different number of convolutional layers and evaluate their performance.

As explained in Section 5.4.1, we produce a total of four training datasets comprised of two downsampling ratios, 1.5 and 2, and two quantization levels, more specifically, QP 25 and QP 35. In order to reduce the number of possible combinations to evaluate, we limit this analysis to the models trained using a downsampling ratio of 2. Furthermore, in order to understand the generalization of the models, the quality assessment is performed using the PSNR values on both the training datasets and the testing datasets. The latter is applying the models on the 14 test sequences presented in Section 5.5.1, which had been subject to the same scaling and compression as the training datasets. The PSNR calculated on the testing datasets is an indication of the performance of the trained CNN on sequences that have not been used to learn the networks' parameters. In the following experiments, the reported PSNR values are computed in the luminance (Y channel) only.

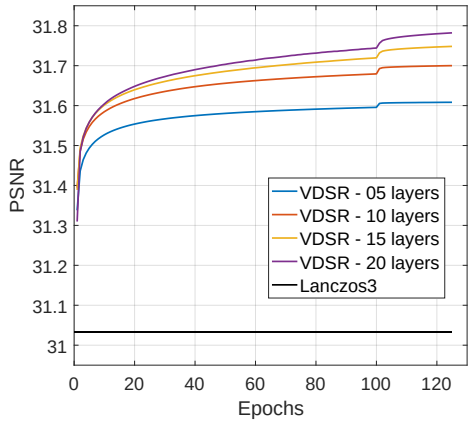
The plots in Figure 6.6 show the variation of PSNR on the training and testing datasets for the low and high quantization levels using networks with 5 to 20 layers. In addition, for comparison purposes, the plots also show the quality achieved by the Lanczos3 filter. By analysing the plots, it can be seen that for all experiments, after only a single epoch, the CNN models are able to achieve higher PSNR values than Lanczos3. This is due to the use of residual learning, in which the network only needs to predict the residual image. The performance on the training dataset then increases steadily and does not reach the saturation point, even after 120 epochs. The boost in performance at 100 epochs is due to the reduction in the learning rate, which allows the model to get out of a local minima.

Different behaviour is found for the testing datasets, which shows that the saturation is achieved much quicker, and for some of the models, the performance decreases after a certain number of epochs. In particular, for the high quantization level in Figure 6.6 (d), the CNN with 20 layers achieves the highest PSNR value at epoch 30 and then gradually decreases quality as more training is performed. This behaviour can be explained by overfitting or overtraining and indicates that training show stop at this point.

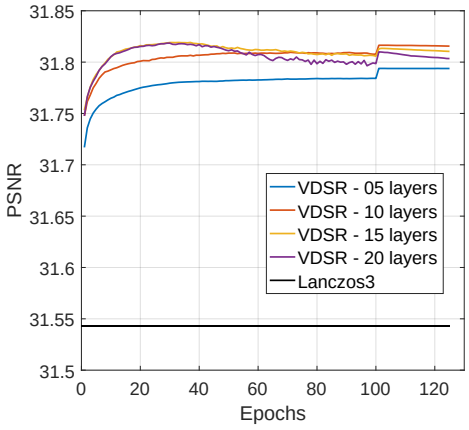


(a) PSNR of training set, low QP

(b) PSNR of test set, low QP



(c) PSNR of training set, high QP



(d) PSNR of test set, high QP

Figure. 6.6 Variation of the reconstructed PSNR during training the VDSR-based CNN using a different number of layers, from 5 to 20. Quality is assessed on both the training dataset (left) and on the test dataset (right) and at different quantization levels.

For the training datasets, Figures 6.6 (a) and 6.6 (c), it can be seen the model with 20 convolutional layers achieves the highest quality compared to models with lower number of layers. For example, for the high QP dataset, the model with 5 layers only achieves a PSNR of 36.4 dB, while the 20 layer model reaches 36.7 dB. When considering the PSNR evaluated on the testing dataset, Figure 6.6, little difference is found between the peak performance of the models with 10, 15 and 20 layers for the low QP range. However, for high QP range, the models with 15 and 20 layers outperform the rest.

Moreover, it is also interesting to notice the differences between the high and low quantization levels. For low quantization, the CNN achieves approximately 0.6 dB gain in PSNR compared to Lanczos3. In addition, there is little overfitting for all networks tested. However, for the high quantization level, the performance improvement achieved is below 0.3 dB. From these figures, we theorize that the task of recovering details from the LR frames is harder for higher quantization levels due to two reasons: the reduced amount of information due to loss of high frequencies and the variability of the compression artifacts introduced by the encoding process.

6.2.1.2 Additional optimizations

As seen in Section 6.1.2, several optimizations have been proposed for image reconstruction networks, in particular, BN and the use of ℓ_1 loss instead of ℓ_2. BN has often several benefits for deep neural networks including the promotion of faster learning and the use of higher learning rates [161]. In SRResNet [7], the network architecture includes BN layers after every convolutional layer. However, in [155], the authors found the use of BN to affect negatively the performance for image super-resolution due to the decrease in variability of the image features after the normalizations. At the same time, several works [155, 8, 163] have reported higher quality in image reconstruction tasks by replacing the popular ℓ_2 loss with ℓ_1 loss. Although minimizing ℓ_2 is a direct minimization of PSNR, empirical results show that higher PSNR values are achieved when minimizing ℓ_1 [163].

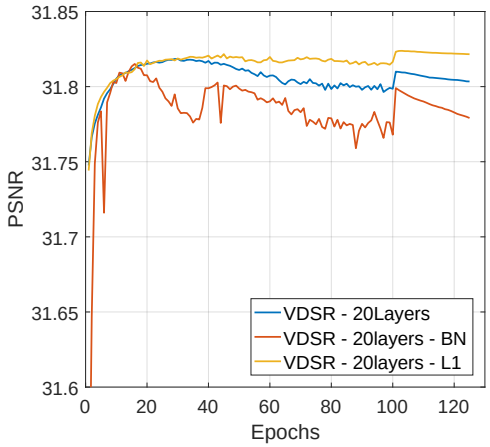


Figure. 6.7 Variation of the reconstructed PSNR training of the proposed CNN using the VDSR architecture using 20 convolutional layers with the following modifications: ℓ_1 loss and BN.

In order to understand the impact of these techniques for our problem, an experiment was conducted. We use the same network architecture, based on VDSR, using 20 convolutional layers and restrict the analysis to the high quantization model with an upsampling ratio of 2. Figure 6.7 shows the performance of the three models trained: the original network, the network with added BN layers

every all convolutional layers and the network optimized using ℓ_1 loss. Here, the PSNR values are calculated on the test dataset.

It can be seen that the model trained with the addition of BN achieves a much lower PSNR value compared to the other models in the early training stages and exhibits unstable performance after a few epochs. In contrast, the network trained using ℓ_1 loss has an initial performance similar to the one trained with ℓ_2 (original), but later outperforms the others. It also shows less of an overfitting behaviour with more training iterations. From the results of these experiments, it can be concluded that the network achieves the highest reconstruction quality without BN and using the ℓ_1 loss function.

6.2.2 Network architecture

Inspired by previous work in the field of for single-image super-resolution, including VDSR [6], SRResNet [7] and EnhanceNet [169], and the conclusions of our own experiments, described above, we propose a new architecture for the super-resolution of compressed videos, which we named Compressed Super-Resolution CNN, CSRCNN. Figure 6.8 presents an illustration of our network architecture, in which *r* represents the upsampling ratio.

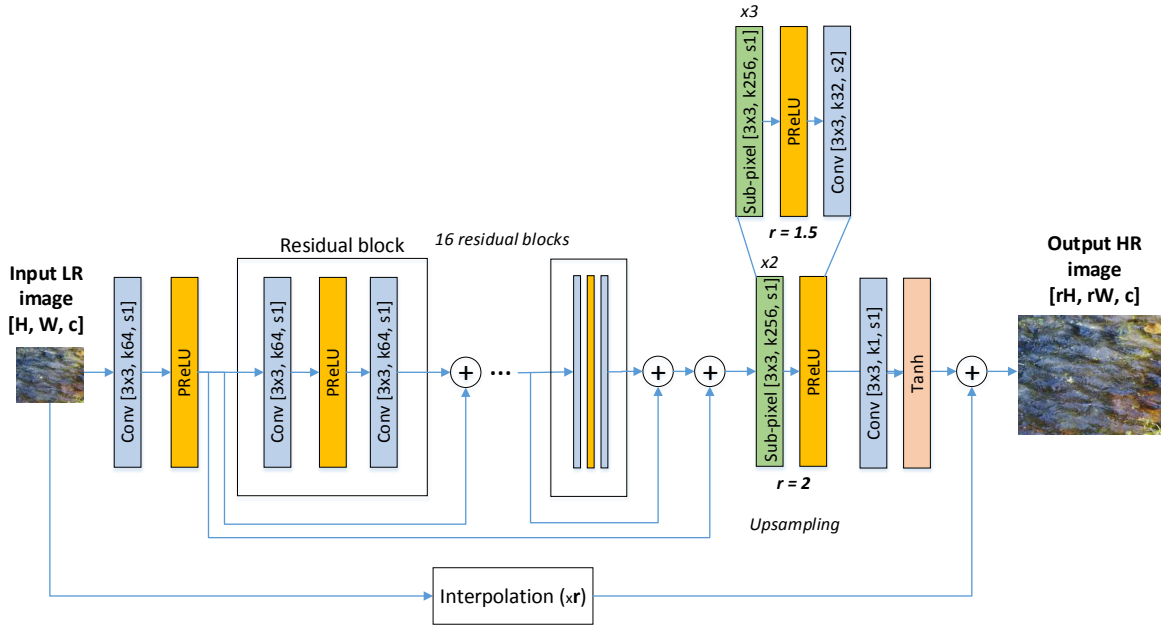


Figure. 6.8 Proposed CNN architecture for compressed image super-resolution.

The first modification is to employ Residual Blocks (RB) without BN, as we have found the latter not to be beneficial. Next, we use Sub-pixel convolutional layers to upsample the features after the residual blocks. By using these layers, the network is able to learn the mapping between LR and HR features. Additionally, as discussed in Section 6.1, by employing these layers towards the end of the network, the majority of the processing is done in the LR space which helps keep the computational

complexity low. Additionally, we apply residual learning by first interpolating the input LR image to the HR and adding it to the network output. The interpolation is performed using bicubic interpolation. The reason that Lanczos3 is not used in this case is twofold: to the best of our knowledge, there is no efficient implementation of Lanczos3 filter that can be easily integrated with Tensorflow 1.2; and according to [164], using a more complex filter might reduce reconstruction ability of the CNN by introducing high frequency artifacts.

Since our framework supports two scaling ratios, these require network modifications in the modules that are responsible for the upsampling. More specifically, in order to upsample using a fractional ratio of 1.5, a Sub-pixel convolutional layer first upsamples 3× and then an extra convolutional layer with stride 2 is used to decrease the resolution 2× to the desired HR, as can be seen in Figure 6.8. Finally, we use PReLU throughout the network and a *tanh* activation in the output layer.

6.2.2.1 Comparison with other architectures

Figure 6.9, we compare the performance of our proposed architecture, CSRCNN, with other architectures proposed in the literature, more specifically, VDSR and SRResNet with a few modifications concerning the use of BN and ℓ_1 loss function. As in most of the experiments presented in Section 6.2.1, all CNN models were trained using sequences from the training set that were downsampled using a ratio of 2 and compressed using HM 16.18 with a base QP of 35 (high quantization level). The quality is then evaluated by PSNR on the test dataset.

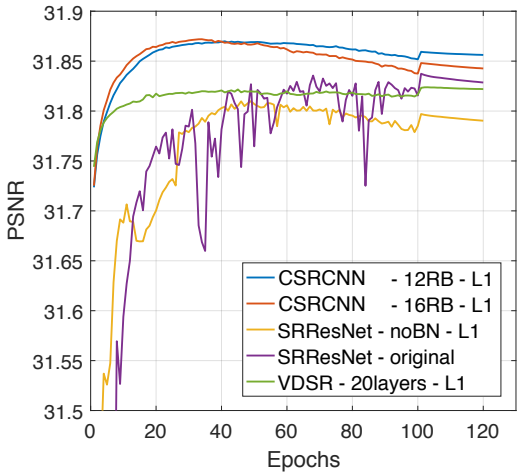


Figure. 6.9 Variation of the reconstructed PSNR of different network structures, including the proposed CSRCNN, during training evaluated on the test dataset with downsampling ratio of 2 and the high quantization level.

It can be seen that VDSR with ℓ_1 loss achieves a good reconstruction quality and saturates after a few epochs at around 31.82 dB. In contrast, SRResNet starts with a much lower quality but

then increases and at around 70 epochs, achieves slightly higher quality than VDSR. However, its performance is quite unstable between training epochs. Our proposed networks, CSRCNN, with 16 and 12 residual blocks, on the other hand, achieve high quality with very little training iterations and quickly outperform the other models peaking at 31.87 dB after 40 epochs. Moreover, the network with 16 RBs slightly outperforms the one with 12 RBs. For this reason, we decide to employ 16 RBs for our system and train only until the model begins to show overfitting behaviour on the training data.

6.2.2.2 Luminance only vs YUV vs RGB

In all CNN models described until this point, only the luminance (Y) channel was processed by the trained models, while the chroma channels were upsampled using the Lanczos3 filter. The decision to not apply the CNN to the chroma channels was justified by the fact that the Human Visual System is highly sensitive to changes in intensity but has a lower acuity for variations in color. Therefore, in theory, increasing the quality of chroma channels of the output video frames should result in reduced benefits to the overall perceptual quality of the video. Nonetheless, in order to fully understand the possible benefits, we redesign our proposed network to process chroma components and evaluate the quality using per channel PSNR values (PSNR-Y, PSNR-U and PSNR-V).

When considering the chroma channels, it is important to notice the effect of chroma subsampling, a common practice in video coding, where the chroma channels are represented using a lower spatial resolution compared to the luminance channel. In our case, our videos use 4:2:0 chroma subsampling, which means that the U and V channels contain one fourth of the pixels of the luminance channel. Our proposed CNN contains an interpolation step (residual learning), applied to all channels. As mentioned above, bicubic interpolation is used for the Y component. In contrast, we have empirically found that higher quality was achieved by applying nearest neighbour for the chroma channels. This might be explained by the fact that the chroma channels contain less information and the use of a more complex interpolation filter, such as bicubic, would introduce high frequency artifacts making it harder for the network to reconstruct the original signal.

In the super-resolution literature, the inputs and outputs of the CNN usually correspond to either the Y component only (2-D matrix) or the three RGB components of the image to be scaled (3-D matrix). In the case of video frames, especially ones that have been previously chroma subsampled, we believe that a better representation is to consider the YUV colour space. Since the U and V channels have lower resolution than the Y, they need to be scaled to the same resolution to form a 3-D matrix of inputs and outputs for training the networks. However, not all pixels of chroma component of the output matrix are required when computing the network loss (due to the chroma subsampling). Therefore, a different strategy is proposed for the loss function. Instead of considering the average absolute difference (SAD) on the three components, required for the ℓ_1 loss, for the chroma channels,

only 1/4 of the pixels are used to compute the absolute value. The SAD is then computed as follows:

$$SAD_Y = \frac{1}{W_Y \cdot H_Y} \sum_{x=0}^{W_Y-1} \sum_{y=0}^{H_Y-1} I_Y(x,y) \tag{6.8}$$

$$SAD_U = \frac{1}{W_U/2 \cdot H_U/2} \sum_{x=0}^{W_U/2} \sum_{y=0}^{H_U/2} I_U(2x,2y) \tag{6.9}$$

$$SAD_V = \frac{1}{W_V/2 \cdot H_V/2} \sum_{x=0}^{W_V/2} \sum_{y=0}^{H_V/2} I_V(2x,2y) \tag{6.10}$$

$$\ell_{1_{YUV}} = 4 \cdot SAD_Y + SAD_U + SAD_V \tag{6.11}$$

where W_Y, H_Y, W_U, H_U, W_V and H_V are the width and height of the Y, U and V components, respectively.

