ROBUST P2P LIVE STREAMING

AUTHOR: Alberto José González Cela

DIRECTOR: Jesús Alcober Segura

DEPARTMENT: Telematics Engineering

DATE: October, 2009

Title: Robust P2P Live Streaming

Author: Alberto José González Cela

Director: Jesús Alcober Segura

Date: October, 2009

Abstract

The provisioning of robust real-time communication services (voice, video, etc.) or media contents through the Internet in a distributed manner is an important challenge, which will strongly influence in current and future Internet evolution. Aware of this, we are developing a project named Trilogy leaded by the i2CAT Foundation, which has as main pillar the study, development and evaluation of Peer-to-Peer (P2P) Live streaming architectures for the distribution of high-quality media contents. In this context, this work concretely covers media coding aspects and proposes the use of Multiple Description Coding (MDC) as a flexible solution for providing robust and scalable live streaming over P2P networks. This work describes current state of the art in media coding techniques and P2P streaming architectures, presents the implemented prototype as well as its simulation and validation results.

INDEX

CHAPTER 1. INTRODUCTION 1

1.1 Objectives 2

1.2 Tasks specification 2

CHAPTER 2. STATE OF THE ART 3

2.1 Peer-to-Peer architectures for media streaming 6

2.1.1 Types of overlay 8

2.2 Video Coding Techniques 9

2.2.1 Multiple Description Coding (MDC) 10

2.2.2 Scalable Video Coding (SVC) 16

CHAPTER 3. BACKGROUND 19

3.1 CoolRuc Architecture 19

CHAPTER 4. DESIGN AND IMPLEMENTATION 23

4.1 MDC Techniques 23

4.1.1 Spatial MDC 24

4.1.2 Temporal MDC 24

4.1.3 Hybrid MDC 25

4.2 MDC System Overview 26

4.3 Transmission Module Component: the Splitter 28

4.3.1 Splitter Module Configuration - Inputs 31

4.4 Reception Module Component: the Merger 31

4.4.1 Merger Module Configuration - Inputs 34

4.5 Software Specification of the Video Coding Component 34

4.5.1 Splitter Module 34

4.5.2 Merger Module 35

4.5.3 Software Components Architecture 36

CHAPTER 5. METHODOLOGY AND EXPERIMENTAL RESULTS 39

5.1 MDC Simulation Results 39

5.1.1 MDC simulation with MPEG-4 codec 40
5.1.2 MDC simulation without codec 41
5.1.3 Results 42
5.2 Point-to-Point Prototype Evaluation Results 44
5.2.1 Hardware Specification 44
5.2.2 Software requirements 45
5.2.3 Test description 45
5.2.4 Results 46
5.2.5 Conclusion 49
5.3 P2P Integrated System Evaluation Results 50
5.3.1 Testbed 50
5.3.2 Metrics 51
5.3.3 Parameters of the experiment 52
5.3.4 Results 53
CHAPTER 6. CONCLUSIONS AND FUTURE WORK 55
6.1 Realized work 55
6.2 Work conclusions 56
6.3 Future work 58
REFERENCES 61
ANNEX A. Fundació i2CAT and TRILOGY Project Overview 65
ANNEX B. Connectivity 67
ANNEX C. MDC vs. LC Comparative 68
ANNEX D. Example of MDC system 71
ANNEX E. H.264 AVC overview 73
ANNEX F. Types of NAL units 75
ANNEX G. RTP Payload for SVC 76
ANNEX H. Simulation commands 77
ANNEX I. Simulation Results with MPEG-4 Codec 78

ANNEX J. Simulation Results without Codec 84
ANNEX K. P2P Results 97
ANNEX L. Used Software 98
ANNEX M. Generated Publications 100

FIGURE INDEX

Figure 2.1 Example of poly-phase down-sampling system [30] 11

Figure 2.2 MDC-STHI [32] 12

Figure 2.3 MDC framework [33] 13

Figure 2.4 Multiple Description Coding based on RDWT 13

Figure 2.5 Wavelet decomposition 14

Figure 2.6 Wavelet filter bank structure 14

Figure 2.7 Scalable Video Coding (SVC) general process 16

Figure 3.1 CoolRuc architecture of a node 20

Figure 4.1 Spatial MDC 24

Figure 4.2 Temporal MDC 25

Figure 4.3 Hybrid MDC 25

Figure 4.4 MDC transmission scheme 26

Figure 4.5 MDC reception scheme 26

Figure 4.6 Meta-data publication 27

Figure 4.7 YUV example 28

Figure 4.8 MDC Sender components (Splitter Module) 29

Figure 4.9 MDC Packet format 30

Figure 4.10 Hexadecimal representation of an I frame coded with MPEG-2 30

Figure 4.11 Merger general scheme 31

Figure 4.12 Receiver components (Merger Module) 32

Figure 4.13 Segment Format 32

Figure 4.14 Interpolation by previous and closest pixel copy in a frame 33

Figure 4.15 Stream to mobile phone (Merger Module Output) 34

Figure 4.16 Splitter thread state chart 35

Figure 4.17 Splitter main thread state chart 35

Figure 4.18 Merger thread state chart 35

Figure 4.19 Merger main thread state chart 36
Figure 4.20 MDC Video Coding Components, software block diagram 37
Figure 5.1 Simulation scenario components 40
Figure 5.2 Simulation scenario 41
Figure 5.3 LAN Scenario 46
Figure 5.4 Physical topology of the test scenario 50
Figure 5.5 Statistics generation process 51
Figure C. 1 MD 2 and SD Reed-Solomon FEC. 70
Figure D. 1 Scheme of a MDC system 72
Figure G. 1 NAL Unit encapsulation into RTP 76
Figure J. 1 PSNR vs. loss, Foreman, temporal MDC 85
Figure J. 2 PSNR vs. loss, Akiyo, temporal MDC 85
Figure J. 3 PSNR vs. loss, Carphone, temporal MDC 86
Figure J. 4 PSNR vs. loss, Foreman, spatial MDC 87
Figure J. 5 PSNR vs. loss, Akiyo, spatial MDC 88
Figure J. 6 PSNR vs. loss, Carphone, spatial MDC 88
Figure J. 7 Spatial vs. Temporal, 2 descriptors, Foreman 89
Figure J. 8 Spatial vs. Temporal, 4 descriptors, Foreman 90
Figure J. 9 Spatial vs. Temporal, 8 descriptors, Foreman 90
Figure J. 10 Spatial vs. Temporal, 16 descriptors, Foreman 91
Figure J. 11 Spatial vs. Temporal, 24 descriptors, Foreman 91
Figure J. 12 Spatial vs. Temporal, 2 descriptors, Akiyo 92
Figure J. 13 Spatial vs. Temporal, 4 descriptors, Akiyo 92
Figure J. 14 Spatial vs. Temporal, 8 descriptors, Akiyo 93
Figure J. 15 Spatial vs. Temporal, 16 descriptors, Akiyo 93
Figure J. 16 Spatial vs. Temporal, 24 descriptors, Akiyo 94
Figure J. 17 Spatial vs. Temporal, 2 descriptors, Carphone 94
Figure J. 18 Spatial vs. Temporal, 4 descriptors, Carphone 95

Figure J. 19 Spatial vs. Temporal, 8 descriptors, Carphone 95

Figure J. 20 Spatial vs. Temporal, 16 descriptors, Carphone 96

Figure J. 21 Spatial vs. Temporal, 24 descriptors, Carphone 96

TABLE INDEX

Table 1.1. Planned tasks 2

Table 2.1 A taxonomy of typical Peer-to-peer Applications [15] 7

Table 4.1 Splitter input parameters 31

Table 4.2 Merger input arguments 34

Table 5.1 Characteristics of the video used and details for the performed tests 46

Table 5.2 RAM consumption for Splitter 47

Table 5.3 RAM consumption for merger 47

Table 5.4 CPU consumption for Splitter 47

Table 5.5 CPU consumption for Merger 48

Table 5.6 Resolution testing 49

Table 5.7 Parameters of the experiment 52

Table 5.8 Fixed values 53

Table D. 1 Blocks composing the MDC system 71

Table F. 1 Original NAL unit types 75

Table K. 1 Single Description 97

Table K. 2 Multiple Description Coding 97

ACKNOWLEDGEMENTS

This work would not have been possible without the support of many people. The author wishes to express his gratitude to his supervisor, Dr. Jesus Alcober (UPC-MediaEntel) for the opportunity of working in this issue in the context of the Trilogy Project of the i2CAT Foundation. Thanks also to André Ríos (UPC-MediaEntel) who was helpful and offered support and guidance. Deepest gratitude are also due to the members of the Trilogy Project, José Francisco Crespo (UPC-LCFIB), Sergi Laencina (UPC-MediaEntel) and Guillermo Enero (UPC-MediaEntel) without whose knowledge and assistance this study would not have been successful.

Special thanks also to all project partners, especially group members URL-LaSalle and UPF-Nets, and also to Dr. David Rincón (UPC) and Dr. Francesc Tarrés (UPC) for sharing literature and invaluable assistance.

The author would also like to convey thanks to the i2CAT Foundation for providing the financial means, laboratory facilities and support.

The author wishes to express his love and gratitude to his beloved parents; for their understanding and endless love, through the duration of his studies and professional activities.

Finally, Laura, I'll never be able to thank you enough for all your love, support, help and guidance, specially during all this rough times.

Not forgetting to his bestfriends who always been there.

CHAPTER 1. INTRODUCTION

Peer to Peer (P2P) technology has been successfully and widely deployed in many areas over the Internet, from one-to-one communications such as Voice over IP (VoIP) and Instant Messaging (IM) applications to one-to-many communications such as file-sharing, gaming and streaming.

Nowadays, the impact of P2P traffic is growing fast and supposes the major part of current Internet traffic. However, today, video streaming applications are the bandwidth most-hungry applications. In the streaming area, the popularity of P2P real-time communications has rapidly grown. Despite the massive usage of these kinds of applications and growing interest (both in academic research and commercial level) Internet real-time P2P streaming applications still present open issues.

A big challenge for current applications is dealing with network access and terminal heterogeneity whose annoying effects degrades the quality of experience of a user and, sometimes, makes the user to abandon the platform. In this context, this master thesis gives a novel solution in order to improve live media streaming communications in P2P environments, characterized by intrinsic harsh conditions. Concretely, it tries to cope with problems derived from heterogeneity and loss effect by providing a robust and scalable video coding technique. Multiple Description Coding (MDC) is proposed as error-resilient source coding scheme that can be used to reduce the detrimental effects caused by packet loss on best-effort networks.

The proposed solution was integrated in a P2P streaming application called CoolRuc (see CHAPTER 3) which was developed in the TRILOGY Project of the Fundació I2CAT [11] (see ANNEX A). Concretely, MDC Splitter and MDC Merger modules were developed, which are part of the Video Coding component in the architecture of a CoolRuc node.

This master thesis presents in CHAPTER 2 the State of the Art, focusing on two media coding techniques: MDC and SVC. CHAPTER 3 presents the Architecture of a CoolRuc node as background to understand where is contextualized the presented solution. Then, CHAPTER 4 shows design and implementation issues. Moreover, CHAPTER 5, presents simulation and evaluation results of the proposed solution. Finally, conclusions and future work can be seen in CHAPTER 6.

Several annexes containing supporting information detailing specific topics mentioned in this work are also provided.

1.1 Objectives

A first objective is to analyse current state of the art in Media Coding Schemes suitable for P2P environments.

Secondly, it will be proposed a design for a robust, flexible and scalable solution for high quality media transmission over P2P networks in live mode by means of using robust techniques such as Multiple Description Coding (MDC) or Scalable Video Coding (SVC). This solution will be then validated through simulation.

Third, a prototype will be implemented in order to verify its feasibility.

Finally, the proposed system will be integrated and validated into the CoolRuc P2P Streaming platform (software desktop application) developed within the TRILOGY Project, supported by the Fundació i2CAT.

1.2 Tasks specification

The main tasks considered in this master thesis were (in chronological order):

Table 1.1. Planned tasks

Task	Description
Background analysis (state of the art)	Different P2P distribution mechanisms and video coding schemes will be analysed.
Design	1) Critical evaluation of the studied techniques and selection of the candidates to be analysed and implemented. 2) Simulation of the selected techniques in order to evaluate and validate the design. 3) It will be proposed an initial design for the whole system, indicating its main components.
Implementation	1) Implementation of a Point to Point prototype 2) Integration into P2P platform (CoolRuc) and validation
Documentation	Documentation of the tasks

CHAPTER 2. STATE OF THE ART

In the audiovisual streaming area, the popularity of P2P real-time and Video on Demand (VoD) streaming applications such as PPLive [1], PPStream [2], UUSee [3], Pando[4], Zattoo [5] has been demonstrated. As an example, PPLive has registered over 110 million users, 2 million users concurrently connected, offers more than 600 channels and has users in more than 200 countries. In addition, Youtube [6], a worldwide well-known web 2.0 streaming applications, is preparing to use P2P computing in order to improve its download rate while reducing transmission costs (especially interesting in current financial crisis time).

Nowadays, P2P supposes the major part of current Internet traffic (more than 60% of Internet traffic in 2006 was consumed by P2P file-sharing applications [7]). In addition, regarding to high-consuming P2P video applications, statistics in one of the biggest Chinese Internet Service Providers (ISP) show that PPLive accounts 10% of the total Internet backbone traffic, even more than file-sharing (Bittorrent [8] represents the 8% of the traffic). Some studies [9] manifest that streaming is taking over P2P users for video content. So, video and peer-to-peer contents are both rapidly increasing Internet bandwidth demands. Recent reports predict an “exaflood” [10] from advances in video over the Internet, rich media content, and User Generated Content (UGC). Moreover, it is expected that by 2013, the sum of all forms of video (TV, VoD, Internet video, and P2P) will exceed 90 percent of global consumer IP traffic. In that sense, new systems and studies to optimize future P2P and video traffic may have a very high impact on the future of the Internet.

However, live streaming introduces new challenging problems different to ordinary file-sharing. In general, media streaming solutions have different features that determine the operation of the applications. For example: the large volume of media data along with stringent timing constraints, the dynamic and heterogeneous nature of P2P networks and the unpredictable behaviour of peers. These effects can degrade the Quality of Experience of a user and, sometimes, makes the user to leave the platform.

Despite this work is focused on media coding schemes, there are several issues in multimedia P2P streaming that must be taken into consideration when facing these kinds of applications. Some of them are the following ones:

- **Managing peer dynamicity (churn)**: Since the peers (network nodes) are end users terminal, their behaviour remains unpredictable. Due to dynamic nature of

P2P networks, they are free to join and leave the service at any time without making any prior notification to other nodes. Thus, dynamicity management is crucial for the smooth play back rate during streaming session.

- **Peer heterogeneity**: Peers are heterogeneous in their capabilities. At network level, this heterogeneity may be caused either by different access networks (FTTH, ADSL, WiFi, UMTS/3G, etc.) connecting the peers, or by difference in the willingness of the peers to contribute. Each sender peer can have different available bandwidth and that might fluctuate after the connection is established. Another important issue at device level is that future applications may support different types of terminals (Desktop PC, laptop PC, netbook PC, PDA, mobile, etc.) each one with very different display capabilities such as resolution (Full HD monitor, HD Ready TV, SD, CIF, etc). However, all of these applications deliver just a single quality or, in order to access through different terminals, the user must use different services or applications. Currently, P2P live streaming applications offer to users limited quality of the distributed media (order of 1Mbps), and they are still very far from supporting true High Definition (HD) quality.
- **Efficient overlay network construction**: The objective is to organize participating peers into a logical topology that must infer the underlying topology. In fact, a non-suitable overlay topology can result in extra overhead and can reduce the system performance drastically. The overlay construction should be scalable.
- **Selection of the best peers**: An efficient and flexible strategy must be introduced for the selection of sender peers and intermediate peer. In fact, another feature that must be captured in a streaming multimedia system is minimizing end-to-end delay performance metric where keeping the global overhead reasonable. In fact, the less this delay is, the more live the multimedia content is.
- **Monitoring of network conditions**: The network condition during streaming phase can be changed dramatically due to the dynamic nature of P2P architecture
- **Incentives for participating peers**: In many studies, it is found that many peers join the P2P network to benefit from share other's resources (more often data content) but they never share their own resources (bandwidth). The issue

can be resolve by offering some incentives to peers participating in streaming mechanism.

- **Appropriate video coding scheme**: The nature of multimedia content makes it highly sensible to the transmission over networks offering nonguaranteed transmission. Therefore, a reliable multimedia transmission system must involve a reliable video coding scheme. The use of an appropriate video coding scheme is more than essential, such a scheme must be sufficient flexible to meet the P2P network dynamics and its heterogeneity.

According to this, P2P streaming applications should be able to deal well with heterogeneity in order to properly work in different types of networks, support different devices and adapt well to high dynamic environments. In addition it must be able to self-adapt in presence of losses or when context changes (for instance when a user who is watching a TV film in its desktop computer wants to continue watching it in its mobile, this is known as session mobility).

In order to solve that, this work is focused on video coding schemes. Concretely, it describes the two major techniques for video coding in this type of P2P environments for distributing media contents: **Multiple Description Coding (MDC)** and **Scalable Video Coding (SVC)**, also known as Layered Coding (LC).

MDC and SVC are useful in the case of varying bandwidth and losses or erasures due to congestion (e.g. Internet) and unrecoverable errors (such as wireless channels). Scalable Video Coding provides a scalable representation that enhances rate control but it is sensitive to transmission losses. On the other hand, Multiple Description Coding provides increased resilience to packet losses by creating multiple streams that can be decoded independently.

In terms of operation, both schemes split the video resource into many descriptions/layers where each description/layer can contribute to the definition of one or more characteristics of multimedia data. Each descriptor can be encoded in a different way (i.e. Different resolution or bitrate) in an unbalanced approximation or, in contrast, all descriptors can have the same weight following a balanced approach. Then, the nodes wishing to visualize the media (receivers) just need to get the amount of data they support (according to its device capabilities) and can support (according to network access type and current capacity) at any time. The quality of the received video improves with each received description/layer, but the loss of any one of these descriptions/layers does not cause complete failure. If one of the streams is lost or

delivered late, the video playback can continue with only a slight reduction in overall quality.

The difference between MDC and SVC lies in the dependency among description/layer. In the case of SVC, layers are referred to as "base layer" and "enhancement layers". The base layer is the most important layer while the enhancement layers are referenced to the base layer. Enhancement layers cannot be decodable independently to base layer. In contrast to SVC, in MDC each description can be decoded individually to get the base quality. However, if more descriptions are acquired and decoded, the video distortion can be reduced and the larger is the output signal quality.

These features are of special interest, specially focusing on the provisioning on media resources with different requirements among heterogeneous peers and networks, including future next High Definition content or 3DTV.

2.1 Peer-to-Peer architectures for media streaming

Media streaming applications can be classified according to the delay tolerance just as shown in [12]. Real-time applications need to have low delay tolerance because of the interaction between the end-to-end users (no longer than a 150ms is accepted [13]. in order to achieve fluid interactivity. According to this, we made some research contributions during the realization of this master thesis as shown in [13][14][15]. Here the authors propose a novel P2P multiconference system. However, live broadcast applications typically have no interactivity requirements and, consequently, longer delays are tolerated, commonly up to 30 seconds. This delay cannot be detected without interactivity or without a reference point. In the end, on-demand media applications present greater delay tolerance because the existent interactivity is limited to change the channel or due to VCR-like control.

Video streaming has the following main constraints:

- **Scalability**: The system must scale according to the number of users who are connected to the service (VoD and live streaming service).
- **Bandwidth constraint**: The video streaming rate should not exceed the channel capacity.
- **Real-time constraint**: The delay in video packet delivery should not exceed the play-out deadline of a video frame at reception time.

- **Quality of Service (QoS)**: Must guarantee a minimum decoded video quality and a maximum transmission error rate over the duration of the streaming session despite the variation in channel conditions.

The above characteristics combined yield a unique application scenario that differs from other typical peer-to-peer applications, including on-demand streaming, audio/video conferencing, and file download (see Table 2.1).

Table 2.1 A taxonomy of typical Peer-to-peer Applications [15]

Category	Bandwidth-sensitive	Delay-sensitive	Scale
File download	No	No	Large
On-demand streaming	Yes	Yes	Large
Audio/video conferencing	Yes / No	Yes	Small
Live broadcast	Yes	Yes	Large

The key problem in a peer-to-peer video broadcast system consists on organizing the peers into an overlay for disseminating a video stream. The main criteria to be considered regarding to overlay construction and maintenance operations can be found in the following list:

- **Overlay efficiency**: The construction of the overlay network must be efficient, because the streaming video requires high bandwidth and low latencies. However, if these applications are not interactive then, a start-up delay can be tolerated.
- **Scalability and Load Balancing**: The overlay must be scalable in order to support a large amount of receptors, and the associated overhead must be reasonable at theses large scales.
- **Self-organization:** The overlay must be built in a distributed manner and it also must be robust enough to support dynamic changes of the peers which take part in the overlay. Moreover, the overlay should continuously adapt to changes in the network, such as bandwidth and latency variances. The system should be self-improving, that is, the overlay should evolve towards a better structure as more information becomes available.
- **Bandwidth constraints**: The system depends on the bandwidth contribution of the present peers so, it is important to assure that the total contribution of the bandwidth of a user must not exceed its access bandwidth capacity.

- **Other system considerations**: Selection of a suitable transport protocol which allows to overcome connectivity restrictions, such as NAT and firewall traversal.

Other issues to consider in order to implement a real P2P live streaming are the connectivity between peers, and the transport protocols used for exchanging data. Some connectivity issues are commented in ANNEX B.

2.1.1 Types of overlay

2.1.1.1 Tree-based Overlay

In tree-based systems [17] the overlay is hierarchically organized. Data is sent from the source node to the rest of nodes of the tree. This approach is known as source-driven. The technique used to send data all along the tree is push-based, that is, when a node receives a data packet, it forwards the packet to each of its children.

There are several requirements for tree-based systems. It is desirable the system to build an efficient tree that matches the underlying network, and also it should provide a shallowed and optimal tree. That tree must be maintainable, that is, if a node leaves or crashes, all the branches of this node will stop receiving packets, in order to repair the tree. Thus, a shallow tree is beneficial.

Finally, cycles are not allowed in a tree-based overlay structure. Moreover, this approach has an important weak point: the failure of nodes, especially when the node is on the top of the tree. The reason is that this “upper-level” node failure may interrupt data delivery to an important part of the tree, momentarily dropping the overall performance. A shallow tree minimizes this problem, but also decreases the upload rate, since most of nodes are leafs and the upload bandwidth of leafs is not used.

2.1.1.2 Mesh-based Overlay

The construction of mesh-based overlays is an unstructured approach where it is not constructed or maintained any explicit structure data delivery. Normally, this mesh-based overlay uses a data-driven approach for exchanging data.

Data-driven approach is guided by data availability, which is used to route the data in the overlay. With pull-based techniques (such as CoolStreaming [18]), each node keeps a set of neighbour peers (partners) and periodically exchanges its data availability. Later, one node may retrieve unavailable data from one or more partners and, at the same time, will supply its available data to its partners. Another technique that can be used is the push-pull based approach (such as GridMedia [19]), where each node is autonomous and pushes data to its partners without the pull request. The

push is performed on time-based prediction, but a wrong prediction leads under flow or duplication issues.

This approach is robust to failures, because available data is redundant (kept in several partners), and the departure of a node simply implies that its partners will use other nodes to receive segments of data (local impact only). The potential bandwidth of partners can be totally used for exchanging data between partners. In mesh-based applications, the scheduling algorithm is a key component which must schedule the segments that are going to be downloaded from various partners to meet playback deadlines. It is the brain or core component of the system.

An interesting question then is: “Is there a future for mesh-based live video streaming?” The answer to this question is found in [20] where the authors conclude that mesh-based systems can achieve near-optimal rates in practice with negligible chunk misses (1%) at 90% maximum stream rate, and comparable or better rate than other tree-based systems (AQCS [21], GridMedia [19]).

Also, mesh-based overlay is still an attractive choice thanks to high rates, follows a simple unstructured overlay (no tree-like structure difficult to maintain), is scalable (delays grows slowly with overlay size) and has chunk-tolerance. But the main disadvantage in comparison with tree-based is that the delay remains higher than some tree-based systems (GridMedia [19]).

The P2P streaming platform used in this master thesis, named CoolRuc, constructs a mesh-based overlay inspired by CoolStreaming.

2.2 Video Coding Techniques

Peer-to-peer networks let us share information without any centralized component thanks to the cooperation of nodes (nodes are both servers and clients). These systems have well known advantages in terms of scalability, robustness, fault-tolerance. However, if all users receive and serve data, the probability that one stream breaks is higher because of the replication rate of the video streams. Furthermore, the connectivity to the network and the different paths used is strongly variable. Some robust and scalable video coding techniques such as Multiple Description Coding (MDC) or Scalable Video Coding (SVC) can be applied in media streaming. These techniques are well suited for situations where the quality and availability of connections vary over time. Using MDC or SVC in a P2P streaming scenario, the demanding host can choose the best hosts/servers candidate to make the transfer, and ask for different descriptors or layers in each case. As all information is travelling by

using different routes, if one of the descriptions or layers suffers packet loss or delay, the receiver is still able to decode the video. As it will be seen, in our scenario (live streaming), the selected technique was MDC because it allows real-time software coding of the media that will be spread all along the P2P network. Currently, real-time coding using SVC technique is still a challenge due to the high computational requirements (to date, there is no SVC real-time software encoder, just there are few real-time hardware encoders below HD resolution) which supposes an important limitation at implementation stage.

Next, MDC and SVC are explained in more detail. ANNEX C presents a comparison between both approaches based on some studies. Additionally, it is recommended to have an overview to [22], an interesting Internet Draft which explains MDC and LC and also proposes how to use them in order to provide robust and scalable transmissions over RTP.

2.2.1 Multiple Description Coding (MDC)

MDC [27][28][29] is a source coding technique which encodes a signal (this case a media resource: audio/video) into a number of N different sub-bitstreams (where N≥2). Each bitstream is called “descriptor” or “description”. The descriptors, which are all independently decodable, are meant to be sent through different network paths in order to reach a destination. The receiver can make a reproduction of the media when any of the descriptors is received. The quality of the reproduced media is proportional to the number of descriptors received, that is; the more descriptors are received, the better the reconstruction quality is. The idea of MDC is to provide error resilience to media streams. Since an arbitrary subset of descriptors can be used to decode the original stream, network congestion or packet loss, which are common in best-effort networks such as the Internet, will not interrupt the reproduction of the stream (continuity) but will only cause a (temporary) loss of quality. The quality of a stream can be expected to be proportional to data rate sustained by the receiver or depending on the terminal capabilities (adaptation process).

An example of MDC System can be seen in ANNEX D.

2.3.1.1 Approaches to MDC scenarios

Possible cases to implement MDC are shown in the next sub-sections:

a) MDC based on pixels

One option is the proposed by Andrea Vitali in [30]. This MDC method is essentially based on inter-pixel correlation. Since, in raw image data, the value of any given pixel can be reasonably predicted by the value of its neighbours, there is a strong correlation of inter-pixel information. By exploiting this, it is possible to create a multiple description algorithm where the source is split into N descriptions by a poly-phase downsampler (PD) along rows and columns. Because of this inter-pixel correlation each generated description maintains the main features of the original image. Figure 2.1 shows, for a better understanding of the PD multiple description (PDMD) procedure, the general scheme for N=4. If the transmission is error-free, the receiver over-samples the descriptions, combining them to restore the original source. In the worst case, when only one description is received, the receiver can exploit the available information in order to obtain a good low-resolution image. The number of decoders needed at the receiver equals the number of descriptions.

