UNIVERSITA DEGLI STUDI DI PAVIA
DOTTORATO IN INGEGNERIA ELETTRONICA,
INFORMATICA ED ELETTRICA

CICLO XXIX
A.A 2013-2016

Adaptive video coding to optimize video streaming in Internet and wireless ad-hoc networks

A DISSERTATION
SUBMITTED TO THE DEPARTMENT
OF ELECTRONICS ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF UNIVERSITY OF PAVIA
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Amparito Alexandra Morales Figueroa
2016

SUPERVISOR: Prof. Lorenzo Favalli

Esta tesis va dedicada a mis padres
Por su amor infinito y apoyo constante

Acknowlegments

First and foremost, I have to thank my parents for all the love, support and encouragement received throughout my life. Thank you both for believing in me, by the teachings and values instilled, and for all you have done to give me always the best. To my beloved grandma whose role in my life was and remains immense. I am sure she is proud of me, and is smiling from heaven.

A great thank you to my dear husband Gianni for his patience, love and support during the all PhD period. His determination and desire to success inspired me to continue reaching my goals without fainting. You give me the strength to overcome the adversities, I love you!.

I would also like to sincerely acknowledge to my advisor, Prof. Lorenzo Favalli for the assistance in the development of this thesis. I appreciate all his contributions of time and knowledge, which have contributed to do this PhD a productive and stimulating experience. I wish also to express my gratitude to Prof. Paolo Gamba for his guidance and support through this phase.

To my lab colleagues, thank you for your friendship, help, advices and for encouraging me in many moments of crisis. Moreover, I am grateful with the University of Pavia and Cariplo Foundation for the financial support.

Thank you, God, for always be with me and guide my life.

In memory of Prof. Luiz Fernando Gomes Soares (LF), an outstanding professional and an extraordinary and inspiring person.

Contents

Chapter 1

Introduction

Delivering variable bit-rate streaming over the Internet is a challenging task. Normally, transmission of video requires high bandwidth and low delay, while a certain amount of packet loss can be accepted. In such way, in order to cope with bandwidth variations, the use of adaptive and robust coding schemes as well as different mechanisms to control and restrict the packet loss are required.

H.264/SVC (Scalable Video Coding) emerges as an interesting and innovative video coding extension of the H.264/AVC (Adaptive Video Coding) standard. SVC provides a unique scalable bit-stream encoded at the highest quality, thus, it can be decoded from partial streams according to the different user requirements and capabilities (i.e. resolution, frame rate or quality). Thus, problems such as storage, video management at the server level, and redundancy of packets from the video can be solved. SVC provides three types of scalability: spatial, temporal and fidelity scalability. In turn, the last one offers three modes of scalability: Coarse-Grain scalability (CGS), Finer-Grain scalability (FGS) and Medium-Grain scalability (MGS). This thesis exploits the benefits of MGS to produce more bit-rates using a reduced number of enhancement layers, and thus improving the rate-distortion. The features of the H.264/SVC encoder, as well as the selection of the encoding parameters, are discussed in Chapter 2.

Moreover, in order to increase the possibilities of receiving the transmitted video content at the receiver side, *Multiple Description Coding* (MDC) techniques jointly with SVC are presented as a viable solution. The purpose of MDC is to provide path diversity using one or more video descriptions, which can be sent through diverse paths in the network. There are several MDC methods used to create these descriptions. It is important to mention that two or more descriptions can be generated depending on the application; in this thesis we use two descriptions. Furthermore, we evaluate two common MDC approaches: spatial-domain and temporal-domain. The first one generates the descriptions dividing a frame into two sub-images composed by even and odd rows. The second one uses all the odd frames to create the first description, and the even frames to create the second one. In Chapter 3, the performances of the aforementioned MDC methods is evaluated and compared. Additionally, in section 3.3, some of the H.264 encoding schemes, such as Adaptive Video Coding, SVC with MGS and MDC are contrasted. In this manner, it is possible to analyze and define the scenarios where each one of these schemes gives a good performance and produce a suitable Peak Signal-to-Noise (PSNR).

On the other hand, the estimation of the available bandwidth (AB) of an end-

to-end path is critical in video streaming applications. In the literature, there are many tools which use different techniques to estimate the available bandwidth. The more relevant tools are based on the probe rate model (PRM) and probe gap model (PGM). These two mechanisms employ packet pair dispersion to estimate the AB. Based on [1] which develops a tool called *Traceband*, we propose a bandwidth estimation approach based on Hidden Markov Model (HMM-BE) applied to SVC video sequences. The bandwidth model establishes that the AB in the network at specific instant of time can be modeled by a Hidden Markov chain, where each state represents a certain level of availability (i.e. bit-rate). Using discrete observations representing probing packets pair dispersions (i.e. the delay measured, at the receiver side, between two consecutive packets) is possible to estimate the dynamics of the network. The estimation of the AB permits a better use and management of the resources. For instance, buffer fullness depends on the difference between encoding rate and available bandwidth. The details about the proposed Bandwidth Estimation Model (BEM), the used parameters, as well as obtained results are presented in Chapter 4.

During video transmission, factors, such as bandwidth decreasing, and buffer fullness may produce packet loss, and thus affecting the final quality of the video. To face this problem, buffer management and drop packet policies are proposed in the literature. The main related works are detailed at the beginning of Chapter 5. These works include among their solutions, buffer management schemes assigning a specific priority, either to the packets or frames before dropping them. Moreover, a recent study proposes to use multiple buffers with the purpose of applying a priority layer level.

A critical point detected in the existent works is the way in which the priority of packets is defined. This priority is only associated to the temporal level (i.e. frame rate). In other cases, the discarding of packets is governed by a drop probability determined by the queue size and the packets transmission probability. Using a different approach which exploits the advantages of SVC, and especially quality scalability (MGS mode), this thesis proposes a quality discard packets (QDP) strategy. This approach, besides of considering the status of the queue, defines the priority of a packet using the information concerning the quality and temporal level, and the dependencies among packets. In such way, the most relevant data (i.e. base layer data) is certainly transmitted. Then, according to the available network conditions and buffer resources, the quality can be improved as much as possible.

Finally, considering the fact that in the last years video streaming has had a massive growth, due to the increase of smart devices and the deployment of high speed wired and wireless networks, it is essential to use an adaptive solution which is able of working in heterogeneous environments, while provides both QoS and Quality of Experience (QoE) to the users. Dynamic Adaptive Streaming over HTTP (DASH) is presented as an adequate and valuable solution for video streaming delivery. The main advantage of DASH lies in offering several versions of the same content, where each one of these versions owns different features (i.e. video resolution, frame rate and quality). Therefore, depending on the characteristics of the client's device, capabilities of the network and available resources at a specific instant of time, DASH permits the distribution of customized and suitable content.

However, the benefit of satisfying many different devices implies negatively on the storage requirements. To deal with this issue, the blending of DASH and SVC

represents an interesting form of providing many representations of the content (i.e. content with different encoding parameters), without storing redundant information which occupies unnecessary space into the DASH server. Chapter 6 addresses the DASH specification as well as the advantages of using SVC with the aim of providing flexibility to DASH. On the other hand, at the client side, the QoE perceived by the user could be affected by various parameters, such as the number of switches between the different available qualities, and the amount of interruptions produced during the video playback. Accordingly, the adoption of bit-rate/quality adaptation techniques are imperative to obtain a smooth transition between available qualities, and thus guarantee a pleasant display of the requested video.

In order to pick the most suitable representation, the DASH client typically employs an adaptive bitrate selection (ABR) algorithm. The ABR algorithms proposed in the literature commonly use parameters like the average segment download, available bandwidth or buffer occupancy. In a recent work presented in [2] is exposed the fact that despite of video segments in DASH have the same playback duration, these do not own the same segment size. In such way, this information can be used to estimate a more accurate download time required by the client, and thus define the most appropriate representation which improves the video quality delivered. However, [2] presents two main drawbacks. The first one is the high demand of storage capacity due to the use of AVC instead of SVC; the other one refers to the Media Presentation File (MPD). Specifically, the MPD file, which contains the data about the available representations, is modified by the addition of a new parameter that represents the segment size. In this thesis we propose a *DASH Quality Decision Algorithm* (DQD) which consists in using the sizes of the segments to determine the best segment representation that should be downloaded, but without altering the MPEG-DASH specification (i.e. keeping the MPD as established into the standard).

In summary, this thesis presents three main contributions: a Bandwidth estimation model based on HMM to estimate the network conditions; A discard packets strategy which considers the priority of the packets before discard them. This priority is assigned considering the quality (Qid) and temporal (Tid) level of a packet. The proposed discard algorithm takes into account the current available network bandwidth, given by a Bandwidth Estimation Model (BEM), to apply an intelligent discard of the packets stored in the buffer. Our *QDP* strategy ensures that the video transmitted owns the highest possible quality under the given network conditions and buffer resources; and a DASH quality decision algorithm which improves the QoE perceived by users. It minimizes the number of switches, offers a smooth transition between available qualities, and avoid presentation waiting times.

1.1 Motivation

The quality of the video received and the playback delay have a crucial impact in the QoE perceived by the user. The quality can be improved depending on the amount of available transmitted packets at the receiver side. However, the decisive factor to recover all the frames which comprise the original video sequence, is related with the type of the packets recovered and its importance. Regarding that H.264/SVC defines a hierarchical architecture, where the upper layers depend to the bottom ones, it is essential to consider this fact at the moment to drop a packet. Because in case of one or more “important” packets are dropped (i.e. packets belonging to

the base layer), the decoder will not be able to reconstruct the video sequence with the same features as the original one.

On the other hand, the QoE in MPEG-DASH video streaming applications is affected by the number of switches realized, and abrupt quality transitions. In order to provide a likable presentation, the aforementioned issues must be controlled and minimized.

1.2 Objectives

The current thesis has three main general objectives:

- Develop a bandwidth estimation model based on Hidden Markov Model, which is able to determine the network conditions.
- Define a buffer management strategy which exploits the packet priority information from the H.264/SVC encoder, considers network status and monitors the buffer fullness.
- Develop a quality adaptation technique to MPEG-DASH applications which takes advantage of scalable video coding to minimize the up-down quality switches. It attempts to preserve the highest quality, delivering a smooth presentation, thus improving the QoE.

Moreover, some specific targets are derived from the objectives cited above.

- Since that a suitable encoding configuration can lead to attain a better quality, an initial goal of this work is to determine the appropriate encoding parameters to SVC, especially when the quality scalability mode (SNR) is used.
- Compare the performance produced by the H.264/SVC-MGS with other H.264 encoding approaches, such as Adaptive Video Coding (AVC) and MDC. Thus, it is achievable to define the scenario in which each one of these schemes deliver better results.
- Implement a DASH client and server which employ SVC-MGS to add more flexibility to DASH.

1.3 Thesis structure

This thesis is organized as follow. Chapter 2 addresses the scalable video coding standard (H.264/SVC), the types of scalability and its principal features. Chapter 3 analyzes and evaluates Multiple Description Coding techniques in combination with SVC. In Chapter 4, a Bandwidth Estimation Model based on Hidden Markov Model is proposed. Chapter 5 presents the developed buffer management strategy and describes the *Quality Discard Packets* algorithm. In Chapter 6, a MPEG-DASH Server and Client using SVC-MGS are implemented. Additionally, our *DASH Quality Decision* algorithm is introduced. Finally, in chapter 7 the conclusions and possible future works are summarized.

Chapter 2

Adaptive and robust video coding techniques

2.1 Scalable Video Coding (H.264/SVC)

H.264/SVC is the scalable video coding extension of the H.264/MPEG-4 Advanced Video Coding (AVC) standard (H.264/AVC) [3]. Both the H.264/AVC standard and its SVC extension were developed by the Joint Video Team (JVT), which is constituted by experts from ITU-T's Video Coding Experts Group (VCEG) and ISO/IEC's Moving Pictures Experts Group (MPEG). SVC extends the target applications of the H.264/AVC standard to enable video transmission with heterogeneous clients and multicast with clients of different capabilities [4]. Efficient SVC provides a number of benefits in terms of applications. In contrast with AVC, where to attain bit-stream adaptation, the same video is encoded with different characteristics. A SVC-coded signal is encoded once at the highest quality with appropriate packetization. Hence, it can be decoded from partial streams for a specific resolution, frame rate or quality requirements. In such way, problems as storage, video management at server level and redundancy packages from the video can be solved. Another benefit of SVC is that a scalable bit stream usually contains parts with different importance in terms of decoded video quality. This property in conjunction with unequal error protection is especially useful in any transmission scenario with unpredictable throughput variations and/or relatively high packet loss rates [5].

Similar to the underlying H.264/AVC standard, the SVC design includes a video coding layer (VCL) and a network abstraction layer (NAL). While the VCL represents the coded source content, the NAL formats the VCL representation and provides header information appropriate to conveyance by transport layers and storage media. The coded video data is organized into NAL units, which represent packets with an integer number of bytes. NAL units are classified into VLC NAL units, which contain coded video data, and non-VLC NAL units, which provide associated additional information. An access unit is a set of NAL units that results in exactly one decoded picture. A set of successive access units with certain properties constitutes a coded video sequence, which represents and independently decodable part of a bit-stream.

Scalability is provided at the bit-stream level. A bit-stream with reduced spatio-temporal resolution and/or fidelity can be obtained by discarding NAL units or corresponding network packets from a scalable bit-stream. In figure 2.1, a simplified

block diagram of an SVC encoder is illustrated. SVC defines a hierarchical structure composed by one base layer, which is independently decodable, and one or several enhancement layers that employ data of the lower layers for coding. These last are just decoded if all the data belonging to the bottom layers is available. H.264/SVC supports up to 128 layers, but the number of layers present in an SVC bit stream depends on the application needs. With the currently specified profiles, the maximum number of enhancement layers is constrained to 47 layers. The input pictures of each spatial or fidelity layer are split into macroblocks and slices. A macroblock (MB) represents a square area of 16x16 luma samples and, in the case of 4:2:0 chroma sampling format, 8x8 samples of the two chroma components. The macroblocks are organized in slices that can be parsed independently. For the purpose of intra prediction, motion-compensated prediction, and transform coding, a MB can be split into smaller partitions or blocks [4].

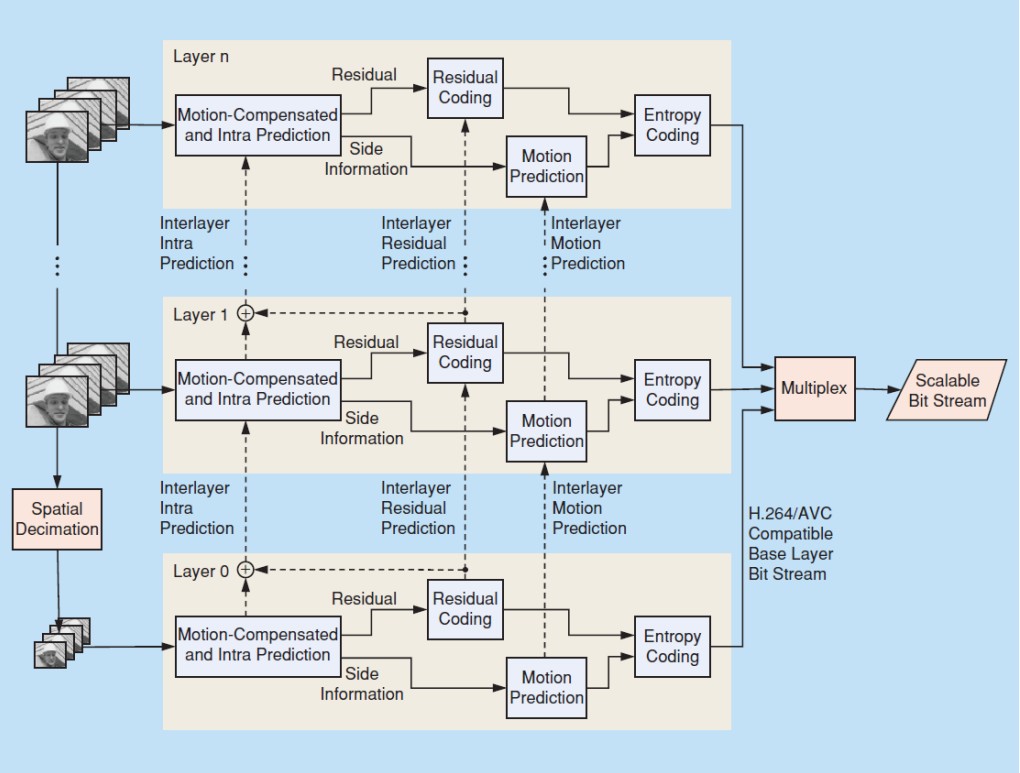


Figure 2.1: SVC encoder structure [4]

2.2 Types of scalability

The SVC extension of H.264/AVC presents three types of scalability: Temporal, Spatial and Quality scalability. Encoded video streams can be composed by three distinct types of frames: I (intra), P (predictive) or B (bi-predictive). I frames only explore the spatial coding within the picture and do not use references to any other picture. On the contrary, both P and B frames do have interrelation with different pictures, as they explore directly the dependencies between them. While in P frames inter-pictures predictive coding is performed based on one preceding reference picture, B frames consist of a combination of inter-picture and bi-predictive coding. Additionally, the H.264 standard requires the first frame to be an Instantaneous

Decoding Refresh (IDR) access unit, which corresponds to the union of one I frame with several critical non-data related information. Mainly, the Group of pictures (GOP) structure specifies the arrangement of those frames within an encoded video sequence [6].

2.2.1 Temporal scalability

Temporal scalability consists in representing video content with different frame rates by as many bit-stream subsets as needed. In H.264/SVC, the bases of temporal scalability are on the GOP structure, since it divides each frame into distinct scalability layers (by jointly combining I, P and B frames). Temporal scalability is offered using the hierarchical B pictures, where the most popular one is the dyadic prediction structure. This structure is constructed by a hierarchical GOP which starts with an IDR picture and ends with either I or P picture. In general, the GOP is composed by 8 pictures. Enhancement layer pictures are typically coded as B pictures, the B_i picture of level $i > 1$ belonging to temporal layer T_i may uses the surrounding of the key pictures I or P in addition to the B_j picture, such that level j is lower than i with T_j for prediction [7].

2.2.2 Spatial scalability

SVC supports multilayer coding in order to supply resolution variety. Each layer corresponds to a spatial resolution, where the base layer owns the lowest resolution which is increased with the addition of the enhancement layers. The maximum number of spatial layers can be no more than 7. In order to reduce redundancy between spatial layers and improve the coding efficiency, inter-layer prediction is applied. In fact, it exploits spatial layer's information to predict the next layer's inter residual data, motion vector (MV) and intra texture.

There are three additional inter-layer prediction concepts that have been added in SVC: inter-layer intra prediction (ILIP), inter-layer residual prediction (ILRP) and inter-layer motion prediction (ILMP). The first one is defined to predict the enhancement layer macroblock from the previous layer intra MB. The second category (ILRP), is used to work with the residual information coded in a lower resolution. On the other hand, when the reference MB is inter-coded, the enhancement MB is also inter-coded. Thus, the motion vectors, the partitioning data and the reference indexes are derived from lower resolution using ILMP.

2.2.3 SNR/Quality scalability

Quality scalability can be considered as a special case of spatial scalability with identical spatial-temporal resolution for base layer and enhancement layers. The difference is that diverse fidelity levels (bit-rate) can be specified to each one of them. The H.264/SVC supports three quality scalability modes such as Coarse-Grain Scalability (CGS), Finer-Grain Scalability (FGS) and Medium-Grain Scalability (MGS).

This thesis focuses on the quality scalability in H.264 SVC and particularly in the MGS mode. The different aforementioned quality scalable methods are addressed below, given special detail on MGS.

Coarse-Grain scalability (CGS)

Similar to spatial scalability, but keeping constant the spatial resolution, CGS uses inter-layer prediction mechanisms without up-sampling operations. Further the inter-layer residual and intra prediction are achieved in the transform domain. In CGS, the residual texture signal in the enhancement layer is re-quantized with a quantization step size, that is smaller than the quantization step size of the preceding CGS layer. Moreover, SVC supports up to eight CGS layers, corresponding to eight quality rate points, but the inter-layer prediction is constrained to at most three CGS layers including the base layer.

Fine-Grain Scalability (FGS)

This approach employs the concept of Progressive Refinement (PR) slices, where each PR slice represents a refinement of the residual signal. This refinement corresponds to a bisection of the quantization step size. The key issue of FGS is the order in which the transform coefficient levels are allocate into the PR slices. This singularity allows to truncate the corresponding PR Network abstraction layer (NAL) units, at any arbitrary byte-aligned point. Hence, the quality of the SNR base layer can be refined in a finer granularity way.

FGS presents a low performance because only the base layer is used to Motion Compensated Prediction (MCP). Additionally, the coding efficiency of the enhancement layers is notably decreased in comparison to single-layer coding. In the other approaches, such as CGS and MGS, the MCP uses the highest quality reference of temporal refinement pictures. This difference improves the coding efficiency without increasing the complexity when hierarchical prediction pictures are used.

Medium-Grain scalability (MGS)

This mode uses similar techniques of CGS. However, MGS provides more flexibility for bit stream adaptation increasing the number of rate points. While CGS supports quality scalability by dropping complete enhancement layers, MGS provides a finer granularity level of quality scalability by partitioning a given enhancement layer into several MGS layers. So, an individual MGS layer can then be removed at any point in the bit stream for quality and bit rate adaptation.

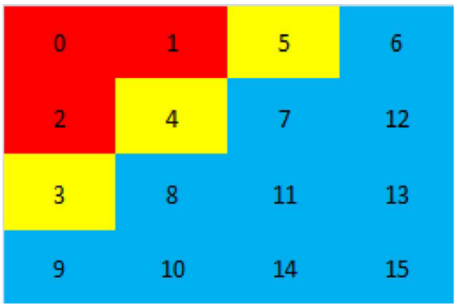
Medium grain scalability splits a given enhancement layer of a given video frame into up to 16 MGS layers. Specifically, MGS divides the transform coefficients, obtained through transform coding of a given macroblock, into multiple groups. Each group is assigned to a prescribed MGS layer.

In order to explain how MGS works, initially we consider a 4 x 4 macroblock and denote W_m, where m=1,2,...,16, represents the number of transform coefficients in MGS layer m within an enhancement layer. The number of transform coefficients W_m is also referred to as the "weight" of MGS layer m. Figure 2.2 (a) shows the splitting of the transform coefficients of a 4 x 4 macroblock intro three MGS layers with the weights W=[3,3,10], where each MGS layer of a given video frame forms a single NALU. It is important to note that a macroblock can be divided into sixteen MGS layers, each one of them containing one transform coefficient [8].

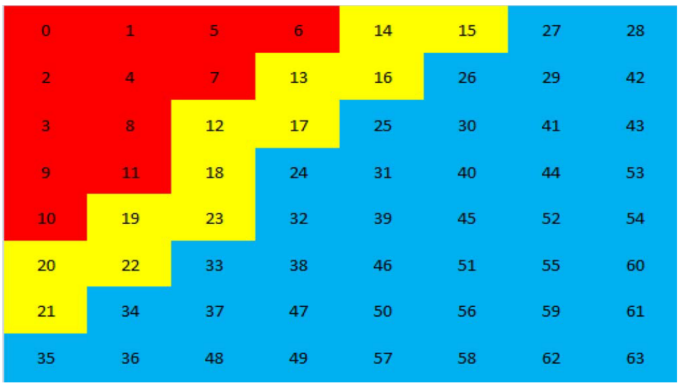
When 8 x 8 macroblocks are used, there are two approaches for splitting the transform coefficients. The first one is usually employed in combination with context-

adaptive variable length coding (CALVC) entropy coding, and it consists in dividing a given 8 x 8 macroblock into four 4 x 4 submacroblocks, and then breaking up the coefficients as indicated above. On the other hand, when the context-based adaptive binary arithmetic coding (CABAC) is utilized, the 8 x 8 macroblock is not subdivided. Instead, the approach used for splitting the transform coefficients of a 4 x 4 macroblock is extended to the 8 x 8 macroblock by multiplying each weight by a factor of four, as it is depicted in Figure 2.2 (b).

With MGS encoding, the video bit-rate is adjusted by dropping enhancement layer NALUs , one at a time, until the target bit rate is achieved. The NALUs are dropped in decreasing order, the NALUs from the highest indexed MGS layer are discarded first, and then if additional rate reduction is demanded, NALUs from the next highest MGS layer are dropped, and so on.



(a) 4x4 macroblock



(b) 8x8 macroblock

Figure 2.2: Splitting transform coefficients into MGS layers regarding a weight vector W=[3, 3, 10]. The coefficients in red constitute MGS layer 1, while the coefficients in yellow constitute MGS layer 2 and the remaining coefficients constitute MGS layer 3.

2.3 Setting encoding parameters

For the encoding process, we use the JSVM (Joint Scalable Video Model) software, which is the reference software for the Scalable Video Coding (SVC) project of the Join Video Team (JVT). The JSVM Software is written in C++ and is provided as source code. Moreover, the JSVM is still under development and it changes frequently. In this thesis, we use JSVM version 9.19.14 [9].

Table 2.1: Static configuration parameters for the main encoding file .

Parameter	Value	Description
OutputFile	out.264	Specifies the file name for the bit-stream to be encoded.
FrameRate	30	Frame rate of the input sequence.
NonRequiredEnable	1	Non-required picture SEI messages are included.
CgsSnrRefinement	1	SNR enhancements are coded using MGS.
EncodeKeyPictures	1	Pictures at temporal level 0 are coded as key pictures. For MGS configurations a value of 1 improves the error robustness as well as the supported degree of bit-stream adaptation.
IntraPeriod	-1	IntraPeriod equal to -1, only the first picture is intra coded.
BaseLayerMode	2	Specifies whether sub-sequence SEI messages are included for the base layer. 2 - AVC compatible base layer with sub-sequence SEI messages for supporting temporal scalability without prefix NAL units.
SearchMode	4	Specifies the motion search algorithm to be applied. The fast motion (value equal to 4) provides a comparable rate-distortion efficiency, but reduced the encoding time.
SearchFuncFullPell	3	Specifies the distortion measure that is applied for the motion search on integer-sample positions. Value 3: Sum of absolute differences (SAD) for all color components.
BiPredIter	4	Specifies the number of iterations of the motion search iterations for bi-predictive blocks. The coding efficiency for B pictures is usually increased when the parameter is set to a value greater than or equal to 2.
IterSearchRange	8	Search range for the motion search iterations for bi-predictive blocks.
Lardo	1	Specifies whether loss-aware rate-distortion optimized macroblock mode decision is used.
PreAndSuffixUnitEnable	1	Specifies whether to add prefix NAL units before the NAL units of AVC slices. This parameter shall always be on in SVC contexts.

Considering that the main target is to improve the video quality, it is crucial to define the key parameters that should be set in the encoder to obtain the best coding efficiency as well as achieve the best peak signal-to-noise ratio (PSNR). Based on the existing literature, some meaningful considerations that should be taken into account to encode videos using the scalable scheme, especially MGS, are detailed below.

There are two principal parameters which demand to be optimized: the quantization parameter (QP) and the MGS fragmentation. The other parameters, such as GOP size, GOP pattern, entropy coding scheme, inter-layer prediction options, etc. are selected considering the type of video sequence and its application. Empirically, [10] found that a GOP size equal to 8 or 16 usually achieves the best rate-distortion performance. Furthermore, the GOP size also determines the total number of temporal layers. Another notable observation is that the number of frames to be encoded are closely related with the GOP size. Thus, this number might be a multiple of the GOP size plus 1, to its maximum (The extra one is added for accomplishing the reference pictures of the hierarchical B structure). For instance, if the GOP size is equal to 16, 289 frames are used for image sequence of 300 frames.

On the other hand, [10] also states that the value of the *CgsSnrRefinement* parameter will not change the coding results if no more than 3 CGS layers are used. At the same time this work mentions the fact that MGS sub-layers degrade PSNR performance. However, when incorporating MGS layers, the encoding time is significantly reduced as compared to using CGS alone. Therefore, the point is to find an optimal balance between the number of CGS layers and the number of MGS layers to get a good PSNR with a high coding efficiency.

In [8] some encoder options and parameters that should be used in order to get a higher rate-distortion (RD) efficiency are raised. From experiments, [8] analyzes the GoP pattern and concludes that it should be selected considering the type of video sequence, the frame rate and its resolution. However, a GoP pattern with a great number of hierarchical B frames (e.g. IBBBBBBBBBBBBBBB , 16 frames with 15 B frames per I frame) may gives better RD performance compared to other patterns with a reduced number of B frames, such as G16B7, G16B3 and G16B1. Moreover, the MeQP values (encoder parameters of the JSVM software) which determine the Lagragian[1] parameters for motion estimation and mode decision of key pictures, should be set to values smaller than the QP values to attain a higher RD performance. This setting resulted in RD-efficient coding. The macroblock adaptive inter-layer prediction, which employs a rate distortion optimization framework, is used (InterLayerPred, ILMotionPred = 2). Likewise, the CABAC coding scheme and the 8x8 transform (Enable8x8Transform=1) are employed. The combination of sum of absolute difference (SAD) for full pixel (SearchFuncFullPel=3), Hadamard for sub pixel motion estimation (SearchFuncSubPel=2) and the Fast Search Block (SearchMode=4) with a Search Range of 16 are employed. Furthermore, a decisive parameter to the RD performance is the existing distance between the QPs of the encoded layers, which is named Δ QP. A wider spread between base and enhancement layer quantization parameters (eg., B=40, E=25) provides a wider range of quality (and corresponding bit-rate adaptation) at the expense of slightly reduced RD performance, when compared to encodings with a narrower quantization parameter

[1]The Lagrange multiplier method is used to facilitate the selection of the appropriate coding parameters, in order to minimize the video distortion D for a given rate R_c.

spread.

Based on the considerations described above, the main configuration file as well as the configuration files for the base and enhancement layers were defined, and they are presented in Table 2.1 - 2.6. Moreover, the configuration parameters were classified in two groups: statics and variables. The first one is related with the parameters which are required to obtain the best coding efficiency when MGS is used. On the other hand, the variable parameters can take different values for trying to obtain better results.

Table 2.2: Variable configuration parameters for the main encoding file .

Parameter	Value	Description
GOPSize	16	This parameter must be equal to a power of 2. The maximum allowed value is 64. The GOP size is specified at the frame rate given by FrameRate.
FramesToBeEncoded	289	The maximum value to FramesToBeEncoded is a multiple of GOPSize plus 1.
MGSControl	2	Specifies what pictures are used as references for motion estimation and motion compensation. Value 2: pictures of highest EL are used for ME and MC. Value 0: corresponds to CGS coding.
SearchRange	32	Specifies the maximum search range for the motion search.
NumLayers	4	Number of layers.

Table 2.3: Static configuration parameters for the Base Layer file.

Parameter	Value	Description
SymbolMode	1	Specifies the entropy coding mode. 1 - the video sequence is encoded using context-adaptive binary arithmetic coding (CABAC). CABAC usually provides an increased coding efficiency.
InterLayerPred	0	Specifies the usage of inter-layer prediction. It shall always be equal to 0 for the base layer.
ILModePred	0	Specifies the usage of inter-layer mode prediction. ILModePred should always be equal to 0 for the base layer.
ILResidualPred	0	Specifies the usage of inter-layer residual prediction. For the base layer is always 0.
MGSVectorMode	0	MGS vector usage selection.

Table 2.4: Variable configuration parameters for the Base Layer.

Parameter	Value	Description
Enable8x8Transform	1	Specifies whether the 8x8 transform (High Profile) is enable. The coding efficiency is generally increased by enabling the 8x8 transform, especially for high-resolution source material.
QP	36	Quantization parameter.

Table 2.5: Static configuration parameters for the Enhancement Layers.

Parameter	Value	Description
InterLayerPred	2	Inter-layer prediction. 2- adaptative.
ILModePred	2	For enhancement layer, the best coding efficiency is usually obtained when ILModePred is set equal to 2.
ILResidualPred	2	When ILResidualPred is equal to 2, the inter-layer residual prediction is arbitrarily selected via a rate-distortion optimization framework. For enhancement layers, the best coding efficiency is usually obtained when ILResidualPred is set to 2.
MGSVectorMode	1	MGS vector usage selection.

Table 2.6: Variable configuration parameters for the Enhancement Layers.

Parameter	Value	Description
Enable8x8Transform	1	Specifies whether the 8x8 transform (High Profile) is enable. The coding efficiency is generally increased by enabling the 8x8 transform, especially for high-resolution source material.
QP	32	Quantization parameter.
MGSVector1	6	Specifies Xth position of the vector.
MGSVector2	10	The sum of the MGSVectors has to be 16.

2.3.1 Analysis of the impact of the GOP size

With the aim of comparing the impact of the GOP size in the encoded video quality, three experiments were performed. In these experiments, only the GOP size has been modified and it takes values equal to 16 or 8 bits. For the simulations, a CIF format video sequence is used (Foreman[2]) with 300 frames and 30Hz. Table 2.7

[2]The video sequences used in this thesis belong to the Xiph.org Video Test Media collection and are available in : https://media.xiph.org/video/derf/

shows the different configurations employed. All the configurations use 1 CGS layer (base layer) and 1 MGS layer (enhancement layer) which defines a different number of MGS sub-layers for each configuration. In Figure 2.3 the relationship between the GOP size and the achieved PSNR for each one of these configurations is depicted.

Although in some previous works is defined that video sequences owning a frame rate of 30 fps should employ a GOP size of 16. Simulations results presented in Figure 2.3, demonstrate that both a GOP size of 8 as well as a GOP size of 16 deliver a good PSNR. However, a GOP size equal to 8 produces a higher bit-rate with a suitable PSNR value. It is important to take into account that the selection of the GOP size depends on the application and also on the type and video complexity. In this thesis, for CIF video sequences, a GOP size equal to 8 will be used in all the cases.

Table 2.7: GOP size comparison.

Config	GOP Size	QP BL	QP EL	Rate Points
1	16	35	25(1,2,2,3,4,4)	7
	8			
2	16	32	23(1,1,1,1,1,1,1,1,1,1,3,3)	13
	8			
3	16	32	23(6,10)	3
	8			

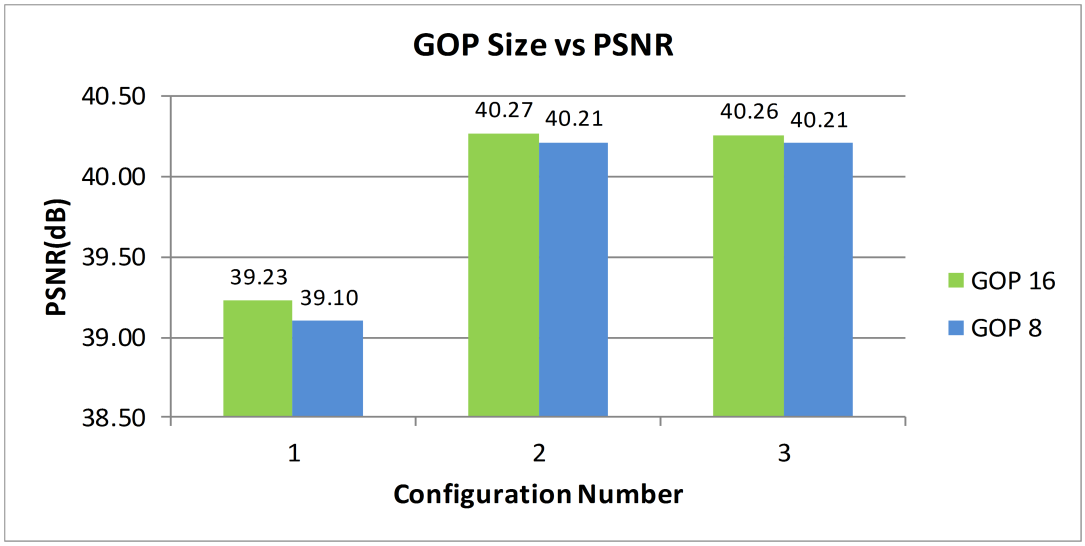


Figure 2.3: GOP Size vs. PSNR

2.3.2 Analysis of the impact of the number of CGS layers and MGS sub layers

According to [10], when a video sequence is encoded using MGS, more than 3 CGS layers should be used to improve the RD results. It is important to note that a MGS layer is a special type of CGS layer, which can be split in up 16 sub-layers and each of which defines a different bit-rate or quality. In contrast, the MGS layer

fragmentation increases the rate adaptation points, but it also leads to a negative impact due to each sub-layer added introduces a cost in terms of rate-distortion. For this reason, the number of MGS sub-layers cannot be more than five, as stated in [11]. Table 2.8 presents different configurations that were used to analyze the impact of the number of MGS layers and the MGS sub-layers, in the increase or decrease, of the rate-distortion in an encoded video sequence. In Table 2.8, all the configurations use a GOP size equal to 8. Moreover, the Base Layer must to be mandatory a CGS layer.

Table 2.8: Encoding configurations varying the number of CGS layers and MGS sub-layers.

Config	QP BL	QP EL	Rate Points
4	32	23(2, 2, 2, 2, 2, 2, 4)	8
5	36	28(4, 4, 8), 20(2, 4, 5, 5)	8
6	36	32(6, 10), 26(6, 10), 20(4, 4, 8)	8
7	35	25(16)	2
8	35	25(2, 14)	3
9	35	25(4, 12)	3
10	35	25(8, 8)	3
11	35	25(2, 2, 12)	4
12	35	25(4, 6, 6)	4
13	35	25(4, 4, 8)	4

First, we analyze the effect that the number of MGS layers produces in the final quality of the encoded video sequence. In doing so, we compare configurations 4, 5 and 6 which supply the same number or rate points employing a different number of MGS layers.

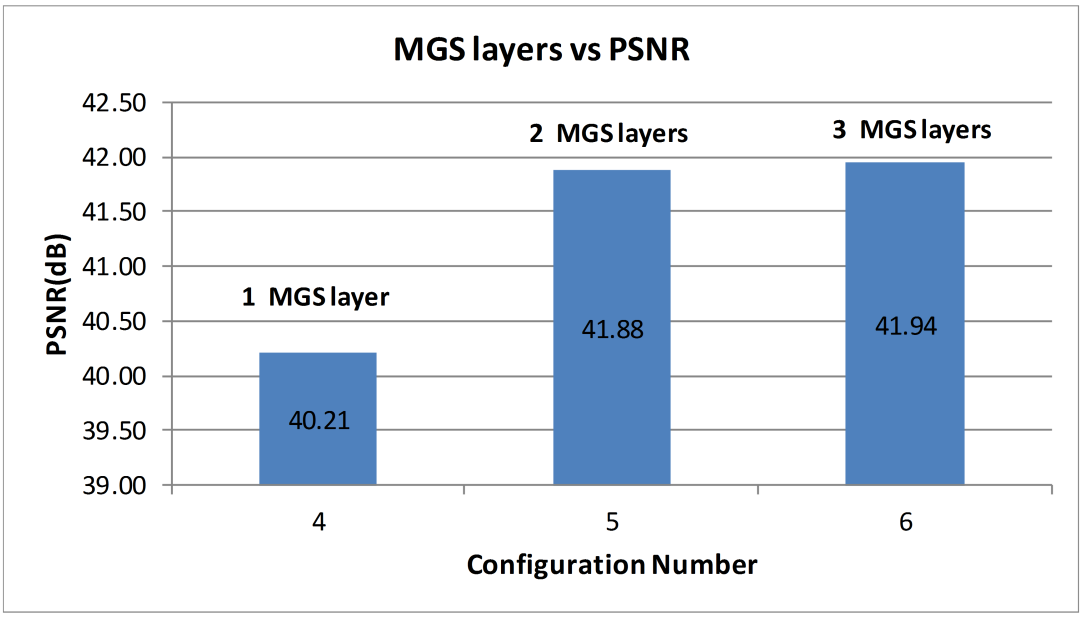


Figure 2.4: MGS layers vs. PSNR regarding the number of MGS layers.

Figure 2.4 shows that configuration 6 which defines in total 4 layers (1 base layer and 3 MGS enhancement layers) produces a greater PSNR than the other ones. Furthermore, as explained in Section 2.2.3, a MGS layer can be partitioned in several sub-layers. These sub-layers are represented by vectors (MGSVector), which correspond to the transform coefficients of a 4x4 macroblock. The sum of all values of these vectors must be equal to 16, it means that n combinations can be defined. Now, to evaluate the impact of the MGS vector values, configurations 8, 9 and 10 are compared. The obtained results are depicted in Figure 2.5. These configurations set the same number of rate points and employ identical QP parameters. Only the values assigned to the MGSVectors are different. As we can see, in Figure 2.5, the PSNR value for the three configurations is the same. Moreover, configurations 11, 12 and 13 were processed in the same way as before attaining the same outcome. Therefore, from these experiments we can conclude that the values of the MGSVectors do not affect the final PSNR attained.

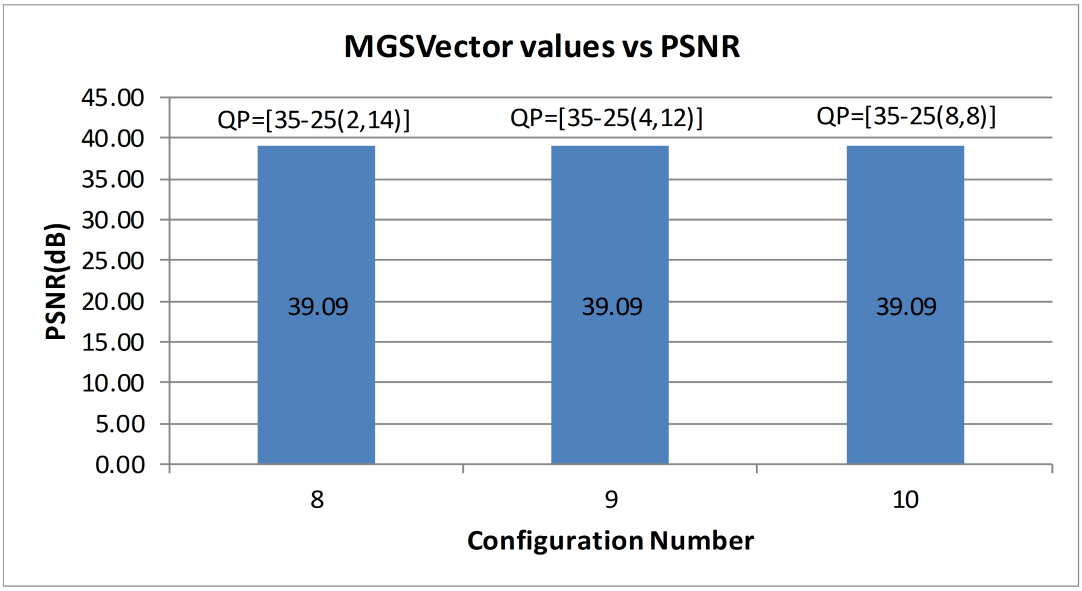


Figure 2.5: MGSVector values vs. PSNR .

A final test is carried out to decide the influence of the number of MGS sub-layers. We compare configurations 7, 8 and 9 which define one base layer and one enhancement layer (MGS layer) with 1, 2 and 3 sub-layers respectively. From the results presented in Figure 2.6, we can deduce that more than one MGSVector must be utilized to take advantage of the MGS mode, otherwise it would be the same that using a CGS layer. Furthermore, when more than one MGS sub-layers are used, the produced results are quite similar delivering a good RD efficiency.

In recap, despite of the high encoding complexity of CGS, it could be employed when four or fewer rate points are required. It is imperative to taking into account the fact that, despite at most 8 CGS layers are allowed, a large number of CGS layers may incur in significant PSNR degradation and encoding complexity. Therefore, if more rate points are needed, it is better to add MGS sub-layers to existing CGS layers, as a way of improving the distortion performance. However, the number of MGS sub-layers has to be carefully selected because, as explained above, not more than five sub-layers must be used in order of not increasing distortion. For the encoding process of the video sequences utilized along this thesis, the considerations

raised in this chapter will be considered.