This means that the network only needs to learn to minimize the difference between the original chroma samples at a lower resolution instead of the scaled versions. Notice also that in Equation 6.11, the weight of the luminance component is 4 times the weight of the chroma, which encourages the network to optimize more for the Y channel compared to the chroma due to the higher number of samples used to represent luma.

For these experiments, we consider the same training and testing datasets as previous experiments, but compute PSNR on the first intra period of the test sequences only. Results can be found in Table 6.1, which shows the average PSNR per channel on the test dataset achieved by: Lanczos3 (L3), the model trained using Y channel input (Y-CNN), RGB input (RGB-CNN) and YUV input (YUV-CNN). As mentioned before, we apply Lanczos3 to upsample the U and V channels when using the Y-model, which explains why the PSNR values are the same for Lanczos3 and the Y-CNN model for U and V.

Table 6.1 Comparison between PSNR values calculated separately on the YUV channels achieved by Lanczos3 and two CNN models, one trained to minimize the loss on the Y channel only (Y-CNN) and another trained to minimize the loss on RGB components (RGB-CNN). For Y-CNN, the U and V channels are upsampled using Lanczos3 (L3).

Model	**PSNR-Y [dB]**	**PSNR-U [dB]**	**PSNR-V [dB]**
L3	32.233	38.012	38.241
Y-CNN	32.577	38.012	38.241
RGB-CNN	32.502	38.365	38.535
YUV-CNN	32.530	38.545	38.645

On average, for the sequences considered, the Y-CNN model outperforms the rest in terms of PSNR calculated on the luminance channel by a small margin, more specifically, 0.05 dB. However,

(a) Lanczos3 (b) YUV-CNN model (c) Original

Figure. 6.10 Comparison between the reconstructed U chroma channels of a patch of *CatRobot* using (a) Lanczos3 and (b) YUV-CNN and the original uncompressed (c). Corresponds to the lowest bitrate rate-point tested.

with regards to the chroma components, reconstruction quality is improved by approximately 0.5 and 0.4 dB for U and V, respectively, when applying the YUV-CNN model compared to applying the Lanczos3 filter. Also, comparing those the YUV-CNN and the RGB-CNN models, it can be seen that the YUV-CNN model outperforms the RGB-CNN model by a significant margin for the chroma components.

In addition to evaluating the PSNR values, Figure 6.10 presents a visual comparison between the reconstructed U channel by the means of the Lanczos3 filter and the YUV-CNN for the test sequence *CatRobot*. It also shows the original uncompressed patch for reference. By visual inspection, it can be seen that the YUV-CNN reconstructed patch contains more details, sharper edges are is closer to the original compared to the one upsampled using Lanczos3. We believe that the additional details obtained are due to the fact that the model uses the all components together to obtain the final HR frame, and therefore, the U and V reconstruction benefits from information obtained from the Y component.

In conclusion, it is important to consider the colour format used to process the decoded frames at the decoder. Although less important, results show clear benefits in applying a joint training of YUV components. However, the model trained using Y only achieves a slightly higher Y-PSNR, due to the fact that it only needs to minimize the loss on one channel.

6.2.3 Experimental results

The networks described above, trained using Y-CNN and YUV-CNN inputs were integrated into our proposed spatial resolution adaptation framework for video compression, presented in Chapter 5. In this section, we present experimental results for these approaches compared to encoding using HM 16.18 at a fixed resolution. For these results, the same test methodology applied in Section 5.5 was used. In addition, since the only modification compared to Chapter 5 was the CNN model at the decoder, optimal resolution encoding decisions performed by the QRO module remain the

same. Evaluation of the video quality was accessed using per channel PSNR and VMAF (which only considers the Y component). BD-rate calculation results for each metric can be found in Table 6.2 for all test sequences.

It can be seen that the Y-CNN model achieves the highest gains for the Y channel, with an average BD-rate of -13.06% based on PSNR and -17.19% based on VMAF. Compared to the previous CNN models presented in Chapter 5, this is an improvement of approximately 1.5% and 2.2%, for PSNR and VMAF, respectively. On the other hand, the YUV model reports gains of approximately 11.7% for PSNR-Y, which are not as high as the Y only model. For the chroma channels, an average BD-rate loss of 9.72% for U and 12.97% V are reported when using the Y model. We believe that this is due to the use of the linear interpolation using Lanczos3 and due to the fact that the chroma channels are easier to encode and would require a much higher compression level to equalize the distortion caused by the spatial rescaling. Nonetheless, when trained to jointly minimize the loss on all three components, the YUV-CNN model is able to achieve considerable improvements compared to using the Lanczos3 filter and actually achieves gains of 5.38% and 5.62% for U and V components, respectively.

For this reason, although the quality of the chroma channels are not as important as the quality of the luminance channel, overall, given the experimental results, the best solution to upscale a compressed video is to use the YUV-CNN model. In addition, it can be seen that the modifications to the network architecture and training techniques have led to an improvement in reconstruction quality compared to the network employed in Chapter 5.

6.2.4 Perceptual quality optimizations

In previous sections, we were concerned with achieving the highest possible quality measured by PSNR between the original video frames and the reconstructed video frames. As the literature has reported [8], when optimizing convolutional neural networks for a pixel-based loss, such as PSNR, the resulting super-resolved frames contain sharp edges, but might lack high frequency detail such as textures. In the case of the super-resolution of compressed videos, we have found that this is also the case. This effect is reflected by videos that might have a “fake” or “animated”-like appearance, as the example patch presented in Figure 6.11 from the sequence *DaylightRoad*. For this reason, we further explore alternative ways to produce natural-looking reconstructed videos using perceptually-inspired deep learning techniques.

6.2.4.1 Generative Adversarial Networks for compressed videos

The current state-of-the-art technique in the field of image reconstruction and enhancement is the use of Generative Convolutional Networks (GAN). When applied for image or video reconstruction, this approach is similar to texture synthesis given that it attempts to hallucinate texture details, using a learning methodology. Therefore, the proposed adaptive resolution with intelligent resampling at the decoder resembles the analysis-synthesis video compression frameworks described in Chapter 3, which skips the encoding of selected regions in a video and reconstruct those at the decoder using

Table 6.2 Experimental results measured by the per channel PSNR and VMAF BD-rates between the proposed Y-CNN and YUV-CNN compared to the anchor, fixed-resolution HM 16.18, for the 14 test sequences.

	Y-model				YUV-model			
#	PSNR-Y [%]	PSNR-U [%]	PSNR-V [%]	VMAF [%]	PSNR-Y [%]	PSNR-U [%]	PSNR-V [%]	VMAF [%]
S01	-15.75	28.42	30.95	-15.53	-14.90	-4.57	-14.98	-15.66
S02	-14.47	7.58	10.05	-17.64	-12.90	-5.52	3.37	-15.15
S03	-8.34	1.13	3.41	-13.85	-7.47	-11.70	23.09	-12.54
S04	-20.91	5.33	8.96	-26.55	-18.94	-13.36	-10.89	-23.50
S05	-15.68	4.50	23.56	-15.53	-14.63	16.74	-25.73	-15.12
S06	-13.92	21.30	23.12	-17.69	-11.45	-8.93	-20.92	-15.96
S07	-11.19	2.86	1.50	-17.95	-9.99	2.55	-13.73	-17.18
S08	-12.61	2.50	13.51	-19.66	-10.11	-17.36	6.86	-17.84
S09	-23.50	2.42	0.77	-29.64	-22.17	-11.65	-10.62	-27.05
S10	-12.31	9.65	10.26	-18.73	-11.17	-4.26	-14.06	-16.97
S11	-11.79	14.93	10.54	-9.79	-10.99	-24.20	-9.66	-10.18
S12	-1.41	6.02	7.16	-8.40	-0.58	6.00	10.93	-7.59
S13	-14.39	10.32	3.12	-16.38	-13.34	3.90	-9.95	-15.33
S14	-6.54	19.12	34.62	-13.33	-5.59	-2.98	7.64	-10.84
avg.	-13.06	9.72	12.97	-17.19	-11.73	-5.38	-5.62	-15.78

(a) Original patch (b) Reconstructed patch

Figure. 6.11 Comparison between an original (left) and a reconstructed patches of *DaylightRoad* using the proposed CNN-YUV (right) for the downsampling ratio of 2 and the lowest bitrate rate-point.

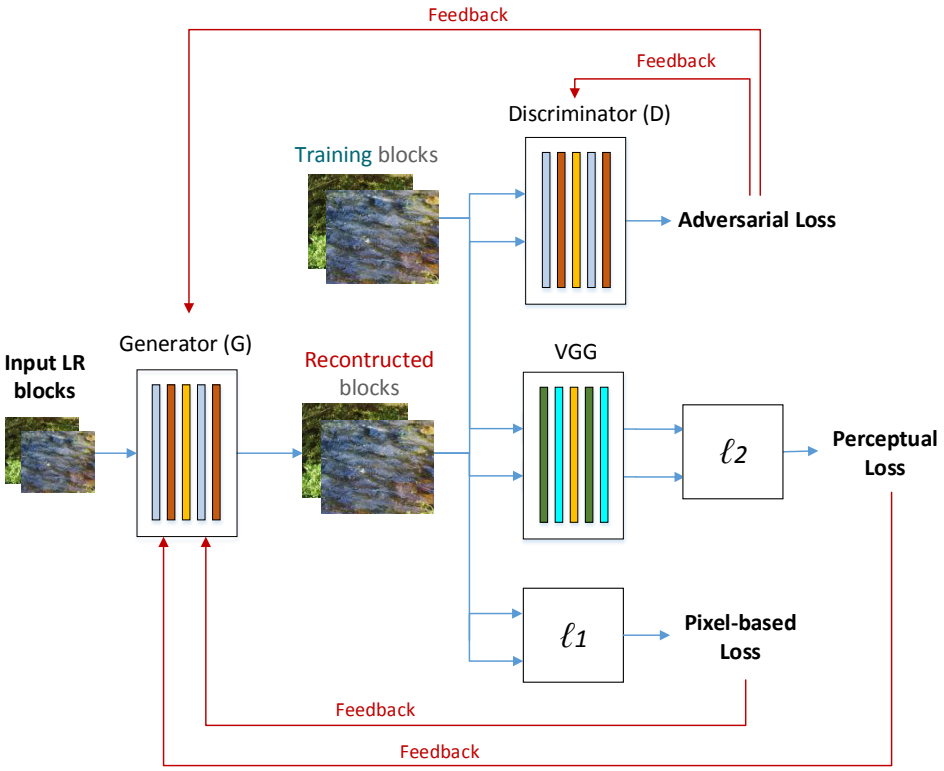


Figure. 6.12 Illustration of the proposed GAN framework for compressed super-resolution.

texture synthesis. Likewise, the current work attempts to remove the spatial redundancy in the input video frames by dynamically encoding at a lower resolution, depending on the characteristics of the video, and then reconstructs the missing information, after decoding by applying a deep convolutional neural network. These offer an efficient alternative to texture synthesis, given their ability to learn complex functions from training examples.

The proposed GAN framework, inspired by SRGAN and EnhanceNet, is presented in Figure 6.12. It contains a generator and a discriminator network. The generator is trained using a weighted function of three individual losses. The following sections are dedicated to describing the proposed generator, discriminator, loss functions and training techniques. Experimental results using objective metrics and visual comparisons are then presented. Finally, the limitations of the proposed system are discussed.

6.2.4.2 Generator

We apply the network proposed in Section 6.2.2 and illustrated in Figure 6.8 as the generator (G), which contains 16 residual blocks without BN, PReLU activation functions, sub-pixel convolutions and predicts the residual to be added to an interpolated result provided. In the adversarial training,

we have empirically found that the model trained using RGB input/output pairs produces better visual results compared to the model trained using YUV samples, in contrast to what observed when optimizing for PSNR.

6.2.4.3 Discriminator

Figure 6.13 shows the proposed discriminator architecture, which is similar to the one employed in SRGAN, however, it contains fewer layers and has an overall lower complexity. This modification was intended to decrease the influence of the discriminator, which in our experiments, had an overpowering effect on the generator during the training process. The proposed discriminator contains 6 convolutional layers, with 3×3 kernels and an increasing number of filters from 64 to 256. As the number of filters increases, the size of the features decreases by the use of strided convolutions (stride = 2). Unlike the generator, it uses LeakyReLU and BN after most convolutional layers.

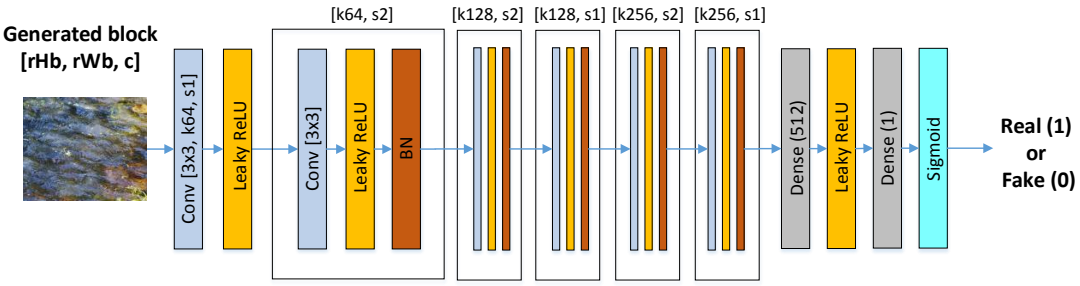


Figure. 6.13 Structure of the proposed discriminator used for the GAN framework.

In order to achieve more stable training, we further explored the use of a recently proposed GAN alternative, known as Relativistic GAN (RaGAN) [170]. In comparison with the conventional GAN, in which the discriminator tries to distinguish between images from the training dataset and generated images, RaGAN estimates the probability that a training image is more realistic than a generated image. In [8], this has been shown to provide higher perceptual quality with an improved reconstruction of edges and more detailed synthesised textures.

Given the discriminator output, $C(x)$, the standard discriminator takes the sigmoid function of this output, $D(x) = \sigma(C(x))$. In contrast, the RaGAN output is formulated as $D_{Ra} = \sigma(C(x_r) - \mathbb{E}_{x_f}[C(x_f)])$, in which x_r is a training or "real" image, x_f is a generated or "fake" image and $\mathbb{E}_{x_f}$ is the operation of taking the average of the network outputs of all fake images in a mini-batch. The discriminator loss, L_D^{Ra}, then becomes:

$$L_D^{Ra} = -\mathbb{E}_{x_r}[\log(D(x_r, x_f))] - \mathbb{E}_{x_f}[\log(1 - D(x_f, x_r))] \tag{6.12}$$

and the adversarial loss for the generator, L_G^{Ra}, is given by the following equation:

$$L_{Adv.} = L_G^{Ra} = -\mathbb{E}_{x_r}[\log(1 - D(x_r, x_f))] - \mathbb{E}_{x_f}[\log(D(x_f, x_r))] \tag{6.13}$$

The adversarial loss of the generator contains both training images and generated images which means that the generator learns from gradients provided by both types of data instead of only learning from the generated data. In our experiments, we have found that, although the effect was not very significant, using the relativistic GAN helped to stabilize the training process.

6.2.4.4 Loss function

We combine three loss types, a pixel-based loss measured by ℓ_1 loss, a perceptual loss using the distortion calculated on the feature space of a layer of a pre-trained VGG19 convolutional network, $L_{VGG/5.4}$, and finally, the adversarial loss, $L_{Adv.}$, described previously. These losses work in different ways to help the generator learn a more perceptually relevant reconstruction results compared to using a pixel-based distortion. The full loss function for our proposed generator, is then represented by Equation 6.14:

$$L_G = 0.0025 \cdot l_1 + 0.001 \cdot L_{VGG/5.4} + 0.05 \cdot L_{Adv.} \tag{6.14}$$

The different weights assigned to each individual loss are intended to normalize the losses to a similar range. The goal of the perceptual loss, $L_{VGG/5.4}$, is to provide a measure of the difference in terms of high level concepts between the content represented in the generated frames and the original frames. It is calculated from taking the MSE on the features at the 4th convolutional layer and the 5th maxpooling layer, and is generally represented as VGG19_54. As in [8], we extract the features before the activation function.

