Ministry of Higher Education and Scientific Research
وزارة التعليم العالي والبحث العلمي

Badji Mokhtar Annaba University
Université Badji Mokhtar – Annaba
Faculty of Technology

Department of Electronics

جامعة باجي مختار – عنابـــة
كـليـة التكنـولـوجيـا
قســم الإلكترونيك

Thesis

Submitted to obtain the diploma of

Doctorate Third Cycle

Field: Telecommunication

Specialty: Multimedia and Digital Communications

By:
MANSRI Islem

Title:

Amélioration de la transmission à distance de la vidéo compressée: Application à un système de vidéosurveillance à faible latence

Thesis defended on July 2, 2023 in front of the jury composed of:

N°	Name and Surname	Grade	Establishment	Quality
01	**BENOUARET Mohamed**	Prof.	Badji Mokhtar Annaba University	President
02	**DOGHMANE Noureddine**	Prof.	Badji Mokhtar Annaba University	Supervisor
03	**KOUADRIA Nasreddine**	MCA	Badji Mokhtar Annaba University	Co-Supervisor
04	**BOUMEHREZ Farouk**	MCA	Khenchela University	Examiner
05	**BOUKAACHE Abdenour**	MCA	Guelma University	Examiner
06	**HARIZE Saliha**	MCA	Badji Mokhtar Annaba University	Invited

This thesis is dedicated to my parents, ***M. Nasser*** and ***B. Sakina***,
to my aunt ***M. Hakima***, and
to my sister ***M. Anfel***.

"The noblest pleasure is the joy of understanding"
- Leonardo da Vinci -

Acknowledgments

I am deeply grateful to Almighty Allah for providing me with the blessings and strength to successfully complete my research.

I would like to express my sincere appreciation and gratitude to the members of the jury for their valuable time and expertise in evaluating our thesis. In particular, the president of the jury Prof. BENOUARET Mohamed and all the jury members, Dr. BOUMEHREZ Farouk, Dr. BOUKAACHE Abdenour, and Dr. HARIZE Saliha.

I also would like to express my sincere gratitude to my supervisor, Prof. DOGHMANE Noureddine and Co-supervisor Dr. KOUADRIA Nasreddine for their invaluable guidance and support throughout the research. Their expertise and patience have been instrumental in the success of this project.

I am grateful to my colleagues and peers at the LASA Laboratory | University of Badji Mokhtar Annaba (UBMA) for their support and helpful discussions. In particular, I would like to thank ALIOUAT Ahcen, NEILI Zakaria, KEBIR Abdeldjalil, MEKAHLIA Mohammed Saddek and HADOUNE Oussama.

I would like to acknowledge the support of my family, who have been a constant source of encouragement and motivation throughout this journey.

This work would not have been possible without the help and support of all these individuals, and I am deeply grateful to them.

"تحسين جودة الإرسال اللاسلكي عن بعد للفيديو المضغوط : تطبيق على نظام مراقبة الفيديو متأخر الوقت"

الملخص:

مكنت تطورات الاتصالات اللاسلكية وشبكات الهاتف المحمول الفيديو من احتلال أكبر جزء من حركة المرور التي تعبر الإنترنت. في الواقع، إن توصيل المعلومات في الوقت الفعلي من خلال وسيط لاسلكي يخضع لمشاكل فقدان البيانات، خاصة عند إرسال معلومات ذات ارتباط عالي مثل الفيديو. يرجع ذلك أيضا إلى الخصائص المتغيرة بمرور الوقت للقناة اللاسلكية. بناء على ذلك، يجب استخدام تقنيات مقاومة الأخطاء لحماية الفيديو المرسل عبر الشبكات اللاسلكية. من خلال استغلال الميزات المتقدمة الجديدة لمعايير ترميز الفيديو، توفر هذه الأطروحة حماية على مستوى المصدر لمعلومات الفيديو المرسلة عبر الشبكات المعرضة للخطأ. وتستند المساهمات الرئيسية لأطروحتنا على معيار HEVC لأنه يوفر قدرات ضغط متفوقة مقارنة مع قرائنه.

كمرحلة أولى، ولتطوير خوارزميات المرونة للخطأ، توفر هذه الأطروحة تحليلا عميقا لميزات ترميز HEVC عن طريق دراسات مقارنة في بيئات خالية من الأخطاء و بيئات معرضة للخطأ. ثانيا، تقترح الأطروحة خوارزمية تتجنب إستعمال الصور الخاطئة المشفرة مسبقا. باستخدام هذا، يقوم المشفر بتغيير صوره المرجعية بناء على معلومات حالة الخطأ الواردة من وحدة فك التشفير عبر تحديثات قناة الإرسال المتخلفة. في الواقع، جنبا إلى جنب مع النهج السابق، عملنا يستخدم بذكاء ناقلات الحركة للحد من ضعف خوارزمية اختيار الصورة المرجعية. علاوة على ذلك ، فإنه يقتطع انتشار الخطأ وبالتالي يعزز جودة الفيديو النهائي المصور. بالنظر إلى ما سبق ، تهدف هذه الأطروحة إلى ضمان أفضل مساومة بين الموثوقية، سرعة الارسال وكفاءة الضغط .

في أطروحتنا، تم القيام بأعمال تجريبية واسعة النطاق لتقييم أداء أساليبنا المقترحة. تتضمن الاختبارات بيئات خالية من الأخطاء و و بيئات معرضة للخطأ مع تكوينات شبكة مختلفة. في الواقع، تظهر النتائج التي تم الحصول عليها أن آلياتنا تعمل على تحسين أداء مرونة الخطأ مقارنة بالخوارزميات الأخرى. في الواقع، يمكن رؤية التحسينات من خلال التقييم الغير موضوعي وكذلك التقييم الموضوعي باستخدام YUV-PSNR.

كلمات مفتاحية: المراقبة بالفيديو، ضغط الفيديو، HEVC/H.265، AV1, VVC، H.264/AVC، جودة الخدمة، المرونة ضد الخطأ، نقل الفيديو، تأخير منخفض، انتشار الخطأ، إخفاء الخطأ.

« Amélioration de la transmission à distance de la vidéo compressée : Application à un système de vidéosurveillance à faible latence »

Résumé :

Les progrès de la communication sans fil et des réseaux mobiles permettent à la vidéo d'occuper la plus grande partie du trafic traversant l'internet. Cependant, la communication d'informations en temps réel par le biais d'un support sans fil est sujette à des problèmes de perte de données, surtout lorsqu'il s'agit de transmettre des informations hautement liées comme la vidéo. De plus, cela est dû aux caractéristiques variables dans le temps du canal sans fil. À cette fin, des techniques de résilience aux erreurs doivent être employées pour protéger la vidéo transmise par les réseaux sans fil. En exploitant les nouvelles fonctionnalités avancées des normes de codage vidéo, cette thèse fournit une protection au niveau de la source de l'information vidéo transmise à travers des réseaux sujets aux erreurs pour les applications de vidéosurveillance à faible latence. Les principales contributions de notre thèse sont basées sur le standard HEVC car il offre des capacités de compression supérieures à celles de ses prédécesseurs.

Dans un premier temps, et afin de développer des algorithmes résistants aux erreurs, cette thèse fournit une analyse approfondie des caractéristiques de codage HEVC au moyen d'études comparatives dans des environnements sans erreur et avec erreur. Dans un deuxième temps, la thèse propose un algorithme qui évite le référencement à partir d'images erronées précédemment encodées. Ainsi, l'encodeur change ses images de référence en fonction des informations sur l'état des erreurs reçues du décodeur via des mises à jour du canal de rétroaction. En effet, combiné à la première approche, notre travail fait une utilisation intelligente des vecteurs mobiles pour résoudre la limitation de l'algorithme de sélection des images de référence. En outre, il tronque la propagation des erreurs et améliore ainsi la qualité perçue de la vidéo finale. Compte tenu de ce qui précède, cette thèse vise à assurer le meilleur compromis entre la fiabilité, l'interactivité et l'efficacité de la compression.

Dans notre thèse, des travaux expérimentaux approfondis ont été menés pour évaluer les performances des méthodes que nous proposons. Les tests incluent des environnements sans erreur et avec erreur avec différentes configurations de réseau. En fait, les résultats obtenus montrent que nos mécanismes améliorent les performances de résilience aux erreurs par rapport aux autres algorithmes. Cependant, les améliorations peuvent être constatées par une évaluation subjective ainsi que par une évaluation objective en termes de YUV-PSNR.

Mots clés : Vidéo surveillance, compression vidéo, HEVC/H.265, VVC, AV1, H.264/AVC, QoS, résilience aux erreurs, transmission vidéo, faible latence, propagation des erreurs, dissimulation des erreurs.

« Improving the wireless transmission of compressed video: Application to a low-latency video surveillance system»

Abstract:

The advances in wireless communication and mobile networks enable video to occupy the largest portion of the traffic crossing the internet. However, communicating real-time information through a wireless medium is subject to data loss issues, especially when delivering highly related information such as video. Moreover, it is due to the time-varying characteristics of the wireless channel. To this end, error resilience techniques must be employed to protect the transmitted video through wireless networks. By exploiting new advanced features of video coding standards, this thesis provides source-level protection of the video information transmitted through error-prone networks addressing low-delay video surveillance applications. The main contributions of our thesis are based on the HEVC standard as it offers superior compression capabilities compared to its predecessors.

As a first phase, and to develop error-resilient algorithms, this thesis provides a deep analysis of the HEVC encoding features by means of comparative studies in error-free and erroneous environments. Second, the thesis proposes an algorithm that avoids referencing from previously encoded erroneous pictures. With this, the encoder changes its reference pictures based on the error status information received from the decoder via backward feedback channel updates. Indeed, combined with the former approach, our work makes intelligent use of moving vectors to tackle the limitation of the reference picture selection algorithm. Moreover, it truncates the error propagation and thus enhances the perceived end-video quality. Considering the above, this thesis aims to ensure the best trade-off between reliability, interactivity, and compression efficiency.

In our thesis, extensive experimental works have been conducted to assess the performance of our proposed methods. The tests include error-free and erroneous environments with different network configurations. In fact, the obtained results show that our mechanisms improve the error resilience performance compared to other algorithms. However, the enhancements can be seen through subjective evaluation as well as objective evaluation in terms of YUV-PSNR.

Key words: Video Surveillance, Video Compression, HEVC/H.265, VVC, AV1, H.264/AVC, QoS, error-resilience, video transmission, low-delay, error-propagation, error concealment.

List of Figures

1 Trade-off for video delivery. 3

1.1 HEVC video encoding diagram 10
1.2 High-level overview of HEVC system layers [43]. 11
1.3 HEVC bit-stream structure diagram. 12
1.4 (a) CTU Z-scan partitioning from 64×64 to CUs of size 8×8. (b) Its coding tree structure representation [113]. 13
1.5 Utilization of Tiles with Slices (a) One slice containing three tiles. (b) One tile containing two slices [114]. 14
1.6 HEVC Intra Prediction angular modes. [116]. 15
1.7 Motion vector predictor candidates in HEVC [103]. 16
1.8 Taxonomy of error resilience methods 21
1.9 Video Communication systems block diagram 21
1.10 MDC videos transmission through an erroneous multipath channel [17]. 23

2.1 The SI/TI plan for class E and Class E' test sequences. 35
2.2 Spatial and temporal information indices distribution for class C videos. 37
2.3 First pass rate-distortion curves for all codecs regarding class E and E' videos. 40
2.4 Second pass rate-distortion curves for all codecs regarding class E and E' videos. . . 44
2.5 Rate distortion curves for all codecs regarding class C video test sequences. 47

3.1 General video evaluation framework 53
3.2 Eval-Vid, ns-3 based framework for video quality evaluation. 55
3.3 FFMPEG based framework for HEVC video quality evaluation using RTP/UDP/IP stack. 57

3.4 Rate-Distortion curves for HM, x265-M and x265-U regarding class C video sequences: (left upper): BasketBall-Drill, (left lower) PartyScene, (right upper): BQMall, (right lower) RaceHorses 59

3.5 Error propagation plots for BasketballDrill 1% loss 61

3.6 Error propagation plots for BasketballDrill 10% loss 62

3.7 Diagram of the conducted work for HM and FFMPEG error concealment. 64

3.8 Snapshot of frame 4 in the event of Whole Fame Loss/Header Loss decoded with FFMPEG all EC. 65

3.9 Snapshot of frame 4 in the event of Whole Fame Loss decoded with HM frame copy. 65

3.10 Snapshot of frame 3 in the event of slice portion loss decoded with FFMPEG all EC. 66

3.11 Snapshot of frame 4 in the event of slice Header Loss/ whole Slice loss decoded with FFMPEG all EC. 67

3.12 Snapshot of frame 3 in the event of slice portion content loss decoded with FFMPEG all EC. 67

3.13 Snapshot of frame 3 in the event of slice Header Loss/ whole Slice loss decoded with FFMPEG all EC. 68

3.14 Snapshot of frame 3 in the event of slice portion content loss decoded with FFMPEG all EC. 68

3.15 Diagram for the conducted experiments with and without TMVP. 69

3.16 PSNR map difference between the reference frame encoded using TMVP and the reference frame encoded without TMVP for BasketballDrill. 70

3.17 Error propagation curves when frame 4 is lost. 71

4.1 The overall diagram of our proposed approach. 75

4.2 Decreased temporal dependency strategies. (a) TMVP for all frames (b) No-TMVP for all frames (light green) (c) NTR4 frames at the beginning of each GOP (Pink). . . 78

4.3 Spatial and temporal information plot of the test sequences. 79

4.4 PSNR R-D curves in error free environment (a) BasketballDrill; (b) BQmall. 83

4.5 SSIM R-D curves in error free environment (a) BasketballDrill; (b) BQmall. 84

4.6 PSNR R-D curves when frame 20 is dropped (a) BasketballDrill; (b) BQmall. 84

4.7 SSIM R-D curves when frame 20 is dropped (a) BasketballDrill; (b) BQmall.. 85

4.8 Frame by frame visual quality evaluation, frame 27 snapshot of all the test sequences: (a) default, (b) No-TMVP, (c) NTR4 88

A.1 The spatial and temporal information indices of the test sequences 103

List of Tables

1.1 All different methods included in HEVC and VVC [126]. 19

2.1 Selected encoding settings for all software implementations for case one 33
2.2 Test Video Sequences for Interactive applications 35
2.3 Test Video Sequences for mobile applications 37
2.4 Selected encoding settings for all software implementations for case two 38
2.5 Summary of the both similarities and differences for both works 39
2.6 Average bitrate savings for HM performance for first pass mode 41
2.7 Average bitrate savings for AV1 encoder. 42
2.8 Average bitrate savings for VTM encoder. 42
2.9 Summarized bitrate saving results for the first-pass experiments 43
2.10 Average bitrate savings for the performance of HM for second pass mode. 43
2.11 Average BD-BR savings for the compared performance of AV1 encoder. 45
2.12 Average bitrate savings for the performance of the VTM encoder. 45
2.13 Summarized bitrate saving results for the second pass experiments 46
2.14 Average bitrate savings of the HM encoder (U refer to Ultra in X264 AND X265) . . 48
2.15 Average BD-BR savings for the AV1 encoder. 48
2.16 Summarized bitrate saving for all codecs regarding mobile applications. 48
2.17 Encoding and Decoding time ratios for low delay configuration setup. 49

3.1 Coding performance comparison of HM, x265-M (Medium) and x265-U (Ultra) . . . 60
3.2 Error resilience performance of HM, x265-M and x265-U for 1% loss. 60

3.3 Error resilience performance of HM, x265-M and x265-U for 10% loss. 61

3.4 The overall error resilience gain of HM, x265-M and x265-U in lossy environments. . 62

3.5 HM and FFMPEG decoder characteristics 63

4.1 Selected Test Video Sequences. 80

4.2 PSNR Results for the Default algorithm with different intra-refresh rates. 82

4.3 Average PSNR and SSIM gains for compared performance of proposed strategies. . 86

4.4 PSNR and SSIM Results for all the algorithms when frame 20 is dropped 87

A.1 All the test video sequences. 104

List of Abbreviations

4G Fourth Generation.

ACK positive Acknowledgment.
AI All-Intra.
AOM Alliance for Open Media.
ARQ Automatic Repeat reQuest.
AV1 AOMedia Video 1.
AVC Advanced Video Coding.

BD-PSNR Bjontegaard Delta Peak Signal-to-Noise Ratio.
BD-rate Bjontegaard Delta Rate.

CIF Common Inter-mediate Format.
CTU Coding Tree Unit.
CU Coding Unit.

DPB Decoded Picture Buffer.
DPCM Differential Pulse Code Modulation.
DVD Digital Versatile Disc.

E-UTRAN Evolved Universal Terrestrial Radio Access.
EC Error Concealment.

FEC Forward Error Correction.
FMO Flexible Macroblock Ordering.

GOP Groupe of Picture.

HD High-Definition.
HEVC High Efficiency Video Coding.
HM HEVC Test Model.
HVS Human Visual System.

IP Internet Protocol.
ISDN Integrated Services Digital Network.
ISO/IEC International Organization for Standardization/International Electrotechnical Commission.
ITU-T International Telecommunication Union-Telecommunication Standardization Sector.

JCT-VC Joint Collaborative Team on Video Coding.
JEM Joint Test Model.
JSCC Joint Source-Channel coding.
JSCD Joint Source-Channel decoding.
JVET Joint Video Experts Team.

Kbps Kilobit per second.

LC Layer Coding.
LD-P Low-Delay P.
LDPC Low-Density Parity-Check.
LTE Long-Term Evolution.
LTE-Sim Long-Term Evolution simulator.

MDC Multiple-Description Coding.
MPEG Moving Picture Experts Group.

MSE Mean Squared Error.
MV Moving Vector.

NACK Negative Acknowledgment.
NAL Network Abstraction Layer.
ns-3 Network Simulator 3.
NTR4 No-TMVP Refresh.

POC Picture Order Count.
PPS Picture Parameter Set.
PSNR Peak Signal to Noise Ratio.
PUs Prediction Units.

QoS Quality of service.
QP Quantization Parameter.

R-D Rate-Distortion.
RA Random-Acces.
RDO Rate-Distortion Optimization.
ROI Region of Interest.
ROPE Recursive Optimal per-Pixel Estimation.
RPS Reference Picture Selection.
RTP Real Time Protocol.

SDP Session Description Protocol.
SI/TI Spatial Information / Temporal Information.

SPS Sequence Parameter Set.
SSIM Structural Similarity Index Measure.

TCP Transmission Control Protocol.
TD Trip Delay.
TMVP Temporal Moving Vector Predictor.

U-HD Ultra-High-Definition.
UDP User Datagram Protocol.

VoD Video On Demand.
VPS Video Parameter Set.
VTM Versatile video coding Test Model.
VVC Versatile Video Coding.

Contents

List of Figures vi

List of Tables ix

List of Abbreviations xi

Contents xiii

Introduction 1

1 Theoretical background of video compression and error resilience 6

1.1 Introduction 6

1.2 Overview of old video coding standards 7

1.3 High Efficiency Video Coding (HEVC) 8

1.3.1 Hybrid Video Compression using HEVC 9

1.3.2 Packaging of Compressed Video Data 10

1.3.3 HEVC Block Partitioning 11

1.3.4 Intra-Picture Prediction in HEVC 15

1.3.5 Inter-Picture Prediction in HEVC 16

1.3.5.1 Motion Vector Encoding 16

1.3.5.2 Reference Pictures Management system 17

1.3.6 Transform coding in HEVC 18

1.3.7 Comparison of HEVC and VVC standards 18

1.4 Error Control mechanisms to cope with packet loss 20

1.4.1 Forward Error Resilience 20
1.4.1.1 Source Coding Level protection 20
Layered Coding with Transport Prioritization 20
Multiple-Description Coding 22
Encoder Robustness in prediction process 22
1.4.1.2 Channel Coding 25
1.4.1.3 Joint Source-Channel Coding 26
1.4.2 Interactive Error Resilience 27
1.4.3 Error Concealment by Postprocessing 27
1.5 Conclusion 28

2 Compression Efficiency Evaluations of Video Coding Standards 29
2.1 Introduction 29
2.2 Methodology 31
2.2.1 Case 1: Comparative studies with two-pass encoding 31
2.2.1.1 Software implementations 32
2.2.1.2 Configuration of encoders 33
2.2.1.3 Selection of video sequences 34
2.2.1.4 Evaluation 35
2.2.2 Case 2: Comparative studies with one-pass encoding 36
2.2.2.1 Configuration of encoders 38
2.3 Results and Discussion 39
2.3.1 Case 1: Results and Discussion with two-pass encoding mode 40
2.3.2 Case 2: Results and Discussion with one pass encoding mode 46
2.4 Conclusion 50

3 Error Resilience Evaluations for HEVC standard 52
3.1 Introduction 52
3.2 Video transmission platforms 53
3.2.1 Simulation based frameworks 54
3.2.2 Realistic testbeds 56

3.3 HEVC implementations Error Resilience in error-prone environments 58
3.3.1 Methodology 58
3.3.2 Results and Discussion 59
3.3.2.1 Error Resilience Evaluation 60
3.4 The error resilience performance of new methods introduced to HEVC 63
3.4.1 The effectiveness of different HEVC decoder implementations 63
3.4.1.1 Phase one: One slice Per frame 64
3.4.1.2 Phase two: 6 CTUs Per Slice 66
3.4.2 Activating TMVP in HEVC under Erroneous channels 69
3.5 Conclusion 71

4 Proposed Approach to Enhance HEVC Robustness in a Real-time video transmission 73
4.1 Introduction 73
4.2 Motivation 74
4.3 Proposed solution 75
4.3.1 Proposed solution for the RPS method 76
4.3.2 Proposed approach to decrease the temporal dependency 77
4.4 Evaluation setup 79
4.4.1 Dataset 79
4.4.2 Configuration setup 80
4.4.3 Test conditions 81
4.5 Results and Discussions 81
4.5.1 Experimental results for Intra-refresh updates 82
4.5.2 Experimental results for the proposed strategies 83
4.6 Conclusion 89
Conclusion and future works 90

Bibliography 92

Appendix A HEVC Video Dataset (Test Sequences) 101

Introduction

This thesis investigates the development of error-resilient mechanisms in dealing with packet loss issues due to wireless network impairments. Indeed, to fulfill this need several approaches were adopted. However, before delving into error resilience strategies, this section provides the aim and the motivation behind our work. In the first part, the background and motivation are presented. After that, we elaborate on the faced problems when developing an error-resilient approach. Next, the objectives and the attained contributions are summarized. While the final part contains the description of the thesis outline in a bullet points manner.