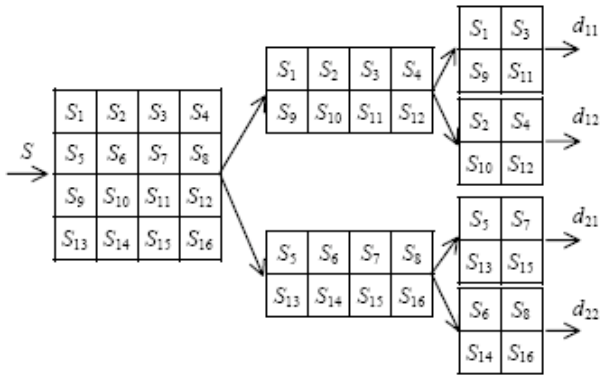

Figure 2.1 Example of poly-phase down-sampling system [30]

In Figure 2.1, the original image dimension is 4×4 pixels and the number N of multiple descriptions is 4. Each poly-phase stage (description) is composed of 4 pixels that can be organized according to their spatial location.

b) Hybrid MDC

Another option is the idea of Lu Meng-ting in his paper [31]. It talks about the Multiple Description Coding with spatial-temporal hybrid interpolation (MDC-STHI) for video streaming in peer-to-peer networks, which is based on the concept of multiple state video coding.

MDC-STHI consists of four bitstreams as shown in Figure 2.2. Stream Ef and Of represent the even and odd streams encoded from the original video. Eq and Oq are similar streams encoded at a low bit rate from the down-sampled version of the original

sequence. In this case it is used a 2:1 down sampling in each dimension, resulting in one quarter of the original resolution.

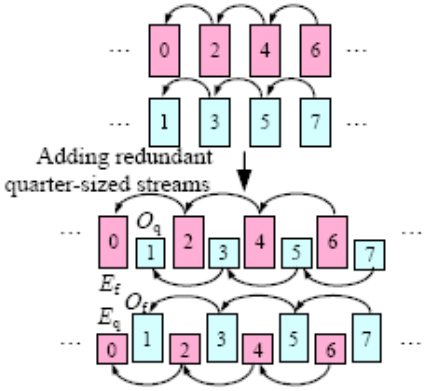

Figure 2.2 MDC-STHI [32]

Compared to other MDC designs, MDC-STHI offers superior flexibility for P2P video streaming as the four streams are independently encoded and a node may adaptively deliver suitable combinations according to the capabilities of a peer.

The combination of Ef+Oq or Of+Eq poses an interesting challenge as it contains two bitstreams of different resolutions. During transmission, the bits from Oq may piggyback in the same IP packet with Ef of the previous frame, and vice versa for Oq. This saves the overhead and ensures same time arrival of adjacent frames.

c) MDC + SVC based on FGS (Fine Grain Scalability)

Other option is the idea in the paper of Zhao Anbang [33]. Scalable coding could be combined with MDC to offer both rate scalability and error resilience. This can be achieved either by creating multiple description coders in which each description is scalable or by designing layered coders in which each layer is formed by multiple descriptions with a different amount of redundancy. This paper discusses an efficient, multiple description scalable video coding scheme based on FGS, in which each description is fine-grained scalable.

The MDC framework is shown in Figure 2.3. In the pre-processing stage, the original video sequence is shifted spatially to generate a new sequence. Then, two independent video encoders, e.g., H.264/AVC (or MPEG4 FGS or H.264 SVC), are used to encode the sequences to generate two independent coding streams. At the decoder side, after decoding the two channel streams, the shifted reconstructed video is shifted back with a weighted average used to reconstruct video in the post-processing stage. Kawada et al [34] showed that averaging always lead to better

quality (interpolation). The key question is how to optimally combine the reconstructed videos when the qualities of the two channels differ.

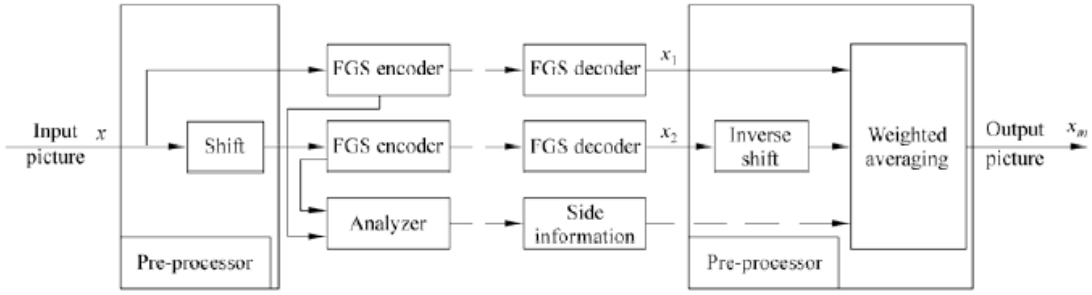

Figure 2.3 MDC framework [33]

d) MDC based on wavelet

There is also a video coding scheme which combines MDC with Redundant Discrete Wavelet Transform (RDWT) (Figure 2.4). It can improve robustness of transmission, and is benefit from RDWT to realize scalable coding to adapt to bandwidth variations.

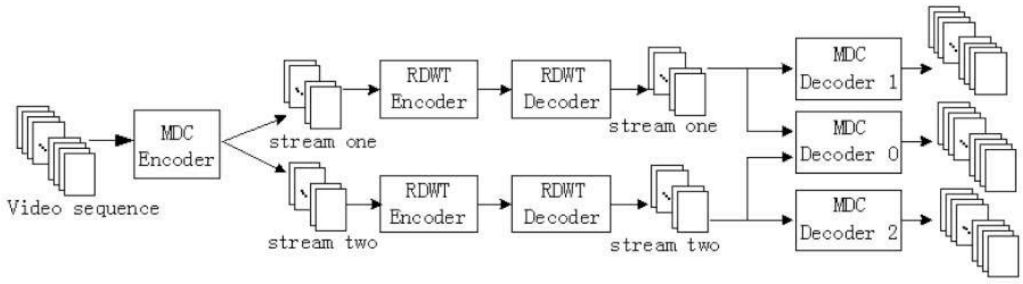

Figure 2.4 Multiple Description Coding based on RDWT

It is considered the case of dual multiple descriptions, and splitting the video into even and odd frames. Each description is coded by RDWT-based video coding scheme separately.

Each description has separate prediction loop, the odd/even frames are predicted from the frontal reconstructed odd/even frames. For conventional single-description coding (SDC) method, if decoding of P-frame is not correct, then the system may freeze its P-1-frame until new I-frame's emergence. This system can display the even frames or odd frames behind the P-frame as usual, and use temporal interpolation to recover the lost frames.

The Discrete Wavelet Transform (DWT) is shift-variant due to downsampling (Figure 2.5). With wavelet transform coding, a video frame is not divided into blocks as with the DCT-based MPEG coding. Instead, the entire frame is coded into several subbands using the wavelet transform.

The most commonly used set of discrete wavelet transforms is based on the use of recurrence relations to generate progressively finer discrete samplings of an implicit mother wavelet function; each resolution is twice that of the previous scale.

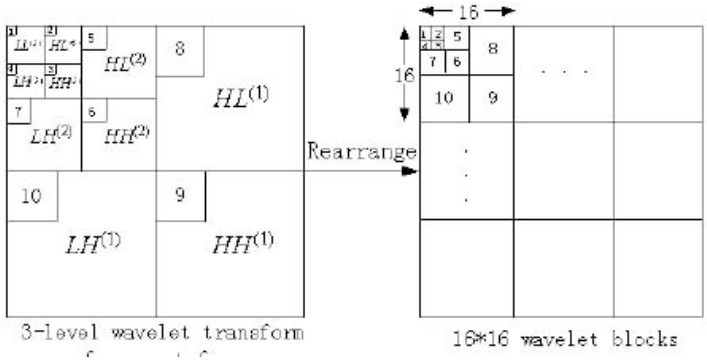

Figure 2.5 Wavelet decomposition

Each video frame undergoes a four-stage spatial decomposition which is recursively performed on the low frequency subband. The first stage of a filter bank structure used for the spatial decomposition is illustrated in Figure 2.6.

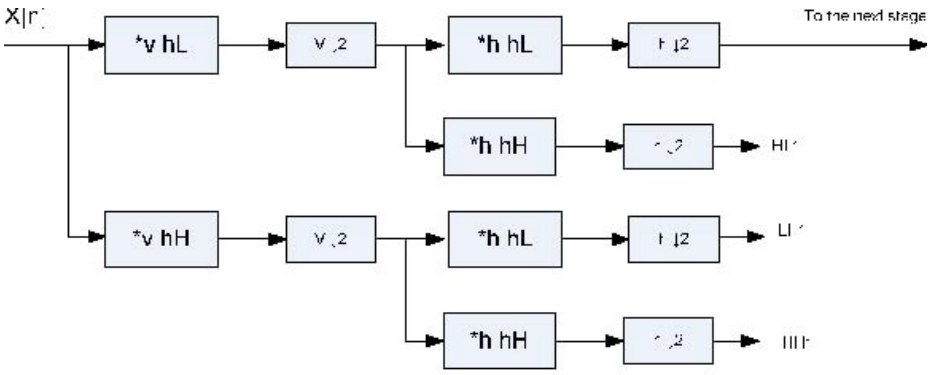

Figure 2.6 Wavelet filter bank structure

Here, x(n) is the input image. **v* and **h* represent the convolution in the vertical direction and convolution in the horizontal direction. The impulse response of the low pass filter and high pass filter are represented by hL and hH blocks. HL1 and LH1 and HH1 represent the outputs of the filters of the first decomposition stage. Each stage creates three subbands, while the fourth (lowest frequency band in both horizontal and vertical dimensions) is fed into the next stage of the spatial decomposition. The HL1, LH1 and HH1 coefficients are to be coded in individual substreams. These resulting substreams will correspond each one to a specific descriptor.

The discrete wavelet transform has a huge number of applications in science, engineering, mathematics and computer science. Most notably, it is used for signal coding, to represent a discrete signal in a more redundant form, often as a preconditioning for data compression.

2.3.1.2 MDC coding methods evaluation for high quality schema

MDC can be seen as another way for enhancing error resilience without using complex channel coding schemes. The goal of MDC is to create several independent descriptions which can have the same importance (balanced MDC) or they can have different importance (unbalanced MDC). The robustness comes from the fact that it is unlikely that the same portion of the same picture is corrupted in all descriptions. The coding efficiency is reduced depending on the amount of redundancy left among descriptions. The cost of this operation is to insert a certain amount of redundancy among descriptions, which are then compressed into bit-streams. The literature shows how some MDC approaches are more flexible than others in redundancy insertion. Varying the amount of redundancy in accordance with channel performance is important for the final reconstruction quality of the original resource: it is needed less redundancy insertion for error-free transmissions in comparison with unreliable packet transfers (such as Best Effort Internet traffic). Many approaches have been proposed to realize MDC coding: scalar quantizer [35], pair-wise transform coding (PTC) [36], spatial and temporal down-sampling [37][32][38], correlating filter-bank [39], frame expansion [40], matching pursuits algorithms [41]. Such approaches differ in terms of redundancy management and complexity. Some algorithms are designed for a general source, other more specifically for a type of signal such as: speech, image or video.

There are many techniques to create multiple descriptions: MDC quantization, correlating transforms and filters, quantized frames or redundant bases, FEC combined with layered coding, spatial or temporal polyphase downsampling. Many of these schemes can be adapted to existing video codecs, which are based on prediction, transform, quantization and entropy coding: it is possible to create descriptions in the pixel domain, in the error-prediction domain or in the transform domain.

Working in the error-prediction domain or in the transform domain produce very efficient but complex schemes. If not all descriptions are received correctly, the prediction fails because the frame memory in the decoder will not be the same as the one used in the encoder. To solve this problem, prediction may be removed, but this greatly reduces the video compression capability of the codec. A solution is to send a drift compensation term together with each description, but if there are more than two

descriptions, the number of drift compensation terms increases dramatically, again reducing efficiency. Taking into account this issue, a first appropriated approach could be to work in the pixel domain.

Working in the pixel domain has the advantage that MDC can be completely decoupled from the underlying video codec: descriptions can be created in a pre-processing stage before compression. Then descriptions successfully received can be merged in a post-processing stage after decompression. Spatial and temporal descriptions can be created by using Polyphase Downsampling, programmable lowpass filters controls redundancy, SNR descriptions can be created by means of MDC quantizers (either scalar or vector) and the structure of quantizers controls redundancy.

Finally, MDC is preferable when packet loss rate is relatively high or there is no time for retransmission.

2.2.2 Scalable Video Coding (SVC)

SVC, also called Layered Coding (LC) [38], adapts the video information to the network constrains splitting the images into different layers (similar to MDC). These layers represent the quality of the image, so, from the base layer, each successive layer improves the image quality, getting the full picture quality with the total amount of layers used. Concretely, SVC is the name given to an extension of the H.264/MPEG-4 AVC video compression standard. H.264/MPEG-4 AVC was developed jointly by ITU-T and ISO/IEC JTC 1. These two groups created the Joint Video Team (JVT) to develop the H.264/MPEG-4 AVC standard (ANNEX E). This system is compatible with MPEG-4 in its base layer. It must be noticed that the main difference between MDC and SVC is that MDC creates independent descriptors (can be balanced or unbalanced) while SVC creates dependent descriptors (unbalanced).

According to SVC, we can apply different techniques to make the video data scalable, in the same way as MDC.

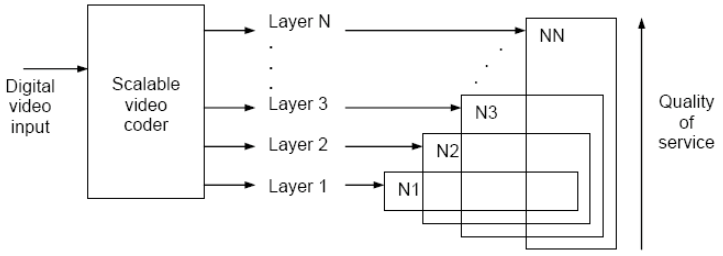

Figure 2.7 Scalable Video Coding (SVC) general process

a) Spatial scalability

It is based on the Laplacian pyramid to differentiate the layers of spatial resolution. With the total number of samples of an image one can divide it until getting the base layer. To get the next layer from the first one, it is necessary to reconstruct the image with twice number of samples, and only send the difference between the image and the real one with the same number of samples.

b) Temporal scalability

The number of frames per second or frame-rate can be modified with the elimination of some frames to be implemented by the receiver with prediction techniques, or using Motion-Compensated Temporal Filtering (MCTF) in order to assure reconstruction. This last option has been recently deprecated due to the high complexity in the decoder, it was not compensated in an effective coding improvement.

c) Quality / Fidelity / SNR scalability

The quality or fidelity scalability, also known as SNR quality, can be considered as a spatial scalability special case, where the images size of the base layer and refinement layers are identical. The main difference between layers is the quantization step size. This yields to different qualities while maintaining the spatial resolution. The higher the refinement layer is, the smaller quantization step size is used.

d) Combined scalability

In spatial/quality modalities, the data and decoded samples of lower resolutions/qualities can be used to predict data or samples of higher resolutions/qualities in order to reduce the bit rate to code the higher resolutions/qualities.

Three quality scalability coding types can be distinguished:

- **Coarse-grain scalability (CGS):** it works like the spatial scalability without up-sampling and provide discrete quality refinement.
- **Medium-grain quality scalability (MGS):** it is a medium solution amongst FGS and CGS. It embeds a FGS refinement into a CGS refinement.
- **Fine-grain quality scalability (FGS):** it plays with the quality of the data to create the layers. We can decide the quality of the quantification process. So, to obtain the base layer, a coarse quantifier will be used. After this, the difference with the original image will be sent in upper layers using the same process.

2.3.2.1 Transport methods

The H.264 video codec covers all forms of digital compressed video from low bit-rate Internet streaming applications (baseline profile) to HDTV broadcast (main profile) and Digital Cinema applications with nearly lossless coding (extended profile). Compared to the current state of technology, the overall performance of H.264 is such that bit rate savings of 50% or more are reported. Digital Satellite TV quality, for example, was reported to be achievable at 1.5 Mbit/s, compared to the current operation point of MPEG 2 video at around 3.5 Mbit/s.

H.264 codec specification [42] distinguishes conceptually between a video coding layer (VCL) and a network abstraction layer (NAL).

- **Video Coding Layer (VCL):** It mainly contains the signal processing functionality of the codec, such as: mechanisms such as transform, quantization, and motion compensated prediction. The VCL outputs slices.
- **Network Abstraction Layer (NAL):** It encapsulates the slice output of the VCL encoder into Network Abstraction Layer Units (NAL units), which are suitable for transmission over packet networks or use in packet oriented multiplex environments.

A NAL unit of H264/AVC consists of a one byte header and the payload byte string. The header indicates the type of the NAL unit (shown in ANNEX F), the (potential) presence of bit errors or syntax violations in the NAL unit payload, and information regarding the relative importance of the NAL unit for the decoding process.

These NAL units can be encapsulated using several transport methods, concretely in TRILOGY project we used RTP (see ANNEX G) as it is a requirement in CoolRuc application.

CHAPTER 3. BACKGROUND

Over the last years, there have been significant research studies about a variety of issues related to P2P media streaming [15][43] However, there are some popular media streaming applications whose internal operations are barely known. Therefore, it is difficult to analyse the algorithms and mechanisms used. On the other hand, there are academic developments associated to universities or developer communities that have been well specified and published, so it is possible to know their functionalities and determine the main advantages and the possible limitations of them. Currently, there is a long list of P2P media streaming platforms such as PPLive, CoolStreaming, TVAnts, StarCast, Anysee, among others. Each of them uses different ALM (Application Layer Multicast) mechanisms and video coding schemes. Nevertheless, among so many systems, only a small number of them apply media coding techniques such as MDC or SVC. CoolRuc platform is based on an unstructured P2P live streaming that implements a mesh-based overlay inspired by CoolStreaming [18] and Pulse [44] (buffer operation). For instance, CoolStreaming is based on DONet (Data-driven Overlay Network), which is a P2P technique that does not need any kind of complex tree structure for data transmission. It includes a gossip-based partnership management algorithm and an intelligent scheduling algorithm in order to provide a continuous distribution of streaming contents. This system is one of the most widespread but does not use MDC or SVC coding. CoolRuc is a P2P media streaming platform developed within the TRILOGY project which also uses DONet concepts and takes advantage from services and protocols provided by the Java-based JXTA Technology [24]. This platform incorporates new video coding techniques: MDC researched by MediaEntel UPC group and SVC [45] researched by TAM-URL group. Also includes new facilities in the field of P2PSIP [46] researched by UPF-Nets Group, which allows to support innovative continuous media services and collaborative complementary services such as Instant Messaging (IM) or Voice over IP (VoIP).

3.1 CoolRuc Architecture

CoolRuc architecture is formed by several components (shown in Figure 3.1). These components are used for broadcasting a channel in live or pre-recorded modes. For each broadcast channel an own overlay exists. A channel can be mapped to a virtual overlay composed by a source and the nodes joining the broadcasting (audience). Note that signalling messages are represented by blue vertically stripped arrows, while data messages are represented by the brown horizontally stripped ones. These

components are explained more detailed in [47].

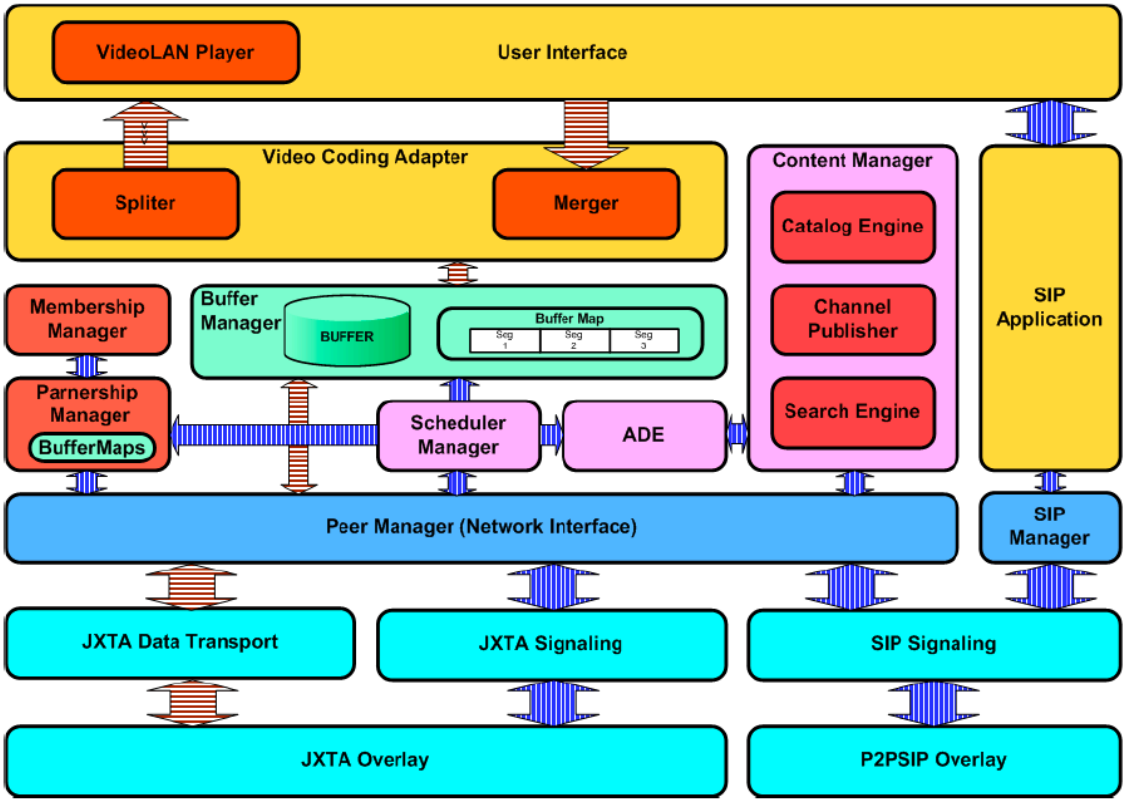

Figure 3.1 CoolRuc architecture of a node

It is important to understand how some components are related. Peer Manager is the abstract interface for a transport message between each peer. This component separates JXTA [24] interface and SIP interface in a CoolRuc node for possible future changes in the underlying transport and signalling mechanisms. Further, Peer Manager is a component which is able to start the different services required by a node according to its role (source or receiver). When the node acts as receiver peer: Membership Manager, Partnership Manager, Buffer Manager, Scheduler and Splitter; or when acting as source peer: Membership Manager, Partnership Manager, Buffer Manager and Merger.

JXTA is a platform that provides mechanisms for creating P2P overlays and communicates the peers. In CoolRuc, each channel is a JXTA logical group with an own Membership Manager. According to CoolRuc, JXTA Overlay only manages the members of a channel and each channel will be managed by the P2PSIP Overlay. For data transport, CoolRuc uses JXTA Sockets for establishing the communication among peers.

Video Coding Adapters are responsible for controlling the Merger and Splitter components and their interaction with the User Interface and Buffer Manager. The Merger component is used at each receiver peer. Its input is a set of coded subbitstreams distributed all along a channel. The resulting output is a single video stream with the maximum possible quality for the peer depending on the input of the Merger (received bitstreams) and the peer device features. The Splitter component provides the pre-processing stage necessary to distribute a video source according to the video coding technique being used (MDC/SVC). This stage is performed when considering the peer as a video source itself. The input for this component is a media flow and the output generated is the corresponding set of sub flows processed and encoded accordingly to the coding mechanism (MDC/SVC). This master thesis contributes to the design, implementation and validation of the Video Coding component using the MDC technique.

The Video Coding Adapter interacts with the Buffer in order to generate a Buffer Map for latter sharing in the P2P network the availability data of the info contained in each node.

Finally, the Content Manager component is in charge of publishing, discovering and maintaining the information (meta-data) which describes channels in order to notify their existence into the network and, consequently, allowing their consumption by the users.

CHAPTER 4. DESIGN AND IMPLEMENTATION

This chapter presents the main design issues and makes an overview over the implemented video coding system integrated in the high quality P2P live streaming application called CoolRuc. This solution proposes MDC as a proper video coding scheme suitable for these kinds of networks, mainly characterized by heterogeneity. As it has been said, MDC offers a robust solution against loss effect for video streaming over heterogeneous networks. In addition, this solution was implemented because it allows to separate the desired media into descriptors in real-time, then, it is appropriated for building a live streaming solution. It is important to mention that currently this is not possible to be done by using SVC, because SVC real-time software encoding is still a challenge due to high demanding computational cost. It is just possible to encode SVC streams in real time only if dedicated hardware is used. In addition, SVC real-time hardware encoding supports up to SD resolutions.

This chapter describes the design of the different MDC techniques that were considered (spatial, temporal and hybrid). It is expected that these techniques will offer considerable improvement in visual quality experience (continuity index) in heterogeneous networks despite of loss effect. Finally, it presents the design of the implemented solution (system components). All the implemented components are software. It must be mentioned that the implemented solution is based on a spatial balanced MDC technique.

4.1 MDC Techniques

Now, three different MDC techniques that were considered are described. All of them were implemented into Matlab in order to carry out some simulations (CHAPTER 5). The simulation process allowed to validate the suitable design of these techniques in order to guide further developments according to the specific characteristics of each one.

MDC techniques can be of two basic types: balanced or unbalanced. Balanced descriptors imply that each description has the same importance (weight) as the rest, while unbalanced descriptors imply that the different descriptors can have different weights or importance (this can be seen as a similar approach to SVC). The described techniques are considered to be balanced.

All the techniques are explained from the splitting process point of view as the merging process just consists on applying the inverse process (composition of the split

information). The splitting process performs a subsampling over the original resource while the merging process applies an upsampling process.

4.1.1 Spatial MDC

The spatial technique consists in splitting each individual frame into several descriptors, working in the pixel-domain. Resultant descriptors will have a portion of the original spatial information. Additionally, it must be noticed that they will have lower resolution than the original resource, but they maintain the same aspect ratio. As it is obvious, each description has a lower bitrate in comparison with the original video resource. This process can be seen as a spatial subsampling. Concretely, the spatial technique used allowed us to generate very similar descriptors (visually, they contain the same frames, nevertheless, they do not contain the same information, but very close). The spatial splitting process is depicted in Figure 4.1.