6.2.4.5 Training methodology

The training of the proposed GAN for scaled and compressed videos is conducted in a similar way as the pixel-based models described in Section 5.4.1 in which pairs of blocks extracted from the original and downsampled compressed blocks are used as the input and reference output of the generator and discriminator networks. Training of the GAN is performed in two steps: first, the generator is trained independently using only the ℓ_1 loss until it reaches a good performance measured on the test dataset. This training is indented to improve the potential of the GAN by providing a "heads-up" to the generator before starting to jointly train the discriminator and has been applied in the literature. Then, starting from the parameters learned from the pre-training of the generator, a joint training of the generator and the discriminator within the GAN framework is performed.

The proposed GAN takes a total of around 1.5 days for training, including approximately 14 hours for the generator pre-training, for 50 epochs, and 19 hours for the adversarial training, for another 50 epochs. These figures were obtained on a NVIDIA 1080ti GPU. In addition, the following

hyper-parameters were used: batch size of 16, 96×96 LR input blocks, Adam optimizer [64] with beta1=0.9 and beta2=0.999 and a fixed learning rate of 1e-4.

6.2.5 Experimental results

The network trained using the described GAN framework was employed on the 14 test sequences described in Section 5.5.1 for the highest quantization rate-point of the experimental design employed, which corresponds to a downsampling factor of 2 and a QP of 35. Our evaluation was restricted to this rate-point since it provides a good basis for improving the perceptual quality and contains both a high level of scaling and compression artifacts.

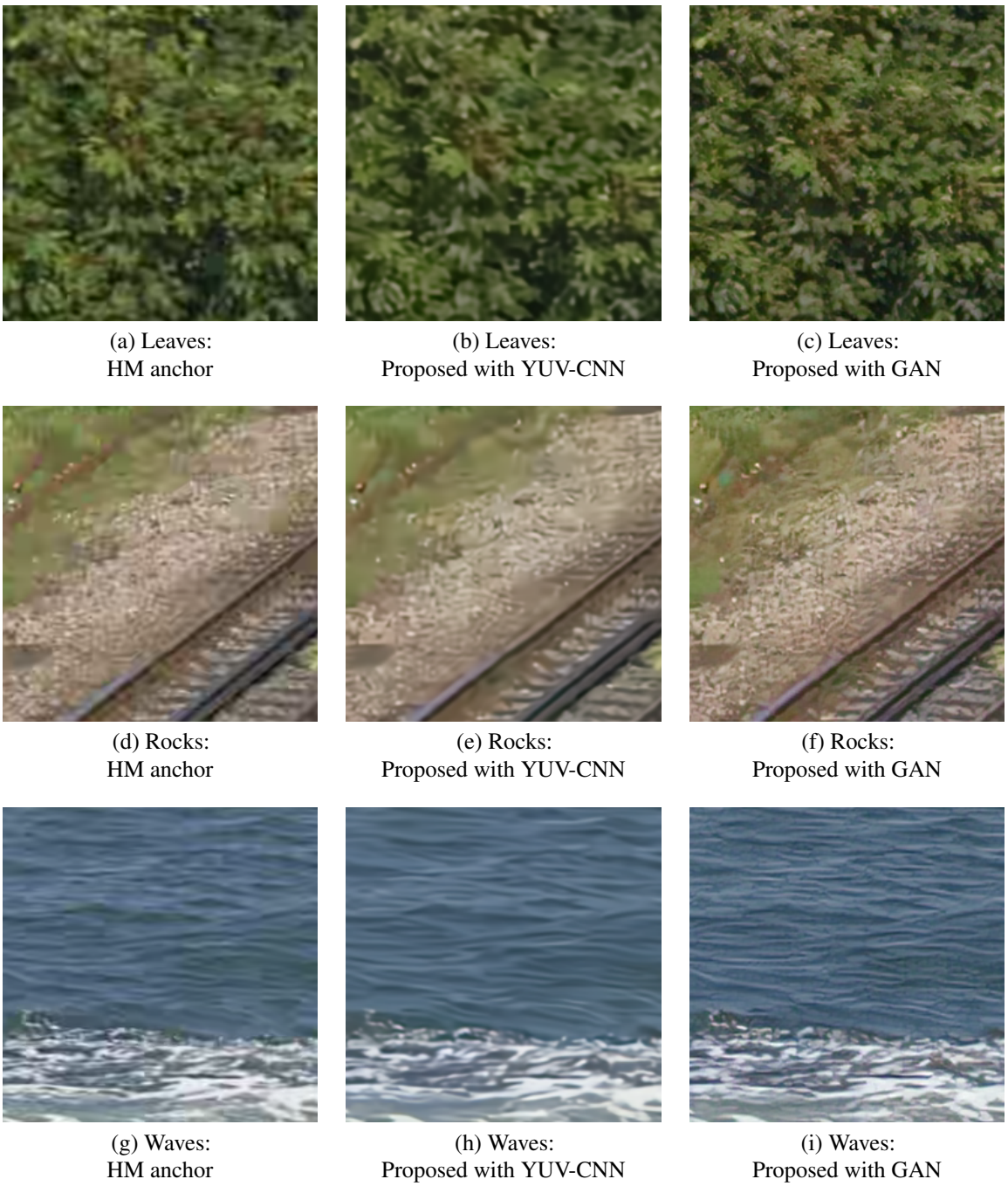
To evaluate the results, both PSNR and VMAF were computed on the reconstructed videos. It is important to note that, as PSNR is based on pixel distortion, is not well suited to evaluate the quality of the video frames resulting from the proposed GAN methodology. In addition to computing the objective quality metrics, visual comparisons between the proposed GAN, the proposed YUV-CNN model and HEVC encoding at a fixed resolution are also presented.

Table 6.3 reports the absolute PSNR and VMAF differences between the proposed spatial resolution adaptation methods using the YUV-CNN and the GAN models compared to the original sequences. These results show that, as expected, the GAN produces lower PSNR values compared to the YUV model for all sequences, with an average PSNR-Y difference of 0.65 dB. However, when it comes to VMAF, for most sequences, the GAN achieves slightly higher VMAF values, with an average of 0.35 VMAF points higher than the YUV-CNN model. Nevertheless, this is not the case for all sequences. For instance, for sequence S12, *Manege*, there is a reduction in the VMAF score of 1.75.

6.2.5.1 Visual quality

As mentioned before, the networks trained to optimize distortion metrics, such as the MSE, produce frames that lack texture detail. We have found that the GAN framework is able to partially reconstruct some of the textures. To illustrate this, Figure 6.14 shows several patches from reconstructed test sequence frames, with different texture types: leaves, rocks and waves. The patches on the left were obtained by applying the YUV-CNN model described in 6.2.2 and the right patches were produced by the trained GAN model. It can be seen that, overall, the textures are better represented by the GAN model compared to the YUV-CNN model, especially in the first texture type, leaves, although the reconstructed textures do not look exactly like the original textures. Nonetheless, in fact, when a user watches a real-world video, he or she does not know what the uncompressed video looks like. For this reason, synthesised or reconstructed textures need not to match the originals exactly but should resemble the high level concepts in a perceptual manner.

Figure 6.15 presents visual comparisons between examples patches comparing the anchor, HM 16.18 encoding at a fixed resolution and the proposed spatial resolution adaptation approach with the GAN training at similar bitrates. From these results, we can see that the anchor (the middle patches)



(a) Leaves: HM anchor
(b) Leaves: Proposed with YUV-CNN
(c) Leaves: Proposed with GAN
(d) Rocks: HM anchor
(e) Rocks: Proposed with YUV-CNN
(f) Rocks: Proposed with GAN
(g) Waves: HM anchor
(h) Waves: Proposed with YUV-CNN
(i) Waves: Proposed with GAN

Figure. 6.14 Visual comparison between the HM anchor and the proposed approach using the YUV-CNN model and the GAN for three different texture types, top: leaves, middle: rocks, and bottom: waves. Patches were extracted from sequences *CentralLineCrossing* and *Petibato* and correspond to the second frame of the lowest bitrate rate-point. Better viewed in the digital version.

Table 6.3 Quality difference, measured by the per-channel PSNR and VMAF values, between the reconstructed videos achieved by employing the proposed GAN and the proposed YUV-CNN model. Positive values mean that the GAN achieves higher quality). These were computed on the lowest bitrate rate-point of the test methodology.

Sequence	**PSNR-Y [dB]**	**PSNR-U [dB]**	**PSNR-V [dB]**	**VMAF**
S01: CalmingWater	-0.25	-0.59	-0.93	0.87
S02: CatRobot	-0.55	-0.98	-1.52	-0.32
S03: CentralLineCrossing	-1.09	-1.32	-1.91	0.64
S04: Chimera-ep08	-0.52	-1.24	-0.81	1.02
S05: Chimera-ep12	-0.46	-0.78	-0.68	-0.58
S06: Chimera-ep17	-0.44	-1.21	-0.79	0.30
S07: Chimera-ep20	-1.96	-0.67	-1.00	1.44
S08: DaylightRoad	-0.41	-1.13	-0.85	0.42
S09: ElFuente-ep33	-0.83	-0.80	-1.26	-0.35
S10: FoodMarket	-0.63	-1.11	-1.49	1.23
S11: LampLeaves	-0.49	-1.09	-1.02	0.88
S12: Manege	-0.62	-1.11	-1.36	-1.75
S13: NingyoPompons	-0.45	-0.75	-1.10	0.03
S14: Petibato	-0.39	-0.40	-0.49	1.09
Average	**-0.65**	**-0.94**	**-1.09**	**0.35**

contain higher visible artifacts compared to the spatially adapted result with the GAN (the right patches). In general, as illustrated by these visual comparisons, encoding using a fixed resolution and an increased quantization level (QP step) tends to introduce blocking and ringing artifacts. In contrast, by dynamically resampling before encoding, we are able to employ a lower quantization level and minimize the scaling artifacts using the proposed deep neural networks, which leads to overall visual quality improvement. Compared to the conventional CNN training, the GAN framework helps the network learn to generate additional details and generate more natural-looking video frames. Further visual comparisons can be found in the Appendix E. Furthermore, demo videos comparing the anchor and our adaptive resolution approach using both the YUV-CNN and the GAN are available online [1].

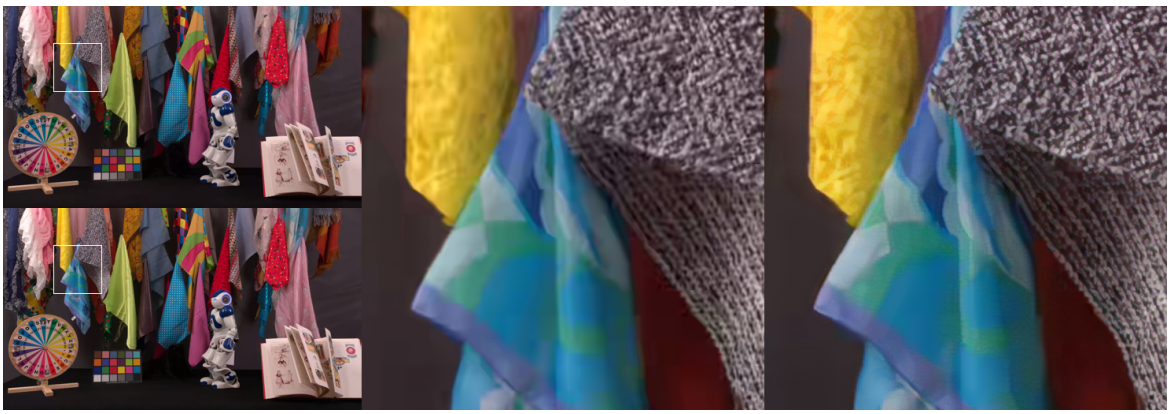
6.2.5.2 Limitations

Although the proposed network produces promising results in terms of perceptual video quality, we have identified a few limitations. The first is the instability during the training process, which is a common problem of GAN frameworks and is caused by the training of competing networks, the generator and the discriminator. However, this problem can be minimized, in part, by the use of special training techniques and tips [174] and improved GAN frameworks [175, 170], as employed in this work.

[1] https://github.com/marianaAfonso/demosBristolCNNcompression

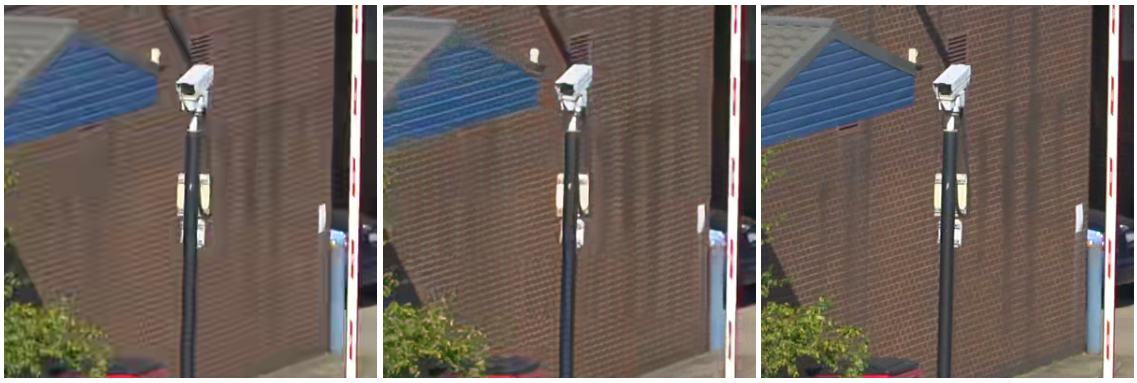
(a) CentralLineCrossing @ 1800 Kbps

(b) DaylightRoad2 @ 1300 Kbps



(c) CatRobot1 @ 1500 Kbps

Figure. 6.15 Visual comparison between the anchor, HM 16.18 at a fixed resolution (left), and our proposed resolution adaptation framework using the GAN model (right), for three test sequences at the lowest bitrate. All patches were extracted from the first B frame.



(a) HM 16.18 (b) Proposed with GAN (d) Original

Figure. 6.16 Comparison between HM 16.18 encoding at a fixed resolution, the proposed approach spatial resolution adaptation with the GAN and the original for a patch of the *CentralLineCrossing* sequence.

Another limitation is the capacity of the network to generalize to different content types. For instance, consider the images in Figure 6.16, which shows a patch encoded by HM 16.18 at a fixed resolution, the proposed approach with the GAN and the original patch at the lowest bitrate rate-point of the test sequence *CentralLineCrossing*. It can be seen that the GAN attempts to add textured detail to the brick wall, which does not correspond to the original structured pattern. We believe that this behaviour is associated with a possible imbalance between regular textures and stochastic textures found in the training dataset. For this reason, without additional prior knowledge about the semantic content of the scene to be reconstructed, it is difficult for the network to synthesize all texture types accurately.

6.2.5.3 Complexity analysis

As the improved CNN reconstruction models are only applied at the decoder, the encoding remains unchanged. Therefore, as reported in Section 5.5.4, the encoding time is reduced by approximately 51% compared to encoding at a fixed resolution using HM 16.18 on the test sequences. On the other hand, complexity is significantly higher at the decoder, with the proposed CNN implementation taking an average of 1.5 seconds per frame for the upsampling of an HD frame to UHD resolution using the proposed YUV-CNN or GAN models, which share the same architecture. Therefore, the overall decoder complexity is increased by approximately 5.7 times, on average, which is slightly higher than with the previous CNN model introduced in Chapter 5. These values were calculated a PC running with an Intel Core i7 4770K CPU @ 3.5 GHz and a NVIDIA 1080ti GPU.

6.3 Discussion

In this chapter, two types of improvements to the compressed super-resolution technique proposed in Chapter 5, as part of the spatial resolution adaptation framework for video compression, were presented. First, we have explored new techniques and architecture choices for improved quality measured by PSNR including different number of layers, loss function and batch normalization. Later, another approach, based on Generative Adversarial Networks (GANs), was proposed with the purpose of achieving a reconstructed video with higher perceptual quality, similar to natural videos.