1. Context and motivation

In today's world, users over the internet are demanding more HD and U-HD real-time video services and applications. This is mainly due to two factors: 1) the flourishing of inexpensive mobile devices such as tablets, notebooks, and smartphones. 2) the evolution of the internet and wireless communication technologies [1]. According to [2], in 2022, video occupies eighty-two percent of the global internet traffic for both business and consumer. Note that, of all the applications, video surveillance accounts for 3.4 percent of all video traffic [2]. In addition, remote video surveillance systems are rapidly increasing due to the decrease in video cameras cost [2]. However, based on CISCO's statistics [2], video surveillance traffic is now seven times that in 2017 [2]. Accordingly, it is evident that video is the most desirable source of information and entertainment traveling the internet.

In the 1970s, the first video surveillance cameras were initially deployed in public transportation systems, especially in Europe and the United States. The primary objective was simply to ensure the proper functioning of the equipment and the proper management of the passenger flow. However, due to the increase in robbery and terrorist attacks, it is being applied to public transportation, for public safety. In fact, currently, authorities are interested in implementing onboard video surveillance systems in buses. The vast majority of these systems will employ outdoor advanced wireless technologies.

However, storing and transmitting video in RAW format is impossible to accomplish due to the huge size of the video and the limited bandwidth of wireless networks. Thus, throughout history, the

International Standardization Organization (ISO/IEC) and International Telecommunication Union (ITU-T) united their forces to develop video-compressing standards. Beginning with the H.261 and arriving at the newest H.265/HEVC and H.266/VVC coding standards. Each video compression technology aims to outperform its predecessor in terms of coding performance; providing sophisticated tools and techniques to fulfill the targeted objectives. New tools bring not only compression efficiency [3], it decreases the error resilience of the coding standard in the event of packet loss leading to error propagation and thus degrading the user's end video quality. This is more critical for wireless networks as they are time-varying channels. TCP retransmission techniques such as Automatic Repeat Request (ARQ) are error control techniques that are not suitable for real-time video delivery as they impose an additional delay in order to compensate for the lost information. Therefore, this thesis aims to develop a source-level protection strategy to protect the bitstream. Based on the HEVC encoder this thesis enhanced the end video quality in the event of packet loss by truncating the error propagation.

In the literature, there exist numerous error resilience methods [4]. However, these methods are appropriate for older standards, and their implementation with newer codecs such as HEVC is difficult [5]. This difficulty is due to the introduction of novel coding tools such as flexible block partitioning and enhanced reference picture selection method. Consequently, developing error resilience approaches for the newer standards that take new tools into consideration is crucial in order to make their transmission more robust to network errors. Hence, this thesis focuses on the error resilience methods developed for HEVC.

2. Problem Statement

During the standardization of HEVC, the ISO/IEC and ITU aim to increase its bitrate saving by 50 percent according to its predecessor H.264 [6]. Indeed, at this stage, they do not consider the error resilience performance for the newly introduced techniques [1, 6]. The same is applied to each newly born codec such as VVC. Consequently, each time a new video coding standard is developed, its error resilience will be a major concern. Hence, a research gap is created and studies of the efficiency of novel techniques that achieve good compression performance is crucial to efficiently transmit video data over error-prone networks.

As mentioned earlier, error-resilience methods are either specific to older standards such as H.264/AVC and H.263, which makes its adaptation to new standards difficult or not appropriate, or methods that are HEVC compliant. As error resilience approaches provide different performances regarding the nature of implemented systems e.g., real-time applications, a tradeoff between compression efficiency, reliability and interactivity should be considered when developing new techniques. Figure 1. shows the trade-off that should be followed in order to effectively develop an error resilience method.

In this thesis, we deal with real-time video transmission systems that are prone to network errors. Hence, mainly our thesis delves into the implementation of error resilience mechanisms that respect the

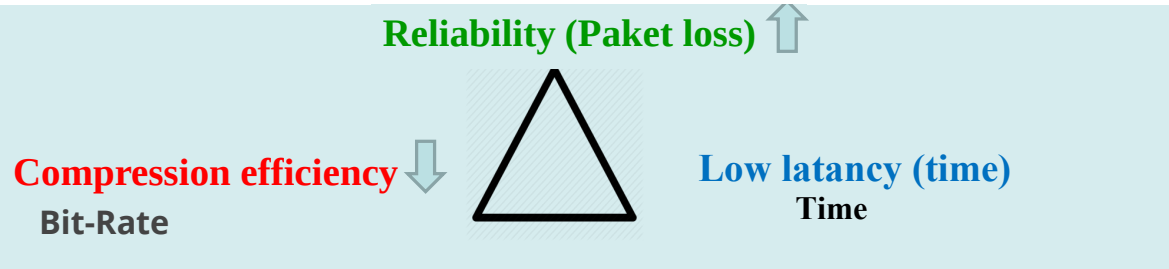


Figure 1 – Trade-off for video delivery.

time-varying characteristics for real-time wireless video delivery. In other words, our major concern is how to properly configure encoding parameters at the source level to enhance the error-robustness of an HEVC bitstream taking into account complexity, bitrate overhead, and the structure of the reference pictures.

3. Objectives and contributions

The research goals of our thesis are outlined within the context of robust video transmission through impaired wireless networks. Mainly, our objective is to provide a technique that enables real-time video transmission and to study the effect of different coding parameters on end-user quality. However, to improve the video quality of an HEVC codec over a lossy environment, this work proposes a unique approach. Indeed, in our work, the following specific goals were pursued:

— Select a convenient video codec based on performance experimental evaluation of new video coding standards: Taking into account complexity, quality and bitrate saving.

— Study the performance and limitations of real-time video transmission through error-prone networks.

— Ensuring a good perceived video quality to end users in real time through mobile and wireless networks.

— Develop an HEVC/H.265 error resilience method at the source level to truncate the error propagation and hide the effect of channel errors on the perceived quality.

In view of the mentioned objectives, we were able to provide the following contributions:

Contribution 1:

J1 Mansri, Islem, Nasreddine Kouadria, Noureddine Doghmane, Saliha Harize, and Amara Bekhouch. “Reference Picture Selection with Decreased Temporal Dependency for HEVC Error Resilience.” Journal of Visual Communication and Image Representation 90 (February 1, 2023): 103724. https://doi.org/10.1016/j.jvcir.2022.103724.

Contribution 2:

C1 N. Kouadria, I. Mansri, S. Harize, N. Doghmane, K. Mechouek. "Lossy compression of color images based on discrete Tchebichef transform". Conference: 14-th International Symposium on Signals, Circuits and Systems (ISSCS 2019). Roumania 2019

Contribution 3:

C2 I. Mansri, N. Doghmane, N. Kouadria, S. Harize. "Comparative Evaluation of VVC, HEVC, H.264, AV1, and VP9 Standards". 8th Algerian Thematic School on Signal Processing & its Applications (ATSPA 2019). Annaba, ALGERIA 2019

Contribution 4:

C3 Mansri, Islem, Noureddine Doghmane, Nasreddine Kouadria, Saliha Harize, and Amara Bekhouch. "Comparative Evaluation of VVC, HEVC, H. 264, AV1, and VP9 Encoders for Low-Delay Video Applications." In 2020 Fourth International Conference on Multimedia Computing, Networking and Applications (MCNA), 38–43. IEEE, 2020.

Contribution 5:

C4 I. Mansri, N. Doghmane, N. Kouadria "HEVC Reference Picture Selection based Error Resilience technique". Eighth doctoral days: June 27-28, 2021, Annaba, ALGERIA 2021

Papers being prepared for publication:

— Mansri, Islem, Noureddine Doghmane, Nasreddine Kouadria “A systematic review of HEVC error resilience: techniques and evaluation frameworks”

— Mansri, Islem, Noureddine Doghmane, Nasreddine Kouadria “Error Resilience performance of different HEVC implementations in lossy environment”

4. Thesis outline

This thesis consists of four chapters organized as follows.

– Introduction

This Section outlines the research work including the motivation behind it and our general interests.

Chapter 1 – Theoretical background of video compression and error resilience

First, the theoretical background of the HEVC video coding standard is presented. In addition, it provides descriptions of the techniques and tools exploited to provide a bitstream that is robust to network errors. Furthermore, this chapter presents an overview of the error resilience methods developed for the HEVC standard and its predecessors H.264 and H.263. Indeed, the chapter provides a detailed description of all the technologies found in the literature to combat packet loss issues. More specifically, recently published works regarding the HEVC error resilience.

Chapter 2 and Chapter 3 – Comparative evaluations

Both chapters consist of different comparative studies. Chapter 2 compares the compression efficiency of different software implementations for a variety of video coding standards including HEVC, H.264,

AV1 and VVC. In this chapter we provide two different works, both are related to real-time video compression. Chapter 3 is related to the error resilience performance of new methods introduced to HEVC and the efficiency of different HEVC implementations (HM and x265) in an error-prone environment. In addition, this chapter, provides simulation platforms and how the channel impairment is modeled using different multimedia frameworks and network simulators such as FFMPEG and ns-3 respectively.

Chapter 4 Proposed Approach to Enhance HEVC Robustness in a Real-time video transmission

Proposes the combination of three error-resilient methods to enhance the end video quality in a real-time HEVC video transmission. The proposed work includes the reference pictures selection algorithm, the Temporal Moving Vector Prediction and Intra refresh methods. In addition, it provides all the methodologies and experiments used with the proposed approach. Finally, it discusses the obtained results and the findings and proposes possible future woks regarding the proposed algorithm.

– Conclusions and future works

This section includes the final remarks and conclusions summarizing the thesis by highlighting the main research accomplishments and limitations. However, further future work recommendations are present as well.

Chapter 1

Theoretical background of video compression and error resilience

This chapter provides an entry into digital video coding technologies. Section 1.2, presents an overview of the old video coding standards developed throughout history. While section 1.3 talks about the HEVC video standard. However, a review of error resilience methods and techniques that combat packet loss issues are elaborated in section 1.4.

1.1 Introduction

Due to the nature of real-time video transmission, several factors affect the quality of the video being received, such as network congestion, varying internet speeds and packet losses. As a result, the video may experience buffering, lag, or poor quality. Indeed, the reliability issue with video surveillance systems especially in real-time is a major concern [4]. As video streams cannot tolerate errors or packet losses, and retransmission methods result in high delays, source-level protection is a great area of interest for such delay-bounded video transmission systems.

High video compression efficiency on the other hand is another factor that affects the end video quality. As the main objective of video coding standards is to double the coding performance according to older standards while preserving the same visual quality, their bitstreams become more sensitive to network errors [7]. In other words, to achieve high compression performance, codecs present enhanced motion estimation and motion compensation techniques. Indeed, these approaches increase the dependency between the temporal motion information of adjacent frames [6]. Thus, if an error hits one area, it will propagate to the next dependent area within the successive frame.

Knowing all the factors that decrease the end video quality will help the development of error resilience methods to combat packet losses. However, in real-time systems, channel characteristics are time-

varying and the Quality of Service (QoS) is not guaranteed. Thus, exploiting source coding techniques is crucial to make bitstreams robust to network errors. In the literature [8], error resilience methods are classified into three categories: *Forward Error Resilience, Error Concealment by post-processing and Interactive Error Resilience.* ***Forward Error Resilience*** techniques add redundancy at the source end. In this group, the encoder is fully responsible for all the work. ***Error Concealment*** techniques exploit the spatial and temporal video characteristics to recover the lost areas. These methods rely on the ability of the human visual system (HVS) to tolerate certain levels of distortions. Indeed, all the data recovery is performed only within the decoder. However, ***Interactive Error Resilience*** makes use of feedback channel updates from the decoder to enhance the robustness of bitstreams. Thus, to be able to perform error enhancements with such approach, communication between encoder and decoder should be present.

To develop an error resilience method that enhances video quality in error-prone networks, a solid knowledge of video coding techniques is crucial. Therefore, the following section provides an overview of video coding standards.

1.2 Overview of old video coding standards

Over the past years, there has been a significant increase in the efficiency of video coding standards. This development is driven by the need to support the growing demand for visual communication systems and to improve the quality of service in real-time video delivery. With the advent of new technologies and the rapid advancement in computing power, video coding algorithms have become more sophisticated and efficient. This has led to a reduction in the amount of data required to transmit video, resulting in a more efficient use of bandwidth and storage resources. Throughout the years, the ITU-T and ISO/IEC have jointly developed a number of video coding standards. Some of the key standards developed by these organizations solely or jointly include:

— H.261: Developed in 1988, this standard is used for video conferencing with Common Intermediate Format (CIF). It was designed to work with the existing Integrated Services Digital Network (ISDN). Hybrid video coding technology was the main coding technology used in this standard. Note that, it was the first standard that adopted such technology, as the previous standards were only relying upon Differential Pulse Code Modulation (DPCM) i.e., H.120. In this paradigm, the encoder replies on both transform coding and predictive coding to efficiently exploit temporal and spatial redundant data. Consequently, good compression gain levels will be achieved. However, the gain in terms of coding efficiency over H.120 was achieved by the introduction of motion-compensated prediction.

— H.262/MPEG-2: This standard can be considered the first video codec developed jointly by the Video coding Experts from of ITU and the Moving Picture Experts Group of ISO/IEC. It was developed in early 1995, this standard is used for digital television broadcasting and DVD

video. MPEG-2 was designed to provide high-quality video at bit rates that are typical of digital television broadcasting. It is also widely used in DVD videos and other consumer applications. One of the key features of MPEG-2 is to support the coding of interlaced video material.

— H.263: H.263 video compression standard was developed in 1996, this standard is used for low-bandwidth applications such as video-conferencing over narrowband networks. It was designed to be used with IP-based networks such as the Internet and was able to achieve good video quality at low bit rates. The initial version of the standard was designed with error-correction capabilities to enhance its robustness to data loss. Newer versions of the standard, H.263+ and H.263++ further increased the error resilience performance. This was achieved by introducing slice structured mode and other tools. However, an improved reference picture selection mode was induced for H.263++ in 2000.

— H.264/AVC: Developed in 2003, this standard is widely used for high-definition television broadcasting and internet video streaming. It was designed as a network-friendly standard and to support both conversational and non-conversational applications. This was achieved by introducing the Network Abstraction Layer concept (NAL). H.264 is developed by both ITU as H.264 and ISO/IEC as MPEG 4 part 10. H.264 provide bitrate saving of about 40-50 % compared to H.263. It introduces new techniques such as Variable block-size motion compensation with small block sizes, Quarter-sample-accurate motion compensation, Multiple reference picture motion compensation and Small block-size transform.

In summary, these standards have been developed over the years to improve video quality and reduce the bitrate required to transmit video. The standards were tailored to different use cases and network conditions, and they continue to be updated and refined as new coding standards to improve their performance and keep pace with advances in technology.

1.3 High Efficiency Video Coding (HEVC)

The increased demand for High-Definition (HD) and Ultra-High-Definition (UHD) video resolutions has led to the development of new video compression standards, such as HEVC [6] and Versatile Video Coding (VVC) [59]. These standards offer significant improvements in compression efficiency over their predecessors. Indeed, they use advanced tools to reduce the amount of spatial and temporal redundant information within video frames. In this section, we will discuss the theory behind HEVC and its essential building blocks. Further comparison of HEVC and VVC implemented methods will be highlighted by the end of this section.

1.3.1 Hybrid Video Compression using HEVC

Since the early stages of video coding technologies, the hybrid video coding paradigm was chosen as it provides better performance by exploiting temporal and spatial redundancy [6]. This paradigm employs two different coding strategies, predictive coding and transform coding. HEVC is one of many standards that work with this solution. Indeed, HEVC introduces new coding tools to provide half bitrate saving according to its predecessor H.264/AVC while preserving the same perceived quality [6]. The encoding diagram of HEVC is depicted in Fig. 1.1.

After a RAW video is fed to the encoder, the process starts by dividing the video frame into smaller blocks called Coding Tree Units (CTUs). These CTUs are then divided into smaller square regions called coding units (CUs), which are the basic units of coding. After the frame partitioning, a residual signal is generated. This residual signal represents the difference between the input CUs and predicted CUs. The predicted CUs are either Intra-predicted or Inter-predicted. In fact, the resulting signal (error signal or residual signal) is then transformed using Discrete Cosine Transform (DCT) or Discrete Sine Transform (DST). In this step, the signal is transformed from the spatial domain to the frequency domain. The next step is to quantize the transformed coefficients, which reduces their precision and reduces the amount of data that needs to be stored. After that, the quantized coefficients are entropy-coded, which is a process of compressing the data by removing the statistical redundancy. In HEVC this is conducted using Context-adaptive binary arithmetic coding (CABAC).

Note that, the quantized coefficients are further processed to be used in the inter-prediction process. This is done by dequantization and inverse transforms. The previously predicted blocks are added to the dequantized residual signal to reconstruct the blocks. In fact, HEVC contains a decoding loop within the encoder. With this, both the encoder and decoder will generate identical predictions for subsequent data. Moreover, this is due to the fact that it uses a technique called rate-distortion optimization (RDO). RDO is a method that allows selecting the best coding options for a given CU based on the trade-off between the rate (i.e., the number of bits required to represent the CU) and the distortion (i.e., the difference between the original and decoded CU). The decoding loop is also used by the in-loop filters that are applied to the reconstructed blocks after decoding. In fact, HEVC utilizes two types of filters, Deblocking filter and a Sample Adaptive Offset (SAO) filter. The first is used for smoothing the blocking artifacts resulting from transform coding and quantization steps while SAO is applied to enhance the quality of the reconstructed video depending on generated offset values from lookup tables. The resulting output is stored in the Decoded Picture buffer for further use in the inter-prediction process. At this stage, all the required data by the decoder should be fed to the CABAC entropy coder. This includes, intra-predicted data, transformed and quantized intra/inter residual signals, motion data, filter control data, and general encoder control data.

On the decoder side, the inverse operations of the encoding should be performed on the bitstream to obtain the reconstructed video. Indeed, to ensure that video can be decoded correctly across different

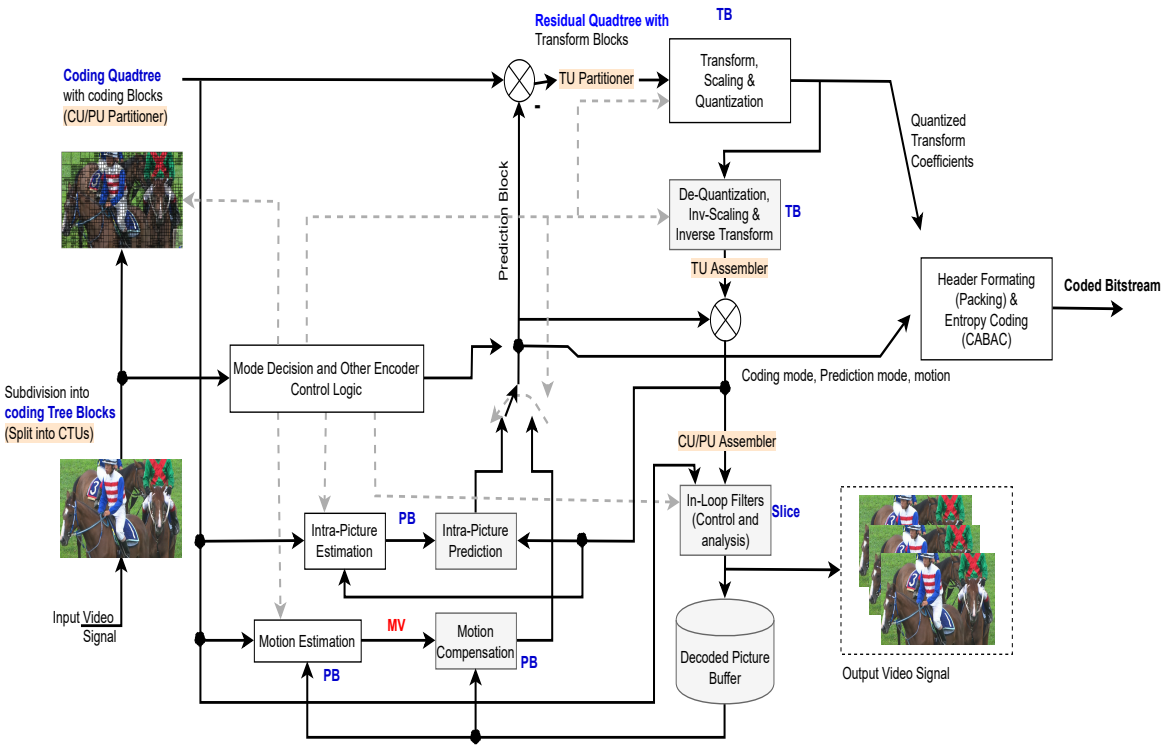


Figure 1.1 – HEVC video encoding diagram

devices, video coding standards aim at restricting the bitstream structure and its syntax elements. These restrictions will allow successful decoding regardless of the implemented encoding methods. More details about the bitstream structure are elaborated in the following subsection.