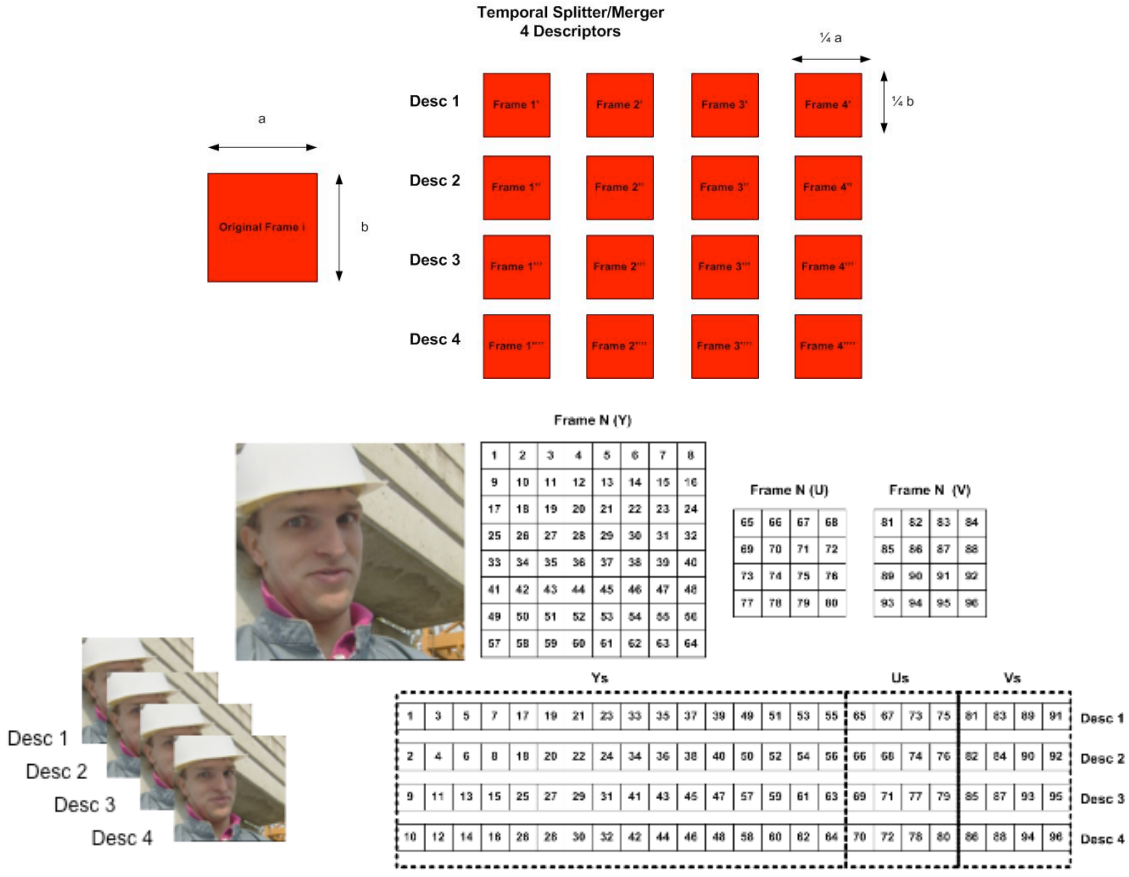

Figure 4.1 Spatial MDC

4.1.2 Temporal MDC

This technique consists in allocating each frame of a video sequence in a different descriptor. The easiest manner to implement this technique is using a round robin strategy, weighted or non-weighted depending on the needs. Resultant descriptors

have lower bitrate due to the reduction in frame rate in comparison with the original resource. This process can be seen as a time subsampling. The temporal splitting process is shown in Figure 4.2.

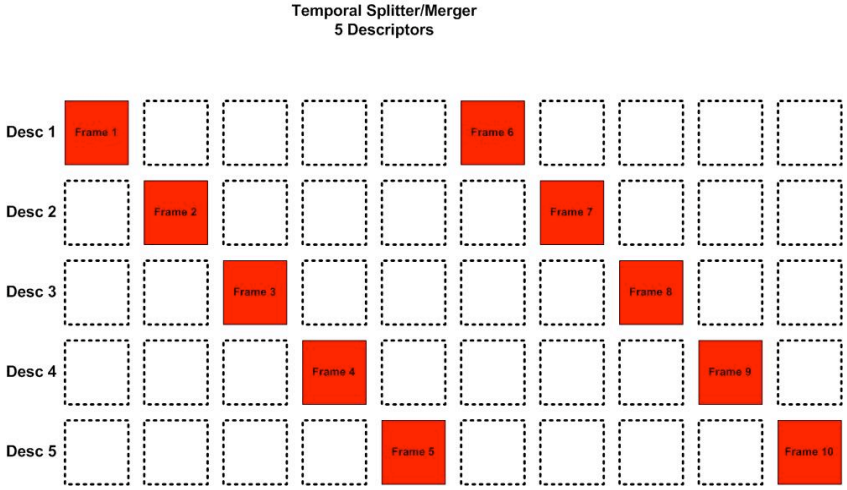

Figure 4.2 Temporal MDC

4.1.3 Hybrid MDC

Intuitively, this technique is based on a combination of spatial and temporal techniques. Concretely, the considered hybrid technique sends one full frame in each description while in the rest of the descriptors it is sent a part of the original frame which is different from the other descriptors.

This technique introduces more redundancy in data in comparison with the other two techniques. However, the resulting descriptors are generated with a lower bitrate in comparison with the original video resource. The process is presented in Figure 4.3.

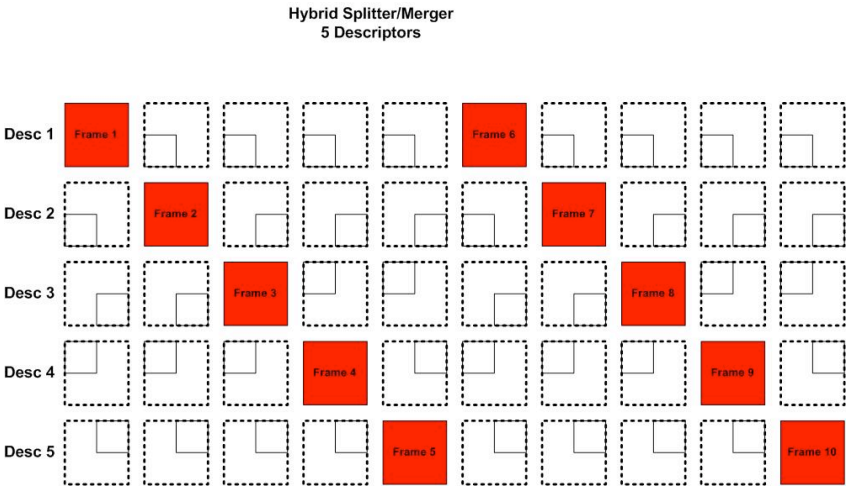

Figure 4.3 Hybrid MDC

4.2 MDC System Overview

The MDC system has two main parts: transmission system (shown in Figure 4.4) and reception system (depicted in Figure 4.5). The main components of the MDC system are the Splitter and the Merger which provide the specific processing of the video streams. The transmission system, which contains the splitter, is executed in the source peer only. Oppositely, the reception system, which contains the merger, is executed in each receiver peer. A coding stage (codec aware) is also present after.

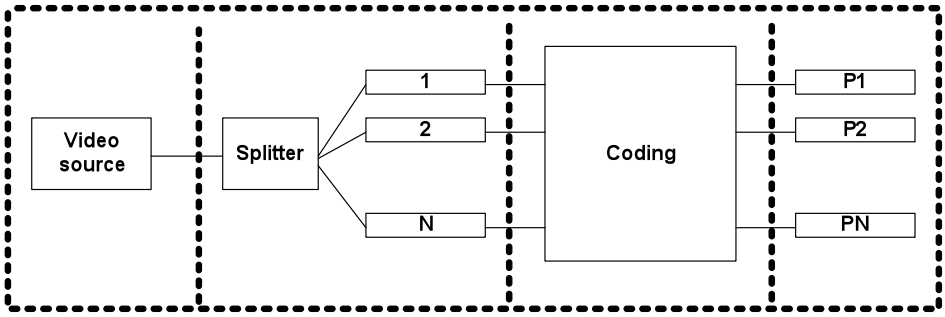

Figure 4.4 MDC transmission scheme

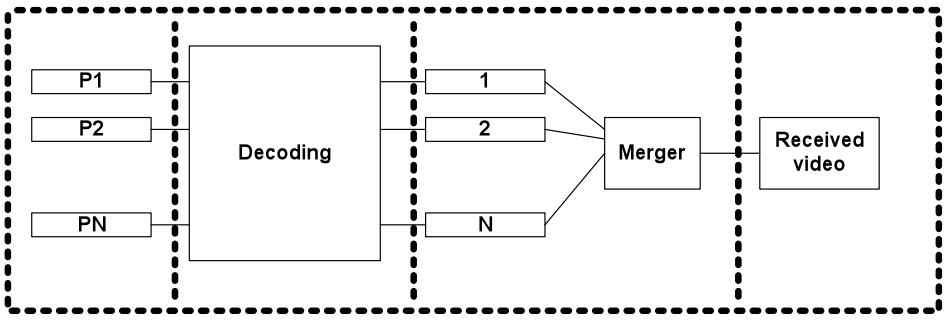

Figure 4.5 MDC reception scheme

The first step in media processing according to MDC is the pre-processing of the source video stream in order to obtain the different substreams that can be coded and displayed independently (Splitter). Each substream, as already mentioned, is coded (the pre-processing makes this scheme codec independent). Then, each coded substream is ready to be transmitted. In this case, it is sent to the CoolRuc (InputAdapter component). The fact of having different subflows that are related and that are to be sent separately has an impact on the transmission and packetization. Regardless of the architecture deployed for transmission and signalling it is clear that

the different subflows must have an identifier that relates them to the corresponding video source.

The pre-processing stage can be implemented with different pixel/frame-processing techniques considered for MDC. The number of descriptors depends on the technique used. The first approach is to work with a spatial downsampling selecting odd or even columns and rows.

The transmission of each subflow can be done independently and thus it does not restrict the transmission architecture. When a channel is created, the number of descriptors corresponding to a channel must be published along with the rest of information of a channel (it could be done in the DHT in a publish/subscribe approximation or by advertisement propagation and caching). The meta-data associated to a channel can be used both to provide user-friendly-and-readable information (description of the channel, duration, genre, quality, etc.) and application configuration parameters (inputs to configure the merger modules at the receivers: number of descriptors, rates, formats, etc.) (Figure 4.6).

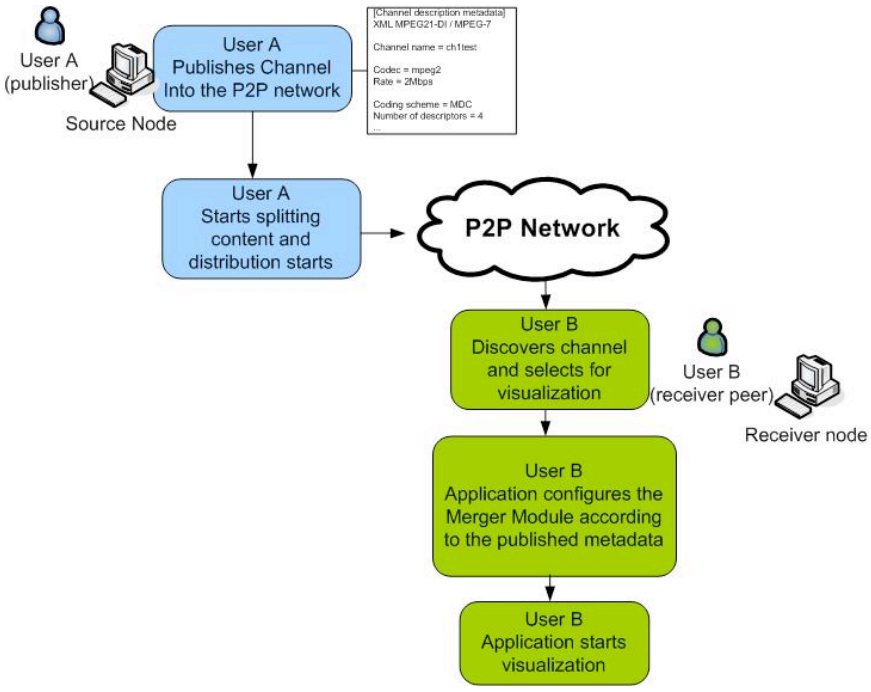

Figure 4.6 Meta-data publication

The reception system for the MDC coding solution is the following: the different subflows received are decoded and the Merger stage is applied to obtain the maximum quality by mixing the received descriptors. Depending on the number of available descriptions for each channel and the use of content adaptation (Adaptation Decision Engine component in CoolRuc architecture), a user can receive more or less

descriptions, that affect to the number of buffers needed in reception. The expected input for our decoding block is a set of subflows belonging to the same channel.

Next sections, describe the system components that were implemented in the final prototype in more detail.

4.3 Transmission Module Component: the Splitter

The Splitter module is the responsible for processing the media source and generating the corresponding descriptors that will be transmitted by the CoolRuc application. The integrated MDC Module makes spatial splitting. The spatial splitting process can be seen in the following representation (Figure 4.7). Concretely, it is represented the YUV splitting process and its decomposition into four descriptors.

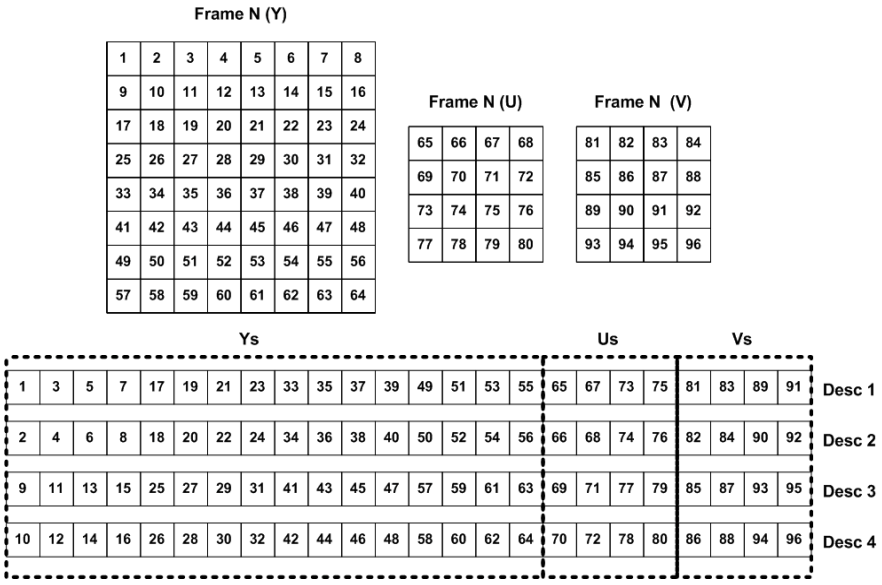

Figure 4.7 YUV example

The general approach for the Splitter module is the following: with a .YUV media resource as input the module is able to provide a set of sub streams with lower rate that will be sent locally to the CoolRuc application. The splitting module is developed in C++. The communication with the CoolRuc (Java based) is done using the JNI framework.

The input resource is in .YUV format. It can be pre-stored or live content (for instance, a DV stream from a DV camera). This flow is locally sent to the Splitter module which according to a set of input configuration parameters will generate the corresponding number of descriptors (in the final prototype implemented in CoolRuc, we set the number of descriptors to four by default for simplicity and for being a typical value in research and practice). These descriptors are also locally sent to the CoolRuc

application which is responsible of transmitting them through the P2P overlay, using chunks. We use local TCP connections in order to deal with the order of MDC Packets which are sent to CoolRuc.

Prior to send data to CoolRuc, the Splitter Module encodes the YUV 4:2:0 input into MPEG-2 streams. These streams are then transferred to CoolRuc using the TCP available connections for each descriptor. The complete operation of the MDC Splitter Module can be seen in the following block diagram (see Figure 4.8).

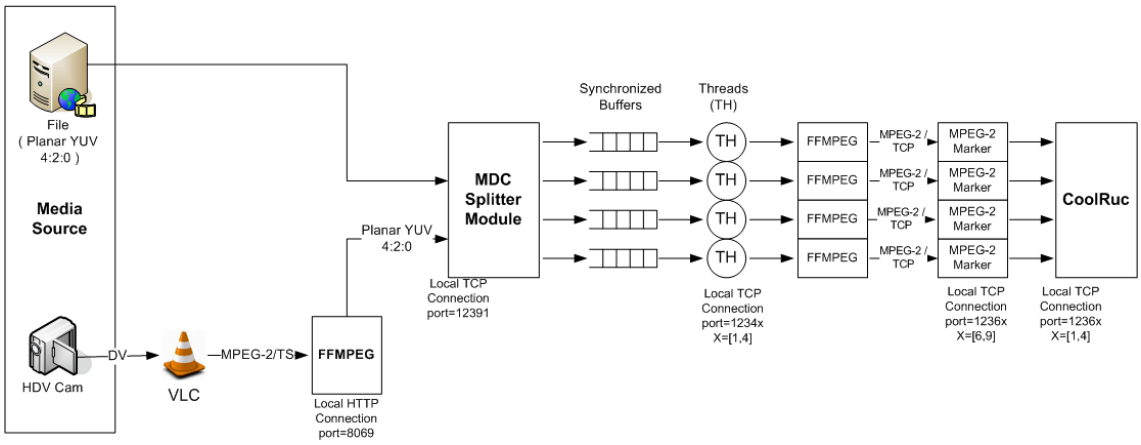

Figure 4.8 MDC Sender components (Splitter Module)

It must be clarified the operation of the following components:

- **Synchronized buffers**

  It has been implemented one synchronized buffer per descriptor after the splitting process carried out by the MDC Splitter Module to allocate the video stream corresponding to each descriptor. These buffers are synchronized because the access to them is shared by both the MDC Splitter Module (writes data into each buffer) and the THs (read data from the buffers), so, the buffer must be synchronized.

- **TH: Threads**

  It has been implemented four threads that read from the synchronized buffers and send the gathered YUV data to the FFMPEG instances. FFMPEG will encode the data for transmission.

In this integrated prototype of the MDC Module, it was fixed the coding GOP (Group Of Pictures) to 1 as an input of the FFMPEG process execution. Fixing the GOP to 1 allowed us to fix some problems generated by the FFMPEG decoder. FFMPEG does not implement a robust MPEG-2 decoder and constantly informs of decoding errors in presence of missing frames (I, B, P). Note that using GOP = 1 implies that we are

obtaining just I frames from the FFMPEG MPEG-2 coding stage, so we are providing a solution very close to a MJPEG codification. It is proposed to use GOP distinct from 1 for further releases and research. Afterwards, each coded frame is identified and it is added a signalling header in order to identify each frame which is sent to CoolRuc. The result consists in the creation of an MDC Packet (shown in Figure 4.9) that contains an I frame into its DATA field.

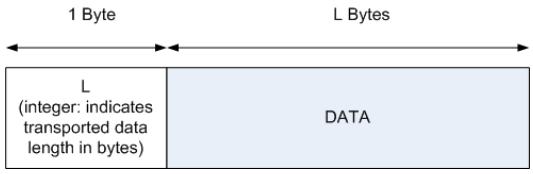

Figure 4.9 MDC Packet format

Header (L) is added to each frame in order to properly packetize them into chunks (data transmission units exchanged in CoolRuc). This action is carried out by the MPEG-2 Marker Components. We need to identify the coded frames in order to accommodate them into chunks. This is done this way in order to avoid future reception errors produced by FFMPEG as said before.

The detection of each frame is done by reading the MPEG bitstream. Concretely, the init sequence header 00 00 01 B3 (remember we are using a sequence of 1 frame, that is GOP=1) is recognized. Figure 4.10 shows the content of an MDC Packet.

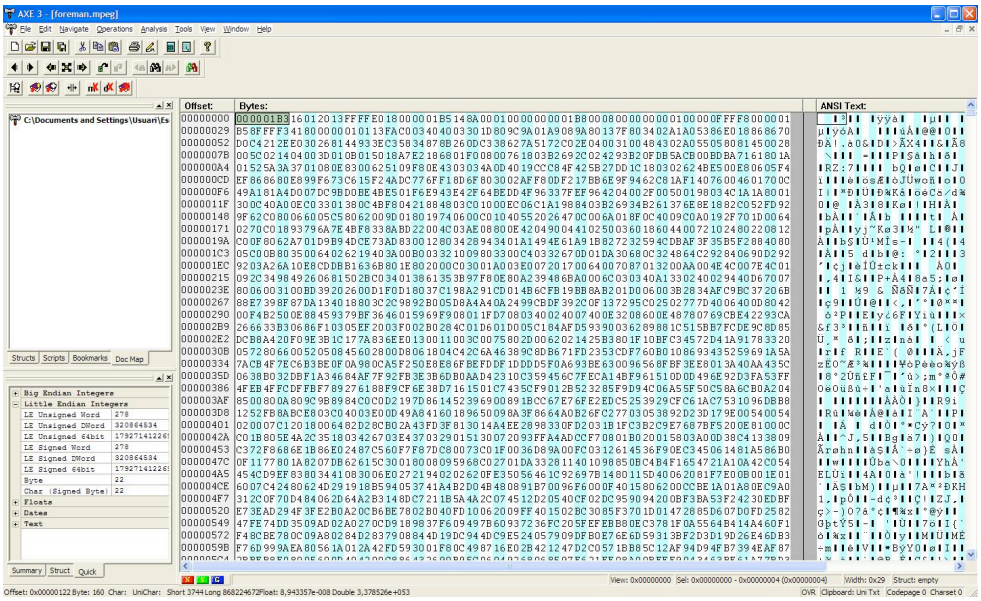

Figure 4.10 Hexadecimal representation of an I frame coded with MPEG-2

4.3.1 Splitter Module Configuration - Inputs

The Splitter module requires a set of parameters in order to configure the splitting mechanism. This configuration depends on the source characteristics as well as on the type of splitting processing. Next, Table 4.1 shows the corresponding parameters.

Table 4.1 Splitter input parameters

Media source inputs
Resolution: spatial resolution of the original resource
Source path: source path of the original resource (only if input type equals to 'file')
Input type: specifies the media input. It can take to values 'file' or 'camera'
Ports: ports reserved by the module

These parameters must be provided by the Java component that manages the Splitter module in order to configure it.

4.4 Reception Module Component: the Merger

The Merger module is the responsible for processing the received descriptors from the CoolRuc application and generating a displayable content (merges the descriptors). A general scheme is depicted in Figure 4.11.

The general approach for the merger module is the following: the CoolRuc application provides a set of ordered segments for each descriptor that are sent locally through a TCP connection to the Merger module. The Merger module generates a displayable content using these descriptors. This module is developed in C++.

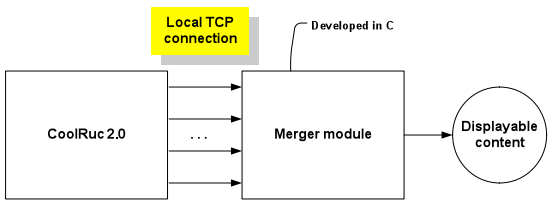

Figure 4.11 Merger general scheme

The final implemented MDC Merger Module components can be seen in Figure 4.12.

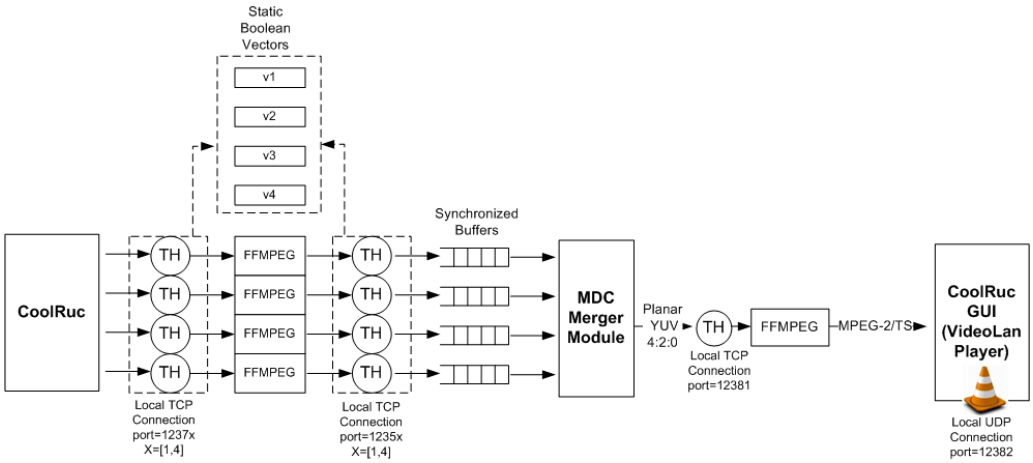

Figure 4.12 Receiver components (Merger Module)

The threads (TH) placed before and after the first FFMPEGs are the modules that allow to adapt the data received (input format) from CoolRuc in order to fit the desired format at the input of the MDC Merger Module. Remember that we are receiving MDC Packets that contain MPEG-2 frames from the CoolRuc and it is required to have a YUV input at the MDC Merger Module. So, these TH first receive the segments received from CoolRuc. The corresponding segments provided by the CoolRuc application follow the format shown in Figure 4.13.

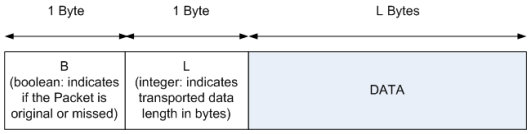

Figure 4.13 Segment Format

The B field is a 1-Byte Boolean that indicates if the DATA was received by CoolRuc well or not. This field is used as signalling in order to notify the received data. B can take the following values:

- B=1: indicates that the MDC Packet was well received by CoolRuc
- B=0: indicates that the MDC Packet was not received by CoolRuc (missed packet or frame)

The L field indicates the length of the carried DATA if is present.

- L!=0: indicates the DATA length
- L=0: indicates that there is no DATA (because it was not obtained from the P2P Network)

The first TH read these packets and writes the value of the B field into four static Boolean Vectors. These vectors contain then the “signalling” of the original and missed

frames. The data, which contains an I frame, is sent to FFMPEG. The TH placed after FFMPEG's first read the value of B in order to know when they have to read a received frame. This later TH just places the DATA into four synchronized buffers in a proper way.

Note that, if it was read one frame as B=0 (missing frame), the data placed in the synchronized buffer is a padding of zeros. This will allow the merger to interpolate the missing positions. Currently, in the final release of the module, it is implemented an interpolation mechanism based on the copy of the previous well-received pixel which is closest to the lost one. Figure 4.14 shows how the interpolation is done within a single frame.

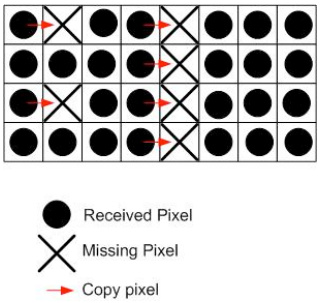

Figure 4.14 Interpolation by previous and closest pixel copy in a frame

Thanks to the interpolation of missing pixels, we improved the average PSNR in 2.5 dB approximately. This supposes better objective quality and also better perceived quality by the user.

Finally, the YUV output of the Merger must be adapted into a supported format for the CoolRuc Media Player, which is based in VideoLan. This adaptation is done thanks to the last FFMPEG instance. In this last coding stage, it is coded the resulting merged YUV video into MPEG-2 format. This is done because, at the present time, VideoLan cannot play YUV format coming from network (streaming mode).

Additionally, a last transcoding stage is used in order to stream video to a mobile device associated to a CoolRuc node. That is, CoolRuc application allows a user to send the received video stream to its mobile device in order to continue the visualization of a video resource in its portable device (Mobile phone, Laptop, etc.). Figure 4.15 shows the result of moving the visualized media from a desktop to a mobile.

Figure 4.15 Stream to mobile phone (Merger Module Output)

4.4.1 Merger Module Configuration - Inputs

The configuration of the merger module is to be done by providing a set of parameters by the CoolRuc application. The required parameters are listed in Table 4.

Table 3.2 Merger input arguments

Media source inputs
Resolution: spatial resolution of the original resource
Output paths: save stream to file or streaming mode
Ports: ports reserved by the module

4.5 Software Specification of the Video Coding Component

This section provides a more detailed specification of the operation of Splitter and Merger software modules. The corresponding state charts are provided for each one and allows the reader to know better their main operations.