We have proposed improvements to the CNN architecture presented in Chapter 5, including the addition of residual blocks instead of cascaded convolutional layers, the use of sub-pixel convolution to perform the upsampling of the features and parametric ReLUs. Experimental results show that the proposed network, CSRCNN, slightly outperform other architectures for compressed content. In addition, we propose to jointly train the YUV components instead of applying the CNN to the Y channel only and a linear interpolation using Lanczos3 for the chroma. Experimental results show that better gains are achieved for the chroma samples with little effect on the luma channel quality by employing this methodology.

Visual comparisons between the proposed GAN training methodology and the conventional CNN show that the GAN is able to achieve very promising perceptual quality. In general, it results is a more natural-looking video, with a better reconstruction of texture content. Objective metrics computed on the results of the GAN training show that, as expected, PSNR values are decreased, but the perceptual quality metric, VMAF, is slightly improved on most sequences tested.

Chapter 7

Conclusion and future work

Video compression is a crucial topic of research that has a high impact on modern society. Decades of developments in video codecs have resulted in modern formats that achieve very impressive performance. Nevertheless, pressure for greater visual experiences from consumers drives the demand for even better compression efficiencies. For this reason, more unconventional techniques might be required for future coding standards.

This thesis has explored approaches for improving the compression efficiency of modern video codecs by employing intelligent resampling methods that make decisions whether to encode less information from the input video based on its content, and then synthesize the missing information after decoding using a reconstruction model.

The main contributions of this work, organized by chapters, are summarized below:

- In Chapter 3, a dataset of homogeneous static and dynamic video textures, HomTex, is proposed. This dataset has proven to be useful for research in the field of texture analysis and synthesis. In addition, an extensive study of the encoding statistics and performance of HEVC for different texture types has been conducted. This work provides a better understanding of the limitations of current codecs and defines a better definition of dynamic textures by differentiating between continuous and discrete textures. Finally, we explore an alternative approach to dynamic texture synthesis, using optical flow, which is able to provide promising results for discrete dynamic textures.

- Chapter 4 proposes another form of analysis-synthesis technique based on an adaptive spatial resolution resampling methodology, where instead of encoding at the full resolution, a low resolution version is encoded and later upsampled at the decoder. A new low complexity framework for intra coding that adaptively decides which frames to resample based on a single feature, is proposed. In addition to reducing complexity, this method achieves improved rate-distortion, especially for content with lower spatial detail, with average savings of 4% using PSNR BD-rate for sequences tested.

- Limitations and improvements to the work presented in Chapter 4 are addressed in Chapter 4. The proposed resolution decision process is expanded to include both intra and inter frames by applying a machine learning-based model using low level visual features extracted from the uncompressed video frames. In addition, to achieve improved reconstruction quality of the compressed videos, a deep Convolutional Neural Network (CNN) is applied. Experimental results show that the proposed spatial resolution adaptation framework with the trained CNN provides improved compression efficiency compared to the HEVC reference encoder. On the 14 UHD sequences tested, BD-rate savings of approximately 12% and 16% as measured by PSNR-Y and VMAF, respectively, are achieved by employing proposed method.

- Lastly, inspired by recent advances in the field of super-resolution, further improvements to the reconstruction quality of the deep learning technique is presented in Chapter 6. Two different types of models are explored by focussing on the reconstruction quality based on (i) pixel-wise metrics like PSNR and; (ii) by using a perceptually inspired technique using Generative Adversarial Networks (GANs). Based on a comprehensive study of different techniques, an improved network architecture for compressed super-resolution, CSRCNN, is proposed, which achieves slightly higher performance compared to previous networks. In addition, instead of only applying the CNN to the luminance channel, a new model jointly exploits the luminance and chroma channels. Experimental results show that the modifications result in improved compression efficiency, as measured by PSNR and VMAF. On the hand, by employing the proposed GAN training strategy, the perceptual quality of the reconstructed video is enhanced, especially for textured content. This is reflected by a small increase in the VMAF score but is also demonstrated by visual comparisons.

Our work is one of the first contributions to leverage the potential of deep neural networks applied to video compression, which we believe is a technique that can provide further improvements in terms of compression efficiency for future video codecs. In addition, in our opinion, most video compression systems do not focus enough on perceptual quality over pixel-wise metrics. We have shown that our proposed methods can provide both improvements in terms of objective metrics, but also perceptual quality.

One downside of the proposed spatial resolution approach is that, although reducing complexity at the encoder side due to the possibility of encoding at lower spatial resolutions, the use of a deep convolutional neural network significantly increases the complexity at the decoder. However, we believe that future work on complexity optimizations as well as developments in deep-learning-based parallel processing hardware has the potential to enable deep learning-based approaches to be feasible for future systems.

Although promising, we believe this work can be further improved and built upon. The following is a non-exhaustive list of research directions that we consider to be worthwhile or likely to yield positive returns:

- The application of the resolution adaptation decision at a lower level, potentially inside the encoding loop. This would allow for a more flexible resampling scheme where frames within the same intra period could be encoded using different spatial resolutions. More specifically, it would be possible to encode only the I and P pictures using a lower resolution. It also means that the reconstructed frames could be used as a reference to predict other frames. This modification would also require adjusting other types encoded information on the low resolution encoded frames to be used as references, such as motion vectors and partitioning structure.

- The exploration of a joint spatial, temporal and bit depth encoding model which dynamically decides the best parameters for a given frame/group of frames, including the study of the interactions between these parameters. For instance, for a given intra period, the optimal decision might be to reduce the frame rate from 60 fps to 30 fps and spatially downsample using a ratio of 2 before encoding. At the decoder, the reconstructed video would be obtained by applying the reverse process.

- The current deep learning based super-resolution model is trained and applied to all frames independent of type (I, B and P). Since these pictures generally contain different types of artifacts, it would be interesting to investigate the possibility of training multiple models for each frame type.

- Conduction of subjective tests to verify what types of super-resolutions artifacts are more pleasing for the viewers. In our experiments, we only calculate VMAF and consider the subjective quality of individual frames. However, a more extensive subjective experiment could be conducted to understand which CNN model produces the best subjective quality for each type of content, especially textured vs non-textured. We hypothesize that, given the visual results shown, subjects would prefer the CNN trained using the GAN framework over the distortion-based CNN for more textured content, whereas the opposite would be the case for structured content.

- In this work, we trained single-image super-resolution methods applied on compressed video frames. However, it is known that video super-resolution tends to provide higher reconstruction performance compared to upsampling each image independently, due to the availability of more information on a given scene. Therefore, we believe this would be something worth exploring when it comes to improving the reconstruction quality of the model.

- In the last chapter, the GAN super-resolution model was trained for all types of content, including textured and non-textured regions. A potential improvement could be to consider texture and structure separately or define more classes that could be used to support the learning process.

- We postulate that the training dataset plays a key role in the performance of the deep learning model. Although care was exercised when selecting a large of 100 diverse video sequences

that cover adequately the SI/TI space, a more in-depth study of the influence of other factors could employed when working on future developments. These factors could include, but are not limited to, evaluating the performance for different number of training sequences, content types or the effect of further data augmentation such as reflection, warping or colour variations.

Publications

1. **M. Afonso**, A. Katsenou, F. Zhang, D. Agrafiotis and D. Bull, “Video texture analysis based on HEVC encoding statistics", in Picture Coding Symposium (PCS), 2016.

2. A. Katsenou, **M. Afonso**, D. Agrafiotis and D. Bull, “Predicting video rate-distortion curves using textural features", in Picture Coding Symposium (PCS), 2016.

3. **M. Afonso**, F. Zhang, A. Katsenou, D. Agrafiotis and D. Bull, “Low complexity video coding based on spatial resolution adaptation", in IEEE International Conference on Image Processing, 2017.

4. A. Katsenou, T. Ntasios, **M. Afonso**, D. Agrafiotis and D. Bull, “Understanding video texture - A basis for video compression", in IEEE International Workshop on Multimedia Signal Processing, 2017.

5. A. Mackin, **M. Afonso**, F. Zhang and D. Bull, “SRQM: A Video Quality Metric for Spatial Resolution Adaptation", in Picture Coding Symposium (PCS), 2018.

6. A. Mackin, **M. Afonso**, F. Zhang and D. Bull, “A study of subjective video quality at various spatial resolutions", in IEEE International Conference on Image Processing, 2018.

7. **M. Afonso**, F. Zhang and D. Bull, “Video Compression based on Spatio-Temporal Resolution Adaptation", Transactions letter, IEEE Transactions on Circuits and Systems for Video Technology (CSVT), 2018.

8. **M. Afonso**, F. Zhang and D. Bull, “Spatial Resolution Adaptation Framework for Video Compression", in Applications of Digital Image Processing XV, International Society for Optics and Photonics, 2018.

9. F. Zhang, **M. Afonso** and D. Bull, “Enhanced Video Compression based on Effective Bit depth Adaptation", in IEEE International Conference on Image Processing, 2019 (submitted).

10. A. Katsenou, F. Zhang, **M. Afonso** and D. Bull, “A Subjective Comparison of AV1 and HEVC for Adaptive Video Streaming", in IEEE International Conference on Image Processing, 2019 (submitted).

Patents

D. Bull, **M. Afonso** and F. Zhang, "A video processing method", UK patent no 1714791.9 (filed).

Contributions to standardization committees

M. Afonso, F. Zhang and D. Bull, “Description of SDR video coding technology proposal by University of Bristol", San Diego, JVET-J0031, Joint Video Experts Team (JVET), April 2018.

Awards

M. Afonso, F. Zhang and D. Bull, “Spatio-temporal adaptation framework for video compression", Joint winner of the Video Compression Technology Grand Challenge, IEEE International Conference on Image Processing (ICIP), September 2017.

References

[1] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural distortion measurement," *Signal Processing: Image Communication*, vol. 19, no. 2, pp. 121–132, 2004.

[2] J. Byrne, S. Ierodiaconou, D. Bull, D. Redmill, and P. Hill, "Unsupervised image compression-by-synthesis within a JPEG framework," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2008, pp. 2892–2895.

[3] F. Zhang and D. R. Bull, "A parametric framework for video compression using region-based texture models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1378–1392, 2011.

[4] M. A. Papadopoulos, F. Zhang, D. Agrafiotis, and D. Bull, "A video texture database for perceptual compression and quality assessment," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2015, pp. 2781–2785.

[5] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[6] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1646–1654.

[7] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network." in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, no. 3, 2017, p. 4.

[8] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, C. C. Loy, Y. Qiao, and X. Tang, "ESRGAN: Enhanced super-resolution generative adversarial networks," *arXiv preprint arXiv:1809.00219*, 2018.

[9] A. Mackin, F. Zhang, and D. R. Bull, "A study of subjective video quality at various frame rates," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2015, pp. 3407–3411.

[10] F. Bossen, "Common test conditions and software reference configurations," *Joint Collaborative Team on Video Coding (JCT-VC), JCTVC-L1100*, 2013.

[11] T. U. Ultra Video Group, http://ultravideo.cs.tut.fi/, [Online; accessed 2017-01-23].

[12] D. T. M. Collection, http://media.xiph.org/video/derf/, [Online; accessed 2017-01-23].

[13] M. Papadopoulos, F. Zhang, D. Agrafiotis, and D. Bull, "A video texture database for perceptual compression and quality assessment," in *IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 2781–2785.

[14] I. Cisco, "Cisco visual networking index: Forecast and methodology, 2011–2016," *CISCO White paper*, vol. 518, 2012.

[15] L. M. Wilcox, R. S. Allison, J. Helliker, B. Dunk, and R. C. Anthony, "Evidence that viewers prefer higher frame-rate film," *ACM Transactions on Applied Perception (TAP)*, vol. 12, no. 4, p. 15, 2015.

[16] D. Bull, *Communicating pictures: A course in Image and Video Coding*. Academic Press, 2014.

[17] "Video Codec for Audiovisual Services at p × 64 kbit/s," ITU-T, Geneva, Switzerland, Standard, 1990.

[18] "ISO/IEC 11172, Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1:5 Mbit/s," MPEG, Standard, 1993.

[19] "ISO/IEC 13818, Generic Coding of Moving Pictures and Associated Audio Information," MPEG, Standard, 1995.

[20] "Advanced Video Coding for generic audiovisual services," ITU-T, Standard, 2003.

[21] G. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[22] "JVET-H1002, Joint Call for Proposals on Video Compression with Capability beyond HEVC," JVET, Standard, 2017.

[23] "VP8 Data Format and Decoding Guide," Google, Tech report, 2011.

[24] "VP9 Video Codec," Google, Tech report, 2012.

[25] "Alliance for open media," https://aomedia.org/, [Online; accessed 2018-11-20].

[26] "AV1 Bitstream and Decoding Process Specification," AOM, Tech report, 2018.

[27] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," *VCEG-M33 ITU-T Q6/16, Austin, TX, USA*, 2001.

[28] P. Hanhart and T. Ebrahimi, "Calculation of average coding efficiency based on subjective quality scores," *Journal of Visual communication and image representation*, vol. 25, no. 3, pp. 555–564, 2014.

[29] B. Girod, "What's wrong with mean-squared error?" *Digital images and human vision*, pp. 207–220, 1993.

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[31] M. H. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *IEEE Transactions on Broadcasting*, vol. 50, no. 3, pp. 312–322, 2004.

[32] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, "Toward a practical perceptual video quality metric," 2016.

[33] V. Q. E. G. (VQEG), "Report on the validation of video quality models for high definition video content," 2010.

[34] K. Seshadrinathan, R. Soundararajan, A. C. Bovik, and L. K. Cormack, "Study of subjective and objective quality assessment of video," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1427–1441, 2010.

[35] F. Zhang and D. R. Bull, "A perception-based hybrid model for video quality assessment," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 6, pp. 1017–1028, 2016.

[36] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3. IEEE, 2004, pp. iii–709.

[37] S. Li, F. Zhang, L. Ma, and K. N. Ngan, "Image quality assessment by separately evaluating detail losses and additive impairments," *IEEE Transactions on Multimedia*, vol. 13, no. 5, pp. 935–949, 2011.

[38] C. G. Bampis, A. C. Bovik, and Z. Li, "A simple prediction fusion improves data-driven full-reference video quality assessment models," in *2018 Picture Coding Symposium (PCS)*. IEEE, 2018, pp. 298–302.

[39] H. Schwarz, D. Marpe, and T. Wiegand, "Analysis of hierarchical b pictures and mctf," in *IEEE International Conference on Multimedia and Expo*, 2006, pp. 1929–1932.

[40] M. J. Nadenau, S. Winkler, D. Alleysson, and M. Kunt, "Human vision models for perceptually optimized image processing– a review," *Proceedings of the IEEE*, vol. 32, 2000.

[41] "WebM project," https://www.webmproject.org/, accessed: 2018-10-02.

[42] T. Daede, N. Egge, J. Valin, G. Martres, and T. Terriberry, "Daala: A perceptually-driven next generation video codec," *CoRR*, vol. abs/1603.03129, 2016. [Online]. Available: http://arxiv.org/abs/1603.03129

[43] G. Bjontegaard, T. Davies, A. Fuldseth, and S. Midtskogen, "The thor video codec," in *IEEE Data Compression Conference (DCC)*. IEEE, 2016, pp. 476–485.

[44] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi *et al.*, "An overview of core coding tools in the AV1 video codec," in *Picture Coding Symposium (PCS)*, 2018, pp. 24–27.

[45] "JVET-G1001, Algorithm Description of Joint Exploration Test Model 7 (JEM 7)," JVET, Standard, July 2017.

[46] "Algorithm description for Versatile Video Coding and Test Model 3 (VTM 3)," JVET, Output document, October 2018.

[47] M. Afonso, F. Zhang, and D. R. Bull, "Video compression based on spatio-temporal resolution adaptation," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2018.