1.3.2 Packaging of Compressed Video Data

One of the main goals of HEVC is to provide easy integration to the transport system and support most H.264 applications [43]. Indeed, HEVC inherited the packaging technology introduced in H.264 with degrees of enhancements. Network Abstraction Layer (NAL) is a key component in the HEVC standard, serving as the interface between the Video Coding Layer (VCL) and the transmission layer [43]. It provides a flexible way of encapsulating video data onto different transport layers such as RTP/UDP/IP, Adaptive bitrate using Dynamic Adaptive Streaming over HTTP (DASH), and other different varieties of protocols. Figure 1.2 depicts the possible HEVC system layer integrations.

The NAL units constitute the main building block of an HEVC bitstream. Each NAL unit contains a two-byte header that identifies its contents, either video data (VCL-NAL) or encoding parameters (Non-VCL-NAL) [44]. The Non-VCL-NAL unit is crucial as it includes information necessary for decoding such as Video Parameter Sets (VPS), Sequence Parameter Sets (SPS) and Picture Parameter Sets (PPS). The VCL-NAL units contain video slice segments and video slice syntax elements needed to decode an Access Unit (AU). The AU is at the top-level structure of a bitstream as depicted in Fig.

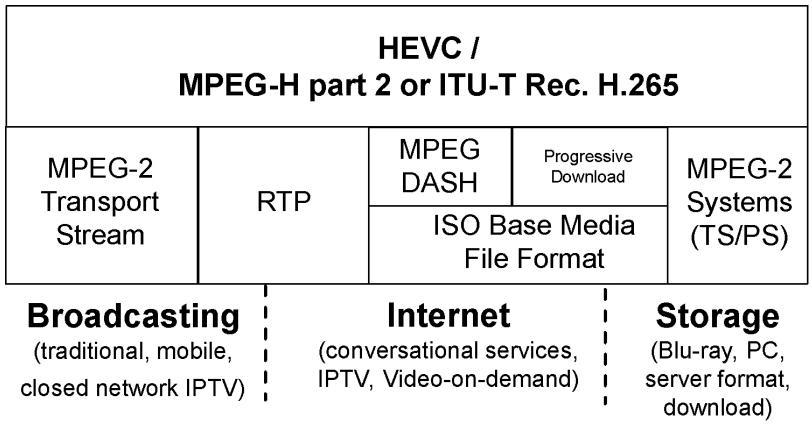


Figure 1.2 – High-level overview of HEVC system layers [43].

1.3. To summarize:

— H.265|HEVC encoded video is transmitted as a bit sequence of NAL units. Also known as NAL Unit Byte streams (Annex B) [112].

— Each NAL unit has a 2-byte header that identifies its contents.

— NAL units are classified into 2 categories: VCL-NAL (video data) and Non-VCL-NAL (encoding parameters).

— Non-VCL-NAL includes important information for decoding, such as VPS, SPS, PPS, and SEI messages.

— VCL-NAL units contain video slice segments and syntax elements to decode an Access Unit (AU).

A detailed representation of an encoded video using HEVC can be found in Fig. 1.3. As can be seen from the diagram, the HEVC NAL unit header size is set to two bytes. Indeed, the header size length for H.264 was only set to one byte. The new associated bits intend to support newly developed features or new video standard extensions. These features include bitstream compliance to Multiview and 3D video applications [44].

1.3.3 HEVC Block Partitioning

According to the fact that HEVC adopts the conventional hybrid video coding structure, it provides significant modifications to the data partitioning compared to earlier standards such as H.264/AVC [113]. H.264/AVC splits each frame into Macroblocks (MBs), comprised of a 16×16 block of luminance samples and two 8×8 blocks of chrominance samples represented as YUV 4:2:0 chrominance sub-sampling format [61]. MBs are treated as the basic processing element in H.264/AVC. Indeed,

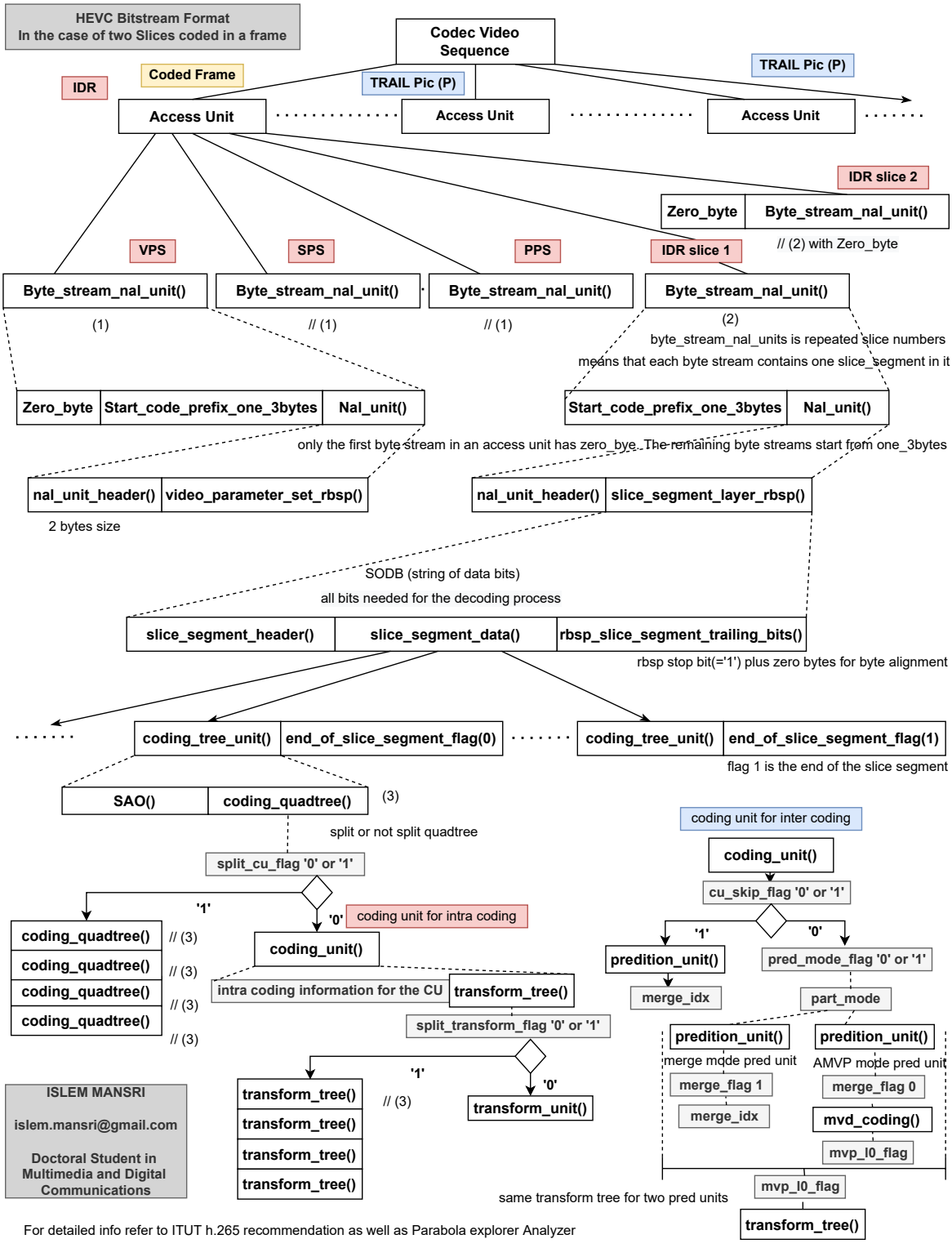


Figure 1.3 – HEVC bit-stream structure diagram.

the coding method determines if the pixels are encoded as intra or inter-predicted. MBs can also be divided into smaller subblocks that maintain the same prediction method.

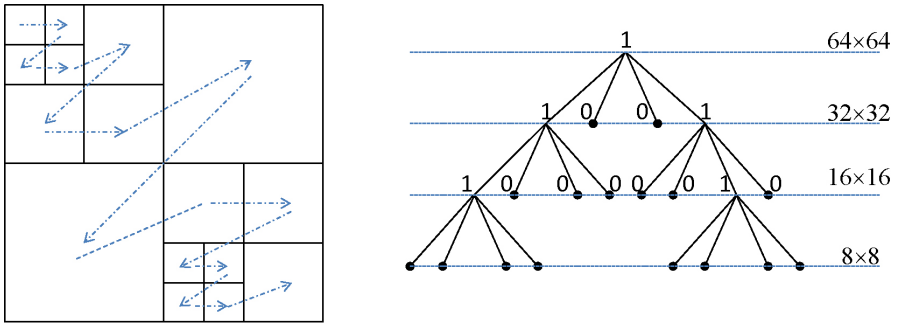


Figure 1.4 – (a) CTU Z-scan partitioning from 64×64 to CUs of size 8×8. (b) Its coding tree structure representation [113].

However, the data structures used in HEVC are different from those used in H.264/AVC. To divide a frame, HEVC uses Coding Tree Units (CTUs), which can have a maximum size of either 64×64, 32×32, or 16×16 pixels, as specified by the encoder [113]. Note that, the size information should be signaled to the decoder. The larger data structures used in HEVC aim to improve coding efficiency, particularly for high-resolution videos [6]. CTUs can also be subdivided into smaller units called Coding Units (CUs) until they reach the smallest size of 8×8 pixels, which is set by the encoder. For instance, when there are homogeneous areas, HEVC will represent them with fewer symbols by using a single large CU instead of using multiple smaller blocks. Figure 1.4 illustrates the CTU quadtree picture partitioning concept adopted by HEVC. This design enables the specification of the multi-level hierarchical quadtree structure in a straightforward and sophisticated manner.

The Coding Unit (CU) is the cornerstone of the HEVC [6]. It is the basic processing unit that is handled by all video compression operations. The CU is divided into two types of data blocks, namely the Prediction Unit (PU) and the Transform Unit (TU) [113]. The PU is used for prediction operations, while the TU is used for transform operations. Note that, the choice between intra-frame or inter-frame prediction modes is made at the CU level. Indeed, the intra-frame prediction mode uses information from the same frame to make predictions, while the inter-frame prediction mode uses information from previously decoded frames [6]. These two types of predictions will be discussed in the following subsections. The partitioning of the CU into PUs can result in different intra or inter-coding modes being used for each PU. This allows for a more flexible and efficient compression process, as different parts of the frame can be treated differently depending on their unique characteristics [74]. By utilizing the CU structure, HEVC is able to achieve a high level of compression while maintaining high video quality.

Indeed, there exist other data partitioning concepts which are different from block partitioning, these include slices and tiles.

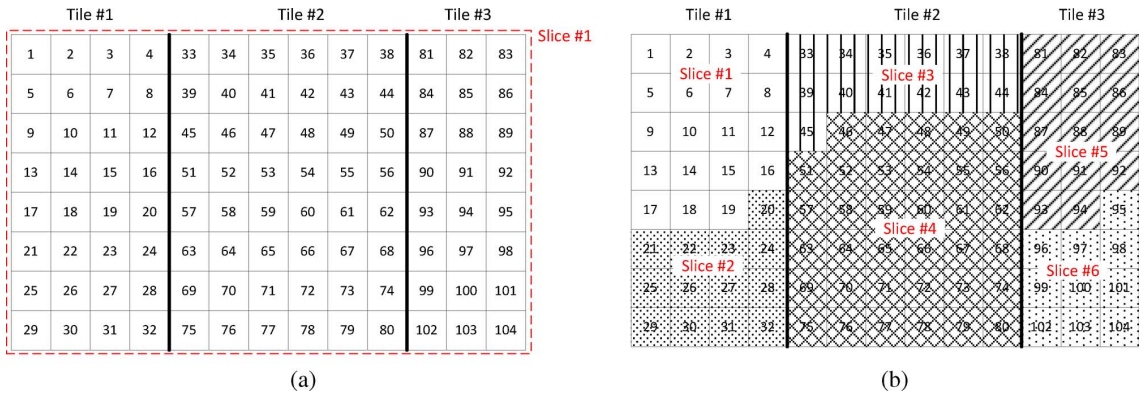


Figure 1.5 – Utilization of Tiles with Slices (a) One slice containing three tiles. (b) One tile containing two slices [114].

Slices

The HEVC standard preserves the slice structure introduced for H.264, this concept is considered a key aspect of the HEVC standard, as it allows for better parallel processing and improved error resilience. By partitioning the video into slices, HEVC enables the data to be decoded and reconstructed independently, providing better flexibility and scalability. The slice partitioning tool also allows for efficient parallel processing, which can significantly reduce the processing time for large videos.

However, the use of slice headers in HEVC results in increased coding overhead, which can negatively impact compression efficiency. To address this, the standard introduced dependent slices, which are more efficient in terms of signaling overhead and can be used to reduce end-to-end delay. Dependent slices allow for partial transmission during processing, making it possible to transmit a video while it is still being processed. This can greatly improve the overall transmission time and reduce the end-to-end delay.

Tiles

High-Efficiency Video Coding introduced a specific data partitioning feature known as "Tiles" [114]. Tiles allow dividing an encoded image into smaller segments containing CTUs through the use of both horizontal and vertical boundary divisions. Figure 1.5 illustrates frame partitioning using vertical boundaries. In this example, we have three Tiles. Indeed, the boundary division is signaled through the non-VCL Network Abstraction Layer (PPS). This will allow efficient partitioning with minimal overhead.

HEVC permits the utilization of Tiles and Slices simultaneously, though constraints have been imposed to simplify the implementation. There are two cases to be considered: Each CTU within a Tile must be part of the same Slice, or all CTUs in a Slice must be contained within the same Tile. The two cases

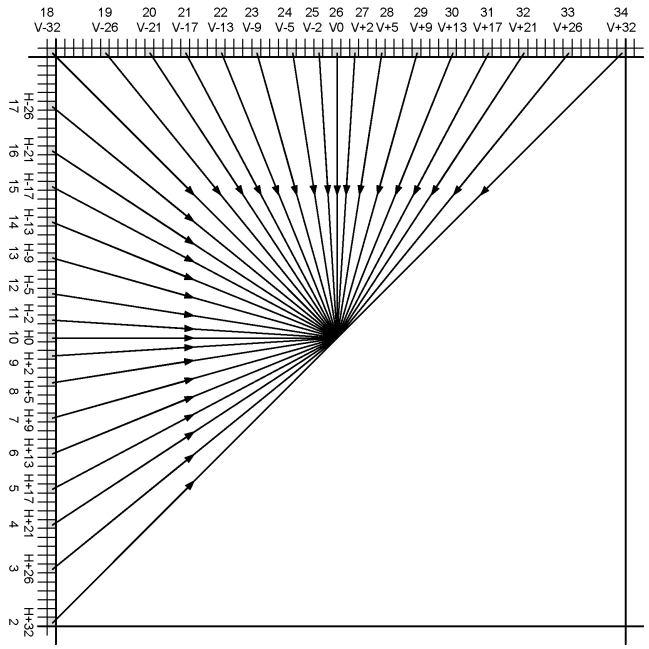


Figure 1.6 – HEVC Intra Prediction angular modes. [116].

can be seen in Fig. 1.5. It is worth mentioning that, Tiles provide several benefits, including improved parallel processing [115], more efficient MTU size targeting, decreased memory requirements for line buffers, and the ability to define Regions of Interest for asymmetrical video coding [114]. These advantages make Tiles a valuable addition to the HEVC standard.

1.3.4 Intra-Picture Prediction in HEVC

Intra-frame prediction is a key technique in the field of video compression, it is used to reduce the amount of redundant information within a single frame. The technique works by analyzing the pattern of pixels within the frame, and making predictions about what the pixels in one part of the frame are likely to be based on the pixels in another part of the frame.

The process of intra-frame prediction begins by dividing the frame into smaller units known as CUs and PUs. If a CU is decided to be intra-coded, the size of the Prediction Unit should typically match the size of the CU, with the exception being when the 8×8 CU block size is selected, at which point the CU may be further divided into four smaller PUs.

HEVC provides a wide range of intra-frame modes to choose from, including Planar, DC, and 33 angular modes [116, 117]. At this stage, the encoder must make a decision on which intra-prediction mode to use. This choice is based on the previously reconstructed pixels and on which mode would be suitable. Figure 1.6 illustrates the possible directional modes used in HEVC.

One of the challenges of intra-frame prediction is ensuring that the encoder has access to all of the

necessary reference pixels in order to make an accurate prediction. In some cases, reference pixels may be missing, which can negatively impact the quality of the prediction. To address this issue, HEVC includes a feature known as reference sample substitution, which allows the encoder to replace any missing reference pixels with substitute pixels in order to maintain the accuracy of the intra-prediction process.

1.3.5 Inter-Picture Prediction in HEVC

Inter prediction plays a crucial role in video compression, as it helps reduce the amount of data that needs to be encoded. The purpose of inter-prediction is to exploit the temporal redundancy between consecutive video frames by using similar blocks or regions in a previously encoded frame, known as the reference frame, to predict the blocks in the current frame [119].

The process of inter-prediction involves finding similar blocks or regions in the reference frame and using them to predict the blocks in the current frame. In HEVC, this is done by using the block-matching algorithm [6]. After the prediction is conducted, the motion vectors are calculated.

HEVC uses advanced techniques to improve the accuracy of the inter-prediction process. For instance, the standard employs more complex motion vector search algorithms, multi-reference frame prediction, and advanced block partitioning techniques. These techniques enable HEVC to provide improved compression efficiency compared to previous video compression standards. Indeed, HEVC utilizes the Advanced Motion Vector Prediction (AMVP) and Merge Mode [119] as two ways to encode or signal the motion information.

1.3.5.1 Motion Vector Encoding

In low-bitrate video applications, motion vectors occupy a significant portion of the total bitrate. Thus, to economically encode this motion data, HEVC introduces two strategies, predictive coding and merge mode encoding. The new tools are based on the fact that the motion vectors of either the spatial or temporally neighboring blocks are highly correlated to the moving vector of the block being encoded.

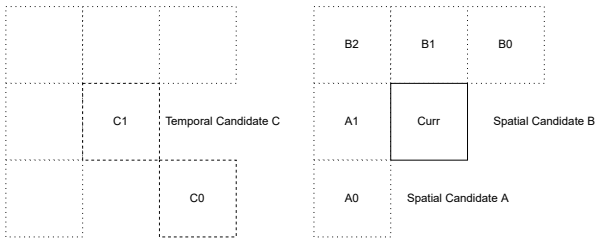


Figure 1.7 – Motion vector predictor candidates in HEVC [103].

Advanced Motion Vector Prediction (AMVP)

In HEVC, predictive coding is conducted using a concept known as the Advanced Motion Vector Prediction technique (AMVP). In AMVP, a list of possible motion vector predictors (MVP) to the MV of the block being encoded is selected. In this list, we can find up to two spatial candidates and one temporal candidate. The possible candidates for the AMVP are illustrated in Fig. 1.7. After examining the neighboring blocks for inclusion and selecting one predictor to the moving vector of the block being encoded, the difference between the predictor and the moving vector is signaled to the decoder. In addition to that, the index of the closest motion vector to the vector of the block being encoded will be signaled as well.

Merge Mode

In the merge mode [119], almost the same procedure presented in AMVP to select the predictors is used. However, the candidate list contains five motion vectors of which, four are spatial candidates and one temporal candidate. After selecting the closest MV to the moving vector of the block being encoded, the former is inherited directly to be used as MV of the current block and not coded using the difference. The only signaled information to the decoder is the merge index i.e., the index of the derived MV.

Temporal Motion Vector Predictor (TMVP)

The introduction of TMVP is new to HEVC compared to its predecessor H.264/AVC. Indeed, it is included to gain compression performance. Note that, The TMVP is used by both the Advanced Motion Vector Prediction and merge mode. To include the temporal moving vector candidates in the list for the AMVP or Merge mode, the HEVC test Model (HM) provides a separate piece of code where the MVs of temporal neighbors are allowed to be selected. You can activate or deactivate it from the configuration file.