4.5.1 Splitter Module

The main states of the Splitter module are detailed in this section. The first step is the loading of the required media source parameters and the second step involves the MDC configuration parameters. This allows to configure the splitter processor. When the splitter processor is configured the number of descriptors is established and thus the required buffers can be created. Also a number of threads equal to the number of descriptors is needed to manage the buffering and transmission of the sub streams. Therefore, the next step is the creation of the threads. Each thread is provided with the appropriate buffer identifier (Figure 4.17).

The threads created to manage the coding and transmission of each descriptor have the following scheme: read data while available from the buffer with the provided ID, encode this data and send it to the CoolRuc application (Figure 4.16).

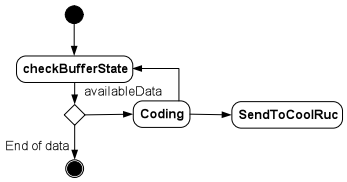

Figure 4.16 Splitter thread state chart

The main execution thread starts to receive data from the media source and initiates the processing stage (splitting). This main thread has to control the availability of source data. The result of the splitter processor is stored in the corresponding buffers.

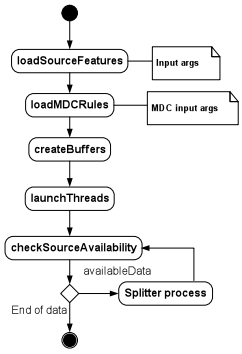

Figure 4.17 Splitter main thread state chart

4.5.2 Merger Module

The main states of the Merger module are detailed in this section. The first step consists on loading the MDC features obtained from the Cataloguing module (JXTA channel advertisement) in CoolRuc in order to configure the Merger. Then, according to the number of descriptors the required buffers will be created. The third step consists on creating the threads responsible for receiving the CoolRuc data, decoding it and passing it to the Merger processor. Then, this main thread starts retrieving data from the buffers and performing the merging process. See Figure 4.19 to see the process.

The threads responsible for each descriptor retrieve the data provided by the CoolRuc application (through a local TCP connection) and decode it. Then the decoded segments are stored in the Merger buffers (Figure 4.18).

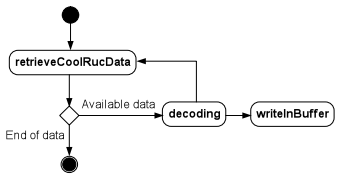

Figure 4.18 Merger thread state chart

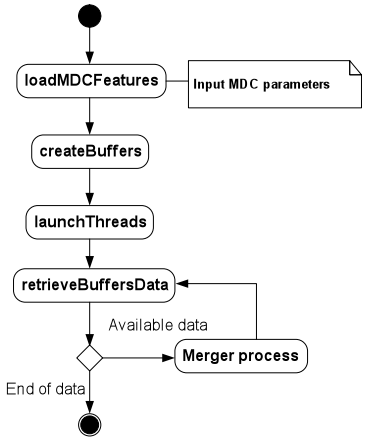

Figure 4.19 Merger main thread state chart

4.5.3 Software Components Architecture

This section makes an overview of the main MDC software components which are a part of the Media Coding module (Figure 4.18). Also, it details their interaction with the CoolRuc component responsible of managing these C-based modules.

As it can be seen, the MediaCodingModule component is composed by a set of components that provide processing, coding and buffer management functionalities.

The SplitterManager and MergerManager are the components of the MediaCodingModule that perform the communication with the Java application. The Java JSplitterManager and JMergerManager are the Java-side components responsible for this communication and provide the expected input parameters required for the MediaCodingModule components configuration.

- **SplitterProcessor**: is the responsible of performing the media source processing in order to generate the expected sub-streams. It communicates with the BufferManager to write the generated sub-streams in the output buffers. It also communicates with the SourceReader to retrieve data from the media source.
- **MergerProcessor**: this component is the responsible for the merging stage with the received data from each descriptor. It communicates with the BufferManager to read the data from the reception buffers. It also communicates with the NetworkManager in order to send the merged stream to the module of the application that displays it.
- **DescriptorManager**: the handling of each thread (related to a specific descriptor) is performed by this component. It interacts with the SourceReader

when considering receiver side to obtain the data from the CoolRuc. In the splitter side it interacts with the BufferManager to retrieve data from the buffers. It also uses the Encoder and Decoder components (in this case the ffmpeg application).

- **BufferManager:** the buffer writing and reading is controlled by this component that interacts with the BufferReader and BufferWriter.
- **SourceReader**: when retrieving data from a local TCP connection, this component interacts with the NetworkManager to get the corresponding media data.

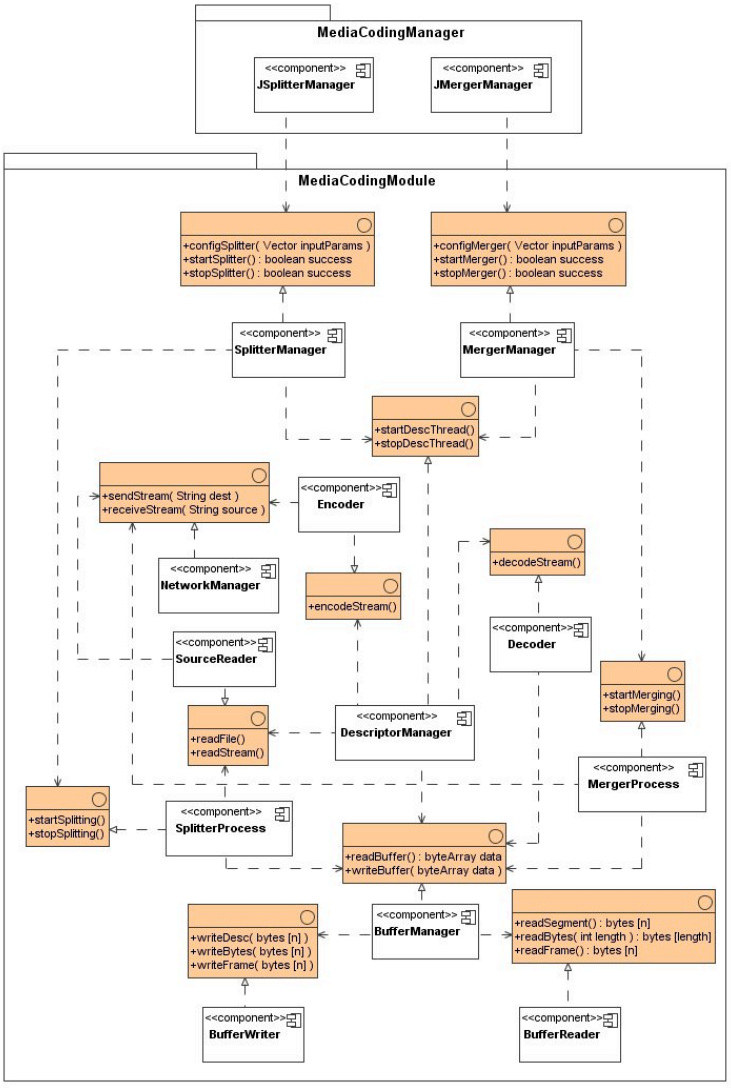

Figure 4.20 MDC Video Coding Components, software block diagram

CHAPTER 5. METHODOLOGY AND EXPERIMENTAL RESULTS

Simulation was an essential part of this master thesis. It allowed to validate the correctness of the proposed solution before implementing a real prototype. In the simulation process, the three specified MDC techniques (spatial, temporal and hybrid) were evaluated by means of the calculation of the objective quality measured in the overall system. The objective image quality metric used has been Peak-Signal-to-Noise-Ratio (PSNR). In these tests, there were used different well-known videos:

- Foreman: high motion video sequence
- Akiyo: low motion video sequence
- Carphone: medium motion video sequence

The next step was to implement a real point-to-point prototype according to simulation results. This resulting prototype was evaluated too. However, in addition to objective quality measurement, some performance metrics were quantified.

Finally, the implemented prototype was integrated into the whole platform, concretely in this case the CoolRuc P2P live streaming platform. The final evaluation of the system was done by means of monitoring the continuity index of the nodes visualizing the same distributed video content.

First, this chapter describes the simulation environment and the obtained results. Then, the evaluation of the first point-to-point MDC prototype is shown. Finally, we show preliminary results obtained from the integrated P2P scenario.

5.1 MDC Simulation Results

We have carried out two kinds of simulations:

a) Applying MPEG-4 video codec stage to each generated description

b) Without video codec stage, that is, transmitting each generated description without encoding stage

The propose to make simulations evaluating the system with and without codec was done in order to verify that MDC allowed to add a coding stage for providing higher performance at transmission time. Remember that MDC splitting is a preprocessing operation which is codec-independent. Each descriptor can be sent in “raw” format or, on the contrary, encoded with a codec. Coding allowed to significantly reduce the bandwidth required to transmit each description, while maintaining very good quality at

the receiver. The distortion introduced by lossy codecs, such as MPEG-4 and MPEG-2 was more than acceptable.

5.1.1 MDC simulation with MPEG-4 codec

This section explains the MDC simulation using a MPEG-4 coding stage after the splitting process is executed. The scenario was the following one (Figure 5.1).

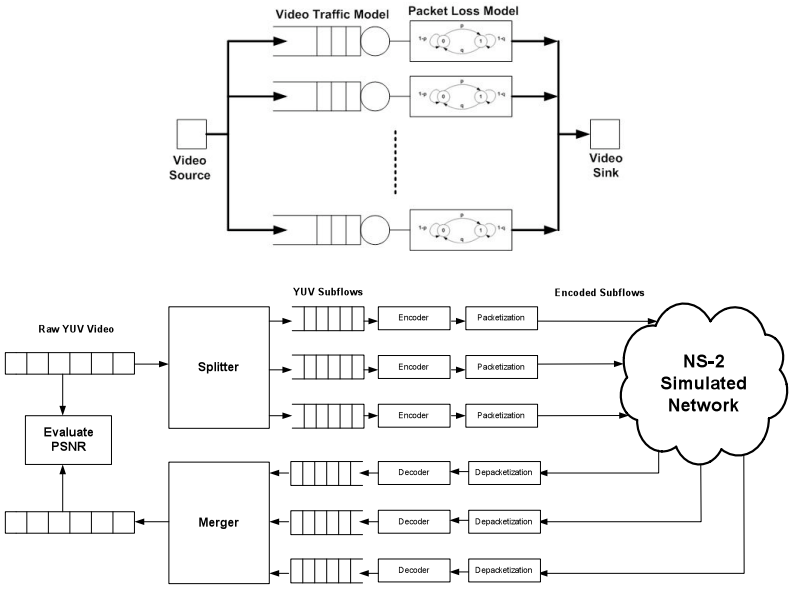

Figure 5.1 Simulation scenario components

There are defined parallel links where each descriptor follows a different path that is, simulating a multipath communication. In this simulation it is used 2, 4, 8, 12 and 16 descriptors, so there are defined as many links as number of descriptors.

Also, a packet loss model is defined in each path. Specifically, in this thesis we used a Bernoulli model (random) for modelling the channel. The packet loss percentages used in this simulation were 1%, 5% and 10%.

For MDC simulations with codec, temporal and spatial MDC techniques were used. The reference videos used were the following ones:

- Foreman QCIF (176x144), 288 frames
- Akiyo QCIF (176x144), 288 frames

The descriptors were generated by means of the created Matlab code.

Each test was repeated 10 times. For example, it was repeated 10 times the simulation for Foreman video, with temporal MDC technique and with a 10% of random packet loss. It should be considered to repeat these tests more times in order to obtain more rigorous values.

The main parameters defined for this simulation were:

- Codec: MPEG-4
- Packet size: 1024 bytes
- GOP: 7

The network simulation tool was NS-2. The simulation tools involved in the process and some execution command-line examples are presented in ANNEX H.

Once the simulation was finished, it was applied a discard criterion of frames depending on the type of frame which has been lost into the GOP. The criterions applied where the following ones:

- When a packet lost contains an I frame, all the GOP is lost.
- If the lost packet corresponds to a P frame, it is lost the P frame and all the following P frames till the next I frame arrives. To do that, it is used a Java code that analyses the output files of the simulation (rd) and compares it to the trace file (st) to detect looses.
- B frames are not considered to avoid buffering requirements. We reduce compression efficiency but we achieve reducing processing complexity and delay.

Finally, in order to evaluate the MDC techniques that were used in the simulation, it was calculated PSNR values in order to determine the objective quality obtained at reception. These results can be seen in ANNEX I.

The Simulation process used in this master thesis is inspired by [48].

5.1.2 MDC simulation without codec

Now, MDC simulation without codec is explained. The scenario is shown in Figure 5.2.

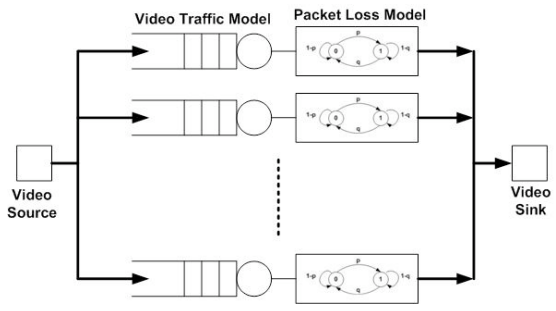

Figure 5.2 Simulation scenario

The only difference with the scenario described in Figure 5.1 is that now encoding and decoding stages are omitted.

There are defined parallel links where each descriptor follows a different path. In this simulation it is used 2, 4, 8, 16 and 24 descriptors, so there are defined as many links as number of descriptors.

Also, a packet loss model is defined in each path, which follows the random Bernoulli model. The packet loss percentages applied in this simulation were 1%, 5%, 10% and 15%.

For MDC simulations without codec are used the temporal, spatial and hybrid MDC techniques and the videos used are the following:

- Foreman QCIF (176x144), 288 frames
- Akiyo QCIF (176x144), 288 frames
- Carphone QCIF (176x144), 288 frames

Each test was repeated 3 times.

For this simulation each frame corresponds to one packet. In this case, for the simulation it is used a Java code to simulate packet loss. It is generated a random number that depending on the loss percentage it discards a packet (or frame). In this simulation ns-2 is not used.

Note that, in this case the discard criteria followed is that each frame corresponds to one packet, so if one packet is lost, only the corresponding frame is lost. We are not using any encoding phase, so all the frames generated by the MDC splitting process are of the same type. It can be said that each frame contained in each descriptor corresponds to an I frame (analogous to MPEG encoding terminology), but uncompressed.

Finally, PSNR values for different test were calculated. These results can be seen in ANNEX J.

5.1.3 Results

Several tests were realized in order to verify the correctness of the different designed MDC techniques (temporal, spatial and hybrid). The simulation scenario allowed us to test the transmission of several descriptors both, considering a MPEG-4 coding stage and omitting this stage. These descriptors were sent across different paths, with or without loss. In these tests we measured the Peak Signal to Noise Ratio (PSNR), which indicates the similarity between the transmitted and received sequence by

calculating the Mean Square root Error (MSE) pixel by pixel. It is a Full Reference quality metric.

PSNR and MSE are calculated according to the expressions 5.1 and 5.2, respectively.

$$PSNR[n] = 10 \cdot \log_{10} \frac{255^2}{MSE[n]} \qquad [5.1]$$

$$MSE[n] = \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{c=1}^{C} [F_n^{(c)}(i,j) - R_n^{(c)}(i,j)]^2}{N \cdot M \cdot C} \qquad [5.2]$$

where $MAX = 2^B - 1$, MAX is the maximum value that a pixel can take (255 is a typical value). N is the number of frames in a video sequence. M and C are the height and width of a frame.

PSNR denotes a value for calculating the objective quality of the transmitted frame or video sequence. If the subjective quality is required, we can convert the PSNR value to the correspondent value in terms of Mean Opinion Square (MOS) [49]. The conclusions inferred from the simulation results are the following ones:

- When considering a little or no motion video sequence (for example, Akiyo), the temporal technique works better than spatial and hybrid techniques. Nevertheless, when the video sequence presents some amount of motion (for example, Foreman), the best technique is the spatial one, although, when increasing the number of descriptors, the hybrid one outperforms the spatial technique.
- The Hybrid MDC technique presents better results than spatial technique when considering a little motion video sequence, but it presents worse results than temporal one (even when increasing the number of descriptors).
- When considering a sequence with some amount of motion, the hybrid technique performs a little bit worse than spatial technique but a little bit better than temporal one. However, when increasing the number of descriptors, the hybrid outperforms both, spatial and temporal techniques.
- Hybrid technique presents more redundancy data. This is why increasing the number of descriptors it presents very good results in comparison with other techniques.
- In a scenario with few losses (1%), the temporal technique is a little bit better than spatial one (without considering MPEG4 coding stage).

- When taking into account the MPEG-4 coding stage, PSNR values are lower than without considering it. However, the required bandwidth to send the descriptors is notably lower.
- PSNR values increase when the number of descriptors increase
- When the number of losses increases, PSNR values decrease
- PSNR values decrease more from 1% to 5% of loss than 5% to 10% and 10% to 15%. It seems to tend to a stable value.

To sum up, we can say that the results gathered from the simulation tests were satisfactory. It could be compared the designed techniques under the test environment described. Finally, it could be identified when to apply a technique or other, that is, we are able to choose the best video coding technique according to the nature of the video sequence to be transmitted. It can also be affirmed that the use of MDC coding is a good approach when considering a multipath environment because we can achieve better quality when increasing the number of descriptions. If we send, as was assumed, each description via different paths, we can also construct a more scalable and robust system in lossy environments.

The effectiveness of each method depends on the network conditions as well as on the characteristics of the video itself.

Finally, it is important to remark the next steps that should be carried on.

- Run more tests in order to obtain more reliable values from the test bank.
- Run unbalanced path / description tests
- Create a P2P simulation environment using tools such as PeerSim [50] or OverSim [51].

5.2 Point-to-Point Prototype Evaluation Results

This section makes an overview of the testing environment for the point-to-point MDC prototype described in CHAPTER 4. Finally, the obtained results are shown. This is a previous stage before integrating the whole prototype into the CoolRuc architecture.

5.2.1 Hardware Specification

The test of this module has been performed with the following computers:

Machine	CPU	RAM	SO
PC1	Intel Core 2 Quad @ 2.40GHz	2.00 GB	Windows XP
PC2	Intel Pentium 4 3.20GHz	1.00 GB	Windows XP

The differences of hardware resources for both machines are useful for testing the MDC module in different hardware conditions.

The tests have been performed through a 1Gbps LAN network.

5.2.2 Software requirements

The MDC module has been developed using the following programming environment:

- NetBeans 6.0
- Java JDK 6 Update 10
- Microsoft Visual C++ 2008
- Boost C++ Libraries 1.36.0
- ffmpeg r15169

The C++ native implementations have been developed and compiled using the Microsoft Visual C++ 2008 IDE. The JNI interfaces have been developed and compiled using NetBeans 6.0 IDE. Boost C++ Libraries are used for a better handling of the thread management.

Once the libraries are compiled, the DLLs and JARs were prepared in order to allow their insertion into the CoolRuc application.

The software requirements for its use are the following ones:

- Java JDK 6 Update 10
- Windows XP Operating System

5.2.3 Test description

The MDC module implementation works as following:

1. Splits the video into 4 different balanced substreams (descriptors)
2. Each substream can be displayed independently
3. The splitting is performed according to the spatial technique

4. Each substream is encoded and sent independently
5. At the receiver side the substreams are decoded and merged
6. The applied codec is MPEG-4

Two machines have been used within the same LAN network. The scenario is shown in Figure 5.3.

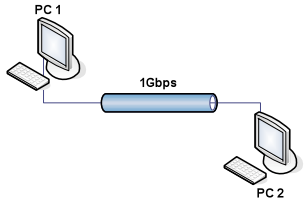

Figure 5.3 LAN Scenario

The characteristics of the video sequences and the performed tests are detailed in Table 5.1.

Table 5.1 Characteristics of the video sequences used and details for the performed tests

First video sequence	Second video sequence
Foreman.yuv Frames: 280 Resolution: 176 x 144 Raw video rate (one substream): 1.9 Mbps Encoded video rate (one substream): 529 Kbps	hgast.yuv Frames: 1000 Resolution: 640 x 480 Raw video rate (one substream): 23.04 Mbps Encoded video rate (one substream): 753 Kbps
Performed tests	
Test 1: PC1 does the splitting and sends to PC2 that does the merging. **Test 2:** PC2 does the splitting and sends to PC1 that does the merging.	**Test 3:** PC1 does the splitting and sends to PC2 that does the merging. **Test 4:** PC1 does the splitting and sends to PC2 that does the merging.

5.2.4 Results

The main parameters evaluated in this test are RAM, CPU (performance of the application) and PSNR (objective quality measurement) of the received video.

a) RAM

The following tables show the RAM consumption for both the splitter and the merger.

- Splitting

Table 5.2 RAM consumption for Splitter

Test	Initial RAM	RAM peak	RAM consumption
Test 1	50 MB	200 MB	76 MB
Test 2	65 MB	310 MB	90 MB
Test 3	120 MB	359 MB	150 MB
Test 4	170 MB	390 MB	185 MB

- Merging

Table 5.3 RAM consumption for merger

Test	Initial RAM	RAM peak	RAM consumption
Test 1	40 MB	250 MB	60 MB
Test 2	30 MB	210 MB	70 MB
Test 3	100 MB	300 MB	150 MB
Test 4	95 MB	330 MB	110 MB

As it can be seen from the results there is no big difference between PC 1 and PC 2 in terms of RAM consumption. It can be said that splitting process is a little bit more RAM consuming than the merging process (7% more aprox.).

b) CPU

The following tables show the CPU consumption for both the splitter and the merger.

- Splitting

Table 5.4 CPU consumption for Splitter

Test	Initial CPU	CPU peak	CPU average
Test 1	10%	30%	22%
Test 2	25%	73%	45%
Test 3	33 %	72 %	41 %
Test 4	59 %	89 %	65 %

- Merging

Table 5.5 CPU consumption for Merger

Test	Initial CPU	CPU peak	CPU average
Test 1	12%	32%	15%
Test 2	8%	15%	11%
Test 3	15 %	37 %	29 %
Test 4	10 %	43 %	23 %

As it can be seen, CPU consumption is more critical and PC 2 reaches higher levels of CPU. The splitting process is more CPU demanding than the merging process. The big difference between Splitting and Merging processes was produced by the Frame recognition (Frame Marker sub-module) and marking process introduced to improve the operation of the system against decoder weakness in the Splitter Module. It analyses the MPEG-2 generated bytestream, and it introduces an additional computational cost. So, a powerful node in order to split high quality media (source node) is needed.

c) Objective Quality Measurement

The first test (video 1) has been performed twice and the results are 30.102 dB and 29.878 dB which indicates a MOS equivalent of fair quality (25 – 31 is fair quality and 31 – 37 is good quality).

For the second test (video 1) the two obtained results are 29.982 dB and 30.003 dB which again are on the MOS fair quality range.

For the third test (video 2) the two obtained results are 42.768 dB and 41.986 dB which indicates a MOS equivalent of excellent quality.

For the fourth test (video 2) the two obtained results are 42.958 dB and 42.114 dB which again are one the MOS fair quality range.

These tests have been performed with the first release of the MDC module. This first release has a synchronism problem that was fixed in the final version. Because of that problem the PSNR results are lower than expected according to the Simulation results. We realized more tests (**¡Error!No se encuentra el origen de la referencia.**) with the final release of the MDC module, but the PSNR measurements could not be gathered. However, the perceived quality using the last release was excellent (except when the computers where overloaded, CPU at 100% of usage).

5.2.5 Conclusion

After the realization of the tests described above and the study of the obtained results, several conclusions can be obtained.

Regarding to RAM consumption it can be seen from the results that the machines are not a bottleneck.

About CPU consumption, it can be said that it is more critical and it can be seen that the machine with less CPU has a worst performance. The splitting process clearly consumes more CPU in both cases. So, the more powerful the machines are, the more quality can be handled (transmitted and received). The system limitation is focused in processing capacity, not the used technique. Some additional tests were done in order to verify HD transmission using MDC, but the results were not registered. These tests allowed to distribute HD media (1280 x 720) using the Quad Core (PC1) computer as source node (in charge of splitting process), although the CPU consumption was high too (85%). The Intel Pentium 4 (PC2) was completely overloaded and could not carry this task on.

Finally, quality measurement tests have been performed with the first release of the MDC module. This first release has a synchronism problem that was fixed in the last version. Because of that implementation problem, the PSNR results are lower than expected according to the simulation results.

We carried out more tests in order to verify the final MDC Module when splitting different resolutions. We could appreciate an excellent visual quality (subjective perception) using the MDC Module, but we did not gather PSNR measurements. The resolutions that were tested were the following ones:

Table 5.6 Resolution testing

Resolution	bitrate [kbp]	Visual (Subjective) perception
CIF [352X288]	800	OK - excellent
	1500	OK – excellent
	3000	OK – excellent
	5000	OK – excellent
	8000	OK – excellent
VGA [640X480]	1500	OK- excellent
	3000	OK- excellent
	5000	OK- excellent
	8000	OK- excellent

Resolution	bitrate [kbp]	Visual (Subjective) perception
720x480	3000	OK- excellent
	5000	OK- excellent
	8000	KO (CPU at 100%)
DV [720X576]	5000	OK- excellent
	8000	KO (CPU at 100%)
HD [1280 x 720]	10000	OK (using Quad Core splitter, source node)

In addition, we made some tests using a DV camera. The tests were done fixing the native resolution to 720x576. According to visual perception, the visual quality was excellent.

5.3 P2P Integrated System Evaluation Results

In order to validate the correctness of the proposed solution, it was compared a Single Description (SD) transmission with a MDC balanced spatial based transmission. This section describes the testbed and the gathered results.

5.3.1 Testbed

The testbed used during the realization of the measurements is shown in Figure 5.4.

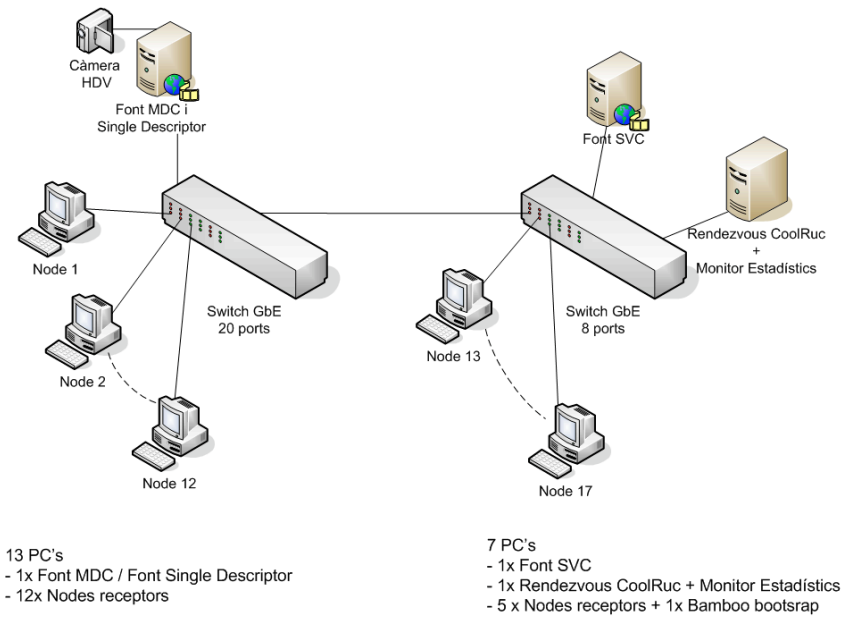

Figure 5.4 Physical topology of the test scenario

The statistics are generated by a centralized service. The architecture is presented in Figure 5.5. This system was implemented by the LCFIB-UPC research group.