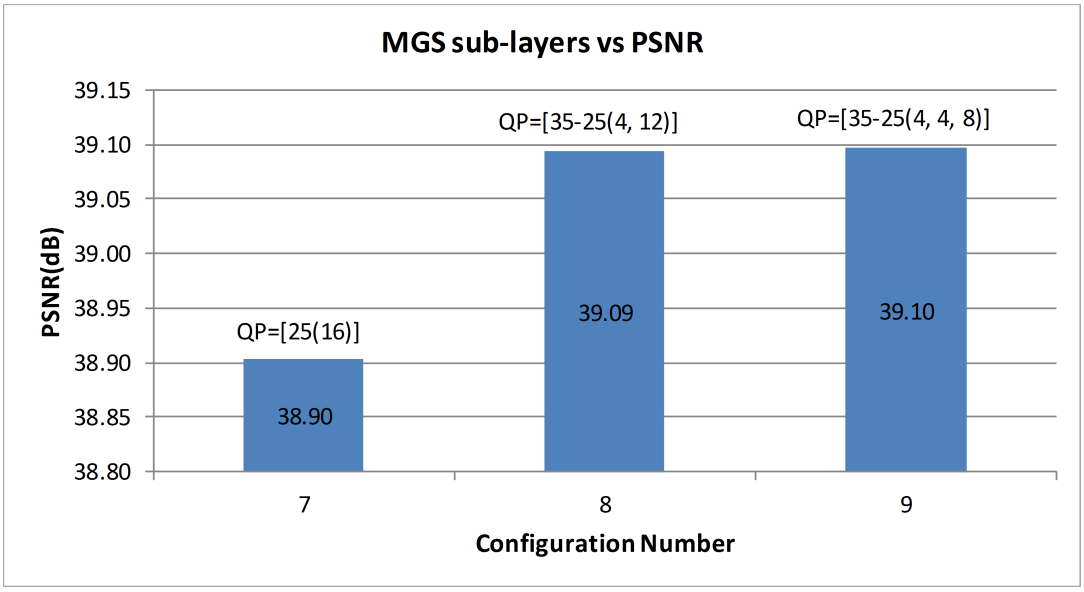


Figure 2.6: MGS sub-layers vs. PSNR (different number of MGS sub-layers).

Chapter 3

Multiple Description Coding with SVC

In a video streaming transmission over networks, there are two principal drawbacks which lead to video quality degradation. These are: bandwidth fluctuations and packet loss. To cope with this problem, suitable coding schemes and path diversity have been proposed in the literature, such in [12], [13], [12] and [14]. As seen in the previous chapter, the wide adoption and versatility of H.264 standard (H.264/AVC) has influenced to the inclusion of scalability tools. H.264 Scalable Video Coding (H.264/SVC) provides scalable video in one encoded bit-stream, thus increasing the flexibility of bit-stream adaptation. This property is effective when adaptive source/channel coding is required, or in case of variable available bandwidth. However, this coding scheme is not as robust to packet losses as those that could be experienced by the transmission over IP.

Multiple Description Coding (MDC) is one of the promising solutions for live video delivery over lossy networks. In MDC a raw video source is coded such that multiple complementary descriptions which are individually decodable are generated. Figure 3.1 illustrates a block diagram of MDC with two descriptions. When two descriptions are built, they can be transmitted separately, possibly through different network paths with different characteristics (i.e. bandwidth). At the receiver side, if only one description is available, it is decoded by the *side decoder* and the resulting quality is called *side distortion* (Dside). When both descriptions are available, they are decoded by the *central decoder* and the resulting quality is called *central distorsion* (Dcentral). In central decoder the descriptions are merged, so a video with higher quality is achieved [15].

If all the descriptions are received by the decoder, then the original video quality is generated. If, on the other hand, only one description is received, then a lower quality video is obtained. This independent decodability feature is made possible at the expense of additional coding overhead, also known as "data redundancy" between the various descriptions [16]. The cost of redundancy is accepted because of the error resiliency achieved by MDC. For channels with high packet loss rate (PLR), the probability of side decoding is greater, and thus higher side quality is of interest, and therefore more redundancy is needed. However, for cases where all descriptions are received, this redundancy must be minimized. Accordingly, the design of multiple description coders, focuses on minimizing the redundancy while it maintains an acceptable level of distortion. How to add the redundancy,

or equivalently enhance the side quality, depends exclusively on the MDC method employed.

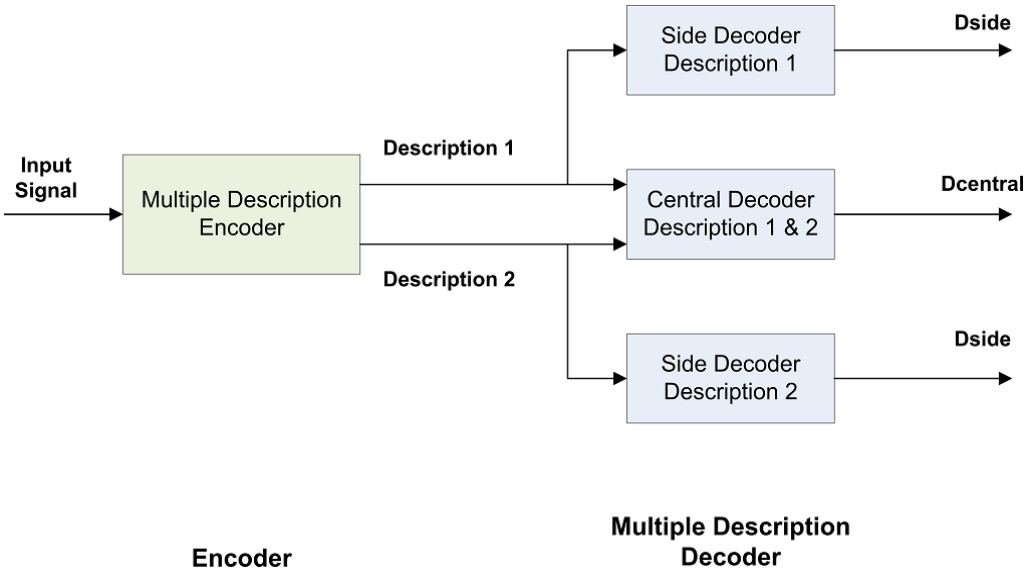


Figure 3.1: Block diagram of MDC.

3.1 Multiple Description Coding schemes

There are several MDC schemes. These schemes or methods are based on the way that descriptions are generated. The MDC process, which consists in generating two or more descriptions from splitting the information, can be carried out in several domains. These domains are spatial, temporal, frequency, and compressed, in which the descriptions are generated by partitioning pixels, frames, transformed data, and compressed data, respectively. Moreover, there exist also MDC approaches working in multiple domains, and MDC methods which are not based on split namely unpartitioning methods.

In the present work, we use two MDC methods to generate the descriptions: Spatial-domain MDC and the Temporal-domain MDC. More information regarding the remaining methods are available in [15].

Spatial-domain MDC

In this category, the MDC process is carried out in the pixel domain. The simplest approach consists in dividing the image/frame into multiple sub-images and encoding each one independently. One form to get this is breaking up a frame into two sub-images composed by even rows and odd rows. Each one of these sub-images (descriptions) owns a resolution that corresponds to half of the height of the images in the original video sequence. Despite of this alteration in the resolution, the number of total frames is preserved in both descriptions. At the decoder side, if all descriptions are received, the sub-images are merged and the image in full resolution is reconstructed. Otherwise, the missed description must be recovered using interpolation or similar techniques.

Temporal-domain MDC

In temporal domain MDC, the descriptions are generated by a process performed at the frame level. A simple case is frame distribution between descriptions: odd frames are assigned to description one and even frames compose the other description. At side decoder, the lost frames are substituted by frame freezing or estimated by concealment methods.

3.2 Comparison of the MDC methods

The two MDC approaches explained above are used to create the two descriptions. Then, each description is encoded using H.264/SVC with MGS. In such way, we generate a scalable MDC bit-stream for each description. This bit-stream contains one base layer and three MGS enhancement layers which improve the quality. Using the Peak-signal-to-noise ratio (PSNR) obtained at the receiver side, we compare the quality achieved by the two MDC schemes. Our experimental environment assumes the following constraints:

- The descriptions are transmitted through network without losses. So, at the receiver side the two descriptions are received without packet losses.
- We assume the use of a Multipath Transmission (MPT) where each description is sent over a separate path, and each path presents different bandwidth. In this case, we use two descriptions and two different paths.
- We assume that the bandwidth of the two paths fluctuates between a range of values. This range is defined by the minimum and maximum value specified in the H.264/SVC- MGS encoder configuration file. The minimum bandwidth value corresponds to the bit-rate reached by the Base Layer, and the maximum one is associated to the bit-rate of the highest Enhancement Layer.
- We consider bandwidth oscillations, which are measured each second. In such way, at each second of time a new sub-stream with a quality equivalent to the available bandwidth is generated.

The first step is to generate the descriptions. So, we split the video source using the two MDC schemes indicated above (temporal and spatial domain). Then, each Description is divided into video chunks of one second duration (30 frames).

Once video chunks have been created, we proceed with the encoding process. To encode the descriptions, H.264/Scalable Video Coding (SVC) with SNR scalability, Medium Grain Scalability (MGS)is used. The employed encoder configuration is : QP=[36, 32(6-10), 28(6-10), 24 (4-4-8)], where 4 layers and 8 rate points are defined. The encoder process is done using the JSVM (Joint Scalable Video Model) software and the *FixedQPEncoderStatic* tool. This tool offers the option of specifying the PSNR or the bit-rate that is desired to reach. In this work, we applied the bit-rate option. Then, with the purpose of finding the suitable bit-rates values for encoding the different video sequences, the Kush Gauge [17] formula (3.1) is utilized.

$$\begin{aligned} Bitrate(kbps) = Width \times Height \times Frame\_rate \times \\ motion\_factor \times 0.07 \div 1000 \end{aligned} \tag{3.1}$$

Equation 3.1 considers the factors that lead to needing higher bit rates to achieve a given level of quality, such as, number of pixels in each frame, number of frames per second and the amount of motion in the image (low/mid/high). The two first parameters are known and the last one (also called "motion rank") is defined considering the features of the video. The motion factor can take three values: Low = 1 , Medium = 2 and High = 4. To assign these ranks, we regard the definition of these three categories of videos, raised in [17].

- **Low motion** is a video that has minimal movement. For instance, a person talking in front of a camera without moving much, while the camera itself and the background is not moving at all.

- **Medium motion** would be some degree of movement, but in a more predictable and orderly manner, which means some relatively slow camera and subject movements. But not many scene changes or cuts or sudden snap camera movements, or zooms where the entire picture changes into something completely different instantaneously.

- **High motion** would be something like the most challenging action movie trailer, where not only the movements are fast and unpredictable but the scenes also change very rapidly.

According to the Kush Gauge approach, in case of constant bit-rate (CBR), a value close to the estimated one can be used. On the other hand, in case of variable bit rate (VBR), as in our case, a value that is about 75% of the estimate can be used as a target, and a value around 150% of it can be used as the maximum rate. It is clear that these values could vary depending on the nature of the content and type of application required.

Furthermore, with the purpose of simulating bandwidth variations in the network paths, we use the bit-rates attained by the encoded bit-stream to generated a text file. This text file contains many values (bit-rates) as the duration of the video in seconds (i.e. For a 10 seconds video sequence, a text file with 10 values must be desired). These bit-rates are randomly created, respecting the specific range of admitted values which is defined by the minimum and maximum bit-rates achieved by the encoded video. At each second of time, a new bit-rate is sequentially acquired from the text file. Then, employing the current bit-rate, a sub-stream is produced. Moreover, to extract the sub-stream which matches with the current bandwidth, we use the *BitStreamExtractorStatic* tool included into the JSVM software.

It is important to take into account that before to extract a sub-stream, it is necessary to embed the information about the quality layers because this improves the rate-distortion efficiency of the extracted sub-streams. Using the *QualityLevelAssignerStatic* tool, it is possible to add this information. By default, the *QualityLevelAssignerStatic* assigns Quality layer Ids in such a way that, the Quality layer Ids associated with a lower layer(dependency_id) are in a higher range compared to the Quality layer Ids associated with a higher layer. This ensures that the extractor removes the MGS packets associated with a higher layer before removing MGS packets of a lower one.

At the receiver side, we consider three possible situations that may occur: 1) only the first description is received, 2) only the second description is received or 3) Both,

Description 1 and Description 2 are received. When the two first aforementioned cases take place, the missing Description should be replaced in somehow. We achieve this with the Interpolation process.

In the Spatial Domain (MDC-SD), the missing description is recovered by the interpolation between the pixels of two consecutive lines belonging to one frame. For example, if only Description 1 is received (description composed by the odd lines of frames) the missed lines will be replaced by the interpolation of line one with line three, line three with line five and so on, for all the frames that compose the video sequence. On the other hand, in the Temporal Domain (MDC-TD), the missing Description is attained replaying the previous frame. For instance, if just Description 2 is received (description composed by the even frames of video sequence) the missing frames will be replaced by the corresponding previous frame.

Finally, the received description and the complementary file, generated by interpolation, are merged to have a video sequence with the same features than the original one. It means that the resultant video will have the same width, height, number of frames and frame rate. When the two descriptions (Description 1 and Description 2) are received, these can be directly merged. Then, the PSNR is calculated using the *PSNRStatic* tool. Some experiments and the obtained results are analyzed in the sub-sections below.

3.2.1 Experiments with CIF video sequences

Several CIF (352 x 288) video sequences with different characteristics (fast/complex sequences and slow/simple sequences) were encoded using MDC and H.264 SVC/MGS. The frame rate of these sequences is equal to 30 frames per second. Table 3.1 presents the encoded bit-rates employed, to encode each description. It is essential to mention that when both description are received at the client side, the total bit-rate achieved is equal to the sum of the bit-rates of the two descriptions separately. Moreover, the two first video sequences correspond to complex sequences, while the last one is a simple one.

Table 3.1: CIF videos - Encoding bit rates.

Video seq	Layer	Bit-rate (Kbit/s)
Football Calendar	BL	100
	EL1	200
	EL2	300
	EL3	500
Bridge-Close	BL	100
	EL1	200
	EL2	300
	EL3	350

1) Football

Table 3.2 presents the data corresponding to the bit-rates and the PSNRs achieved when MDC-TD is used. These bit-rates represent the available bandwidth at a specific instant of time. For instance, at second one the current bandwidth is equal

to 319.71 kbit/s. Then, using this bit-rate we proceed to extract the sub-stream at this target rate, do the interpolation process and calculate the PSNR. The aforementioned mechanism will be practiced in all the experiments presented in this section. Additionally, Figure 3.2 shows in a graphical way the obtained results. The total number of frames in this sequence is 260 frames.

Table 3.2: Bit-rates and PSNR of Football MDC-TD.

Football MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	319.71	22.3379	422.81	22.6828	26.7703
2	313.31	22.6994	255.29	22.4668	27.2924
3	407.61	26.5892	214.26	25.9128	31.8737
4	251.85	28.3109	329.27	28.4833	34.287
5	370.94	26.0179	454.35	26.2028	31.7027
6	250.81	23.5613	405.51	24.2995	29.4201
7	277.65	25.3432	254.05	25.1617	30.0776
8	279.26	22.1651	367.44	22.4958	26.5666

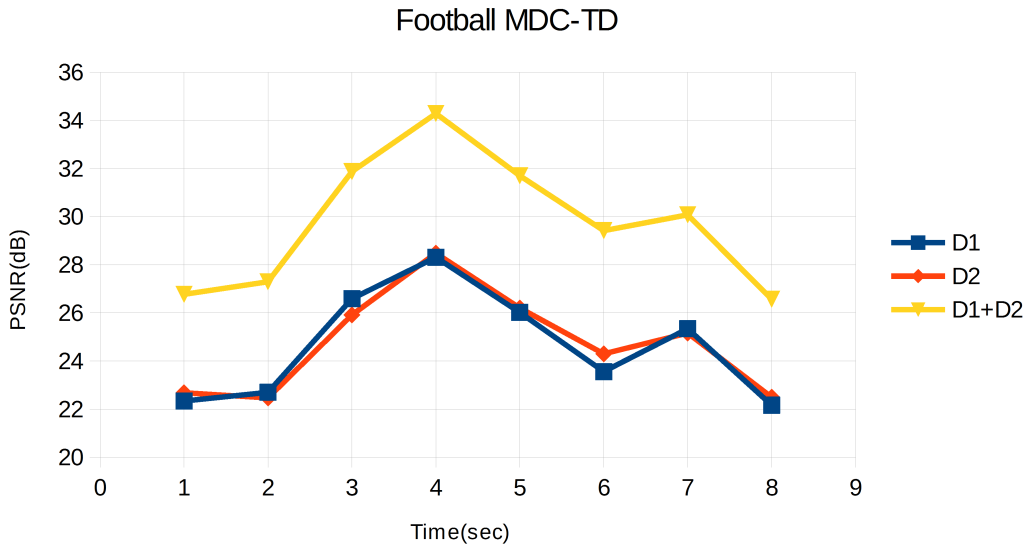


Figure 3.2: PSNR Football (MDC-TD).

As we can see in Figure 3.2, if just one description is received, the PSNR is quite similar for the two descriptions, during the entire video presentation. The minimum PSNR value is 22.03 dB and the maximum one is 28.48 dB. However, if both descriptions are received, the PSNR is increased at least by 4 dB and 6 dB in the best case (34.28 dB). On the other hand, Table 3.3 presents the PSNR obtained when MDC-SD is used, and in Figure 3.3 this data is represented graphically.

As in the previous case, when MDC-SD is applied and only one description is received, the achieved PSNR is very similar. Nonetheless, if the two descriptions

Table 3.3: Bit-rates and PSNR of Football MDC-SD.

Football MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	319.71	23.5245	422.81	24.0215	25.272
2	313.31	24.8724	255.29	24.6035	25.8088
3	407.61	29.0052	214.26	28.2859	30.5287
4	251.85	30.6266	329.27	30.9204	33.3702
5	370.94	27.5334	454.35	27.7135	30.0819
6	250.81	25.682	405.51	26.5364	27.6824
7	277.65	27.1656	254.05	27.0297	28.7558
8	279.26	23.5311	367.44	23.8895	25.172

are received the PSNR is increased at maximum by around 3 dB and minimum by 1 dB. The two MDC techniques are compared in Figure 3.4. It is seen that, if just one description is received, a best PSNR is obtained using MDC-SD. However, when both descriptions are received, MDC-TD overcomes the other approach. The PSNR obtained by MDC-TD is increased at minimum by 1 dB and maximum by 1.6 dB. Then, regarding the resolution of the video sequence (CIF) and the fact that this video is fast, it is likely the users do not perceive this little increase in quality.

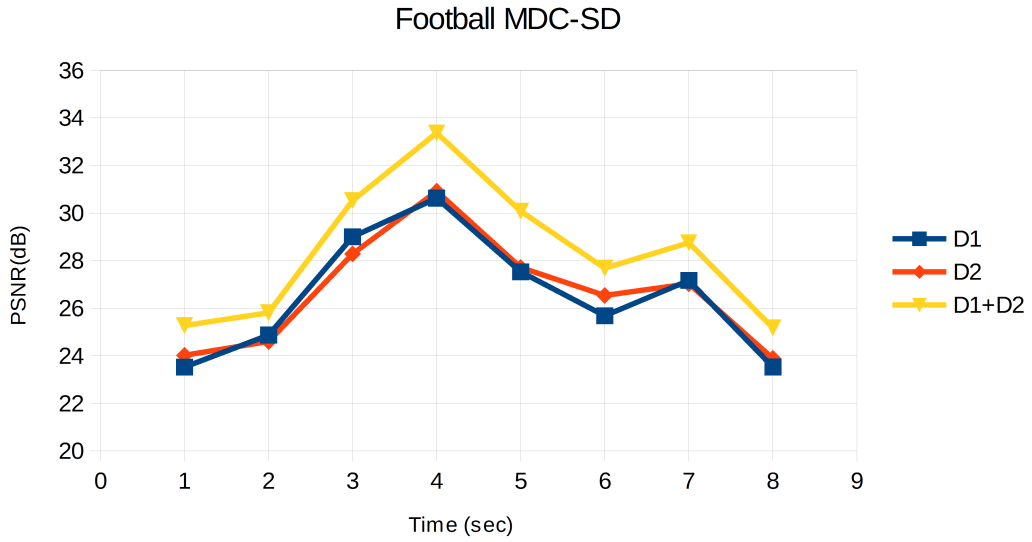


Figure 3.3: PSNR Football (MDC-SD).

Furthermore, the quality perceived on the Football video sequence, when the two MDC methods are used, is evaluated below. Figure 3.5 illustrates some pictures corresponding to each description separately and both descriptions jointly, using MDC-TD. Hence, Figure 3.5 (c) shows evidently that quality is improved when both descriptions are received. The image is more clear and also there does not exist the pixelated effect presented in Figure 3.5 (a) and 3.5 (b). It is important to highlight that when the MDC-TD approach is applied and just one description is

received, the frames of the lost description are replaced by the previous frames. In that way, the previous frame is copied in the position of the missing frame.

The previous observation can be noticed in the orange and purple circles drawn in Figure 3.5. Here, the same frame (frame corresponding to 04" 80 seconds) is captured to all the three cases, and then the differences between them are enclosed in colored circles. These differences are due to the Football video sequence is a fast and complex video sequence, where the vectors of movement change quickly. When MDC-TD is used, the first description is composed by the odd frames and the second one is composed by the even frames. Consequently, if the video contains a lot of movement, it is very likely the missing frame owns vectors of movement which are different to the vectors in the previous frame. Therefore, when the reconstructed video sequence is displayed, it is possible that the users do not perceive these details due to the video's speed. However, if it is analyzed frame by frame, these differences will be detected.

Referring to Figure 3.6, the quality perceived in Figure 3.6 (a) and (b) is relatively comparable because the PSNR in these two descriptions is almost equal, varying only thousandths. A significant difference is observed when both descriptions are received, Figure 3.6 (c). The retrieved image is clearer and less pixelated than the previous ones. These contrasts can be perceived in several parts of the image, which are enclosed by orange circles.

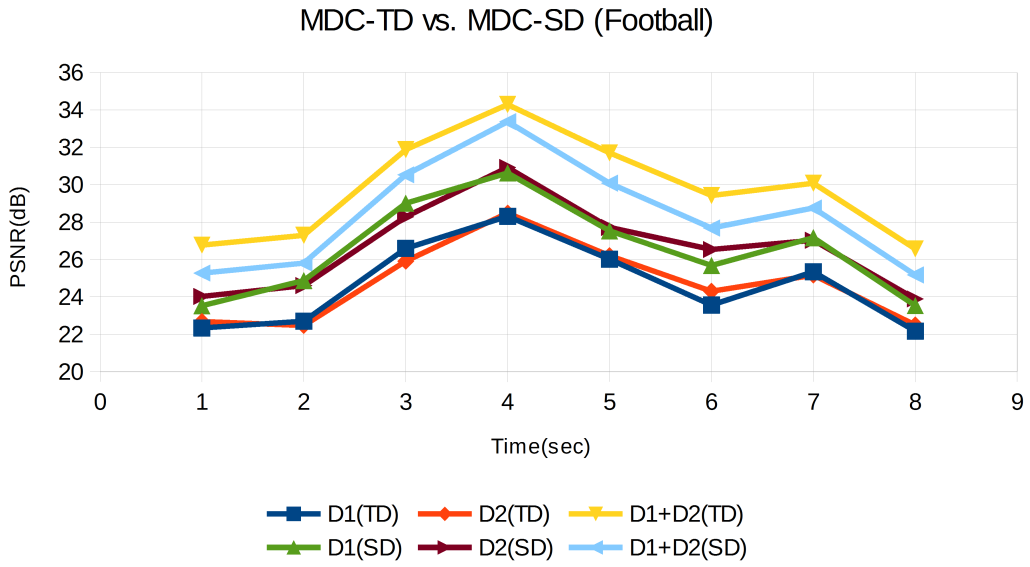


Figure 3.4: MDC-SD vs. MDC-TD (Football).

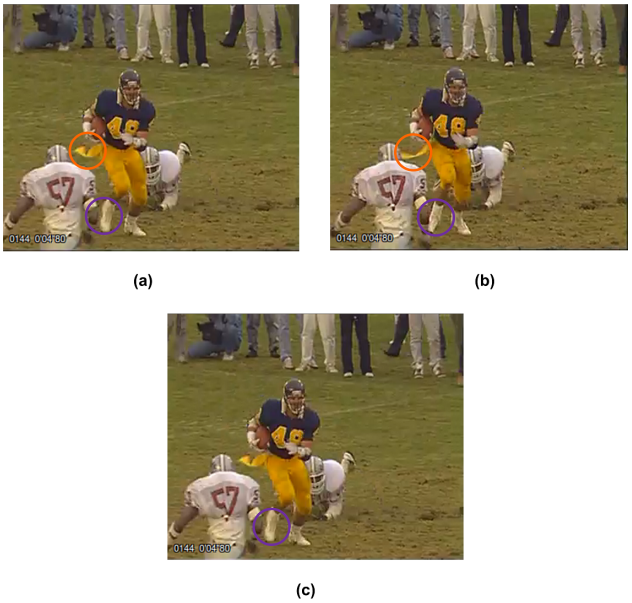


Figure 3.5: Visual Comparison of Football (MDC-TD).
(a) Description 1 (b) Description 2 (c) Two descriptions.

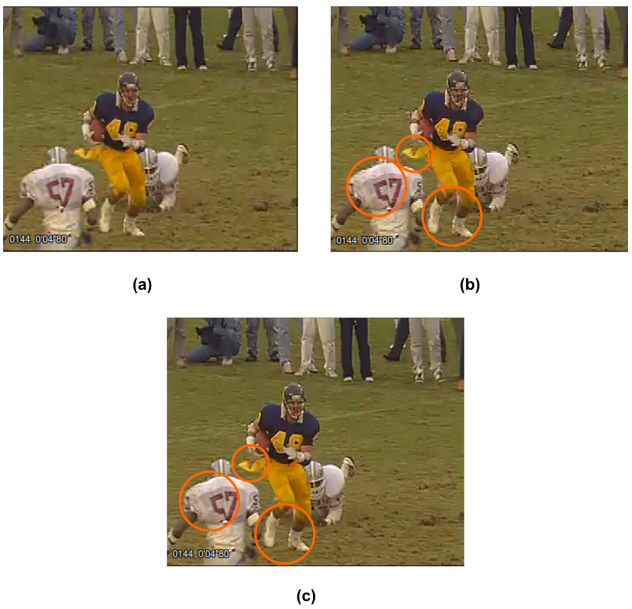


Figure 3.6: Visual Comparison of Football (MDC-SD).
(a) Description 1 (b) Description 2 (c) Two descriptions.

2) Calendar

The same procedure is performed with the Calendar video sequence, which is composed by 300 frames (10 seconds video duration). Table 3.4 presents the bit-rates and the PSNR attained using MDC-TD. Then, Figure 3.7 shows graphically the obtained results.

Figure 3.7 shows that when just one of the two descriptions is received, the PSNR values fluctuate between 21.5 dB to 23.7 dB. However, the PSNR behavior

Table 3.4: Bit-rates and PSNR of Calendar MDC-TD.

Calendar MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	319.71	23.1609	422.81	23.7284	26.7563
2	313.31	23.3534	255.29	23.1982	25.8148
3	407.61	23.6195	214.26	22.2586	25.4321
4	251.85	22.5766	329.27	22.9913	25.0787
5	370.94	22.4432	454.35	22.7007	26.4223
6	250.81	22.2243	405.51	23.453	25.587
7	277.65	22.971	254.05	22.4163	24.7883
8	279.26	22.8817	367.44	23.5973	25.5044
9	403.21	24.0366	212.84	22.7627	25.6729
10	212.22	22.05	481.38	23.682	26.261

is different for Description 1 and Description 2. On the other hand, when the two descriptions are received the PSNR increases considerably, at least by 2 dB.

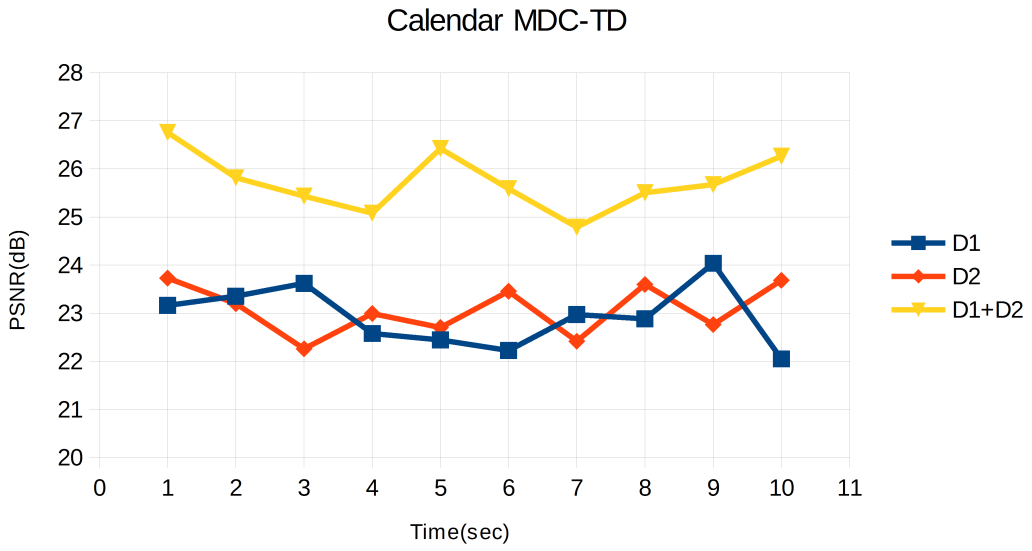


Figure 3.7: PSNR Calendar (MDC-TD).

Table 3.5 presents the data corresponding to the PSNR obtained when MDC-SD is used. Then, Figure 3.8 illustrates these results.

Opposite to what occurs with MDC-TD, when MDC-SD is employed and only one description is received, the PSNR is quite similar in the two descriptions. In this case, the difference in receiving Description 1 or Description 2 is reduced. Moreover, when both descriptions are received the PSNR is increased notably, at least by 2 dB.

Furthermore, the two MDC techniques applied in the current video sequence are compared in Figure 3.9.

As is shown in Figure 3.9, when just one description is received, the best PSNR

Table 3.5: Bit-rates and PSNR of Calendar MDC-SD.

Calendar MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	319.71	20.3402	422.81	20.5392	23.2118
2	313.31	20.4277	255.29	20.2248	22.742
3	407.61	20.3155	214.26	19.7517	22.2338
4	251.85	19.9614	329.27	20.1117	22.1335
5	370.94	20.1972	454.35	20.3794	23.2019
6	250.81	19.8085	405.51	20.2163	23.2161
7	277.65	19.9733	254.05	19.876	22.9059
8	279.26	19.8835	367.44	20.0353	22.9178
9	403.21	20.2111	212.84	19.9134	23.4324
10	212.22	19.856	481.38	20.4968	23.634

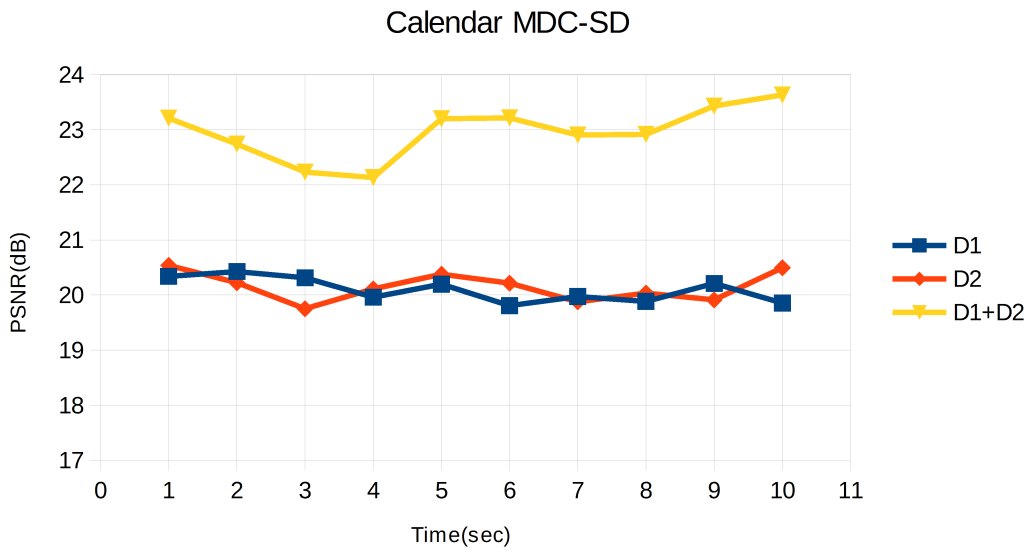


Figure 3.8: PSNR Calendar (MDC-SD).

is obtained by MDC-TD. The average PSNR achieved with MDC-TD is almost 3 dB greater than the PSNR obtained with the other approach. A similar situation occurs also when both descriptions are received. The PSNR increases in a range of 1.6 db to 3.5 dB in comparison with MDC-SD.

A visual comparison of the MDC approaches is depicted in Figure 3.10 and 3.11. Figure 3.10 (b) which represents the fact that just Description 2 is received using MDC-SD, presents a slight improvement in relation to Description 1 (Figure 3.10 (a)). The image is more clear and less blurry. Moreover, some differences between Figure 3.10 (a) and 3.10 (b) are identified and enclosed by orange circles. On the other hand, the quality perceived in Figure 3.10 (c), that corresponds to the case in which both descriptions are received, is enhanced.

In contrast to the previous case, the quality perceived when MDC-TD is used, is

quite similar in the three cases (Figure 3.11 (a) , 3.11 (b) and 3.11 (c)). In Figure 3.11 (c) the quality is increased, but it is not perceived as a considerable improvement. Nonetheless, it is evident that the quality is improved applying MDC-TD.

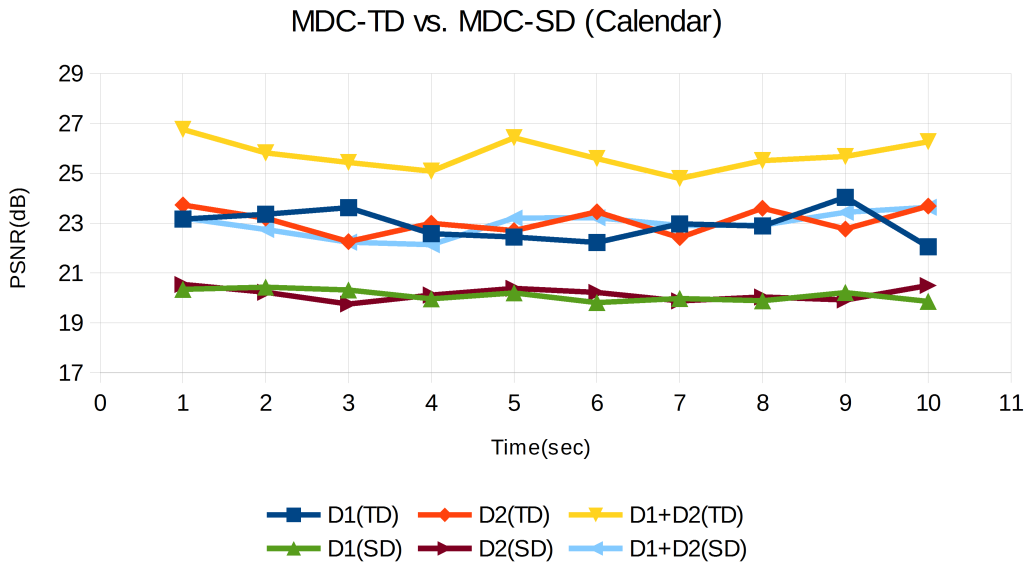


Figure 3.9: MDC-SD vs. MDC-TD (Calendar).

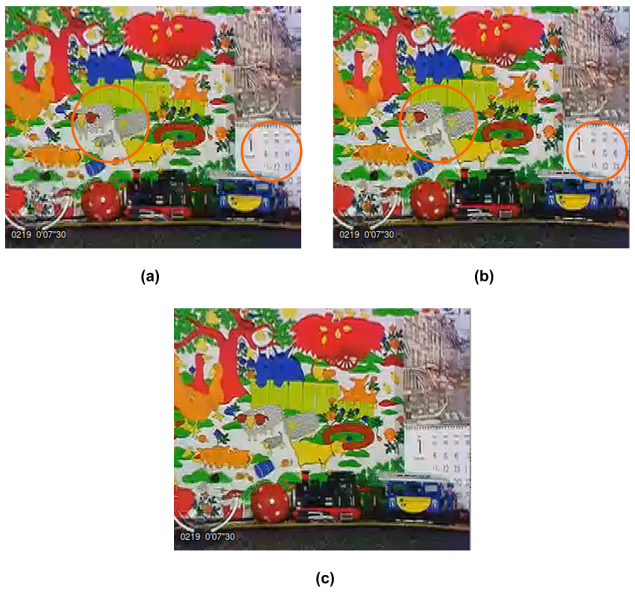


Figure 3.10: Visual Comparison of Calendar (MDC-SD).
(a) Description 1 (b) Description 2 (c) Two descriptions .

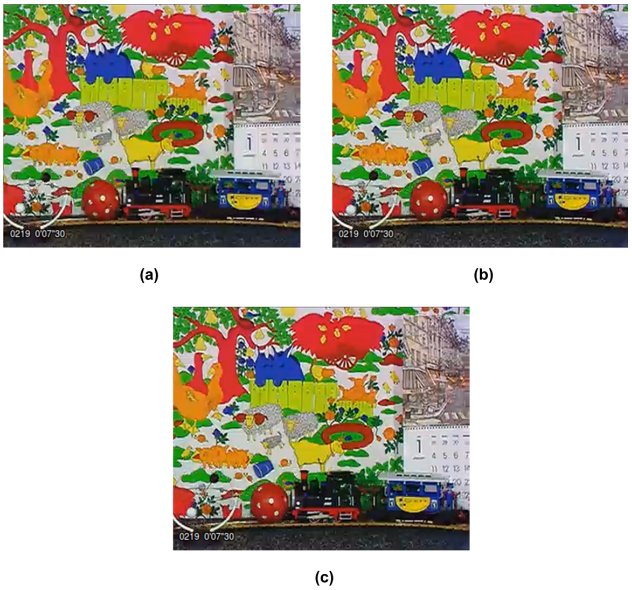


Figure 3.11: Visual Comparison of Calendar (MDC-TD).
(a) Description 1 (b) Description 2 (c) Two descriptions .

3) Bridge-Close

This video sequence has in total 2000 frames, but just the first 20 seconds of the video sequence are analyzed, it corresponds to 600 frames. Following the same procedure, Table 3.6 presents the bit-rates and PSNR attained when MDC-TD is applied and Figure 3.12 depicts graphically these results.

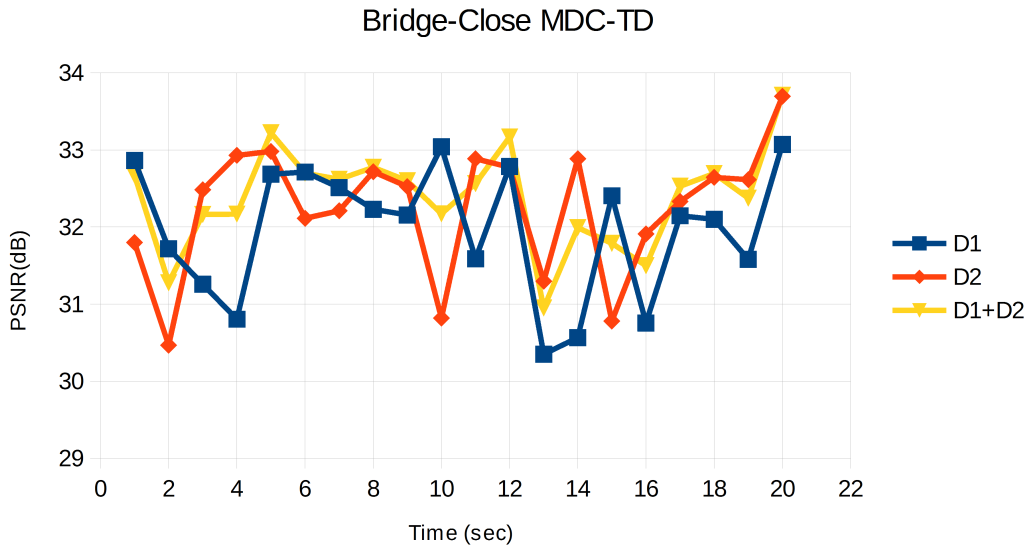


Figure 3.12: PSNR Bridge-Close (MDC-TD).

As shown in Figure 3.12, The PSNR oscillates between 30 and 34 dB. It is because the available bandwidth at each second varies for each description, then, the PSNR increases or decreases depending on this factor. When the two descriptions are received, there is not a significant improvement on the PSNR. In the best case the PSNR is increased by 1 dB.

Table 3.6: Bit-rates and PSNR of Brigde-Close MDC-TD.

Bridge-Close MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	320.35	32.8638	205.34	31.7984	32.6716
2	225	31.7186	120.4	30.4669	31.2762
3	150.51	31.2572	265.03	32.4837	32.1627
4	118.87	30.8034	309.76	32.9283	32.1671
5	261.33	32.6839	302.91	32.9806	33.2226
6	268.51	32.7131	207.56	32.1132	32.6977
7	242.29	32.5118	203.06	32.2104	32.6191
8	208.55	32.2296	261	32.7167	32.7729
9	238.69	32.1564	241.86	32.5246	32.5983
10	293.69	33.0394	143.88	30.8175	32.1709
11	167.59	31.5875	298.44	32.8851	32.5637
12	292.72	32.7843	295.67	32.7746	33.1662
13	111.25	30.3503	159.5	31.2966	30.9492
14	127.92	30.5656	306.85	32.8844	31.9912
15	246.7	32.4047	126.62	30.78	31.7879
16	121.12	30.7545	217.05	31.9109	31.499
17	231.95	32.1462	246.12	32.3307	32.529
18	237.45	32.0996	282.47	32.643	32.6936
19	161.14	31.5797	256.32	32.6142	32.3745
20	242.23	33.0714	322.48	33.6942	33.7071

It is important to highlight that in all the three cases (receiving only Description 1, Description 2 or both descriptions) the PSNR is good. Considering that typical values for the PSNR in lossy images and video compression are between 30 and 50 dB, where higher is better. In Bridge-close MDC-TD, the obtained PSNR in all the situations is higher than 30 dB, which is inside the aforementioned range.

Moreover, Table 3.7 presents the PSNR when the MDC-SD approach is applied in the current video sequence and Figure 3.13 depicts this information graphically.

Figure 3.13 shows a great increase in the PSNR value when both descriptions are received at the receiver side. Hence, the PSNR is at least 7 dB higher than the one achieved with only one description. On the other hand, the PSNR of Description 1 and Description 2 is quite similar. In Figure 3.13 is possible to appreciate that the PSNR of the two descriptions are practically overlapped. Moreover, the PSNR of these descriptions fluctuates between 23.50 dB and 24.20 dB.

With the aim of analyzing which of the two MDC approaches provides a better PSNR, in the figure below these approaches are presented simultaneously.

As we can see in Figure 3.14, if only one description is received, MDC-TD presents a better PSNR (in average 8 dB more than MDC-SD). Moreover, if just one description is received and the MDC-SD approach is used, the average PSNR value obtained is under 30 dB, which corresponds to bad quality. On the other hand, both approaches achieve similar PSNR if the two descriptions are received.

Table 3.7: Bit-rates and PSNR of Brigde-Close MDC-SD.