[48] Harmonic Inc 4K demo footage, https://www.harmonicinc.com/4k-demo-footage-download/, [Online; accessed 2017-05-01].

[49] I. Katsavounidis, "Chimera video sequence details and scenes," Netflix, Tech. Rep., November 2015. [Online]. Available: https://www.cdvl.org/documents/NETFLIX_Chimera_4096x2160_Download_Instructions.pdf

[50] "JVET common test conditions and software reference configurations," JVET, San Diego, USA, Standard, Feb. 2016.

[51] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[52] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[53] S. Dodge and L. Karam, "A study and comparison of human and deep learning recognition performance under visual distortions," in *IEEE International Conference on Computer Communication and Networks (ICCCN)*, 2017, pp. 1–7.

[54] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[55] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Ieee, 2009, pp. 248–255.

[56] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[57] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[58] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1, 2013, p. 3.

[59] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

[60] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[61] M. A. Nielsen, *Neural networks and deep learning*. Determination press USA, 2015, vol. 25.

[62] A. Karpathy, J. Johnson, and L. Fei-Fei, "Course notes in cs231n convolutional neural networks for visual recognition," http://cs231n.github.io/, 2018.

[63] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," *arXiv preprint arXiv:1804.07612*, 2018.

[64] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[65] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[66] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[67] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[68] M. Mirmehdi, *Handbook of texture analysis*. Imperial College Press, 2008.

[69] M. Afonso, A. Katsenou, F. Zhang, D. Agrafiotis, and D. Bull, "Video texture analysis based on hevc encoding statistics," in *Picture Coding Symposium*, 2016.

[70] V. Kwatra, "Part iii: Dynamic texture synthesis," in *ACM SIGGRAPH 2007 Courses*, ser. SIGGRAPH '07. New York, NY, USA: ACM, 2007. [Online]. Available: http://doi.acm.org/10.1145/1281500.1281614

[71] M. H. Bharati, J. Liu, and J. F. MacGregor, "Image texture analysis: methods and comparisons," *Chemometrics and Intelligent Laboratory Systems*, vol. 72, no. 1, pp. 57 – 71, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169743904000528

[72] R. M. Haralick, K. Shanmugam *et al.*, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

[73] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *ACM Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 479–488.

[74] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *ACM Annual Conference on Computer Graphics and Interactive Techniques*. ACM, 2001, pp. 341–346.

[75] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: image and video synthesis using graph cuts," in *ACM Transactions on Graphics (ToG)*, vol. 22, no. 3. ACM, 2003, pp. 277–286.

[76] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *ACM Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424.

[77] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[78] S. Ierodiaconou, J. Byrne, D. R. Bull, D. Redmill, and P. Hill, "Unsupervised image compression using graphcut texture synthesis," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2009, pp. 2265–2268.

[79] J. Portilla and E. P. Simoncelli, "A parametric texture model based on joint statistics of complex wavelet coefficients," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 49–70, 2000.

[80] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *IEEE International Conference on Computer Vision (ICCV)*, vol. 2. IEEE, 1999, pp. 1033–1038.

[81] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa, "Video textures," in *ACM Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 489–498.

[82] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *International Journal of Computer Vision*, vol. 51, no. 2, pp. 91–109, 2003.

[83] P. Ndjiki-Nya, T. Hinz, and T. Wiegand, "Generic and robust video coding with texture analysis and synthesis," in *IEEE International Conference on Multimedia and Expo*, 2007, pp. 1447–1450.

[84] M. Bosch, F. Zhu, and E. J. Delp, "Segmentation-based video compression using texture and motion models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1366–1377, 2011.

[85] J. Ballé, A. Stojanovic, and J.-R. Ohm, "Models for static and dynamic texture synthesis in image and video compression," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1353–1365, 2011.

[86] P. Ndjiki-Nya, D. Doshkov, H. Kaprykowsky, F. Zhang, D. Bull, and T. Wiegand, "Perception-oriented video coding based on image analysis and completion: A review," *Signal Processing: Image Communication*, vol. 27, no. 6, pp. 579–594, 2012.

[87] N. Kingsbury, "Complex wavelets for shift invariant analysis and filtering of signals," *Applied and Computational Harmonic Analysis*, vol. 10, no. 3, pp. 234–253, 2001.

[88] R. Péteri, S. Fazekas, and M. J. Huiskes, "Dyntex: A comprehensive database of dynamic textures," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1627–1632, 2010.

[89] M. M. Subedar and L. J. Karam, "A no reference texture granularity index and application to visual media compression," in *IEEE Inter. Conf. on Image Processing*, 2015, pp. 760–764.

[90] F. M. Moss, F. Zhang, R. Baddeley, and D. Bull, "What's on TV: A large scale quantitative characterisation of modern broadcast video content," in *IEEE International Conference on Image Processing (ICIP)*, 2016.

[91] B. Butterworth, "History of the BBC redux project," *BBC Internet Blog*, 2008.

[92] F. J. Massey Jr, "The Kolmogorov-Smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[93] U. S. Thakur, M. Bhat, M. Bläser, M. Wien, D. Bull, and J.-R. Ohm, "Synthesis of fine details in B picture for dynamic textures," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 2751–2755.

[94] O. Chubach, P. Garus, M. Wien, and J.-R. Ohm, "Analysis/synthesis coding of dynamic textures based on motion distribution statistics," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 3016–3020.

[95] A. V. Katsenou, T. Ntasios, M. Afonso, D. Agrafiotis, and D. R. Bull, "Understanding video texture — a basis for video compression," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, Oct 2017, pp. 1–6.

[96] "HEVC reference software," https://hevc.hhi.fraunhofer.de/, accessed: 2018-03-13.

[97] F. Yang, G.-S. Xia, G. Liu, L. Zhang, and X. Huang, "Dynamic texture recognition by aggregating spatial and temporal features via ensemble svms," *Neurocomputing*, vol. 173, pp. 1310–1321, 2016.

[98] Y. Sun, Y. Xu, and Y. Quan, "Characterizing dynamic textures with space-time lacunarity analysis," in *IEEE International Conference on Multimedia and Expo*, 2015, pp. 1–6.

[99] M. A. Papadopoulos, F. Zhang, D. Agrafiotis, and D. Bull, "An adaptive QP offset determination method for HEVC," in *IEEE International Conference on Image Processing (ICIP)*, 2016.

[100] F. Zhang and D. R. Bull, "An adaptive lagrange multiplier determination method for rate-distortion optimisation in hybrid video codecs," in *2015 IEEE International Conference on Image Processing*, 2015, pp. 671–675.

[101] J. Lei, S. Li, C. Zhu, M.-T. Sun, and C. Hou, "Depth coding based on depth-texture motion and structure similarities," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 2, pp. 275–286, 2015.

[102] J. Stankowski, T. Grajek, D. Karwowski, K. Klimaszewski, O. Stankiewicz, K. Wegner, and M. Domański, "Analysis of frame partitioning in HEVC," in *Computer Vision and Graphics*. Springer, 2014, pp. 602–609.

[103] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, Mar 2011. [Online]. Available: https://doi.org/10.1007/s11263-010-0390-2

[104] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Third Edition*. Orlando, FL, USA: Academic Press, Inc., 2006.

[105] B. K. Horn and B. G. Schunck, "Determining optical flow," in *1981 Technical symposium east*. International Society for Optics and Photonics, 1981, pp. 319–331.

[106] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM computing surveys (CSUR)*, vol. 27, no. 3, pp. 433–466, 1995.

[107] D. Fleet and Y. Weiss, "Optical flow estimation," in *Handbook of mathematical models in computer vision*. Springer, 2006, pp. 237–257.

[108] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010, pp. 2432–2439.

[109] H. Chen and N. Kingsbury, "Efficient registration of nonrigid 3-D bodies," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 262–272, 2012.

[110] V. Nguyen, Y. Tan, and W. Lin, "Adaptive downsampling/upsampling for better video compression at low bit rate," in *IEEE International Symposium on Circuits and Systems*, 2008, pp. 1624–1627.

[111] S. Uslubas, E. Maani, and A. Katsaggelos, "A resolution adaptive video compression system," in *Intelligent Multimedia Communication: Techniques and Applications*. Springer, 2010, pp. 167–194.

[112] M. Shen, P. Xue, and C. Wang, "Down-sampling based video coding using super-resolution technique," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 6, pp. 755–765, 2011.

[113] J. Dong and Y. Ye, "Adaptive downsampling for high-definition video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 3, pp. 480–488, 2014.

[114] G. Georgis, G. Lentaris, and D. Reisis, "Reduced complexity superresolution for low-bitrate video compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 2, pp. 332–345, 2016.

[115] J. Lin, D. Liu, H. Yang, H. Li, and F. Wu, "Convolutional neural network-based block up-sampling for HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2018.

[116] M. Afonso, F. Zhang, A. Katsenou, D. Agrafiotis, and D. Bull, "Low complexity video coding based on spatial resolution adaptation," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 3011–3015.

[117] R. Wang, C. Huang, and P. Chang, "Adaptive downsampling video coding with spatially scalable rate-distortion modeling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 11, pp. 1957–1968, 2014.

[118] B. Hosking, D. Agrafiotis, D. Bull, and N. Easton, "Enhancement of intra-coded pictures for greater coding efficiency," in *Picture Coding Symposium*, 2016.

[119] B. Hosking, D. Agrafiotis, D. Bull, and N. Eastern, "An adaptive resolution rate control method for intra coding in hevc," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 1486–1490.

[120] R. Wang, M. Chien, and P. Chang, "Adaptive down-sampling video coding," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2010, pp. 75 420P–75 420P.

[121] R. C. Gonzalez, R. E. Woods *et al.*, "Digital image processing," 2002.

[122] W. Burger, M. J. Burge, M. J. Burge, and M. J. Burge, *Principles of digital image processing*. Springer, 2009.

[123] Y. Li, D. Liu, H. Li, L. Li, and F. Wu, "Convolutional neural network-based block up-sampling for intra frame coding," *IEEE Circuits and Systems for Video Technology*, 2018.

[124] G. Bjontegaard, "Calcuation of average psnr differences between rd-curves," *Doc. VCEG-M33 ITU-T Q6/16, Austin, TX, USA*, 2001.

[125] M. Afonso, F. Zhang, and D. Bull, "Spatial resolution adaptation framework for video compression," in *Applications of Digital Image Processing XL*. International Society for Optics and Photonics, 2018.

[126] K. Nasrollahi and T. B. Moeslund, "Super-resolution: a comprehensive survey," *Machine Vision and Applications*, vol. 25, no. 6, pp. 1423–1468, 2014.

[127] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[128] R. Timofte, V. De Smet, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 1920–1927.

[129] J. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5197–5206.

[130] R. Timofte, V. De Smet, and L. Van Gool, "A+ adjusted anchored neighborhood regression for fast super-resolution," in *Asian Conference on Computer Vision*, 2014, pp. 111–126.

[131] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos, "Super-resolution of compressed videos using convolutional neural networks," in *IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 1150–1154.

[132] I. . S. Group, "Description of sdr video coding technology proposal by university of science and technology of china, peking university, harbin institute of technology, and wuhan university," 2019.

[133] C. Dong, Y. Deng, C. Change Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 576–584.

[134] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[135] C.-W. Hsu, C.-Y. Chen, T.-D. Chuang, H. Huang, S.-T. Hsiang, C.-C. Chen, M.-S. Chiang, C.-Y. Lai, C.-M. Tsai, Y.-C. Su, Z.-Y. Lin, H. Yu-Ling, J. Klopp, I.-H. Wang, Y.-W. Huang, and S.-M. Lei, "Description of sdr video coding technology proposal by mediatek," 2019.

[136] J. Pfaff, P. Helle, D. Maniry, S. Kaltenstadler, W. Samek, H. Schwarz, D. Marpe, and T. Wiegand, "Neural network based intra prediction for video coding," in *Applications of Digital Image Processing XL*. International Society for Optics and Photonics, 2018.

[137] L. Song, X. Tang, W. Zhang, X. Yang, and P. Xia, "The sjtu 4k video sequence dataset," in *International Workshop on Quality of Multimedia Experience (QoMEX)*. IEEE, 2013, pp. 34–35.

[138] AWS Elemental, https://www.youtube.com/playlist?list=PLwIpNYl7S0G_C5I76Tf46n6ImKssMn2kT/, [Online; accessed 2018-07-02].

[139] S. Winkler, "Analysis of public image and video databases for quality assessment," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 6, pp. 616–625, 2012.

[140] I.-T. recommendation P.910, "Subjective video quality assessment methods for multimedia applications," 1999.

[141] MathWorks, "Block matching: Computer vision system toolbox," https://uk.mathworks.com/help/vision/ref/blockmatching.html, [Online; accessed 2018-09-20].

[142] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[143] J. P. Lewis, "Fast template matching," in *Vision interface*, vol. 95, no. 120123, 1995, pp. 15–19.

[144] A. Mackin, M. Afonso, F. Zhang, and D. Bull, "SRQM: a video quality metric for spatial resolution adaptation," in *Picture Coding Symposium (PCS)*, 2018.

[145] A. V. Katsenou, M. Afonso, D. Agrafiotis, and D. R. Bull, "Predicting video rate-distortion curves using textural features," in *Picture Coding Symposium (PCS), 2016*. IEEE, 2016, pp. 1–5.

[146] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and

X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[147] "BBC research and development," https://www.bbc.co.uk/rd/projects/video-coding, accessed: 2018-10-21.

[148] J. Le Feuvre, J. Thiesse, M. Parmentier, M. Raulet, and C. Daguet, "Ultra high definition HEVC DASH data set," in *ACM Multimedia Systems Conference*. ACM, 2014, pp. 7–12.

[149] I. Katsavounidis, "El Fuente video sequence details and scenes," Netflix, Tech. Rep., July 2015. [Online]. Available: https://www.cdvl.org/documents/ElFuente_summary.pdf

[150] Deloitte, "Hitting the accelerator: the next generation of machine-learning chips," https://www2.deloitte.com/content/dam/Deloitte/global/Images/infographics/technologymediatelecommunications/gx-deloitte-tmt-2018-nextgen-machine-learning-report.pdf, 2017, [Online; accessed 2018-07-01].

[151] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[152] ITU-R Recommendation BT.500-13, "Methodology for the subjective assessment of the quality of television pictures," 2012.

[153] F. Zhang, F. M. Moss, R. Baddeley, and D. R. Bull, "BVI-HD: A video quality database for HEVC compressed and texture synthesised content," *IEEE Transactions on Multimedia*, 2018.

[154] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.

[155] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, vol. 1, no. 2, 2017, p. 4.

[156] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European Conference on Computer Vision*. Springer, 2016, pp. 391–407.

[157] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[158] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016.

[159] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[160] ——, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[161] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[162] J. Bruna, P. Sprechmann, and Y. LeCun, "Super-resolution with deep convolutional sufficient statistics," *arXiv preprint arXiv:1511.05666*, 2015.

[163] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[164] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 47–57, 2017.

[165] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2. Ieee, 2003, pp. 1398–1402.

[166] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[167] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[168] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[169] M. S. Sajjadi, B. Schölkopf, and M. Hirsch, "Enhancenet: Single image super-resolution through automated texture synthesis," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 4501–4510.

[170] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard GAN," *arXiv preprint arXiv:1807.00734*, 2018.

[171] Y. Blau, R. Mechrez, R. Timofte, T. Michaeli, and L. Zelnik-Manor, "2018 pirm challenge on perceptual image super-resolution," *arXiv preprint arXiv:1809.07517*, 2018.

[172] C. Ma, C.-Y. Yang, X. Yang, and M.-H. Yang, "Learning a no-reference quality metric for single-image super-resolution," *Computer Vision and Image Understanding*, vol. 158, pp. 1–16, 2017.

[173] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a completely blind image quality analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, 2013.

[174] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," *CoRR*, vol. abs/1606.03498, 2016. [Online]. Available: http://arxiv.org/abs/1606.03498

[175] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

Appendix A

HomTex sequences specification

Table A.1 Specification of the 120 HomTex video sequences. All sequences are 256 × 256 pixels. BVI-TEX corresponds to the BVI texture database [13].