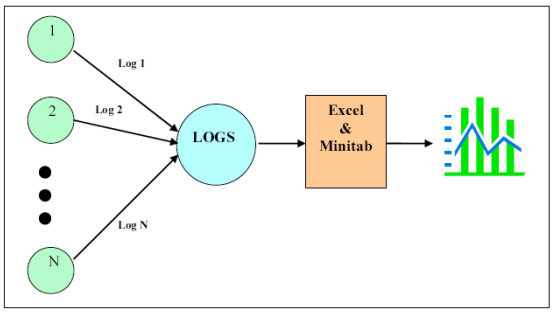

Figure 5.5 Statistics generation process

Basically, all nodes [1,N] send their logs to a centralized process allocated in a machine (Statistics monitor in Figure 5.4).

5.3.2 Metrics

The main metrics considered to evaluate the performance of the system in presence of losses are the following ones:

a) Continuity Index

This metrics quantifies the continuity of media. It requires storing the number of received chunks and the number of visualized chunks, discretized by time.

$$ContinuityIndex = \frac{receivedChunks}{totalChunks} \qquad [5.3]$$

In MDC, the discontinuity occurs only if no descriptor chunk is received from any of the four descriptors distributed over the P2P network (channel).

b) Missed Chunk Rate

Missed chunk rate measures the number of non received chunks over the total of the expected chunks, discretized by time.

$$MissedChunkRate = \frac{missedChunks}{totalChunks} \qquad [5.4]$$

For a Single Description transmission, the missed chunk rate and continuity index are the same as can be seen next:

a) $missedChunks = totalChunks - receivedChunks$, only for a single description distribution.

b) $MissedChunkRate = \frac{totalChunks - receivedChunks}{totalChunks} = \frac{totalChunks}{totalChunks} - \frac{receivedChunks}{totalChunks} =$

$= 1 - \frac{receivedChunks}{totalChunks}$

c) $MissedChunkRate = 1 - ContinuityIndex$

Note that this is only valid for Single Description. As it has been said before, when using MDC technique the Continuity Index must take into account the descriptor which a frame belongs to. So, if it is received only one frame of one descriptor, the continuity index is maintained despite missing the rest of the frames belonging to the rest of the other descriptors. In this thesis we proved that MDC improves the continuity index under loss effect.

5.3.3 Parameters of the experiment

Table 5.7 describes the parameters considered in the experiments that were carried out:

Table 5.7 Parameters of the experiment

Experiment Time (T)	Interval time
Broadcasters (B)	Number of channel broadcasters
Consumers (C)	Number of consumers of a channel
Stream Bit Rate (SBR)	Chunks per second
Buffer Time (BT)	Size of the buffer in seconds
Partnership size (PS)	Maximum number of partners of a peer
Bandwidth per peer (BW)	Bandwidth in bps
Media Bit Rate (MBR)	Bit Rate of the media in bps

The values that were fixed in the MDC experiments were the following ones:

Table 5.8 Fixed values

Experiment Time (T)	10 minutes
Broadcasters (B)	1
Consumers (C)	17
Stream Bit Rate (SBR)	8 chunks / second
Buffer Time (BT)	10 seconds
Partnership size (PS)	8
Bandwith per peer (BW)	1000000
Video Resolution	176 x 144 pixels (QCIF)
Number of descriptors	4
Raw Media Bit Rate	1.9 Mbps (each descriptor)
	7.6 Mbps (total)
Encoded Media Bit Rate (MBR)	529 Kbps (each descriptor)
	2.1 Mbps (total)
Transport protocol	MPEG-TS/TCP

5.3.4 Results

In this test scenario we could confirm that the use of MDC video coding schemes allowed to improve the continuity index in a P2P environment.

Comparing the continuity index with the missed chunk rate, it is observed that the highest missed chunk rate is 0,1569 so the continuity index is 0,8431. But with MDC, when the missed chunk rate is 0,2365, the system has a continuity index of 0,8387. The continuity index should be worse using single description with this missed chunk rate value.

Then, we can state that MDC are more robust to chunk loss and should adapt the video quality to heterogeneous receptors.

It must be said that the gathered results were not using the last release of the implemented MDC prototype. The final release offered better results as we could see empirically and looking at some collected logs.

Additionally, we implemented a loss generation function in CoolRuc application which allowed us to discard chunks at the receiver [0% - 100%]. Some initial tests that we have done at testing time varying the PLR bar allowed to observe that the continuity index was greatly improved when introducing 5%, 10% and 20% of chunk rate loss.

A more detailed comparative of the obtained results can be seen in ANNEX K.

CHAPTER 6. CONCLUSIONS AND FUTURE WORK

6.1 Realized work

The proposed objectives and tasks (see Table 1.1) were successfully achieved. In this master thesis we made an overview of two source video coding techniques used for media distribution over IP networks, concretely suitable for P2P environments in order to provide efficient video streaming solutions: Multiple Description Coding (MDC) and Scalable Video Coding (SVC).

In this work MDC was used for implementing the live streaming module because:

- it provides robustness in transmission, it is an error-resilient technique
- it provides scalability
- it is a flexible solution which can be adjusted in many ways
- it is codec independent
- it allows real-time coding of live content
- it allows real-time coding of prerecorded content
- it is feasible at implementation stage: SVC encoding is slower due to computational cost. Nowadays, Real time SVC coding is only possible by means of specific hardware components (and just up to SD resolution).

There were analyzed three different MDC techniques (spatial, temporal and hybrid) and were also simulated in order to evaluate a first design. Thanks to the conclusions inferred from this stage, it was implemented a first point-to-point prototype (using 4 balanced spatial descriptors). After its evaluation, it was integrated into a real P2P streaming system (CoolRuc), as the MDC-based Video Coding Component into the architecture of a node (Splitter and Merger MDC modules). Hence, it was possible to make a point-multipoint live streaming. All the evaluation stages allowed to validate the improvement of the transmitted system by means of measuring the received objective quality (PSNR) and the improvement of the continuity index in the P2P application. Afterwards, it was tuned the integrated modules in order to improve its operation in the whole system (implementation issues). The final integrated prototype was able to distribute up to HD content (1280 x 720 resolution) and it was checked its benefits against loss effect. It is important to point out that the main limitation when scaling in quality (spatial resolution) was the required CPU, as the splitting and merging processes are demanding operations. Concretely, the Splitter Module is the most CPU

consuming module, which is executed at the source node. However, it is usual that, when very-high quality transmission is required, the source node presents advanced capabilities in comparison with receivers (so, this cost can be assumed by the source). So, the main problem is not the used technique, is the required hardware.

Finally, during the elaboration of this master thesis, the author contributed to research in P2P and media issues. The generated publications are referenced in ANNEX M.

6.2 Work conclusions

Understanding of source-coding techniques and related challenges (dynamicity, heterogeneity, etc.) helps to the choice of adequate techniques that will help to optimize resource usage and to improve substantially the overall P2P streaming system quality and performance.

Error propagation and packet loss take place frequently over today IP networks. These effects can considerably reduce the video quality at the receiver nodes. According to this, handling packet loss and providing error resilience are critical issues in streaming applications and more specially when talking about P2P due to high dynamicity and heterogeneity operating under harsh conditions. At this point, MDC and SVC source video coding techniques are solutions which have been developed to provide scalability and robustness to video transmission, which allow to enhance the overall quality and to protect multimedia traffic against severe network conditions. These techniques can deal with heterogeneous environments well, so, they are appropriated to be applied in P2P streaming environments, allowing to overcome some of the above mentioned problems.

MDC improves loss resilience because each generated bitstream (description) can be decoded independently, then, it is difficult to have the same amount of corrupted data in each transmitted description. Information follows different paths in the P2P network, under different conditions (MDC exploits path diversity). In contrast, SVC can improve error resilience when the protection level for a given layer can be adapted to its importance so that, the base layer is more protected (it can be sent, for instance, through multiple paths: wireless or wired).

These schemes allow to distribute media resources in heterogeneous networks such as P2P ones where many different kind of devices are connected (from full-featured desktop PC to a limited mobile device such as a PDA). Every client connected to the network can have different features or capabilities (two of the most common ones are available bandwidth depending to the network access and display resolution).

Therefore, each peer in a P2P network can visualize as many descriptors or layers as its features and conditions allow. That is, the more capabilities (network and terminal) the peer has, the more descriptors or layers can merge in order to obtain better user experience (better resolution can be displayed). For instance, if a peer receives just a single one descriptor, it will be able to display a CIF resolution resource, however, if it receives three descriptors it will be able to achieve higher resolutions (Standard Definition, High Definition and so on). The granularity used in the original media splitting process yields to the possible combinations. According to this, note that these schemes allow distributing many video qualities into the network simultaneously (from low quality such as CIF or QCIF up to current Standard Definition, High Definition, etc.). In some way, notice that there is a coarse-grained transcoding implicit in that process which allows to “adapt” (run at receiver-side) the content to the different peers on the network according to their capabilities.

When applying these kinds of techniques, the user experience is not only improved in terms of better resolution achieved. The user experience is improved when talking about continuity at viewing time. Suppose there are lots of losses in the network and the user is displaying a media resource sent via the P2P application. At a given instant, it can stop receiving one out of many descriptors but the resource can continue being displayed although at less resolution (less quality, but the reception continues). Once the network conditions are restored, and more descriptors can be received, the resource can add the new descriptors received in order to improve the resolution (better quality). Note that MDC and SVC approaches behave differently in that situation. MDC offers completely independence among descriptors, that is, just one descriptor is necessary to start receiving a valid bitstream to be displayed. In contrast, the dependency among layers introduced by SVC adds a specific constraint that consists on the need of receiving layers in a specific order, that is, first it must be received the base layer (minimum required) and then, the successive dependent enhancement layers:

base layer
base layer + enhancement layer 1
base layer + enhancement layer 1 + enhancement layer 2
base layer + enhancement layer 1 + enhancement layer 2 + … + enhancement layer N

Independence among descriptors supposes an advantage for MDC approach, especially in P2P network environments due to its dynamicity and heterogeneity. In

P2P networks it cannot be guaranteed the reception of any specific descriptor so, independence is a key feature. Specifically, it is well suited for high dynamic mesh-based applications such as the one used for the presented implementation (CoolRuc). It is a more flexible and adaptative solution.

Another key point that makes MDC a very interesting approach consists in its codec independency. This feature is strongly suitable for transmitting almost any kind of media, encoded with different codecs, allowing many applications to be deployed in order to distribute different media resources. This is specially important to allow the development of new media application such as the ones supporting future media 3D resources (3DTV applications).

To sum up, it can be said that MDC and SVC approaches are suitable for P2P environments (as could be seen in TRILOGY WP2 Project), both structured (for instance those based on tree structures) and unstructured ones (for instance based on chunk spreading over the network in a mesh-based overlay). They provide very interesting features such as scalability, redundancy when transmitting and codec-independency. However, the main disadvantage of these techniques is that they introduce a new processing stage before transmitting and this operation can be expensive in terms of processing cost and, consequently, extra-delay can be added at transmission time (there must be a trade-off between processing cost and the overall introduced delay). This can be a critical point when talking about a live streaming service, specially when interactivity is required. Nevertheless, the choice between the two techniques will be determined by the specific application or service considered and according to specific requirements.

6.3 Future work

According to research issues, it is proposed to do research in the following points:

- Study incentive mechanisms to improve P2P operation in terms of global fairness and performance (avoid free-riding for instance), specially when working with descriptions or layers [work in progress].
- Study interpolation techniques to allow improving the received quality (depending on the type of losses produced). It must be evaluated if the adoption of these kinds of techniques allows increasing the received quality noticeably, despite introducing higher latency and processing load. Note that MPEG group only takes into consideration improvements of more than 0.5 dB.

- Research on homogenising the overall delay. Currently, each peer experiences different delays at playback time. This can be an important problem for instance when a group of users are watching the same channel but at different time spaces and they want to interact by means of interactive services (such as Instant Messaging).
- Reduce start-up delay [work in progress].
- Enable fast Zapping between channels by means of unbalanced MDC [work in progress].

Regarding to implementation, it is proposed to cover the following aspects in future versions of the current prototype:

- Incorporate audio in the Media Coding System. Currently, MDC is only applied to video.
- Make the MDC Merger Module more flexible in order to allow selecting the number of descriptors to be used. Now it can just work with 4 or 1 descriptor.
- Make more quality measurements using the whole P2P system (CoolRuc)
- Make more tests considering more restrictive network conditions, for instance, stressing the network, in order to check the behaviour of the whole application and to identify new vulnerabilities.
- Make massive tests (by means of platforms such as PlanetLab or by making tests with real users as betatesters).

REFERENCES

[1] PPLive official site, www.pplive.com (last visit July 2009)

[2] PPStream official site, www.ppstream.com (last visit July 2009)

[3] UUSee official site, www.uusee.com (last visit July 2009)

[4] Pando official site, www.pando.com (last visit July 2009)

[5] Zattoo official site, http://zattoo.com/ (last visit July 2009)

[6] Youtube official site, www.youtube.com (last visit July 2009)

[7] Cachelogic official site, www.cachelogic.com/ (last visit July 2009)

[8] Bitttorrent official site, www.bittorrent.com/ (last visit July 2009)

[9] Internet Study 2008-2009 by Ipoque, http://www.ipoque.com/resources/internet-studies (last visit July 2009)

[10] CISCO IP traffic forecast, http://newsroom.cisco.com/dlls/2009/prod_060909.html (last visit July 2009)

[11] TRILOGY site: http://www.i2cat.cat/i2cat/servlet/I2CAT.MainServlet?seccio=21_53 (last visit July 2009)

[12] M. van der Schaar and P. Chou, “Multimedia over IP and Wireless Networks: Compression, Networking, and Systems” 2007.

[13] A. Ríos, A. J. González, and J. Alcober. “Prototype of P2P Multiconferencing System based on SIP”. Telecom I+D 2008, Bilbao, Spain (Local Conference in Spanish). ISBN 978-84-9860-135-0, October 2008.

[14] A. Rios. A. J. González, A. Oller, J. López, and J. Alcober. “P2P Multipoint Conference System using SIP”. HETNET 2008. Kalskrona, Sweden, February 2008

[15] A. Rios, A. J. González, and J. Alcober. “Peer-to-Peer Multimedia Conferencing System based on SIP Signaling”. Traffic and Performance Engineering for Heterogeneous Networks, Three Research Volumes by River Publishers, February 2009. ISBN 978-87-92329-16-5. (Book Chapter)

[16] J. Liu, S. Rao, B. Li, and H. Zhang, “Opportunities and Challengesof Peer-to-Peer Internet Video Broadcast” *Proceedings of the IEEE,* vol. 96, pp. 11-24, Jan. 2008.

[17] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. Mohr, Chainsaw: “Eliminating Trees from Overlay Multicast” *Lecture Notes in Computer Science*, vol. 3640, p. 127, 2005.

[18] X. Zhang, J. Liu, B. Li, and Y.-S. Yum, “CoolStreaming/DONet: adata-driven overlay network for peer-to-peer live media streaming” *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, pp. 2102-2111, vol. 3, March 2005.

[19] L. Zhao, J.-G. Luo, M. Zhang, W.-J. Fu, J. Luo, Y.-F. Zhang, and S.-Q.Yang, “Gridmedia: A Practical Peer-to-Peer Based Live Video Streaming System” *Multimedia Signal Processing*, 2005 IEEE 7th Workshopon, pp. 1-4, Oct. 2005.

[20] F. Picconi and L. Massoulié, “Is There a Future for Mesh-Based liveVideo Streaming?” *in Proceedings of the 2008 Eighth International Conference on Peer-to-Peer Computing-Volume 00*, pp. 289-298, IEEE Computer Society Washington, DC, USA, 2008.

[21] Y. Guo, C. Liang, and Y. Liu, “AQCS: Adaptive Queue-Based Chunk Scheduling for P2P Live Streaming” *Lecture Notes in Computer Science*, vol. 4982, p. 433, 2008.

[22] “Standard-compatible Multiple-Description Coding (MDC) and Layered Coding (LC) of Audio/Video Streams”, A. Vitali, M. Fumagalli, Internet Draft (Expired), http://tools.ietf.org/tools/rfcmarkup/rfcmarkup.cgi?draft=draft-vitali-ietf-avt-mdc-lc-00.txt#page-18 (last visit July 2009)

[23] J. Rosenberg, C. Huitema, R. Mahy, and D. Wing, “RFC 3489: Simple Traversal of UDP Through Network Address Translators(NAT)(STUN)” 2006.

[24] S. Microsystems, “JXTA Community Projects.” https://jxta.dev.java.net/, 2001.

[25] E. Kohler, M. Handley, and S. Floyd, “RFC 4340: Datagram congestion control procotol” *IETF Internet Standards*. IETF, 2006.

[26] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, “RFC2960: Stream Control Transmission Protocol” RFC Editor United States, 2000.

[27] Y. Wang, A. Reibman, and S. Lin, “Multiple Description Coding for Video Delivery” *Proceedings of the IEEE*, vol. 93, no. 1, pp. 57-70, 2005.

[28] V. K. Goyal, "Multiple Description Coding: Compression Meets the Network," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 74 -- 94, Sept. 2001.

[29] R. Puri and K. Ramchandran, “Multiple description source coding through forward error correction codes,” IEEE Proceedings Asilomar Conference on Signals, Systems, and Computers, Asilomar, CA, October 1999.

[30] A.L. Vitali, Borneo A., Fumagalli M., Rinaldo R. “Video over IP using standard-compatible multiple description coding”: an IETF proposal 2006.

[31] LU Meng-ting, LIN Chang-kuan, YAO Jason, CHEN Homer H. “Multiple description coding with spatial-temporal hybrid interpolation for video streaming in peer-to-peer networks”, 2006.

[32] S. Shirani, M. Gallant, F. Kossentini, “Multiple Description Image Coding Using Pre- and Post-Processing”. *International TCC. Las Vegas, Nevada, USA* 2000.

[33] Zhao Anbang, Wang Wensheng, Cui Huijuan, Tang Kun. “Efficient Multiple Description Scalable Video Coding Scheme Based on Weighted Signal Combinations” 2007.

[34] Kawada R, Koike A, Wada M, Matsumoto S., “An efficient parallel video transmission system including codecs with functions of failure detection and coding noise evaluation”. IEEE Trans. Circuits Syst. Video Technol., 2005, 15(11):1341-1353.

[35] V.A. Vaishampayan “Design of multiple description scalar quantizer” *IEEE Trans. Inform. Theory* 1993, 39(3):821-834. [doi:10.1109/18.256491]

[36] Zhengye Liu, Yanming Shen, Shivendra S. Panwar, Keith W. Ross and Yao Wang. “Using Layered Video to Provide Incentives in P2P Live Streaming”, in *SIGCOMM*, 2007.

[37] W. Jiang, A. Ortega “Multiple Description Coding via Polyphase Transform and Selective Quantization” *Proc. VCIP 99,* 1999.

[38] N. Franchi, M. Fumagalli, R. Lancini, S. Tubaro “Multiple description video coding for scalable and robust transmission over IP”. *IEEE Trans. on CSVT*, 15(3): 321-334, 2005.

[39] X. Yang, L. Ramchandran “Optimal Multiple Description Subband Coding”. *Proc. of IEEE ICIP 1998*. Chicago, IL.

[40] R. Bernardini, M. Durigon, R. Rinaldo, P. Zontone, A. Vitali, “Multiple Description Coding of Video Streams Using Frames and the H.264/AVC Standard.” *Proc. ICIP 2004*. Singapore.

[41] X. Tang, X.A. Zakhor “Matching Pursuits Multiple Description Coding for Wireless Video”. *Proc. ICIP 2001*. Thessaloniki, Greece.

[42] H. Schwarz, D. Marpe and T. Wiegand, “MCTF and Scalability Extension of H.264/AVC” *Proc. Picture Coding Symposium,* San Francisco, USA, December 2004

[43] W. Gao and L. Huo, “Challenges on Peer-to-Peer Live Media Streaming”, *Lecture notes in Computer Science*, vol. 4577, p. 37, 2007.

[44] F. Pianese, D. Perino, J. Keller, and E. Biersack, “PULSE: An Adaptive, Incentive-Based, Unstructured P2P Live Streaming System” *Multimedia, IEEE Transactions* on, vol. 9, no. 8, pp. 1645-1660, 2007.

[45] V. Reguant, F. Prats, R. de Pozuelo, F. Margalef, and G. Ubiergo, “Delivery of H264 SVC/MDC streams over wimax and DVB-T networks” *Consumer Electronics, 2008. ISCE 2008. IEEE International Symposium* on, pp. 1-4, April 2008.

[46] D. Bryan, P. Matthews, E. Shim, and D. Willis, “Concepts and Terminology for Peer to Peer SIP” *draft-ietf-p2psip-concepts-00* (work in progress), July, 2007.

[47] X. Milà, J. F. Crespo, A. J. González, A. Ríos, F. Enrich, A. Vidal, “A Peer-to-Peer Live Streaming Platform for High-Quality Media Services” in *Network & Electronic Media*, 2008

[48] MDC simulation environment, http://140.116.72.80/~smallko/ns2/MDC.htm (last visit July 2009)

[49] Mean Opinion Score (MOS) definition, http://en.wikipedia.org/wiki/Mean_opinion_score (last visit July 2009)

[50] Peersim official site, http://peersim.sourceforge.net/ (last visit July 2009)

[51] Oversim official site, http://www.oversim.org/ (last visit July 2009)

ANNEX A. Fundació i2CAT and TRILOGY Project Overview

Fundació i2CAT

The Fundació i2CAT is a non-profit organisation whose aim is to promote research and innovation in advanced Internet technology. i2CAT promotes the deployment of services and wideband applications from both public and private research and innovation communities. The aim is to encourage telecommunication, state operators as well as businesses to make infrastructures and experimental services available. The i2CAT Foundation is a research and innovation centre organised on a network basis in order to:

- Develop research and innovation projects.
- Promote and maximise the use of advanced research in the area of networks and wideband applications.
- Create platforms for collaboration between the business sector and universities.
- Promote working teams in association with institutions in the rest of the world whose aims and research are in line with those of the Foundation.

TRILOGY Project

The main purposes of the TRILOGY project are the study, development and evaluation of Peer-to-Peer (P2P) architectures for high quality media contents in live streaming distribution. TRILOGY provides novel solutions related to media coding, meta-data distribution, content adaptation, improvements on P2P transport plane and signalling. TRILOGY is a multidisciplinary main research project aimed at three specific thematic areas: Optical Networks, Audiovisual P2P Services and Sensor Networks. These three areas try to study in a complementary way from the physical layers to the service layers of the future or next generation Internet. TRILOGY is a project of 19 months duration, from October 2007 to April 2009, and counts on the work of 7 research groups coming from 3 different universities: Grup de Comunicacions Òptiques (GCO)

from UPC, Grup de Xarxes Òptiques de BAMPLA (GXO-BAMPLA) from UPC, Network Technologies and Strategies (NeTS) from UPF, Tecnologies Audiovisuals i Multimedia (TAM) from URL, Laboratori de Càlcul of the Information Technology Faculty of Barcelona (LC-FIB) from UPC, Media de BAMPLA (MEDIAENTELBAMPLA) from UPC and Xarxes Sense Fils (XSF) from UPC. This tesis is situated inside TRILOGY project, specifcally, in Audiovisual P2P Services thematic area. Starting with the CoolRuc P2P application developed in the research project from the i2CAT Foundation called MACHINE, some open points required more research in order to improve their performance in different parts:

- New advances in video coding techniques which allow providing robust and scalable solutions for P2P media distribution, by MediaEntel and TAM research groups.
- Metadata search has to be improved finding a compromise between search speed and accuracy of the query results, by TAM research group.
- A better suitable adaptation of the media data delivered to end user (content adaptation), by TAM, NeTs and LCFIB.
- A scheduling with different chunk selection strategies together with incentives to improve the playback continuity and to reduce start-up latency, by LCFIB and MediaEntel.
- The use of an overlay based in ongoing IETF standards is also a quite interesting issue to improve the applications in order to be used in several scenarios using a common SIP interface, by XSF research group.

TRILOGY project specify a new architecture for CoolRuc (2.0), integrating the last results in research from the participating Research Groups from three Universities, and offering to i2CAT companies and institutions to participate in a fresh and challenging applied-research project.

ANNEX B. Connectivity

Other issues to consider in order to implement a real P2P live streaming are the connectivity between peers, and the transport protocols used for exchanging data.

The connectivity between peers must take into account existing NATs and Firewalls. Solutions such as STUN protocol [23] allow overcoming connections restrictions between peers under the influence of NAT. There exist also platforms such as JXTA [24] that allows creating an easy P2P solution which can work through NAT and Firewalls.

Related to transport protocols, P2P live streaming solutions normally use TCP and UDP. The start-up delay can increase with TCP, but UDP is a datagram oriented protocol and it does not offer control over the streaming. Datagram Congestion Control Protocol (DCCP) [25] and Stream Control Transmission Protocol (SCTP) [26] are alternative transport protocols which can be considered to implement a solution.

ANNEX C. MDC vs. LC Comparative

This section provides a comparative between MDC and SVC according to the results extracted from [30]. The comparative regards to generic layered coding (LC) where SVC can be considered as a specific type of LC defined inside the standard MPEG-4/AVC.

The aim of this comparative is to obtain a global vision of MDC and SVC coding techniques and extract conclusions of their different applications.

a) MDC vs. LC

In order to combat channel-induced impairments, Layered Coding (LC) and Multiple Description Coding (MDC) have been proposed as source coding techniques that are robust against inevitable transmission errors. In contrast to a conventional media coder that generates a single bit-stream, LC and MDC encode a media source into two or more sub-bit-streams. For LC, one base layer bit-stream and several enhancement layer bit-streams are generated. The base layer can be decoded to provide a basic quality of video while the enhancement layers are mainly used to refine the quality of the video that is reconstructed from the base layer. If the base layer is corrupted, the enhancement layers become useless, even if they are received perfectly. Therefore, the base layer is critically important and is usually protected using either Automatic Repeat Request (ARQ) or Forward Error Correction (FEC) codes. Compared to LC, MDC has the following unique properties: the descriptions are mutually refining, equally important, and independent. Each description can be independently decoded. There is no decoding dependency between any two of the descriptions. MDC usually does not require prioritized transmission, except when there is a specific application requirement.

b) LC+FEC vs. MDC+FEC

FEC-coded LC with MDC has been analysed and compared on a Binary Symmetric Channel (BSC) and a Random Erasure Channel (REC). For the BSC, Cyclic Redundancy Check (CRC) codes and Rate-Compatible Punctured Codes (RCPC) have been used together to detect and recover corrupted packets. For the REC, Hamming codes have been used to recover lost packets. The encoding method used for LC has been similar to the SNR-scalability method defined in the MPEG-2 standard. The channel coding has been applied to both layers but the base-layer data were protected with stronger channel codes. For MDC, two descriptions have been

generated and were equally protected. The conclusion inferred from this study is that MDC was more effective than LC only in very high error probability situations, and it was suggested that a Multiple Description source coder should use channel coding to obtain acceptable performance on memory-less channels.

c) LC+ARQ vs. MDC

In this case, for LC, the SNR scalability method defined in the H.263 standard was adopted. For MDC, it was used a multiple description motion compensation coder. It has been compared the performance of LC, MDC, and LC with ARQ, where ARQ was used to protect the base-layer data only. The simulation results show that MDC was better than LC when the underlying application had a very stringent delay constraint and that the RTT on each path was relatively long. However, LC performed better than MDC when limited retransmission of the base-layer was acceptable.

d) LC+FEC/ARQ vs. MDC+FEQ/ARQ

Here it was carefully compared the performance between LC and MDC in multi-path environments at various packet loss rates. Two different video codes were chosen for LC and MDC: hybrid transformation and motion compensation codec and a wavelet-based video codec. Three different error protection scenarios were also considered for both LC and MDC: no error protection, ARQ-based error protection, and FEC-based error protection. According to the simulation results, MDC was more suitable for delay-sensitive applicants or for which the underlying transmission channels do not support a feedback link, while LC may be a better choice for applications that can tolerate a certain amount of delay. If FEC-based error protection can be used, MDC and LC perform similarly. However, MDC is preferred at high error rates because of its better error-resilience at those rates.

e) MDC vs FEC

MDC was compared to Single Description whose robustness is increased by Reed-Solomon FEC at application layer (Figure C.1). Experiments showed that for video sequences with high motion content the inefficiency of MD could make the SD+FEC preferable. However the decoded quality of MD has a lower variance and it is higher, especially at high loss rates (>5%). MD is preferable to compensate for long burst of losses and in the presence of on-off channels.