Bridge-Close MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	320.35	23.627	205.34	23.6203	32.0135
2	225	23.6138	120.4	23.5049	30.331
3	150.51	23.7791	265.03	23.8804	31.3736
4	118.87	23.8066	309.76	23.9203	31.2577
5	261.33	23.942	302.91	23.9309	32.5471
6	268.51	23.981	207.56	23.8994	32.0642
7	242.29	23.9237	203.06	23.909	31.8092
8	208.55	23.9417	261	23.9726	32.2384
9	238.69	23.9353	241.86	23.9521	32.0524
10	293.69	23.9806	143.88	23.8465	31.2369
11	167.59	23.8716	298.44	23.8925	31.7598
12	292.72	23.761	295.67	23.7801	32.6383
13	111.25	23.6653	159.5	23.7465	30.3335
14	127.92	23.659	306.85	23.7742	31.1336
15	246.7	23.9443	126.62	23.7969	30.9244
16	121.12	23.7278	217.05	23.7953	30.725
17	231.95	23.74	246.12	23.7885	31.7387
18	237.45	23.7631	282.47	23.8017	31.9038
19	161.14	23.9155	256.32	23.9272	31.5738
20	242.23	24.187	322.48	24.2096	33.0069

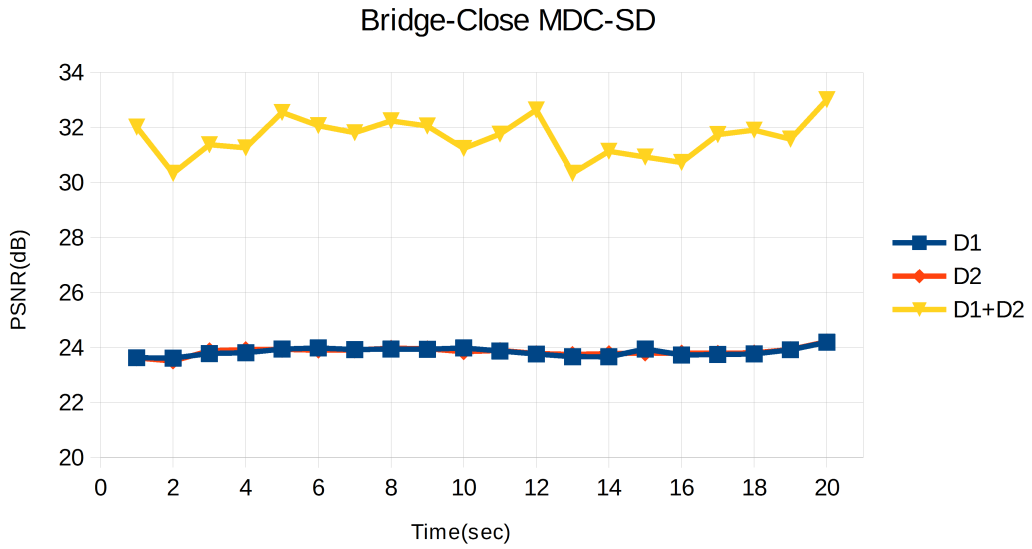


Figure 3.13: PSNR Bridge-Close (MDC-SD).

To evaluate the QoE related with the quality of the video perceived by the user, the video sequence Bridge-Close is played and analyzed. Figure 3.15 and 3.16

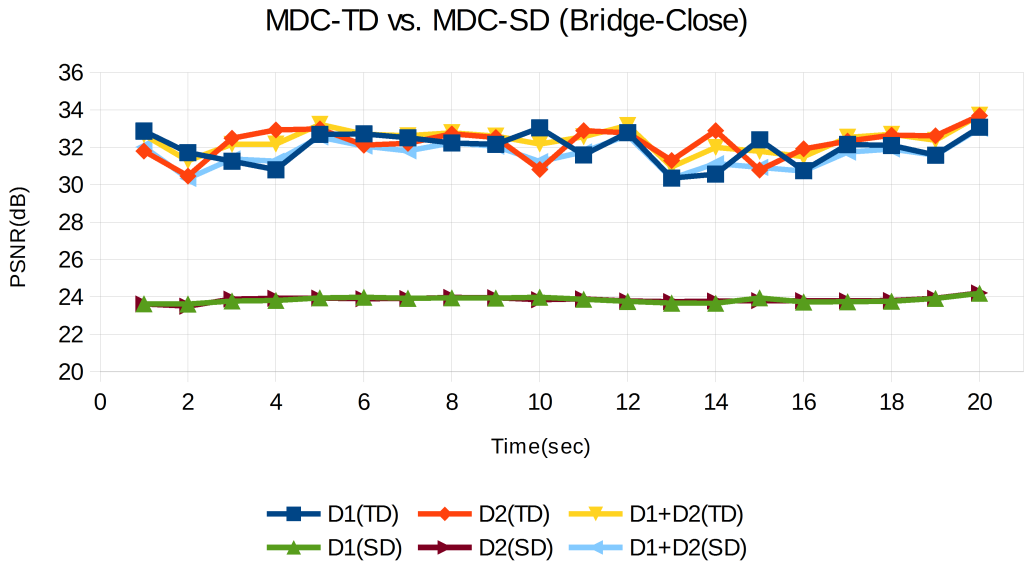


Figure 3.14: MDC-SD vs. MDC-TD (Bridge-Close).

illustrate a visual comparison of the three possible cases that could occur when the current video sequence is received at the receiver. The Figure 3.15 does not show a noticeable difference between the three cases, because the PSNR in all of them is over 30 dB. Conversely, in Figure 3.16 can be observed that the quality is greatly increased when the two descriptions are received. This is because the average PSNR when just one of the two descriptions is received is equal to 23.8 dB, in comparison with 31.6 dB achieved when both descriptions are received. Moreover, the parts of the image where this increase of quality is perceived are enclosed by an ellipse.

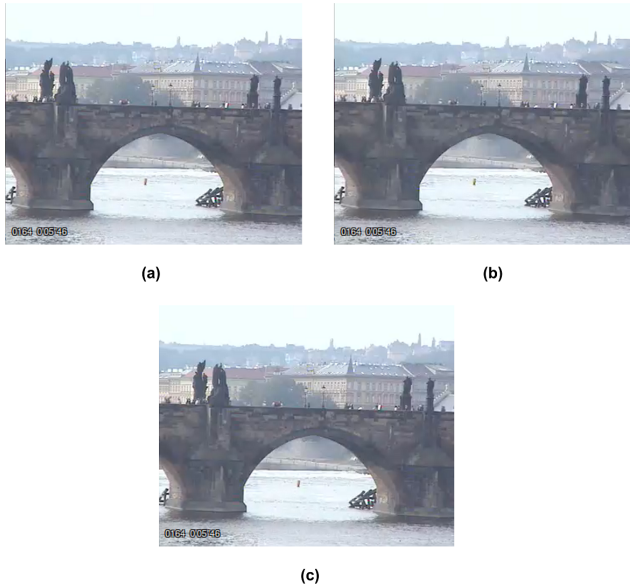


Figure 3.15: Visual Comparison of Bridge-Close (MDC-TD).
(a) Description 1 (b) Description 2 (c) Two descriptions .

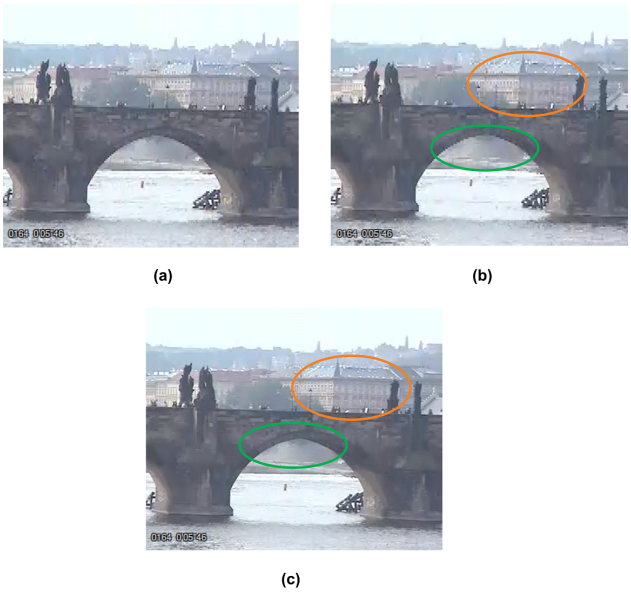


Figure 3.16: Visual Comparison of Bridge-Close (MDC-SD).
(a) Description 1 (b) Description 2 (c) Two descriptions .

Considering the obtained results in the three CIF video sequences analyzed in this sub-section, we conclude that before to apply either of the two MDC methods, the features of the video must be evaluated. In the case when both descriptions are received, the MDC-TD approach produces a higher PSNR and consequently a better quality. On the opposite, when just one of the two descriptions is received, we could get different performances. If we use complex video sequences with a great amount of movement, like Football video sequence, MDC-SD attains a better PSNR than MDC-TD. Moreover, because of this, the possibility of the vectors of movement vary drastically from one frame to the next one is high. In this case, MDC-SD preserves the frame with the same characteristics as the original one. On the other hand, when video sequences with a low or medium amount of movement are employed, the best PSNR is achieved with MDC-TD. Must be remarked that this happen in any case (receiving one or both descriptions).

3.2.2 Experiments with HD video sequences

With the aim of evaluating MDC in HD video sequences, the two well known approaches: MDC-SD (Spatial Domain) and MDC-TD (Temporal Domain) are used to encode video sequences which present a 16:9 aspect ratio. Several HD videos are employed to carry out this experiment. However, since similar behavior was found in all the analyzed video sequences, only three of them will be exposed below. The encoded bit-rates of the descriptions and some characteristics about the used videos are shown in Table 3.8.

1) Stockholm

In Stockholm video sequence, the camera pans horizontally and exhibits the city. This sequence has 360 frames and each description is encoded at a maximum bit-rate equal to 3250 kbit/s. In such way, when both descriptions are received, the total bit-rate, in the best case, will be 6500 kbit/s. Table 3.9 presents the PSNR values attained by MDC-TD. In Figure 3.17 these results are illustrated.

Table 3.8: HD videos - Encoding bit rates.

Video seq	Resolution	Frame-rate	Layer	Bit-rate (Kbit/s)
Stockholm Shields	720p	50	BL	1000
			EL1	1500
			EL2	2500
			EL3	3250
Ducks ParkJoy	1080p	50	BL	10000
			EL1	12000
			EL2	14000
			EL3	14500
Tennis	1080p	24	BL	4000
			EL1	5000
			EL2	6000
			EL3	7000

Table 3.9: Bit-rates and PSNR of Stockholm MDC-TD.

Stockholm MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	2632.17	28.8228	1602.9	28.4238	33.2686
2	2469.63	27.6015	2002.13	27.1404	32.1359
3	3183.91	27.7599	1262.75	26.8234	31.9924
4	2554.53	27.6281	3173.11	27.7385	32.8628
5	1422.43	26.8235	2971.09	27.7265	31.9774
6	1964.9	26.8142	2255.83	26.8669	30.9997
7	1641.18	26.8476	2117.52	26.9145	30.855
8	2341.05	27.621	1344.54	26.767	30.8962
9	2246.74	28.0575	1153.82	27.1099	30.7943
10	1707.44	27.4724	1367.6	27.2605	29.9647
11	2677.07	28.4063	1135.77	27.4329	30.5617
12	1926.18	30.3919	1430	30.0463	30.3921

As shown in Figure 3.17, when only one description is received the average PSNR obtained is quite similar and equal to 27.5 dB. On the other hand, when both descriptions are received, the PSNR increases to 31.40 dB. A peculiar performance is presented at second 12, where the PSNR achieved by Description 1, Description 2 and the both descriptions together is quite similar. The reason for this nature is that second 12 corresponds to the end of the video sequence and, at this time, the panoramic video capture is finished. Consequently, the pictures belonging to second 12 present a low amount of movement because the background is static and there are only few cars in movement.

Table 3.10 and Figure 3.18 present the PSNR obtained when the MDC-SD approach is used in the current video sequence. From this graph, it is seen that if only one description is received, the attained PSNR is identical and equal to 27.5 dB,

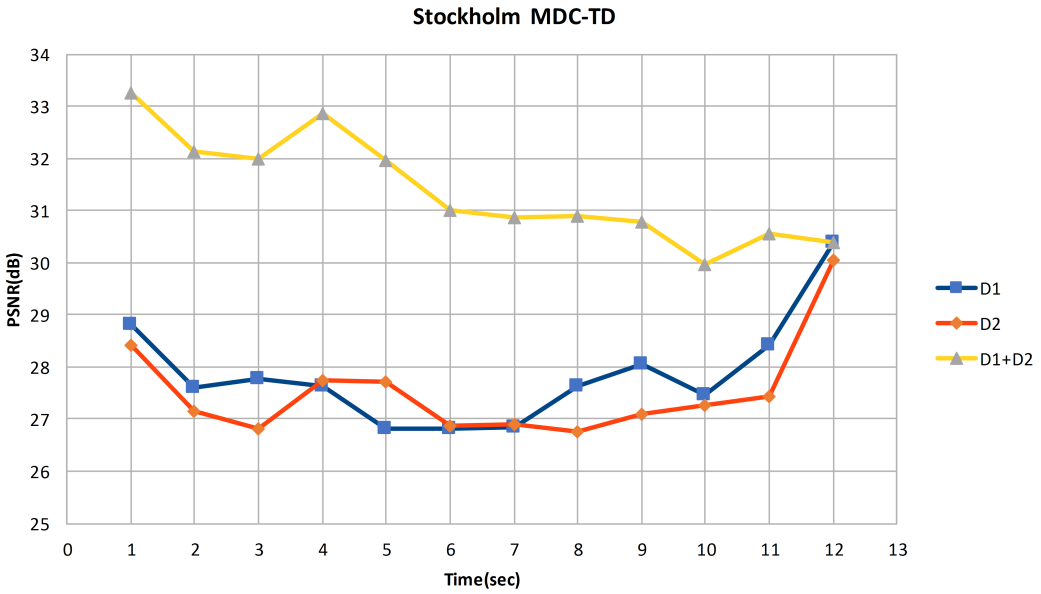


Figure 3.17: PSNR Stockholm (MDC-TD).

on average. Whether both descriptions are received, the average PSNR increases to 29.7 dB. Unlike that what occurs at second 12 when MDC-TD is applied, with MDC-SD if both descriptions are received the PSNR is increased by 2 dB. This is because the interpolation process to reconstruct the video at the original resolution is more complex. Hence, when only one description is arrived, it is not able to improve the quality as much as when both descriptions are received.

Table 3.10: Bit-rates and PSNR of Stockholm MDC-SD.

Stockholm MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	2632.17	29.2786	1602.9	28.7639	31.3228
2	2469.63	28.5521	2002.13	27.9476	30.4613
3	3183.91	28.5589	1262.75	27.4552	30.0189
4	2554.53	28.2904	3173.11	28.2069	31.1453
5	1422.43	27.5389	2971.09	28.0607	30.0217
6	1964.9	27.4976	2255.83	27.8491	29.8449
7	1641.18	27.3349	2117.52	27.5591	29.5624
8	2341.05	27.6528	1344.54	26.7971	29.0535
9	2246.74	27.5821	1153.82	26.7502	29.0036
10	1707.44	27.1311	1367.6	26.6582	28.5535
11	2677.07	27.4485	1135.77	26.6186	28.9553
12	1926.18	26.6283	1430	26.8828	28.5505

Finally, Figure 3.19 shows a comparison between the two MDC approaches applied in the Stockholm video sequence. When just one description is received, there is not a significant difference between the PSNR obtained using the different ap-

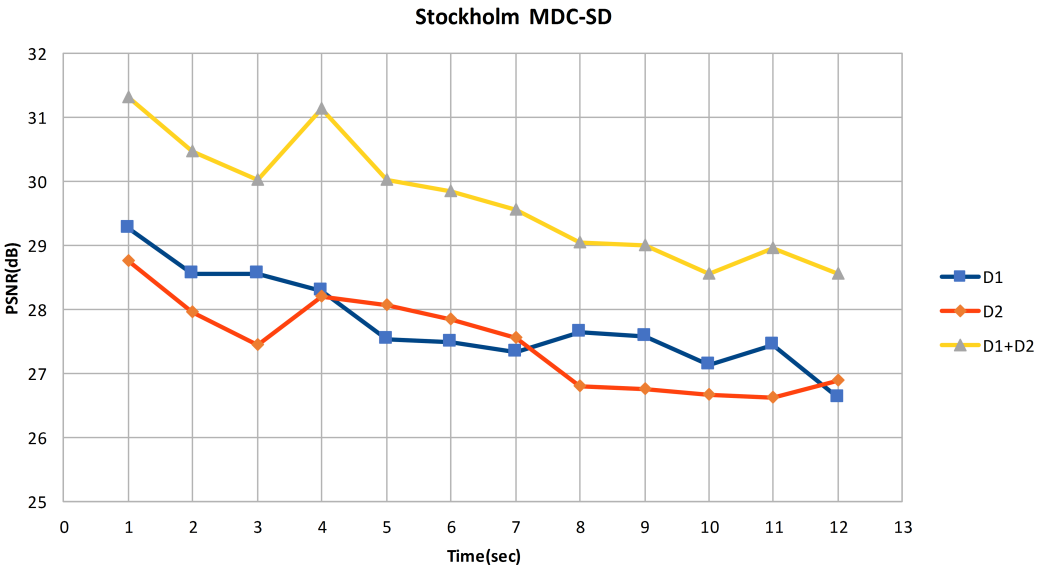


Figure 3.18: PSNR Stockholm (MDC-SD).

proaches. Furthermore, when both descriptions are received the best PSNR is attained when MDC-TD is used. However, this approach does not far overcome the PSNR value obtained by the other one. Then, the two approaches give identical results. On the other hand, about the quality of experience perceived by the user, when both descriptions are received, the image is noticeably clearer than when only one of the two descriptions is received. This results for the two approaches similarly.

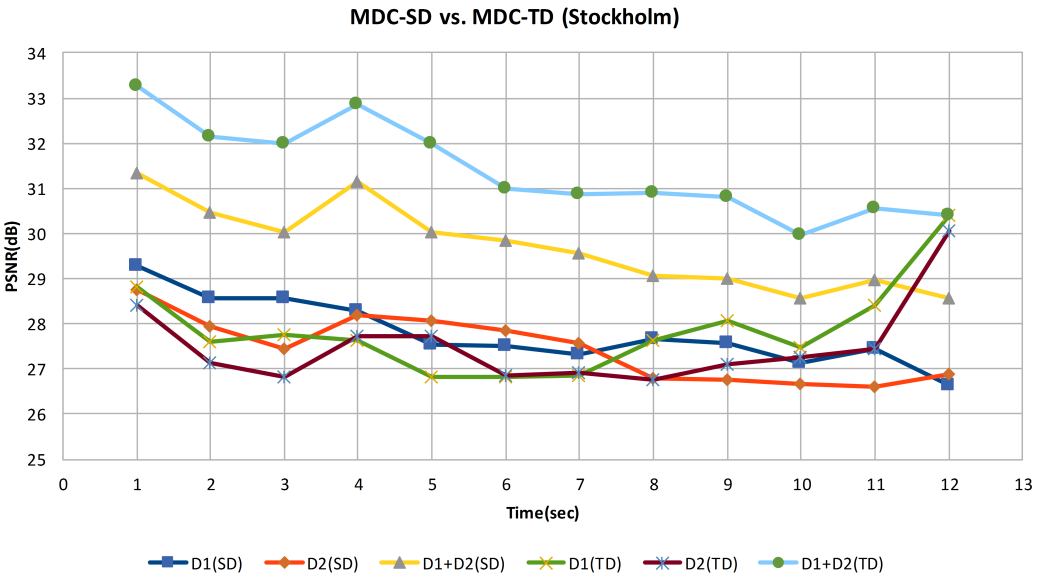


Figure 3.19: MDC-SD vs. MDC-TD (Stockholm).

2) Ducks

Ducks is a complex and coding difficult video sequence. This sequence has 300 frames and each description is encoded at a maximum bit-rate equal to 14500 kbit/s. In this manner, when both descriptions are received at the receiver side, the total

bit-rate, in the best case, is equal to 29000 kbit/s. Table 3.11 and Figure 3.20 present the PSNR obtained using MDC-TD. From this figure we can see that if just one description is received, the average PSNR is quite similar and equal to 26.4 dB. On the other hand, when both descriptions are received the PSNR is increased by almost 2 dB (28.18 dB).

Table 3.11: Bit-rates and PSNR of Ducks MDC-TD.

Ducks MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	14203.86	29.3336	14456.21	29.2503	31.1831
2	11162.73	27.8086	11015.28	27.5941	30.0627
3	11122.02	25.251	10224.84	25.1199	27.6108
4	11125.97	25.3878	14332.91	25.5586	26.9818
5	13028.89	25.2725	11768.63	25.1465	26.568
6	12468.56	25.5872	11761.94	25.4255	26.9205
7	14041.72	26.2735	11301.29	25.8641	27.6272
8	11504.72	26.4954	10092.89	26.1511	27.8714
9	11886.18	26.8072	10103.03	26.3882	28.2415
10	12846.97	27.0918	12254.75	26.998	28.8003

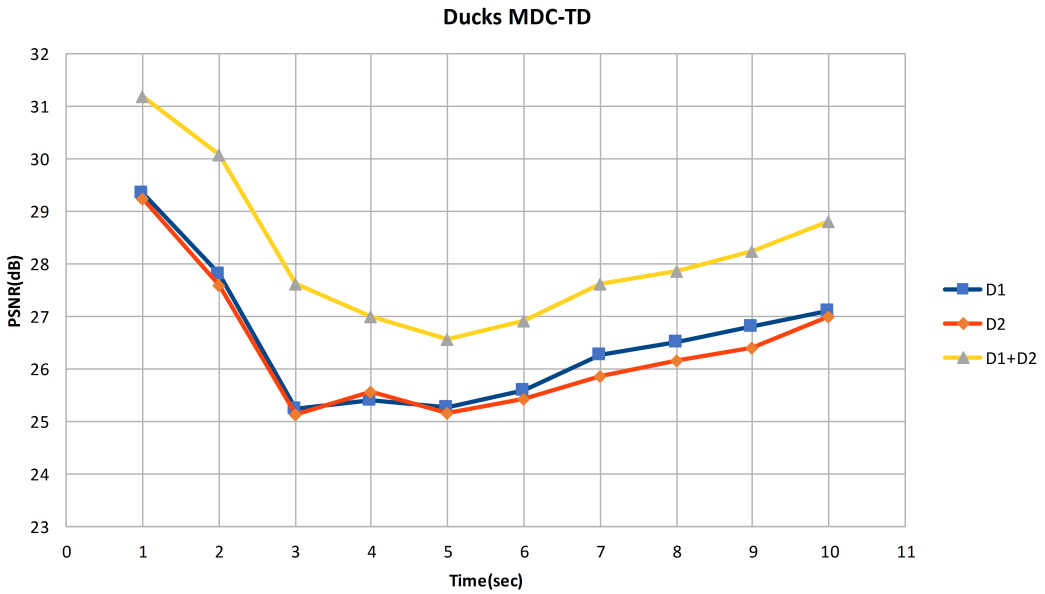


Figure 3.20: PSNR Ducks (MDC-TD).

Table 3.12 and Figure 3.21 show the PSNR obtained when the MDC-SD is used. When only one description is employed, the average PSNR is comparable and equal to 24.8 dB. If the both descriptions are received, the PSNR is increased at minimum by 2 dB and maximum by almost 4 dB. The average PSNR value, in this case, is equal to 27.6 dB. In Figure 3.22 is presented a comparison between the two MDC approaches applied in the current video sequence.

As we can see in Figure 3.22, when only one description is received, the MDC-TD achieves a higher PSNR than MDC-SD. The first one overcomes by almost 2 dB to the other approach. Moreover, when both descriptions are received, the two approaches present similar PSNR values. However, the MDC-TD presents a minimal increase equivalent to 0.5 dB with respect to the other approach.

Table 3.12: Bit-rates and PSNR of Ducks MDC-SD.

Ducks MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	14203.86	26.9414	14456.21	26.8319	30.5682
2	11162.73	26.6018	11015.28	26.5238	29.4014
3	11122.02	25.2318	10224.84	25.0972	26.9904
4	11125.97	24.2195	14332.91	24.3473	26.4183
5	13028.89	23.9304	11768.63	23.8248	26.1125
6	12468.56	23.9947	11761.94	23.8897	26.395
7	14041.72	24.2978	11301.29	24.1561	27.0641
8	11504.72	24.4755	10092.89	24.3213	27.3022
9	11886.18	24.6662	10103.03	24.4773	27.602
10	12846.97	24.8802	12254.75	24.7384	28.1519

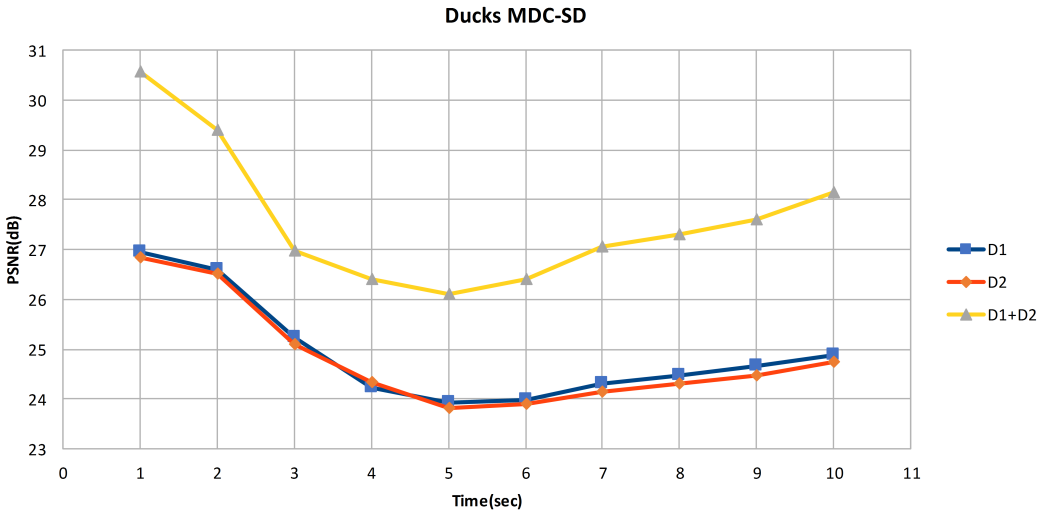


Figure 3.21: PSNR Ducks (MDC-SD).

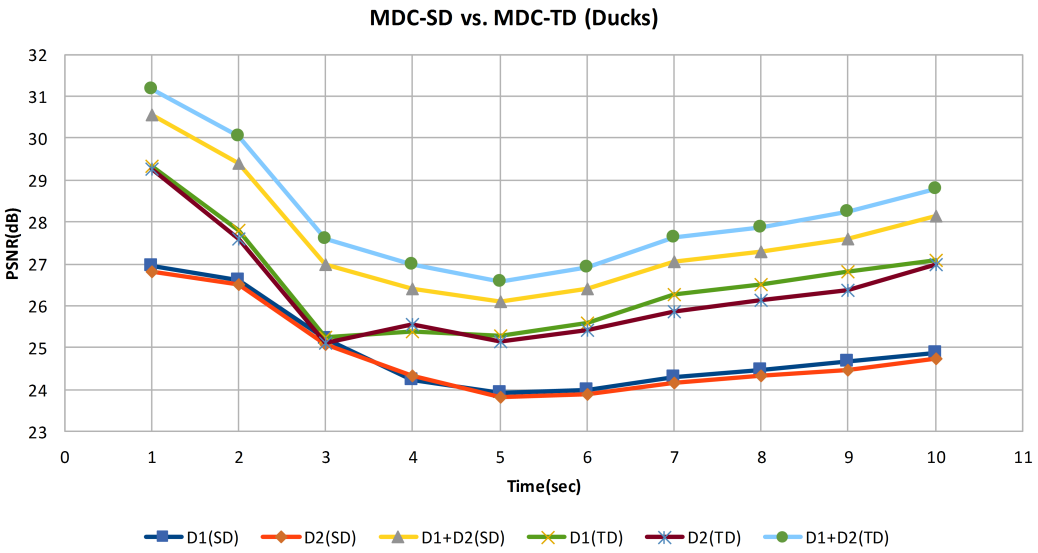


Figure 3.22: MDC-SD vs. MDC-TD (Ducks).

3) ParkJoy

As the previous video sequences, ParkJoy belongs to the same category, it is a complex video sequence. This video has 300 frames and each description is encoded at a maximum bit-rate equal to 14500 kbit/s. In such way, when both descriptions are received, the total bit-rate, in the best case, is equal to 29000 kbit/s. Table 3.13 and Figure 3.23 expose the PSNR obtained using MDC-TD. When just one description is received, the average PSNR is quite similar and equal to 23.6 dB. On the other hand, when both descriptions are received, the average PSNR increases significantly, attaining a value equivalent to 28 dB.

Table 3.13: Bit-rates and PSNR of ParkJoy MDC-TD.

ParkJoy MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	10366.99	24.5291	10963.39	24.5614	30.2241
2	10526.46	23.0066	14092.4	23.274	27.6497
3	11675.46	22.6218	12292.05	22.7098	27.0831
4	11774.54	22.4733	10013.04	22.4328	27.2052
5	13925.78	23.0603	12966.29	22.9369	28.4526
6	13569.18	22.6085	13301.24	22.6097	27.2224
7	12330.18	22.3707	11984.64	22.3793	26.9737
8	11439.78	23.0056	12607.3	23.1426	27.3641
9	13083.07	25.6582	10826.8	25.3999	29.5803
10	12647.4	27.733	10497.72	27.3904	29.1789

Moreover, Table 3.14 and Figure 3.24 illustrate the PSNR obtained when the MDC-SD is used. When only one description is used, the average PSNR is comparable and equal to 24.5 dB. If the both descriptions are received, the average PSNR

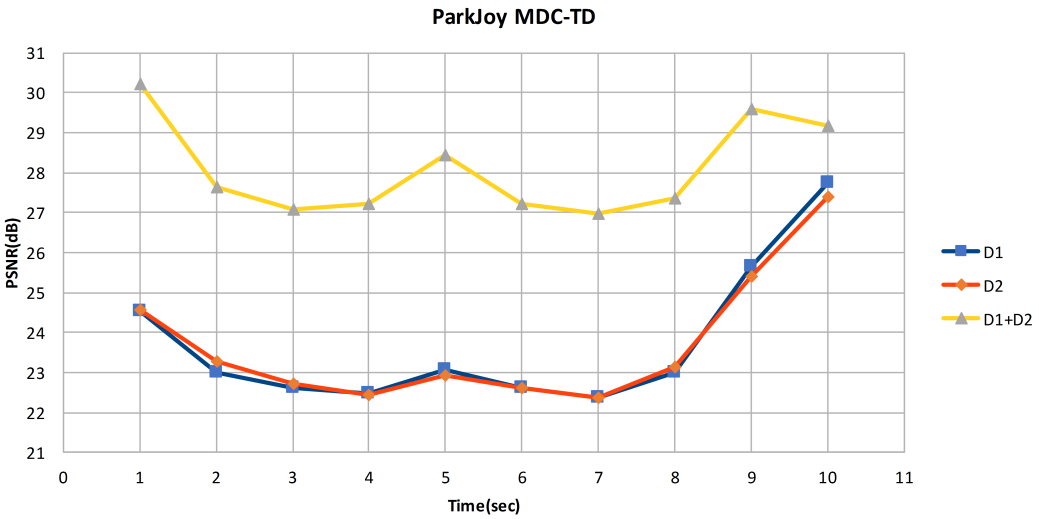


Figure 3.23: PSNR ParkJoy (MDC-TD).

is increased by around 2 dB (26.7 dB). Figure 3.25 presents a comparison between the two MDC approaches applied in the current video sequence.

Table 3.14: Bit-rates and PSNR of ParkJoy MDC-SD.

ParkJoy MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	10366.99	26.583	10963.39	26.637	28.7817
2	10526.46	24.2363	14092.4	24.4418	26.2921
3	11675.46	23.9164	12292.05	24.0649	25.9948
4	11774.54	24.2576	10013.04	24.2105	25.9494
5	13925.78	25.1519	12966.29	25.0964	27.1495
6	13569.18	23.9532	13301.24	23.9581	25.9224
7	12330.18	23.6829	11984.64	23.6573	25.7068
8	11439.78	24.0019	12607.3	23.9622	26.1212
9	13083.07	25.3622	10826.8	25.2473	27.9848
10	12647.4	24.269	10497.72	24.0978	27.4445

As shown in Figure 3.25, if just one description is received, a higher PSNR is obtained with MDC-SD. MDC-SD overcomes MDC-TD at maximum by 2 dB. This is because *ParkJoy* is a complex video sequence with great amount of details. Accordingly, the vectors of movement change substantially from one frame to another. Hence, if MDC-TD is applied the performance is expected to be lower than the obtained with the other approach. On the other hand, when both descriptions are received, the best PSNR is attained when MDC-TD is employed. In this case, MDC-TD overcomes by minimum 1 dB and maximum 1.7 dB the other approach.

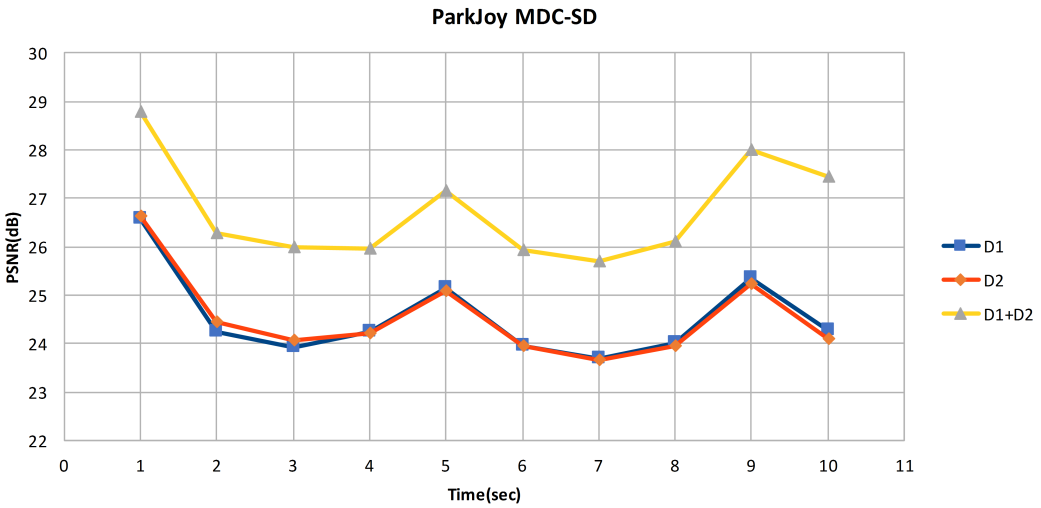


Figure 3.24: PSNR ParkJoy (MDC-SD).

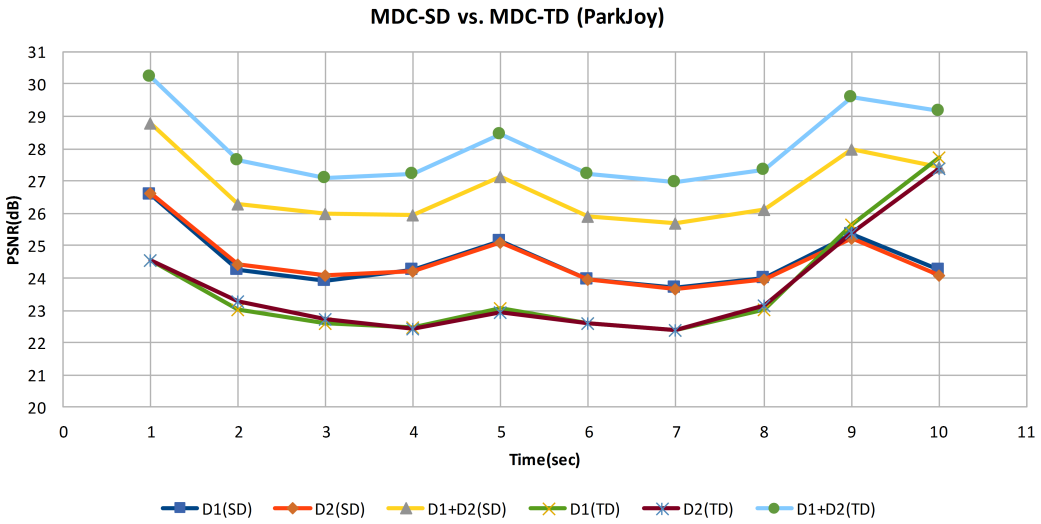


Figure 3.25: MDC-SD vs. MDC-TD (ParkJoy).

4) Shields

This video sequence has 504 frames and it is encoded at 50 frames per second. Moreover, Shields has been encoded using the same parameters of Stockholm. Table 3.15 and Figure 3.26 exhibit the PSNR achieved by MDC-SD. When only one description is received, the average PSNR is equal to 26.6 dB. Otherwise, if both descriptions are received, the PSNR is increased by around 2 dB, resulting in a PSNR equal to 28.26 dB.

On the other hand, Table 3.16 and Figure 3.27 present the PSNR achieved when the MDC-TD approach is applied in the current video sequence. When just one description is received, the average PSNR obtained is equal to 26.2 dB. Furthermore, when both descriptions are received, the PSNR presents a significant increment equivalent to almost 4 dB more than when only one description is received (30.2 dB).

In Figure 3.28 the two MDC approaches are compared. When just one descrip-

Table 3.15: Bit-rates and PSNR of Shields MDC-SD.

Shields MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	2875.34	27.3357	1276.17	26.3056	28.3795
2	1276.17	26.6388	2157.68	26.7837	28.3891
3	2211.04	27.1517	3105.4	27.2243	29.5979
4	2911.9	26.7114	2174.37	27.1621	29.0497
5	2938.22	26.7501	2535.01	25.5999	27.4811
6	1128.24	25.7779	2490.46	24.8369	26.7408
7	1368.4	24.5303	1898.24	25.7609	26.7904
8	3082.36	25.9712	2436.32	26.0678	27.1757
9	2622.5	27.7044	2773.53	27.6873	29.0889
10	2444.14	27.8244	1990.4	28.6575	30.0024

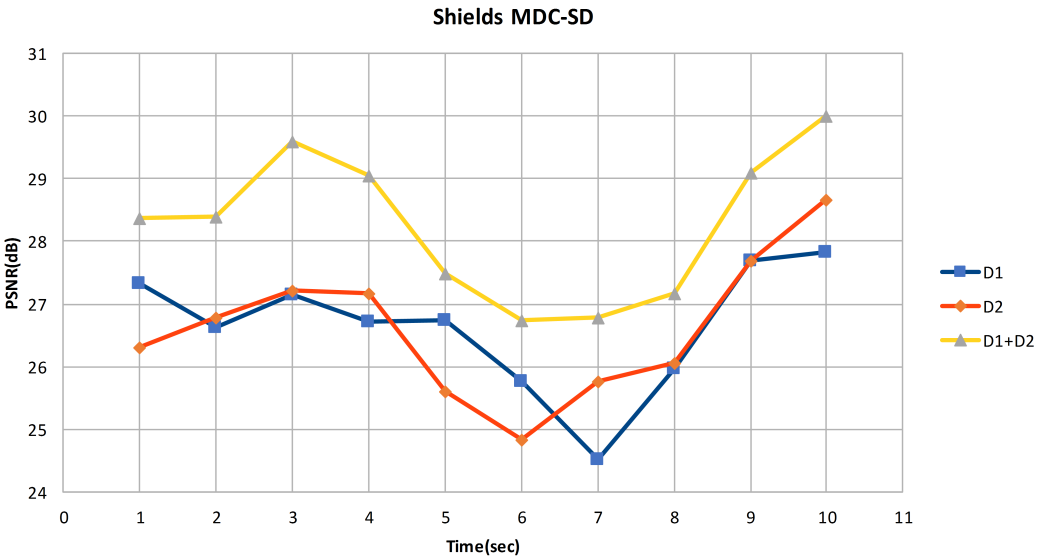


Figure 3.26: PSNR Shields (MDC-SD).

tion is received, the two MDC approaches produce quite similar results. However, when both descriptions are received, MDC-TD attains a better PSNR value. The PSNR obtained using MDC-TD is 2 dB greater than the PSNR achieved with MDC-SD.

Regarding the quality of experience perceived by the user, when just one description is received, the quality achieved by the two MDC approaches is comparable. However, when both descriptions are received, a better image is obtained when MDC-TD is used.

Table 3.16: Bit-rates and PSNR of Shields MDC-TD.

Shields MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	2875.34	26.6828	1276.17	25.4165	31.0394
2	1276.17	25.1246	2157.68	25.2501	30.3174
3	2211.04	25.782	3105.4	25.8045	31.4962
4	2911.9	25.2715	2174.37	25.6975	30.9796
5	2938.22	25.5243	2535.01	24.3819	29.5428
6	1128.24	25.591	2490.46	24.3374	28.6147
7	1368.4	26.5022	1898.24	28.1244	28.8082
8	3082.36	25.5392	2436.32	25.706	28.6308
9	2622.5	26.3923	2773.53	26.3586	30.3922
10	2444.14	30.1623	1990.4	31.0118	32.2247

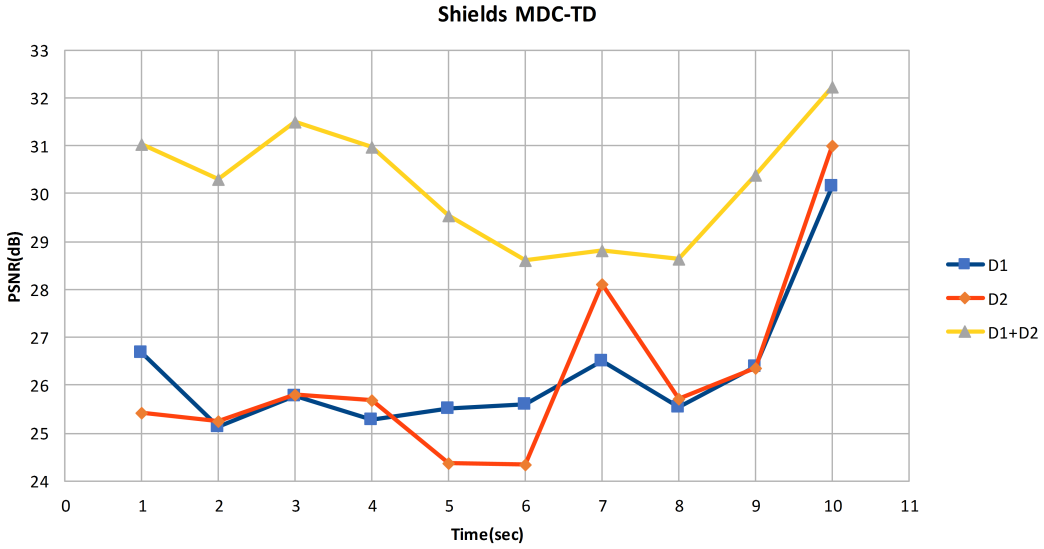


Figure 3.27: PSNR Shields (MDC-TD).

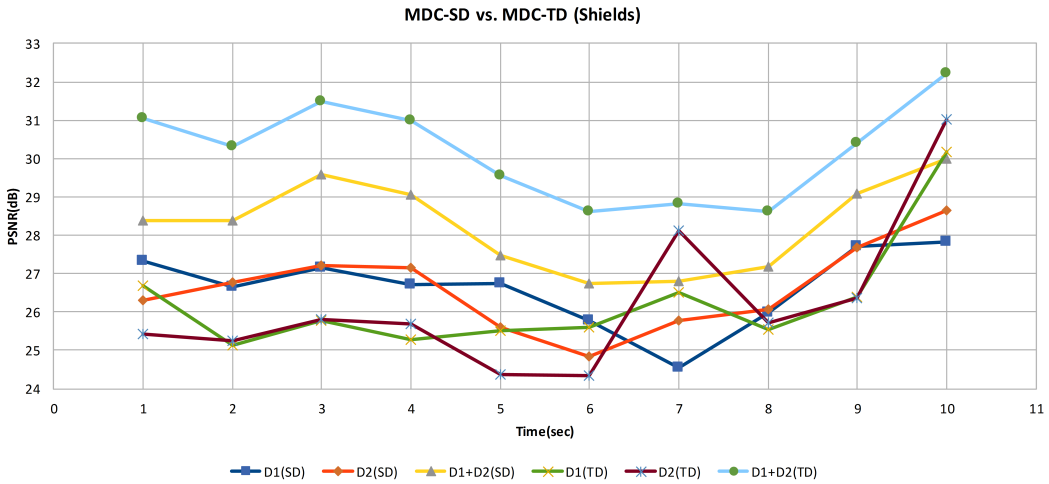


Figure 3.28: MDC-SD vs. MDC-TD (Shields).