Sequence name	Dynamics	Structure	Granularity	Source
CalmSea_scaled	dynamic	continuous	high	DynTex
RiverDrops_scaled	dynamic	continuous	high	DynTex
ShinyWater_scaled	dynamic	continuous	high	DynTex
WavyShinnySea_scaled	dynamic	continuous	high	DynTex
BoilingWater	dynamic	continuous	low	DynTex
BoilingWater_scaled	dynamic	continuous	low	DynTex
CalmingWater-leftBottom	dynamic	continuous	low	BVI-TEX
CalmingWater-middle	dynamic	continuous	low	BVI-TEX
CalmingWater_rightBottom	dynamic	continuous	low	BVI-TEX
Flag	dynamic	continuous	low	DynTex
Flag_scaled	dynamic	continuous	low	DynTex
FlagSlow	dynamic	continuous	low	DynTex
FlagSlow_scaled	dynamic	continuous	low	DynTex
FlowingRiver	dynamic	continuous	low	DynTex
GreenWater	dynamic	continuous	low	DynTex
GreenWater_scaled	dynamic	continuous	low	DynTex
HeavyShower	dynamic	continuous	low	DynTex
RainSplash	dynamic	continuous	low	DynTex
ShinnyBlueWater	dynamic	continuous	low	DynTex
Shower	dynamic	continuous	low	DynTex
SmokeClear_middle	dynamic	continuous	low	BVI-TEX
SmokeClear_side	dynamic	continuous	low	BVI-TEX
VeryHeavyShower	dynamic	continuous	low	DynTex
WaterfallHomo2	dynamic	continuous	low	DynTex
Waves	dynamic	continuous	low	DynTex
WavySea	dynamic	continuous	low	DynTex
Wool	dynamic	continuous	low	DynTex
BallUnderWater	dynamic	continuous	medium	BVI-TEX
BlueReflection_scaled	dynamic	continuous	medium	DynTex

Sequence name	Dynamics	Structure	Granularity	Source
CalmSea	dynamic	continuous	medium	DynTex
FlowingRiver_scaled	dynamic	continuous	medium	DynTex
HeavyShower_scaled	dynamic	continuous	medium	DynTex
PaperStatic-corner	dynamic	continuous	medium	BVI-TEX
PaperStatic-middle	dynamic	continuous	medium	BVI-TEX
RainSplash_scaled	dynamic	continuous	medium	DynTex
ReflectionWater	dynamic	continuous	medium	DynTex
ShinnyBlueWater_scaled	dynamic	continuous	medium	DynTex
Shower_scaled	dynamic	continuous	medium	DynTex
VeryHeavyShower_scaled	dynamic	continuous	medium	DynTex
WaterFall	dynamic	continuous	medium	DynTex
WaterfallHomo	dynamic	continuous	medium	DynTex
Waves_scaled	dynamic	continuous	medium	DynTex
WavySea_scaled	dynamic	continuous	medium	DynTex
WavyShinnySea	dynamic	continuous	medium	DynTex
Wool_scaled	dynamic	continuous	medium	DynTex
Bamboo_scaled	dynamic	discrete	high	DynTex
BlowingLeaves_scaled	dynamic	discrete	high	DynTex
MovingField_scaled	dynamic	discrete	high	DynTex
SlowTree_scaled	dynamic	discrete	high	DynTex
SunnyBushes_scaled	dynamic	discrete	high	DynTex
SunnyBushes2_scaled	dynamic	discrete	high	DynTex
ThinBranches_scaled	dynamic	discrete	high	DynTex
TreeFlowers_scaled	dynamic	discrete	high	DynTex
WindLeaves2_scaled	dynamic	discrete	high	DynTex
BushesFruits	dynamic	discrete	low	DynTex
DarkTree	dynamic	discrete	low	DynTex
DarkTree2	dynamic	discrete	low	DynTex
PinkFlowers	dynamic	discrete	low	DynTex
RiceField	dynamic	discrete	low	DynTex
Stairs	dynamic	discrete	low	DynTex
SunnyBushes	dynamic	discrete	low	DynTex
WindLeaves	dynamic	discrete	low	DynTex

Sequence name	Dynamics	Structure	Granularity	Source
ZoomedLeaves	dynamic	discrete	low	DynTex
Bamboo	dynamic	discrete	medium	DynTex
BlowingLeaves	dynamic	discrete	medium	DynTex
BricksBushesStatic-bushes1	dynamic	discrete	medium	BVI-TEX
BricksBushesStatic-bushes2	dynamic	discrete	medium	BVI-TEX
BushesFruits_scaled	dynamic	discrete	medium	DynTex
DarkTree_scaled	dynamic	discrete	medium	DynTex
DarkTree2_scaled	dynamic	discrete	medium	DynTex
FlickeringGrass	dynamic	discrete	medium	DynTex
GrassField	dynamic	discrete	medium	DynTex
GrassField2	dynamic	discrete	medium	DynTex
HeavySnow_scaled	dynamic	discrete	medium	DynTex
LampLeaves-bushes1	dynamic	discrete	medium	BVI-TEX
LampLeaves-bushes2	dynamic	discrete	medium	BVI-TEX
LampLeaves-bushes3	dynamic	discrete	medium	BVI-TEX
LampLeaves-bushesBackground	dynamic	discrete	medium	BVI-TEX
LeavesRotating_scaled	dynamic	discrete	medium	DynTex
MovingField	dynamic	discrete	medium	DynTex
OrangeSeaLife_scaled	dynamic	discrete	medium	DynTex
PinkFlowers_scaled	dynamic	discrete	medium	DynTex
PlasmaFree	dynamic	discrete	medium	DynTex
RiceField_scaled	dynamic	discrete	medium	DynTex
SeaPlant_scaled	dynamic	discrete	medium	DynTex
SlowTree	dynamic	discrete	medium	DynTex
Stairs_scaled	dynamic	discrete	medium	DynTex
SunnyBushes2	dynamic	discrete	medium	DynTex
ThinBranches	dynamic	discrete	medium	DynTex
TreeFlowers	dynamic	discrete	medium	DynTex
WildLeaves	dynamic	discrete	medium	DynTex
WindLeaves_scaled	dynamic	discrete	medium	DynTex
WindLeaves2	dynamic	discrete	medium	DynTex

Sequence name	Dynamics	Structure	Granularity	Source
WindLeaves2	dynamic	discrete	medium	DynTex
YellowFlowers	dynamic	discrete	medium	DynTex
ZoomedLeaves_scaled	dynamic	discrete	medium	DynTex
CarpetPanAverage	static	continuous	low	BVI-TEX
FastClouds	static	continuous	low	DynTex
FastClouds_scaled	static	continuous	low	DynTex
SlowCloud	static	continuous	low	DynTex
Cloud	static	continuous	medium	DynTex
BlueCarpet_scaled	static	discrete	high	New
MovingPattern_scaled	static	discrete	high	DynTex
BlueCarpet	static	discrete	low	New
BricksTilting-wall1	static	discrete	low	BVI-TEX
BricksTilting-wall2	static	discrete	low	BVI-TEX
Ceiling	static	discrete	low	DynTex
Ceiling2	static	discrete	low	DynTex
PaintingTilting1	static	discrete	low	BVI-TEX
PaintingTilting2	static	discrete	low	BVI-TEX
PinkRoses	static	discrete	low	DynTex
PurpleFlowers	static	discrete	low	DynTex
SquaredFloor	static	discrete	low	DynTex
TreeTrunk	static	discrete	low	DynTex
TreeTrunk2	static	discrete	low	DynTex
Ceiling_scaled	static	discrete	medium	DynTex
Ceiling2_scaled	static	discrete	medium	DynTex
MovingPattern	static	discrete	medium	DynTex
PinkRoses_scaled	static	discrete	medium	DynTex
PurpleFlowers_scaled	static	discrete	medium	DynTex
SquaredFloor_scaled	static	discrete	medium	DynTex

Appendix B

Performance of the optical-flow interpolation algorithm

Table B.1 Performance comparison between the dynamic texture synthesis approach of [3] and the optical flow interpolation proposed for the **discrete dynamic textures** of HomTex dataset, based on the average PSNR of the synthesised frames.

Sequence name	**Avg. PSNR DTS (dB)**	**Avg. PSNR OF (dB)**	**diff. DTS - OF**
Bamboo	24.19	22.85	-1.33
Bamboo_scaled	25.65	24.08	-1.57
BlowingLeaves	15.38	16.59	**1.21**
BlowingLeaves_scaled	17.51	19.22	1.71
BricksBushesStatic60fps_bushes1	29.20	28.17	-1.03
BricksBushesStatic60fps_bushes2	31.48	29.14	-2.34
BushesFruits	17.98	19.16	**1.19**
BushesFruits_scaled	21.24	23.25	**2.01**
DarkTree2	25.47	25.71	**0.24**
DarkTree2_scaled	26.49	26.53	**0.04**
DarkTree	22.22	23.37	**1.14**
DarkTree_scaled	23.18	24.37	**1.19**
FlickeringGrass	20.68	20.08	-0.60
GrassField2	37.58	36.33	-1.26
GrassField	36.59	34.64	-1.95
HeavySnow_scaled	14.70	14.47	-0.23

Sequence name	Avg. PSNR DTS (dB)	Avg. PSNR OF (dB)	diff. DTS - OF
SunnyBushes2	29.74	27.73	-2.00
SunnyBushes2_scaled	33.03	30.72	-2.31
SunnyBushes	20.37	21.71	1.34
SunnyBushes_scaled	22.41	23.06	**0.65**
ThinBranches	17.47	18.15	**0.68**
ThinBranches_scaled	20.74	20.61	-0.13
TreeFlowers	22.90	21.82	-1.07
TreeFlowers_scaled	26.45	25.33	-1.13
WildLeaves	21.40	21.23	-0.17
WindLeaves2	24.95	25.77	**0.82**
WindLeaves2_scaled	25.52	26.81	**1.28**
WindLeaves	25.18	26.71	**1.53**
WindLeaves_scaled	26.62	27.81	**1.19**
YellowFlowers	27.88	25.87	-2.02
ZoomedLeaves	26.99	28.95	**1.96**
ZoomedLeaves_scaled	26.96	28.80	**1.85**
LeavesRotating_scaled	19.26	22.53	**3.27**
MovingField	17.72	17.29	-0.43
MovingField_scaled	19.78	19.94	**0.16**
OrangeSeaLife_scaled	31.31	29.40	-1.91
PinkFlowers	29.96	29.12	-0.84
PinkFlowers_scaled	32.45	31.63	-0.82
PlasmaFree60fps	26.17	25.13	-1.03
RiceField	21.36	20.93	-0.43
RiceField_scaled	21.66	21.24	-0.42
SeaPlant_scaled	31.28	31.05	-0.23
SlowTree	31.47	27.97	-3.49
SlowTree_scaled	34.16	30.09	-4.06
Stairs	20.86	11.49	-9.37
Stairs_scaled	22.66	23.42	**0.76**
LampLeaves60fps_bushes1	25.34	24.65	-0.69
LampLeaves60fps_bushes2	27.33	26.24	-1.10
LampLeaves60fps_bushes3	22.22	22.15	-0.07
LampLeaves60fps_bushesBackground	33.15	31.50	-1.65

Appendix C

Training sequences

C.1 Specifications

Table C.1 Specification of the 100 video sequences used for training the models in Chapters 5 and 6. Legend: NFX - Netflix, HAR - Harmonic Inc, UVG - Ultra Video Group, ELE: Elemental.

Sequence Name	Res.	Frame rate	Bit depth	Chroma	Number frames	Source
Aerial	UHD	60	10	420	64	NFX
Air-Acrobatics-Scene1	UHD	60	10	420	64	HAR
American-Football-Scene2	UHD	60	10	420	64	HAR
American-Football-Scene3	UHD	60	10	420	64	HAR
American-Football-Scene4	UHD	60	10	420	64	HAR
American-Football-Scene5	UHD	60	10	420	64	HAR
American-Football-Scene6	UHD	60	10	420	64	HAR
Animals-Scene1	UHD	60	10	420	64	HAR
Animals-Scene10	UHD	60	10	420	64	HAR
Animals-Scene11	UHD	60	10	420	64	HAR
Animals-Scene3	UHD	60	10	420	64	HAR
Animals-Scene5	UHD	60	10	420	64	HAR
Animals-Scene7	UHD	60	10	420	64	HAR
Animals-Scene9	UHD	60	10	420	64	HAR
Asian-Fusion-Scene3	UHD	60	10	420	64	HAR
Asian-Fusion-Scene5	UHD	60	10	420	64	HAR
Asian-Fusion-Scene7	UHD	60	10	420	64	HAR
Barscene	UHD	60	10	420	64	NFX
Bosphorus	UHD	120	10	420	64	UVG
Boxingpractice	UHD	60	10	420	64	NFX
Bundnightscape	UHD	60	10	420	64	NFX

Sequence Name	Res.	Frame rate	Bit depth	Chroma	Number frames	Source
Campfireparty	UHD	60	10	420	64	SJTU
Coastguard	UHD	60	10	420	64	ELE
Constructionfield	UHD	60	10	420	64	SJTU
Costa-Rica-Scene1	UHD	60	10	420	64	HAR
Costa-Rica-Scene2	UHD	60	10	420	64	HAR
Costa-Rica-Scene3	UHD	60	10	420	64	HAR
Costa-Rica-Scene4	UHD	60	10	420	64	HAR
Crosswalk	UHD	60	10	420	64	NFX
Dinnerscene-Scene1	UHD	60	10	420	64	NFX
Dinnerscene-Scene2	UHD	60	10	420	64	NFX
Drivingpov	UHD	60	10	420	64	NFX
Fjords-Scene1	UHD	60	10	420	64	HAR
Fjords-Scene2	UHD	60	10	420	64	HAR
Fjords-Scene3	UHD	60	10	420	64	HAR
Foodmarket-Scene1	UHD	60	10	420	64	NFX
Fountains	UHD	60	10	420	64	SJTU
Honeybee	UHD	120	10	420	64	UVG
Hong-Kong-Scene1	UHD	60	10	420	64	HAR
Hong-Kong-Scene2	UHD	60	10	420	64	HAR
Hong-Kong-Scene3	UHD	60	10	420	64	HAR
Hong-Kong-Scene4	UHD	60	10	420	64	HAR
India-Buildings-Scene2	UHD	60	10	420	64	HAR
India-Buildings-Scene3	UHD	60	10	420	64	HAR
India-Buildings-Scene4	UHD	60	10	420	64	HAR
Jockey	UHD	120	10	420	64	UVG
Library	UHD	60	10	420	64	SJTU
Marathon	UHD	60	10	420	64	SJTU
Mobile	UHD	60	10	420	64	ELE
Myanmar-Scene3	UHD	60	10	420	64	HAR
Myanmar-Scene4	UHD	60	10	420	64	HAR
Myanmar-Scene5	UHD	60	10	420	64	HAR
Myanmar-Scene6	UHD	60	10	420	64	HAR
Myanmar-Scene7	UHD	60	10	420	64	HAR
Narrator	UHD	60	10	420	64	NFX
Pierseaside-Scene1	UHD	60	10	420	64	NFX
Pierseaside-Scene2	UHD	60	10	420	64	NFX
Raptors-Scene1	UHD	60	10	420	64	HAR
Raptors-Scene2	UHD	60	10	420	64	HAR
Readysetgo	UHD	120	10	420	64	UVG