The main difference between HEVC and H.264/AVC is the usage of the temporal neighboring motion vectors as candidates as well as the introduction of new spatial possible candidate blocks. However, using the temporal information will increase the dependencies between the motion vectors of different frames [3]. Consequently, the loss of motion information will affect the decoding process as it uses the motion information of the previous block to predict the current block.

1.3.5.2 Reference Pictures Management system

As it is known, the gain in compression efficiency of HEVC compared to H.264/AVC is not related to only one improved strategy but to the overall encoding chain, of which is the reference management system. In H.264/AVC, the signaled information about reference pictures is related only to the changes

that happened in the Decoded Picture buffer (DPB). However, H.264/AVC updates the reference pictures in the DBP after the current frame is completely decoded.

In contrast to that, HEVC introduces a new concept known as Reference Picture Sets to manage decoded frames. With this concept, the encoder manages to signal the entire reference picture list to the decoder via the slice segment header at the beginning of each frame [118]. Once the slice header signal is available, the decoder parses the reference picture set information and updates the DPB. In other words, HEVC does not rely on the DPB state of the previous frame, avoiding the usage of incorrect reference pictures. Therefore, resulting in enhanced error resilience performance.

1.3.6 Transform coding in HEVC

The purpose of the transform coding in HEVC is used to decorrelate the image data, making it easier to compress. The transformation process in HEVC transforms the spatial domain information into the frequency domain, which is then quantized and entropy coded [120]. By transforming the image data into the frequency domain, the most important visual information can be preserved, while the less important information can be removed.

According to the fact that video coding standards have essentially the same baseline for encoding, HEVC utilizes a transformation followed by quantization to code the residual signal. However, the residual block is separated into multiple square segments, commonly known as Temporal Units (TUs) [120].

Indeed, the main transform matrices are derived from Discrete Cosine Transform (DCT) basis functions, which have proven to be effective in image and video compression [121]. However, a 4×4 luma Intra block is coded using the Discrete Sine Transform (DST) instead of DCT. The reason for this is that the DST provides improved coding performance without any expanse of complexity specifically for 4×4 blocks.

Quantization is the next step after the transformation of residual information. It follows the same principle as previous video compression standards and applies a uniform-reconstruction quantizer scheme based on the quantization parameter (QP) value. The QP values range from 0 to 51, with lower values resulting in less coding distortion and vice versa. However, it is worth noting that quantization is the main cause of coding distortion, so finding a good balance between quantization accuracy and compression efficiency is crucial.

1.3.7 Comparison of HEVC and VVC standards

A detailed comparison of the coding tools introduced to HEVC and VVC are highlighted and summarized in table 1.1.

Table 1.1 – All different methods included in HEVC and VVC [126].

	HEVC	VVC
Block partitioning	◆ Coding Tree Unit (CTU) quadtree (QT) structure ◆ From 64×64 to 8×8 Coding Unit (CU) size	◆ CTU quadtree structure with nested multi-type tree (QT+MTT) ◆ From 128×128 to 4×4 CU size ◆ Chroma separate tree (CST) ◆ Local dual tree ◆ Virtual pipeline data units (VPDUs)
Intra prediction	◆ DC, planar ◆ 33 directional prediction modes ◆ Linear interpolation	◆ DC, planar ◆ 65 directional prediction modes ◆ Wide-angle prediction modes ◆ 4-tap interpolation filters (IFs) using 2 sets of filters ◆ Position-dependent prediction combination (PDCP) ◆ Multiple reference lines (MRL) ◆ Matrix-based intra-picture prediction (MIP) ◆ Cross-component linear model (CCLM) ◆ Intra sub-partitions (ISP)
Inter prediction	◆ Merge mode ◆ Advanced motion vector prediction (MVP) ◆ 8-tap IFs for luma ◆ 4-tap IFs for chroma	◆ Extended merge mode and MVP with ▪ History-based MVP (HMVP) ▪ Pair-wise average MVP candidate ▪ Subblock-based temporal MVP (SBTMVP) ▪ Merge with motion vector difference (MMVD) ▪ Symmetric motion vector difference (SMVD) ◆ Adaptive motion vector resolution (AMVR) ◆ 8-tap or 6-tap IFs for luma ◆ 4-tap IFs for chroma ◆ Geometric partitioning mode (GPM) ◆ Bi-prediction with CU-level weight (BCW) ◆ Combined Intra/Inter prediction (CIIP) ◆ Decoder-side motion vector refinement (DMVR) ◆ Bi-directional optical flow (BDOF) ◆ Affine motion ◆ Prediction refinement with optical flow (PROF)
Forward/inverse transform and quantization	◆ Square transform (up to 32×32) ◆ Discrete cosine transform and discrete sine transform ▪ DCT-II and DST-VII ◆ Sign data hiding (SDH)	◆ Square and rectangular transform (up to 64×64) ◆ Multiple transform selection (MTS) ▪ DCT-II, DST-VII, and DCT-VIII ◆ Non-separable secondary transform (LFNST) ◆ Subblock transform (SBT) ◆ Adaptive chroma QP offset ◆ SDH ◆ Dependent quantization (DQ) ◆ Joint coding of chroma residuals (JCCR)
Entropy coding	◆ Context-adaptive binary arithmetic coding (CABAC) ◆ Coefficient group ◆ Reverse diagonal, horizontal and vertical coefficient scan	◆ CABAC with multi-hypothesis probability estimates ◆ Additional coefficient group size ◆ Reverse diagonal coefficient scan only ◆ Improved probability model sections for absolute transform coefficient levels
Loop filtering	◆ Deblocking filter (DF) ◆ Sample adaptive offset (SAO)	◆ Luma mapping with chroma scaling (LMCS) ◆ Deblocking boundary handling modifications ◆ Deblocking long filter ◆ Luma-adaptive deblocking ◆ Sample adaptive offset (SAO) ◆ Adaptive loop filter (ALF) ◆ Cross-Component ALF (CC-ALF)
Parallelization	◆ Slices ◆ Tiles ◆ WPP with CTU row delay of two CTUs	◆ Slices ◆ Tiles ◆ Subpictures ◆ WPP with CTU row delay of one CTU

1.4 Error Control mechanisms to cope with packet loss

In this section, we will discuss the main error control mechanisms to mitigate the effect of errors on the end video quality. As mentioned previously, error resilience methods can be categorized into three groups. Encoder-based techniques, Decoder based techniques and encoder-decoder-based techniques. A taxonomy of all the techniques is depicted in fig 1.8. However, to understand the details, it is advised to fully understand how video communication systems work. Figure 1.9 presents the diagram of a real-time video communication system. In such systems, we have three main parts: Source codec, Transport codec and the transmission channel. Note that by codec we are referring to both the coder and decoder. However, source coder is when the data compression paradigm is concerned. In our case, the hybrid video coding scheme, such as that used by HEVC. This contains a predictive coder, transform coder and entropy coder. The transport coder contains the channel coder, the packetizer and the modulation.

After compressing and providing an HEVC-compliant bitstream by the source coder, the bitstream is further put through the channel coder. For channel coding, Forward Error Correction (FEC) can be used as a channel error protection method. After that, the RTP protocol [9] is used for packetization. RTP is the best protocol that can be utilized for transmitting real-time data. Usually, RTP is used with UDP as RTP/UDP/IP protocol stack. Indeed, after encapsulating the bitstream, we send it via the transmission channel and perform all the inverse operations on the decoder side.

1.4.1 Forward Error Resilience

One of the main categories is forward error resilience. This category aims at enhancing the bit-stream robustness to network errors by applying source, channel, and joint source-channel mechanisms. In this section, we first talk about different techniques that exist within source-level protection group. Techniques such as Layered video coding and multiple description coding will be reviewed. After that, we will talk about Encoder Robustness in the prediction process and the parameters or techniques used within the encoding process which enhance the end video quality. Finally, we will discuss, channel coding and joint source-channel techniques.

1.4.1.1 Source Coding Level protection

Layered Coding with Transport Prioritization

Layer Coding is a video coding strategy that employs encoding data into different segments. It is a special case of scalable video coding. Note that scalable video coding is developed for H.264 and HEVC as an extension to their baseline codec [10, 11] i.e., Main single layer H.264 and HEVC codec. In layer coding, the video is segmented into a base layer with one to several enhancement

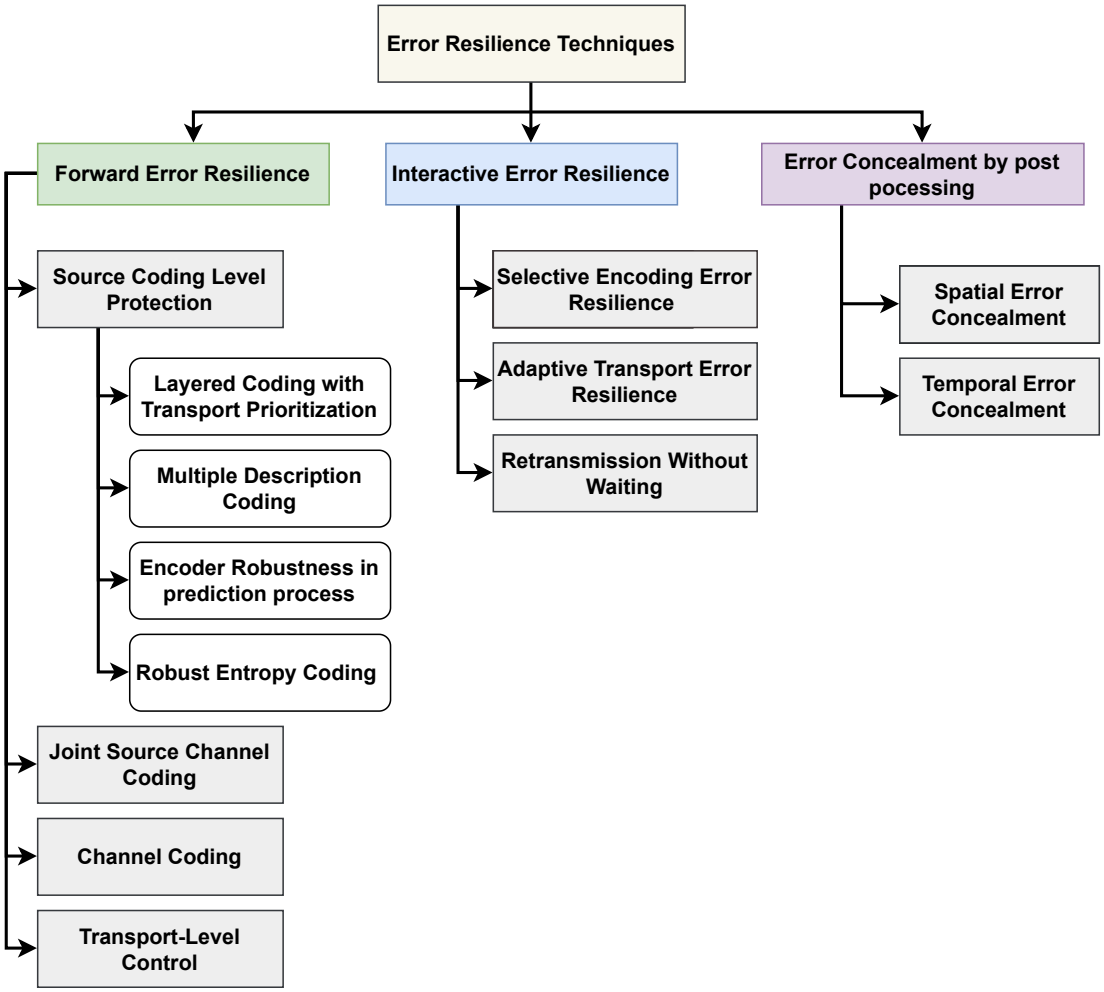


Figure 1.8 – Taxonomy of error resilience methods

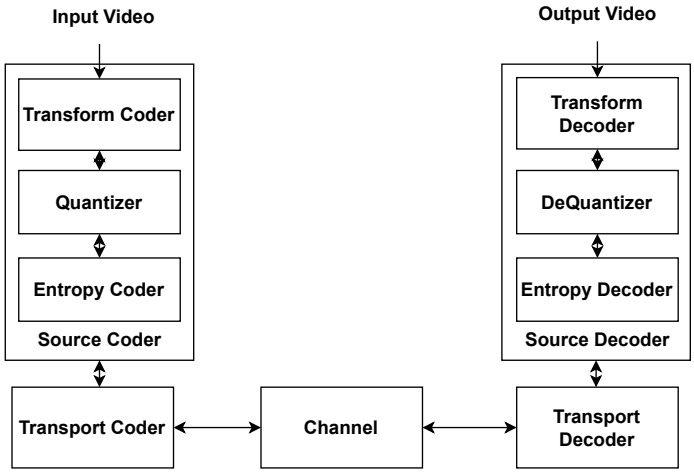


Figure 1.9 – Video Communication systems block diagram

layers. The main purpose of such technology is to allow different decoders with different capabilities to efficiently reconstruct the encoded video with different levels of quality. The quality will be ranging from acceptable to high based on the fact that the decoder can either receive the base layer solely or the base layer with enhancement layers. To enhance the error resilience performance of Layered video coding, transport prioritization is employed [12]. The transport of the layers is prioritized based on their level of significance, with the most critical layers being protected against channel error. There are several ways to enhance the error resilience of layer coding including:

— Unequal Error Protection by applying FEC to the Base Layer.

— Retransmission: In which the base layer will be retransmitted if affected by network channels.

— Decrease the inter-prediction between layers in order to mitigate the error propagation between them.

— Increase independently decodable units by adding a header to each layer and other informative syntax elements such as information used for encoding.

The only drawback with this approach is that it cannot guarantee a fully reliable transmission for the base layer using FEC. Therefore, the re-transmission technique is the only promising way for such a scenario. Based on this fact, Layered Coding may not be convenient for real-time video surveillance systems as the latter cannot tolerate high transmission delays caused by retransmission techniques. To overcome this issue, Multiple-Description Coding (MDC) is seen as a promising alternative.

Multiple-Description Coding

Multiple Description coding is another error resilience mechanism. In this approach, video data is encoded as descriptions. These descriptions are different representations of the video data. Within the description, redundant data exists which will help in video reconstruction at the decoder side in the event of packet loss. However, MDC assumes that bitstreams will be sent through a number of parallel paths and these channels may suffer from burst losses. Indeed, the loss in all the paths is considered independent, meaning that there is a low possibility that the loss may happen to the same portion of the bitstream within different channels. Each description provides an acceptable quality at the decoder. However, the highest quality will be noticed if all the descriptions are successfully decoded. Figure 1.10 shows an MDC framework with four descriptions. Note that, the black blocks in this figure indicate the lost information. According to [13, 14], multiple description coding provides better performance than Layered video coding for real-time video applications. However, for VoD applications, LC provides better quality than MDC with low to moderate loss rates. Conversely, according to [13, 14], MDC is seen to be suitable for a high loss rate in this type of applications. For an in-depth analysis of MDC refer to [15, 16, 17]. Several MDC works related to HEVC can be found in [18, 19, 20, 21, 22, 23, 24]

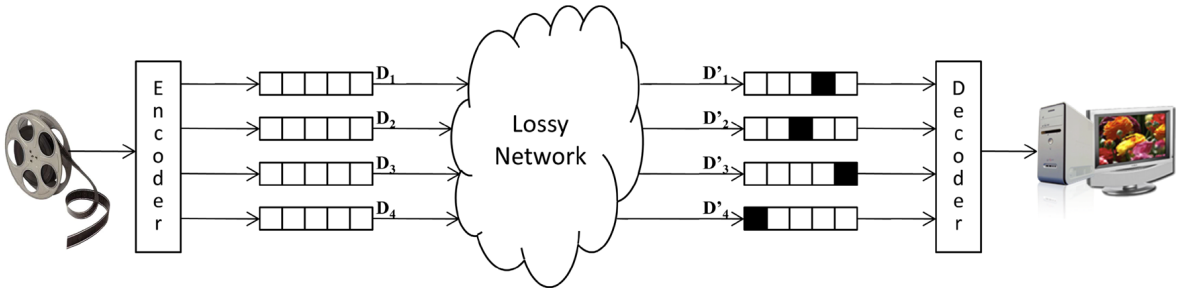


Figure 1.10 – MDC videos transmission through an erroneous multipath channel [17].

Encoder Robustness in prediction process

Robust Source Coding is to optimize the parameters of the source encoder carefully in order to get the best tradeoff between compression efficiency and error resiliency. In this approach, it is crucial to take into account the effect of losses and the effectiveness of the error concealment technique. However, we can enhance error resilience within two main processes: Prediction and entropy encoding. Due to the fact that robust entropy coding was not used with newer codecs such as HEVC, entropy-based error resilience is not covered in this section. Refer to [8] for detailed information about robust video coding with older standards. However, used approaches within the prediction process include:

— ***Reduce motion-dependent data:*** This approach is based on increasing the number of independently decodable units based on the inter-prediction loop. In other words, reducing the relation between the encoded moving vectors. To do so, several works have been conducted [25, 3, 26]. All works have evaluated the error resilience of motion compensation techniques in HEVC. The results show that the Temporal Moving Vector Predictor (TMVP) improves coding performance while decreasing bitstream robustness to network errors. To overcome this issue, they propose solutions such as deactivating TMVP at the frame level, reducing temporal dependency at the block level, and conveying selected moving vectors as redundant side information to the decoder to enhance the reconstructed video quality.

— ***Intra-refresh:*** it is a well-known method that breaks the propagation of errors due to the high inter-prediction dependencies. Its main drawback is the large amount of the bitrate budget allocated to these intra-frames in a rate-constrained transmission [27]. Therefore, when frame level Intra refresh is applied, the quality of the subsequent inter-coded frames in the same Groupe of Picture (GOP) will be degraded. However, Intra-refresh if used properly will serve as an efficient tool for Error Resilience. Note that it can be employed for some randomly chosen blocks or based on the R-D cost function. Moreover, it can be used for whole-frame coding. Indeed, there exist several works that deal with Intra coding to enhance error resilience. In [28, 29], the authors proposed an algorithm called Recursive Optimal per-Pixel Estimation (ROPE). This algorithm estimates the decoder-side distortion by applying pixel-wise operations.

Based on the estimated distortion, the encoder adaptively switches between Intra/Inter modes. Other works that extend the ROPE algorithms for error resilience usage in lossy environments can be found in [30, 31, 32, 33, 34]. Note that all previous works are related to older standards. However, for HEVC-related works that utilize ROPE, we can find [35, 36]. In [36], the authors decreased the error propagation at a deeper level. Based on the fact that each Coding Unit (CU) in a Coding Tree Unit (CTU) can be either Intra/Inter coded, they found that when this flexible mode is used, more errors can propagate. Moreover, this is due to the sub-pixel interpolation in Intra CUs. To solve this issue, based on ROPE, they proposed a fixed mode coding resulting in a good performance in terms of end-to-end Peak Signal to Noise Ratio (PSNR) in lossy environments. The drawback of the ROPE pixel-based end-to-end prediction model is that it is highly complex. To overcome such issues, researchers in [37] propose a Low Complexity Adaptive Intra/Inter mode selection algorithm based on deep learning. This model achieved good end-user quality in the event of packet loss. However, in [38], the authors presented an adaptive intra-refresh algorithm in which they predict the end video quality at the decoder prior to transmission. They achieve that by enforcing Prediction Units (PUs) to be encoded using intra mode. The weakness with this approach is that they assign a high number of Intra PUs than is needed and thus lowering the performance for low bandwidth networks and when small loss rates are in the channel. For further reading about Intra coding usage as an error resilience tool refer to [39, 40, 41, 42, 111].

— ***Data Partitioning:*** Data partitioning also known as data isolation, is a technique that is used to improve the error robustness of codecs by dividing the coded data into different parts. This is done to make sure that the important information is stored in different partitions. For instance, header information is stored in one partition, while the rest of the data is stored in another partition. This technique was used in codecs such as H.263, MPEG-4 and H.264/AVC. In H.264/AVC, data is partitioned into three parts, namely A, B, and C. Part A stores header information and motion vectors, part B stores intra-coded blocks, and part C stores inter-coded information. This way, if one or more partitions are lost, the decoder can still use the information in the remaining partitions to conceal the error. However, it is not supported by newer codecs such as HEVC [43].

— ***Parameter sets:*** VVC, HEVC and H.264 standards include parameter sets that can be used to enhance error resilience [44, 45, 46]. These sets, known as SPS, PPS, and VPS, store parameters including video information and picture sets information, making the encoding and decoding process more efficient. Although parameter sets are not inherently error resilient, they can be utilized in combination with other error-resilience techniques to maximize the robustness of the video stream against errors. By ensuring the reliable reception of parameter sets and transmitting them frequently or with robust error correction, the decoder can effectively adapt to changes in the video stream and maintain a high level of video quality even in the presence of errors.