5) Tennis

Tennis video sequence is encoded with a maximum bit-rate equal to 7000 kbit/s for each description. In such way, when both descriptions are received, the total bit-rate, in the best case, is equal to 14000 kbit/s. Moreover, Tennis owns 240 frames and it is encoded at 24 frames per second.

Table 3.17 and Figure 3.28 present the PSNR achieved using MDC-TD. When just one description is received, the average PSNR is quite similar and equal to 31 dB. Furthermore, when both descriptions are received the average PSNR increases to 40.95 dB, which is a significant increment, almost more than 10 dB.

Table 3.17: Bit-rates and PSNR of Tennis MDC-TD.

Tennis MDC-TD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	5946.73	29.3609	4729.62	29.1624	38.9966
2	5810.62	28.6906	4227.44	28.3405	38.2126
3	5081.83	28.5968	6393.24	28.792	38.4522
4	4302.38	28.45	5118.14	28.6586	38.0946
5	6795.85	28.6848	5166.36	28.4028	38.3877
6	4206.38	33.7511	5208.62	34.0926	41.7292
7	4959.42	33.08	4028.35	32.9391	43.8434
8	6648.85	33.785	5250.83	33.661	43.9396
9	6565.23	33.8123	5733.32	33.7566	43.0686
10	6611.72	32.8984	4885.17	32.8061	44.8361

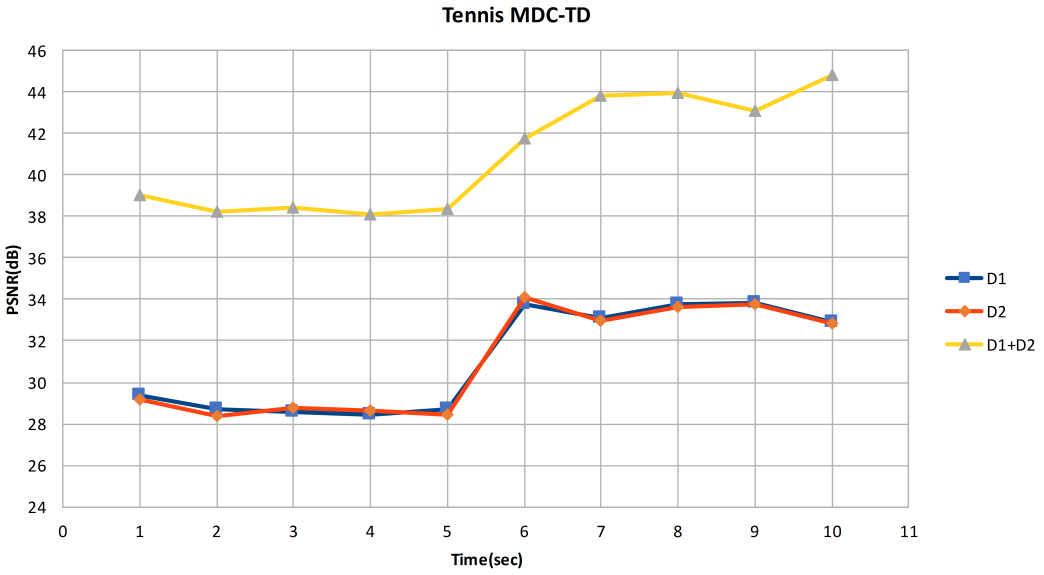


Figure 3.29: PSNR Tennis (MDC-TD).

Moreover, Table 3.18 and Figure 3.30 illustrate the PSNR obtained when MDC-SD is applied. When just one description is utilized, the average PSNR is identical

and equal to 37.4 dB. If the both descriptions are received, the PSNR is increased from 2 dB to maximum 3.5 dB (40 dB in average). Finally, the two MDC approaches applied in the current video sequence, are compared in Figure 3.31.

Table 3.18: Bit-rates and PSNR of Tennis MDC-SD.

Tennis MDC-SD					
	Description1		Description2		D1+D2
Time(sec)	Bit-rate (kbit/s)	PSNR(dB)	Bit-rate (kbit/s)	PSNR(dB)	PSNR(dB)
1	5946.73	36.0017	4729.62	35.8362	37.9837
2	5810.62	35.476	4227.44	35.1715	37.1891
3	5081.83	35.3241	6393.24	35.5381	37.4398
4	4302.38	35.2101	5118.14	35.387	37.1102
5	6795.85	35.4754	5166.36	35.2247	37.3616
6	4206.38	37.1461	5208.62	37.5847	40.7286
7	4959.42	39.3353	4028.35	39.072	42.8475
8	6648.85	39.7562	5250.83	39.5988	43.0219
9	6565.23	39.3539	5733.32	39.1682	42.3387
10	6611.72	41.6052	4885.17	41.4615	44.0559

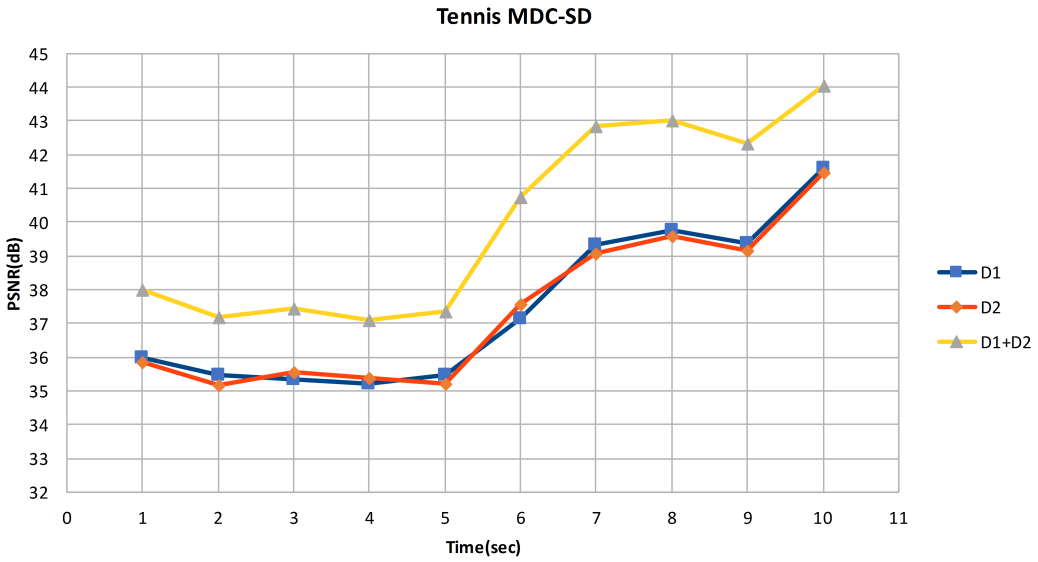


Figure 3.30: PSNR Tennis (MDC-SD).

As it is seen in Figure 3.31, if only one description is received, the best PSNR is attained when the MDC-SD approach is used. However, when both descriptions are received, the PSNR obtained with the two approaches is similar and really close. However, MDC-TD minimally overcomes MDC-SD by 1 dB.

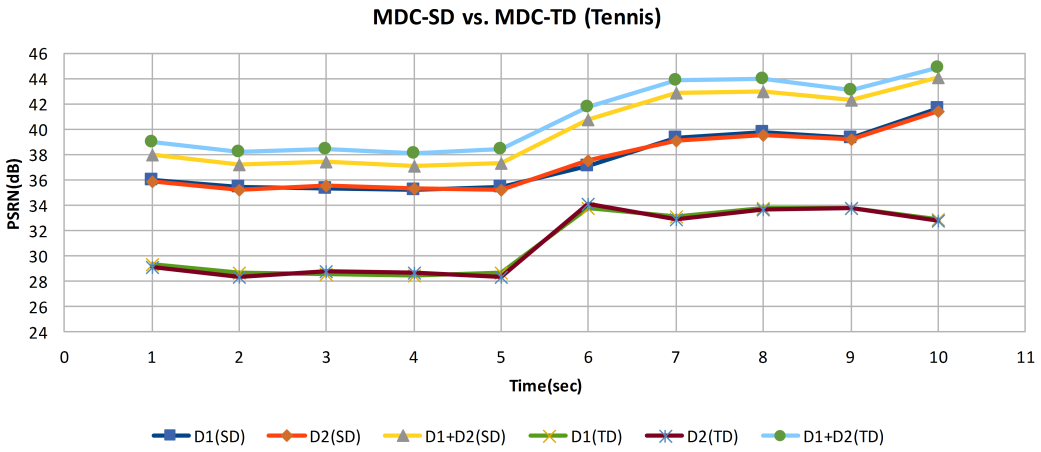


Figure 3.31: MDC-SD vs. MDC-TD (Tennis).

Final Considerations

From the obtained results applying the two MDC methods in the HD video sequences analyzed in this sub-section, we can deduce some important points which should be considered at the moment to select which method to use. These considerations are mentioned below:

- In most of cases, when complex video sequences with a great amount of movement are used, MDC-SD produces a higher PSNR, when just one of the two descriptions arrives to the receiver.
- When simple video sequences (low amount of movement)are used, MDC-TD gives better PSNR than MDC-SD. However, it is important to mention that the difference between the PSNR value attained by the two approaches is not so considerable. Therefore, in this case, any MDC approach could be used.
- If video sequences with a medium amount of movement are employed, MDC-TD widely overcomes the other approach when only one description is received.
- In all the cases, when both descriptions are received, the best PSNR is attained by MDC-TD.
- From the evaluated video sequences (CIF and HD), we can see that independently of the applied MDC approach, in most of cases, although that the available bit-rate for the two descriptions is quite different, the PSNR obtained is pretty similar. This is because the quality is lost during the interpolation process trying to reconstruct the video to its original characteristics (frame rate and resolution).
- Before applying any of the two MDC methods evaluated in this section, first we recommend to examine the features and the type of video sequence. Then, if you are not sure which would be the best MDC method to be applied in a specific case, we recommend to use MDC-SD. The reason is because if just one description is received, it is possible to obtain a good PSNR. Although, it is true that the PSNR value is not higher than the PSNR attained with

MDC-TD, it is relatively close. In the same way, when both descriptions are received, the PSNR obtained is not higher as the one achieved with MDC-TD, but it is acceptable.

3.3 Comparison of the performance of the H.264 encoding approaches

In this section all the H.264 encoding approaches are compared to determine which of them achieves the highest PSNR. The approaches evaluated are: Multiple Description Coding Spatial Domain (MDC-SD), Multiple Description Coding Temporal Domain (MDC-TD), Scalable Video Coding with Medium Grain Scalability (H.264 SVC/MGS), and Adaptive Video Coding (AVC). It is important to highlight that when AVC is used, we have just one bit-stream encoded at specific bit-rate, which depends to the QP parameter defined in the encoder configuration file. Moreover, to these AVC video sequences we used three different QP values to observe the difference between the bit-rates and the PSNR achieved depending to the QP assigned.

The PSNR is calculated at the maximum bit-rate achieved for the encoded stream. It is around 1000 kbit/s for videos encoded with MDC, and 2000 kbit/s approximately to videos encoded with SVC. In the last case, we doubled the target bit-rate because we have one stream instead of two, as occurs in MDC when the two descriptions are received, which is the best case. Four video sequences are analyzed below.

1) Calendar (CIF, 30 Hz)

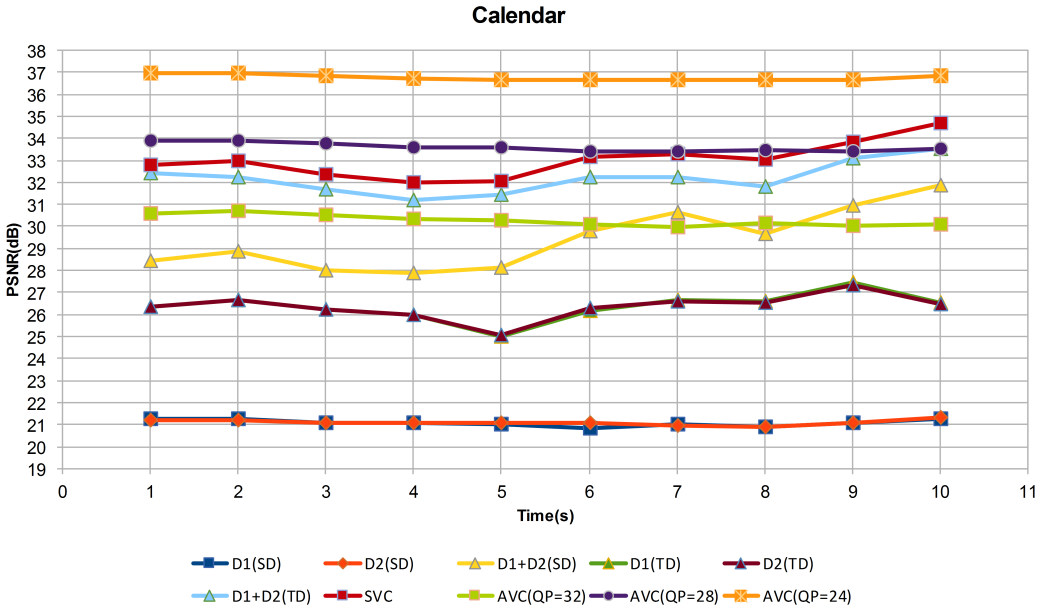


Figure 3.32: MDC vs. SVC and AVC (Calendar).

Figure 3.32 illustrates the PSNR attained for the different approaches. As we can see, it is evident that SVC overcomes the MDC approaches. However, SVC and MDC-TD (when both descriptions are received) are quite similar. SVC overcomes

MDC-TD by around 1 dB, which is not so considerable. An opposite situation occurs with MDC-SD because, also when the both descriptions are received, the PSNR achieved by SVC is greater. In this case, SVC overcomes MDC-SD at minimum by 2.6 dB and maximum by 4.37 dB. It is important to notice that when just one of the two descriptions is received, the PSNR obtained is under 30 dB.

On the other hand, when AVC is used, different bit-rates and PSNR are achieved depending of the QP parameter specified in the configuration file. The highest PSNR is attained when QP=24 is applied. A PSNR equal to 36.75 dB is produced by a bit-rate equal to 2900.19 kbit/s. Additionally, with QP=28, a PSNR of 33.58 dB is attained with a bit-rate equivalent to 1785.30 kbit/s. Finally, with QP=32 the PSNR achieved is equal to 30.27 dB with a bit-rate of 968.05 kbit/s. The drawback of AVC resides in the fact that several encoded streams should be generated to get different versions of the video sequence.

2) Football (CIF, 30 Hz)

Figure 3.33 shows the PSNR attained using different H.264 encoding approaches in the football video sequence. SVC overcomes both MDC-SD and MDC-TD (when the two descriptions are received) by 3.20 dB and 1.42 dB respectively. Similar to Calendar, MDC-TD (two descriptions) is closer to the PSNR achieved by SVC than to the other methods. SVC achieves a bit-rate equal to 1987.63 kbit/s and PSNR equivalent to 36.63 dB.

Moreover, SVC presents a higher PSNR than MDC-TD and MDC-SD, in the case that only one description is received. The PSNR of SVC overcomes by 8.27 dB and 9.35 dB the MDC approaches respectively. On the other hand, when AVC is used, the values obtained are: when QP is equal to 24, the video achieves 2157.78 kbit/s and 39.26 dB; when QP is equal to 28, the video attains 1445.56 kbit/s and 36.65 dB; when QP is equal to 32 we get 940.89 kbit/s and 34.03 dB.

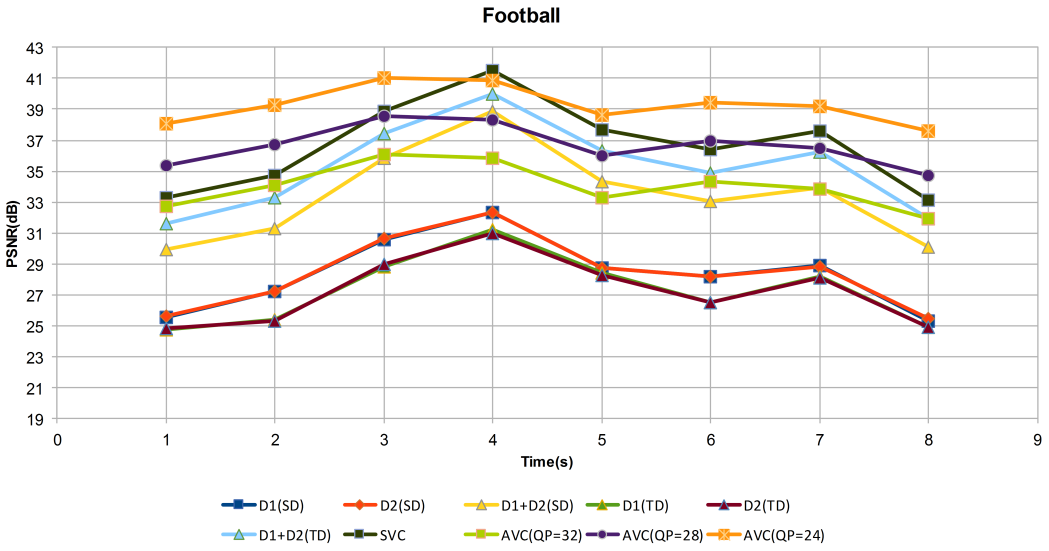


Figure 3.33: MDC vs. SVC and AVC (Football).

3) Bridge-Close (CIF, 30 Hz)

Figure 3.34 shows that SVC overcomes both MDC-SD and MDC-TD (when both descriptions are received) by around 0.4 dB. This difference is irrelevant, and thus

we could conclude that a quite similar PSNR is obtained by the three approaches mentioned above. The average bit-rate attained by SVC is 2003.535 kbit/s and the PSNR is equal to 38.34 dB.

Furthermore, the PSNR achieved by MDC-TD (when just one description is received) is higher than the one attained by MDC-SD. However, also in this case, the SVC approach overcomes the MDC-TD (one description) by almost 2 dB.

On the other hand, when AVC is used the values obtained are: when QP is equal to 32 the video achieves 213.63 kbit/s and 32.644 dB; with a QP equal to 28 the video attains 464.48 kbit/s and 34.93 dB; Finally, with a QP equal to 24 the video achieves 1157.46 kbit/s and 37.58 dB. It is worth to mention that despite using a low QP value, the PSNR obtained by AVC is not so high as the one gotten in the video sequences analyzed previously. It could be due to the type of the video sequence.

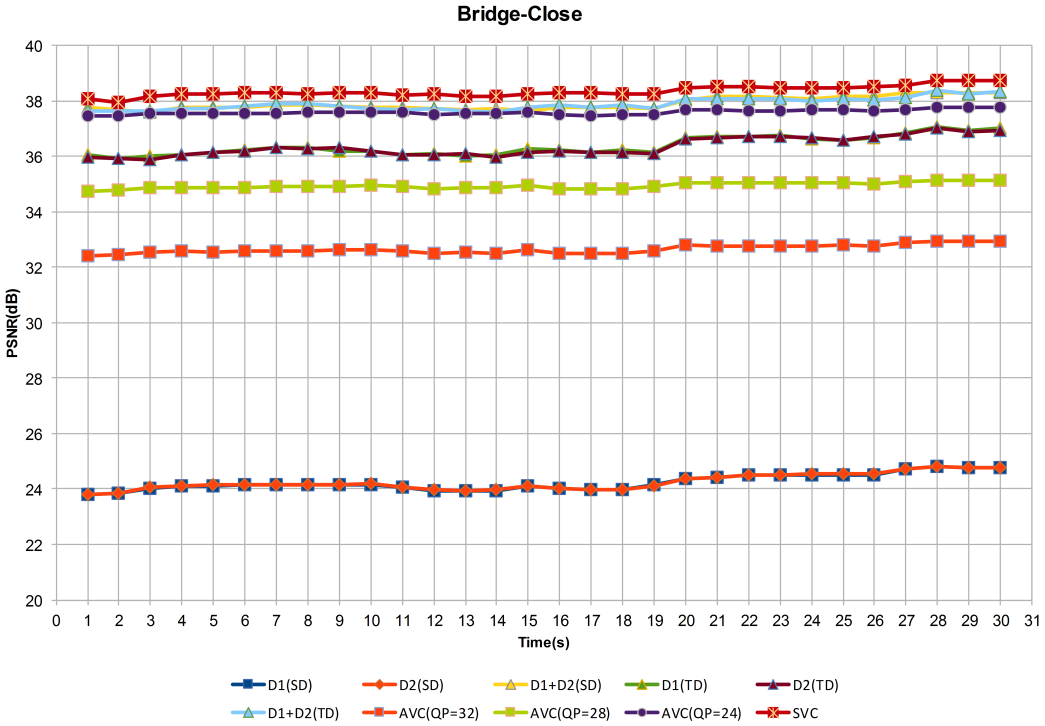


Figure 3.34: MDC vs. SVC and AVC (Bridge-Close).

From the results presented above, we can conclude that the best PSNR is obtained by SVC and MDC, when both descriptions are received. So, these two approaches are analyzed in detail and in this occasion, in presence of bit-rate fluctuations. It is important to take into account that the bit-rate used to calculate the PSNR of SVC is obtained by the sum of the bit-rates employed in the two MDC descriptions. All the video sequences analyzed previously are evaluated another time in a bandwidth variation environment, where the bit-rate differs each second.

1) Calendar

As we can see in Figure 3.35, the SVC-MGS approach and the MDC-TD (when both descriptions are received) present a identical PSNR value. However, the first one overcomes by 0.77 dB to the other approach. Furthermore, SVC-MGS produces a better PSNR than MDC-SD (both descriptions), overcoming it by almost 4 dB.

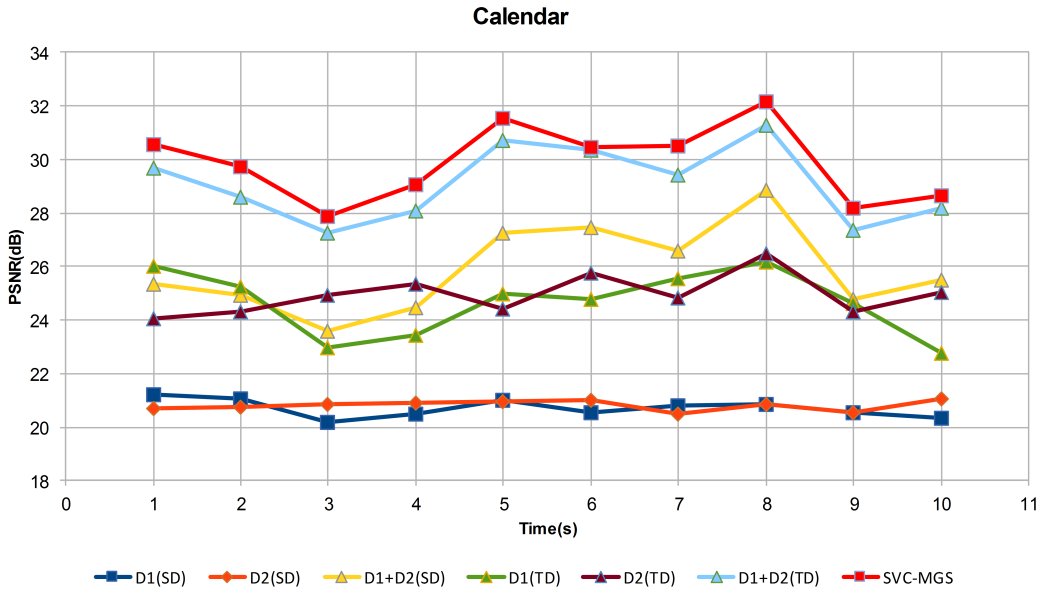


Figure 3.35: MDC vs. SVC-MGS (Calendar).

2) Football

Figure 3.36 shows that even when the two descriptions are received, SVC-MGS overcomes MDC-TD by 1.586 dB in average, and by 3.33 dB to MDC-SD.

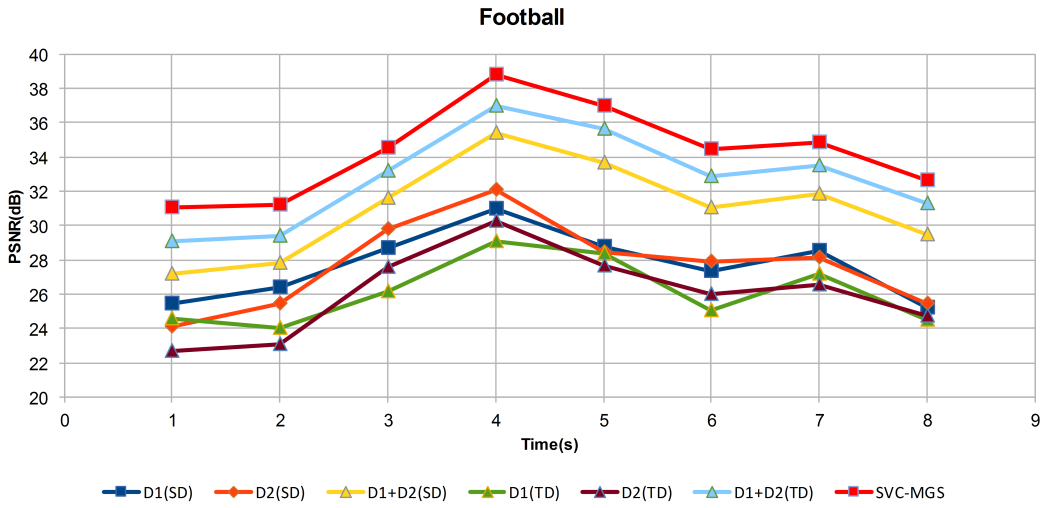


Figure 3.36: MDC vs. SVC-MGS (Football).

2) Bridge-Close

As shown in Figure 3.37, the PSNR obtained with MDC-TD, MDC-SD (both descriptions) and SVC-MGS is similar. There is a minimal difference between the three approaches aforementioned, SVC overcomes by 0.7 dB in average to MDC-TD and by 0.88 dB in average to MDC-SD.

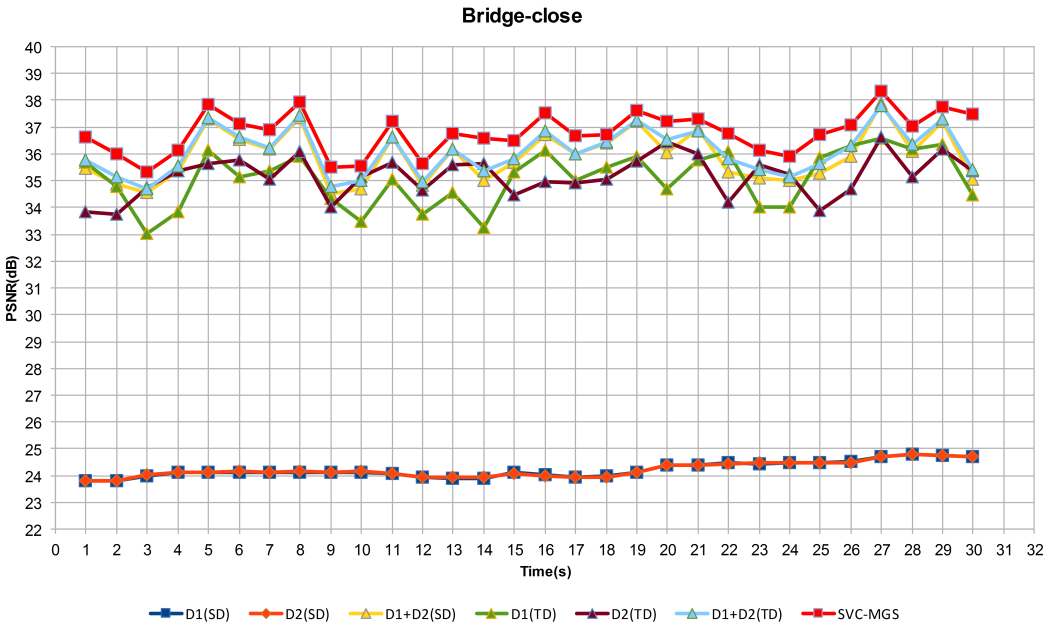


Figure 3.37: MDC vs. SVC-MGS (Bridge-Close).

Final Considerations

- Before selecting a specific H.264 encoding approach is crucial to analyze and define the scenario. For instance, if we have an environment where the network presents a constant bandwidth, we could encode the video sequence using AVC with a suitable QP value in order to achieve the available bit-rate. Furthermore, if the bandwidth oscillates in the time, it would be better to use SVC or Multiple Descriptions Coding with MGS.

  In this section, we assumed no packet losses, but in a real environment this aspect must be considered. In this case, if MDC is used and some packets belonging to Description 1 are lost, these could be reconstructed with the packets of the other description that were received. On the other hand, if SVC is employed and packet losses take place, it would not be possible to recover these packets and consequently the PSNR would be decreased.

- Considering no packet losses, and calculating the PSNR at the maximum bit-rate achieved by the encoder, the best PSNR value is attained by Scalable Video Coding (SVC-MGS). But, it is closely followed by MDC-TD when both descriptions are received.

Chapter 4

Bandwidth Estimation Model using HMM

The estimation of the available bandwidth (AB) of an end-to-end path has received noticeable attention due to its relevance in several network applications. Transport layer protocols might also use AB information to change the transmission rate according to the amount of available bandwidth in the path, using the network resources efficiently while avoiding congestion [18].

In the literature, various techniques for estimating the bandwidth to increase the network throughput have been proposed. A recent survey of bandwidth estimation (BE) techniques is presented in [19], where 55 BE techniques are classified and analyzed. [19] divides the BE techniques into four categories: active probing techniques (APT), passive techniques (PT), techniques only for wireless networks (TOWN) and other BE techniques (OBET).

The active probing techniques (APT) consist in estimate the available bandwidth along the path by measuring the packet inter-arrival times. For doing this, probing packets are sent through the network at multiple traffic rates from sender node to receiver node. The objective of this technique is to understand the network characteristics. APT is further categorized as single packet APT (SPAPT) and packet pair APT (PPAPT). In the first one, probing packets are injected with predefined time interval into the network path with the purpose of measuring the link capacity. On the other hand, the PPAPT consists in sending back-to-back two packets which are called packet-pair with the aim of determining the time separation between packets at the target node. In turn, PPAPT is sub-divided into another four categories which are detailed in [19].

The behavior of a probing packet pair after leaving the tight link is shown in Figure 4.1. As it is seen in the figure, if two consecutive packets are sent to the network path, they arrive to the node with a determined initial time-separation between them (Δ_{in}). After interacting in the tight link queue with the cross traffic coming from different sources, the pair of probing packets will leave the router with a new time-separation (Δ_{out}). The difference between them Δ_{out} - Δ_{in} is the packet pair dispersion. This dispersion value (delay) can be positive, when $\Delta_{out} > \Delta_{in}$; negative if $\Delta_{out} < \Delta_{in}$ and equal to zero when the link has no enough cross traffic to affect the initial packet separation ($\Delta_{out} = \Delta_{in}$).

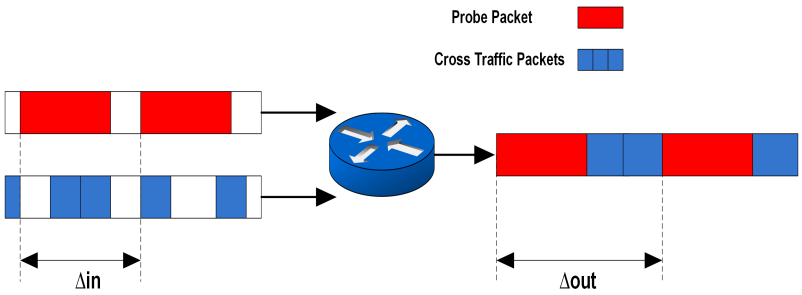


Figure 4.1: Probe Gap Model.

The well-known Probe Gap Model (PGM) and Probe Rate model (PRM) use packet pair dispersion to estimate the available bandwidth. The PGM is also known as direct probing and it bases the estimation on the gap dispersion between two consecutive probing packets at the receiver [20]. It is a lightweight and fast AB technique. Further, PRM is based on the idea of induced congestion, in which the AB is determined by the variation in the probing packet rate from sender to receiver.

On the other hand, passive techniques (PT) unlike APT do not inject probing packets for calculating the bandwidth, it works on network traces collected earlier. The dispersion and delay are observed for the data flow and ACK packets. Moreover, PT is divided in generic passive techniques and proactive PT for wireless network. The third category corresponds to techniques only for wireless networks (TWON). Bandwidth estimation techniques developed for wired networks are focused on point-to-point dedicated links. These techniques are not directly usable in wireless networks due to additional data which is introduced in the communication. For instance, in wireless ad hoc networks (MANETs) the idea of a point-to-point link does not exist as an independent communication resource between a pair of neighbor nodes due to the shared nature of the transmission medium, and the random nature of multiple access protocols. Finally, OBET compiles several techniques that have not been categorized.

Using the different BE techniques, many protocols and tools have been reported in the literature. The most relevant works are presented below.

4.1 Overview

A lot of available bandwidth estimation tools have been proposed in the literature. In this section, the most relevant tools are addressed. Pathload [21] and PathChirp [22] are two tools which use the Self Loading Periodic Streams (SLoPS)[23] technique which follows the same principle of the probe rate model. In general terms, SLoPS is based on the fact that the one way delay (OWD) of a periodic packet stream increases when the rate of the probing traffic is higher than the available bandwidth in the path. Therefore, a fleet of streams are sent at varying rates and the OWD trend of each stream is then characterized at the receiver as either increasing or decreasing. Hence, the available bandwidth is given by the rate at which an "increasing" trend in the stream starts to be observed.

Pathload is presented as an active measurement tool which proposes an algorithm to detect the "increasing" trend in the OWDs of a stream. The method consists in divide the K OWDs measurements in groups. Then, for each group, it is calculated

the median of the group. Ultimately, using two statistics it is possible to determine if the stream shows an "increasing" trend or not. The applied statistics are: the pairwise comparison test (PCT) and the pairwise difference test (PDT). Contrary to typical approach where a strong "increasing" trend in the OWDs will be detected when PCT is close to one, Pathload reports an "increasing" trend when PCT $>$ 0.55, a "non-increasing" trend if PCT $<$ 0.45, and an "ambiguous" trend otherwise.

Same as above, when PDT is used, a strong "increasing" trend in the OWDs will be detected when PDT is close to one. In Pathload, an "increasing" trend is notified if PDT $>$ 0.66, a "non-increasing" trend if PDT $<$ 0.54, and an "ambiguous" trend otherwise. Pathload employs both statistics, PDT and PDT, to determine the trend of the stream. Thus, if one of the PCT and PDT values reports "increasing" trend, while the other is either "increasing" or "ambiguous", the stream is defined as "increasing". If one metric reports "non-increasing" trend while the other is either "non-increasing" or "ambiguous", the stream is defined as "non-increasing". Finally, if both metrics report "ambiguous", or when one is "increasing" and the other is "non-increasing", the stream is discarded [24].

On the other hand, PathChirp [22] increases the probing rate within each stream in an exponential manner due to the use of *chirp probing trains*. Chirp trains are highly efficient and provide several advantages over packet pair. The first one is that each chirp of N packets has N - 1 packet spacings that would normally require $2N$ - 2 packets using packet pairs. Second, exponentially spaced packets require only $log(G_2)$ - $log(G_1)$ packets to probe the network over the range of rates [G_1,G_2] Mbps. One other advantage of chirps is that they capture critical delay correlation information that packet pairs do not. Hence, exploiting the properties of chirps, PatChirp rapidly estimates the available bandwidth using few packets.

Another interesting tool is the Spread Pair Unused Capacity Estimate (Spruce) which is presented in [25]. Based on the PGM approach and using a Poisson process of packet pairs, Spruce samples the arrival rate at the bottleneck by sending pairs of packets spaced, so that the second probe packet arrives at a bottleneck queue before the first packet leaves the queue. Then, using the dispersion of the probe packets measured at the receiver, it calculates the average rate of the traffic that arrives to the queue between the two packets. Finally, Spruce computes the available bandwidth as the difference between the path capacity and the arrival rate at the bottleneck.

Initial gap increasing (IGI) [26] is another method which uses the probe gap model. The authors develop two packet pair techniques to characterize the available bandwidth. One is IGI and the other one is PTR (packet transmission rate). The algorithm presented in this work uses the information about changes in gap values of a packet train to estimate the competing bandwidth on the tight link of the path. The PTR method uses the average rate of the packet train as an estimate of the available bandwidth. With purpose of evaluating the performance of available bandwidth estimation tools and techniques, [27] proposes a low cost and flexible testbed over a fully controlled network. This testbed employs real networking equipment and specialized software that allows researchers to test the various tools under different networking scenarios and conditions. Moreover, [27] introduces a novel analytical model based on network queues which exploits Jackson's model [28]. This model can be apply to obtain reference values in order to compare them with the experimental results. As a proof of concept of both the testbed and the

model, [26] analyzes and compares the performance of Pathload, IGI and Spruce. From experiments, [27] concludes that Pathload is the most accurate tool but the slowest to converge. On the other hand, IGI is the fastest tool but the least accurate. Meanwhile, Spruce is the least intrusive tool with intermediate accuracy and convergence time.

With the aim of improving some issues presented in the techniques analyzed above such as: the long convergence times, accuracy errors, and the amount of overhead that is introduced, [1] proposes a client-server tool to estimate the available bandwidth. This tool is called *Traceband* and is based on hidden Markov model (HMM). Using probing packets pair dispersion information and HMM is possible to estimate the available bandwidth during a period of time. In order to get information about the AB dynamics, [1] uses the PGM for sampling the network and to obtain the delay between the consecutive packets at the receiver. These values are therefore discretized and used in the HMM to represent different levels of availability denominated states. Then, a modified version of the Baum-Welch algorithm presented in [29] is applied to determine the most likely sequence of states or AB that is generated by the observations. It is important to mention that the employed HMM model considers that large dispersion times in the packets are related to a loaded network and vice versa. Using this HHM approach, Traceband provides fast, continuous, and accurate AB estimates.

Thanks to the HMM, at the Traceband client is possible to get a reduction in the number of packet pairs employed to run the estimations. It is because HMM is able to learn the AB dynamics with an initial sample and keep the model updated with samples of reduced size. Moreover, in order to keep the overhead controlled and low, Traceband utilizes different values for the intra-gap and inter-gap times of packet pairs. The intra-gap or Δ_{in} refers to the time between the two packets of each packet pair. It is specified at the sender and it is set equal to the transmission time of a single probing packet in the tight link. On the other hand, the inter-gap corresponds to the time between pairs of probing packets. Eventually, Traceband is compared with Pathload and Spruce providing better performance results overall. Traceband is as accurate as the other two approaches but considerably faster, and introduces less overhead. Additionally, Traceband is able to react and accurately estimate the available bandwidth under abrupt changes in cross-traffic.

A recent study presented in [30] proposes an accurate and non-intrusive probing scheme and a rate adjusted algorithm that can be used for estimating the available bandwidth of an end-to-end network path. The proposed algorithm, New Enhanced Available Bandwidth Measurement Technique (NEXT) is based on the concept of self-induced congestion and like PathChirp, it estimates the AB by launching a number of packet chirps from sender to receiver. Unlike to other approaches, NEXT features a new probing train structure in which there is a region where packet rates are sampled more frequently than in other region. Furthermore, [30] develops a rate adjustment algorithm that adjusts the rates every round in such a way that the expected AB fits into the high density regions. The rate adjustment algorithm adjusts the range between the lower rate and the upper rate appropriately using two spread factors, which enables to keep the number of packets small. Combination of these two ideas lead to estimate the AB with greater accuracy and stability, outperforming the results obtained by the PathChirp algorithm.

Finally, a extended version of NEXT [30], NEXT-V2 is presented in [31]. It

is an active probing algorithm which offers and efficient measurement scheme for end-to-end AB estimation in a fixed, WLAN and 4G/LTE network. In addition to the packet generation and the rate adjustment algorithm, this new version includes the modified excursion detection algorithm and the packet loss recovery algorithm. Based on the Excursion Detection Algorithm (EDA) proposed in PathChirp, this work develops a modified EDA. This EDA is based on the principle of self-induced congestion and it assumes that increasing queuing delay implies less AB than the instantaneous packets rate and that decreasing queuing delay signifies the opposite. On the other hand, the Packet Loss Recovery (PLR) algorithm is created from the fact that packet loss has a great impact in the accuracy of AB estimation. Some tools previously analyzed as PathChirp and Pathload discard estimates when packet loss occurs to avoid errors in AB estimation computation. However, this derives in longer and more variable measurement times. Then, instead of discarding estimates when packet loss occurs, [31] reconstructs the one-way queuing delay curve by considering if a single packet loss occurs or multiple packet losses occur. Simulations and experimental results demonstrate that NEXT-V2 attains AB estimations in real time and overcomes other AB estimation tools in terms of accuracy, intrusiveness, and convergence time.

Based on [18] and [1], we propose a bandwidth estimation model based on hidden Markov model applied to scalable video coding (SVC). The HMM, parameters and employed methodology are explained below.

4.2 Hidden Markov Model

The available bandwidth can be modeled by N states, each one representing a certain level of availability. We consider a one-step transition Markov chain to estimate the probability of being in a particular state or AB range. As the available bandwidth cannot be directly observed, the Probe Gap Model is employed to get the dispersion between packets. This dispersion is used to estimate the amount of cross-traffic in the tight link during a period of time T which is subtracted from the capacity to estimate the AB in the path. The dispersion times used in this work have been obtained from measurements of YouTube traffic and available online network traces of video streaming sessions. The network traces dataset containing information about the exchange of packets during YouTube and Nextflix streaming sessions is accessible in [32] and [33]. We use Wireshark [1] for both packets capture and analysis of the pcap files. When a network trace file is parsed, it is possible to know relevant information about packets such as : packet timestamp, tcp sequence number, and packet length of each packet exchanged between the streaming server and client. Using the packet timestamps, the dispersion times are calculated. These times correspond to the time employed to download each packet. For instance, dispersion time of packet i will be equal to $timeStamp_i$ - $timeStamp_{i-1}$. Figure 4.2 represents the Hidden Markov Model (HMM) with discrete hidden states X representing the available bandwidth levels and discrete observation variables ξ representing probing packet pair dispersions. A particular observation has associated a probability B to be generated by a particular hidden state. Transitions between states are governed

[1]Wireshark is a free and open source packet analyzer. It is available in: https://www.wireshark.org/

by probabilities specified in the transition probability matrix A.

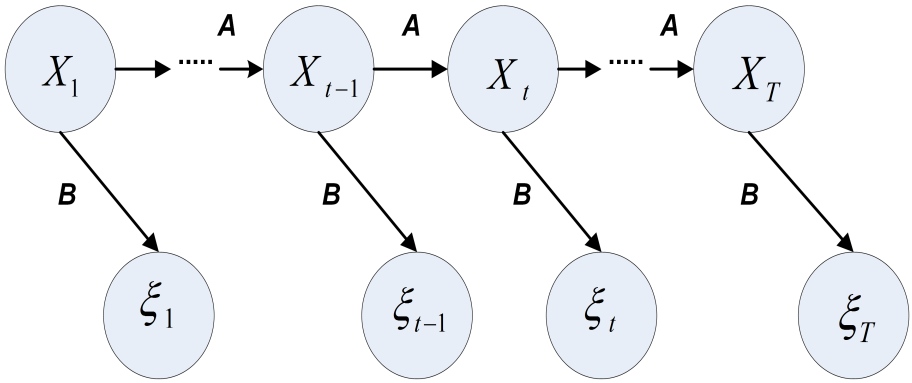


Figure 4.2: Hidden Markov Model.

As defined in [34], the Hidden Markov Model is defined by a tuple of five elements:

1. number of states (N)
2. number of distinct observation symbols per state (M)
3. state transition probability matrix (A)
4. observation probabilities (B)
5. initial state probabilities (Π)

Since with MGS the video sequences are encoded with a finite number of possible rates, a finer granularity in bandwidth is not useful for our purposes. Therefore, the number of states in the model (N) is related with the number of the target bit-rates used to encode the video sequence. The set of states is defined by $S = S_1, S_2, ... ,S_N$ where the available bandwidth grows from S_1 (low) to S_N (high). The state at time t is denoted by X_t.

The number of symbols (M) is the set of symbols denoted by $V = V_1, V_2, ... ,V_N$ corresponding to observed dispersions from the probing sampling method. These symbols are decimal number corresponding to the delay between consecutive packets; these delay values are grouped in M intervals of values to convert every single observation into a discrete symbol.