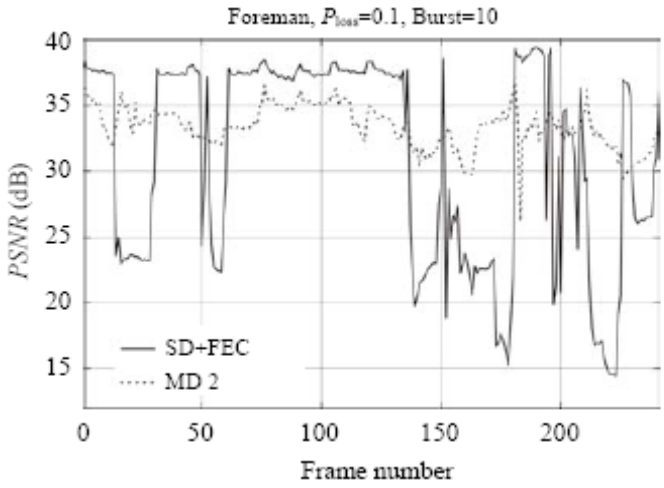

Figure C. 1 Two descriptions (MD 2) and single description plus (SD) Reed-Solomon FEC. PSNR frame by frame. As can be seen the decoded quality of MD is higher for high loss rate (>5%) and the variance is lower [30]**.**

Performance summary

Summarizing the results from these studies, they conclude that MDC and SVC/LC have some aspects in common, such as the sub-streams representation and the intrinsic scalability, but they differ in error-resiliency capability and the importance of having a feed-back channel. In general, about the performance, it can be stated that MDC has advantages over LC for networks with no feedback or a long RTT, or for those applications that have very stringent delay constraints. However, if the networks or the applications support prioritized transmission or error control, LC might be better than MDC (but in MDC, prioritization can be implemented at application level).

ANNEX D. Example of MDC system

Here it is presented a simple example of a MDC system (Figure B.1). The System is composed by the following main blocks:

Table D. 1 Blocks composing the MDC system

TRANSMITTER	**Source:** the source contains the original video resource in raw data (for example) in order to make easier the splitting process.
	Splitter: this element divides the original source into different sub-streams according to specific rules. These rules consist on the type of MDC that is going to be applied. This stage on the coding chain supposes a pre-processing stage before encoding the resulting media stream. Many options are available, some examples are the following ones: - Even-odd pixel selection - Temporal frame selection - Wavelet based
	Encoder: this element encodes the sub-stream generated in the splitting stage.
	Parser: this block packetizes the data generated by the encoder in order to distribute the stream into the network. This block can add application-level overhead necessary to identify the data/substream.
RECEIVER	**Decoders:** the receiver decodes the received sub-streams according to the specific codec used in the encoding stage on the transmitter.
	Merger: this element constructs a new stream by mixing the different decoded sub-streams which are received according to the type of MDC which was applied in the emitter. This stage allows to add additional process to the received descriptors in order to achieve better performance at reconstruction time, for example, if just 2 out of 3 descriptors are received, some pixels can be interpolated in order to reduce the effects of losing a descriptor.
	Player: once the image is merged, the resulting media stream is able to be displayed.

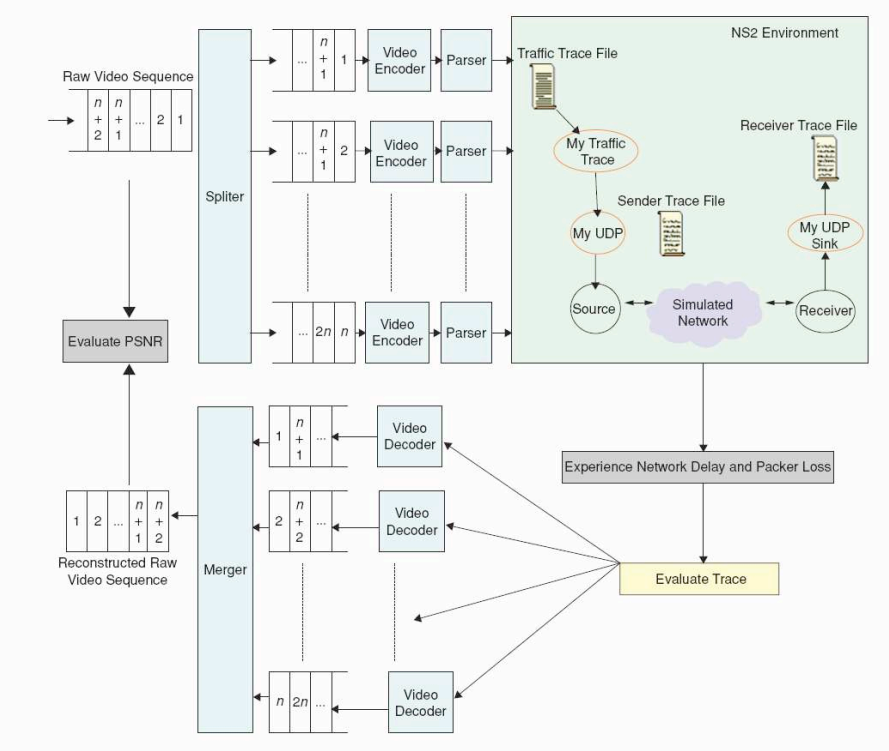

Figure D. 1 Scheme of a MDC system

Some calculation of quality metrics such as PSNR (object metric for quality measurement) can be gathered in order to evaluate the quality of the transmitted video resource.

Another interesting point is that this approach is codec-independent. The coding stage can be implemented using any codec.

ANNEX E. H.264 AVC overview

Possibly, the H.264 / MPEG-4 codec will be the dominant protocol for High Definition (HD) and Three Dimension (3D), which will be a requirement in future next-generation media transmission systems and applications, so it requires a special mention. It was written by the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG) as the product of a partnership effort known as the Joint Video Team (JVT). The ITU-T H.264 standard and the ISO/IEC MPEG-4 Part 10 standard (formally, ISO/IEC 14496-10) are jointly maintained so that they have identical technical content.

The intent of the H.264/AVC project was to create a standard capable of providing good video quality at substantially lower bit rates than previous standards, without increasing the complexity of design so much that it would be impractical or excessively expensive to implement. An additional goal was to provide enough flexibility to allow the standard to be applied to a wide variety of applications on a wide variety of networks and systems, including low and high bit rates, low and high resolution video, broadcast, DVD storage, RTP/IP packet networks, and ITU-T multimedia telephony systems.

The initial idea of H.264 was to join Mpeg-2 and H.263 to obtain a better compression rate and the possibility to define different codecs for different uses.

It is based in temporal and spatial redundant or negligible information to reduce the volume of data needed to send. It uses 2 modes of codification: interframe and intraframe, and it divide each image in blocks or group of pixels.

H.264 is based in MPEG-2. Next, there is a list of the principal improvements.

- ✓ The image is understood as a group of independent objects to improve the motion prediction in continues movements.
- ✓ The size of the MacroBlocks MB is variable and gives a better resolution
- ✓ Improve the precision of the motion vectors to 1/4 of pixel
- ✓ Allows multiple prediction using different reference frames
- ✓ Discrete Cosine Transform is used in blocks of 4x4 to obtain a more accurate transformation
- ✓ Introduce the intraframe prediction to identify homogeneous zones of the image and codify it in a efficient way

- ✓ In-loop deblocking filter for smoothing edges and avoiding the occurrence of visual artefacts.
- ✓ Unrestricted motion search allows displacements outside of a reference frame using spatial prediction.

Entropy coding takes in account context of data being encoded and provides two alternative methods: **Context-Adaptive Variable Length Coding (CAVLC)** that employs multiple variable length codeword tables to encode data, and **Context-Adaptive Binary Arithmetic Coding (CABAC)** that provides an improved encoding scheme with unmatched bit/symbol ratios.

Three profiles have been defined in order to cover the main application domains:

I. **Baseline Profile**, which is dedicated to conversational applications, such as video-telephony and video-conferencing;

II. **Main Profile** designed for television applications;

III. **Extended Profile**, more appropriate for streaming and mobile video services.

ANNEX F. Types of NAL units

Table F. 1 Original NAL unit types

Type	NAL Unit Type name
0	Unspecified
1	Coded slice
2	Coded slice data partition A
3	Coded slice data partition B
4	Coded slice data partition C
5	Coded slice of an IDR picture
6	Supplemental enhancement information (SEI)
7	Sequence parameter
8	Picture parameter
9	Picture delimiter
10	End of sequence
11	End of stream
12	Filler data

ANNEX G. RTP Payload for SVC

RTP Payload Format for SVC Video (October 2006) [22] describes an RTP Payload format for the scalable extension of the ITU-T Recommendation H.264 video codec which is the technically identical to ISO/IEC International Standard 14496-10 video codec. The RTP payload format allows for packetization of one or more Network Abstraction Layer Units (NALUs), produced by the video encoder, in each RTP payload. The payload format has wide applicability, as it supports applications from simple low bit-rate conversational usage, to Internet video streaming with interleaved transmission, to high bit-rate video-on-demand. The RTP payload specification is designed to be unaware of the bit string in the NAL unit payload. One of the main properties of H.264 is the complete decoupling of the transmission time, the decoding time, and the sampling or presentation time of slices and pictures. The decoding process specified in H.264 is unaware of time, and the H.264 syntax does not carry information such as the number of skipped frames. Also, there are NAL units that affect many pictures and that are, therefore, inherently timeless. For this reason, the handling of the RTP timestamp requires some special considerations for NAL units for which the sampling or presentation time is not defined or, at transmission time, unknown.

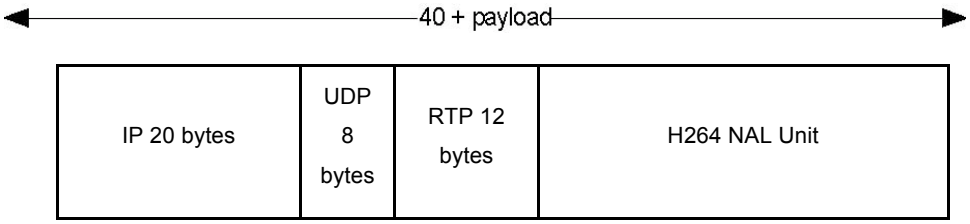

Figure G. 1 NAL Unit encapsulation into RTP

ANNEX H. Simulation commands

The followed process was:

- **Xvid_encraw**: Codes (m4v) and allows to define the GOP (max_key_interval)

  Example: ./xvid_encraw.exe –i foreman0.yuv –w 176 –h 144 –framerate 7.5 –max_key_interval 7.5 –o foreman0.m4v

- **MP4Box**: Codes to mp4 and allows to define how are going to packetize the frames (1024 Bytes each frame)

  Example: ./MP4Box.exe –hint –mtu 1024 –fps 7.5 –add foreman0.m4v oreman0.mp4

- **Mp4trace**: Generates the trace files of the simulation.

  Example: ./mp4trace.exe –f –s 192.168.48.208 123456 foreman0.mp4 > st0

The files used for the simulation are the following ones:

- **Trace files:** Shows how it is packetized each frame and if it is an I or a P frame. It is generated from Mp4trace.
- **Sender file:** It is generated by ns-2 and indicates when a packet is sent.
- **Receiver file**: It is generated by ns-2 once the simulation is finished and shows the packets received to the receiver (rd).

ANNEX I. Simulation Results with MPEG-4 Codec

The results of this simulation, shown in the following tables, are the PSNR values (measured in dB) of the different tests. They are separated according to the type of video, MDC technique and number of descriptors. Each test has been repeated according to three different probabilities of loss (1%, 5% and 10%).

- **Temporal MDC**

Foreman video

2 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	36,4019	30,3653	40,2495	34,2456	36,1309	38,6552	32,2265	35,6315	33,2403	30,5528	35,8789
5%	21,0979	25,9348	24,3729	24,7067	26,6888	24,0416	25,4003	20,7508	25,9695	24,2350	24,6783
10%	19,6820	21,4775	21,3925	21,7276	20,3005	21,1702	19,3008	22,0092	21,3523	19,0806	20,8622

4 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	33,5833	34,5544	33,6202	35,5950	31,9002	37,9340	33,8039	33,4451	32,7559	40,1496	35,5237
5%	24,0108	23,6369	22,5401	25,3780	23,2532	26,0975	25,4847	23,6684	26,2106	24,9201	24,6844
10%	20,6478	22,8768	20,4980	21,1618	19,7293	19,0020	20,4258	21,8851	19,9835	19,2567	20,6997

8 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	36,5490	34,8748	35,1234	36,2018	33,3384	32,6736	37,2146	33,8494	34,8370	35,3067	35,2052
5%	26,1789	25,4058	22,9489	26,8366	27,1014	27,0047	24,8988	23,9766	27,4487	25,9010	25,9773
10%	21,6476	20,8557	21,6972	23,5763	21,1844	23,3018	20,5650	22,1292	22,8278	21,9286	22,0781

12 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	34,8355	34,6958	36,1533	35,9173	33,6433	34,7746	33,8253	29,3225	36,2642	40,0473	35,6759
5%	22,9025	23,1417	23,5856	25,3511	26,8790	26,1994	21,6192	26,4220	27,0460	24,5064	25,1180
10%	20,4046	18,7504	19,6582	19,5549	21,6735	20,7915	22,5427	21,2783	23,1584	21,7747	21,1584

16 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	33,1835	34,8819	34,7830	34,8259	35,4383	32,7048	36,0105	35,0425	37,4385	35,6839	35,1848
5%	25,5139	24,3273	25,1107	25,1066	26,9682	25,1031	25,7279	25,7149	24,9991	25,4144	25,4499
10%	23,3534	23,2615	20,4722	20,5523	24,2055	19,9186	22,0960	22,3464	25,3908	20,9178	22,5962

Akiyo video

2 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	56,2715	52,9118	49,0039	53,3193	58,0907	43,4262	54,3865	56,9361	49,1954	58,5400	54,9774
5%	43,3993	40,7578	34,0994	33,9814	34,3792	45,4070	34,1724	43,7833	41,7170	46,8491	42,2034
10%	38,5136	32,9585	39,7310	31,7021	34,6430	36,7617	37,6728	39,6328	39,4145	40,2191	37,9177

4 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	56,9654	59,4708	56,8701	57,4923	53,8880	63,4602	59,1506	57,2206	50,3750	58,7601	58,5148
5%	45,3522	42,8118	42,6829	43,7333	47,3324	45,8933	46,2345	43,7905	41,5876	40,5953	44,4771
10%	39,8492	38,7111	41,4051	39,1383	41,8046	38,0673	37,5434	39,9031	36,5016	40,2991	39,6053

8 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	53,3289	55,4639	46,7287	55,7779	54,6284	54,2964	50,9121	54,2250	53,0659	55,2677	53,9419
5%	45,7180	45,6713	45,3286	45,6103	46,5669	46,6400	45,3576	40,8139	43,8664	45,6673	45,3636
10%	45,1979	42,7027	38,8237	40,6844	39,3355	38,2745	39,6412	40,3358	38,7950	35,5322	40,7014

12 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	56,5432	53,4919	53,3850	53,7410	57,4270	56,8080	52,8019	54,3366	46,2501	54,0935	54,6578
5%	46,6506	44,4646	43,5863	43,4511	43,6306	46,8064	45,0839	43,6004	45,9796	42,4383	44,8070
10%	39,9634	38,0509	39,9802	42,6141	37,8280	40,4562	41,8430	40,4516	41,8430	41,7700	40,7329

16 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	49,8783	55,4254	53,5837	60,6216	51,5527	54,1819	52,8905	52,1132	54,0895	56,1366	55,1289
5%	42,6338	42,3061	45,8848	44,4873	44,0647	44,5184	43,3869	46,2891	44,0617	45,3815	44,4775
10%	40,4556	38,8698	40,4222	40,2784	41,2667	39,9886	42,6289	41,2025	40,5813	40,5973	40,7289

- **Spatial MDC**

Foreman video

2 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	38,8928	42,0460	39,6284	38,9398	40,5252	36,6964	32,1618	39,2155	34,0547	37,6190	38,7538
5%	28,0232	29,7606	30,4556	26,8494	29,4714	31,7834	27,5806	29,5259	27,8030	27,6473	29,1502
10%	25,5320	15,9796	19,8774	26,0705	25,3950	15,7835	26,1616	25,3575	25,8019	22,5175	24,1474

4 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	43,2312	42,0649	41,3672	42,0871	41,8738	41,9774	40,2474	42,8978	40,8275	40,5104	41,8065
5%	33,1715	33,4585	32,4085	34,6020	33,3670	33,9634	33,1794	31,9517	32,6210	32,6229	33,1986
10%	30,1870	28,2995	29,6641	30,7281	29,4124	28,3589	28,9656	29,0948	29,5597	28,1228	29,3143

8 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	43,4060	44,8638	43,6148	42,1193	42,4155	44,1882	43,1327	43,5160	42,5475	42,9181	43,3466
5%	35,9236	35,8721	34,4465	35,3888	35,9681	35,8314	35,9791	34,4902	35,6349	35,8172	35,5693
10%	31,8575	31,2660	30,9490	30,7879	31,3152	31,7179	32,3859	31,3744	31,5365	31,0406	31,4470

12 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	41,1785	40,6578	40,9680	40,7406	41,2334	41,5985	40,6623	41,5148	41,4639	41,2294	41,1376
5%	36,1128	35,7457	36,5599	36,7289	36,4999	36,1149	35,8744	36,6884	37,0269	35,7872	36,3345
10%	32,6298	32,6154	33,0415	32,5077	32,5916	32,2573	32,7631	33,1017	32,6427	32,6814	32,6895

16 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	42,9259	42,2327	42,8417	42,5657	41,7230	42,1317	41,8358	42,1330	42,2856	41,8215	42,2679
5%	37,1185	37,0064	37,3046	36,9538	36,8976	36,9157	37,2846	36,9423	37,1955	36,6283	37,0291
10%	33,7062	33,5772	33,5891	33,3219	33,5014	33,6872	33,2127	33,8442	33,8872	33,9019	33,6284

Akiyo video

2 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	41,9234	43,2039	45,8487	42,9845	45,2469	46,5216	43,9896	45,7305	45,2371	45,4734	44,8332
5%	38,9620	39,3584	37,3944	38,6478	38,5081	37,7207	38,6414	38,6564	38,7205	39,8731	38,7003
10%	36,1520	36,7075	35,6943	22,0899	36,0571	36,5591	36,1569	36,0701	19,1731	37,0223	35,3827

4 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	43,8372	43,7505	44,5212	45,3745	45,5429	44,9300	46,3241	46,1688	47,0873	45,2578	45,3987
5%	39,1927	37,8315	37,8536	38,1902	38,5883	38,5345	37,9637	38,1847	38,4541	37,7654	38,2769
10%	34,8208	34,4540	35,1576	35,1079	35,3722	34,5330	35,0088	35,7949	34,7999	35,0877	35,0300

8 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	45,7485	45,7127	45,2137	46,8606	46,4933	52,2851	48,5519	47,5431	44,8183	46,3114	47,5737
5%	39,0085	38,3170	38,8816	39,2724	38,9562	39,2403	38,5397	38,5715	38,9352	38,7669	38,8585
10%	35,2007	35,6960	35,5667	34,3894	35,6015	34,6676	35,2129	35,4229	35,6495	34,8941	35,2506

12 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	44,1887	43,7607	43,5331	44,2847	44,0393	44,5494	44,1069	44,2890	43,9917	44,4826	44,1326
5%	38,2674	38,5968	38,5728	38,6862	38,5196	38,8146	38,9743	38,8197	38,8086	38,3874	38,6497
10%	34,9677	35,0214	35,1762	35,3467	34,4061	35,2149	34,6724	35,1000	35,6260	34,8537	35,0507

16 Descriptors:

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Average
1%	44,1223	45,6499	44,6012	44,1788	44,1346	43,9953	44,5383	44,3701	44,6331	45,1141	44,5621
5%	39,1577	39,0622	38,8091	38,7946	38,9997	39,1187	39,2626	38,8117	38,9055	38,5114	38,9484
10%	34,9125	35,2701	35,6356	35,0777	34,6535	35,5293	35,7498	35,8272	35,3560	34,4264	35,2665

ANNEX J. Simulation Results without Codec

The following tables show the results according to the type of video, MDC technique and number of descriptors. In this case is represented directly the average of the 3 tests. Also, for the temporal and spatial MDC are represented some graphs comparing PSNR values with different number of descriptors (2, 4, 8, 16 and 24) and different probabilities of loss (1%, 5%, 10% and 15%).

- **Temporal MDC**

Foreman video

Num Desc / PLR	2	4	8	16	24
1%	53,1825	55,7379	54,3462	53,2435	62,9869
5%	38,9454	42,1472	44,7922	41,8251	44,2268
10%	37,3892	35,0238	39,1781	39,7585	38,1327
15%	34,7806	34,6841	35,7895	36,8999	36,4842

Akiyo video

Num Desc / PLR	2	4	8	16	24
1%	75,6144	69,1665	72,9820	67,2071	72,0940
5%	58,6953	60,0103	59,1108	58,9434	61,5650
10%	55,7788	53,8827	56,5422	53,1013	55,8797
15%	53,7695	53,2927	53,0134	52,3893	53,5446

Carphone video

Num Desc / PLR	2	4	8	16	24
1%	55,9945	55,1134	58,0836	62,0175	52,0460
5%	44,3768	46,5026	46,4753	47,4730	46,0059
10%	39,8242	42,9277	39,3538	42,6855	43,4677
15%	39,0957	40,9114	39,6191	40,7634	39,2164

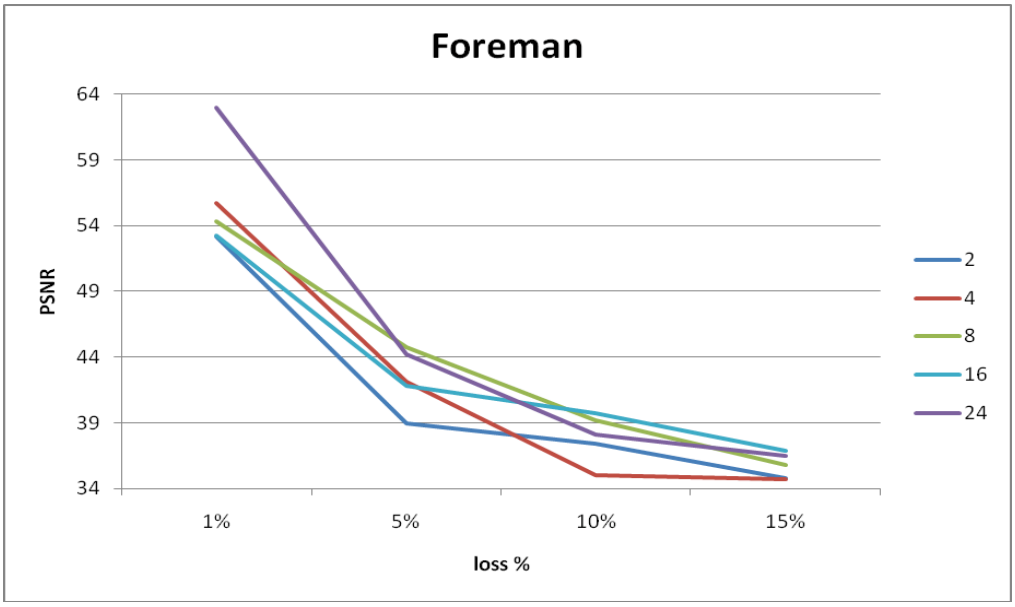

Figure J. 1 PSNR vs. loss, Foreman, temporal MDC

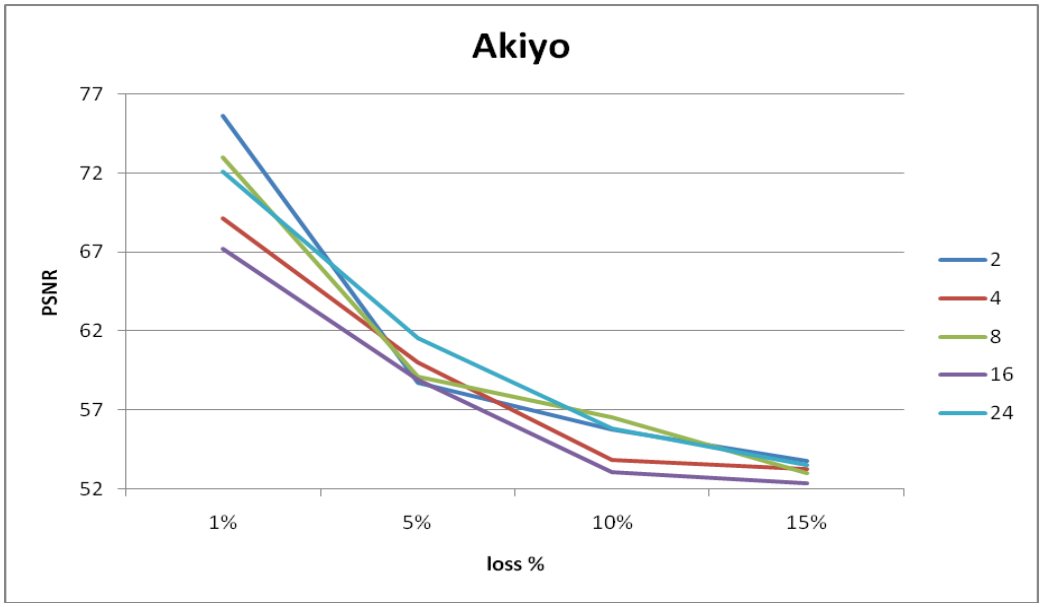

Figure J. 2 PSNR vs. loss, Akiyo, temporal MDC

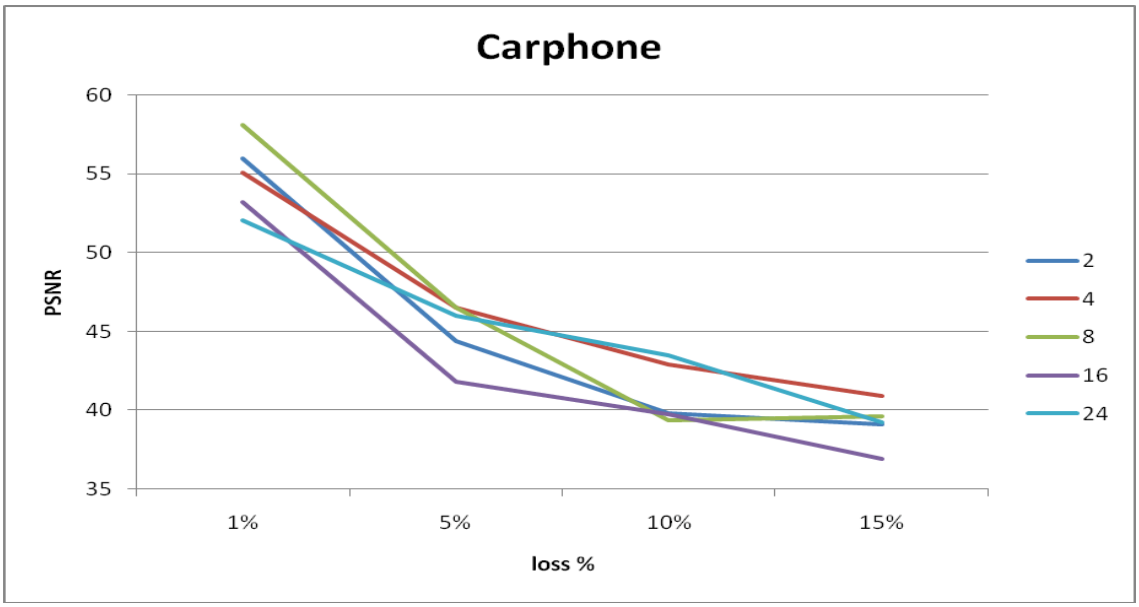

Figure J. 3 PSNR vs. loss, Carphone, temporal MDC

- **Spatial MDC**

Foreman video

Num Desc / PLR	2	4	8	16	24
1%	52,1609	52,7798	53,0346	53,3222	52,7641
5%	44,8401	45,1277	45,1370	45,6578	45,2858
10%	41,9584	42,0063	42,1342	42,2725	42,2428
15%	39,8040	40,2863	40,3136	40,3075	40,2793