Sequence Name	Res.	Frame rate	Bit depth	Chroma	Number frames	Source
Red-Rock-Vol3-Scene1	UHD	60	10	420	64	HAR
Red-Rock-Vol3-Scene3	UHD	60	10	420	64	HAR
Red-Rock-Vol3-Scene4	UHD	60	10	420	64	HAR
Red-Rock-Vol3-Scene5	UHD	60	10	420	64	HAR
Red-Rock-Vol-Scene6	UHD	60	10	420	64	HAR
Residentialbuilding	UHD	60	10	420	64	SJTU
Ritualdance	UHD	60	10	420	64	NFX
Rollercoaster	UHD	60	10	420	64	NFX
Runners	UHD	60	10	420	64	SJTU
Rushhour	UHD	60	10	420	64	SJTU
Scarf	UHD	60	10	420	64	SJTU
Shakendry	UHD	120	10	420	64	UVG
Skateboarding-Scene12	UHD	60	10	420	64	HAR
Skateboarding-Scene7	UHD	60	10	420	64	HAR
Skateboarding-Scene8	UHD	60	10	420	64	HAR
Snow-Monkeys-Scene1	UHD	60	10	420	64	HAR
Snow-Monkeys-Scene2	UHD	60	10	420	64	HAR
Snow-Monkeys-Scene3	UHD	60	10	420	64	HAR
Snow-Monkeys-Scene5	UHD	60	10	420	64	HAR
Snow-Monkeys-Scene6	UHD	60	10	420	64	HAR
Squareandtimelapse	UHD	60	10	420	64	NFX
Streets-Of-India-Scene1	UHD	60	10	420	64	HAR
Streets-Of-India-Scene2	UHD	60	10	420	64	HAR
Streets-Of-India-Scene3	UHD	60	10	420	64	HAR
Tallbuildings	UHD	60	10	420	64	SJTU
Tango	UHD	60	10	420	64	NFX
Toddlerfountain	UHD	60	10	420	64	NFX
Trafficandbuilding	UHD	60	10	420	64	SJTU
Trafficflow	UHD	60	10	420	64	SJTU
Treeshade	UHD	60	10	420	64	SJTU
Tunnelflag-Scene1	UHD	60	10	420	64	NFX
Tunnelflag-Scene2	UHD	60	10	420	64	NFX
Venice-Scene1	UHD	60	10	420	64	HAR
Venice-Scene2	UHD	60	10	420	64	HAR
Venice-Scene3	UHD	60	10	420	64	HAR
Venice-Scene4	UHD	60	10	420	64	HAR
Windandnature-Scene1	UHD	60	10	420	64	NFX
Windandnature-Scene2	UHD	60	10	420	64	NFX
Wood	UHD	60	10	420	64	SJTU
Yachtride	UHD	120	10	420	64	UVG

C.2 RD peformance of the training sequences across resolutions

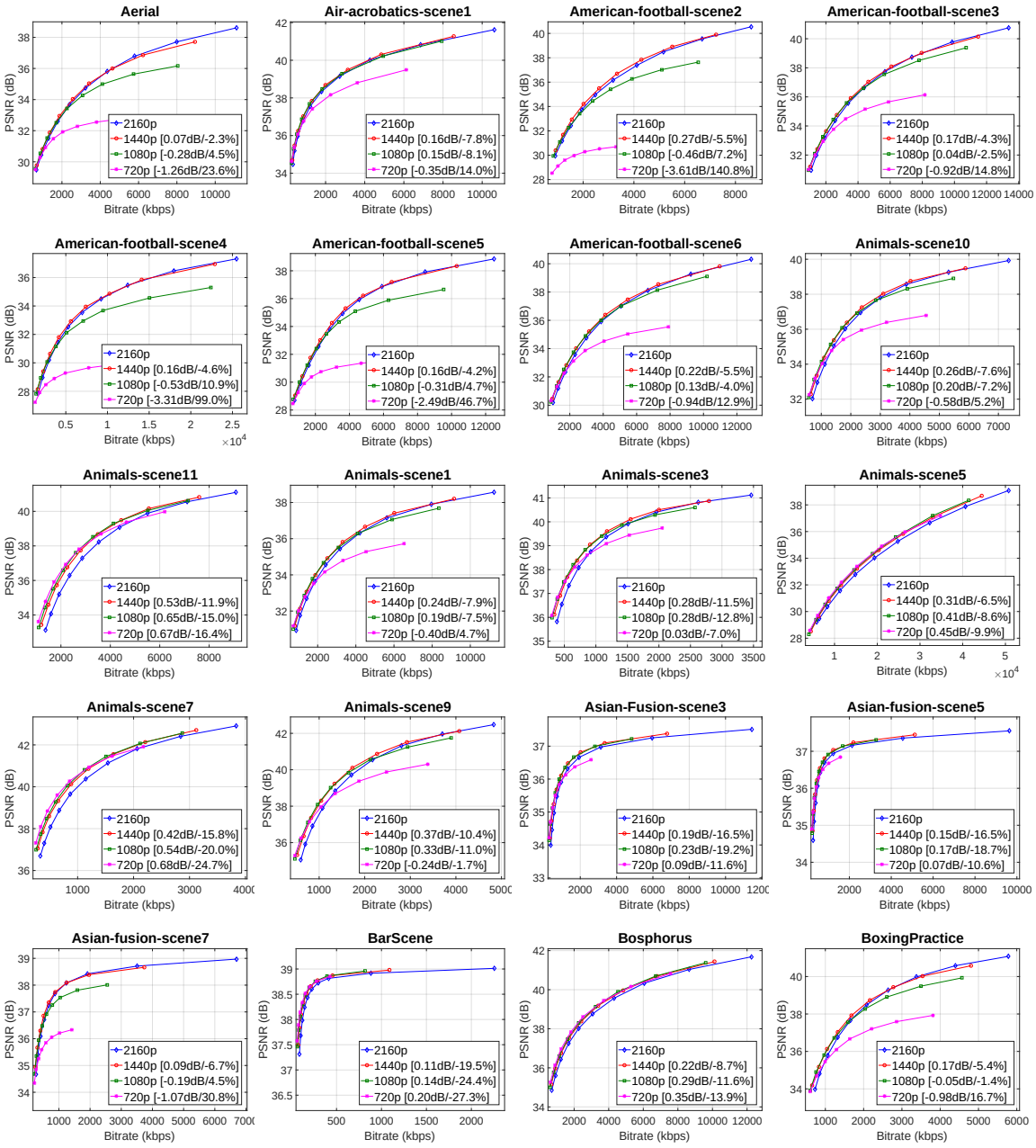
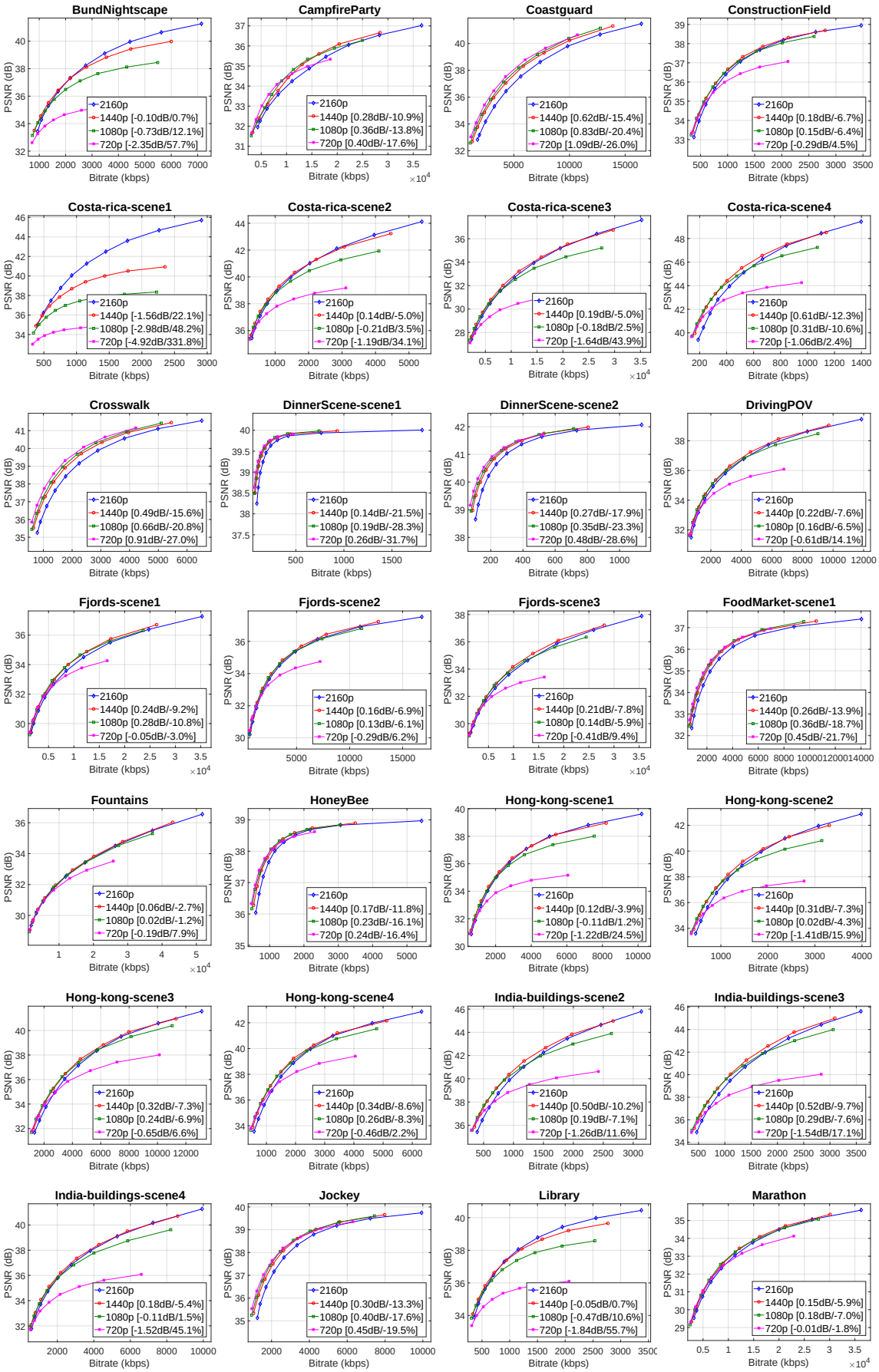
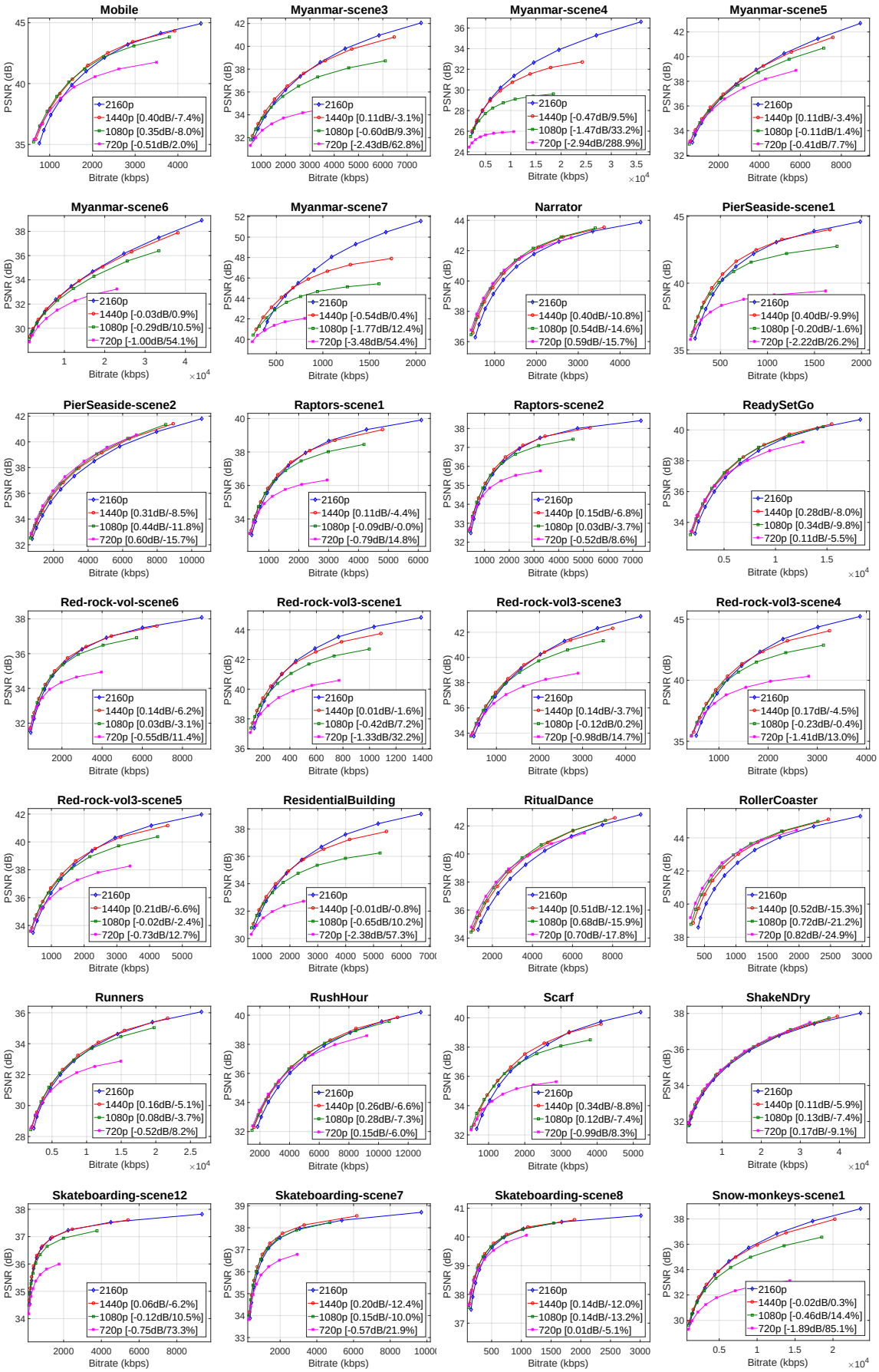
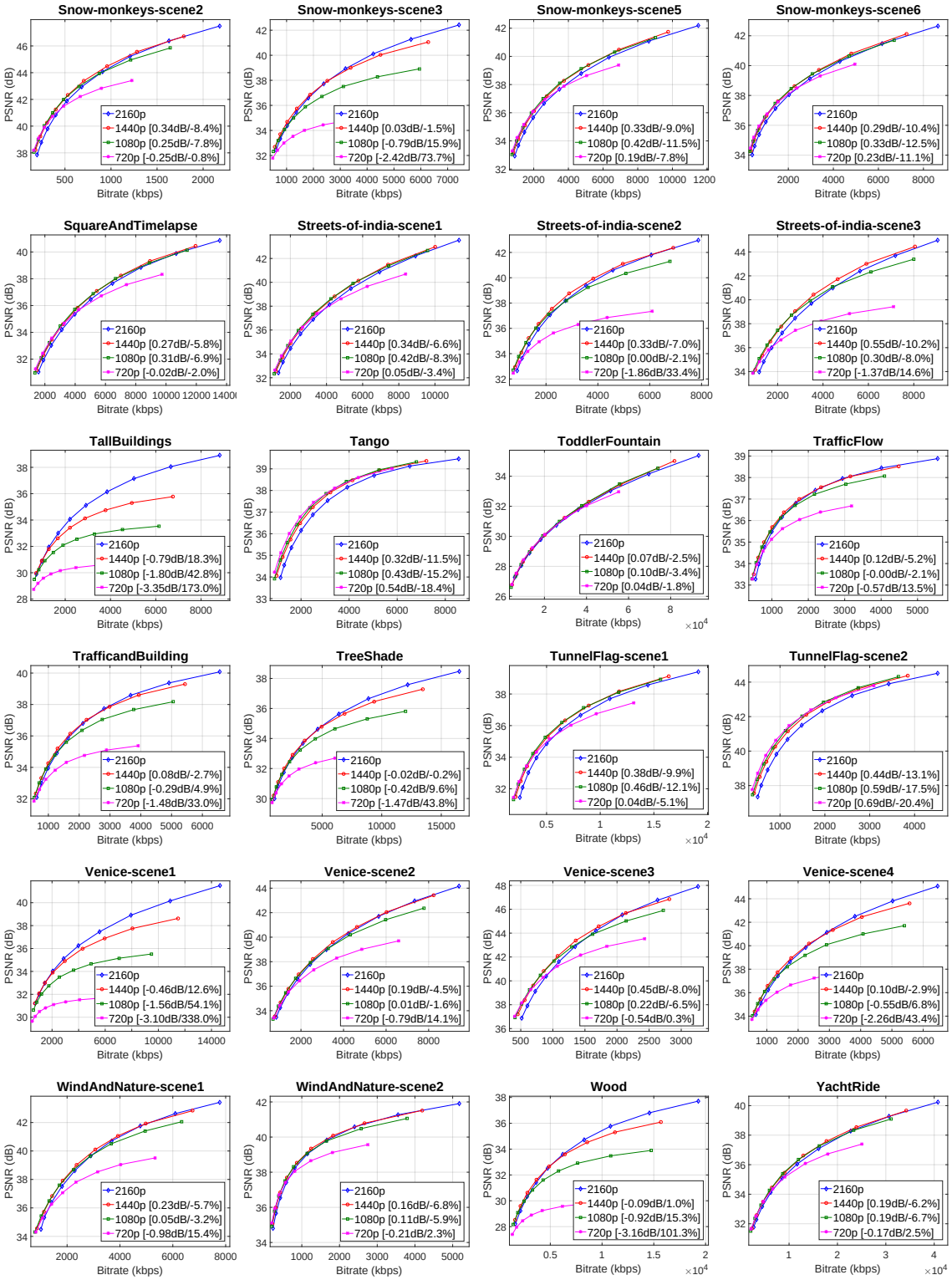


Figure. C.1 RD performance (based on PSNR) of encoding the training sequences with HEVC test model HM 16.18 across different spatial resolutions: {2160p, 1440p, 1080p and 720p}. BD-PSNR and BD-rate is also computed and shown on the legend of each plot.