The matrix A contains the transition probability between the states. $A = [a_{ij}]$ where $a_{ij} = P(X_{t+1} = S_j \mid X_t = S_i)$, $1 \leq i,j \leq N$. Since only one-step transitions between states are considered possible, the number of unknown elements in the matrix is reduced to the three main diagonals:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & 0 & \dots & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & a_{N-1,N-2} & a_{N-1,N-1} & a_{N-1,N} \\ 0 & \dots & 0 & a_{N,N-1} & a_{N,N} \end{bmatrix}$$

The matrix B has the set of probabilities that indicates how likely is that at time t a specific observation symbol ξ_t is generated by each state from the set S. More specifically, $B = [b_j(m)]$ where $b_j(m) = P(\xi_t = v_m \mid X_t = S_j)$ for $1 \leq m \leq M$ and $1 \leq j \leq N$:

$$\begin{aligned} b_{S_1} &= \left[P\left(\frac{\xi_1}{S_1}\right), \ldots, P\left(\frac{\xi_M}{S_1}\right)\right] \\ b_{S_2} &= \left[P\left(\frac{\xi_1}{S_2}\right), \ldots, P\left(\frac{\xi_M}{S_2}\right)\right] \\ &\vdots \\ b_{S_N} &= \left[P\left(\frac{\xi_1}{S_N}\right), \ldots, P\left(\frac{\xi_M}{S_N}\right)\right] \end{aligned} \tag{4.1}$$

It is expected that the small values of ξ are the result of a highly available bandwidth, and therefore more likely generated by a high index state and conversely for high values.

Given an observation sequence $O = \xi_1, \xi_2, \ldots, \xi_T$, that is a set o samples from the network during T, it is desired to estimate the model $\lambda = (A,B,\Pi)$ that most likely has generated that sequence. In order to do this, matrix A and matrix B must be determined based on some experimental data sequence via some estimation algorithm. The most commonly used of which is the Baum Welch (BW) [35] one. Once A and B are given, the HMM can be used to generate an estimated symbol and state sequence with the same statistical properties of the original sequence.

With the purpose of analyzing the implemented model and its accuracy, two experiments changing the values of the transition matrix A and the observation probabilities B have been carried out. These experiments are presented in the section below.

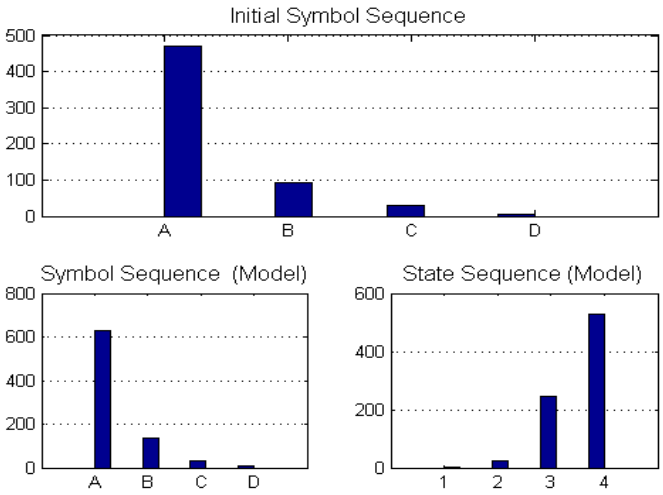
4.3 Model definition and experimental results

Experiment 1: change the A matrix values

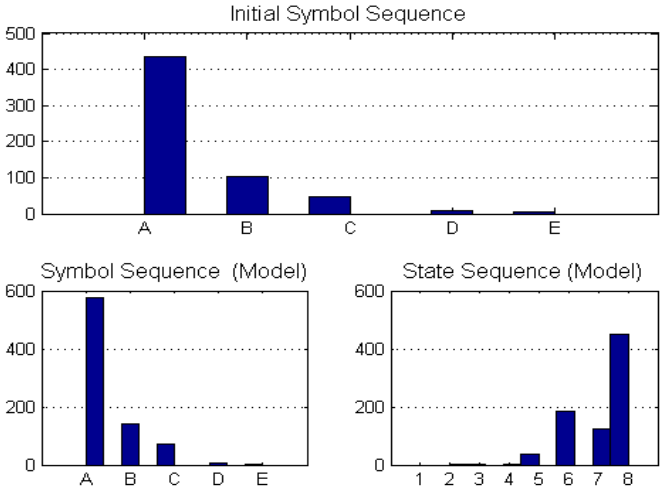
There are two ways to construct the A matrix. The first one is using random values. The second form is to assign the same probability to all the transitions between states for all the defined states. The only condition that must be respected is that the sum of the probabilities of each state should be equal to 1. It is important to take into account that the number of states(N) and the number of symbols (M) vary depending on the application.

a) Equally likely values

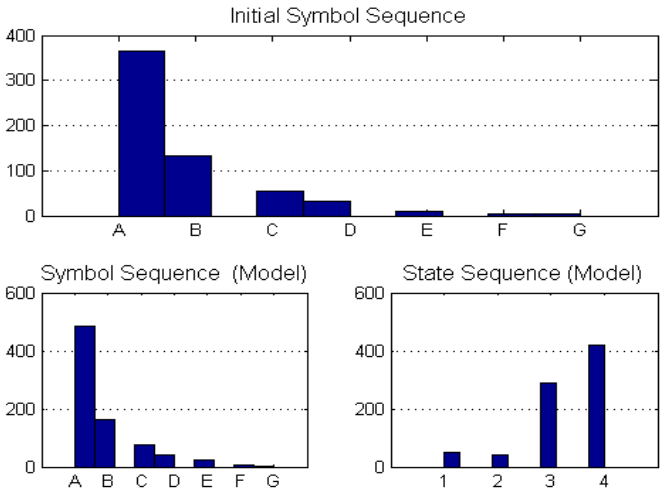
Figure 4.3 illustrates the estimated symbol and state sequence generated by the HMM, when the values of the transition matrix are equally likely. Basically, we analyze three cases: 1) N and M own the same value. 2) N is greater than M and 3) M is greater than N.



(a) N=4;M=4



(b) N=8;M=5



(c) N=4;M=7

Figure 4.3: Symbol and state sequence generated by a matrix A equally likely.

It is important to notice that in all the cases presented above, the estimated symbol sequence is pretty similar to the initial one, which is created from the probing packets. The principal target is getting a state sequence consistent with the symbol sequence. It means that low delays (A,B) should be related with high state indexes (4,3) which represent high available bandwidth and vice versa.

b) Random Values

Figure 4.4 shows the estimate symbol and state sequence generated by the HMM, when the values of matrix A are randomly created. Identical to the previous case, the state sequence, after BW, in all the cases present a similar behavior. This is because the initial A matrix is constructed based on guesses. Then, it is possible to construct the initial A matrix (one-step transition) in a random way without change the final result.

Experiment 2: Change the B matrix values

Using an A matrix with random values, the same set of experiments has been done changing the B matrix values. This matrix can be constructed using random values, equally likely values or a defined pattern.

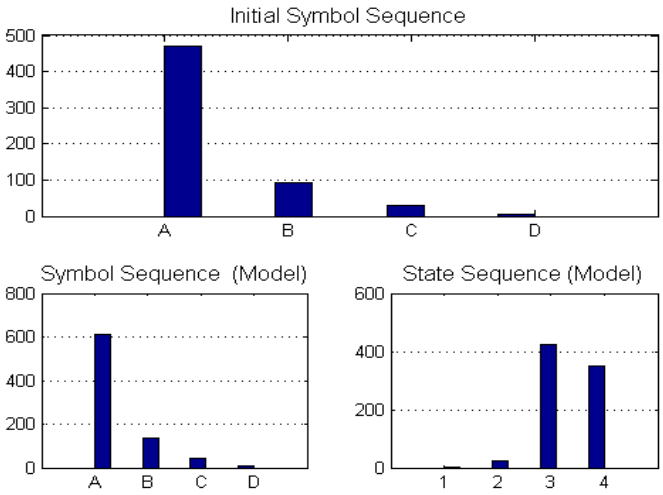
a) Equally likely values

Figure 4.5 illustrates the symbol and state sequence generated by the model when all the symbols own the same probability of appearing in all the states.

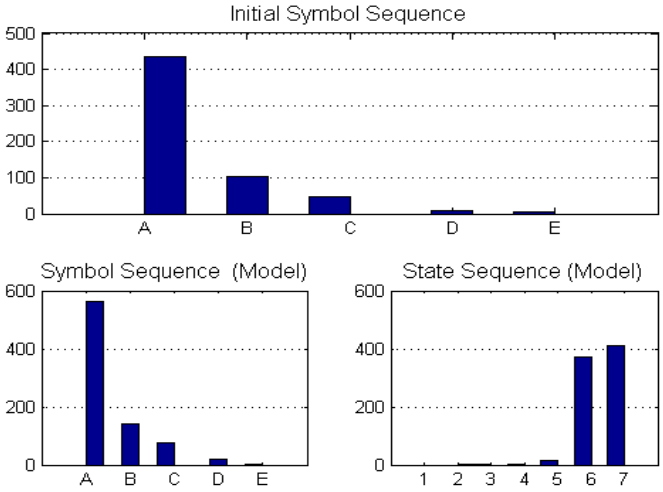
As shown in Figure 4.5, the symbol sequence generated by the model is similar to the initial one but the state sequence, in all the three cases, presents a different and non coherent performance. The reason is due to the fact that all the states have the same occurrence probability, which it is not true. For instance, in Figure 4.5 (c), from the generated symbol sequence, the probability of having high delays(symbols: E,F,G,H) which are related with low states (1,2) is minimal. However, the state sequence produces an unexpected result. Therefore, employing equally likely values to construct the initial B matrix is not reliable.

b) Random values

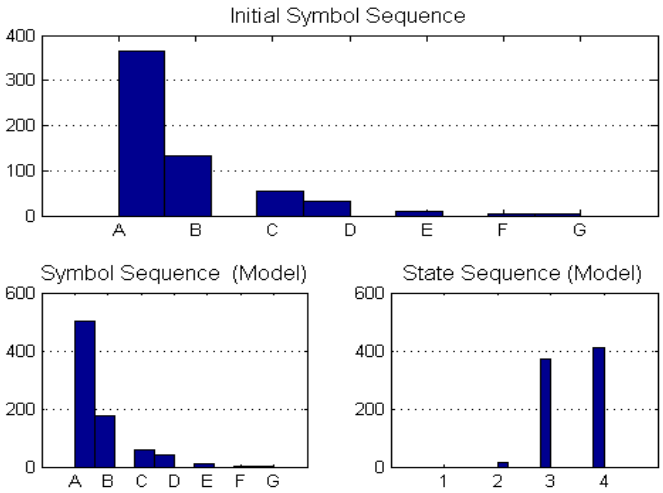
Figure 4.6 shows the symbol and state sequence generated using a B matrix constructed with random values. Like in the previous case, the symbol sequence generated by the model is similar to the initial one. The problem resides in the state sequence, because due to the random nature of matrix B, the state sequences produced by the model vary drastically. For instance, in Figure 4.6 (a) and (c), the state sequence is no so far to the expected result and it could be acceptable. However, this does not happen in Figure 4.6(b) where the state sequence does not reflect what is indicated in its symbol sequence. Then, employing a random B matrix is not completely trustworthy because also the generated state sequence is arbitrary. In some cases this sequence could be as expected and other times not.



(a) N=4;M=4

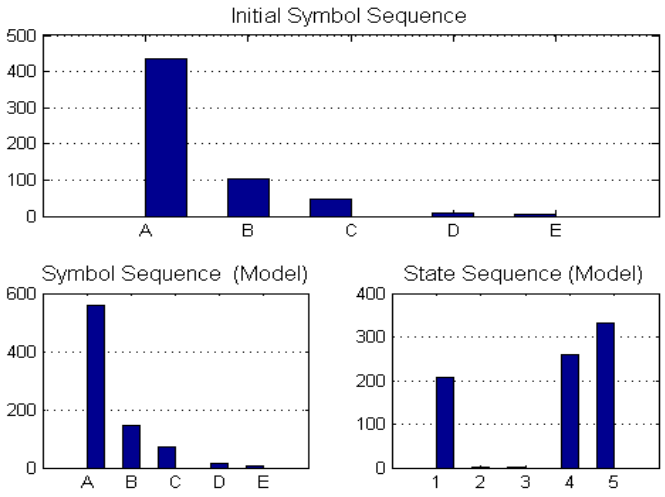


(b) N=7;M=5

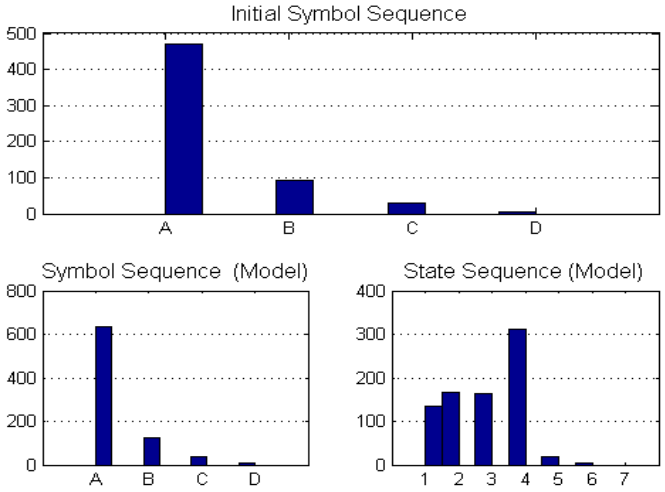


(c) N=4;M=7

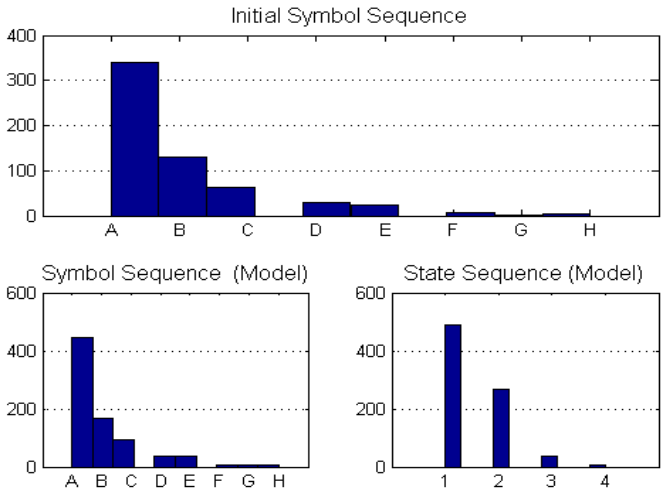
Figure 4.4: Symbol and state sequence generated by a matrix A with random values.



(a) N=5;M=5

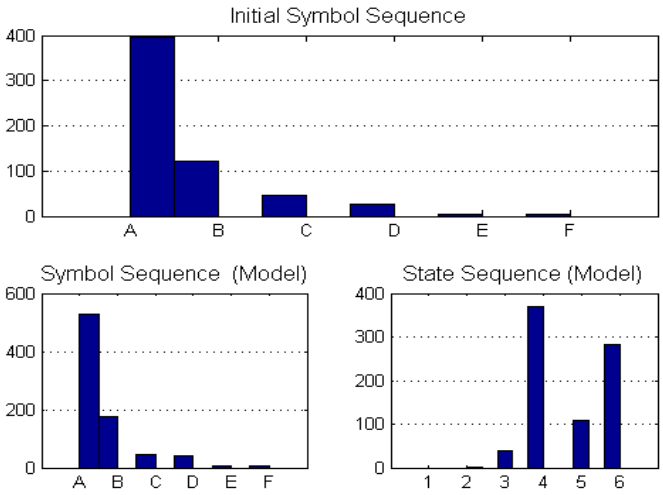


(b) N=7;M=4

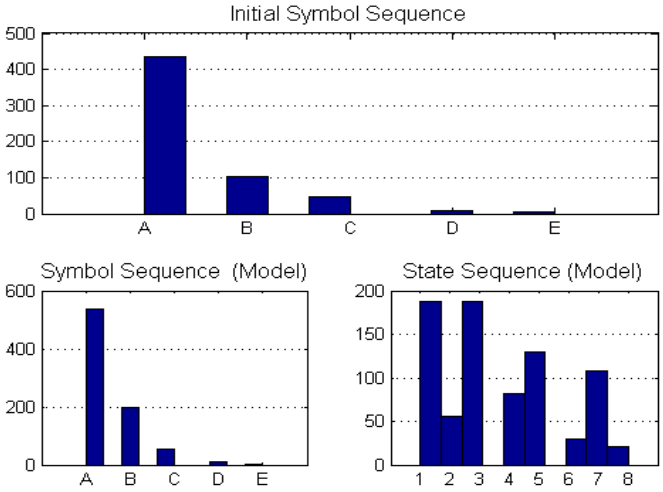


(c) N=4;M=8

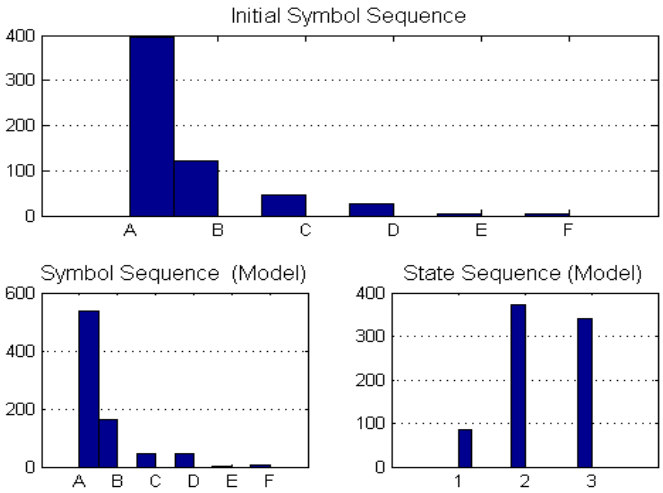
Figure 4.5: Symbol and state sequence generated by a matrix B equally likely.



(a) N=6;M=6

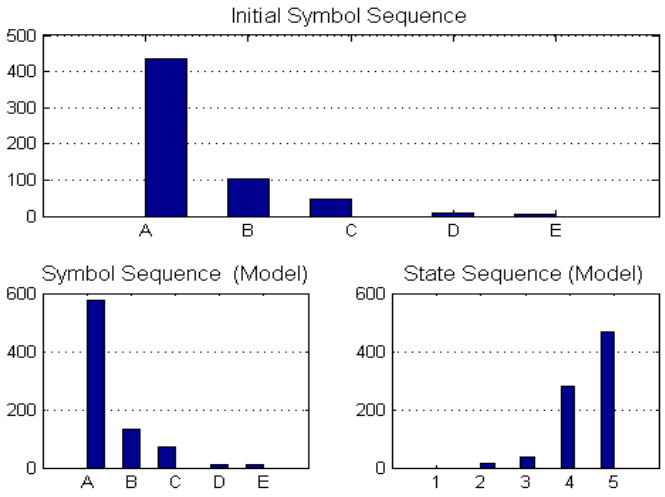


(b) N=8;M=5

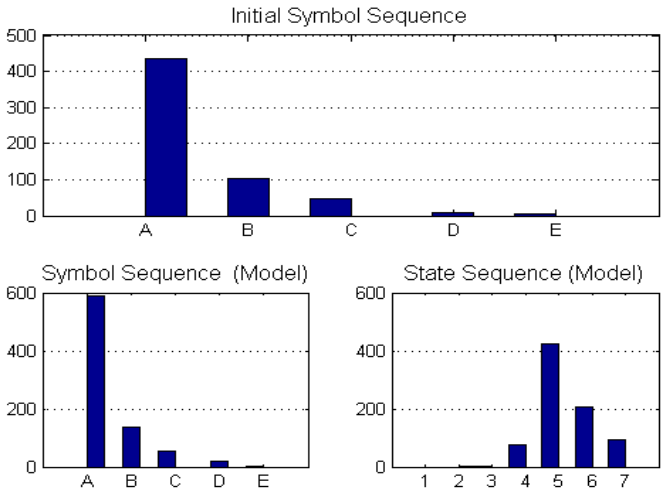


(c) N=3;M=6

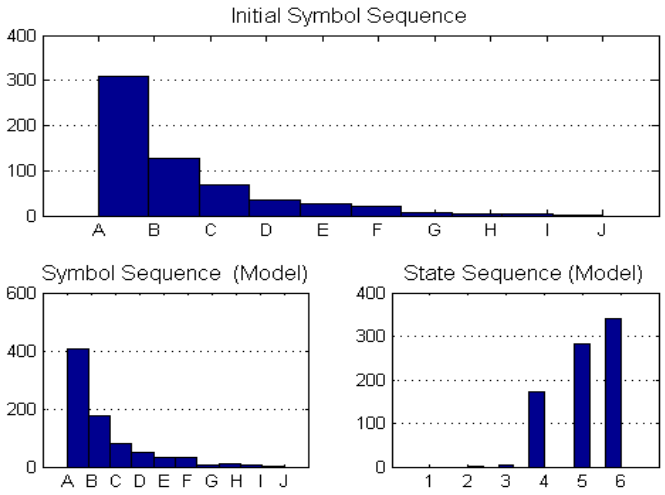
Figure 4.6: Symbol and state sequence generated by a random matrix B.



(a) N=5;M=5



(b) N=7;M=5



(c) N=6;M=10

Figure 4.7: Symbol and state sequence generated by a matrix B using the pattern.

c) Using a defined pattern

From the previous experiments we can deduce that the state sequence depends directly on how the B matrix is constructed. We can create this matrix following a specific pattern. This pattern considers the fact that small values of ξ (delay) are the result of a non loaded network, and therefore more likely generated by a high order state (i.e. high available bandwidth) and vice versa. Figure 4.7 shows how matrix B is composed. The columns represent the symbols which are associated to delay times. The first columns, from left to right, correspond to short delays and the last ones to the opposite. On the other hand, rows represent the states or AB, where state 1 is the lowest AB and state 7 is the highest one. As we can see in the figure, there are symbols where the probability of being produced by an specific state is higher than in others. Then, this probability is equally decreased toward the extremes.

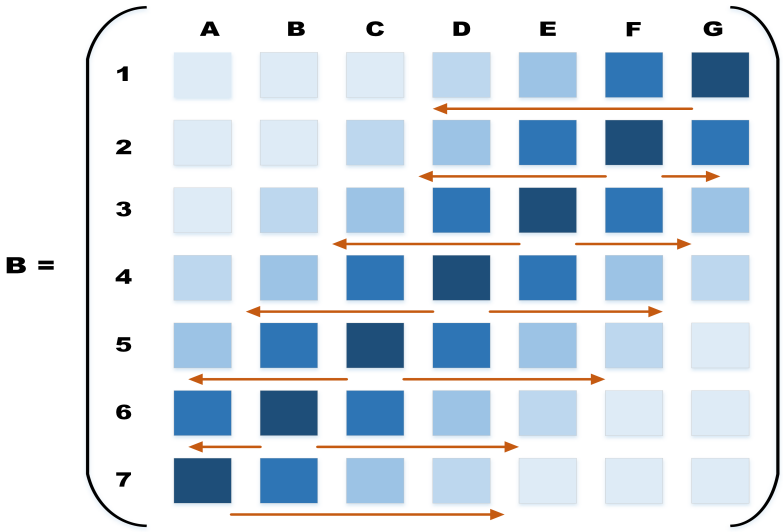


Figure 4.8: Assignation of probabilities to matrix B.

Figure 4.8 illustrates the symbol and state sequence generated using a B matrix as defined above. As it is seen in figure, the state sequence in all the three cases is related with the symbol sequence created by the model. It is important to highlight that a more accurate output is obtained when the number of states and the number of symbols are the same. This is due to each symbol is associated to a single state, in that case the probability of belonging to a specific state in an instant of time is more precise.

Therefore in this thesis, to generate the symbol sequence and the state sequence, we use a matrix A composed by random values and a matrix B made by values which represent the pattern defined by the model. Examples of the A and B matrices employed in this work are presented below, where we have 7 states (available bandwidths) and 7 symbols (delay time ranges). Figure 4.9 shows the symbol and state sequence produced. To construct the matrices we use values which present four digits after decimal point, but it is arbitrary. The only condition that has to be respected is that the sum of the values of each row must be equal to 1.

$$A = \begin{bmatrix} 0.4735 & 0.5265 & 0 & 0 & 0 & 0 & 0 \\ 0.0759 & 0.5460 & 0.3780 & 0 & 0 & 0 & 0 \\ 0 & 0.1057 & 0.3018 & 0.5926 & 0 & 0 & 0 \\ 0 & 0 & 0.4603 & 0.4639 & 0.0758 & 0 & 0 \\ 0 & 0 & 0 & 0.4022 & 0.3966 & 0.2011 & 0 \\ 0 & 0 & 0 & 0 & 0.5867 & 0.1040 & 0.3092 \\ 0 & 0 & 0 & 0 & 0 & 0.5362 & 0.4638 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.0916 & 0.1033 & 0.1345 & 0.1542 & 0.1674 & 0.1676 & 0.1814 \\ 0.0347 & 0.0663 & 0.0739 & 0.1631 & 0.2053 & 0.2365 & 0.2202 \\ 0.0300 & 0.1209 & 0.1577 & 0.1926 & 0.1955 & 0.1708 & 0.1326 \\ 0.0414 & 0.0996 & 0.2271 & 0.4306 & 0.1131 & 0.0881 & 0.0001 \\ 0.0975 & 0.2007 & 0.4875 & 0.1608 & 0.0328 & 0.0084 & 0.0124 \\ 0.1858 & 0.2125 & 0.1452 & 0.1301 & 0.1382 & 0.0929 & 0.0954 \\ 0.1866 & 0.1824 & 0.1732 & 0.1639 & 0.1183 & 0.1058 & 0.0698 \end{bmatrix}$$

The Bandwidth Estimation Model can be applied in SVC video sequences to simulate bandwidth variations. In that case, each one of the defined states corresponds to one target bit-rate attained by the encoder. For instance, based on the example represented in Figure 4.9, the number of states and symbols is equal to seven as the total number of rate points specified in the encoder configuration file. Hence, the state sequence generated by the model simulates the network availability during a period of time (video duration time).

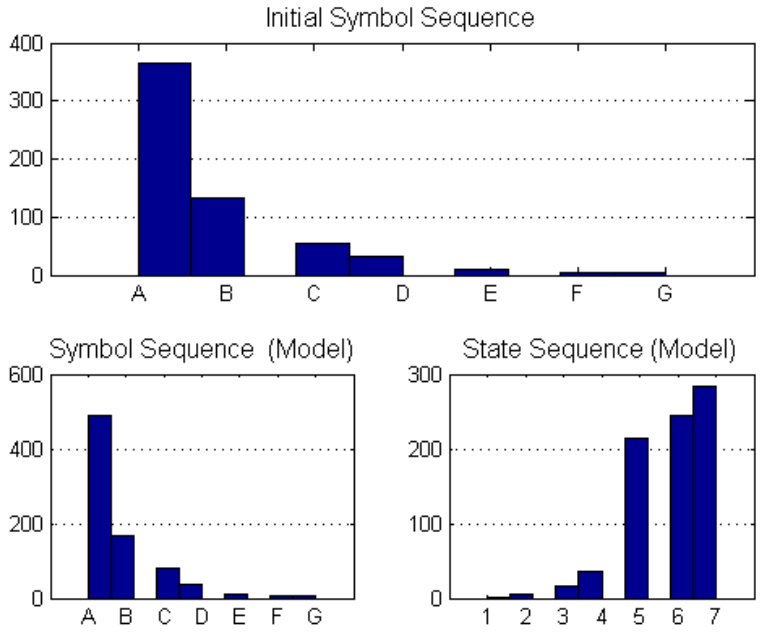


Figure 4.9: Symbol and state sequence generated by a random A matrix and a B matrix defined by the pattern. N=7 and M=7.

Chapter 5

Buffer management for SVC video sequences

Delivering variable bit-rate video streaming over the Internet is a challenging task. [36] exposes the principal hurdles to the effective multimedia transmission, which are indicated below:

- **Loss of important data in the network.** Most video encoding schemes encode video into packets with different relevance. Nevertheless, these packets are dropped in the transmission randomly. So, the main obstacle in effective multimedia streaming is the loss of important data in the network.
- **Jitter.** Video also suffers from jitter due to the variation in rate of the congestion control scheme.
- **Burst losses** in network that lead to lose a set of packets containing information about a single frame, and thus making estimation techniques at the receiver ineffective.
- **Loss of synchronization** between the encoder and decoder due to network losses.
- **Loss of a significant amount of data with a loss of single packet** that renders quality reconstruction almost impossible.

In order to face the issues listed above, and guarantee the presentation quality of multimedia applications, buffer management and scheduling algorithms must be applied to provide the QoS metrics required in a video transmission.

The transmission of video over IP-based networks demands its segmentation into IP-packets. Generally, the size of an IP-packet is smaller than the average size of a video frame. Therefore, a video frame is segmented into several IP-packets during the transmission process. Based on this fact, two buffer management strategies are identified. The first one is a packet-based strategy, where each incoming IP-packet is managed individually. The second one is a frame-based approach, where a video frame is an inseparable unit during the buffer management decisions and actions [37].

Moreover, the buffer management mechanisms can be with or without data differentiation. In the first case, the buffer management decisions additionally depend

on received data type (i.e. frame), and its priority. Another aspect to be considered is the role of the buffer which can be reactive or proactive. The reactive approaches take place when a congestion situation has already occurred, on the contrary, the proactive approaches attempt avoiding the congestion in advance.

Most of today's network elements apply a simple reactive approach commonly known as drop-tail FIFO buffer management strategy, where new arriving data is dropped if the queue is full. In the case of video streaming, this strategy can be extended to a frame-based drop-tail buffer, which discard all IP-packets belonging to the same video frame if one of its IP-packets was dropped. Another reactive approach is the drop-head strategy. It consists in dropping the IP-packets which reside longest into the queue.

The problem with the mechanisms exposed above is that they decrease significantly the video quality. However, it can be solved by proactive approaches which drops packets before a queue becomes full, so end nodes can anticipate congestion before buffers overflow. Active Queue Management (AQM) approach is one of such mechanisms which have provided better control in the recent years. It works at the router for controlling the number of packets in the router's buffer by actively discarding an arriving packet.

According to the metrics used to measure congestion, AQM can be classified into three categories: queue-based, rate based, and schemes based on the combination of the two previous ones. In queue-based schemes, congestion is observed by average or instantaneous queue length, and the aim is to stabilize the queue length. The drawback of queue-based schemes is that an inherent delay is inserted. Rate-based schemes accurately predict the utilization of the link, determine congestion, and take actions based on the packet arrival rate. The goal of the rate-based AQM is to alleviate rate mismatch between enqueue and dequeue, and achieve low loss, low delay and high link utilization. Although these AQM schemes presents fats responsiveness and can obtain better performance than the queue-based approach, they are proned to experiment large queuing delays jitter. These delays may appear specially under dynamic network scenarios due to the lack of no control mechanism of queue size. The last category is composed by AQM schemes which combine the queue length and input rate to measure congestion and achieve a trade-off between queues stability and responsiveness. A deep explanation about the different AQM schemes can be find in [38]. From these AQM techniques, one of the most widely used is the Random Early Detection (RED) and its variants. Basically, RED drops packets probabilistically before the queue actually fills up.

Furthermore, blending proactive approaches with data differentiation permit to improve even more the quality received. For instance, the priority of frames can be assigned where I or P frames own higher priority than B frames. This is because if these frames are missed, other video frames (P or B) which depend on the lost frame can hardly be decoded properly.

Regarding the classification of buffer management methods described above, the next section addresses the most significant strategies presented in the literature.

5.1 Overview

Buffer management and scheduling algorithms represent the main class of techniques traditionally applied to maximize the performance in communication networks, ei-

ther wired or wireless. Video transmission makes no exception. Several works in the literature, present buffer management strategies applied to video communications that consider the specific features of video.

As an example, a study in [39] founds that, for the case of low bit rate coded sequences, higher quality low frame rate videos are mostly preferable to lower quality full frame rate ones. Following this consideration, together with the result from network analysis that outlined the fact that most of the losses occur at the transmission point and not at the nodes inside the network, [40] proposes a simple buffer management scheme implemented at the transmission source which drops low priority packets in response to congestion, where priority is related to frame rate. Hence, packets belonging to higher enhancement layers, i.e. those corresponding to higher frame rates, have the lowest priority.

In [40], the video sequences are encoded into multiple priority layers using H.26L video encoder and a robust scalable sub-band/wavelet video coder. The buffer management algorithm performs a greedy strategy on the packets being transmitted, trying to maximize the quality of the video which can be reconstructed from the transmitted packets. The contribution of each packet to the video quality is indicated by its priority layer. Basically, the algorithm constantly controls the buffer size. When the current size exceeds a defined threshold, packets are dropped considering their transmission probability. On the other hand, if the size exceeds the maximal allowed buffer size, any arriving packet is dropped without considering its priority.

The greedy assignment of probabilities starts from the least important layer, and then, it is reduced until the network rate constraint is satisfied. Moreover in [40], priority layers match the frame rate scalability layers, so this probability assignment facilitate the reception of higher quality video at the lower frame rates, rather than low quality video at the highest frame rate.

In [36], the buffer management and congestion control presented in [40] are applied to video streaming. Also in this case the buffer is implemented at the source and it queues packets from the encoder, and dequeued packets are transmitted using a randomized binomial scheme. This scheme provides a less bursty, TCP friendly transport strategy for the media transfer. Basically, randomization is shown to reduce: bias against flows with higher RTTs, window synchronization, phase effects in flows and correlated losses. The buffer management system is the same as the one presented in [40]. It differentiates the packets according to their priority and sends only the most important packets in the available bandwidth. However, differently from the previous work, the Round Trip Times (RTT) are used to estimate the drop probability of a packet.

An architecture which includes an input buffer at the server coupled with the congestion control scheme of TCP at the transport layer is presented in [41]. This work proposes a selective frame discarding which uses the H.263/H.263+ scalable video coding to produce frames of different importance. Also, it assumes that the available bandwidth is sufficient to deliver the high priority frames. Hence, the goal is to maximize the number of transported low priority frames subject to the constraint, that the loss rate for the high priority frames would be minimal. Since high priority frames which belong to the base layer should be delivered to the receiver without any loss, the drop policy presented in [41] is only applied to low priority frames which belong to enhancement layers. The discarding of the frames is governed by

the inequality: $D_{iS} \leq \alpha\, T_p$, where D_{iS} is the delay between the new frame arriving i and the buffer head-end. This delay is called shaping delay; α is the parameter which controls the selective discard algorithm and it could take values in the interval (0,1); T_p is the initial buffering time of the playout buffer. T_p should be chosen greater than the maximum network jitter. Therefore, a frame is discarded depending on its shaping delay. Hence, the presentation interruptions are reduced.

An integrated video communication scheme for stored variable bit-rate (VBR) video streaming in a congested network is presented in [42]. With the purpose of reducing the client buffer requirement, this scheme regulates the transmission rate through a refined rate control algorithm based on the Program Clock References (PCR) value embedded in the video streams. Furthermore, multiple buffers for different importance levels or priorities are applied at the source. The definition of priority is very flexible. For instance, in a non scalable video, the different queues can be created considering the type of frame (i.e. I, P, and B frame). Thus, 3 queues would be employed, where the queue for I frames has the highest priority. On the contrary, for layered video, different layers can directly be used as importance levels.

The frames with different priorities wait in the corresponding queue until they reach the head-end of the buffer, and a decision is then taken by an intelligent selective frame discard (SFD) algorithm. First, the queue size of each queue is calculated, then if a queue exceeds an established threshold the discarded strategy is applied depending on its priority. Hence, if the queue has low priority, the older frames will be discarded to reduce the queue occupancy until it is less than threshold. This operation permits to put the newer frames into the queue without delay. Nevertheless, for the queue with the highest priority, this kind of discard should be avoided because little lost of key frames will lead to a severe degradation of the presentation quality at the client side. Accordingly, the SFD algorithm is employed in these cases. It calculates the delay for a frame before it is decoded and played at the client. This delay is then compared with its playback deadline. So, if the delay is greater than the deadline, the frame is dropped. Otherwise, the frame is enqueued. Furthermore, the algorithm presented in [42] is compared with three previous approaches, producing significant improvements in required transmission rate and buffer size, especially when HDTV traces are transmitted.

With the purpose of enhancing the objective video quality of a streaming application in a High Speed Downlink Packet Access (HSDPA) network, different reactive and proactive queue management schemes are discussed in [37]. A proactive buffer management scheme with data differentiation is proposed, which significantly increases the video quality by taking into account MPEG frame dependencies. Fundamentally, if a congestion situation is inevitable, and the buffer occupancy exceeds a defined threshold, [37] employs a frame-based scheme called *proactive B-dropping*. This technique drops newly arriving data units belonging to B-frame (B-packets), instead of data belonging to I or P frames (I or P packets). Moreover, a timer is employed to remove the packets which have been in the queue for a long time. In this way the transmission of obsolete data is avoided.

It is important to notice that the *proactive B-dropping* is a drop-tail based implementation. Packet-based and frame-based schemes with proactive B-dropping are compared, considering both infinite and finite play-out buffer. From experiments, it is deduced that frame-based approaches are better than packet-based approaches because of inter-frame dependencies. They avoid the transmission of data which is

not more usable at the receiver. On the other hand, to achieve good results with the proactive B-dropping scheme, is fundamental that the encoded video presents a considerable amount of B-frames. A 8B-coding video was found to deliver a good performance because more data can be dropped in order to avoid I-frame losses. The drawback of this approach is that the requested coding schemes, as 8B-coding, utilize longer encoding times because of the frame dependencies, which can have a negative impact on interactive applications.

Differently from the previous works, [37] takes advantage of H.264/SVC and presents a new active queue management algorithm based on priority dropping and proportional-integral-differential (PID) controller, called *PID_PD*. PID computes the dropping rate of every arriving packet according to the variance of router queue length. In order to ensuring the quality of decoded video, the *PID_PD* algorithm provides more protection to high important layers. For doing this, a priority matrix is built. This matrix contains m layers, and every layer has n priority ranks. First, when video data is packetized in application layer, a priority number is assigned to each packet. Hence, the router maintains a $m \times n$ priority queues, and every priority queue stores the packets which have the same priority number. These queues are updated at each entry or departure of packets. When a new packet arrives, firstly the dropping probability is calculated, then if the current packet is determined to drop, the *PID_PD* strategy looks for the packet whose priority number is less than the current packet in the priority queues. If a lower priority packet is found, then this last one is discarded and the current packet enters into the corresponding queue. Finally, all the queues are updated. Otherwise, the current packet is dropped. From experiments, the algorithm proposed in [43] prevents dropping high priority layers or frames, and thus providing high video quality to the end users.

A Weighted Multi-Playback Buffer Management (Weighted MPBuff) is proposed in [44]. It should be noted that the receiver can not decode a frame if the base layer for that frame is discarded or fails to reach the receiver before the decoding deadline. Therefore, the Weighted MPBuff and scheduling algorithm provides more protection to the lower layers compared to the higher ones. In order to provide weighted protection, the sender schedules the video data considering the transmission sequence, and sending different number of GOPs in each slot to build up an unequal buffer at the receiver. The playout buffer is composed by as many buffers as coded layers, and the buffer size decreases from the base layer to the highest enhancement layer. Each time period, the receiver calculates the video buffer time using the highest timestamp and the timestamp of the current playback time. This information is used to decide whether the video data unit in layer i+1 will be scheduled or not. If the current available bandwidth is not enough to transmit the layer i+1, clearly also the layers above it will not be sent.

The approach presented in [44] uses H.264/SVC, timestamps, and it is conducted in a rapidly fluctuating bandwidth using Network Simulator (NS3). It is compared with a SVC equal multi-playback buffer scheme, achieving a better performance. In general, the weighted multi-playback buffer is more tolerant to bandwidth fluctuation and it gives a better level of continuity presenting a lower percentage of freeze frames and so, attaining a higher average PSNR. The drawback of the Weighted MPBuff algorithm is that in case of congestion the whole layer is discarded.

Moreover, regarding a multi-stream video transmission scenario, [45] proposes a perceptual quality-aware active queue management (AQM), which is designed for

scalable video traffic. In order to reduce the queuing delay and the queue length, the proposed scheme selectively drops packets from layers that have little influence on video quality, introducing a minimal perceptual quality reduction (PQR) in the stream. Similar to [44], the dropping strategy regards the fact that losses in base quality layers cause considerably higher quality reduction than losses in quality enhancement layers.

The system considered by [45] consists of multiple pre-encoded H.264/SVC video streams, where each quality layer in each temporal layer is defined as a scalable unit (SU). SUs from K streams of video enter the scheduler where for each stream, multiple virtual queues (VQs) are created dynamically to queue SUs from different temporal and quality layers. Despite of the loss of SUs in each of the VQs, results in a decrease in the congestion and waiting time in the queue, this causes a certain decrement in the perceptual quality of the reconstructed stream. To face this issue, [45] applies a perceptual quality metric model to calculate the PQR caused by data drops at each video layer. In addition, the quality-aware AQM is compared with the random early detection (RED) and tail drop (TD) schemes, achieving a better performance.

Finally, several works have exploited scalability in wireless environments mixing with adaptive modulation and coding (AMC) and resource allocation. As an example, [46] proposes a dynamic sharing of the resources by combining SVC with appropriate radio link buffer management for multiuser streaming services. Since the pictures belonging to the lowest temporal layer are most important for decoding, these own the highest priority. Then, the respective progressive refinement (PR) fragments have next lower importance, and so on. Hence, by assuming that the temporal resolution has highest priority, this work recommends a dropping order, based on the temporal level of the fragments, in case that a rate adaptation on the fly is necessary. First, the PR fragments of the present highest temporal level are dropped. If there are only packets containing base layer fragments, the base layer of the highest temporal level present is discarded. Furthermore, following the same aforementioned drop concept, this work develops a priority labeling technique, which is applied in the radio link buffer.

In this thesis, we present a proactive packet-based buffer management strategy with a selective packet discard technique for H.264/SVC-MGS video sequences. Differently from the previous works, where a frame-based discard strategy is applied, our approach employs a finer discarding technique applied at packet level and not over the entire frame. On the other hand, several approaches in the literature propose packet discard strategies where the priority of the packets is assigned considering only temporal level, or packets are discarded based on dropping probabilities. Opposite to this, in our strategy, the packets are discarded taking into account both, quality and temporal level, as defined in the MGS-SVC hierarchical dependency between layers. Hence, low priority packets are dropped, and thus sending the most important packets fitting the available bandwidth. The proposed QDP algorithm guarantees that the transmitted video has the highest possible quality under the given network conditions and buffer resources. In the next section the QDP algorithm is described in detail.

5.2 Quality Discard Packets algorithm

In a server-client adaptive streaming environment, a client makes request for multimedia objects to a centralized video server. The server sends the packets sequentially, and usually selects the content from a set of files where the content is encoded at different rate to match the available bandwidth. The switch from one chunk to another is based on the bandwidth estimation obtained by some feedback mechanism. These can be exploited by proprietary or standard protocols such as DASH [47]. This part is transparent to our simulation since we only assume that some information about RTT is received, and thus the HMM-BE model presented in Chapter 4 can adapt the estimated bandwidth periodically. In the present work we assume this occurs every second. All encoded packets are sent to the transmission buffer. Buffer fullness obviously depends on the difference between encoding rate and available bandwidth. Losses can be reduced with larger buffers at the expenses of higher delays. We show that the use of scalable coded sequences in combination with the proposed selective discard procedure, allows reducing losses while maintaining minimum delays.

With the development of scalable video coding schemes, the mechanisms proposed in the literature exploit scalabiltiy assigning different priorities according with the relevance of the layer. In SVC, the upper layers depend upon the lower layers. In such way, if a lower layer is missing, it will not be possible to decode the above layers. Several works in the literature use H.264/SVC jointly with AQM to attain better results. Particulary, [44] encodes the video streams using H.264/SVC-MGS. It further employs multiple unequal size buffers to provide higher protection to lower layers. However, when congestion is detected, all the layers above the current one are discarded.