Akiyo video

Num Desc / PLR	2	4	8	16	24
1%	52,5813	52,9924	52,7867	52,3104	52,9247
5%	45,7664	46,0431	46,4891	46,4519	46,8768
10%	44,1658	43,3085	43,4660	43,8126	43,8260
15%	42,4078	41,3734	41,7197	41,6313	41,6924

Carphone video

Num Desc / PLR	2	4	8	16	24
1%	52,3515	53,0801	52,8575	52,9014	53,2515
5%	45,0295	45,8597	45,8466	45,8157	45,7618
10%	40,6619	42,4478	42,7180	42,6812	42,6611
15%	40,2322	40,7272	40,5286	40,7484	40,7040

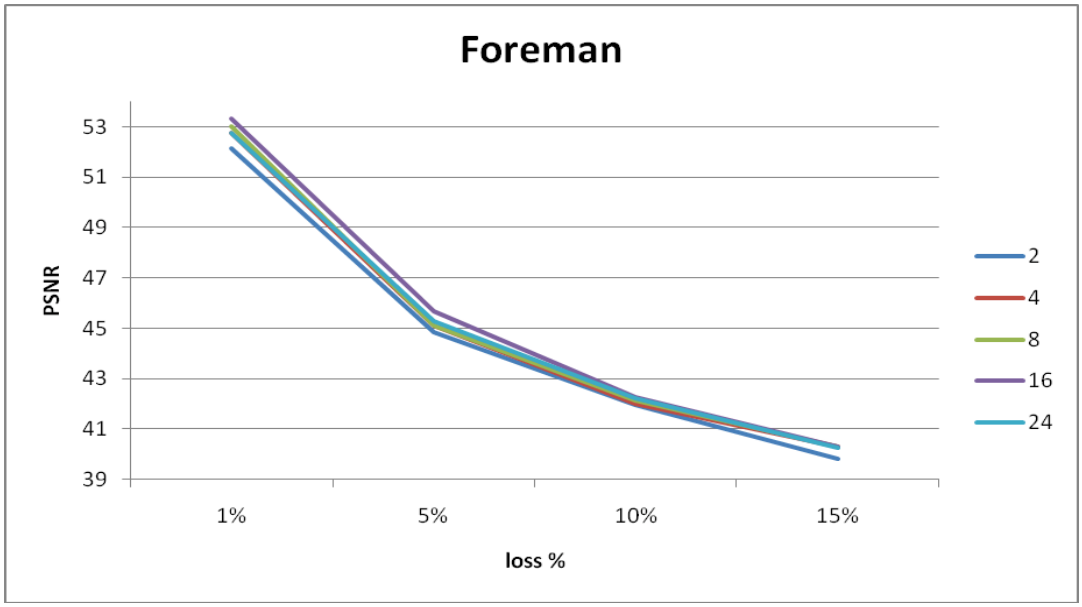

Figure J. 4 PSNR vs. loss, Foreman, spatial MDC

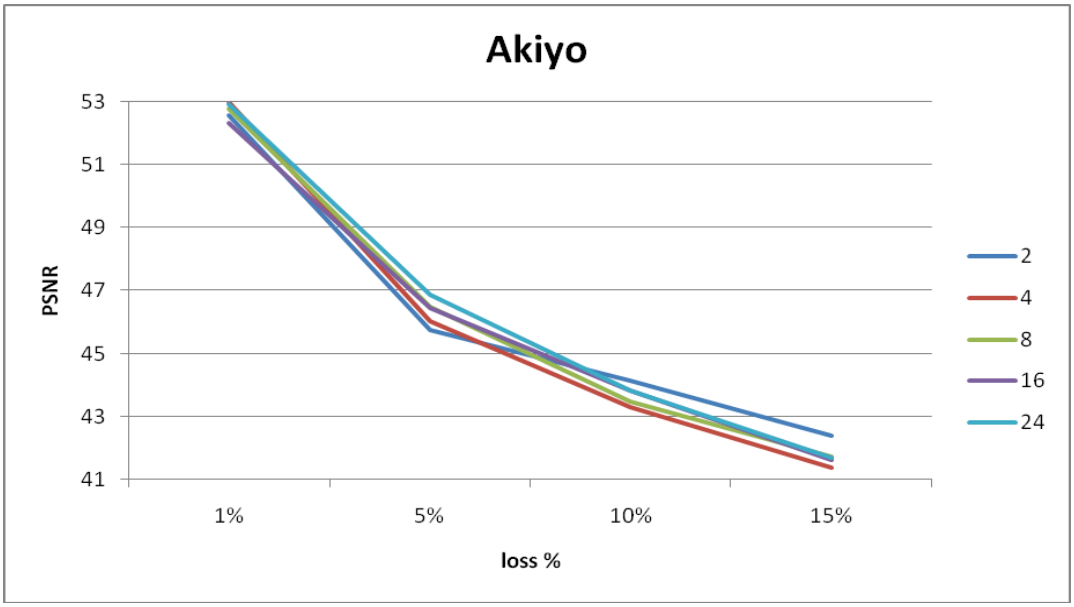

Figure J. 5 PSNR vs. loss, Akiyo, spatial MDC

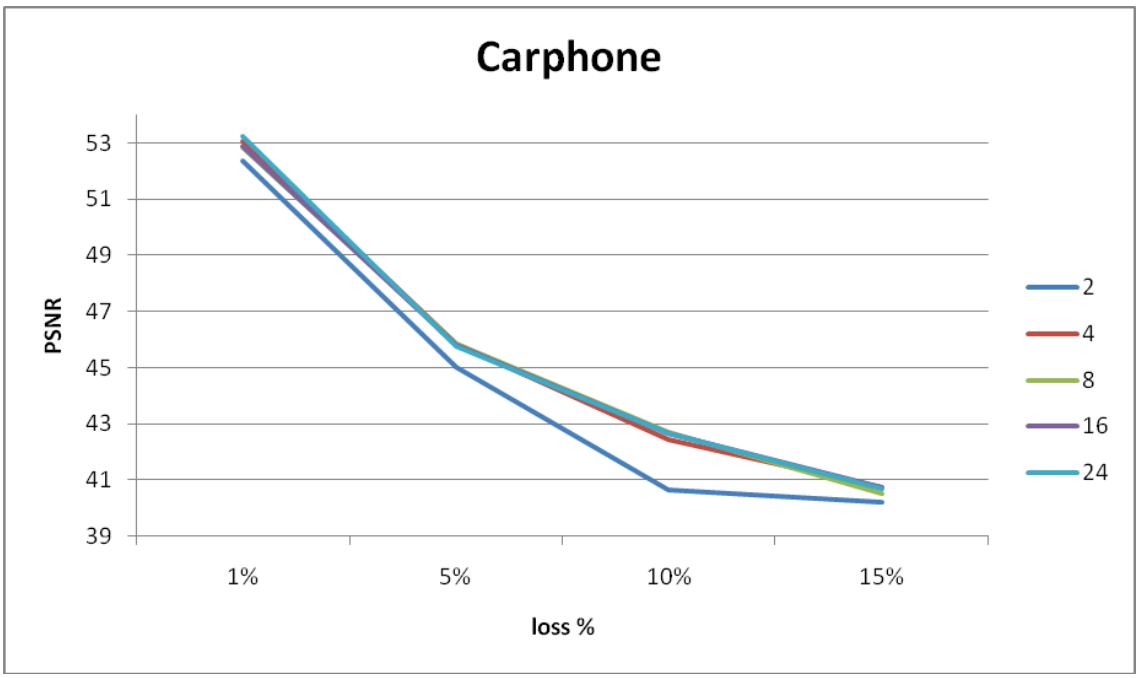

Figure J. 6 PSNR vs. loss, Carphone, spatial MDC

- **Hybrid MDC**

Foreman video

Num Desc / PLR	2	4	8	16	24
1%	52,3445	Inf	Inf	Inf	Inf
5%	43,1908	59,2583	Inf	Inf	Inf
10%	41,4453	50,2893	Inf	Inf	Inf
15%	34,1657	50,2947	Inf	Inf	Inf

Akiyo video

Num Desc / PLR	2	4	8	16	24
1%	Inf	Inf	Inf	Inf	Inf
5%	50,2165	58,3404	Inf	Inf	Inf
10%	51,4133	55,4758	Inf	Inf	Inf
15%	47,8987	50,3708	Inf	Inf	Inf

Carphone video

Num Desc / PLR	2	4	8	16	24
1%	53,4162	Inf	Inf	Inf	Inf
5%	48,2322	Inf	Inf	Inf	Inf
10%	42,4982	53,8214	Inf	Inf	Inf
15%	41,8089	48,7896	Inf	Inf	Inf

Finally, the following graphs represent the comparison of PSNR between temporal and spatial techniques with the same number of descriptors:

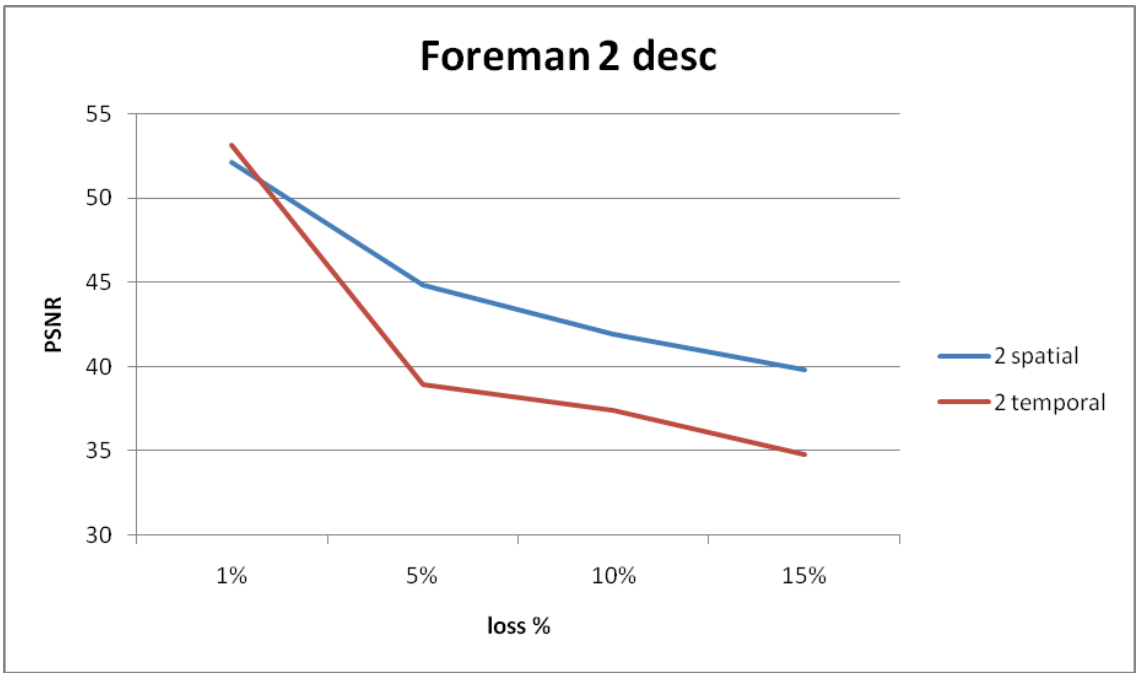

Figure J. 7 Spatial vs. Temporal, 2 descriptors, Foreman

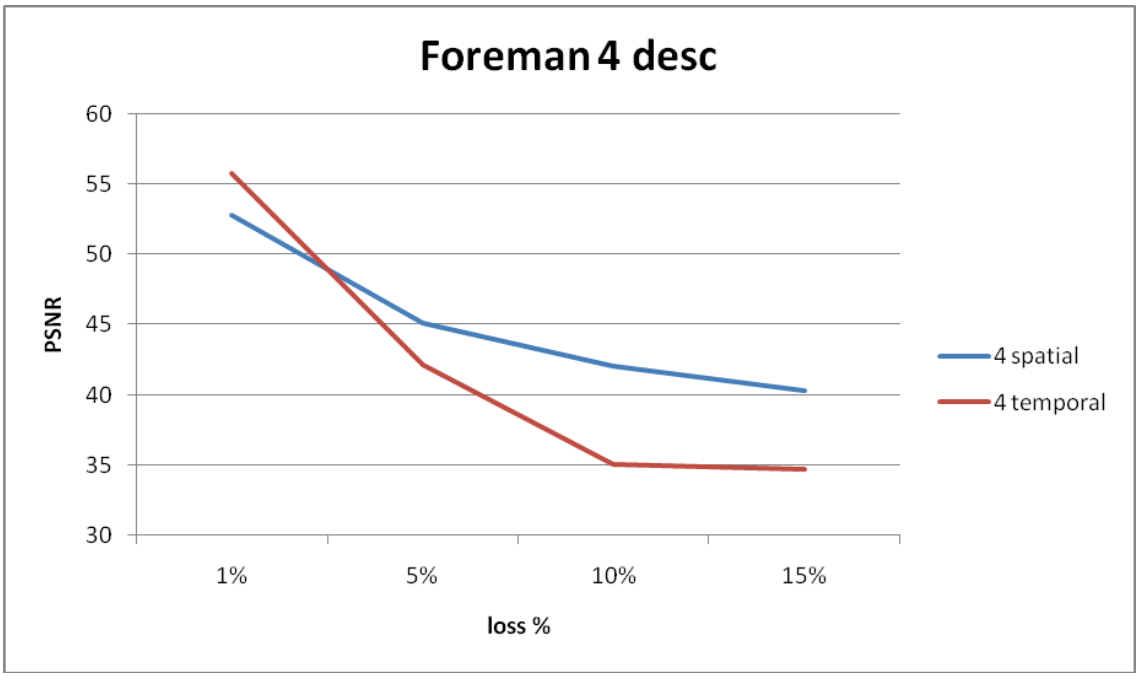

Figure J. 8 Spatial vs. Temporal, 4 descriptors, Foreman

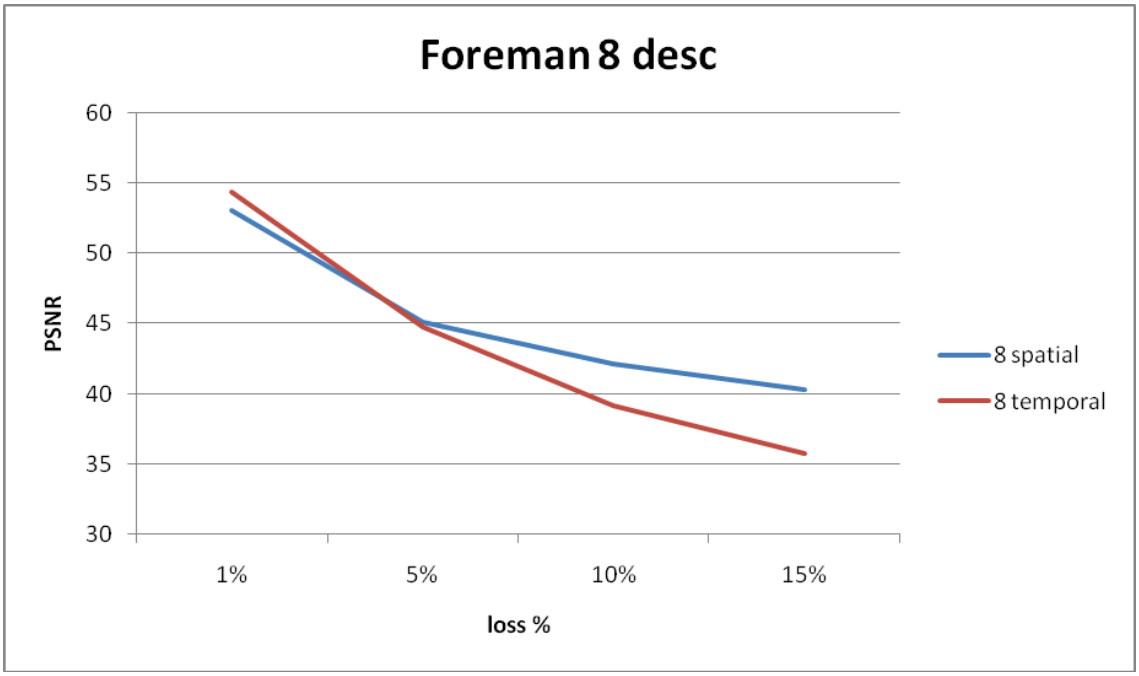

Figure J. 9 Spatial vs. Temporal, 8 descriptors, Foreman

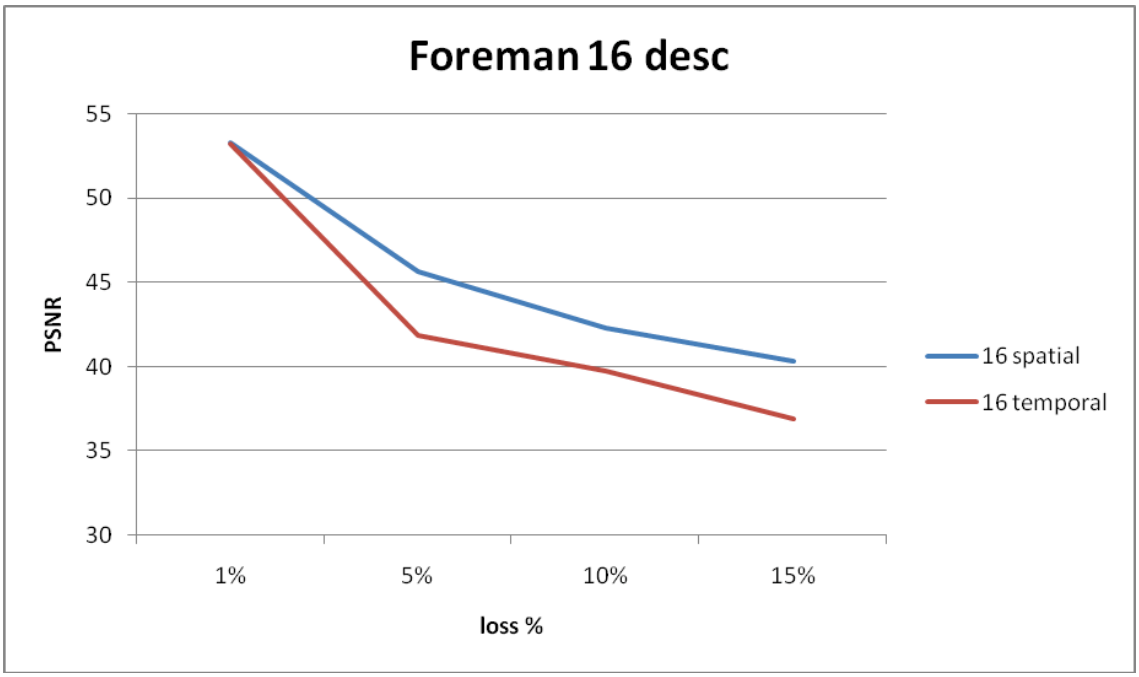

Figure J. 10 Spatial vs. Temporal, 16 descriptors, Foreman

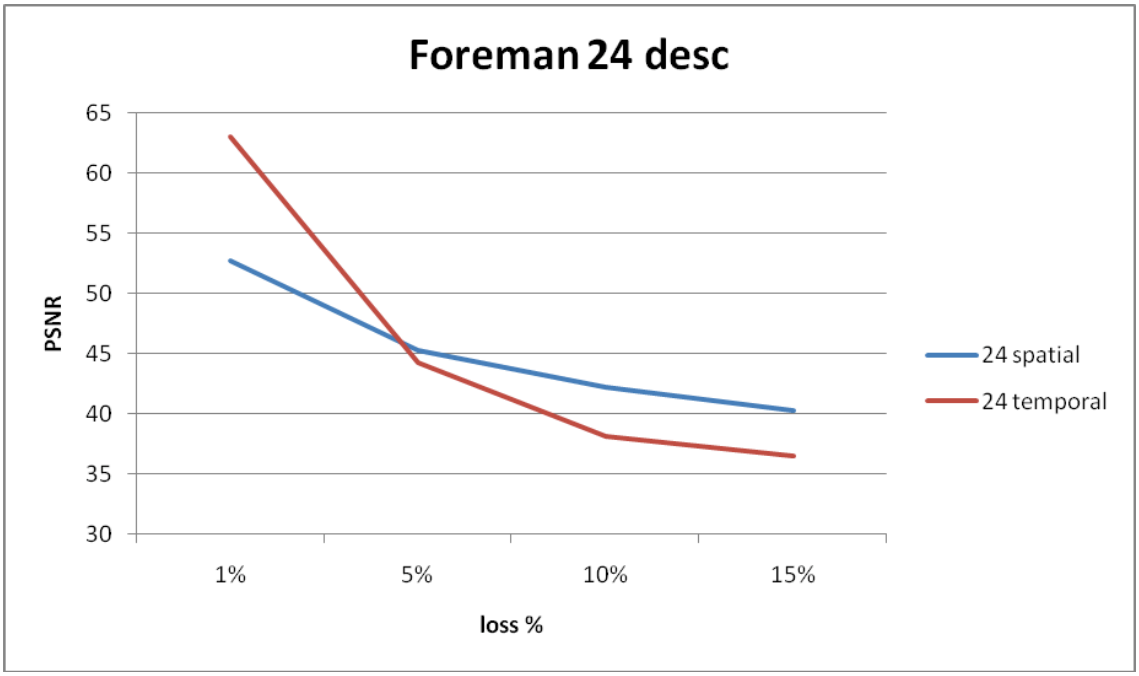

Figure J. 11 Spatial vs. Temporal, 24 descriptors, Foreman

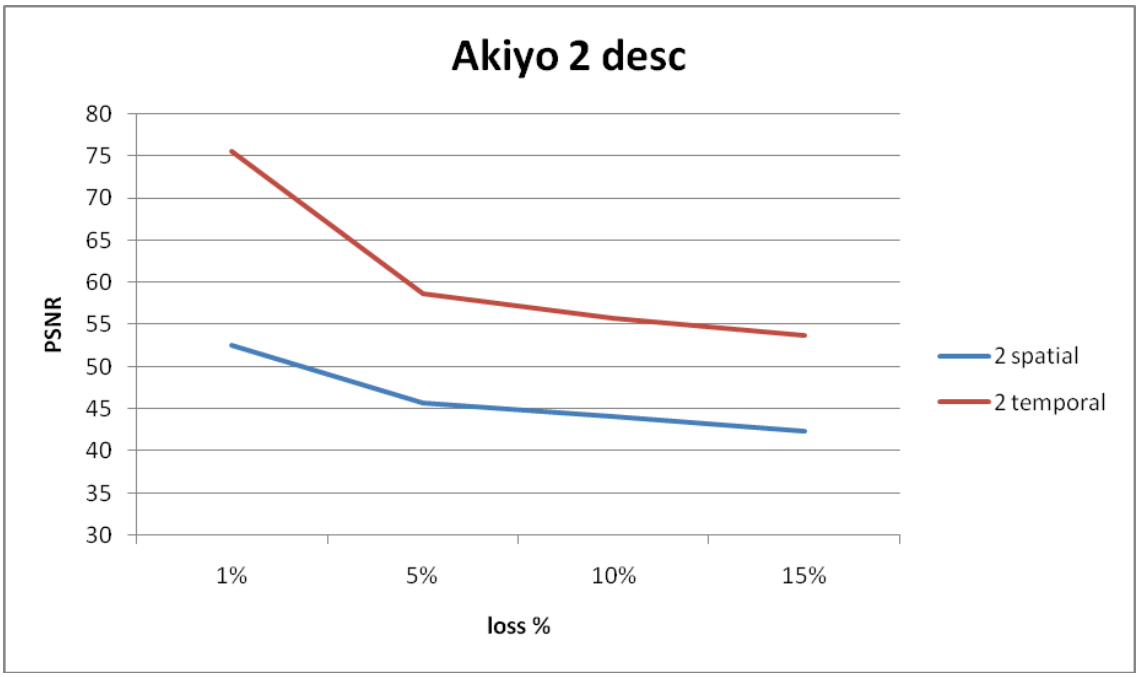

Figure J. 12 Spatial vs. Temporal, 2 descriptors, Akiyo

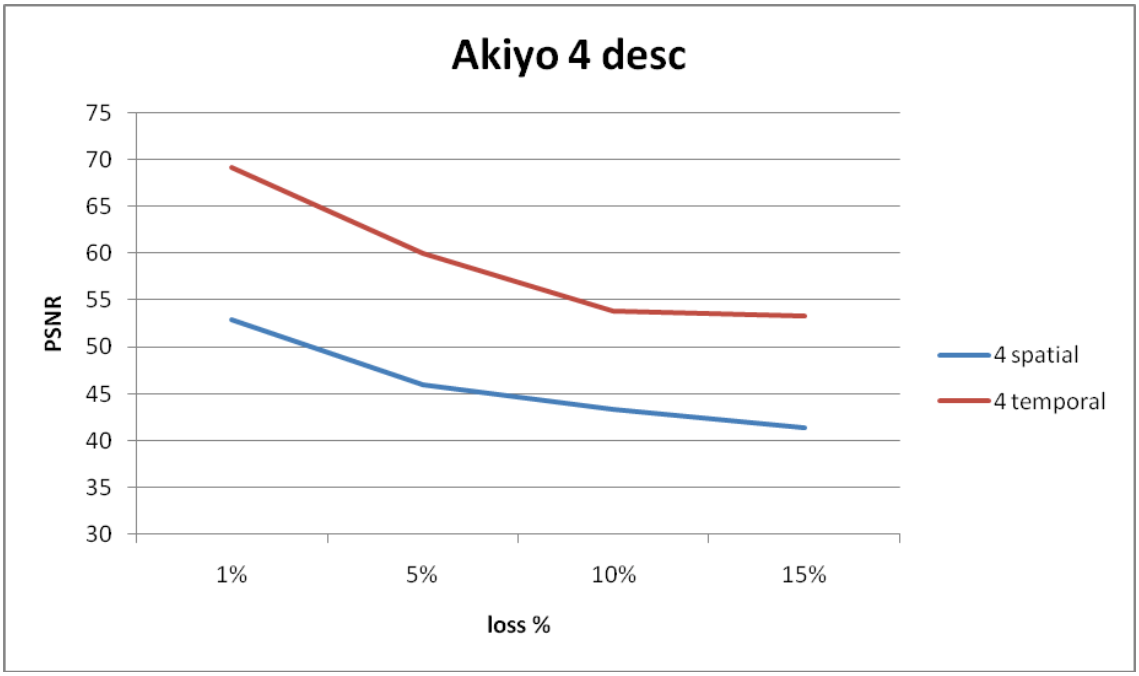

Figure J. 13 Spatial vs. Temporal, 4 descriptors, Akiyo

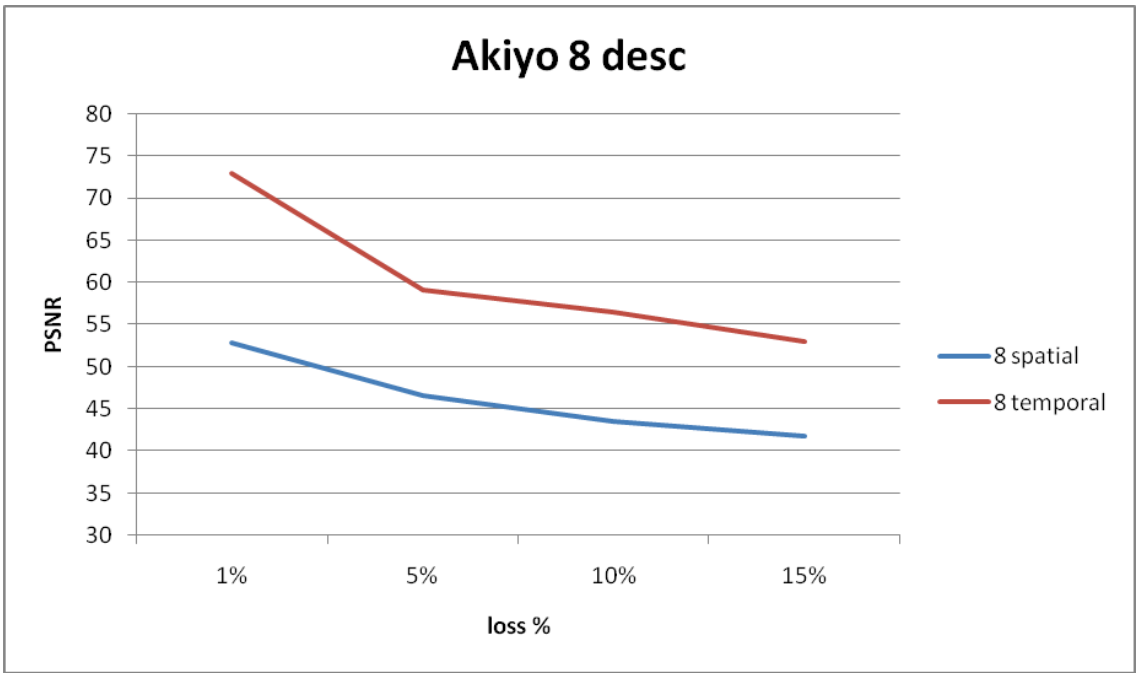

Figure J. 14 Spatial vs. Temporal, 8 descriptors, Akiyo

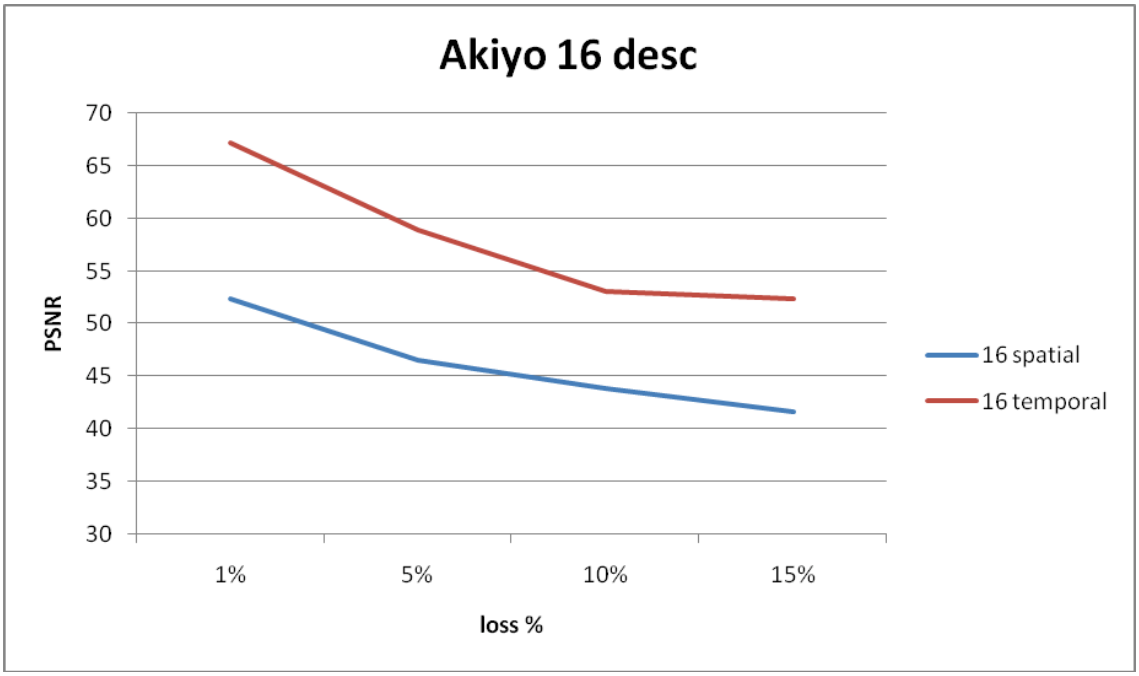

Figure J. 15 Spatial vs. Temporal, 16 descriptors, Akiyo

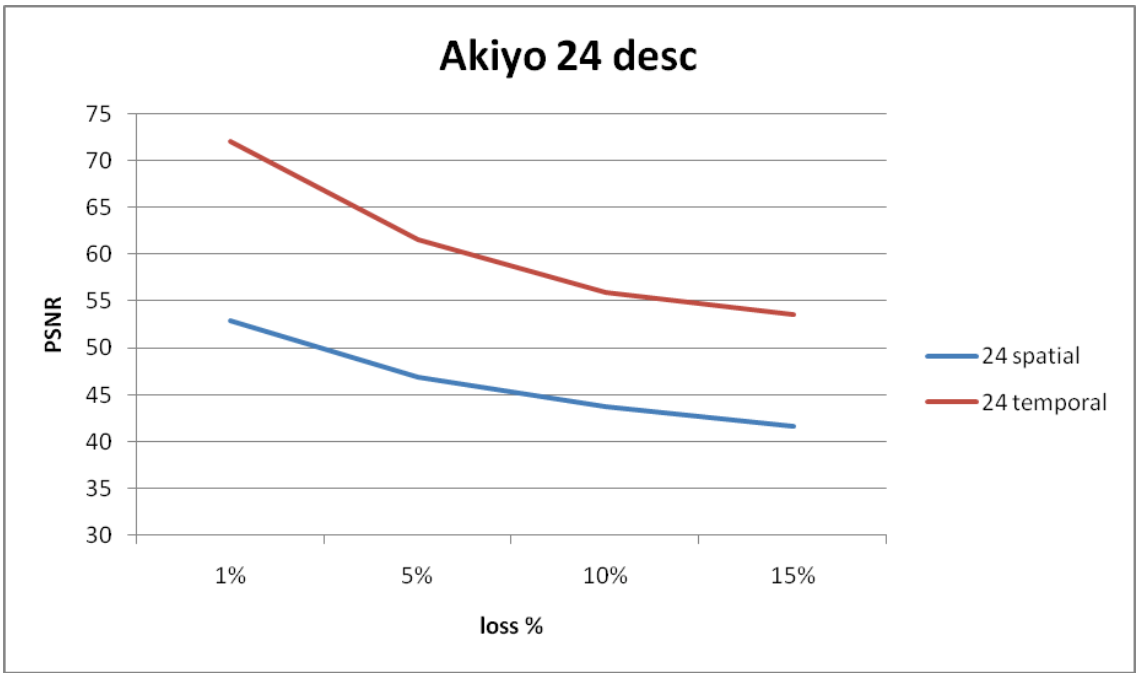

Figure J. 16 Spatial vs. Temporal, 24 descriptors, Akiyo

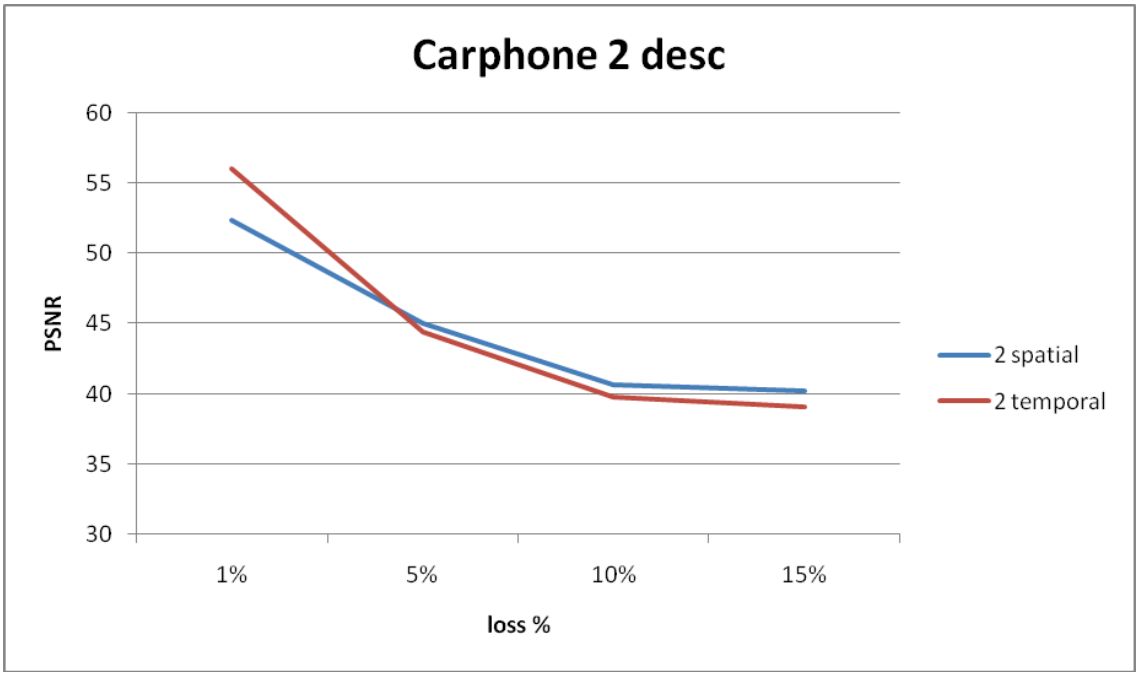

Figure J. 17 Spatial vs. Temporal, 2 descriptors, Carphone

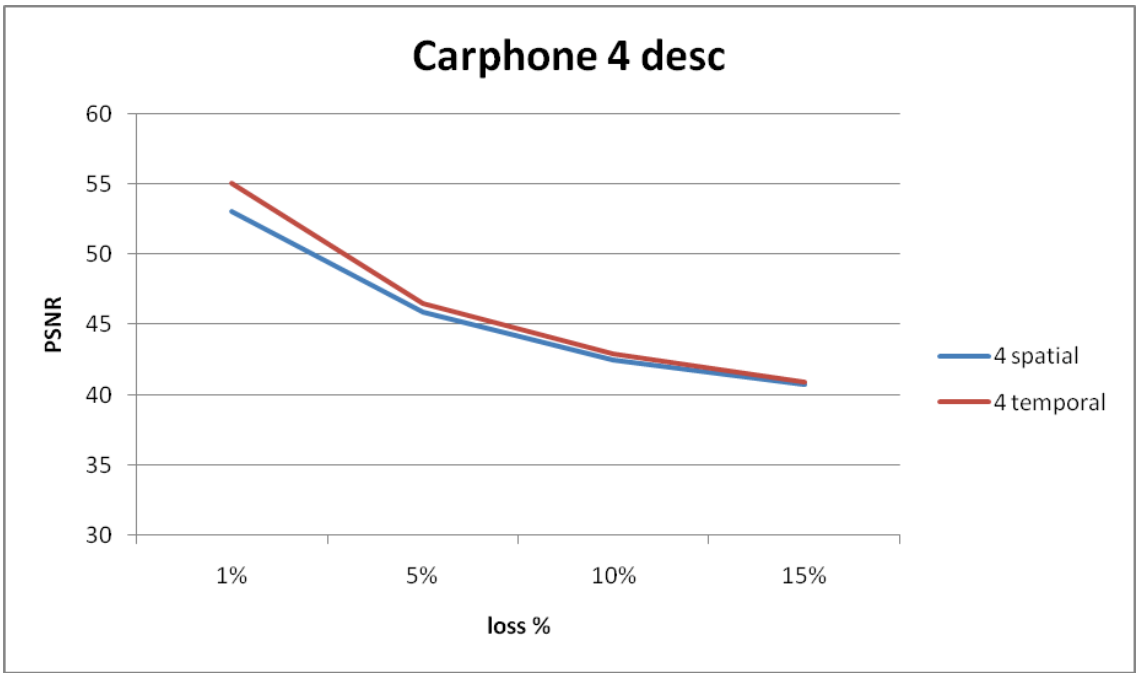

Figure J. 18 Spatial vs. Temporal, 4 descriptors, Carphone

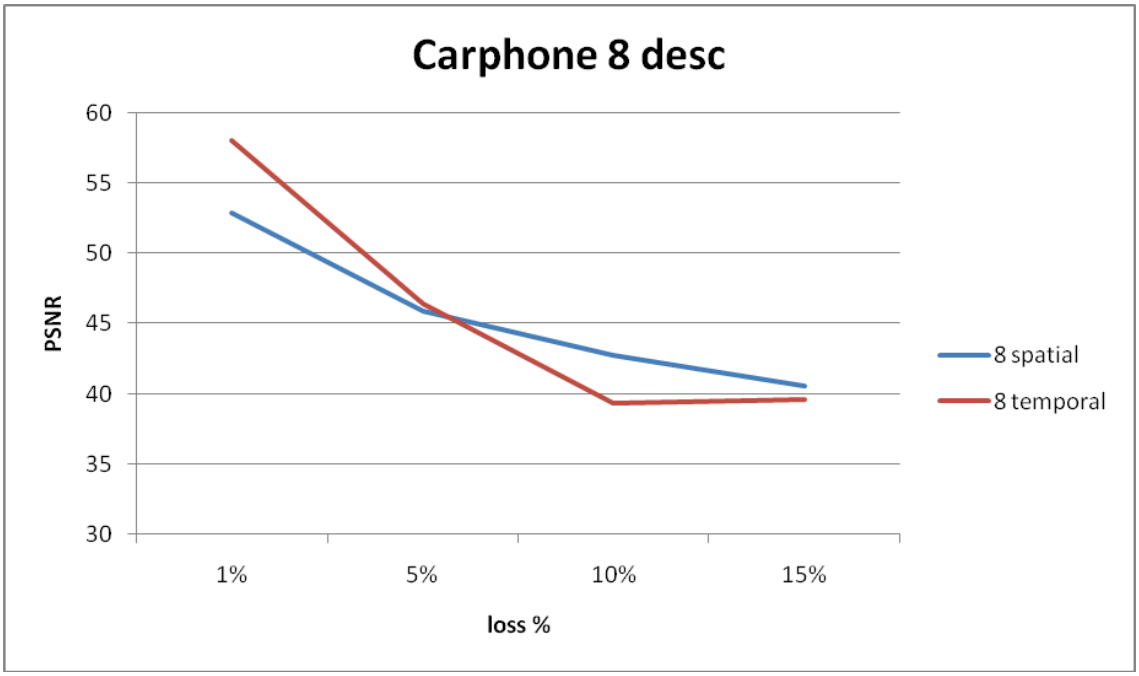

Figure J. 19 Spatial vs. Temporal, 8 descriptors, Carphone

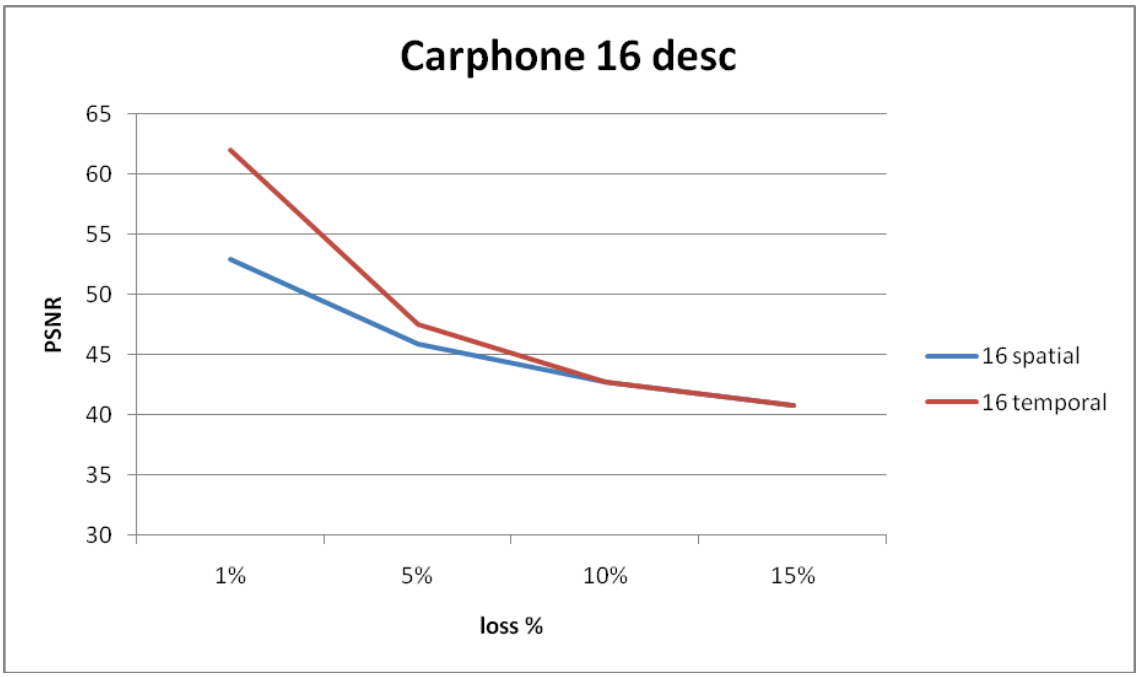

Figure J. 20 Spatial vs. Temporal, 16 descriptors, Carphone

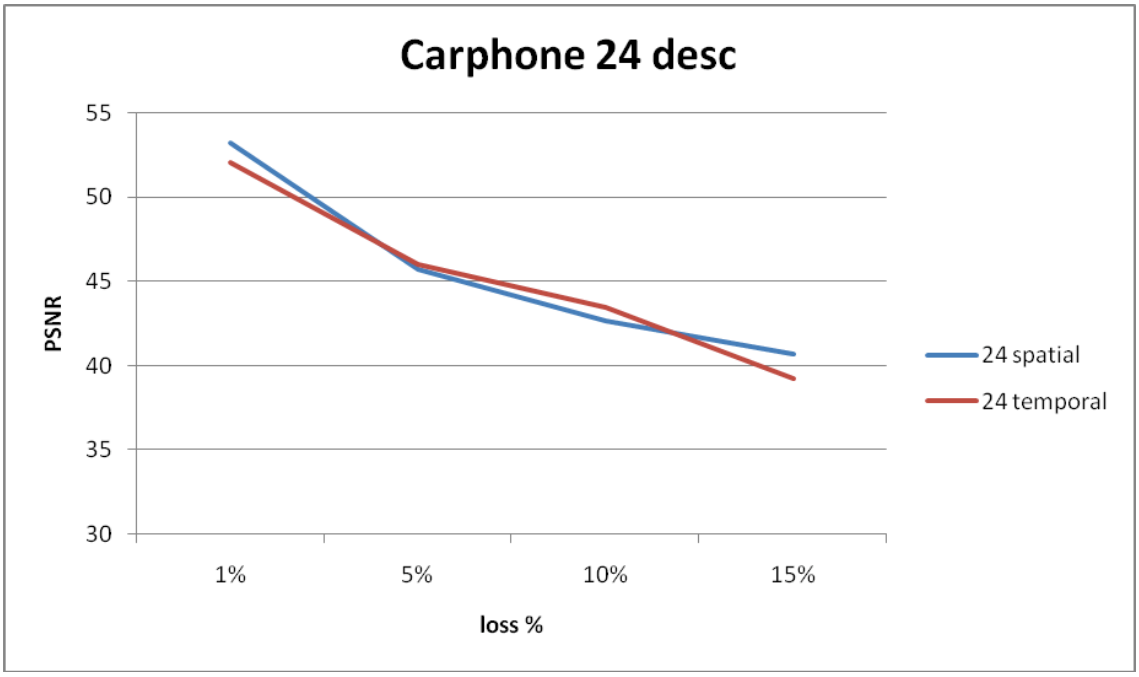

Figure J. 21 Spatial vs. Temporal, 24 descriptors, Carphone

ANNEX K. P2P Results

BT= Buffer Time [seconds]

SBR= Stream Bit Rate [chunks / second]

PS= Partnership Size [number of partners]

Note: 1 chunk supposes 1/SBR seconds

Table K. 1 Single Description

Factors			**Mean Results**			
BT	*SBR*	*PS*	*Initial Start-Up Delay*	*Control Overhead*	*Continuity Index*	*Missed Chunk Rate*
10	8	3	13,93 s	0,82 %	0,8431	0,1569
10	8	8	11,41 s	1,40 %	0,9952	0,0048
10	16	3	13,48 s	0,90 %	0,9040	0,0960
10	16	8	12,04 s	1,98 %	0,9950	0,0050
20	8	3	24,51 s	0,65 %	0,8878	0,1122
20	8	8	29,03 s	1,60 %	0,9984	0,0016
20	16	3	25,86 s	1,13 %	0,8519	0,1481
20	16	8	26,45 s	2,34 %	0,9299	0,0701

Table K. 2 Multiple Description Coding

Factors			**Mean Results**			
BT	*SBR*	*PS*	*Initial Start-Up Delay*	*Control Overhead*	*Continuity Index*	*Missed Chunk Rate*
10	8	3	13,17 s	0,68 %	0,8699	0,1702
10	8	8	13,72 s	1,55 %	0,9967	0,0035
10	16	3	13,77 s	0,61 %	0,8387	0,2365
10	16	8	10,77 s	1,28 %	0,9886	0,0122
20	8	3	19,26 s	0,74 %	0,8354	0,1655
20	8	8	30,97 s	2,48 %	0,8884	0,1127
20	16	3	23,00 s	0,88 %	0,8920	0,1102
20	16	8	24,78 s	1,97 %	0,8549	0,1478

ANNEX L. Used Software

- **FFMPEG [http://ffmpeg.org/]**

FFmpeg is a computer program that can record, convert and stream digital audio and video in numerous formats. FFmpeg is a command line tool that is composed of a collection of free software / open source libraries. It includes libavcodec, an audio/video codec library used by several other projects, and libavformat, an audio/video container mux and demux library.

- **VideoLan [http://www.videolan.org]**

Is a project that develops software for playing video and other media formats. It originally developed two programs for media streaming—VideoLAN Client (VLC) and VideoLAN Server (VLS)—but most of the features of VLS have been incorporated into VLC, with the result renamed VLC media player.

- **Boost library [http://www.boost.org/]**

The Boost C++ Libraries are a collection of open source libraries that extend the functionality of C++. We focused in the use of the thread libraries, present in the concurrent programming category

- **Ns-2 [http://www.isi.edu/nsnam/ns/]**

Ns is a discrete event simulator targeted at networking research. Ns provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks.

- **Matlab [http://www.mathworks.com/]**

Matlab is a tool for doing numerical computations with matrices and vectors. It can also display information graphically.

- **Video processing blockset**

Extends Matlab with a rich customizable framework for the rapid design, simulation, implementation, and verification of video and image processing algorithms and systems.

- **Microsoft Visual Studio C++ [http://msdn.microsoft.com/en-us/visualc/default.aspx]**

Is a commercial integrated development environment (IDE) product engineered by Microsoft for the C, C++, programming languages. It has tools for developing and debugging C++ code.

- **Netbeans 6.5 [http://www.netbeans.org/]**

Platform for the development of applications for the network (using Java, JavaScript, PHP, Python, Ruby, Groovy, C, and C++), and an integrated development environment (IDE) developed using the NetBeans Platform.

We used the Java Netbeans IDE, version 6.5.

- **Axe 3 [http://www.axe-editor.com/]**

Advanced Hexadecimal Editor

ANNEX M. Generated Publications

The publications generated during the realization of this work are listed next.

Book Chapter: A. Rios, A. J. González, and J. Alcober. “Peer-to-Peer Multimedia Conferencing System based on SIP Signaling”.

- Book Title: Traffic and Performance Engineering for Heterogeneous Networks
- Three Research Volumes by River Publishers, Feb 2009. ISBN 978-87-92329-16-5.

Article: “A Peer-to-Peer Live Streaming Platform for High-Quality Media Services”, André Ríos (MediaEntel-UPC), Francesc Enrich (TAM-URL), Xavier Milà (NeTS-UPF), José Fco. Crespo (LCFIB-UPC), Alberto J. González (MediaEntel-UPC), Albert Vidal (i2CAT).

- NEM Summit “Towards Future Media Internet”, 13-15 Oct 2008, Saint-Malo, France. ISBN 978-3-00-025978-4, October 2008.

Article: “P2P Multipoint Conference System using SIP”, André Ríos (MediaEntel-UPC), Alberto J. González (MediaEntel-UPC), Antoni Oller (MediaEntel-UPC), Juan López (MediaEntel-UPC), and Jesús Alcober (MediaEntel-UPC).

- HET-NETs 08 “The Fifth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks”, 18-20 Feb 2008, Blekinge Institute of Technology (BTH), Karlskrona, Sweden.

Article: “Prototype of P2P Multiconferencing System based on SIP”, A. Ríos, A. J. González, and J. Alcober.

- Telecom I+D 2008, Bilbao, Spain. ISBN 978-84-9860-135-0, Oct 2008.

Article: “Streaming P2P robusto en redes Ad-hoc utilizando información social”,

Alberto José González, Daniel Rodríguez, Javier López, Francesc Rillo, Jesus Alcober

- Jitel 2009, Cartagena, Spain. Presented in Sep 2009 (poster session)

Extended Abstract: “Robust and scalable P2P streaming for Future Media Internet”, Alberto J. González, André Ríos, Jesús Alcober

- Future Internet Assembly (FIA) Valencia 2010. Submitted in Nov 2009.