BundNightscape
2160p
1440p [-0.10dB/0.7%]
1080p [-0.73dB/12.1%]
720p [-2.35dB/57.7%]
CampfireParty
2160p
1440p [0.28dB/-10.9%]
1080p [0.36dB/-13.8%]
720p [0.40dB/-17.6%]
Coastguard
2160p
1440p [0.62dB/-15.4%]
1080p [0.83dB/-20.4%]
720p [1.09dB/-26.0%]
ConstructionField
2160p
1440p [0.18dB/-6.7%]
1080p [0.15dB/-6.4%]
720p [-0.29dB/4.5%]
Costa-rica-scene1
2160p
1440p [-1.56dB/22.1%]
1080p [-2.98dB/48.2%]
720p [-4.92dB/331.8%]
Costa-rica-scene2
2160p
1440p [0.14dB/-5.0%]
1080p [-0.21dB/3.5%]
720p [-1.19dB/34.1%]
Costa-rica-scene3
2160p
1440p [0.19dB/-5.0%]
1080p [-0.18dB/2.5%]
720p [-1.64dB/43.9%]
Costa-rica-scene4
2160p
1440p [0.61dB/-12.3%]
1080p [0.31dB/-10.6%]
720p [-1.06dB/2.4%]
Crosswalk
2160p
1440p [0.49dB/-15.6%]
1080p [0.66dB/-20.8%]
720p [0.91dB/-27.0%]
DinnerScene-scene1
2160p
1440p [0.14dB/-21.5%]
1080p [0.19dB/-28.3%]
720p [0.26dB/-31.7%]
DinnerScene-scene2
2160p
1440p [0.27dB/-17.9%]
1080p [0.35dB/-23.3%]
720p [0.48dB/-28.6%]
DrivingPOV
2160p
1440p [0.22dB/-7.6%]
1080p [0.16dB/-6.5%]
720p [-0.61dB/14.1%]
Fjords-scene1
2160p
1440p [0.24dB/-9.2%]
1080p [0.28dB/-10.8%]
720p [-0.05dB/-3.0%]
Fjords-scene2
2160p
1440p [0.16dB/-6.9%]
1080p [0.13dB/-6.1%]
720p [-0.29dB/6.2%]
Fjords-scene3
2160p
1440p [0.21dB/-7.8%]
1080p [0.14dB/-5.9%]
720p [-0.41dB/9.4%]
FoodMarket-scene1
2160p
1440p [0.26dB/-13.9%]
1080p [0.36dB/-18.7%]
720p [0.45dB/-21.7%]
Fountains
2160p
1440p [0.06dB/-2.7%]
1080p [0.02dB/-1.2%]
720p [-0.19dB/7.9%]
HoneyBee
2160p
1440p [0.17dB/-11.8%]
1080p [0.23dB/-16.1%]
720p [0.24dB/-16.4%]
Hong-kong-scene1
2160p
1440p [0.12dB/-3.9%]
1080p [-0.11dB/1.2%]
720p [-1.22dB/24.5%]
Hong-kong-scene2
2160p
1440p [0.31dB/-7.3%]
1080p [0.02dB/-4.3%]
720p [-1.41dB/15.9%]
Hong-kong-scene3
2160p
1440p [0.32dB/-7.3%]
1080p [0.24dB/-6.9%]
720p [-0.65dB/6.6%]
Hong-kong-scene4
2160p
1440p [0.34dB/-8.6%]
1080p [0.26dB/-8.3%]
720p [-0.46dB/2.2%]
India-buildings-scene2
2160p
1440p [0.50dB/-10.2%]
1080p [0.19dB/-7.1%]
720p [-1.26dB/11.6%]
India-buildings-scene3
2160p
1440p [0.52dB/-9.7%]
1080p [0.29dB/-7.6%]
720p [-1.54dB/17.1%]
India-buildings-scene4
2160p
1440p [0.18dB/-5.4%]
1080p [-0.11dB/1.5%]
720p [-1.52dB/45.1%]
Jockey
2160p
1440p [0.30dB/-13.3%]
1080p [0.40dB/-17.6%]
720p [0.45dB/-19.5%]
Library
2160p
1440p [-0.05dB/0.7%]
1080p [-0.47dB/10.6%]
720p [-1.84dB/55.7%]
Marathon
2160p
1440p [0.15dB/-5.9%]
1080p [0.18dB/-7.0%]
720p [-0.01dB/-1.8%]
PSNR (dB)
Bitrate (kbps)


Mobile
2160p
1440p [0.40dB/-7.4%]
1080p [0.35dB/-8.0%]
720p [-0.51dB/2.0%]
Myanmar-scene3
2160p
1440p [0.11dB/-3.1%]
1080p [-0.60dB/9.3%]
720p [-2.43dB/62.8%]
Myanmar-scene4
2160p
1440p [-0.47dB/9.5%]
1080p [-1.47dB/33.2%]
720p [-2.94dB/288.9%]
Myanmar-scene5
2160p
1440p [0.11dB/-3.4%]
1080p [-0.11dB/1.4%]
720p [-0.41dB/7.7%]
Myanmar-scene6
2160p
1440p [-0.03dB/0.9%]
1080p [-0.29dB/10.5%]
720p [-1.00dB/54.1%]
Myanmar-scene7
2160p
1440p [-0.54dB/0.4%]
1080p [-1.77dB/12.4%]
720p [-3.48dB/54.4%]
Narrator
2160p
1440p [0.40dB/-10.8%]
1080p [0.54dB/-14.6%]
720p [0.59dB/-15.7%]
PierSeaside-scene1
2160p
1440p [0.40dB/-9.9%]
1080p [-0.20dB/-1.6%]
720p [-2.22dB/26.2%]
PierSeaside-scene2
2160p
1440p [0.31dB/-8.5%]
1080p [0.44dB/-11.8%]
720p [0.60dB/-15.7%]
Raptors-scene1
2160p
1440p [0.11dB/-4.4%]
1080p [-0.09dB/-0.0%]
720p [-0.79dB/14.8%]
Raptors-scene2
2160p
1440p [0.15dB/-6.8%]
1080p [0.03dB/-3.7%]
720p [-0.52dB/8.6%]
ReadySetGo
2160p
1440p [0.28dB/-8.0%]
1080p [0.34dB/-9.8%]
720p [0.11dB/-5.5%]
Red-rock-vol-scene6
2160p
1440p [0.14dB/-6.2%]
1080p [0.03dB/-3.1%]
720p [-0.55dB/11.4%]
Red-rock-vol3-scene1
2160p
1440p [0.01dB/-1.6%]
1080p [-0.42dB/7.2%]
720p [-1.33dB/32.2%]
Red-rock-vol3-scene3
2160p
1440p [0.14dB/-3.7%]
1080p [-0.12dB/0.2%]
720p [-0.98dB/14.7%]
Red-rock-vol3-scene4
2160p
1440p [0.17dB/-4.5%]
1080p [-0.23dB/-0.4%]
720p [-1.41dB/13.0%]
Red-rock-vol3-scene5
2160p
1440p [0.21dB/-6.6%]
1080p [-0.02dB/-2.4%]
720p [-0.73dB/12.7%]
ResidentialBuilding
2160p
1440p [-0.01dB/-0.8%]
1080p [-0.65dB/10.2%]
720p [-2.38dB/57.3%]
RitualDance
2160p
1440p [0.51dB/-12.1%]
1080p [0.68dB/-15.9%]
720p [0.70dB/-17.8%]
RollerCoaster
2160p
1440p [0.52dB/-15.3%]
1080p [0.72dB/-21.2%]
720p [0.82dB/-24.9%]
Runners
2160p
1440p [0.16dB/-5.1%]
1080p [0.08dB/-3.7%]
720p [-0.52dB/8.2%]
RushHour
2160p
1440p [0.26dB/-6.6%]
1080p [0.28dB/-7.3%]
720p [0.15dB/-6.0%]
Scarf
2160p
1440p [0.34dB/-8.8%]
1080p [0.12dB/-7.4%]
720p [-0.99dB/8.3%]
ShakeNDry
2160p
1440p [0.11dB/-5.9%]
1080p [0.13dB/-7.4%]
720p [0.17dB/-9.1%]
Skateboarding-scene12
2160p
1440p [0.06dB/-6.2%]
1080p [-0.12dB/10.5%]
720p [-0.75dB/73.3%]
Skateboarding-scene7
2160p
1440p [0.20dB/-12.4%]
1080p [0.15dB/-10.0%]
720p [-0.57dB/21.9%]
Skateboarding-scene8
2160p
1440p [0.14dB/-12.0%]
1080p [0.14dB/-13.2%]
720p [0.01dB/-5.1%]
Snow-monkeys-scene1
2160p
1440p [-0.02dB/0.3%]
1080p [-0.46dB/14.4%]
720p [-1.89dB/85.1%]
PSNR (dB)
Bitrate (kbps)


Snow-monkeys-scene2
2160p
1440p [0.34dB/-8.4%]
1080p [0.25dB/-7.8%]
720p [-0.25dB/-0.8%]
Snow-monkeys-scene3
2160p
1440p [0.03dB/-1.5%]
1080p [-0.79dB/15.9%]
720p [-2.42dB/73.7%]
Snow-monkeys-scene5
2160p
1440p [0.33dB/-9.0%]
1080p [0.42dB/-11.5%]
720p [0.19dB/-7.8%]
Snow-monkeys-scene6
2160p
1440p [0.29dB/-10.4%]
1080p [0.33dB/-12.5%]
720p [0.23dB/-11.1%]
SquareAndTimelapse
2160p
1440p [0.27dB/-5.8%]
1080p [0.31dB/-6.9%]
720p [-0.02dB/-2.0%]
Streets-of-india-scene1
2160p
1440p [0.34dB/-6.6%]
1080p [0.42dB/-8.3%]
720p [0.05dB/-3.4%]
Streets-of-india-scene2
2160p
1440p [0.33dB/-7.0%]
1080p [0.00dB/-2.1%]
720p [-1.86dB/33.4%]
Streets-of-india-scene3
2160p
1440p [0.55dB/-10.2%]
1080p [0.30dB/-8.0%]
720p [-1.37dB/14.6%]
TallBuildings
2160p
1440p [-0.79dB/18.3%]
1080p [-1.80dB/42.8%]
720p [-3.35dB/173.0%]
Tango
2160p
1440p [0.32dB/-11.5%]
1080p [0.43dB/-15.2%]
720p [0.54dB/-18.4%]
ToddlerFountain
2160p
1440p [0.07dB/-2.5%]
1080p [0.10dB/-3.4%]
720p [0.04dB/-1.8%]
TrafficFlow
2160p
1440p [0.12dB/-5.2%]
1080p [-0.00dB/-2.1%]
720p [-0.57dB/13.5%]
TrafficandBuilding
2160p
1440p [0.08dB/-2.7%]
1080p [-0.29dB/4.9%]
720p [-1.48dB/33.0%]
TreeShade
2160p
1440p [-0.02dB/-0.2%]
1080p [-0.42dB/9.6%]
720p [-1.47dB/43.8%]
TunnelFlag-scene1
2160p
1440p [0.38dB/-9.9%]
1080p [0.46dB/-12.1%]
720p [0.04dB/-5.1%]
TunnelFlag-scene2
2160p
1440p [0.44dB/-13.1%]
1080p [0.59dB/-17.5%]
720p [0.69dB/-20.4%]
Venice-scene1
2160p
1440p [-0.46dB/12.6%]
1080p [-1.56dB/54.1%]
720p [-3.10dB/338.0%]
Venice-scene2
2160p
1440p [0.19dB/-4.5%]
1080p [0.01dB/-1.6%]
720p [-0.79dB/14.1%]
Venice-scene3
2160p
1440p [0.45dB/-8.0%]
1080p [0.22dB/-6.5%]
720p [-0.54dB/0.3%]
Venice-scene4
2160p
1440p [0.10dB/-2.9%]
1080p [-0.55dB/6.8%]
720p [-2.26dB/43.4%]
WindAndNature-scene1
2160p
1440p [0.23dB/-5.7%]
1080p [0.05dB/-3.2%]
720p [-0.98dB/15.4%]
WindAndNature-scene2
2160p
1440p [0.16dB/-6.8%]
1080p [0.11dB/-5.9%]
720p [-0.21dB/2.3%]
Wood
2160p
1440p [-0.09dB/1.0%]
1080p [-0.92dB/15.3%]
720p [-3.16dB/101.3%]
YachtRide
2160p
1440p [0.19dB/-6.2%]
1080p [0.19dB/-6.7%]
720p [-0.17dB/2.5%]
PSNR (dB)
Bitrate (kbps)
×10^4

Appendix D

Characterization of test sequences

Table D.1 Specifications of the 14 test sequences used for this work. All sequences contain 300 frames.

#	Sequence name	Resolution	Bit depth	Framerate	Chroma	Source
S01	CalmingWater	UHD	10	60	4:2:0	BVI-Texture
S02	CatRobot1	UHD	10	60	4:2:0	B<>COM
S03	CentralLineCrossing	UHD	10	50	4:2:0	BBC
S04	Chimera-ep08	UHD	10	60	4:2:0	Netflix
S05	Chimera-ep12	UHD	10	60	4:2:0	Netflix
S06	Chimera-ep17	UHD	10	60	4:2:0	Netflix
S07	Chimera-ep20	UHD	10	60	4:2:0	Netflix
S08	DaylightRoad2	UHD	10	60	4:2:0	Huawei
S09	ElFuente-scene33	UHD	10	60	4:2:0	Netflix
S10	FoodMarket3	UHD	10	60	4:2:0	Netflix
S11	LampLeaves	UHD	10	60	4:2:0	BVI-Texture
S12	Manege	UHD	10	60	4:2:0	BBC
S13	NingyoPompoms	UHD	10	50	4:2:0	BBC
S14	Petibato	UHD	10	60	4:2:0	BBC

Appendix E

Additional GAN visual comparisons

(a) FoodMarket @ 1500 Kbps

(b) Petibato @ 1700 Kbps

(c) Manege @ 4300 Kbps

Figure. E.1 Visual comparison between the anchor, HM 16.18 at a fixed resolution (left), and our proposed resolution adaptation framework using the GAN model (right), for three test sequences. The patches were extracted from frames 161, 1 and 161, respectively.