Applying a selective discarding strategy, this thesis investigates the influence on video quality that each packet owns individually. So, we propose a Quality Discard Packets (QDP) policy which regards the priority of the packets within the encoded scalable video stream. Hence, the base layer packets have the highest priority, and the packets belonging to the highest enhancement layer have the lowest priority. It is relevant to mention that our QDP strategy besides considering the quality level of the packets also it considers their temporal level. In that way, only the packets with the highest quality and highest frame rate available into the buffer are discarded in order to alleviate it. Therefore, avoiding unnecessary packets discard and thus quality degradation.

Specifically, we take advantage of the quality scalability provided by H.264/SVC-MGS. This type of scalability refers to scaling in terms of the level of compression applied to the source video. With quality scalability, the base layer contains a strongly compressed version of each picture, and enhancement layers incorporate more information to increase the SNR (Signal-Noise-Ratio) value [48]. Moreover, H.264/SVC supports combined scalability, for instance the combination of quality and temporal scalability. Figure 5.1 illustrates a SVC stream encoded with four temporal levels { T_0, T_1, T_2, and T_3 } and two quality levels { Q_0 and Q_1 }. This encoded video stream exploits the hierarchical prediction structure using B-pictures for enabling temporal scalability, and the key picture concept to attain quality scalability.

The *Quality Discard Packets* policy is developed considering the scenario shown

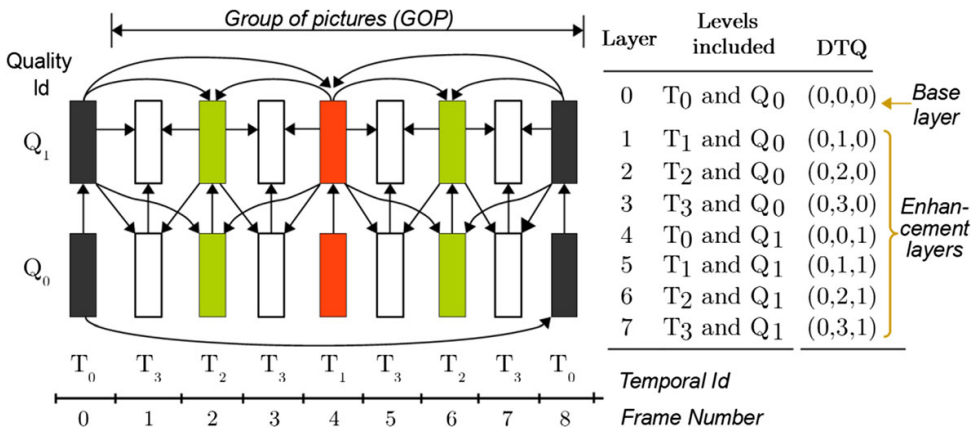


Figure 5.1: Example of coding structure of a SVC stream with temporal and quality scalability [48].

in Figure 5.2. When a H.264/SVC video is encoded, the encoder also generates a trace file which specifies various parameters for each single "packet" inside the bit-stream. The parameters include the start position (in units of bytes) of the packet, its length (in units of bytes), its values of dependency_id (Lid), its temporal level (Tid), and quality level (Qid), its type, and two flags indicating whether the packet is discardable or truncatable [9]. Our QDP strategy uses the information corresponding to Tid and Qid specified in the trace file. These values are represented in the DTQ field in Figure 5.1.

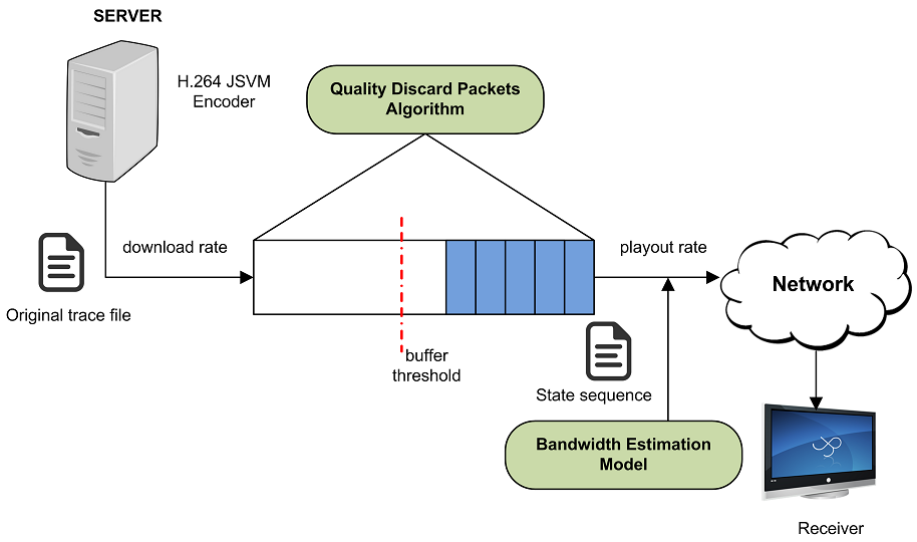


Figure 5.2: Quality Discard Packets scenario.

We set up the buffer size and define a threshold lower than the total buffer size. The choice of the source buffer threshold is crucial for the overall system performance. Defining a small threshold will lead to unnecessary packet drops at the source buffer, while having a large threshold will increase the overall delay and eventually cause the receiver buffer underflow. Regarding this, as *threshold* increases from 0 to ∞ , the loss at the source decreases, while the loss at the receiver increases. Accordingly, the optimal setting for threshold would balance these two types of loss

to achieve the overall minimum [49]. In this work, we consider a threshold equal to 50% of the maximum buffer size. So, when this threshold is attained, we assume the buffer is close to the risk of overflowing and the QDP policy must be applied.

The discarding packets process is governed by Algorithm 1, where q_{size} is the instantaneous buffer size, *MaxBufSize* is the total buffer size and *BufThr* is the buffer threshold indicating the buffer size at which the discard policy starts being enforced. If the queue size exceeds the threshold, the packets belonging to the highest layers, which own the highest quality, are discarded first than the other ones. As mentioned before, video encoded with SVC presents different temporal levels (Tid), exhibiting a hierarchical dependency. So, the packets with the highest Tid are discarded first.

On the other hand, when the queue size exceeds the imposed maximum buffer size, the arriving packet is dropped without considering the type of the data. The buffer works like a FIFO queue. At the end, the video received is decoded using the decoder tool included in JSVM and FFMPEG [50]. Since the last one conceals whole frame losses using temporal frame interpolation, a lost P frame is concealed by copying the pixels from the previous reference frame, and a lost B frame is concealed by temporal interpolation between the frame pixels of the previous and the future frames [51].

Algorithm 1: Quality Discard Packets

```
1  begin
2      for Each Packet ∈ BitStream do
3          calculate the queue size q_size
4          if (q_size + PacketSize) ≤ MaxBufSize then
5              enqueue packet
6              if q_size ≥ BufThr then
7                  discard packets with Highest Qid and Highest
                    Tid
8              end
9          end
10         else
11             drop packet
12         end
13     end
14 end
```

5.3 Simulations Results

To evaluate the performance of the *Quality Discard Packets* algorithm, several *cif* video sequences with a duration of one minute and different characteristics, were

encoded in H.264/SVC with MGS. Table 5.1 presents the main encoding parameters used.

Table 5.1: Encoding parameters.

Parameter	**Value**
FrameRate	30
FramesToBeEncoded	1800
CgsSnrRefinement	1
MGSControl	2
GOPSize	8
BaseLayerMode	2
NumLayers	4

The video sequences are encoded using the *FixedQPEncoderStatic* tool included in the JSVM software. This tool provides two options:

- Encode the video fixing the Quantization Parameter value (QP) to each layer;
- Set the target bit rate that will be reached by each layer.

Here, the second option is used because it allows a better match with the variations of the bandwidth conditions over the network. In this section, we present the obtained results using three different video sequences, whose encoding bit-rates are shown in Table 5.2. The two first video sequences present high spatial details and high/medium amount of movement. On the other hand, the last video sequence (Silent) corresponds to a simple sequence with low amount of movement.

Table 5.2: Encoding bit rates.

Video seq	**Layer**	**Bit-rate (Kbit/s)**
Football Ice	BL	200
	EL1	400
	EL2	600
	EL3	800
Silent	BL	150
	EL1	200
	EL2	400
	EL3	500

Moreover, each enhancement layer defines two MGSVectors, thus allowing videos to be encoded with seven target points.

Using the packet size information provided by the trace file, the next step is to calculate the times when the packets arrive at the buffer from the server, as well as the times in which the packets will leave the buffer. The first one, is obtained dividing the size by the maximum bit-rate at which the video was encoded. The departure time depends on the *playout rate*, which varies each second of time. The values of the *playout rate* are obtained using the Bandwidth Estimation Model presented in

the previous chapter. Figure 5.3 shows an example of the bandwidth fluctuations, obtained with the HMM-BE model, during 60 seconds representing the duration of the *Football* video sequences. Since it has been shown that Quality of Experience (QoE) from a user's perspective is worse when the quality is reduced along the sequence [52], we decide starting with a conservative approach about bandwidth availability.

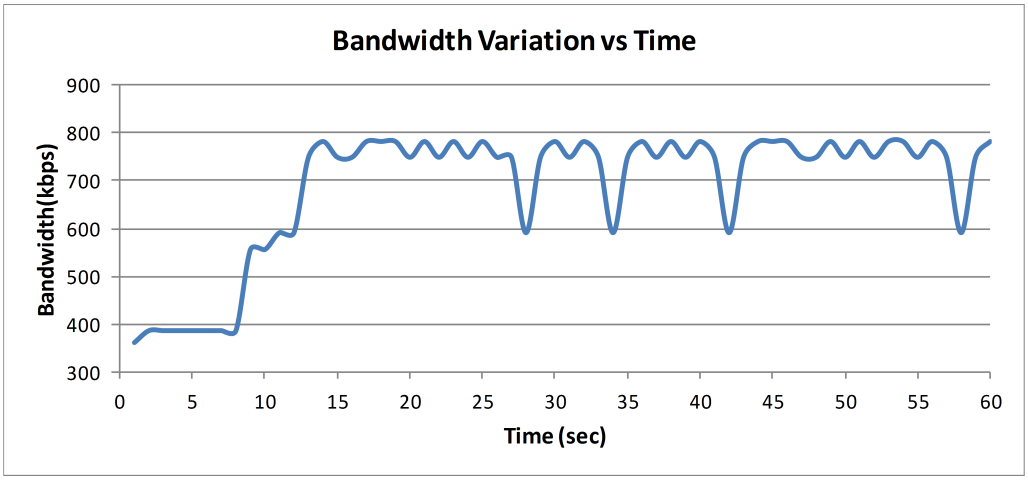


Figure 5.3: Bandwidth obtained by the HMM-BE model.

The performance of the *Quality Discard Packets* algorithm is compared with the widely deployed Tail Drop (TD) mechanism, a pure buffer management strategy unaware of the specific content, the Random Early Detection (RED), and a layer discarding mechanism denominated Simple Discard Quality Layer (SDQL). The TD technique drops arrival packets upon the occurrence of queue overflow. On the contrary, RED calculates the average queue size and then, this value is compared to two thresholds: a minimum Min_{th} and a maximum Max_{th} threshold. When the average queue size is less than Min_{th}, no packets are discarded. When the average queue is greater than Max_{th}, every arriving packet is discarded. When the average queue size is between the Min_{th} and Max_{th} thresholds, each arriving packet is discarded with a probability p_a, where p_a is a function of the average queue size [53]. It is worth highlighting that due to the different scenarios, simulation tools and specific conditions used, it is not possible straightforwardly to compare our approach with the algorithms and strategies proposed in the literature (Section 5.1). However, it has been seen that the general solution applied in most works of interest consists in discarding all the packets belonging to the highest available layer. Therefore, this solution is executed by the SDQL algorithm.

According to [53], the maximum threshold should be set to at least twice the minimum one, or three times Min_{th} following the rule of thumb [54]. However, a higher maximum threshold could be used in order to get a better use of buffer space and reduce the packet drops. Therefore, the RED implementation used in this work, employs a minimum and a maximum buffer threshold equivalent to the 50% and 80% of the total buffer size, respectively. It is important to mention that the minimum threshold was set up to 50 % to compare its performance with our QDP algorithm which takes place when a threshold equivalent to the half of the buffer size is attained. Furthermore, according to [55], the RED performance is

highly dependent on the thresholds and sensitive to its parameter settings. The optimization of these parameters is out of the scope of this thesis. The results of the video sequences presented in Table 5.2 are analyzed below.

a) Football

Tail Drop (TD)			
Buffer Size	PSNR(dB)	discarded packets	Decoded frames
3MB	31.44	*Buffer threshold is never attained	1800
1 MB	21.15	479	1715
500 KB	26.64	13370	298
250 KB	26.91	14607	165
100 KB	22.70	15625	56
50 KB	21.01	15915	25
25 KB	11.63	16018	17
10 KB	11.78	16039	7

(a) No Buffer

Random Early Detection (RED)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	probability	Decoded frames
3 MB	31.44	*Buffer threshold is never attained		1800
1 MB	27.24	0	7	1800
500 KB	17.43	0	493	1738
250 KB	17.08	0	316	1761
100 KB	25.15	15816	0	42
50 KB	24.03	16016	0	21
25 KB	24.16	16137	0	8
10 KB	27.59	16186	0	1

(b) RED

Simple Discard Quality Layers (SDQL)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	Algorithm	Decoded frames
3 MB	31.44	*Buffer threshold is never attained		1800
1 MB	27.24	0	4411	1800
500 KB	30.18	0	4385	1800
250 KB	30.10	0	4173	1800
100 KB	27.24	0	2571	1800
50 KB	27.24	0	2175	1800
25 KB	27.24	0	1692	1800
10 KB	27.24	43	1527	1800

(c) SDQL

Quality Discard Packets (QDP)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	Algorithm	Decoded frames
3 MB	31.44	*Buffer threshold is never attained		1800
1 MB	31.08	0	1666	1800
500 KB	30.78	0	2253	1800
250 KB	30.63	0	2560	1800
100 KB	30.63	0	2420	1800
50 KB	27.24	0	2173	1800
25 KB	27.24	0	2039	1800
10 KB	27.24	37	1841	1800

(d) QDP

Figure 5.4: Football simulations results.

Figure 5.4 presents the results obtained applying the four approaches in the football video sequence. It is important to point out that the total number of frames in the original sequence is equal to 1800 frames, corresponding to one minute of video duration, and that the original trace file is composed by 16209 packets. Moreover, the PSNR in Figure 5.4 corresponds to the PSNR attained for the decoded frames. For instance, when TD is used with a buffer size equal to 10 KB, just 7 frames are reconstructed by the decoder. The achieved PSNR is the result of comparing these 7 frames with the corresponding ones in the original video sequence. This PSNR does not consider the 1793 frames that were missed, because in that case the PSNR would be close to zero.

We can notice that in TD and RED cases, the buffer size must be greater or equal to 3 MB and 1 MB, respectively to get a good PSNR value and recover all the frames. However, when the QDP strategy is used, it is possible to obtain a good PSNR value and decode all the frames, when a buffer size greater or equal than 100 KB is used. This is due to the priority discard packets policy. Considering the fact that the current version of JSVM cannot decode video streams affected by out order, corrupted or missing NALUs, we decided in these cases to employ another decoder, FFMPEG, which uses error concealment techniques to reconstruct the original video sequence. In the tables illustrated in Figure 5.4, blue colored results were decoded using the JSVM decoder and the green ones were decoded by FFMPEG.

As we can see, in Figure 5.4 (d), if FFMPEG is used for the decoding process and the *QDP* strategy is applied, it is possible to attain an acceptable PSNR value recovering all the frames, even with a constrained buffer size. As showed in Figure

5.4 (d), a buffer size of 10 KB produces a PSNR equal to 27.24 dB, decoding all the frames. It is possible because limiting the number of lost packets and applying the concealment and recovery abilities of FFMPEG decoder much better results can be attained.

In the first case, when the maximum buffer size is equal to 3MB, the *QDP* algorithm is not even employed because the threshold, which is established to be the 50% of the total buffer size, is not attained. The same situation occurs in all the other analyzed approaches. In contrast, RED takes place when the buffer size is less or equal than 1 MB. As is shown in Figure 5.4 (b), seven packets are discarded by the algorithm. The loss of these packets does not permit the use of JSVM decoder, but employing FFMPEG all the 1800 frames can be reconstructed.

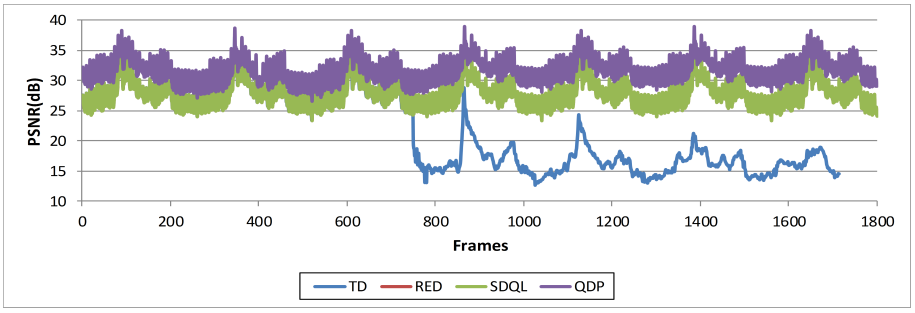
Furthermore, we can see that even when the buffer size decreases, our *Quality Discard Packets* approach overcomes the other ones, recovering all the frames as the original video sequence and getting a good PSNR. A PSNR value is considered good when it is greater than 30 dB. Moreover, when the buffer size is greater than 100 KB, the one hundred percent of the frames are recovered, attaining a PSNR equal or greater to 30.63 dB. On the other hand, to buffer sizes between 50 KB and 10 KB, all the frames are also rebuilt by FFMPEG at the same time that a suitable PSNR (27.24 dB) is obtained.

It is important to mention that when a packet which is signaled as a non discardable packet in the original trace file, is discarded some distortion is inserted in the belonging frame. Moreover, as we can see in Figure 5.4 (b), when RED algorithm is applied to buffer sizes less than 1MB, the 100 percent of the decoded frames is never attained, due to its random packet discard. This prevents further that FFMPEG reconstructs the sequence with the available packets. Consequently, the PSNR is drastically declined.

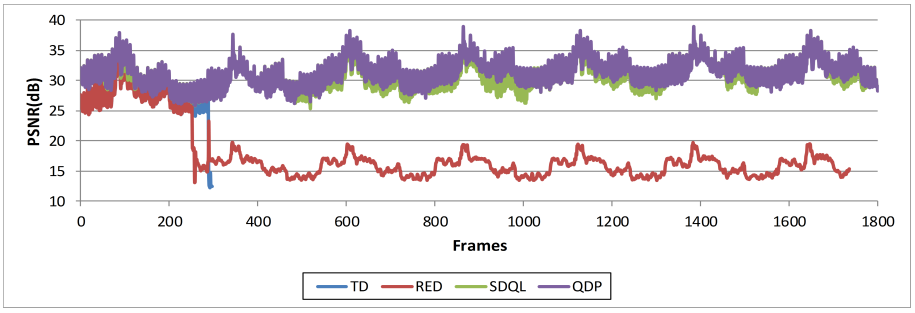
The PSNR attained by SDQL algorithm, in most of the cases, is good and minimally below to the PSNR produced by our QDP strategy. The great difference between these two mechanisms is the decoder used. Due to the complete layer discarding strategy applied by the SDQL algorithm, the use of JSVM decoder is restricted. JSVM is not able to decode the stream if dependent packets have been discarded. Therefore, the use of FFMPEG is imminent required, thus reducing the final quality.

Then, Figure 5.5 presents the PSNRs achieved by each frame of the video sequence. The PSNR of the four approaches is compared considering different buffer sizes. As can be seen in Figure 5.5 (a), when a buffer size of 1 MB is used, all the 1800 frames are recovered by QDP, RED and SDQL. The last two approaches, RED and SDQL, produce the same PSNR and they are overlapped in the graph. On the other hand, TD mechanism reconstructs 1715 frames and this number decreases as the buffer is smaller. The same behavior is generated by RED. For instance, when the buffer size is constrained to 50KB, Figure 5.5 (d), only 1.4 % and 1.16 % of the total number of frames is rebuilt by TD and RED respectively.

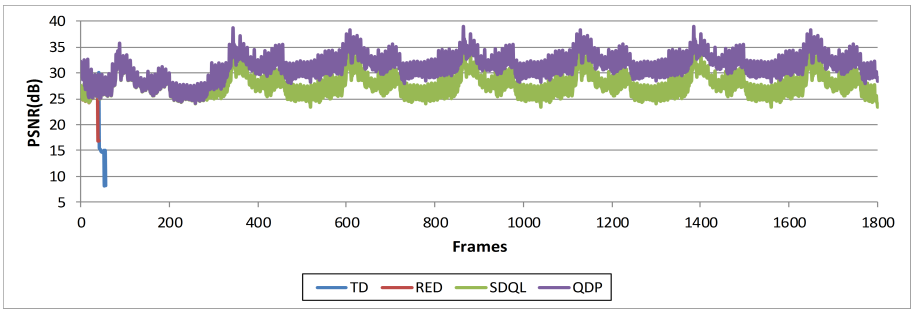
It is worth noting that for all the cases presented in Figure 5.5, the QDP strategy is able to reconstruct all the frames as the original sequence and produce a adequate PSNR. The PSNR achieved by our approach overcomes the other approaches except when the buffer is equal to 50 KB, where the PSNR produced by SDQL and QDP algorithm is identical, so they are overlapped in the figure.



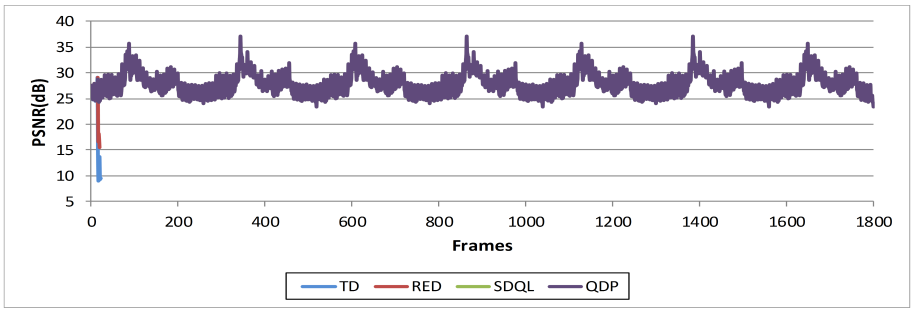
(a) 1MB



(b) 500KB

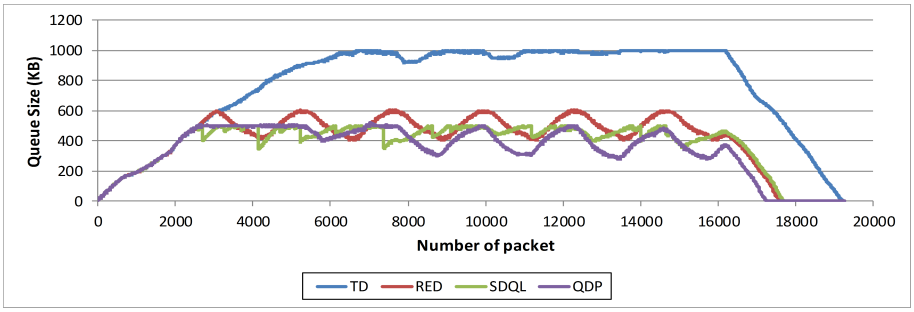


(c) 100KB

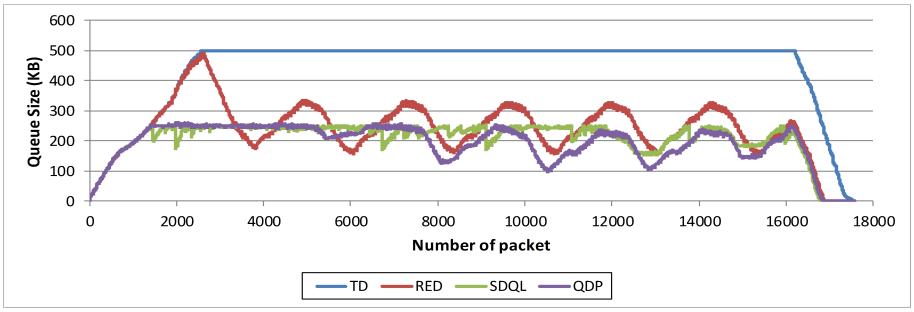


(d) 50KB

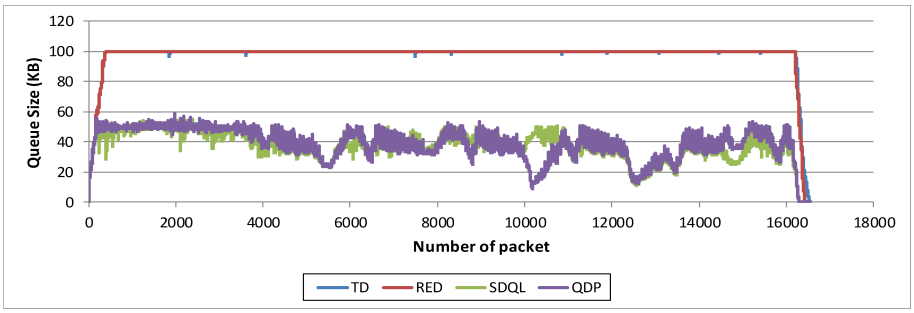
Figure 5.5: PSNR vs. Frames to different buffer sizes (Football).



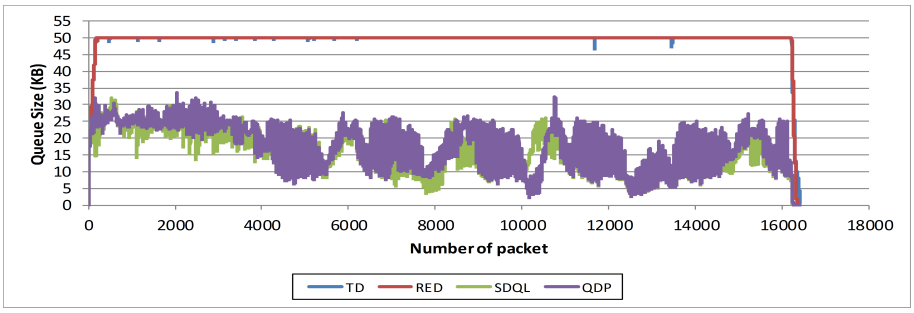
(a) 1MB



(b) 500KB



(c) 100KB



(d) 50KB

Figure 5.6: Buffer occupancy to different buffer sizes (Football).

In order to analyze the variation of buffer occupancy, the queue size is measured each time that a packet is read from the trace file. Depending on the current queue size, the packet will be added or not to the buffer. Figure 5.6 illustrates the buffer occupancy, to all the four analyzed approaches, when the buffer is set to four different buffer sizes.

It is important to remind that the total number of packets in the bit-stream is equal to 16209 packets. Initially, due to the bit-rate with which the packets leave the buffer is less than the bit-rate used to arrive into it, the buffer occupancy is continually increased. Once the queue size has attained the defined buffer threshold, the buffer occupancy fluctuates depending of the number of discarded packets by the QDP strategy and the departure times of the packets. The buffer just starts decreasing when all the packets of the video have been enqueued and it stops of receiving packets and just releases them.

As is shown in Figure 5.6, when TD is employed, the buffer occupancy grows until its fullness. This state is kept during all the enqueue process, then the buffer starts to be emptied. On the other hand, in the first two cases, Figure 5.6 (a) and (b), RED and QDP produce a similar buffer occupancy. But unlike to QDP, when the buffer is equal to 500 KB, RED does not reconstruct the video sequence as the original one. To small buffer sizes, the RED algorithm behaves like TD, achieving rapidly saturation. Contrary to RED, which discards packets randomly, QDP discards packets regarding its priority and influence on the other ones. Moreover, the buffer occupancy produced by the SDQL algorithm in several cases is pretty similar to the presented by the QDP strategy.

Moreover, we have to stress the fact that due to the "intelligent" discard packets used in our QDP approach, the buffer is never saturated, it means that any packet is dropped because the buffer is full. Hence, the possibility of losing some essential packet, which could be indispensable to the video sequence reconstruction, is reduced to zero. This does not occur with RED, where at certain time the maximum buffer size is attained (buffer sizes less or equal than 100 KB) and the upcoming packets have to be dropped.

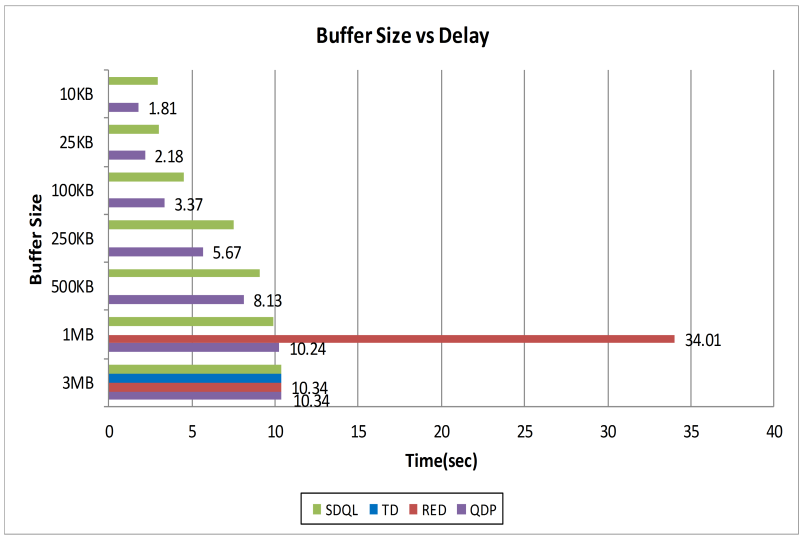


Figure 5.7: Buffer Size vs. Delay (Football).

As we can see in Figure 5.6 (a) to (d), the buffer occupancy to the QDP approach is reduced significantly, corresponding to small playout delays. In Figure 5.7, the average delay obtained with some of the buffer sizes used to test our algorithm are presented. As aforementioned, to 3 MB buffer size, the QDP strategy is not even applied. However, when our approach takes place, the delay is reduced considerably.

We only compare the delay time obtained by the four approaches when the buffer size is equal to 3 MB, because only in this case all the approaches are able to recover all the 1800 frames. Obviously in all the cases, the delay is the same. Moreover, to the remaining buffer sizes, our approach is compared with the SDQL approach. As it is possible to see in the figure, the delay produced by our QDP strategy is lower than the obtained by the other approach. When the buffer size is 1 MB, our approach is also compared with RED. However, the delay produced by RED is more than three times greater than the one obtained with the QDP algorithm. Moreover, combining FFMPEG and the QDP strategy, like in the cases of a buffer size equal to 25 KB and 10 KB (Figure 5.7), the average delay is decreased to 2.18 and 1.81 seconds respectively.

b) Ice

Tail Drop (TD)			
Buffer Size	PSNR(dB)	discarded packets	Decoded frames
3MB	41.9178	*Buffer threshold is never attained	1800
1 MB	35.5312	10509	606
500 KB	35.4548	13661	272
250 KB	34.8671	14785	152
100 KB	33.8587	15658	47
50 KB	30.6201	15862	23
25 KB	11.435	16094	11

(a) No Buffer

Random Early Detection (RED)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	probability	Decoded frames
3 MB	41.9178	*Buffer threshold is never attained		1800
1 MB	22.5838	0	48	1797
500 KB	18.0295	0	72	1790
250 KB	15.818	0	310	1761
100 KB	34.8827	0	15811	44
50 KB	32.8514	0	16034	18
25 KB	28.6981	0	16158	5

(b) RED

Simple Discard Quality Layers (SDQL)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	Algorithm	Decoded frames
3 MB	41.9178	*Buffer threshold is never attained		1800
1 MB	40.136	0	4377	1800
500 KB	36.0276	0	4306	1800
250 KB	39.8868	0	4448	1800
100 KB	36.0276	0	2858	1800
50 KB	36.0276	0	2312	1800
25 KB	36.0276	0	1900	1800

(c) SDQL

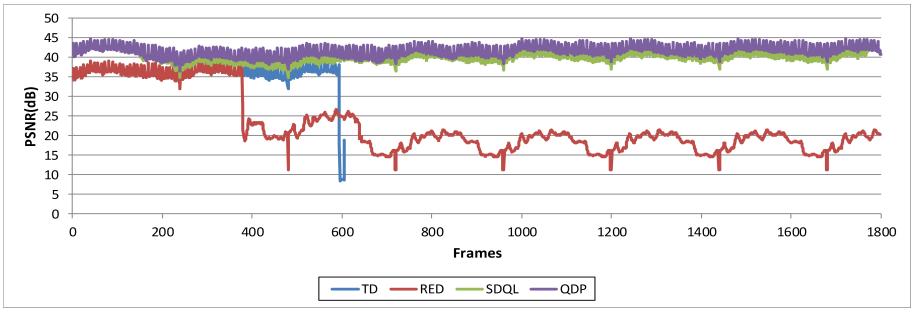
Quality Discard Packets (QDP)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	Algorithm	Decoded frames
3 MB	41.9178	*Buffer threshold is never attained		1800
1 MB	41.2593	0	1807	1800
500 KB	40.8964	0	2507	1800
250 KB	40.8022	0	2593	1800
100 KB	40.6834	0	2586	1800
50 KB	40.7618	0	2413	1800
25 KB	36.0276	0	2174	1800

(d) QDP

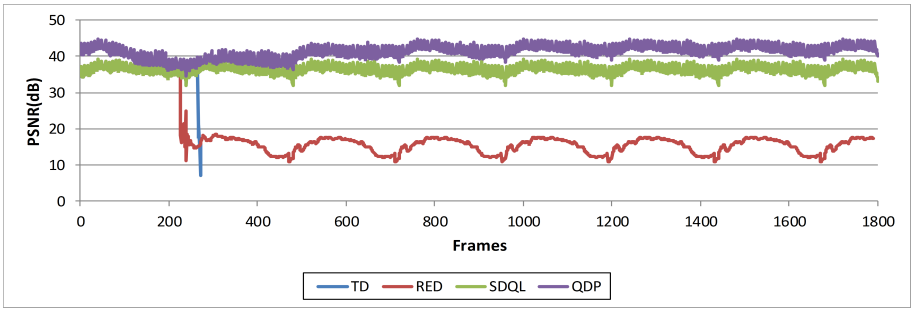
Figure 5.8: Ice simulations results.

Following the same procedure, the four approaches (TD, RED, SDQL and QDP) are applied to the Ice video sequence. This video presents similar features to the previous one, and it has a duration of 1 minute, corresponding to 1800 frames. Figure 5.8 shows the obtained results from the simulations.

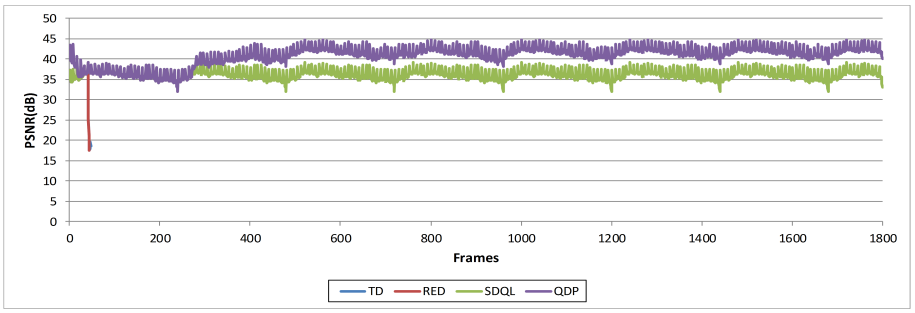
As we can see in the figure below, while the QDP algorithm produces a high PSNR even when constrained buffer sizes are employed, the other two approaches, RED and TD require buffer sizes greater or equal to 3 MB. It is important to notice that SDQL algorithm also achieves a good PSNR, however not as high as the obtained applying our strategy. The Figure 5.9 compares the PSNR achieved by each frame of the video sequence by the four approaches considering four different buffer sizes.



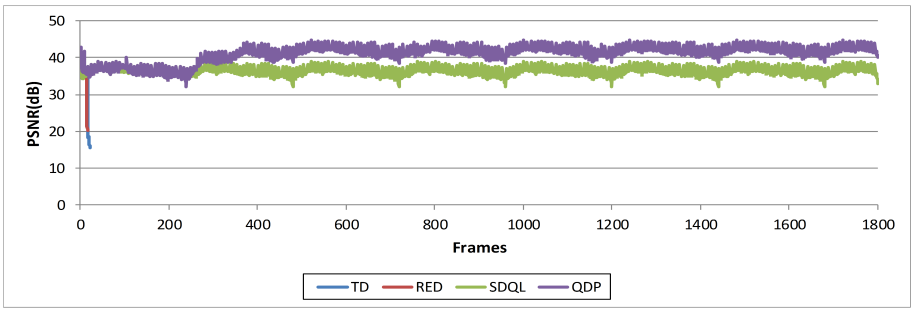
(a) 1MB



(b) 500KB

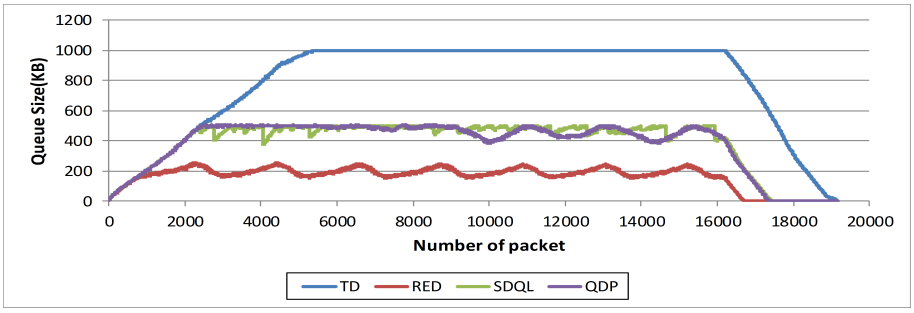


(c) 100KB

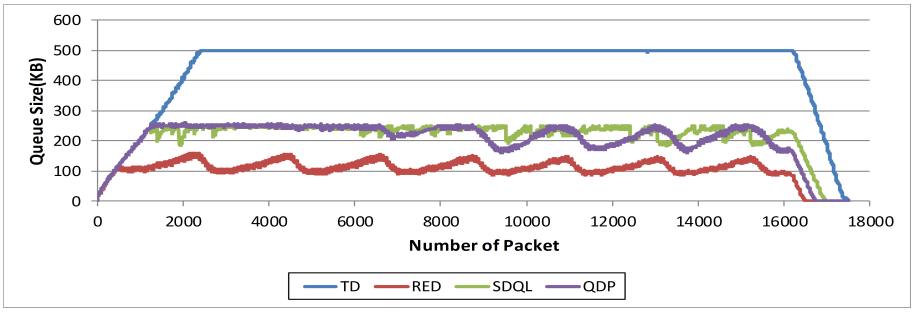


(d) 50KB

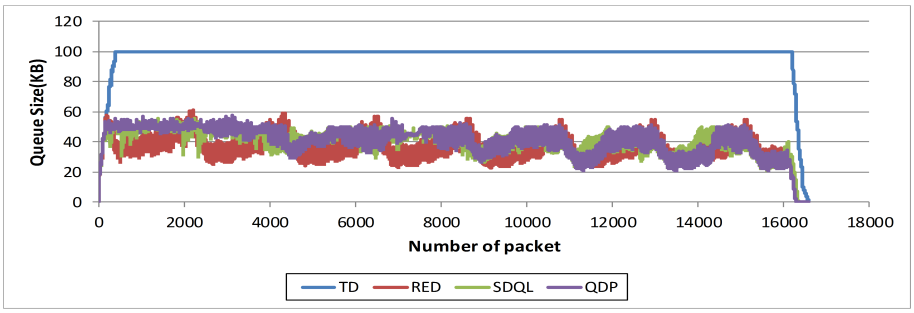
Figure 5.9: PSNR vs. Frames to different buffer sizes (Ice).



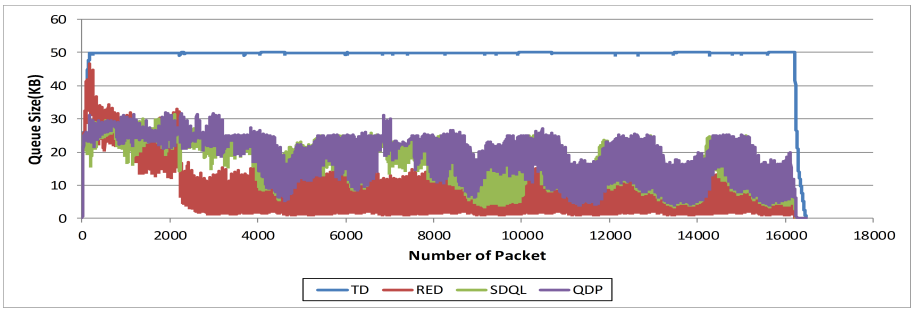
(a) 1MB



(b) 500KB



(c) 100KB



(d) 50KB

Figure 5.10: Buffer occupancy to different buffer sizes (Ice).

As shown in Figure 5.9, when QDP is used, the PSNR is kept high during all the video. In general, the PSNR attained is over 40 dB. Conversely, when RED is applied, the PSNR is drastically decreased. It is because some important packets are discarded by the algorithm. Consequently, even using the FFMPEG decoder, it is not possible to reconstruct all the frames as the original video sequence. On the

other hand, if TD is employed, the buffer size must be necessarily greater or equal to 3 MB. In Figure 5.9 (a), where the buffer size is 1 MB, more than a half (66.3%) of the total number of frames is missed. In the most extreme case which is 25 KB, if QDP and FFMPEG are used jointly, it is feasible to recover all the frames at the same time that a good PSNR is obtained. This does not occur with none of the other two approaches, RED and TD.

Furthermore, Figure 5.10 depicts the buffer occupancy to all the evaluated approaches considering four different buffer sizes. As expected, when TD is used, the queue size grows until to attain the buffer fullness, and it is relieved only when all the packets have been read. On the other hand, RED, SDQL and QDP perform a similar buffer occupancy, and in some cases, QDP is slightly higher than RED. The great difference is that in contrast to QDP, RED does not get to recover all the frames.

In Figure 5.11, the average delay obtained for several different buffer sizes is illustrated. For a buffer size equal to 3 MB, the average delay time produced by QDP, SDQL, RED and TD is the same, 11.87 seconds. It is because to a buffer size equal to 3 MB the threshold is not attained. As we can see in the Figure 5.11, the delay decreases as the buffer size is reduced. For instance, to constrained buffer sizes like 25 KB, is produced a delay equivalent to 2.53 seconds. In spite of SDQL algorithm presents a good performance, the final delay generated is higher than the one obtained by the QDP algorithm.

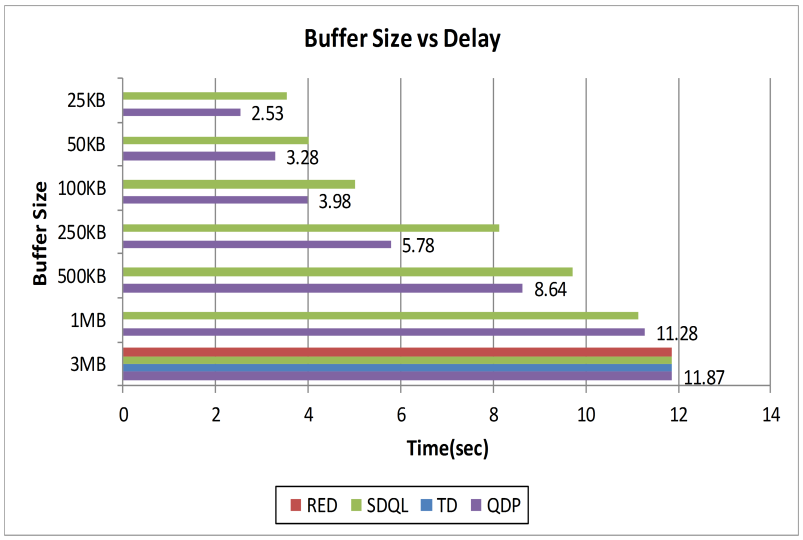


Figure 5.11: Buffer Size vs. Delay (Ice).

c) Silent

Figure 5.12 shows the obtained results by the four analyzed approaches when the Silent video sequence is employed. When TD is used, a good PSNR is attained, recovering all the frames, when the buffer size is greater or equal to 1 MB. At 1 MB the buffer threshold is even not achieved. On the other hand, applying the RED algorithm we are able to obtain good results when the buffer size is greater or equal to 3 MB. Contrary to the results produced by the two aforementioned approaches,

using both SDQL and QDP strategy is feasible to reconstruct all the frames using the JSVM decoder. At the same time, they present a high and quite similar PSNR (over 34 dB). Figure 5.13 illustrates the PSNR achieved by the four approaches, using different buffer sizes.