— ***Picture Segmentation:*** Slices are a powerful tool that can be used to enhance the error resilience

performance of HEVC and H.264 video compression standards [45]. Slices divide a video frame into smaller, independent units that can be encoded separately. This allows the decoder to recover from errors more easily, since only a portion of the frame may be affected by errors, rather than the entire frame. Using slices may decrease the compression efficiency as the prediction across slice borders is not allowed. In addition, the slice header can increase the bitrate overhead. Generally, this can be compensated with the gain in error resilience performance as it provides enhanced video quality at the decoder in error-prone channels. However, more performance can be gained if slices are reordered and not used linearly. A concept known as Flexible Macroblock Ordering (FMO) [47] allows video frames to be encoded in a non-linear order. FMO was used in H.264, its ability to enhance error resilience is seen on the decoder side as it helps recover the lost blocks. Tiles on the other hand can be used as an error resilience tool as well. Authors in [48] inherited the capability of slices to recover packet losses and used it with tiles. In other words, based on the fact that slices are independently decodable units and according to the fact that the rectangle shape of tiles provides good performance, they created a new concept known as tileslice. The tileslice concept treats tiles as slices by adding to a tile a header. This makes bitstreams robust to network errors and provides good video quality on the decoder side.

— ***Redundant data:*** duplicated frames or slices can be sent within the bitstream to help recover the lost information in an erroneous event. This can be achieved by decreasing the compression efficiency of the replications and embedding them within the bitstream.

Indeed, methods such as FMO, redundant slices, data partitioning, and SP/SI pictures are not widely used in real-world applications thus they have been omitted in HEVC [43].

1.4.1.2 Channel Coding

Channel coding is a technique used in digital communication systems to add redundancy to the transmitted data in order to improve the robustness of the transmission against errors. In the context of error resilience video transmission, channel coding can be used to protect video data against errors.

The channel coding is known as Forward Error Correction (FEC). FEC codes add redundant information to the video data, which can be used at the receiver to detect and correct errors that may have occurred during transmission. One of the most efficient FEC codes is the RaptorQ code [49]. Indeed, Raptor codes are a type of rateless code, which means that the sender can generate an unlimited number of encoded symbols from the original data, allowing the receiver to request additional symbols until it has enough information to recover the original data. They are designed to be highly efficient, with the encoded symbols having a much smaller size than the original data. However, RaptorQ code [50], is a new class of rateless codes and is an extension of the Raptor codes that can handle packet loss more efficiently and is more suitable for large data files, such as video streaming. RaptorQ codes are designed to operate over lossy networks and can correct a large number of errors and erasures with low

overhead. Indeed, other examples of error-correcting codes are Reed-Solomon codes, Turbo codes, Convolutional codes, Polar codes, and Low-density parity-check (LDPC) codes. These codes can be applied at the channel level to help detect and correct errors.

In the literature, we can find several works implementing these kinds of methods [51, 48]. In [48], researchers use RaptorQ codes as an FEC method with HEVC video transmission. Based on their obtained results, FEC is seen as mandatory for video communication systems to guarantee acceptable video quality at the decoder.

In a nutshell, using channel coding techniques can help improve the error resilience of video transmission over noisy channels and ensure a high-quality video experience for the viewer.

1.4.1.3 Joint Source-Channel Coding

Joint source-channel coding (JSCC) and joint source-channel decoding (JSCD) are techniques used in video coding and transmission to improve the compression efficiency and robustness of video transmission over noisy channels. These techniques aim to overcome the limitations of traditional separate source-channel coding by jointly optimizing the source and channel encoders and decoders.

Conventionally, source and channel encoding and decoding are optimized separately, as explained in Shannon's separation theorem [52]. Shannon's separation theorems state that source coding and channel coding can be performed separately and independently to achieve optimal performance. However, this theorem relies on the assumption that states that the source and channel are stationary [52, 53]. Moreover, it assumes that their delay can be infinite. In practical applications, separate source-channel encoding and decoding can lead to suboptimal performance. This is because the performance of the source and channel encoders and decoders are interdependent and cannot be optimized separately. For example, the channel encoder may add redundancy to the data to improve error correction capability, but this redundancy can negatively impact the compression efficiency of the source encoder. Similarly, the channel decoder may use error correction codes to correct errors in the data, but this can negatively impact the performance of the source decoder.

JSCC and JSCD aim to overcome these limitations by jointly optimizing the source and channel encoders and decoders. During the JSCC process, the source encoder and channel encoder work together to compress the video data and channel coding information, so that the redundancies hidden in the source coding can be used as extra extrinsic information for channel coding. The channel decoder can use this extra information to improve its accuracy in correcting errors that may occur during transmission. Related HEVC works in this regard can be found in [54, 55, 56].

1.4.2 Interactive Error Resilience

Interactive error resilience is where the network system provides backward feedback channel updates. In this scenario, the encoder and decoder cooperate to enhance the perceived video quality. The cooperation between the two ends can be conducted at the source level or at the transport level. If realized at the source level, the encoding parameters will be changed according to the error status received from the decoder side. When it is used at the transport level, the encoder will adaptively change the redundancy rate for the used channel coding method and thus saving the bandwidth. Based on the fact that HEVC recent works are interested in changing the encoding parameters relying on the error status, we will address this category in this section.

— ***Selective Encoding Error Resilience:*** In HEVC, Reference Picture Selection (RPS) plays an important role in enhancing bitstream error resilience and improving video quality. The decoder can send feedback updates on the error status to the encoder, which can then use that information to adjust the reference pictures for future frames. With this the affected reference frames will be avoided and the end video quality will be enhanced. The issue with this strategy is that the error still propagates at the decoder side when the feedback channel is absent. To deal with the issue, researchers in [57], propose a hierarchical-P Reference Picture selection algorithm, this work truncates the error propagation by relying on a block-based Region of Interest (ROI) method that identifies moving regions in the video and protects them from errors using Intra-prediction encoding. Additionally, the algorithm aims to decrease the complexity of non-ROI regions by limiting the Coding Unit (CU) splitting depth level, which results in significant improvement in terms of objective quality metrics when compared to the conventional RPS algorithm. Other researchers [58] follow the same procedure while providing improvements to the first work. Their algorithm extracts the most active slices of the video as an ROI and encodes it using Intra Mode to decrease the error propagation at the slice level. The ROI extraction is conducted using the frame differencing method and the greyscale projection method. Moreover, the researchers split the non-ROI region and gradually decreasing the quality of ROI and non-ROI regions based on their importance. Results show increased end video quality compared to other works. Although the two works enhance the end video quality, they increase the bitrate overhead which occupies more bandwidth, especially in rate-constrained video transmission. Based on this fact, in chapter 4 we provide a new way to overcome the underlying issues by utilizing new inter-prediction methods rather than the intra-coding methods.

1.4.3 Error Concealment by Postprocessing

Error concealment is a technique that is used in video coding to hide errors at the decoder. It aims to reduce the visibility of errors for the end user, thereby improving the overall quality of the video. Error concealment techniques are crucial as all error-resilient source coding methods cannot ensure the

complete delivery of information to the decoder. However, due to the diversity of video content, it can be challenging to find a suitable error concealment tool. The decoder may have to predict missing information, such as the texture and motion of missing blocks. Error concealment methods utilize the correlation and smooth changes of adjacent pixels, blocks, and frames, with the exception of areas with sharp edges or scene cuts. Before implementing any error concealment, it is essential to first detect the error. Several error detection techniques used for older standards can be found in [8]. However, HEVC can detect the loss of a frame efficiently by using the reference picture sets concept. HEVC reports that a frame is lost if it is included in the Reference Picture Set but not available as a reference in the buffer. Indeed, there exist several different error concealment techniques that can be used, each with its own advantages and disadvantages. Some popular methods include:

— ***Spatial error concealment:*** This technique uses information from surrounding pixels to estimate the value of the missing or corrupted blocks. It relies on the assumption that the image is spatially correlated and that the errors are confined to small regions.

— ***Temporal error concealment:*** this approach uses information from previous or future encoded frames to estimate the value of the missing regions. It is based on the fact that the image is temporally correlated.

Overall, the choice of error concealment technique will depend on the specific application and the types of errors that are likely to occur. Research in this field is ongoing and new methodologies are being developed in order to improve the overall quality of the video.

1.5 Conclusion

In conclusion, this chapter provides an overview of digital video coding technologies, including a history of old video coding standards and an in-depth examination of the HEVC standard. In addition, the chapter provides a review of error resilience methods and techniques related to older standards as well as specific to HEVC. Indeed, it sets the foundation for the rest of the thesis, providing an understanding of video coding and video delivery fundamentals. In other words, it serves as basic knowledge for further exploration and analysis of related topics in subsequent chapters.

Chapter 2

Compression Efficiency Evaluations of Video Coding Standards

Video coding standards are essential for efficiently transmitting and storing video data. In recent years, there has been a proliferation of different video coding standards, each with its unique features and capabilities. In this chapter, we aim to evaluate and compare the performance of several popular video coding standards in terms of compression efficiency. Our evaluations are based on a range of performance metrics such as encoding/decoding time and quality of the decoded video. We also consider the strengths and limitations of each standard, as well as their potential applications and prospects.

The results of our evaluations provide valuable insights into the relative performance of different video coding standards and can inform the selection and design of video coding systems. Overall, this chapter aims to contribute to the growing body of research on video coding standards and to provide a comprehensive and up-to-date analysis of their performance and capabilities.

2.1 Introduction

Video compression is a vital technology for enabling efficient storage and transmission of video content [6]. In recent years, several codecs have been developed to address the increasing demand for higher compression efficiency, including the High-Efficiency Video Coding (HEVC) [6] standard, Versatile Video Coding (VVC) [59], and AOMedia Video 1 (AV1) [60] HEVC is a video compression standard that was developed as a successor to H.264/MPEG-4 AVC [61]. It was developed by the Joint Collaborative Team on Video Coding (JCT-VC) and was officially published in 2013. HEVC is designed to provide higher compression efficiency and improved video quality compared to H.264 standard, while also supporting a wide range of resolutions and applications. It has been widely adopted in various applications and devices, including video streaming, television, video conferencing,

and smartphones. However, its adoption has been met with some controversy due to concerns about licensing and royalty fees for the use of the standard [62]. VVC is the successor to HEVC it was developed by the Joint Video Experts Team (JVET) and was officially published in 2020. It is designed to provide even higher compression efficiency and improved video quality compared to HEVC while addressing the current trends in video technologies. However, as was the case with HEVC, there are some challenges and concerns that need to be addressed, such as licensing and royalty fees for the use of the standard. Despite these challenges, VVC is seen as a promising advancement in the field of video coding [59]. AV1 is a next-generation video coding standard that was developed by the Alliance for Open Media (AOM) as a high-quality, royalty-free alternative to previous standards. It was developed using a collaborative, open-source approach and was designed to improve upon the compression efficiency and video quality of the VP9 standard. AV1 was officially published in 2018 and has since been widely adopted in the streaming industry and used in various devices and platforms. It is well-received as a promising alternative to previous video coding standards and is expected to play a significant role in the future of video.

Note that the efficiency of a codec can greatly impact the quality and speed of video transmission [63]. Indeed, it is important to select the appropriate codec that makes an efficient balance between the aforementioned factors. As mentioned earlier, there are many different available codecs [64, 65, 66, 67, 68], each with its unique features and capabilities. Indeed, by thoroughly evaluating and analyzing the performance of these different codecs, it is possible to determine which one is best suited for a particular application [69]. This can involve comparing the compression efficiency, computational complexity, and other factors of the various codecs.

In the literature, a significant amount of research on the performance of different video codecs are conducted, including HM, VP9, and x264. Research in [69] focuses on evaluating the performance of HM, VP9, and x264 in various contexts and for different applications, such as low-delay and real-time video applications. The obtained objective results show that HM outperforms both x264 and vp9 in terms of compression efficiency at the expense of computational complexity. The same results were obtained in [70] for the same codec implementations. In contrast to [69], we can see that authors in [70] conducted subjective quality evaluations. After the release of VVC and AV1, researchers were interested in comparing their coding efficiency to HEVC. The work in [71], compared the coding performance of HM and Joint Test Model (JEM) as representative of the VVC standard. The obtained subjective results show that VVC outperforms HEVC. The same results were obtained for [72] in the context of random-access configuration. However, in [72], additional codecs were included in the experiments namely AV1 and VP9. Moreover, researchers in [73] conducted extensive experiments on additional video codecs, including VTM, JEM, HM, x265, x264 and AV1. The experiments were on different configurations and different contexts. The obtained results show that VTM outperforms all the tested software implementations.

Upon the examination of the aforementioned research works, we noticed that they yield varying results

for the same standards. This is likely due to various factors, such as the type of video content, the resolution of the test sequences, the software used for encoding and the evaluation metrics used in the studies [63]. However, based on the fact that there is no existing literature that examines the performance of all VTM, HM, x265, JM, x264, AV1, and VP9 for real-time applications, we published the work [63] to fill this gap. Current chapter evaluates the performance of these codecs in more depth using two different experiments providing unique work in this research area. Moreover, it aims to gain a better understanding of the relative strengths and weaknesses of each codec and provide insights into their potential use cases and limitations specifically for real-time applications.

In the following sections, we will describe our research and how it builds upon and extends the work of all previous studies.

2.2 Methodology

In this study, we aim to investigate the performance of different software implementations for video encoding, with a focus on two main cases: comparative studies with multi-pass encoding and comparative studies with one-pass. To achieve this goal, we selected a range of software implementations and video sequences. Indeed, we carefully configured the encoders to meet the specific requirements of each case. In the following sections, we will describe the methods that we used in more detail, including the software implementations, the selection of video sequences, and the configuration of the encoders. Through these methods, we aim to provide a comprehensive evaluation of the performance of the selected software implementations and to identify the strengths and limitations of each approach.

2.2.1 Case 1: Comparative studies with two-pass encoding

Two-pass or multi-pass rate control encoding [69] is a technique that is commonly used in video encoding to improve the efficiency and quality of the resulting bitstream. It involves running the encoding process multiple times, with each pass adjusting the encoding parameters based on the results of the previous pass. In the first pass, the encoder analyzes the input video and generates a statistical summary of the content as a log file, which is used to inform the encoding decisions in the second pass. This summary can include information such as the distribution of colors, edges, and motion vectors in the video, as well as the relative importance of different parts of the frame. This log file is then used in subsequent passes to fine-tune the encoding parameters to achieve the desired balance between file size and video quality. This can include techniques such as rate-distortion optimization [74], which balances the trade-off between quality and bitrate. Moreover, it adjusts the quantization level for different parts of the frame based on their complexity and importance.

Multi-pass encoding can be used with a variety of video codec implementations, including x264, x265, AV1, and VP9. It is often used in applications where a high level of quality or efficiency is required,

such as streaming [69]. In addition, it can offer significant improvements over single-pass encoding. Moreover, it requires more time and computational resources, and may not be suitable for all scenarios [69].

2.2.1.1 Software implementations

For the comparative studies with two-pass encoding, we selected a range of software implementations, including VTM for VVC, HM and x265 for HEVC, JM and x264 for H.264 and AV1 and VP9. These software implementations were chosen for their wide adoption and popularity, as well as their advanced features and capabilities.

VVC has various software implementations to test new tools and methods. Benchmark Set (BMS) [64] and Joint Exploration Model (JEM) [64] are deprecated implementations for VVC. Currently, the final VVC representative implementation is called the Versatile video coding Test Model (VTM) [64]. In our work, we rely on this platform as the main encoder for VVC. Particularly, VTM version 3.2.

HM (HEVC Test Model) [65] is a software tool that can be used to test the performance of High-Efficiency Video Coding (HEVC). It can be used to evaluate the coding efficiency and coding complexity of an HEVC model, as well as other performance metrics. It can also be used to compare the performance of different HEVC implementations under variety of conditions, such as different resolutions and frame rates, different types of video content, and different configuration patterns. Note that HM is considered to be the most widely used HEVC encoder. However, there exist different encoder solutions for HEVC which offer different trade-offs between the amount of time required for encoding and the level of compression efficiency achieved. x265 [66] is an open-source implementation of the HEVC standard, developed by MulticoreWare. It is designed to provide a different level of optimizations regarding the fully-fledged HEVC test model HM and is widely used in a variety of applications, including streaming, video on demand, and broadcast. In this study, we used HM version 16.17 and x265 v2.8.

For H.264 we chose to work with JM [67] version 19 and x264 v.r2935 which are similar to HM and x265 for HEVC. In other words, x264 [68] is the open-source optimized version implementation of the JM implementation developed by VideoLAN. Note that both x264 and x264 support multi-pass encoding mode.

AV1 is designed to outperform the state-of-the-art standard HEVC in terms of coding efficiency. It can be implemented using the AV1 encoder solution. In this work, we selected the AV1 version v1.0.0. VP9 is a video coding standard that was developed before AV1. It can be implemented using the VP9 encoder implementation. Both AV1 and VP9 support multi-pass encoding, which can enhance the rate-distortion performance. The multi-pass mode used in VP9 is similar to that used in x265, x264, and AV1. Note that, VP9 is often used as a reference for benchmarking the performance of AV1. Version v1.7.0-1652-g6b02a12 of VP9 is used in this study.

Table 2.1 – Selected encoding settings for all software implementations for case one

CODEC Version	Parameters
VTM v3.2	Default low-delay P configuration file.
HM [65] v16.17	Default Main profile, low-delay P configuration file as described in [65]
JM 19.0	Default HM-Like, high profile, low-delay P configuration file
x265 v2.8	Similar to HM configuration file –profile main –tune psnr –ref 4 –merange 64 –keyint -1 –min-keyint 99999 –bframes 0 –b-adapt 0 –no-b-pyramid –no-weightb –aq-mode 0 –no-cutree –crf $QP –tskip-fast –tskip –rect –amp –pass $P
x264 [70] r2935	—profile high –tune psnr –ref 4 –direct auto –weightp 2 –level 5.1 –subme 8 –b-pyramid none –bframes 0 –b-adapt 0 –merange 24 –me tesa –no-fast-pskip –trellis 2 –min-keyint=99999 –keyint "infinite" –pass $P –slow-firstpass –qp $QP
AV1 [75] v1.0.0	–cpu-used=0 –threads=0 –profile=0 –lag-in-frames=0 –min-q=$QP –max-q=$QP –auto-alt-ref=1 –passes=$P –kf-max-dist=9999 –kf-min-dist=9999 –drop-frame=0 –static-thresh=0 –arnr-maxframes=7 –arnr-strength=5 –sharpness=0 –undershoot-pct=100 –overshoot-pct=100 –frame-parallel=0 –tile-columns=0 –end-usage=q –cq-level=$QP –tune=psnr
VP9 [70] v1.7.0	–rt –cpu-used=0 –profile=0 –threads=0 –lag-in-frames=0 –min-q=$QP –max-q=$QP –end-usage=vbr –auto-alt-ref=0 –kf-max-dist=99999 –kf-min-dist=99999 –static-thresh=0 –bias-pct=50 –drop-frame=0 –minsection-pct=0 –maxsection-pct=2000 –arnr-maxframes=0 –undershoot-pct=100 –sharpness=0 –codec=vp9 –passes=$P –tune=psnr

2.2.1.2 Configuration of encoders

Low-delay prediction mode [76] is a configuration setting or a prediction pattern that can be used when encoding video using a video codec. It is designed to reduce the delay between the time a frame is captured and the time it is displayed, which can be important for applications such as video surveillance i.e., real-time applications in general.

To use low-delay prediction mode, you will need to set the appropriate configuration settings for your video codec. This may include settings such as the GOP size (the number of frames between intra-coded frames), the number of reference frames, and the type of prediction used for inter-coded frames. In VTM, HM and JM cases, a low-delay P configuration file is provided, it selects by default the appropriate parameters. In contrast, codecs such as x265, x264, AV1, and VP9 do not provide default configuration files for low-delay encoding. Instead, these codecs require the user to specify each parameter manually using the command line shell. It is important to note that using low-delay prediction mode comes with certain trade-offs, such as reduced coding efficiency and potentially lower video quality compared to the Random-access mode. As such, it is important to carefully consider

the use case and the balance between delay and quality when choosing the configuration settings. In this work, as we are addressing real-time applications and encoding time is an important factor, we chose to work with the IPPP prediction structure. Note that for all the software implementations, and to conduct a fair comparison, we used similar configurations as presented in Tab. 2.1.

The choice of the parameters for the manual command cases was not arbitrary. However, our choice was based on previous works such as in [70]. Based on [70] we used the recommended encoding parameters for x264 and VP9 with slight adjustments. While we received the AV1 parameters from a senior developer at Google from the AV1 team [75]. Indeed, based on our experiments with different codecs we provided the coding parameters for x265 as depicted in Tab. 2.1.

To match the default configuration pattern in providing only one Intra frame at the beginning of the video sequence, we used the following options for AV1 and VP9: –kf-min-dist and –kf-max-dist. We set their values to a number that is greater than the number of images. However, for x265 and x264, we set: –keyint to "-1" and –keyint to "infinite" respectively. While –min-keyint to a large value for both x265 and x264.

In video encoding, the quantization parameter (QP) determines the amount of lossy compression that is applied to the video. A lower QP value results in higher quality but also a larger file size, while a higher QP value results in lower quality but a smaller file size. Setting the gap value between min-q and max-q to be 8 for x265, x264, AV1, and VP9 can mimic the hierarchical QP cascading effect used in implementations such as HM, JM and VTM. This can be useful when trying to achieve a balance between quality and bitrate. Moreover, the rate control parameter for industry-driven codecs is set to the constant quality approach.