Tail Drop (TD)			
Buffer Size	PSNR(dB)	discarded packets	Decoded frames
3MB	35.2276	*Buffer threshold is never attained	1800
1 MB	35.2276	*Buffer threshold is never attained	1800
500 KB	31.0335	10102	671
250 KB	31.5597	14337	201
100 KB	29.4775	15413	92
50 KB	30.0261	15798	45

(a) No Buffer

Random Early Detection (RED)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	probability	Decoded frames
3 MB	35.2276	*Buffer threshold is never attained		1800
1 MB	28.654	0	9	1796
500 KB	27.0415	0	44	1793
250 KB	25.9097	0	77	1794
100 KB	31.4188	15546	0	71
50 KB	31.5707	15896	0	35

(b) RED

Simple Discard Quality Layers (SDQL)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	Algorithm	Decoded frames
3 MB	35.2276	*Buffer threshold is never attained		1800
1 MB	34.8204	0	2296	1800
500 KB	33.8938	0	4371	1800
250 KB	33.9364	0	4428	1800
100 KB	34.475	0	2819	1800
50 KB	31.6896	0	2221	1800

(c) SDQL

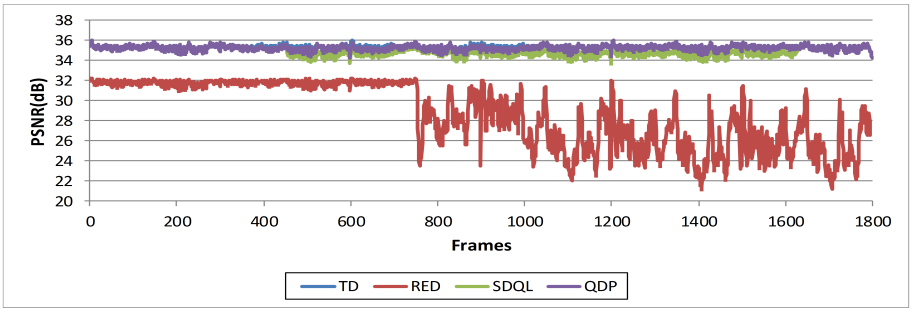
Quality Discard Packets (QDP)				
		Number of discarded packets		
Buffer Size	PSNR(dB)	large size	Algorithm	Decoded frames
3 MB	35.2276	*Buffer threshold is never attained		1800
1 MB	35.1855	0	390	1800
500 KB	34.9401	0	1496	1800
250 KB	34.4211	0	2523	1800
100 KB	34.5743	0	2411	1800
50 KB	34.5521	0	2220	1800

(d) QDP

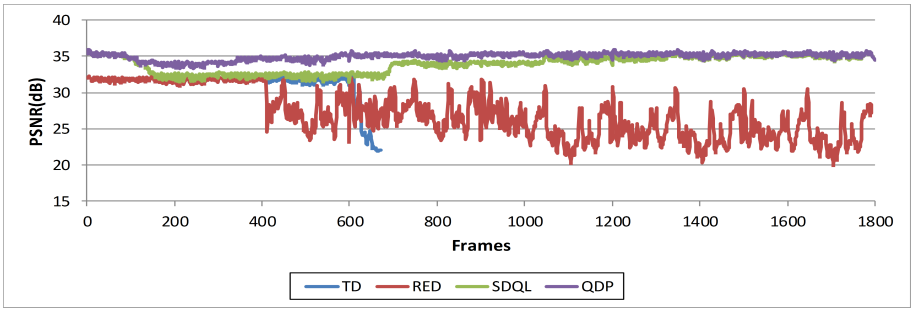
Figure 5.12: Silent simulations results.

As we can see in Figure 5.13, the PSNR attained by the QDP algorithm is higher than the one obtained by the other approaches. It is important to notice that even when the buffer size decreases, the PSNR of the QDP is kept high and constant. Moreover, QDP is able to reconstruct all the frames (1800), also under constrained situations as the presented in Figure 5.13 (d). In opposite, the RED strategy does not get to rebuild all the frames due to its random discard packets mechanism. If TD is employed, the amount of discarded packets increments as the buffer size is reduced. As it is shown in Figure 5.13 (c), with a buffer size of 100 KB, only the 5.11% of the total number of packets is reconstructed by FFMPEG, when TD is applied. A similar situation occurs also with RED, where at 100 KB it is able to recover only the 3.9% of the packets. Similar to the previous experiments, the PSNR achieved by the SDQL algorithm is good. Moreover, considering the obtained PSNR results presented in Figure 5.13, we could say that there is no a significant difference between SDQL and QDP strategies. This could be because the silent video sequence contains reduced amount of movement.

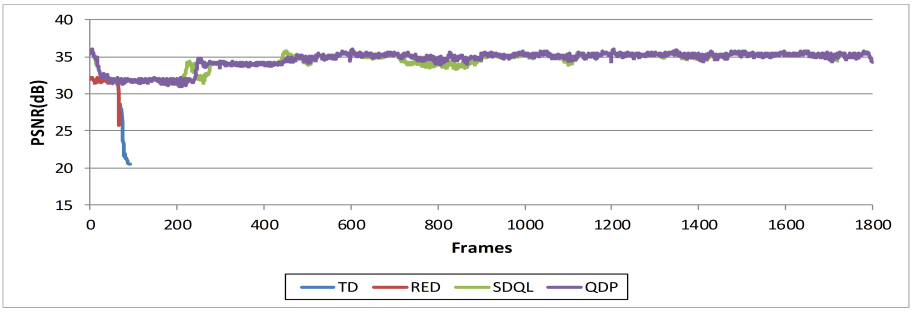
In Figure 5.14, the buffer occupancy produced by all the approaches in different cases (i.e. buffer sizes) is represented. As in the previous video sequences analyzed in this section, the buffer occupancy of the TD presents the same behavior. It grows until to attain the maximum buffer capacity, and starts to decrease at the end of the enqueue process. To constrained buffer sizes, also RED behaves as TD, it can be seen in Figure 5.14 (c) and (d). On the contrary, SDQL and QDP algorithm maintain the queue size on the defined threshold, which is the 50% of the total buffer size. In all the cases, the queue size does not attain the maximum buffer size.



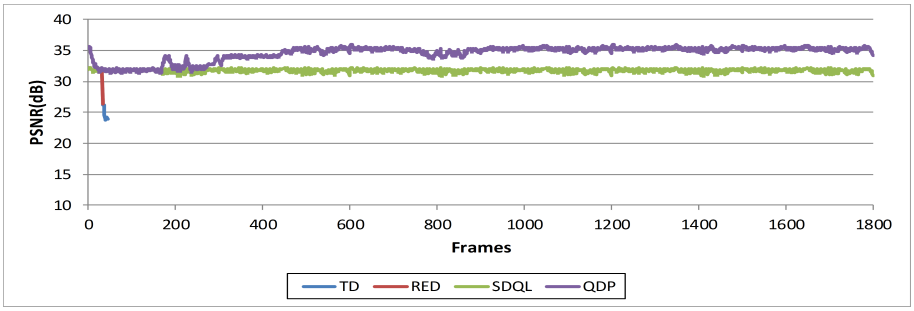
(a) 1MB



(b) 500KB

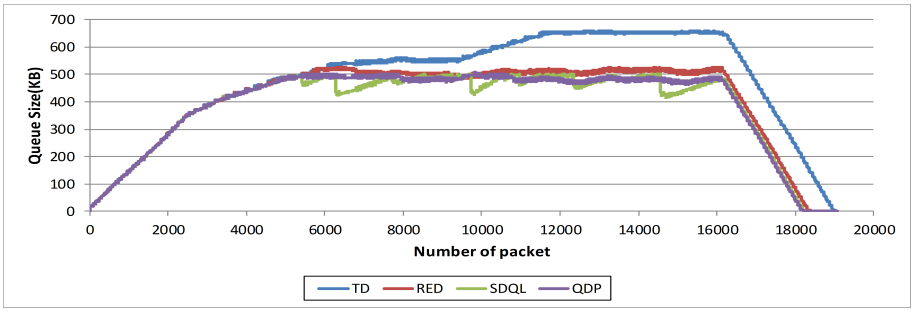


(c) 100KB

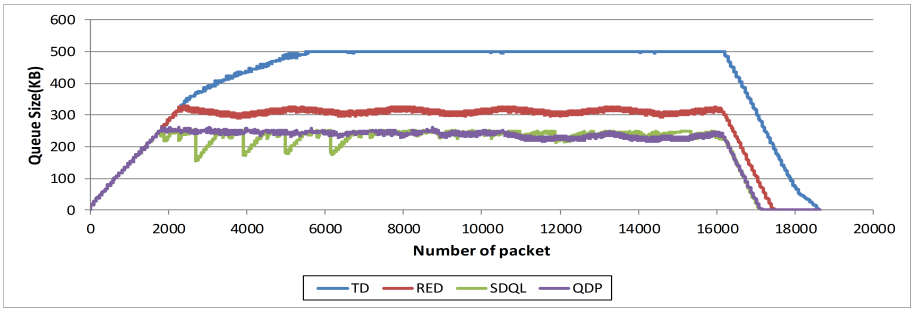


(d) 50KB

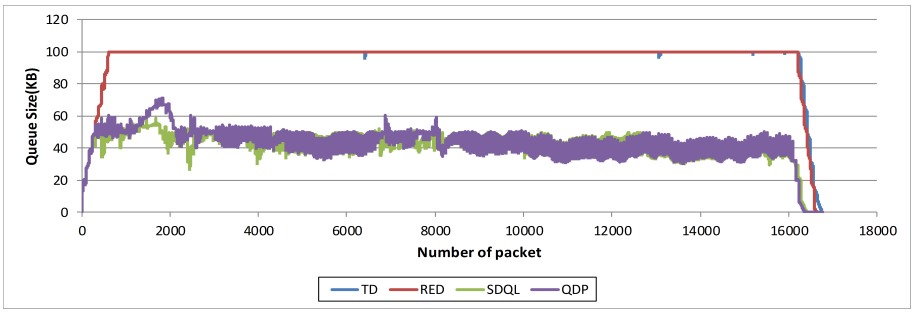
Figure 5.13: PSNR vs. Frames to different buffer sizes (Silent).



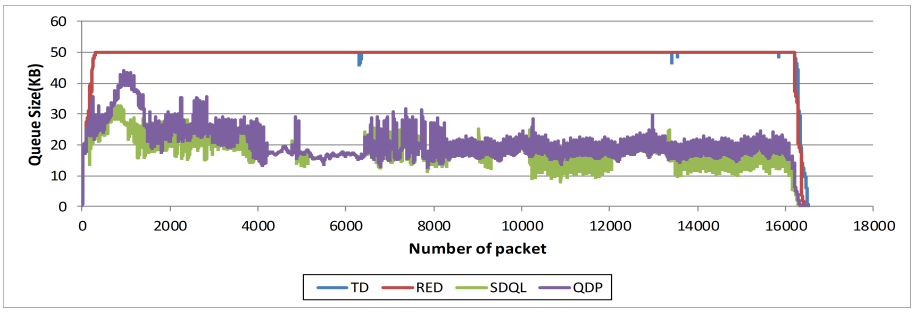
(a) 1MB



(b) 500KB



(c) 100KB



(d) 50KB

Figure 5.14: Buffer occupancy to different buffer sizes (Silent).

As we can see in the two first cases, Figure 5.14 (a) and (b), also RED provides a positive buffer occupancy, but due to its random nature it is not possible to recover the original sequence. Finally, Figure 5.15 illustrates the average delay obtained by the evaluated approaches, when different buffer sizes are used. As we can see in the figure, the produced delay times decrease as the buffer size grows down. It

is important to notice that, in this experiment, the delay times obtained by both, SDQL and QDP algorithm are comparable.

From the results of simulations presented in this section, it is evident that the discard packets algorithm proposed in this thesis provides better results. Moreover, due to its priority discard packets strategy is possible to reconstruct the original video sequence in the highest available quality.

A final consideration is related to the applicability of the proposed approach to a large set of independent connections, and thus comparing it to some adaptive streaming algorithms. It is worth noting that the exploitation of scalable coding avoids the need to pre-encode a large set of copies of the same content. Furthermore, as seen from the results above, the requirements on buffer size to compensate for the bandwidth fluctuations are clearly smaller than for the other two approaches. Therefore, reducing the latency of the transmission chain.

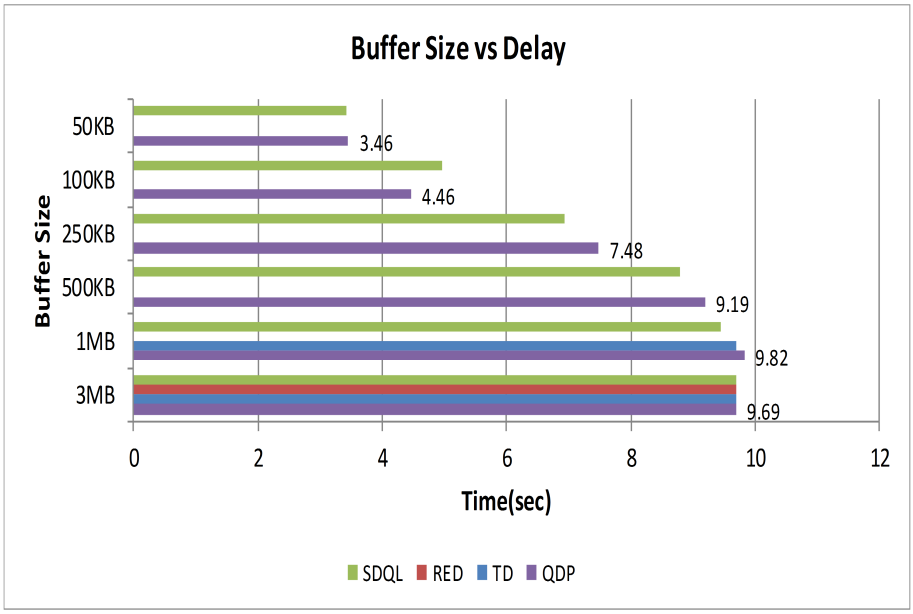


Figure 5.15: Buffer Size vs. Delay (Silent).

Chapter 6

Scalable video streaming over MPEG-DASH

In the last years video streaming has become more and more popular, causing over 50% of traffic on the Internet [56]. Therefore, the use of an adaptive technology able to work in heterogeneous environments and to provide certain level of QoS is required.

Dynamic Adaptive Streaming over HTTP (DASH)[57] offers an elegant solution to these demands. It provides a smart adaptation, adjusting the quality of the streaming according to several parameters placed by the client. Contrary to previous streaming protocols as Real-Time Transport Protocol (RTP), DASH uses a HTTP connection. In such way, it copes with most firewall and proxy settings.

The principal feature of DASH is allowing the distribution of content with different encoding parameters. Hence, it provides support to heterogeneous devices with diverse screen sizes, network connections and decoding capabilities.

However, the benefit of being able to satisfy many different devices and spatial resolutions leads to an increase in the storage requirements and network usage. To deal with this problem, video platform providers have to use external Content Distribution Networks (CDNs), such as Akamai or Amazon CloudFront, in order to handle the enormous amount of traffic.

On the other hand, in order to maintain a stable network and reduced network load, the Future Internet community has proposed caching popular content closer to the consumer. It must be remarked that DASH proposes to use many representations at different quality level. For instance, we could have three different users requesting basic, medium and high respectively video quality. A cache would have to store all three copies, or to store at least a very high quality and provide the lower qualities by transcoding. In the case of streaming multiple representations and many different videos, the caching node(s) would quickly become overload and ineffective [58].

To face this issue, the fusion of Scalable Video Coding (SVC) with DASH is a interesting solution. In SVC, videos are split into several layers, one base layer which provides basic quality and one or more enhancement layers which increase the quality. The SVC presents a hierarchical model where the upper layers depend on the bottom ones. Bending SVC with MPEG-DASH allows to satisfy diverse consumer demands, while furthermore allowing CDNs and caches to be used more efficiently by prioritizing the base layer, and providing enhancement layers only when resources are available. Moreover, in order to get a finer granularity level of quality,

SVC/SNR scalability provides the Medium Grain Scalability (MGS) mode, where a given enhancement layer can be partitioned into several MGS layers. Considering the aforementioned statements, MGS can be exploited to get a larger available range of bit-rates at less overhead than Coarse Grain Scalability (CGS), and thus a better playback video quality can be archived.

6.1 DASH specification

MPEG-DASH (*Dynamic Adaptive Streaming over HTTP*) emerges with the purpose of normalize the *Adaptive Streaming over HTTP*. This standard could be considered as a combination of some streaming protocols such as Apple HLS, Adobe HDS and Microsoft Smooth Streaming.

The multimedia content is captured and stored on a HTTP server and it is delivered using HTTP. The content exists on the server in two parts: Media Presentation Description (MPD), which describes a manifest of the available content, its various alternatives, their URL addresses, and other characteristics; and segments, which contain the actual multimedia bit streams in the form of chunks, in single or multiple files.

Fundamentally, the MPD is a XML document which contains information about the media objects and its features. This file consists in a sequence of one or more consecutive periods. A period is a program interval along the temporal axis which contains one or multiple adaptation sets. In turn, an adaptation set is composed by one or several versions of the same media component. These versions represent various encoding alternatives (i.e. bit-rate, resolution,etc.) and are denominated representations. Each representation consists in one or more segments. A segment corresponds to a chunk of the media stream. Moreover, each segment has a URI- that is, an addressable location on a server. Using the URI associated to each segment, they can be downloaded using HTTP GET or HTTP GET with byte ranges. Figure 6.1 demonstrates the MPD hierarchical data model.

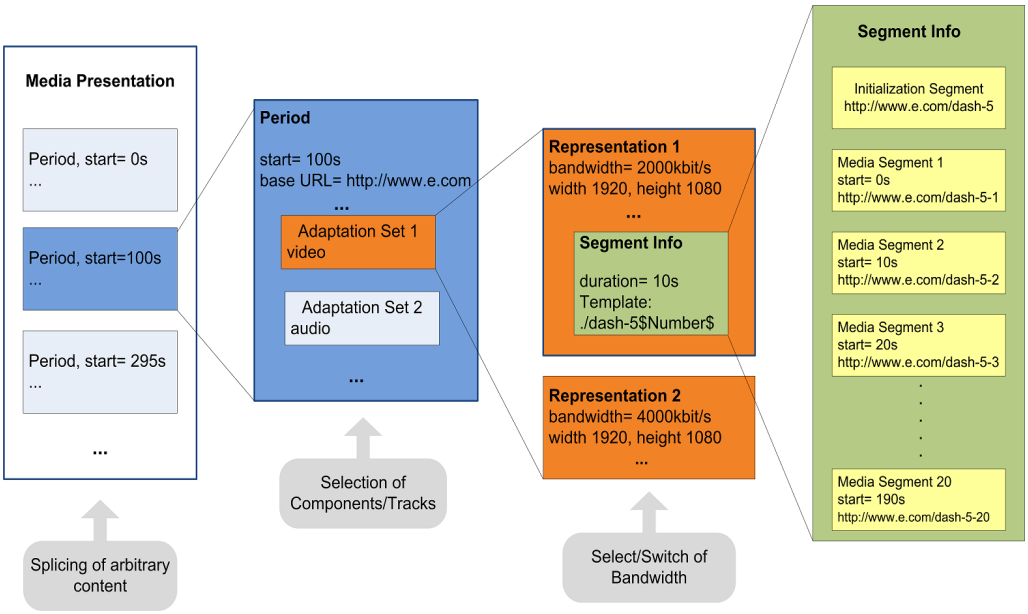


Figure 6.1: The Multimedia Presentation Description hierarchical data model.

To use this data model, the DASH client first receives the MPD of the requested video. Then, this file is parsed and the DASH client learns about the program timing, media-content availability, media types, resolutions, minimum and maximum badwidths, the existence of various encoded alternatives of multimedia components, accessibility features and required digital rights management (DRM), media-component locations on the network, and other content characteristics [47].

Employing the information described in the MPD, the client capabilities and user's choices, the DASH client selects the appropriate encoded alternative and starts streaming the content by fetching the segments using HTTP GET request. When the client buffer attains a suitable level to tolerate network throughput variations, the client continues fetching the successive segments and also constantly monitors the network bandwidth fluctuations. Depending on its measurements, the client decides how to adapt to the available bandwidth by fetching segments of different alternatives (lower or higher bit-rates) to maintain an adequate buffer. The communication between the HTTP server and the DASH client described above is depicted in Figure 6.2.

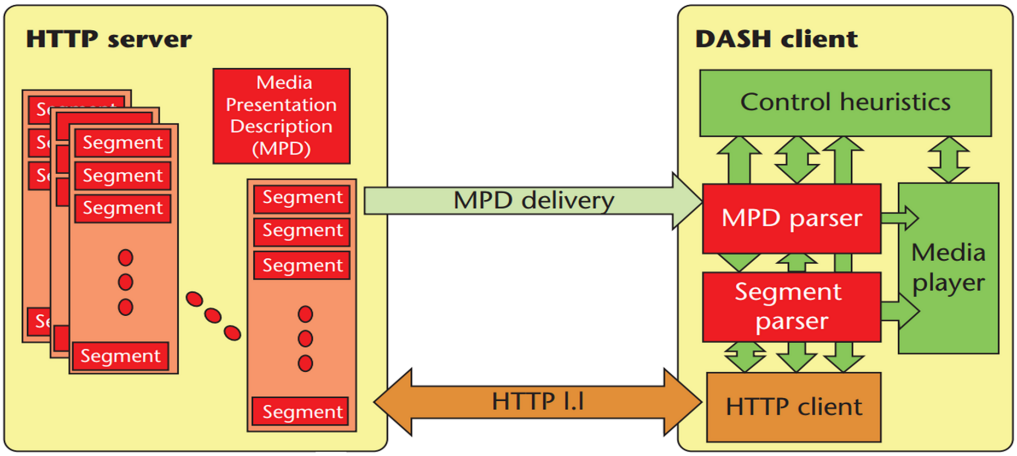


Figure 6.2: MPEG-DASH Standard [47].

It is important to highlight that the MPEG-DASH specification only defines the MPD and the segment formats. The delivery of the MPD and the media-encoding formats containing the segments, as well as the client behavior for fetching, adaptation heuristics, and playing content, are outside of MPEG-DASH scope.

6.2 Scalable Video Coding and DASH

In DASH, a video is offered in several (typically 4 to 10) versions [59]. These different versions are split into chunks, generally around 2 to 10 seconds, depending on the application and scenario, and they are stored in the DASH server.

On the other hand, the DASH clients are responsible of managing and controlling all the streaming process. Considering the client capabilities and the network congestion level, each terminal decides which is the more suitable video version to be downloaded. It must be taken into account the fact that the network varies moment-by-moment, therefore, the next segment to be selected might have different bit-rate than the previous one. In short, all the DASH clients download consecutively the

most appropriate chunks, based on the information obtained by monitoring recently downloaded chunks of the ongoing video.

Nowadays, there are two feasible possibilities to provide DASH. The first is to encode multiple representations of the video, using H.264/AVC, at the server and offer them side-by-side. The second possibility is to use Scalable Video Coding (H.264/SVC) [5]. In that way, all the videos are encoded once at the highest quality offering all these representations embedded in one file. Then, the different representations can be extracted from the bit stream to create new valid sub-streams. Certainly, the feasibility of having several diverse representations of the same video within the same bit stream represents a reduction in the storage requirements at the server, and it might also result in an increase in cache performance.

Taking advantage of the scalable property of H.264/SVC which divides the media content into layers, that correspond to different quality, spatial or temporal representations, SVC provides the possibility of serving a great number of users with different equipment capabilities with a single bit stream. Moreover, SVC provides flexibility to DASH, since it allows dividing media content both per SVC layer and per time intervals, and thus prioritizing very accurately the different elements of the media content according to their importance.

The SVC bitstream is composed by a base layer, which corresponds to the lowest representation, and one or more enhancement layers, which increase the quality, spatial and/or temporal representation when added to the base layer. Therefore, a higher responsiveness and better playback quality under adverse network conditions is attained since a request for a time interval is desintegrated into multiple requests (HTTP GET requests). These requests are performed subsequently, one for each of the layers, and when congestion is detected, requests for higher layers may be omitted. Conversely, for the AVC case only one request is emmited for the entire data of a given interval. This leads to experience longer waiting times to download the data requested until switching to a lower representation can be performed [59].

Furthermore, SVC provides flexibility to DASH but this flexibility might be even more increased applying quality or fidelity scalability (SNR), specially Medium Grain Scalability (MGS). With MGS is possible to achieve multiple Operation Points (OP) or target bit-rates with a reduced number of quality layers. Since the number of desired OPs results in an increase in coding overhead, MGS allows keeping the overall coding overhead within an acceptable range. There are many benefits of using SVC for a HTTP streaming service. In [60], there are presented the improvements of SVC when compared with AVC, in a HTTP streaming scenario. The principal advantages of using SVC are listed below:

- SVC provides multiple versions of the video within the same bit stream, and consequently the caching efficiency is significantly improved.
- A meaningful enhancement of using SVC is the receivers can react much faster to network variations, which prevents the interruption of video when the traffic in the network unexpectedly increases.
- SVC provides flexibility to DASH, prioritizing the elements of the media content.
- Low delay streaming.

- Save uplink bandwidth at the server.
- Better match to the available network resources in live services.

6.3 SVC-MGS DASH Client and Server implementation

6.3.1 Overview

In the literature have been reported two main works which provide mechanisms, and make available some tools to generate a SVC bitstream according to DASH model. One of them is presented in [61], it is a MPEG contribution which specifies the segmentation of the SVC bitstream to be utilized for DASH. In [61] is provided a SVC de-multiplexer deployed at the server side, and a re-multiplexer placed into the client. A normal SVC bit stream has enhancement layers located at each frame. The de-multiplexer for SVC-DASH splits the SVC bit stream into chunks, one per layer. In addition, it also generates the MPD file. Finally, the re-multiplexer reorders SVC layer chunks into a single SVC bit stream, so this resultant bit stream is ready to be decoded.

Based on [61], Christian Kreuzberger et al. [58] provide a *Dynamic Adaptive Streaming over HTTP* (DASH) toolchain for converting SVC encoded videos into DASH-compliant streams. Furthermore, [58] provides a DASH/SVC dataset, where the videos are encoded employing four different variants. The dataset, as well as the enconding configuration files and the generated MPD files are available online[1]. The enhanced tools presented in [58] are open source and provide scripts for de-multiplexing and re-multiplexing video content which has been encoded according to the H.264/SVC. Conversely to the previous work that only supports coarse-grain quality scalability, the current work supports also spatial scalability. The DASH-SVC-Toolchain also implements another useful tools such as a H.264/SVC Analyzer and a SVC MPD file Parser. The first one analyzes the encoded bit stream and it controls that everything is according with the DASH model (i.e. each segment must start with a IDR-frame). The second tool extracts the information related with the features of the encoded video and the URLs of all the segments and layers.

In order to create a DASH-compliant structure, the SVC bit stream is demultiplexed into several segments, where each segment contains frames for multiple layers. These segments must satisfy two mandatory conditions: containing exactly the same number of frames, and starting with a IDR Frame (Instantaneous Decoder Refresh). Moreover, with the aim of exploiting the full potential of SVC, each segment is broken up into multiple files, one per layer. In such way, if the network conditions are favorable in a specific interval of time, the client could request the file corresponding to the highest available quality of that segment. Considering the fact that the files belonging to the upper layers are dependent to the bottom ones, all these files must be acquired by the client (i.e. segment0-L0, segment0-L2, segment0-L3 and segment0-L4). Finally, at the client side the files must be multiplexed to obtain a unique file by segment. This process is done by following the process explained above in reverse order, where the received files corresponding to the different quali-

[1]DASH/SVC Dataset is available in: http://concert.itec.aau.at/SVCDataset/

ties are merged generating the segment that will be decoded. If some enhancement layers are missed, the decoder will notice this and it will only decode the available subset.

Additionally, it is important to remind that MPEG-DASH specification only defines the MPD and the segment formats. However, there is an open source library called *libdash* presented in [62], which is implemented at the client side. It encapsulates the MPD parsing and HTTP module that is responsible to handle the HTTP download per request through the streaming logic. Thus, this library provides interfaces for these modules to access the MPD and the downloadable media segments. In a typical deployment, a DASH server provides segments in several bit-rates and resolutions. The client initially receives the MPD through *libdash*, and based on the MPD information, the client can download individual media segments using *libdash* at any point in time. *Libdash* is used in [58] for parsing MPD files.

In this thesis, with the aim of having control of both the transmission and reception process, a streaming server and client were implemented under the DASH model. They are explained below.

6.3.2 Streaming Server and Client using DASH specifications

At the server side, the raw videos are split in chunks of 2 seconds and then, each chunk is encoded with H.264/SVC-MGS using the JSVM reference software. Research literature typically assumes a coding overhead of 10% per enhancement layer, for this reason MGS become a optimal solution to control this issue. The configuration employed in this thesis is: one base layer and three enhancement layers. In turn, each enhancement layer defines 2 MGS sub-layers (MGS vectors). Accordingly, in total seven different operation points (bit-rates) have been defined. The videos used in this section of the thesis were encoded considering the bit-rate recommendations for SVC streaming indicated in [52]. These bit-rates are based on the encoding recommendations of Industry solutions. Table 6.1 shows the resolutions and bit-rates used for the encoding process. According with our encoding configuration specified above, we will use four bit-rates.

Table 6.1: Devised Bit-rate recommendations for SVC streaming.

Resolution	4 bi-rates(Kbit/s)	2 bit-rates(Kbit/s)
1920×1080	10400, 7200, 5500, 4000	8800, 6050
352×288	1950, 1080, 500, 270	1320, 330

Furthermore, a key parameter to be defined when encoding content to be DASH-compliant is to force an IDR-Frame to be at the beginning of every segment, to make segments independently decodable. In JSVM, this is obtained specifying the *IDRPeriod*. The value of this parameter is equal to a multiple of the frame rate. For instance, if we use video sequences at 24 fps, then, the *IDRPeriod* will be 48 frames corresponding to 2 seconds of video. For a better understanding, IDR frames are intra-coded frames that refresh the coding picture buffer. When a IDR coded picture is received, the decoder marks all pictures in the reference buffer as “unused

for reference". Hence, all subsequent transmitted slices can be decoded without reference to any frame decoded prior to the IDR picture.

Now, employing the *BitStreamExtractor* tool, included in the JSVM software, we extract the sub-streams of the SVC stream. The sub-streams represent streams with a reduced quality (bit-rate). It is important to note that the encoded chunk owns the highest quality. Then, for each encoded chunk we need to extract seven sub-streams representing base layer and enhancement layers (each EL has 2 MGS layers) respectively. Sub-streams of the lower layers are the result of the removal of some packets belonging to the upper layers. Figure 6.3 illustrates the complete segments generation. For instance, El1.1 and El1.2 represent the MGS layers of the enhancement layer one.

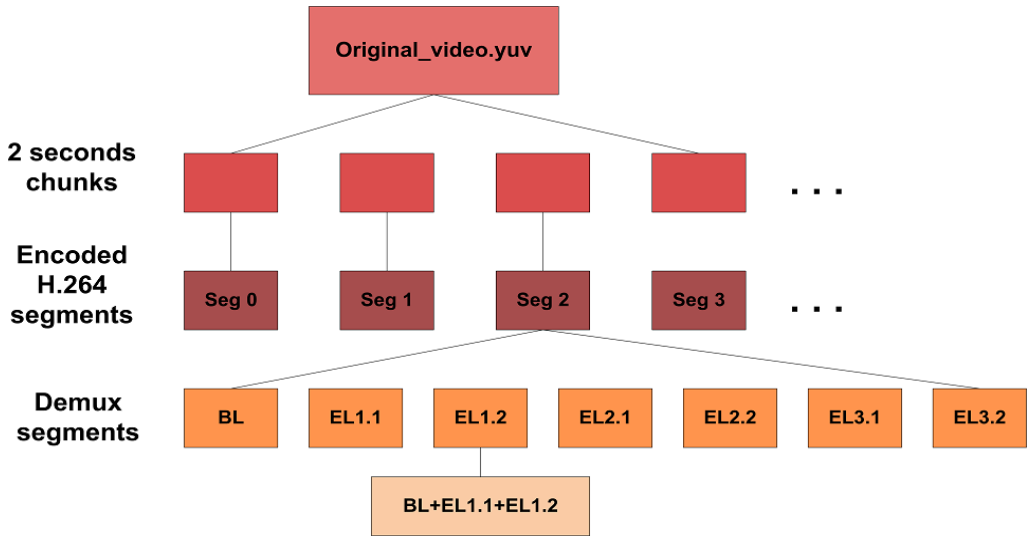


Figure 6.3: Segmentation of SVC-MGS bi-stream.

The MPD file and the segments corresponding to the encoded videos could be stored locally or in a Web Server. In this thesis, a Web Server is built using Apache Web Server[2].

The implementation of the Streaming Server and Client is done using *Boost*[3] C++ libraries. *Boost* libraries provide among others, the *Boost.Asio* library which can process any kind of data asynchronously (it is mainly used for network programming). Network functions are a perfect use for asynchronous operations because the transmission of data over a network may take a long time, which means that acknowledgments and errors may not be available as fast as the functions that send or receive data can execute. *Boost.Asio* provides many I/O objects to develop network programs. We use this library principally to establish a connection with the computer representing our streaming client. Once the communication has been established, the exchange of data can be done through sockets.

Practically, the first step is to establish the communication between server and client. For this, we launch the server specifying the port and, at this point, the server is listening for new connections. Then, we launch the client indicating the ip address of the server that we desire to connect and the listen port. It is important to set up the same port on both, server and client.

[2]Official Apache Server Project Website: https://httpd.apache.org/
[3]Official Boost library Website: http://www.boost.org/

Once the connection is opened, the client requests one of the videos stored into the server. The server receives the request and immediately starts sending the MPD file corresponding to the demanded video. In its turn, the client gets the MPD file and parses it. In such way, the client learns about the available representations. Parsing the MPD, in our implementation, is done employing the *ParseMPD* tool from DASH-SVC-TOOLCHAIN [58]. Finally, in order to request the more suitable segments we need to consider the available bandwidth. For doing this, we use the BEM presented in chapter 4. It is worth to mention that the state sequence, generated for performing the simulations developed in this chapter, is obtained from network trace files, which are the result of measurements of video streaming in a DASH environment. These traces are available in [63]. The state sequence produced by the BEM provides us the information about the current conditions of the network at each instant of time. Based on this data, the client starts sequentially fetching the segments from the server. As soon as segments arrive to the client, they are decoded and then played out.

6.4 SVC-DASH adaptation strategy

The main target of DASH is to allow the client access different representations of video content via the HTTP protocol. Hence, to avoid waiting times during the video playback due to empty playback buffers, the client can flexibly switch between different qualities. However, quality switches produced by the network oscillations, and the playback of lower qualities may reduce the Quality of Experience (QoE) perceived by the user [64].

There are two solutions for delivering video streaming: RTP/UPD and HTTP/TCP. The first one is commonly used in video conferencing and IPTV where the real-time constraints are severe. In this case the insufficient network resources generally lead to lost packets, and consequently, the video pictures are distorted. The HTTP/TCP differs from the previous solution because here the delivery of packets is guaranteed. However, it involves additional waiting times in packets delivery which may lead to an empty video playback buffer, and interruptions of the video playback.

The playback latency is one of the most important parameters to be taken into account when we talk about the Quality of Experience (QoE) perceived by the user. Some studies, such as [65] and [66], conclude that the final users are extremely sensitive to playback stalling. Accordingly, stalling, as much as possible, should be avoided even at the expense of increasing the initial delay. Furthermore, another essential factor that influence the QoE is the frequency with which the quality of the video is switched, accordingly the abrupt transitions should be avoided at any cost. Moreover, quality changes from low quality to high quality are preferred for the users because they perceive an enhancement in the video improving the QoE [67]. Therefore, it is better to start with a low quality and switch up to a higher quality than starting with a high quality and switch down to a lower one.

In order to exploit SVC and DASH, the video quality has to be improved when it is possible, but also a stability needs to be attained. The abrupt quality changes, especially from high quality levels to lower ones should be avoided. These types of transitions must be done in a smooth way. SVC provides the opportunity of offering the same video with a wide range of diverse qualities. This increases the probability to attain a smooth transition between the available qualities. In order to decide and

control which is the more suitable transition to be carried out at each instant of time, respecting the users requirements and the network resources, new adaptation techniques and algorithms are demanded.

6.4.1 Overview

In the literature, there are several works which exploit SVC in order to reduce stalling times during a video presentation. In such way, the QoE is improved. [68] presents a P2P VoD architecture based on Tribler[4] [69] that supports SVC video streaming. The VoD client of Tribler adapts two of the most important mechanisms of BitTorrent[5], which are: the peer selection strategy in the unchoking process, and the chunk selection strategy. Moreover, the most important difference between BitTorrent and Tribler is that a Tribler's user can start watching the video while it is downloaded. Hence, in order to ensure a continuous video playback, the rarest-first [70] chunk selection of BitTorrent is replaced by a strategy based on priority sets.

The approach proposed in [68] separates all the video in 3 groups: high-priority set, medium-priority set and low-priority set. The first set contains all chunks with frames from the playback position until 10 seconds after the beginning. The second one is composed by the following 40 seconds of the video. The remaining frames comprise the low-priority set. With the aim to achieve the maximum video quality that can be supported by the network capacity, the chunks are downloaded considering the groups defined above. In the high-priority set, only the chunks corresponding to the base layer are downloaded, ensuring in that way the video playback. In the second and third set, chunks from a higher enhancement layer are downloaded only if all chunks of the lower layers are locally available. Additionally, [68] only supports temporal scalability and uses a considerable buffering time equal to 60 seconds before starting the video playback.

On the other hand, in [64] an user-centric DASH/SVC streaming algorithm is proposed. This algorithm reduces the number of quality switches by striving for a stable buffer level before increasing the number of consumed SVC layers. It is important to mention that this work only considers spatial resolution. Moreover, the results of the evaluations show a very high and stable overall playback quality of the proposed algorithm compared to other state-of-the-art SVC-DASH adaptation techniques. However, the comparison does not take the amplitude of quality switches into account.

Considering the fact that abrupt quality switches reduce significantly the QoE, [52] introduces the concept of representation switch smoothing and discuss three implementation options for the smoothing component in a SVC-based DASH system. These implementations can be placed either before, within, or after the decoder. The first option, denoted *pre-decoder* implementation, consists in adding a filter component before the decoder. This component removes certain pictures fidelity data. For SVC with MGS enhancement layers, this approach is obtained removing the transform coefficients from the enhancement layers. The second alternative, implemented inside the decoder is referred as *in-decoder* implementation. In the

[4]Tribler is an open source decentralized BitTorrent client which allows anonymous peer-to-peer by default.

[5]BitTorrent is one of the most common communications protocol of peer-to-peer file sharing which is used to distribute data and electronic files over the Internet.

same way that the aforementioned implementation, some picture fidelity data is removed from the coded frames, but without affecting the motion compensation of other frames. The latter one is more accurate and robust than the *pre-decoder* implementation due to the fact that it avoids error propagation. Finally, in the *post-decoder* implementation, a video filter component is added after the decoder for inserting additional noise into the decoded frames. The principal drawback of the last implementation is the high computational complexity, and the need to use a video quality approximation model. Based on a study on quality switches presented in [67], where was concluded that high amplitudes in down-switches should be avoid and that switching up is preferred to switching down, some evaluations and subjective tests were done in [52]. The sequences employed for the tests were encoded to AVC at constant target bit-rates with the FFMPEG encoder. In order to offer a smooth transition, the PSNR for the transition segment was calculated to obtain the rate-distortion performance. Therefore, the encoded bit-rates predict a linear decrease of PSNR over the entire transition duration. It is worth to mention that only down-switching scenarios were considered. The evaluations concluded that there is a slight preference towards representation switch smoothing than a hard quality switch. Moreover, the impact of representation switch smoothing on the QoE depends on the features of the video (i.e. high or low motion), the duration of a quality transition and the amplitude of the representation switch.

Following the same line, a QoE friendly rate adaptation algorithm is proposed in [71]. This algorithm reduces the switching times and achieves a better QoE by smoothing switching from high quality to low quality. The proposed algorithm besides of considering the available bandwidth, considers the buffer level to select the suitable representation. This approach defines a fixed-interval buffer and a reset threshold. When the buffer size, which is measured by the duration of buffered video content, is within the interval, the representation to be selected stays unaltered. On the other hand, when the bandwidth begins to increase, leading to a buffer size higher than the maximum buffer size established, the algorithm decides if it is better to stay at the current representation or switch to the next higher quality. Moreover, when the bandwidth is descending and the buffer level is less than the minimum buffer boundary but greater than the threshold, this approach switches to an intermediate representation. Detecting the decreasing of the buffer in advance, the playback stagnation could be avoided as well as the abrupt switching between the high quality and low quality, thus improving the subjective QoE.

Furthermore, [72] presents a rate adaptation algorithm called QoE-enhanced adaptation algorithm over DASH (QAAD), which preserves the minimum buffer length to avoid interruption, and improves QoE by minimizing video quality changes during the playback. [72] covers two important parts to be considered when an adaptation algorithm in the DASH client is proposed: bandwidth estimation and bit-rate selection. Concerning the bandwidth estimation part, QAAD introduces a more accurate bandwidth estimation which has the aim of balancing the RTT-based and segment-based estimation schemes. The bandwidth estimation employs a periodical estimation scheme, in which a fixed estimation interval is introduced. Then, the bit-rate selection scheme consists in analyzing three different cases before choosing the more suitable quality level. These cases compare if the current video quality level does not exceeds the available bandwidth (l_{best}), is equal, greater or less than the video quality of the previously requested segment (l_{prev}). Thus, if l_{best} is

equal to l_{prev}, the video quality level of the segment to be downloaded remains the same. However, if l_{best} is greater than l_{prev}, the current buffer level is evaluated. In the case where the buffer length is greater than a pre-defined marginal buffer length, the quality of the next segment will be increased in quality by one. Otherwise, the current quality level is kept. Finally, when l_{best} is less than l_{prev}, the current available network bandwidth cannot sustain the previous video quality level, and thus the bit-rate reduction is inevitable. Considering that lowering more than two levels at time would result in significant QoE degradation, the algorithm proposed by [72] tries to keep the next video quality level comparable to the previous video quality level. Furthermore, the QAAD algorithm is compared with an existing buffer aware adaptation algorithm called QDASH [73], where QAAD is more robust and can improve QoE of users even under severe network bandwidth fluctuations.

A different approach is presented in [74]. Here, a Smooth DASH (S-DASH) scheme which adapts the video quality using the media segment duration, the segment fetch time and the buffer state of the client to reduce the change of video quality is proposed. Considering the fact that the duration of the segment, depending on the network conditions, plays an important role in reducing the change of video quality and avoid a client buffer underflow, the S-DASH scheme determines the segment duration based on variance of chunk throughput. Moreover, the video quality adaptation module is based on the buffer state and the segment throughput, which is obtained by dividing a segment by its fetch time. Basically, the video quality algorithm consists in deciding when switch-up, switch-down, or keep a video quality unchanged, taking into account, in all the cases, that the segment throughput has to be higher that the bit-rate of video quality. From the simulations is notable that S-DASH, using smooth video quality adaptation and segment duration determination, improves the QoE of DASH service. However, due to the employment of various segments durations, and each of them in different versions (i.e. different bit-rates), the Web Server requires a considerable storage capacity.