Mention that, in addition to the above settings we enabled the so-called multi-pass encoding mode employed for x264, x265, VP9, and AV1.

2.2.1.3 Selection of video sequences

The video sequences were obtained from [77]. However, to ensure the diversity of our selected dataset we used two video categories i.e., class E and class E'. The two categories include a range of video content types and complexity levels that cover the full range of SI/TI values on the SI/TI plan. The spatiotemporal information (SI/TI) terminology is used by ITU-T [78] to describe the complexity of a video in terms of both spatial (image) and temporal (movement) information. The SI/TI plan is a graphical representation of the range of complexity of video content, with SI on the x-axis and TI on the y-axis. The SI/TI plan is divided into regions, with each region corresponding to a specific level of complexity. Figure 2.1 shows the distribution of the selected sequences on the SI/TI plan. Note that these classes (E and E') are not suitable to evaluate random-access applications. Instead, they are used to study the performance of low-delay configurations. However, they can be used to evaluate Intra configurations. Table 2.2 describes the amount of motions and the scenes in the videos.

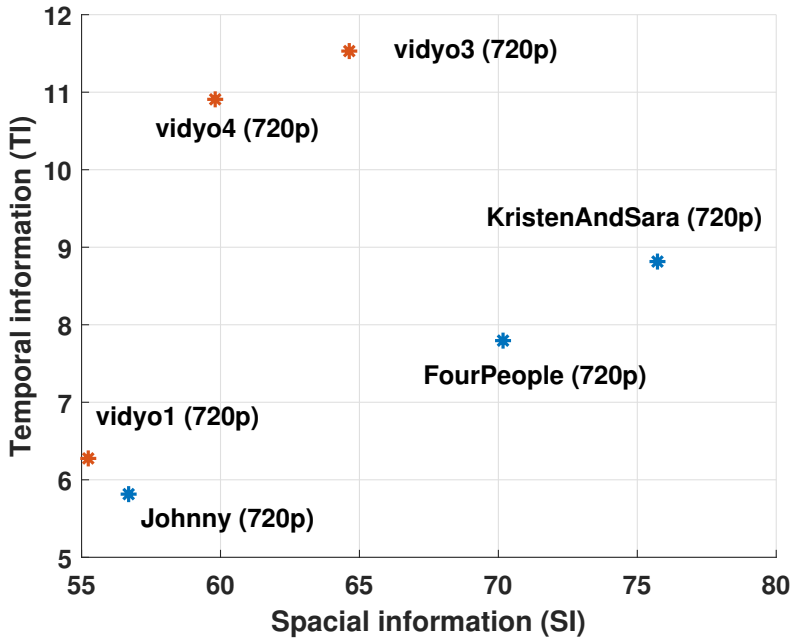


Figure 2.1 – The SI/TI plan for class E and Class E' test sequences.

Note that, utilizing the same video sequences that were used in the development of the H.265/HEVC and VVC (Versatile Video Coding) standards could introduce bias in our outcome. A description of the HEVC dataset is provided in this section **??**.

Table 2.2 – Test Video Sequences for Interactive applications

Class	Sequence name	Content Description
E	FourPeople	Four people doing HD video conferencing inside a meeting room (arms moving).
	Johnny	Still background and low local motion
	KristenAndSara	Still background and moderate local motion
E'	Vidyo1	Still background and low translational motion
	Vidyo3	Moderate texture background and high motion
	Vidyo4	Still background and rapid change in the scene

2.2.1.4 Evaluation

In this study, we used three different metrics to evaluate the compression efficiency of the codecs namely: PSNR ($PSNR_{YUV}$), BD-rate, and BD-PSNR.

PSNR (Peak Signal-to-Noise Ratio) [79] is a measure of the quality of reconstructed data compared to the original data. It is calculated by taking the ratio of the maximum possible power of a signal to the power of the noise present in the reconstructed signal. A higher PSNR indicates a higher-quality reconstruction, while a lower PSNR indicates a lower-quality reconstruction. A PSNR value of around

30 dB or higher is generally considered to indicate a good-quality reconstruction, while a value below 20 dB may indicate a poor-quality reconstruction. Indeed, in our experiments, we used $PSNR_{YUV}$ [74]. It is calculated based on the weighted summation of the three PSNR Luma ($PSNR_Y$) and Chroma ($PSNR_U$, $PSNR_V$) components as demonstrated in Eq. (2.1).

$$PSNR_{YUV} = (6 \cdot PSNR_Y + PSNR_U + PSNR_V)/8 \tag{2.1}$$

Where the three components $PSNR_Y$, $PSNR_U$, and $PSNR_V$ are calculated individually as:

$$PSNR = 10 \cdot log_{10}\left(\frac{MAX^2}{MSE}\right) \tag{2.2}$$

Where MAX is the maximum possible pixel value (e.g., 255 for 8-bit images or 1023 for 10-bit images). MSE (Mean Squared Error) is the average squared difference between the original and reconstructed pixels, calculated as:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left[I(i,j)) - \widehat{I}(i,j)\right]^2 \tag{2.3}$$

Where $m \times n$ is the image size and $\widehat{I}$ is the reconstructed image. Note that i and j are the coordinates for the pixel values.

BD-rate (Bjontegaard Delta Rate) [80] is a metric that compares the bitrate of two codecs at a given objective quality level. It is calculated by taking the difference in bitrate between the two codecs at a given PSNR or another objective quality measure and representing the difference in terms of percentage. A negative BD-rate indicates that the first codec has a lower bitrate than the second codec at the given quality level, while a positive BD-rate indicates that the first codec has a higher bitrate.

BD-PSNR (Bjontegaard Delta Peak Signal-to-Noise Ratio) is similar to BD-rate, but instead of comparing the bitrates of the two codecs, it compares their PSNR values. Similarly, it is calculated as the difference between the two codecs and represented in terms of percentage.

These metrics allow the comparison of the performance of codecs in terms of both quality and bitrate. In addition, they provide a comprehensive view of the compression efficiency of the codecs. According to [62] the Bjontegaard measurement is widely used in the standardization of video codecs as a way to compare the performance of different coding schemes. It provides a convenient way to quantify the overall rate savings or quality difference between two coding schemes, allowing for easy comparison and evaluation.

2.2.2 Case 2: Comparative studies with one-pass encoding

One-pass encoding involves analyzing the video only once and making decisions on how to encode it based on that analysis. In contrast, multi-pass encoding involves analyzing the video multiple times and making more refined encoding decisions based on the multiple analyses.

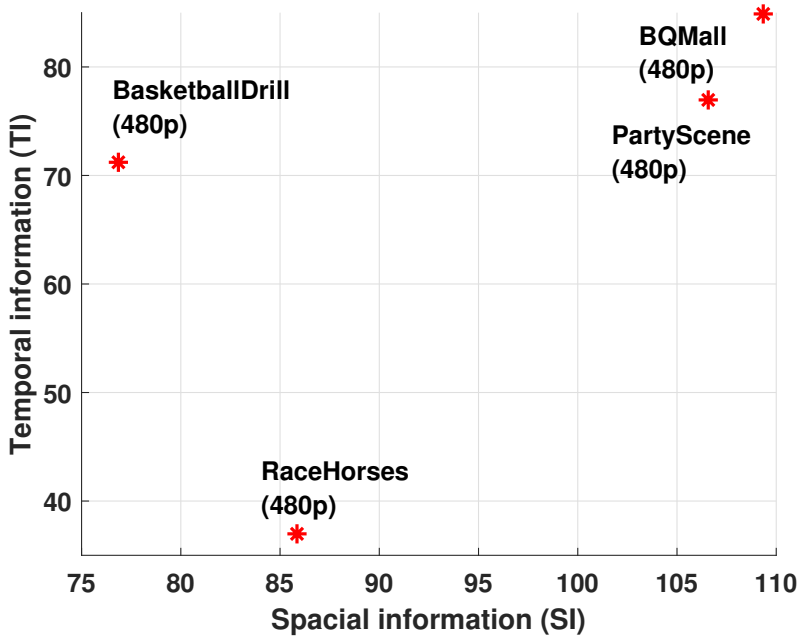


Figure 2.2 – Spatial and temporal information indices distribution for class C videos.

Table 2.3 – Test Video Sequences for mobile applications

Class	Sequence name, fps	Content Description
C	RaceHorses, 30	Low movement and moderate texture
	BQMall, 60	High texture and high translational movement (rapid change)
	PartyScene, 50	High texture background and high motion
	BasketballDrill, 50	Still background and rapid change of the

One-pass encoding is generally faster than multi-pass encoding, but it may not produce as high of a quality result because it does not have the benefit of analyzing the video multiple times. On the other hand, multi-pass encoding is slow for the reason that it involves analyzing the video multiple times. However, it may produce a higher quality result as it can make more refined encoding decisions.

This section aims at comparing the same video codecs namely: AV1, VP9, VTM, HM, x265 and x264 with different encoding configurations. Note that, new versions of the codecs were used. Indeed, besides the real-time nature of our system, in this part, we addressed mobile applications. Hence, we used a different dataset with different encoding parameters that accurately addresses mobile real-time applications i.e., one pass encoding option. Consequently, in this subsection, we only rely on 480p video sequences from class C. We followed the same steps to select the dataset based on the Common test conditions and the ITUT [77] for the SI/TI plan. Figure 2.2 illustrates the distribution of the video sequences addressing mobile applications. While Tab. 2.3 contains more content description including the amount of motion in a scene.

Table 2.4 – Selected encoding settings for all software implementations for case two

CODEC Version	Parameters
VTM v8.0	Default low-delay P configuration file.
HM v16.20	Default Main profile, low-delay P configuration file as described in [65]
JM 19.0	Default HM-Like, high profile, low-delay P configuration file
x265 v3.3	Similar to HM configuration file –profile main –tune psnr –ref 4 –merange 64 –keyint -1 –min-keyint 99999 –bframes 0 –b-adapt 0 –no-b-pyramid –no-weightb –no-wpp –frame-threads 1 –preset $Preset –aq-mode 0 –no- cutree –qp $QP –tskip-fast –tskip –rect –amp
x264 [70] r2991	–profile high –tune psnr –ref 4 –direct auto –weightp 2 –subme 8 –b-pyramid none –bframes 0 –b-adapt 0 – merange 24 –me tesa –no-fast-pskip –trellis 2 –min-keyint=99999 –keyint ”infinite” –threads 1 –lookahead-threads 1 –preset $Preset –qp $QP
AV1 [75] v1.0.0	–cpu-used=0 –threads=0 –profile=0 –lag-in-frames=0 –auto-alt-ref=1 –passes=1 –kf-max-dist=9999 –kf-min- dist=9999 –drop-frame=0 –static-thresh=0 –arnr-maxframes=7 –arnr-strength=5 –sharpness=0 –undershoot- pct=100 –overshoot-pct=100 –frame-parallel=0 –tile-columns=0 –end-usage=q –min-q=$QP –cq-level=$QP –max- q=$QP –tune=psnr
VP9 [70] v1.8.2-98-gc2aa152	–rt –cpu-used=0 –profile=0 –threads=0 –lag-in-frames=0 –min-q=$QP –cq-level=$QP –max-q=$QP –end-usage=q –auto-alt-ref=0 –kf-max-dist=99999 –kf-min-dist=99999 –static-thresh=0 –drop-frame=0 –arnr-maxframes=0 –undershoot-pct=100 –sharpness=0 –codec=vp9 –tune=psnr

2.2.2.1 Configuration of encoders

For the encoding parameters, we used the same configuration files for VTM, HM, and JM as they do not support multi-pass encoding. However, for VP9 and AV1 we used slightly different parameters mostly related to one-pass encoding. For x265 and x264 we used two different presets i.e., “medium” and “Ultra-fast”. The "medium" preset in a video encoding process is typically a compromise between coding efficiency and computational complexity. This preset is often used when there is a need to balance the quality of the encoded video with the amount of time it takes to encode it.

The "Ultra-fast" preset, on the other hand, is designed to minimize the encoding run time at the cost of increased bitrate. This preset is often used when there is a need to encode video faster, even if it results in a slightly lower-quality video or a larger file size.

It’s important to note that the specific characteristics of the "medium" and "Ultra-fast" presets will depend on the specific video encoding software or hardware being used. The terms "medium" and "Ultra-fast" are generally used to describe a range of settings within the software. Table 2.4, provides the encoding parameters with their versions for the one-pass subsection.

Table 2.5, summarizes the differences as well as the similarities between the two case studies: case

one and case two.

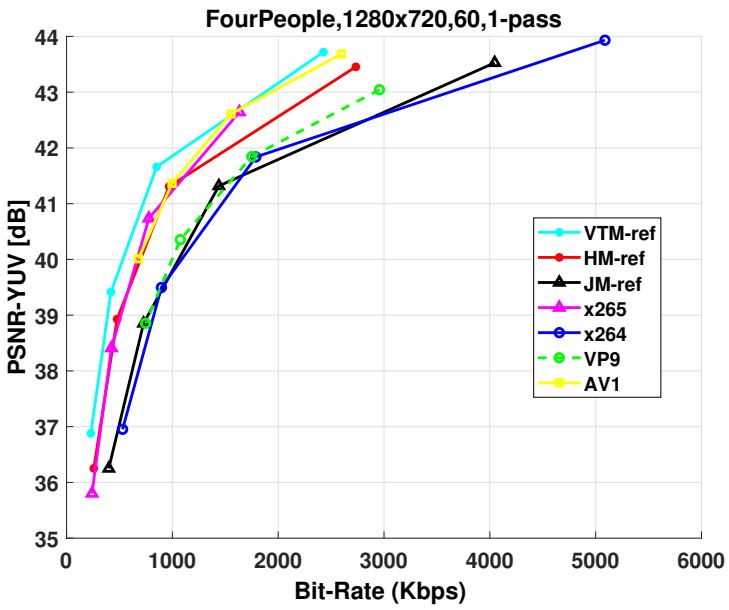
Table 2.5 – Summary of the both similarities and differences for both works

	Work 1	Work 2
Video sequences	- Class E and E' - 150 frames for simplicity	- Class C - All frames were used
Software implementations	- VTM, HM, x265, JM, x264, AV1, and VP9 - older version of encoders	- Same codecs - Newer version of encoders
Configuration of encoders	- Two pass mode - Only medium preset for x265 and x264 - same QP for all codecs	- One pass mode - Medium and Ultrafast presets for x265 and x264 - different QP values for AV1 and VP9
Evaluation metrics	- PSNR and BD-PSNR, BD-rate	- Same metrics

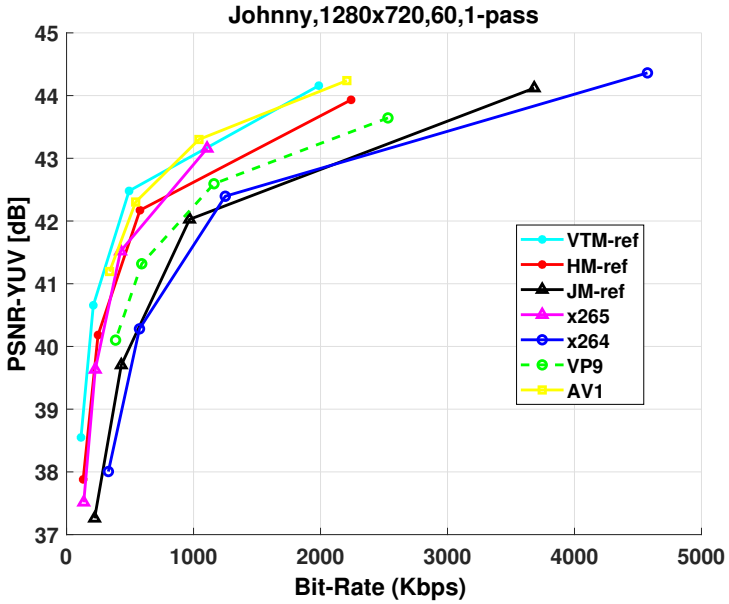
2.3 Results and Discussion

This section presents the results and discussion of the conducted study on the efficiency of several state-of-the-art video compression algorithms. As mentioned earlier, to ensure the robustness of our results, we used a diverse set of test sequences including both static and dynamic content. For case one, and for simplicity, only 150 frames were used. While for the second case we used all the frames present in class C. The tested video codecs were chosen to represent a range of different approaches to video compression, including both traditional and modern techniques.

To ensure that we adequately meet the computational demands of the different codecs, our experiments were conducted on a PC equipped with an AMD 8 Core CPU, 3.7 GHz and 16GB of RAM. Indeed, we selected different QP values for both cases according to [77]. However, for case one, we used QP 22, 27, 32, and 37 for all the codecs including AV1 and VP9. These QP values are recommended to be used for HEVC benchmarking [77]. In contrast, and according to the fact the latter QP values provide different bitrates for AV1 and VP9, and to be as fair as possible, for the second case we used QP 27, 35, 46, and 55 for it instead i.e., changed only for AV1 and VP9. The results of our experiments demonstrate the relative strengths and weaknesses of the various codecs, with some algorithms achieving significantly better performance in certain cases.



(a)



(b)

Figure 2.3 – First pass rate-distortion curves for all codecs regarding class E and E' videos.

2.3.1 Case 1: Results and Discussion with two-pass encoding mode

Performance evaluation of the encoder implementations for first-pass encoding

Figure 2.3 depicts the Rate-Distortion (R-D) curves for all the selected codecs for the first pass mode. As is observed from the R-D curves, the VVC standard achieves the most enhanced compression performance. The results are noticeable for each one of the chosen video sequences regardless of its

Table 2.6 – Average bitrate savings for HM performance for first pass mode

Sequence	**HM vs x265**	**HM vs VP9**	**HM vs JM**	**HM vs x264**
FourPeople	0.74 %	-33.20 %	-33.85 %	-36.69 %
Johnny	-5.10 %	-35.75 %	-46.15 %	-51.46 %
KristenAndSara	-3.07 %	-35.58 %	-43.35 %	-45.19 %
Vidyo1	-9.70 %	-32.90 %	-44.78 %	-45.27 %
vidyo3	-2.32 %	-20.78 %	-27.39 %	-34.71 %
Vidyo4	-0.11 %	-28.51 %	-33.34 %	-38.08 %
Average BD-BR	**-3.26 %**	**-31.12 %**	**-38.14 %**	**-41.90 %**
Encoding Run Time	**~ 1079,25**	**~ 56,96**	**~ 4,46**	**~ 4868,75**

resolution and the scene complexity. While H.264, archives the least amount of coding performance. When comparing the curves of JM and x264, we can clearly see that they are slightly different.

According to the aforementioned description of AV1, it is designed to be superior to HEVC. Indeed, the results of our experiments, as shown in Fig. 2.3, support this claim. However, to better compare each coding implementation in detail, in the following, we provide statistical information on the bitrate savings in the form of tables.

Table 2.6 provides the average bitrate saving for the compared performance of the HM implementation.

As it is seen from the table, the Bjontegared Bitrate (BD-BR) saving of HM according to x265 is 3.26 % on average. This coding enhancement comes at the expense of encoding run time. In addition, we can see that the overall gains for HM are due to the savings achieved with low-motion scenes i.e., Johnny and Vidyo1.

However, when comparing HM with VP9, we can see that HM reduces the bitrate by 31.12 percent at the expense of computational complexity (represented as encoding run time).

Furthermore, when the performance of HM is compared to JM and x264, HM provided a bitrate saving of about 38.2% and 41.9 % respectively. The increase in encoding time of the reference software of HEVC according to JM as well as x264 is by a factor of 4,46 and 4868,75 respectively. The least gain of the HM software implementation can be seen with high-motion video sequences Vidyo3 and Vidyo4.

From the results illustrated in Tab. 2.7, the AV1 encoder shows its superior performance regarding HM. The bitrate saving is seen to be 9.01%. The performance is noticed particularly for the high motion and low motion test sequences Vidyo3 and Johnny respectively. Indeed, an increase in the computational cost by a factor of 2,04 in terms of encoding run time is required to achieve these gains.

Furthermore, it is clearly observed that AV1 outperforms its predecessor VP9 in terms of bit-rate saving by about 36.27 %. However, AV1 is 116,47 times slower than VP9.

Table 2.8 below compares the coding performance of VTM using AV1 and HM reference software.

Table 2.7 – Average bitrate savings for AV1 encoder.

Sequence	**AV1 vs HM**	**AV1 vs VP9**
FourPeople	-3.0467	-32.9343
Johnny	-14.1895	-43.8373
KristenAndSara	2.7922	-32.8222
Vidyo1	-5.7090	-34.7929
vidyo3	-23.2873	-37.9153
Vidyo4	-10.5970	-35.3063
Averages	**-9,01 %**	**-36,27 %**
Encoding Run Time	**~ 2,04**	**~ 116,47**

Table 2.8 – Average bitrate savings for VTM encoder.

Sequence	**VTM vs AV1**	**VTM vs HM**
FourPeople	-20.44 %	-22.98 %
Johnny	-16.68 %	-28.28 %
KristenAndSara	-25.66 %	-25.39 %
Vidyo1	-18.03 %	-22.99 %
vidyo3	-14.05 %	-33.44 %
Vidyo4	-15.49 %	-24.61 %
Averages	**-18,39 %**	**-26,28 %**
Encoding Run Time	**~ 0,65**	**~ 1,32**

As shown in Tab. 2.8, VTM achieves the same visual quality as AV1 with a bit-rate reduction of about 18.4 %. In addition to that, VTM achieves the claimed results with a decreased amount of time by a factor of 0,65.

Furthermore, it can be observed that the performance of VTM relative to AV1 increases when encoding moderate motion video sequences i.e., FourPeople and KristenAndSara.

However, according to the results of VTM versus HM, we can see that VTM provides enhanced performance of about 26,28 % in terms of bitrate saving. In fact, the overhead indicated in terms of encoding time applied to achieve these gains is by a factor of 1,32.

Table 2.9 presents a summary of the experimental results for the first pass mode for all the representative encoders. From the table, we can see that VTM which is the representative implementation for VVC outperforms all the existing video codec implementations in terms of compression efficiency. However, in several cases, VTM encoding time increases to achieve such performance.

On the other hand, we can see from Tab. 2.9 that AV1 outperforms its predecessors i.e., both academia and industrial-driven standards. in terms of bit-rate saving. Note that, the encoding time imposed by AV1 is much higher.

Table 2.9 – Summarized bitrate saving results for the first-pass experiments

CODEC	VVC	HM	x265	JM	x264	VP9	AV1
VVC	/	-26,28	-28,28	-54,15	-56,67	-49,43	-18,39
HM	36,01	/	-3.26	-38.14	-41.90	-31.12	10,87
x265	39,96	3.5	/	-36.97	-41.49	-30.23	10,51
JM	120,28	63,70	60,08	/	-6,04	9,59	71,71
x264	133,14	73,97	72,42	6,54	/	13,99	75,64
VP9	98,30	45,95	44,43	-8,35	-11,90	/	57,50
AV1	22,82	-9,01	-8,17	-41,26	-42,48	-36,27	/

Table 2.10 – Average bitrate savings for the performance of HM for second pass mode.

Sequence	HM vs x265	HM vs VP9	HM vs x264
FourPeople	0.74 %	-29.13 %	-36.62 %
Johnny	-5.10 %	-30.57 %	-51.59 %
KristenAndSara	-3.07 %	-30.03 %	-45.05 %
Vidyo1	-9.71 %	-31.15 %	-45.32 %
vidyo3	-2.20 %	-18.06 %	-34.65 %
Vidyo4	-0.11 %	-23.42 %	-38.04 %
Average BD-BR	**-3,24%**	**-27,06%**	**-41,88%**
Encoding Run Time	**~1077,38**	**~143,77**	**~4865,83**

Further, Tab. 2.9 shows that the HM and JM provide R-D enhanced performance according to their optimized versions x265 and x264 respectively. In fact, this is due to the fact that HM and JM implement all the building blocks of the standards.

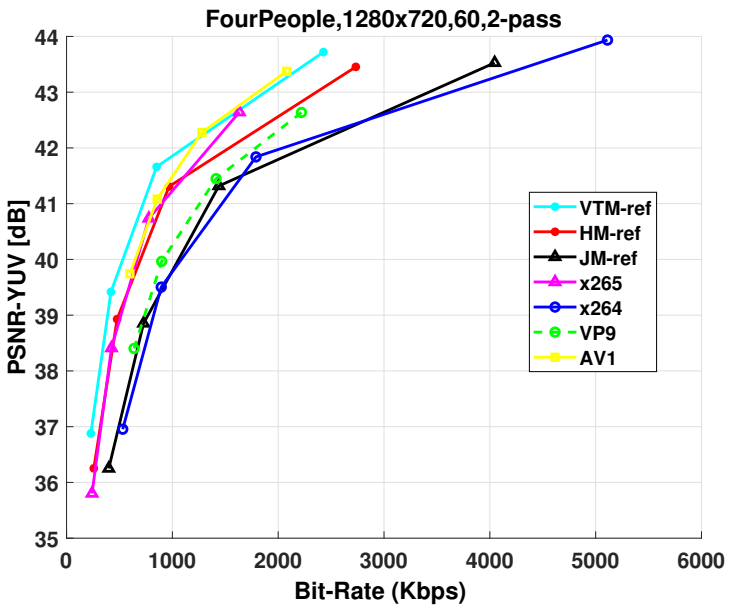
Performance evaluation of encoder implementations for second-pass encoding

This sub-section provides the outcomes obtained from encoding the test sequences with the second pass mode encoding. As mentioned before, the multi-pass encoding mode is only employed for industry-driven encoders i.e., AV1, VP9, x265 as well as x264.

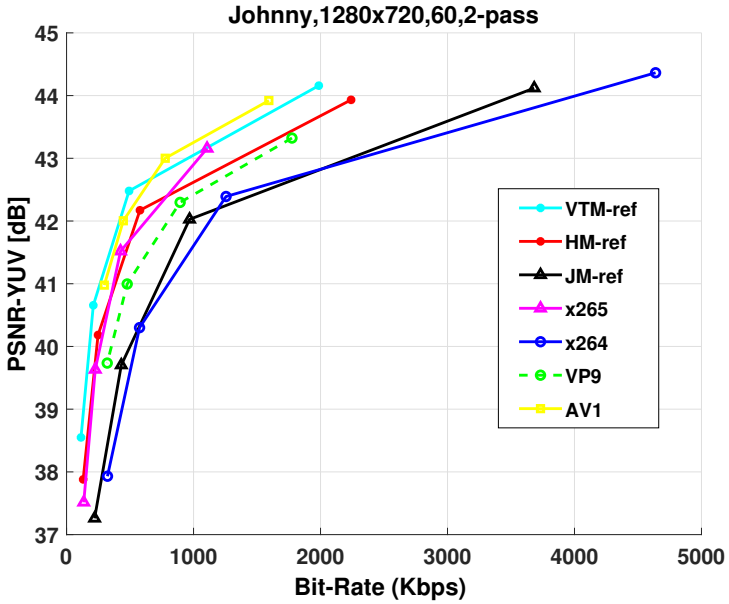
Figure 2.4 depicts the Rate-Distortion (R-D) curves for all the selected codecs for the second pass mode. According to the figure, as with the first pass, the same conclusions can be made regarding compression efficiency performance. All the graphs look similar to the first part except for slight changes regarding AV1 and VP9.

According to Tab. 2.10, the coding performance of HM for the second pass is substantially identical the that obtained in the first pass according to x265 and x264.

On the other hand, the performance of HM relative to VP9 in terms of bit-rate saving is reduced by 4.06%. It should be noted that the reduced performance of HM is significant with moderate motion



(a)



(b)

Figure 2.4 – Second pass rate-distortion curves for all codecs regarding class E and E' videos.

sequences as well as with Vidyo 4 sequences. However, this reduction is due to the R-D enhanced performance applied in the second pass for VP9.

In addition, we can see that the encoding run time of VP9 is 86.81 times faster according to its performance for the first pass.

Table 2.11 below, presents the average BD-BR saving of AV1 according to HM as well as to its predecessor VP9 for the second pass encoding mode.

Table 2.11 – Average BD-BR savings for the compared performance of AV1 encoder.

Sequence	**AV1 vs HM**	**AV1 vs VP9**
FourPeople	-7.99	-31.62
Johnny	-18.57	-41.64
KristenAndSara	-3.89	-31.12
Vidyo1	-9.06	-34.23
vidyo3	-25.58	-36.46
Vidyo4	-16.17	-36.46
Averages	**-13,54 %**	**-35,26 %**
Encoding Run Time	**~2,142**	**~308,01**

Table 2.12 – Average bitrate savings for the performance of the VTM encoder.

Sequence	**VTM vs AV1**
FourPeople	-16.7650
Johnny	-12.9496
KristenAndSara	-21.1937
Vidyo1	-15.5121
vidyo3	-11.5293
Vidyo4	-10.1459
Averages	**-14,68**
Encoding Run Time	**~0,62**

According to Tab. 2.11, when comparing AV1 to HM, we can see that AV1 provides gains in terms of bit-rate saving of about 4.53% relative to its first pass performance. However, this is particularly prominent for moderate motion video sequences as well as the high motion video sequence Vidyo4.

Further, it can be observed that in order to obtain the aforementioned gains, AV1 requires a slight overhead indicated by a factor of 0.1 in terms of encoding run time.

When benchmarking AV1 to its predecessor VP9, we can see that AV1 achieves a 35.26% bit-rate reduction. However, it is clearly seen, that the achieved coding performance is almost the same compared to that in the first pass. Moreover, Tab. 2.11 shows that AV1 is 308.01 times slower than VP9. Indeed, this is due to the fact that VP9 encoding run time decreases during the second pass encoding mode.

As shown in Tab. 2.12, the compression efficiency of VTM relative to AV1 in terms of bit-rate saving is reduced by 3.71% according to the first pass mode. Particularly, for Vidyo4 and moderate motion test sequences. However, we notice that AV1 is slower by a factor of 0.05 in terms of encoding run-time for the second pass.

Table 2.13 shows that, even though the second pass is enabled i.e., enhancing the R-D performance.

Table 2.13 – Summarized bitrate saving results for the second pass experiments

CODEC	VVC	HM	x265	JM	x264	VP9	AV1
VVC	/	-26,28	-28,27	-54,15	-56,65	-46,33	-14,68
HM	36,01	/	-3,24	-38.14	-41,88	-27,06	16,53
x265	39,93	3,49	/	-36,98	-41,47	-25,67	15,86
JM	120,28	63,70	60,10	/	-6,00	17,89	81,81
x264	133,05	73,94	72,34	6,51	/	23,86	86,97
VP9	86,97	37,65	35,15	-14,80	-18,85	/	53,88
AV1	17,43	-13,54	-12,73	-44,54	-45,93	-35,26	/

the VVC representative encoder (VTM) outperforms all the existing video coding software implementations. However, the performance of VTM in terms of encoding run time is almost the same as the first pass.

It is worthwhile to mention that, we notice an insignificant compression efficiency for the optimized versions of HM as well as JM i.e., x265 and x264 respectively according to the second pass encoding mode.

On the other hand, a significant increase in the coding efficiency for AV1 as well as VP9 encoders is observed for the second pass encoding. In addition, we can see that the encoding process of VP9 is faster when enabling the multi-pass encoding.

2.3.2 Case 2: Results and Discussion with one pass encoding mode

Compression Performance Evaluation

Based on Fig. 2.5, which describes the rate-distortion performance of class C videos, we can say that the overall obtained results support the claim stating that newly developed codecs have better coding efficiency compared to their competitors. However, the performance of a codec varies depending on the specific characteristics of the video. For instance, the gap between the R-D curves of BasketballDrill is larger than that for Racehorses. The same is applied to the other different video sequences. Moreover, for some videos, not only there exist a gap difference but some codecs outperform others for a specific video content. For BasketballDrill, BQmall and Racehorses, we can see clearly that AV1 outperforms HM. Conversely, for PartyScene, AV1 provides its least performance compared to HM.

For the detailed performance of all the codecs, the following section provides statistics in the form of videos.

Table 2.14 provides the detailed outcome for the compared performance of HM with its optimized versions as well to its predecessors JM, and x264 for both “medium” and “ultra-presets”. Moreover, in the table, the performance of HM with VP9 is illustrated. Based on the data, we can see that

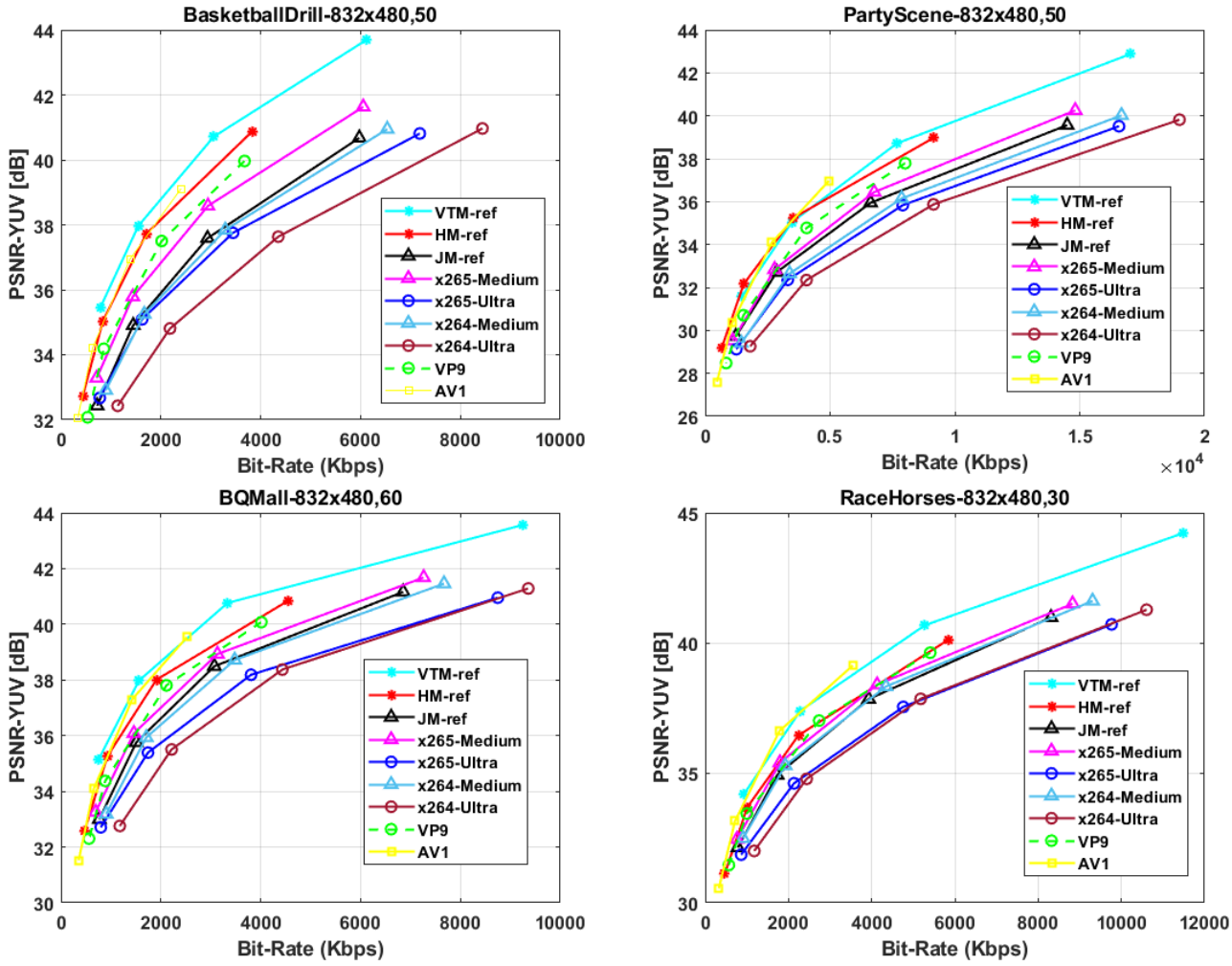


Figure 2.5 – Rate distortion curves for all codecs regarding class C video test sequences.

HM outperforms its optimized versions for both modes. The enhancement is represented in terms of bitrate saving of about 21.69% and 44.44% on average according to x265 and x265-Ultra respectively. Note that, high-motion sequences are responsible for the overall large gain. When we compared the compression performance of HM to VP9, we notice that VP9 provides a bitrate overhead for the same visual quality. For this case, the least performance of HM comes with sequences that contain low movement and moderate texture i.e., RaceHorses video.

The results from Tab. 2.14 support the claim stating that HEVC provides half-bitartrate reduction regarding H.264. However, the results can be observed only for x264-Ultra. In contrast, for both JM and x264-medium we can see that HEVC decreases only 31.25% and 35.46% of the bitrate respectively. Indeed, HM is seen to provide high gains for high-motion sequences regarding VP9, JM, x264 and x264-Ultra.

According to the fact that AV1 is designed as a competitor to HEVC, in Tab. 2.15, we present the compared performance of AV1 to HM. Moreover, AV1 is compared to its predecessor VP9. From

Table 2.14 – Average bitrate savings of the HM encoder (U refer to Ultra in X264 AND X265)

Sequence	**HM vs x265**	**HM vs x265U**	**HM vs VP9**	**HM vs JM**	**HM vs x264**	**HM vs x264U**
BasketballDrill	-27.8	-48.4	-21.1	-43.3	-46.1	-61.6
BQMall	-20.6	-45.2	-15.3	-29.4	-34.4	-53.5
PartyScene	-30.6	-48.1	-26.4	-35.8	-45.3	-56.7
RaceHorses	-7.8	-36.1	-5.2	-16.5	-16.1	-40.2
Average	-21.7	-44.4	-17	-31.2	-35.5	-53

Table 2.15 – Average BD-BR savings for the AV1 encoder.

Sequence	AV1 vs HM	AV1 vs VP9
BasketballDrill	-2.82 %	-24.79 %
BQMall	-8.25 %	-22.49 %
PartyScene	5.60 %	-23.45 %
RaceHorses	-20.72 %	-25.11 %
Average BD-BR	-6.55 %	-23.96 %

the table, we can see that AV1 outperforms both HM and AV9 in terms of compression efficiency. The gains are 6.55% and 23.96% bitrate saving for HM and VP9 respectively. AV1 outperforms HM, especially for low motion and moderate video sequence RaceHorses. In contrast, for VP9 we notice the same gains are obtained regardless of the video type.

Table 2.16 summarizes all the bitrate savings for all the video codecs. From the table, we can say that each newly born codec outperforms its predecessor in terms of coding efficiency. Indeed, the VTM provides the highest gain followed by AV1.

However, for the reference software implementations, HM and JM, we see that they clearly provide

Table 2.16 – Summarized bitrate saving for all codecs regarding mobile applications.

	VTM	**HM**	**x265**	**x265U**	**JM**	**x264**	**x264U**	**VP9**	**AV1**
VTM	/	-13.39	-32.66	-52.48	-40.61	-43.72	-57.78	-26.65	-3.93
HM	16.60	/	-21.69	-44.44	-31.25	-35.46	-53.03	-17.01	8.18
x265	49.23	29.25	/	-28.98	-12.27	-17.73	-39.28	7.28	39.97
x265U	112.34	81.36	40.98	/	23.21	15.54	-15.71	50.01	95.87
JM	71.10	48.40	14.45	-18.45	/	-6.38	-31.30	22.71	60.71
x264	80.97	59.96	22.22	-12.66	7.16	/	-26.12	32.49	73.58
x264U	142.44	118.52	65.80	19.49	45.74	35.65	/	82.38	140.91
VP9	36.82	21.55	-6.72	-33.29	-18.00	-23.82	-44.66	/	31.54
AV1	4.74	-6.55	-28.54	-48.91	-37.38	-41.96	-58.13	-23.96	/

Table 2.17 – Encoding and Decoding time ratios for low delay configuration setup.

Codec	Encoding	Decoding
VTM	15.30	2.93
AV1	6.80	0.70
VP9	0.25	0.20
x265 medium	0.12	/
x265 ultra	0.07	/
x264 medium	0.025	/
x264 ultra	0.014	/

significant bitrate saving in comparison to their optimized versions. This is due to the fact that the reference software implements all the building blocks introduced in the standards.

For the x265 and x264, we see that they clearly decrease the total bitrate by 28.98% and 26.12% compared to their "Ultra" implementations respectively.

In fact, for all the codecs discussed in this subsection, we can see that they offer different degrees of bitrate savings. However, the difference is related to several factors such as the test sequences and the competitor codec implementation used. Indeed, these coding performance improvements come at the expense of increased computational complexity during the encoding stage. The following section delves further into the complexity issue represented in terms of encoding run time.

Codecs run-time encoding performance

Fully-fledged implementations of HM and JM are computationally complex and may not be suitable for real-time applications or devices with limited processing power. In such cases, researchers may choose to use an optimized version of HM, such as x265 or x264, which are open-source software libraries that implement HM and JM, respectively. These libraries are optimized for speed and efficiency, making them more suitable for use in real-world applications.

It's worth noting that while x265 and x264 are widely used and respected in the industry. Ultimately, the choice of which codec to use will depend on the specific needs of the application and the trade-offs that the researcher is willing to make in terms of quality, complexity, and performance.

To ensure a fair comparison between different codecs, we eliminated the use of parallel processing and limited multithreading. By specifying the –threads 1 and –lookahead-threads 1 options for x264 and the –no-wpp and –frame-threads 1 options for x265, we are effectively telling these codecs to use only a single thread for encoding and decoding, which can help to level the playing field and make it easier to compare the performance of different codecs under similar conditions.

Table 2.17 provides the encoding and decoding normalized results according to HM. The values represent the run time for the codecs. Indeed, the data is compared to that of HM and the factor

difference is provided. In other words, we used HM as an anchor to calculate run time differences compared to other codecs.

According to Tab. 2.17, VTM and AV1, which are the newest coding standards, provide the least performance in terms of encoding and decoding run-times. This is due to the fact that several new time-consuming techniques are applied in order for these codecs to achieve the aforementioned compression enhancements. Therefore, we can see that they provide improvements in the compression at cost of run-time increase by a factor of about 15.30 and 6.80 relative to HM respectively.