A DASH client commonly employs an adaptive bit-rate selection (ABR) algorithm to pick the most suitable representation. The ABR algorithms proposed in previous literature use parameters like the average segment download time [75], available bandwidth [73], or buffer occupancy [76]. A recent approach proposed by [2] realized that ABR algorithms ignore the fact the segments sizes vary significantly for a given video bit-rate. Due to this, although an ABR algorithm is able to measure the network bandwidth, it may fail to predict the time required to download the next segment. Thus,[2] proposes a segment-aware rate adaptation (SARA) algorithm which considers the segment size variation in addition to the bandwidth and the buffer occupancy to accurately predict this time. In [77], it is demonstrated how SARA improves the management of the users QoE in a DASH system. Moreover, SARA is compared with a typical throughput-based and buffer-based adaptation algorithm under varying network conditions, given better QoE results, especially in a low bandwidth network.

Finally, [78] investigates the influence of rate adaptation algorithms on the QoE metrics. For this purpose, five different rate adaptation algorithms were implemented and evaluated under varying bandwidth and network scenarios. Among the five analyzed algorithms, it is the SARA algorithm [2]. In this work is deduced that algorithms which consider both buffer occupancy and estimated bandwidth achieve a better QoE than others considering these factors independently. Moreover, the

employment of the segment size as in SARA produces fewer bit-rate switches and no or fewer interruptions in all scenarios, thus outperforming the others algorithms in terms of QoE.

From the literature, we extract the most relevant considerations that must be taken into account at the moment to propose a new DASH adaptation strategy. These points are listed below:

- The quality perceived by the user is influenced by several factors such as: initial delay, stalling delays and frequencies, number of quality switches, and played back video quality.
- To exploit both SVC and DASH, the video quality must be improved when it is possible, but also some form of stability needs to be attained.
- The quality transitions must be done in a smooth way, avoiding abrupt quality changes. Especially from high quality levels to lower ones because the user perceives these changes as a degradation of the quality.
- The quality switches from low quality to high quality are preferred for the users because they perceive an enhancement in the video. Hence, improving the QoE. It is important to highlight that is better to start with a low quality and switch up than start with high quality and switch down.
- With the aim to have a better control of the available resources and aid in the switch quality decision, the buffer level must be considered by the adaptation algorithm. It is useful to define a range in buffer size, i.e., minimum and maximum bounds. Also, it is important to establish a buffer threshold in order to manage extreme cases (e.g. when the bandwidth continues to decrease over a long time).
- Depending on the application, an initial delay could be introduced at the beginning to the presentation, in that way stalling times along the video playback could be minimized. This time has to be defined based on QoE parameters.
- The quality switches must be minimized as much as possible.
- The segment size is a crucial parameter that must be considered to select the more suitable bit-rate of the segment.

6.4.2 DASH quality decision algorithm

According to the MPEG-DASH standard specification, the DASH server is a simple web server where the segments of video are stored and it could be considered as a passive agent. The client, on the contrary, manages and controls all the streaming session, thus playing an active role.

From the previous section it is possible to conclude the importance of considering the segments size variation in order to obtain a more accurate estimation of the download time. As seen before, SARA [2] proposes to consider this parameter at the cost of modifying the original MPD file. The information related with the size of the segments is only available at the server side. Hence, to do available this information also at the client side, this information necessarily has to be transferred

through the MPD file. Consequently, the original format of the MPD is modified due to the addition of a new parameter specifying the size of the segment.

Considering that the MPEG-DASH specification defines the structure and components of the MPD file, for reasons of standardization and compatibility this file must not be altered. Therefore, with base on SARA, but avoiding to change the MPD file, we propose a conservative quality decision algorithm developed at the server side. Basically, in this approach the roles of DASH client and server are inverted in some way. In contrast to the typical DASH scheme, the decision of which is the most suitable version (quality) of the segment to be downloaded and played, is taken by the server. In order to have access to the segments sizes information, we create a Segment Size Map (SSM) which is stored into the server. Using as input data the representation requested by the user, which intrinsically indicates the current available bandwidth, and the SSM, our approach selects the more convenient segment that can be transmitted.

From the literature, it has been defined that the Quality of Experience (QoE) can be improved minimizing the number of up to down quality switches because the users are more perceptible to these variations than to down-up ones, which are considered instead as an increase of QoE. Regarding the aforementioned, our approach aims to minimize the up-down switches, selecting the highest possible representation that can be delivered to the client on time for its presentation, avoiding thus delays. It should be empathized that the awaiting time or re-buffering produced at the client side, when the next segment to be played is not arrived on time has a negative impact on users who could desist of continuing watching the video.

Our implementation presents two essential phases: the generation of the video segments and the development of the DASH quality decision algorithm (DQD) at the sever side. For creating video segments, we take advantage of MGS to increase the rate adaptation points, therefore achieving a smooth transition between the available qualities. For experimental purposes, we used the *Big Buck Bunny* video which presents a duration around 10 minutes, CIF resolution (352 x 288) and 24 frames per second. This video is divided into segments and each one of them is encoded using MGS. The encoder configuration defines one base layer and three enhancement layers. Each one of the enhancement layers defines 2 MGS vectors.

Basically, each segment has a fixed duration equal to four seconds and presents seven different representations, i.e, bit-rates or qualities. It is critical to mention that there are two procedures for encoding video. The first one is fixing the bit-rate and the other one is specifying the desired quality. If we encode segments setting the target bit-rates, we will obtain representations with similar file size, similar bit-rate but different quality. On the other hand, if the quality is established, despite of segments belonging to a specific representation own identical quality, the sizes and bit-rates will vary considerably. For instance, Segment 0, 1, ..., 10, ...,n at the same quality, present sizes and therefore different final attained bit-rates. It is because each segment owns different characteristics such as the number of vectors of movement between frames, which affects directly the size of the encoded file. Hence, the size of one segment at the same representation (quality) than the previous one could be greater or lesser depending of the complexity of their content. Figure 6.4 illustrates the variations on the sizes of the segments at four different representations of the *Big Buck Bunny* encoded video. The total number of segments generated is equal to 150, each one with a duration of 4 seconds. Moreover, in Figure 6.5 is

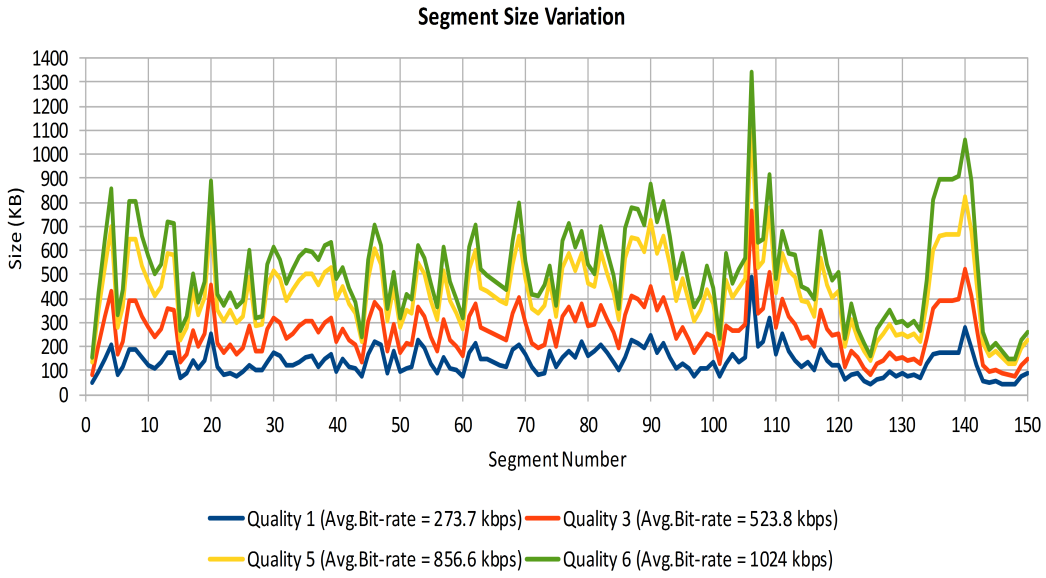


Figure 6.4: Segment size variation.

shown with more detail the drastic changes produced in the sizes of the segments belonging to the highest available quality. Here, we can observe clearly the relevance of considering the segment size at the moment to decide the suitable quality of the next segment requested by the client.

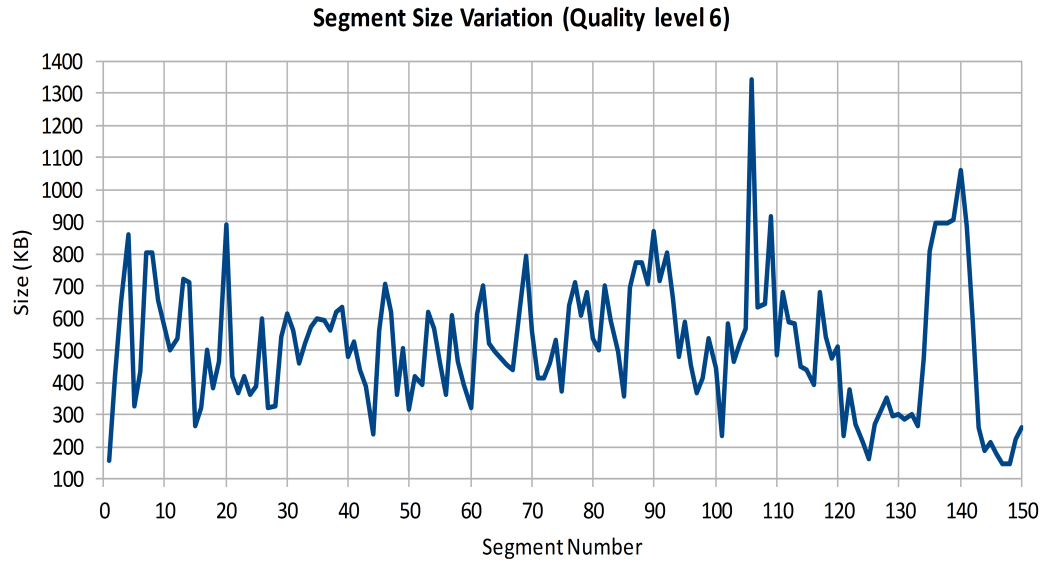


Figure 6.5: Segment size variation at the highest quality.

Once the client's request has been received by the server, the DQD algorithm evaluates if it is feasible to transmit the segment in the demanded quality without inserting delay. In case it is not possible, the DQD algorithm decides the suitable quality for that segment, attempting to deliver it at the highest viable quality and minimizing abrupt quality changes. Using the SSM and the information about the current network conditions, the time required for downloading the segment, the arrival time as well as the time in which the segment will end its presentation can

be calculated. Moreover, it is important to mention that the HMM-BEM presented in Chapter 3 is used at the DASH client to estimate the available bandwidth.

The DQD algorithm is presented below (Algorithm 2). With the objective of reducing the initial delay, the two first segments are downloaded in the lowest quality (line 3). This strategy is commonly used by platforms which offer video on demand (VoD)services such as Netflix, Hulu and Youtube. After that, the quality is increased as fast as the network conditions allow. From the third segment requested, the algorithm controls if the representation requested (r^{rqt}) is greater, equal or less than the representation of the most recent download segment (r^{curr}).

In the first case, if r^{rqt} is greater than r^{curr}, the download time (*Dtime*) required for downloading the current segment at the requested representation is calculated. If this *Dtime* is less or equal to the maximum download time, which is equal to four seconds, r^{curr} is increased (lines 6 - 9). On the other hand, the maximum possible representation is selected (line 11). We defined a maximum download time (*maxDtime*) equivalent to four seconds because the duration of the presentation of each segment is fixed to this value. Hence, in the worst case, the next segment should be available to the client when the previous segment is ending its presentation to avoid inserting stalling times. Actually, at the server side it is not possible to know the status of the client buffer. However, establishing a relative time which starts when the first segment is requested, it is possible to estimate in a very roughly way the segments present on it.

Reminding the fact that the size of the segments varies significantly, it has a direct impact into the download times. Thus, some segments could arrive to the client in less time than expected. This saved time could be used to keep the previous bit-rate or permitting a smooth transition between qualities. In our algorithm, this time is denoted as δ. It is worth say that in the first case, this δ is not applied because the main objective is preserve the quality, instead of increasing it temporarily, and then decrease it again.

Secondly, if r^{rqt} is equal or less than r^{curr}, the download time is computed and it is compared with the maximum available time to download the current segment. In this case, with the aim of staying in the previous quality, this time is equivalent to the *maxDtime* + δ. Otherwise, if the time required to download the current segment in the previous quality exceeds the aforementioned time, the maximum available representation is chosen (lines 14 - 21).

The results obtained when the DQD algorithm is applied are presented in Figure 6.6, where the orange line represents the case when the representation demanded by the client is delivered, without considering if it could be supported or not. On the contrary, the blue line is the result of employing the proposed decision algorithm. The total number of up-down switches produced by our algorithm is equal to 15 and no delay time. Furthermore, when no adaptation mechanism is employed, the number of up-down switches raises to 23 with a total delay of 93.279 seconds (1.55 minutes approximately). Here, we must take into account that the bandwidth of the network which has been considered in this experiment owns a maximum bit-rate corresponding to the average bit-rate achieved by the highest quality (enhancement layer 7) in the encoding process. This value is equal to 1024.076 kbit/s. Moreover, the available bandwidth is estimated by or HMM-BEM with a frequency equal to the segment duration (i.e. four seconds).

On the opposite, in a better network conditions environment where the maximum

Algorithm 2: DASH Quality Decision Algorithm

```
/* Select the suitable representation of the segment to be
   downloaded, considering the request of the client and the
   segment size map.                                          */
```

Data:
$\Re$: set of available representations $\{r^{min},...,r^{i},...,r^{max}\}$
r^{curr}: representation of the most recent download segment;
r^{l}: the lowest representation (quality=0);
$maxDtime$: maximum download time(segment duration);
$Dtime$: download time;
$atime$: arrival time;
$eptime$: ending presentation time;
Input:
r^{rqt}: representation requested by the client;
n: segment number;
Result:
Sq: suitable quality selected to download the requested segment;

```
1  begin
2      if n is 0 OR n is 1 then
3          Sq_n=r^l;                                   // fast start
4      end
5      else
           // requested representation is greater than the previous
              one
6          if r^rqt > r^curr then
7              if Dtime_n r^rqt ≤ maxDtime then
8                  Sq_n=r^rqt;                         // increase quality
9              end
10             else
11                 max { r^i | r^i ∈ ℜ, Dtime_n r^i ≤ maxDtime + δ, i < rqt};
12             end
13         end
           // requested representation is equal or less than the
              previous one
14         else if r^rqt is equal to r^curr or r^rqt < r^curr then
15             if Dtime_n r^curr ≤ maxDtime + δ and atime_n ≤ eptime_{n-1} then
16                 Sq_n=r^curr;                        // stay in the previous quality
17             end
18             else
19                 max { r^i | r^i ∈ ℜ, Dtime_n r^i ≤ maxDtime + δ, i < curr};
20             end
21         end
22     end
23 end
```

bit-rate corresponds not to the average bit-rate as in the previous case, but to the maximum bit-rate attained by the highest quality, applying our DQD algorithm

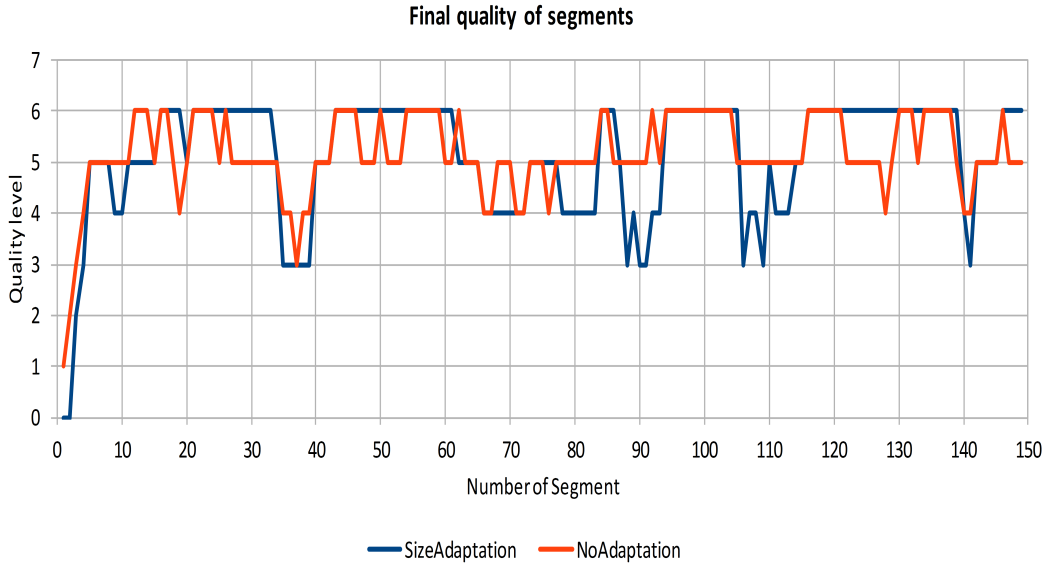


Figure 6.6: Results of appying DQD algorithm on Big Buck Bunny video sequence (bandwidth = 1024.07 Kbit/s).

the quality received at the client side of all video presentation can be improved noticeably. In Figure 6.7, we can see when no bit-rate adaptation is used, there is a great variation between qualities. The total number of up-down switches realized is 48, introducing a delay of 4.212 seconds. While with our approach it is feasible to keep the highest quality during almost all the presentation (0 up-down switches) without producing any delay.

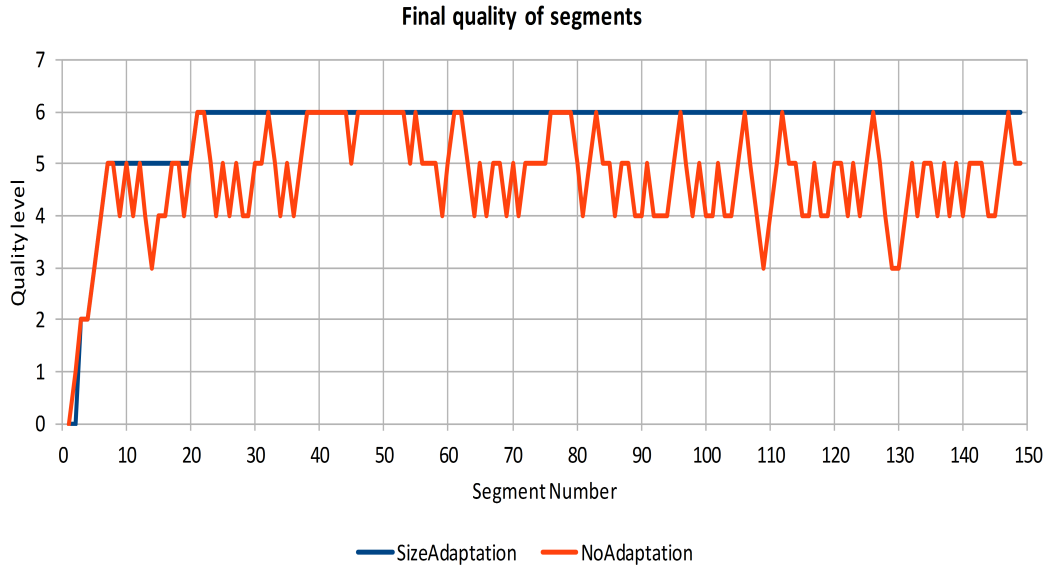


Figure 6.7: Appying DQD algorithm on Big Buck Bunny video sequence (bandwidth = 1650 Kbit/s).

Chapter 7

Conclusions

In order to control the video quality degradation, this thesis presented some approaches to cope with bandwidth variations and packet losses in a streaming scenario. Among the various aspects addressed, the importance of using scalable video coding, and especially the quality scalability mode, is highlighted.

The strategies proposed in this thesis consider the network behavior and the type of application. In such way, for live video delivery over lossy networks, we proposed to use multiple description coding with scalable video coding techniques. The path diversity of MDC increases the possibility that at least one description of the video arrives at the receiver side, in that way its playback is guaranteed. Moreover, the use of SVC allows to improve the quality. It is clear that the quality of the video depends directly on the number of descriptions received. For instance, if only one description is received, the quality is low, but when both descriptions are available, the final quality is high. However, in any case, the video transmitted is not completely lost in case of a drastic decrease of the current bandwidth in one of the paths. On the contrary, the description received can be reconstructed with the same characteristics as the original one, but with lower quality as expected.

In this work, MDC and SVC-MGS were blended in order to increase the flexibility of bit stream adaptation. The two MDC evaluated methods were applied in CIF and HD video sequences. Then, their performance were compared, enabling to define in which cases is better to use each one of these approaches. We conclude that depending on the type of the video sequence, a MDC method can give better results than the other one. However, in case of uncertainty we recommend to apply the MDC-Spatial domain approach.

Another important point addressed in this thesis is the relevance of estimating the current available bandwidth to video transmission. So, a *Bandwidth Estimation Model* based on hidden Markov model was developed. Moreover, the suitable parameters and specifications to its construction were defined. From experiments, it was deduce that the transition probability matrix A can be built using random or equally likely values without affecting the final result. It is because the initial matrix A is generated from guesses. The critical part is the construction of the probability matrix B. The best and consistent performance was obtained when matrix B is created considering a particular pattern. This pattern consists in that large separation times between probing packets (i.e. delays) are most likely generated by low bandwidths and vice versa. Hence, high states corresponding to high bit-rates are more probable to be associated to symbols that represent short delays. On the

other hand, it was decided that a more accurate estimation could be achieved when the same number of states and symbols are used into the model.

Furthermore, from this thesis is concluded the significance of restricting the packet losses produced by bandwidth fluctuations, in order to achieve the best available quality. For doing this, a *Quiality Discard Packets* algorithm was proposed and implemented. Considering the fact that most of the losses occur at the transmission side and no at the nodes inside the network, our QDP algorithm is developed at the server. This algorithm takes place when the source transmission buffer length exceeds a defined threshold. It consists in dropping low priority packets and sending the most important packets fitting the available bandwidth. Our developed algorithm ensures that the transmitted video retains the highest possible quality. The QDP approach is compared with the Random Early Detection algorithm (RED), with a discarding entire layer approach and with the case in which there is no any buffer management mechanism. From simulations, it is evident that our QDP algorithm achieves a better performance (i.e. PSNR), and thus a notable reduction into buffer size requirements and consequently transmission delay is obtained.

Finally, taking advantage of the benefits provided by the combination of DASH and SVC, we realize a SVC-MGS DASH client and server. In order to create the video segments, the suitable encoding configuration jointly with its parameters are defined. Additionally, with the aim of improve the QoE, the *Dash quality decision* (DQD) algorithm is proposed and implemented into the DASH server. The DQD calculates whether it is feasible to delivery on time the current segment into the requested quality. When it is not viable, the DQD algorithm decides which is the segment representation at the possible highest quality, that can be downloaded at the current bandwidth, ensuring the no insertion of awaiting times. Employing our proposed algorithm, we improve the quality of the video presentation reducing both, the number of abrupt up-down switches and the delay introduced.

The section below addresses the principal future works which could be explored starting from this thesis.

7.1 Future works

The work carried out in this thesis open several possibilities to the development of future works. The *QDP* strategy proposed in this thesis does not consider the presence of dependencies among the frames of a SVC encoded bit-stream. This feature of SVC could be exploited to improve even more the video quality obtained, as well as the buffer requirements. Furthermore, to the construction of the Bandwidth estimation model, the initial state probability employed in this thesis, is always state 1. In other words, the distribution of initial states has all of its probability mass concentrated at state 1. However, it is possible to change this behavior and assign a different distribution of probabilities. In the end, the use of adaptive DASH algorithms could be applied in different scenarios, such as multi-hop networks, multimedia sensor networks and video surveillance. Considering a specific scenario, and so application, the QoE in DASH could be enhanced.

Bibliography

[1] Cesar D. Guerrero and Miguel A. Labrador. "Traceband: A fast, low overhead and accurate tool for available bandwidth estimation and monitoring". In: *Computer Networks* 54.6 (2010). New Network Paradigms, pp. 977–990. ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/j.comnet.2009.09.024. URL: http://www.sciencedirect.com/science/article/pii/S1389128609003272.

[2] P. Juluri, V. Tamarapalli, and D. Medhi. "SARA: Segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP". In: *2015 IEEE International Conference on Communication Workshop (ICCW)*. June 2015, pp. 1765–1770. DOI: 10.1109/ICCW.2015.7247436.

[3] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. "Overview of the H.264/AVC video coding standard". In: *IEEE Transactions on Circuits and Systems for Video Technology* 13.7 (July 2003), pp. 560–576. ISSN: 1051-8215. DOI: 10.1109/TCSVT.2003.815165.

[4] Heiko Schwarz and Mathias Wien. "The scalable video coding extension of the H. 264/AVC standard". In: *IEEE Signal Processing Magazine* 25.2 (2008), p. 135.

[5] H. Schwarz, D. Marpe, and T. Wiegand. "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard". In: *IEEE Transactions on Circuits and Systems for Video Technology* 17.9 (Sept. 2007), pp. 1103–1120. ISSN: 1051-8215. DOI: 10.1109/TCSVT.2007.905532.

[6] Iraide Unanue, Iñigo Urteaga, Ronaldo Husemann, Javier Del Ser, Valter Roesler, Aitor Rodríguez, and Pedro Sánchez. "A tutorial on h. 264/svc scalable video coding and its tradeoff between quality, coding efficiency and performance". In: *Recent Advances on Video Coding* 13 (2011).

[7] Amina Kessentini, Imen Werda, Amine Samet, Mohamed Ali Ben Ayed, and Nouri Masmoudi. "H. 264/SVC performance and encoder bit-stream analysis". In: *Proceedings of the International Journal of Computer Applications (0975-8887)* 36.8 (2011).

[8] R. Gupta, A. Pulipaka, P. Seeling, L. J. Karam, and M. Reisslein. "H.264 Coarse Grain Scalable (CGS) and Medium Grain Scalable (MGS) Encoded Video: A Trace Based Traffic and Quality Evaluation". In: *IEEE Transactions on Broadcasting* 58.3 (Sept. 2012), pp. 428–439. ISSN: 0018-9316. DOI: 10.1109/TBC.2012.2191702.

[9] *JSVM Software Manual*. English. Version 9.19.14. 2011. 67 pp.

[10] S. H. Yang and W. L. Tang. "What are good CGS/MGS configurations for H.264 quality scalable coding?" In: *Signal Processing and Multimedia Applications (SIGMAP), 2011 Proceedings of the International Conference on*. July 2011, pp. 1–6.

[11] Burak Görkemli, Yalçın Šadi, and A Murat Tekalp. "Effects of MGS fragmentation, slice mode and extraction strategies on the performance of SVC with medium-grained scalability". In: *2010 IEEE International Conference on Image Processing*. Sept. 2010, pp. 4201–4204. DOI: 10.1109/ICIP.2010.5650771.

[12] Z. Zhao and J. Ostermann. "Video streaming using standard-compatible scalable multiple description coding based on SVC". In: *2010 IEEE International Conference on Image Processing*. Sept. 2010, pp. 1293–1296. DOI: 10.1109/ICIP.2010.5653989.

[13] D. Alfonso, R. Bernardini, L. Celetto, R. Rinaldo, and P. Zontone. "Multiple description for robust Scalable Video Coding". In: *2009 16th IEEE International Conference on Image Processing (ICIP)*. Nov. 2009, pp. 905–908. DOI: 10.1109/ICIP.2009.5414051.

[14] P. Correia, P. A. Assuncao, and V. Silva. "Multiple Description of Coded Video for Path Diversity Streaming Adaptation". In: *IEEE Transactions on Multimedia* 14.3 (June 2012), pp. 923–935. ISSN: 1520-9210. DOI: 10.1109/TMM.2011.2182184.

[15] Mohammad Kazemi, Shervin Shirmohammadi, and Khosrow Haj Sadeghi. "A review of multiple description coding techniques for error-resilient video delivery". In: *Multimedia Systems* 20.3 (2014), pp. 283–309.

[16] H. Mansour, P. Nasiopoulos, and V. Leung. "An Efficient Multiple Description Coding Scheme for the Scalable Extension of H.264/AVC (SVC)". In: *2006 IEEE International Symposium on Signal Processing and Information Technology*. Aug. 2006, pp. 519–523. DOI: 10.1109/ISSPIT.2006.270856.

[17] *Kush Gauge equation is available in:* https://www.adobe.com/content/dam/Adobe/en/devnet/video/articles/h264_primer/h264_primer.pdf.

[18] C. D. Guerrero and M. A. Labrador. "A Hidden Markov Model approach to available bandwidth estimation and monitoring". In: *Internet Network Management Workshop, 2008. INM 2008. IEEE*. Oct. 2008, pp. 1–6. DOI: 10.1109/INETMW.2008.4660326.

[19] Shilpa Shashikant Chaudhari and Rajashekhar C. Biradar. "Survey of Bandwidth Estimation Techniques in Communication Networks". In: *Wirel. Pers. Commun.* 83.2 (July 2015), pp. 1425–1476. ISSN: 0929-6212. DOI: 10.1007/s11277-015-2459-2. URL: http://dx.doi.org/10.1007/s11277-015-2459-2.

[20] D. Xu and D. Qian. "A bandwidth adaptive method for estimating end-to-end available bandwidth". In: *Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on*. Nov. 2008, pp. 543–548. DOI: 10.1109/ICCS.2008.4737243.

[21] Manish Jain and Constantinos Dovrolis. "Pathload: A Measurement Tool for End-to-End Available Bandwidth". In: *In Proceedings of Passive and Active Measurements (PAM) Workshop*. 2002, pp. 14–25.

[22] Vinay Joseph Ribeiro, Rudolf H Riedi, Richard G Baraniuk, Jiri Navratil, and Les Cottrell. "pathchirp: Efficient available bandwidth estimation for network paths". In: *Passive and active measurement workshop*. 2003.

[23] Manish Jain and Constantinos Dovrolis. "End-to-end Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput". In: *IEEE/ACM Trans. Netw.* 11.4 (Aug. 2003), pp. 537–549. ISSN: 1063-6692. DOI: 10.1109/TNET.2003.815304. URL: http://dx.doi.org/10.1109/TNET.2003.815304.

[24] Cesar Dario Guerrero Santander. "End-to-end available bandwidth estimation and monitoring". PhD thesis. UNIVERSITY OF SOUTH FLORIDA, 2009.

[25] Jacob Strauss, Dina Katabi, and Frans Kaashoek. "A Measurement Study of Available Bandwidth Estimation Tools". In: *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*. IMC '03. Miami Beach, FL, USA: ACM, 2003, pp. 39–44. ISBN: 1-58113-773-7. DOI: 10.1145/948205.948211. URL: http://doi.acm.org/10.1145/948205.948211.

[26] Ningning Hu and P. Steenkiste. "Evaluation and characterization of available bandwidth probing techniques". In: *IEEE Journal on Selected Areas in Communications* 21.6 (Aug. 2003), pp. 879–894. ISSN: 0733-8716. DOI: 10.1109/JSAC.2003.814505.

[27] C. D. Guerrero and M. A. Labrador. "Experimental and Analytical Evaluation of Available Bandwidth Estimation Tools". In: *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*. Nov. 2006, pp. 710–717. DOI: 10.1109/LCN.2006.322181.

[28] James R Jackson. "Jobshop-like queueing systems". In: *Management science* 10.1 (1963), pp. 131–142.

[29] Tapas Kanungo. *Hmm toolkit*. 1999. URL: http://www.kanungo.com/%20software/software.html.

[30] A. K. Paul, A. Tachibana, and T. Hasegawa. "NEXT: New enhanced available bandwidth measurement technique, algorithm and evaluation". In: *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*. Sept. 2014, pp. 443–447. DOI: 10.1109/PIMRC.2014.7136206.

[31] A. K. Paul, A. Tachibana, and T. Hasegawa. "An Enhanced Available Bandwidth Estimation Technique for an End-to-End Network Path". In: *IEEE Transactions on Network and Service Management* 13.4 (Dec. 2016), pp. 768–781. ISSN: 1932-4537. DOI: 10.1109/TNSM.2016.2572212.

[32] Arnaud Legout. *Dataset video streaming sessions*. 2011. URL: http://www-sop.inria.fr/members/Arnaud.Legout/Projects/streaming.html.

[33] *Youtube traces from the campus network (UMass Trace Repository)*. 2008. URL: http://traces.cs.umass.edu/index.php/Network/Network.

[34] L. R. Rabiner. “A tutorial on hidden Markov models and selected applications in speech recognition”. In: *Proceedings of the IEEE* 77.2 (Feb. 1989), pp. 257–286. ISSN: 0018-9219. DOI: 10.1109/5.18626.

[35] A. B. Poritz. “Hidden Markov models: a guided tour”. In: *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*. Apr. 1988, 7–13 vol.1. DOI: 10.1109/ICASSP.1988.196495.

[36] Anand Balan, Omesh Tickoo, Ivan Bajic, Shivkumar Kalyanaraman, and John Woods. “Integrated buffer management and congestion control for video streaming”. In: *Proc. of the 10th IEEE International Conference on Image Processing, Barcelona, Spain*. 2003.

[37] Zigmund Orlov and Marc C Necker. “Enhancement of video streaming QoS with active buffer management in wireless environments”. In: *In European Wireless Conference*. 2007.

[38] G. Thiruchelvi and J. Raja. “Survey on Active Queue Management Mechanisms”. In: *IJCSNS International Journal of Computer Science and Network Security, Vol.8, No.12*. 2008, pp. 130–145.

[39] M. Masry and S. S. Hemami. “An analysis of subjective quality in low bit rate video”. In: *Image Processing, 2001. Proceedings. 2001 International Conference on*. Vol. 1. 2001, 465–468 vol.1. DOI: 10.1109/ICIP.2001.959054.

[40] I. V. Bajic, O. Tickoo, A. Balan, S. Kalyanaraman, and J. W. Woods. “Integrated end-to-end buffer management and congestion control for scalable video communications”. In: *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*. Vol. 3. Sept. 2003, III-257-60 vol.2. DOI: 10.1109/ICIP.2003.1247230.

[41] Eren Gürses, G Bozdagi Akar, and Nail Akar. “Selective frame discarding for video streaming in TCP/IP networks”. In: *Packet Video Workshop, Nantes, France (April 2003)*. 2003.

[42] Y. Zhang, W. Huangfu, J. Xu, K. Li, and C. Xu. “Integrated Rate Control and Buffer Management for Scalable Video Streaming”. In: *2007 IEEE International Conference on Multimedia and Expo*. July 2007, pp. 248–251. DOI: 10.1109/ICME.2007.4284633.

[43] Y. Xiaogang, L. Jiqiang, and L. Ning. “Congestion Control Based on Priority Drop for H.264/SVC”. In: *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE’07)*. Apr. 2007, pp. 585–589. DOI: 10.1109/MUE.2007.107.

[44] A. Palawan, J. Woods, and M. Ghanbari. “Weighted multi-playback buffer management for scalable video streaming”. In: *Computer Science and Electronic Engineering Conference (CEEC), 2014 6th*. Sept. 2014, pp. 47–51. DOI: 10.1109/CEEC.2014.6958553.

[45] S. E. Ghoreishi, A. H. Aghvami, and M. G. Martini. “Perceptual quality-aware active queue management for video transmission”. In: *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2015 IEEE 26th Annual International Symposium on*. Aug. 2015, pp. 1267–1271. DOI: 10.1109/PIMRC.2015.7343493.

[46] Thomas Schierl, Gunther Liebl, Thomas Stockhammer, and Thomas Wiegand. "Advancedwireless Multiuser Video Streaming using the Scalable Video Coding Extensions of H.264/MPEG4-AVC". In: *2006 IEEE International Conference on Multimedia and Expo* 00.undefined (2006), pp. 625–628. DOI: doi.ieeecomputersociety.org/10.1109/ICME.2006.262486.

[47] Iraj Sodagar. "The MPEG-DASH Standard for Multimedia Streaming Over the Internet". In: *IEEE MultiMedia* 18.4 (Oct. 2011), pp. 62–67. ISSN: 1070-986X. DOI: 10.1109/MMUL.2011.71. URL: http://dx.doi.org/10.1109/MMUL.2011.71.

[48] Wilder E. Castellanos, Juan C. Guerri, and Pau Arce. "SVCEval-RA: an evaluation framework for adaptive scalable video streaming". In: *Multimedia Tools and Applications* 76.1 (2017), pp. 437–461. ISSN: 1573-7721. DOI: 10.1007/s11042-015-3046-y. URL: http://dx.doi.org/10.1007/s11042-015-3046-y.

[49] Anand Balan, Ivan V Bajic, Omesh Tickoo, Shivkumar Kalyanaraman, and John W Woods. "Integrated End-End Buffer Management and Congestion Control for Video Communications". In: *in: International Conference on Image Processing*. Citeseer.

[50] *FFMPEG official website:* http://ffmpeg.org/.

[51] T. L. Lin, Y. L. Chang, and P. C. Cosman. "Subjective experiment and modeling of whole frame packet loss visibility for H.264". In: *2010 18th International Packet Video Workshop*. Dec. 2010, pp. 186–192. DOI: 10.1109/PV.2010.5706837.

[52] Michael Grafl and Christian Timmerer. "Representation switch smoothing for adaptive HTTP streaming". In: *In Proceedings of the 4th International Workshop on Perceptual Quality of Systems (PQS 2013)*. Sept. 2013, pp. 178–183.

[53] S. Floyd and V. Jacobson. "Random early detection gateways for congestion avoidance". In: *IEEE/ACM Transactions on Networking* 1.4 (Aug. 1993), pp. 397–413. ISSN: 1063-6692. DOI: 10.1109/90.251892.

[54] Sally Floyd. "RED: Discussions of Setting Parameters". In: (1997).

[55] C. M. Patel. "URED: Upper threshold RED an efficient congestion control algorithm". In: *Computing, Communications and Networking Technologies (ICCCNT),2013 Fourth International Conference on*. July 2013, pp. 1–5. DOI: 10.1109/ICCCNT.2013.6726469.

[56] Sandvine. *Global Internet Phenomena Report. Technical Report 1H,2014*. Tech. rep. 2014.

[57] *Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats*. Standard. International Organization for Standardization, 2014.

[58] Christian Kreuzberger, Daniel Posch, and Hermann Hellwagner. "A Scalable Video Coding Dataset and Toolchain for Dynamic Adaptive Streaming over HTTP". In: *Proceedings of the 6th ACM Multimedia Systems Conference*. MMSys '15. Portland, Oregon: ACM, 2015, pp. 213–218. ISBN: 978-1-4503-3351-1. DOI: 10.1145/2713168.2713193. URL: http://doi.acm.org/10.1145/2713168.2713193.

[59] Yago Sánchez de la Fuente, Thomas Schierl, Cornelius Hellge, Thomas Wiegand, Dohy Hong, Danny De Vleeschauwer, Werner Van Leekwijck, and Yannick Le Louédec. "iDASH: Improved Dynamic Adaptive Streaming over HTTP Using Scalable Video Coding". In: *Proceedings of the Second Annual ACM Conference on Multimedia Systems*. MMSys '11. San Jose, CA, USA: ACM, 2011, pp. 257–264. ISBN: 978-1-4503-0518-1. DOI: 10.1145/1943552.1943586. URL: http://doi.acm.org/10.1145/1943552.1943586.

[60] Y. Sanchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, and Y. Le Louédec. "Efficient HTTP-based Streaming Using Scalable Video Coding". In: *Image Commun.* 27.4 (Apr. 2012), pp. 329–342. ISSN: 0923-5965. DOI: 10.1016/j.image.2011.10.002. URL: http://dx.doi.org/10.1016/j.image.2011.10.002.

[61] Christian Timmerer and Michael Grafl. *SVC Bitstream Reordering for SVC-DASH*. Standard. Geneva, CH, Oct. 2013.

[62] C. Mueller, S. Lederer, J. Poecher, and C. Timmerer. "Demo paper: Libdash - An open source software library for the MPEG-DASH standard". In: *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*. July 2013, pp. 1–2. DOI: 10.1109/ICMEW.2013.6618220.

[63] *Video download rate traces (DISEDAN project)*. URL: http://wp2.tele.pw.edu.pl/disedan/software/traces.

[64] C. Sieber, T. Hoßfeld, T. Zinner, P. Tran-Gia, and C. Timmerer. "Implementation and user-centric comparison of a novel adaptation logic for DASH with SVC". In: *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. May 2013, pp. 1318–1323.

[65] T. Hoßfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz. "Quantification of YouTube QoE via Crowdsourcing". In: *Multimedia (ISM), 2011 IEEE International Symposium on*. Dec. 2011, pp. 494–499. DOI: 10.1109/ISM.2011.87.

[66] T. Hossfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen. "Initial delay vs. interruptions: Between the devil and the deep blue sea". In: *Quality of Multimedia Experience (QoMEX), 2012 Fourth International Workshop on*. July 2012, pp. 1–6. DOI: 10.1109/QoMEX.2012.6263849.

[67] Michael Zink, Oliver Künzel, Jens Schmitt, and Ralf Steinmetz. "Subjective impression of variations in layer encoded videos". In: *international Workshop on Quality of Service*. Springer. 2003, pp. 137–154.

[68] Simon Oechsner, Thomas Zinner, Jochen Prokopetz, and Tobias Hoßfeld. "Supporting scalable video codecs in a P2P video-on-demand streaming system". In: *Proc. 21st ITC Specialist Seminar*. Citeseer. 2010.

[69] Johan A Pouwelse, Pawel Garbacki, Jun Wang, Arno Bakker, Jie Yang, Alexandru Iosup, Dick HJ Epema, Marcel Reinders, Maarten R Van Steen, Henk J Sips, et al. "Tribler: A social-based peer-to-peer system". In: *Concurrency and computation: Practice and experience* 20.2 (2008), p. 127.

[70] Arnaud Legout, Guillaume Urvoy-Keller, and Pietro Michiardi. "Rarest first and choke algorithms are enough". In: *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. ACM. 2006, pp. 203–216.

[71] Yuming Cao, Xiaoquan You, Jia Wang, and Li Song. "A QoE friendly rate adaptation method for DASH". In: *2014 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*. IEEE. 2014, pp. 1–6.

[72] Dongeun Suh, Insun Jang, and Sangheon Pack. "QoE-enhanced adaptation algorithm over DASH for multimedia streaming". In: *The International Conference on Information Networking 2014 (ICOIN2014)*. IEEE. 2014, pp. 497–501.

[73] Ricky KP Mok, Xiapu Luo, Edmond WW Chan, and Rocky KC Chang. "QDASH: a QoE-aware DASH system". In: *Proceedings of the 3rd Multimedia Systems Conference*. ACM. 2012, pp. 11–22.

[74] Ukheon Jeong and Kwangsue Chung. "Video Quality Adaptation to Improve The Quality of Experience in DASH Environments". In: *International Journal of Computer science and network security IJCSNS* 14.8 (2014), pp. 22–29.

[75] Chenghao Liu, Imed Bouazizi, and Moncef Gabbouj. "Rate Adaptation for Adaptive HTTP Streaming". In: *Proceedings of the Second Annual ACM Conference on Multimedia Systems*. MMSys '11. San Jose, CA, USA: ACM, 2011, pp. 169–174. ISBN: 978-1-4503-0518-1. DOI: 10.1145/1943552.1943575. URL: http://doi.acm.org/10.1145/1943552.1943575.

[76] Junchen Jiang, Vyas Sekar, and Hui Zhang. "Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE". In: *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*. CoNEXT '12. Nice, France: ACM, 2012, pp. 97–108. ISBN: 978-1-4503-1775-7. DOI: 10.1145/2413176.2413189. URL: http://doi.acm.org/10.1145/2413176.2413189.

[77] P. Juluri, V. Tamarapalli, and D. Medhi. "QoE management in DASH systems using the segment aware rate adaptation algorithm". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*. Apr. 2016, pp. 129–136. DOI: 10.1109/NOMS.2016.7502805.

[78] H. K. Yarnagula, S. Luhadia, S. Datta, and V. Tamarapalli. "Quality of experience assessment of rate adaptation algorithms in DASH: An experimental study". In: *2016 8th International Conference on Communication Systems and Networks (COMSNETS)*. Jan. 2016, pp. 1–8. DOI: 10.1109/COMSNETS.2016.7440008.