For the industry-driven codecs x265 and x264, the lowest coding performance achieved leads to lowering the encoding runtime. A run time decrease by a factor of 0.07 and 0.014 related to HM is seen for x265 and x264 "ultra" respectively. While for the "medium" versions we see that they lower the time by a factor of 0.12 and 0.025 for x265 and x264 respectively compared to HM. Even though we disabled the use of multithreading for x265 and x264, they still provide, enhanced performance in terms of run time compared to the fully-fledged implementations.

Note that the VTM takes a longer time to encode the same video compared to AV1. This time increase comes at a bitrate saving of about only 3.93%. Consequently, we advise you to work with AV1 to test low-delay videos rather than VTM.

Based on the obtained results, we can conclude that the used versions of AV1 and VTM are not suitable for real-world scenarios, especially with applications that have time constraints. Conversely, x265 "medium" is seen to be the best option in such a scenario. This is due to the fact that is provide acceptable compression performance and encoding/decoding time.

The second codec that we recommend using in real-time applications is x264 "medium" rather than x265 "ultra". This is due to the fact that x264 medium provides a bitrate saving of about 12.66% compared to x265 "ultra" while lowering the run time by a factor of 79% compared to x265 "ultra".

In conclusion, to select the appropriate codec, it is important to have a thorough understanding of the needs of the target application, the implemented methods in each standard, and the balance between compression performance and computational complexity. Indeed, this knowledge is crucial for making the best choice. However, it is important to note that compression efficiency is just one aspect of video coding performance. In the next chapter, we will examine the error resilience of the different video coding standards, which is another important factor to consider when selecting a video coding system.

2.4 Conclusion

In conclusion, our evaluations of compression efficiency for the different video coding standards have revealed that each newly developed codec has better coding efficiency compared to its competitors or predecessor. The results can be seen through the two types of studies i.e., multi-pass encoding and one-pass encoding. However, Multi-pass encoding can be computationally intensive, as it involves

running the encoding process multiple times and analyzing the video in detail. As a result, it may not be suitable for real-time applications that require fast encoding speeds. While Single-pass encoding involves adjusting the encoding parameters in real-time as the video is being encoded, without the need for multiple passes. This results in faster encoding speeds. In addition, the one-pass encoding achieves a good level of compression efficiency. Consequently, one-pass encoding proves to be the appropriate mode for real-time applications such as video surveillance. These findings provide valuable insights into the relative performance of the different standards and can inform the selection and design of video coding systems. The research on the evaluation of these codecs has demonstrated their effectiveness in providing improved compression efficiency and their wide use in various applications. Indeed, numerous points can be summarized:

— The multi-pass mode could not provide R-D enhancements for the first work as was expected and stated in [69]. We can say this can be due to the used encoding parameters of low delay.

— One-pass mode is more suitable for real-time applications compared to multi-pass mode.

— Based on our experiment, x265-medium is the best software implementation among other software implementations. As a representative of HEVC, we conclude that HEVC is the best suitable standard for our case. In fact, it provides good tradeoff between R-D performance and complexity.

— If additional time decreasing is desired in the system, we recommend the use of x264 “medium” preset rather than x265 “ultra-fast” [63].

Based on this evidence, first, we chose the HEVC standard as our main standard to study and to enhance its error resilience. Second, we chose to study the error resilience of HM and x265 in an erroneous environment.

Chapter 3

Error Resilience Evaluations for HEVC standard

3.1 Introduction

This thesis intends to develop an efficient video surveillance system that is robust to network errors and can be embedded in buses. We aim to transfer High definition (HD) and Ultra-HD (UHD) videos in real-time through wireless networks. Indeed, Real-Time video delivery over wireless mobile networks is affected by the chosen standard used to encode the video and the network used as a medium for transportation. Hence, it is important to consider both compression efficiency and error resilience when evaluating video coding standards, as both factors can have a significant impact on the overall performance of the video coding system. In the previous chapter, we studied the compression efficiency of different video coding standards. However, in this chapter, we are more concerned about the error resilience performance.

Note that, for wireless video delivery, a higher data rate could lead to a higher packet loss rate, thus degrading the perceived quality [1]. According to the fact that our data is exposed to significant packet losses, together with taking into account that our application is a delay-bounded real-time video delivery, our main contributions will be in the context of making the HEVC transmission robust to these kinds of constraints. Indeed, HEVC is seen to be convenient to convey HD video sequences. This is due to the fact that HEVC provides the best R-D performance regarding low delay systems compared to other coding standards and is more convenient for real-world scenarios. These conclusions were stated in the previous chapter of our conducted comparative studies. Moreover, it is due to the fact that HEVC is efficient for low-delay applications and low bandwidth networks as stated in [74]. As HM and x265 are representative of the HEVC standard providing different degrees of optimizations, in this chapter, we would like to study their error resilience performance. Firstly, we analyze the error resilience capabilities of each implementation and compare their performance under different error

scenarios. This will inform the selection and design of video coding systems. Next, we will test the error resilience efficiency of different new methods introduces to HEVC. For instance, the susceptibility of HEVC bitstreams to motion information loss and its relation to TMVP. By the end of this chapter, readers should have a clear understanding of the error resilience capabilities and limitations of HM and x265 codecs. To this end, before delving into the detailed comparative study, we would like to first describe the different techniques used to introduce losses to an HEVC bitstream.

This chapter is organized as follows, first, we describe different ways on how video transmission and video evaluation are performed. After that, we provide a comparative study in terms of error resilience for HM and x265. Finally, we provide general studies related to HEVC methods regarding their efficiency in error-prone channels.

3.2 Video transmission platforms

Internet networks are packet-switched networks based on IP which is a best-effort protocol. Generally, these IP channels are hostile to network errors as they comprise wireless mediums. Due to the nature of these channels, real-time video transmission is subject to various numbers of constraints namely: delay/jitter, fluctuating bandwidth and packet loss issues. Indeed, packet loss is a critical issue of IP unreliable networks, it is caused mainly by numerous reasons such as congested routers, incorrect routing design, obstacles encountered (interferences), rapid change of topology, ..., etc. Due to the hardware resource limitation and higher prices of the network equipment. Moreover, to understand the behavior of such unfriendly networks in such constrained conditions, we are imposed to make computer-based simulations. Consequently, this section discusses different ways to make video transmissions, and talks about how the channel impairments are modeled.

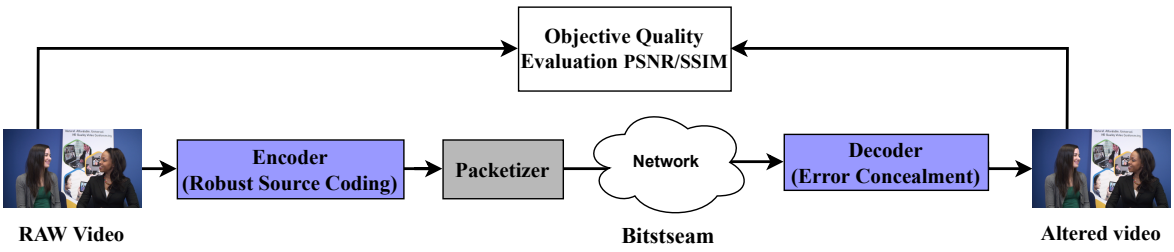


Figure 3.1 – General video evaluation framework

There are several methods and techniques that can be used to evaluate the error resilience of different video coding standards. These methods can include both simulations and real-world tests of the video coding systems under different error conditions. In Fig. 3.1, the general video evaluation framework is depicted. Note that after encoding the video, it will be fed to the error-prone network. After that the decoder will conceal the lost frames and reconstruct the video as an altered version. Next, we compared the original video and its altered version objectively. However, the network impairments can

be modeled using several ways as discussed in the following section.

3.2.1 Simulation based frameworks

Simulations are used to evaluate the error resilience of video coding standards in a controlled and repeatable manner. In simulations, the error conditions can be carefully controlled and varied to assess the performance of the video transmission under a wide range of error conditions and different scenarios. However, error conditions that can be simulated or tested include:

— ***Bit errors:*** These are errors that occur when a single bit of the video data is transmitted incorrectly. Bit errors can be introduced by noise or interference in the transmission channel.

— ***Frame loss:*** This occurs when one or more frames of the video data are lost during transmission. Frame loss can be introduced by errors in the transmission channel.

— ***Packet loss:*** This occurs when one or more packets of the video data are lost during transmission. Packet loss can be introduced by errors in the transmission channel, or by congestion in the network. Note that, a network packet can include one frame or several frames depending on the level of compression introduced.

Indeed, the common tools for simulating error conditions include network simulators, error injectors, and data loss simulators.

Hexadecimal editors: Ultra-Edit Text Editor [81] can be used as an error injector. You can manually modify the actual bitstream data after opening the bitstream in a Hexadecimal format. Not that, a prior knowledge the bitstream structure and the high-level syntax of video standard is crucial in order efficiently address a specific portion of the bitstream.

Nal Unit loss Simulator [82]: is a data loss simulator which allows users to introduce losses to the compressed video data. Usually, it takes the actual Nal Unit byte stream, applies the losses to it and creates an altered version of the input. Indeed, the simulator drops frames packetized as Nal Unit byte streams. Moreover, it can be configured to simulate different error rates with different predefined loss patterns e.g., 1%, 3%, 5% and 10%. The loss patterns include random and burst losses. These patterns were taken from sending video traffic through a real-world IP network. For more details about how the error patterns were generated refer to [83]. For a C++ implementation of the simulator, refer to [84]. In this repository, we provide a C++ version that supports the use of HEVC standard.

Using generated packet loss patterns: Packet loss models are mathematical models that are used to represent the loss of packets in a network. With the help of a loss model, we can generate a packet loss pattern in the form a text file. The text file is then used to apply the losses to the bitstream in order to mimic the wireless channel effect.

There are several different types of packet loss models, including:

— *Gilbert-Elliott model [85]:* This model allows for the representation of two different types of packet loss: an "on" state, in which packets are lost with a high probability, and an "off" state, in which packets are lost with a low probability. This model is more realistic than the other loss models. However, it assumes that packet loss occurs independently. A MATLAB implementation of the code to generate Packet Loss Patters using Gilbert Elliot Model can be found here [86].

Network simulators: Another way to simulate error conditions is to use transmission channel simulations to mimic the effects of different types of errors or losses on the transmitted video data. This can be done using tools such as network simulator 3 (ns-3) [87] and LTE-Sim [88], which can introduce errors or losses to the transmitted data based on specified error rates or patterns.

Simulation frameworks that evaluate the end video quality are crucial in order to use ns-3. EvalVid [89] is a tool that can be used to evaluate the performance of video coding standards, such as HEVC (High Efficiency Video Coding), in terms of their error resilience. EvalVid is designed to simulate the transmission of video data over a real or simulated network using ns-3. Figure 3.2, shows the EvalVid framework use case. Note that, EvalVid was designed for H.264 and older standards and can be used with any network simulator. In order to used Evalvid for HEVC, the source code has to be changed.

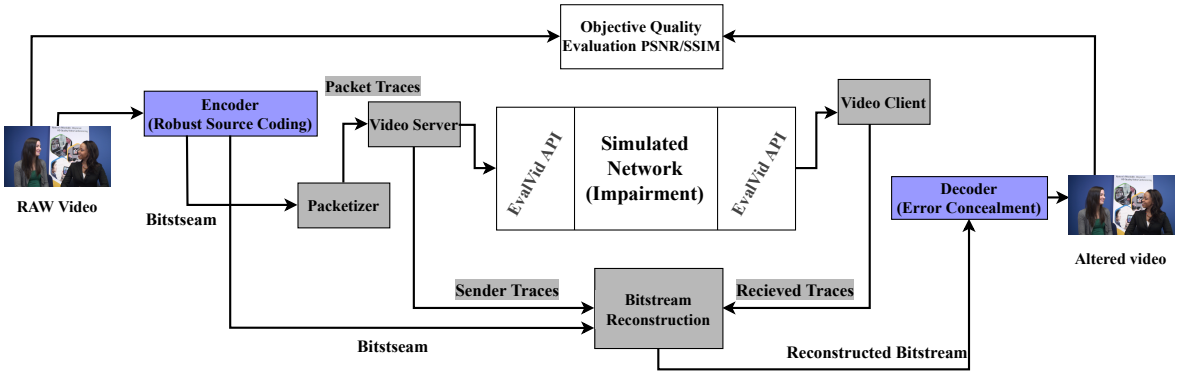


Figure 3.2 – Eval-Vid, ns-3 based framework for video quality evaluation.

LTE-Sim: LTE-Sim is an open-source free framework that simulates 4G/LTE network, it is developed by G. Piro and F. Capozzi [90]. To make the simulation more realistic, LTE-sim provides a simulation of the Evolved Universal Terrestrial Radio Access (E-UTRAN) as well as the Evolved Packet System (EPS). The following paragraphs reveal a step-by-step simulation for H.264 and HEVC video transmission with LTE-Sim.

After building the source code of the simulator under a ubuntu operating system, Traces such Frame_index, frame_type, Frame_time_stamp (ms), Frame_size (bytes) should be extracted from HM or JM. In fact, for test purposes you can use traces located within the simulator. After dealing with the traces, start the simulation by typing the following command.

./LTE-Sim SingleCell radius nbUE nbVoIP nbVideo nbBE nbCBR sched_type frame_struct speed maxDelay videoBitRate

./LTE-Sim SingleCell 1 20 0 1 0 0 1 1 3 0.1 128

This command specifies the parameters for a video transmission through an LTE network with 1 km single cell radius and 20 user equipment that have a speed of 3 km/h.

When the simulation is finished, LTE-Sim will provide the simulation results trace files. However, in order to help assessing the network, metrics such video packet loss rate, number users, delay, cell spectral efficiency and video throughput should be measured. The measurement is conducted through shell Linux script mentioned in [91].

Problems encountered: LTE-Sim was developed in 2011 to support H.264 standard. However, while working with HEVC several authors [92, 93] claimed that there exists a bug within the source code of LTE-Sim. The bug impedes the software from transmitting HEVC video traces. To work with HEVC, researchers in [93] propose to declare two specific variables as uint32_t instead of uint16_t. Indeed, we verified the proposed solution by following the steps in [93]. As a result, we were able to deliver an HEVC video trace through LTE-Sim. However, according to the fact that LTE-sim is developed for H.264, it is self-evident that it does not support RTP encapsulation for HEVC. Another problem is that, LTE-Sim has a limited documentation and it does not have any updates from the first published version.

In a nutshell, we would like to point that it does not allow the reconstruction of the video from the received trace files i.e., we have to develop external codes. Thus, we will not be able to calculate the PSNR.

3.2.2 Realistic testbeds

Real-world tests can also be used to evaluate the error resilience of video coding standards. In these tests, the video coding systems are tested under actual error conditions that may be encountered in real-world transmission. Real-world tests can be useful for evaluating the performance of video coding standards under more realistic error conditions, but may be more difficult to control and repeat than simulations.

There are several ways that real-world tests can be used to evaluate the error resilience of video coding standards. One approach is to transmit video data over a real-world network, such as a wireless or wired network, and introduce errors or losses to the transmitted data using tools such as NetEm. By adjusting the error rates or patterns, it is possible to test the performance of the video coding standard under different error conditions and assess its error resilience.

In our thesis we propose to work with the testbed illustrated in Fig. 3.3.

Figure 3.3 depicts the experimental setup that allows a client-server video transmission through an

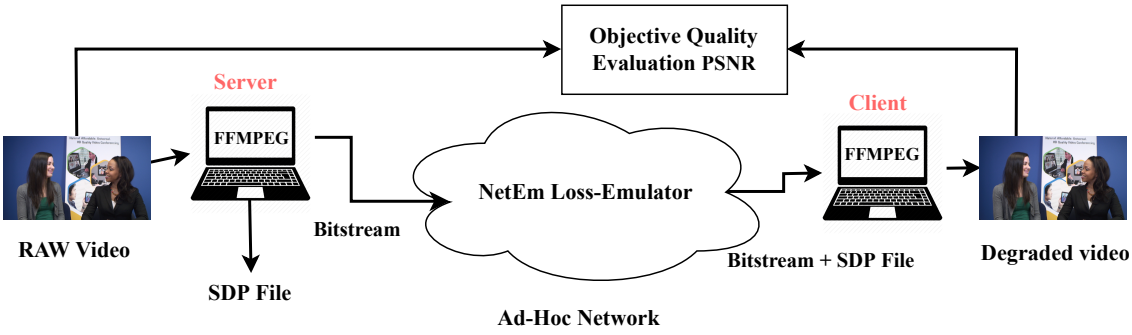


Figure 3.3 – FFMPEG based framework for HEVC video quality evaluation using RTP/UDP/IP stack.

Ad-Hoc network. The transmission is accomplished between two Ubuntu Laptops. The managed streaming is conducted as follows:

First of all, an Ad-Hoc connection between the server and the client must be set. We suppose that the Ubuntu systems are all set and contain the FFMPEG multimedia framework. Next, using FFMPEG, we should encapsulate the Nal Unit Byte stream into the RTP packetized format (RTP/UDP/IP) [94, 95]. An SDP file should be produced from the server. This SDP file is needed from the client side in order to receive the video data from the server. After the execution of the SDP file, the transmission will begin and the reconstruction of the bitstream will start. Once we receive all the bitstream chunks we use FFMPEG to generate the altered RAW video. Note that, a basic level of error concealment is used in the FFMPEG decoder. However, to make impairments to the network, we use NetEm. It is a software which introduces random packet loss patterns to the network interface of the laptop. Generally, 3%, 5%, 10% and 20% data rates are used as recommended by [96]. The actual used command are as follows:

On the server side you have to type the following commands:

Apply 5% errors to the network interface:

sudo tc qdisc add dev wlan0 root netem loss 5%

Here you have to change the IP address and the bitstream file name to start the streaming.

ffmpeg -re -i FourPeople_1280x720_60.hevc -c copy -f rtp rtp://10.42.0.84:1234 > silver.sdp

Then you copy the generated "silver.sdp" file to the client side for receiving the bitstream data and type the following command (in the same path containing the .sdp file):

On the client side:

ffmpeg -protocol_whitelist "file,rtp,udp" -i silver.sdp -c copy -strict -2 Alteredbitstream.hevc

For decoding the bitstream execute the following:

ffmpeg -i Alteredbitstream.hevc AlteredVideo.yuv

After decoding the received bitstream and reconstructing the altered RAW information, video evaluation in terms of objective metrics should be performed at this stage. For instance, calculating PSNR between the Reference video and the altered video.

3.3 HEVC implementations Error Resilience in error-prone environments

As mentioned in the previous chapter, HM and x265 are HEVC software implementations that provide different coding performances. In fact, x265 is the most used HEVC implementation in real-world applications as it provides timely video transmission compared to HM [73]. However, HM provides the same video quality as x265 with a bit rate decrease of about 20 percent [63]. According to [7], HEVC bitrate saving compared to previous standards such as H.264, leads to decreased error resilience performance in error-prone conditions. This is due to the fact that the network packet may contain more HEVC video data than that of H.264. Hence the loss of double the amount of data will lead to severe quality degradation. Consequently, by applying the same theory, we can assume that x265 is more resilient to network errors compared to HM. In the literature, researchers compare the compression efficiency of different codec implementations [63, 97, 98]. However, for error resilience, usually research is conducted to compare different video coding standards [7, 3, 1]. In this work, our goal is to study the error resilience of different software implementations of the same video coding standard in an error-prone environment.

3.3.1 Methodology

To conduct our work, we used the resulting bitstreams from the comparative study with one-pass encoding mode. Refer to chapter 3, case 2: comparative studies with one-pass encoding. The used bitstreams were encoded using the low-delay configuration. In the previous work, we used class C videos which contain a variety of video content. However, to evaluate the error resilience of the HM and x265, first, we selected all the bitstreams of HM, x265-M “medium” and x265-U “Ultra” for different QP values. Next, we sent them through an error-prone environment as shown in Fig. 3.1. We managed to do so by utilizing Nal Unit Loss Simulator. Indeed, we used 1%, 3%, 5% and 10% loss patterns. After that, we decoded all the altered videos using the HM decoder. Note that, frame copy error concealment [99] was used to copy the last correctly received frame in the event of frame loss. For the video quality evaluations, we used objective metrics. Indeed, PSNR and structural similarity index (SSIM) [100] were used to measure the quality of the decoded video.

3.3.2 Results and Discussion

First, we would like to provide an overview of the compression efficiency of the selected software implementations. Figure 3.4, provides rate-distortion plots depicting the compression efficiency of HM, x265-M "Medium" and x265-U "Ultra-fast" codecs. As it is clearly seen, HM outperforms both of the optimized version in terms of compression efficiency. While x265-M outperforms x264-U. The actual bitrate saving values are provided in Tab. 3.1. According to Tab. 3.1, x265-U cannot be a representative for HEVC, as the latter is developed to provide half bitrate saving compared to H.264. Indeed, x265-U provides a bitrate overhead of about 80% compared to HM which is a fully-fledged HEVC implementation. However, x265-M yields the same visual quality as that of HM with only 29% bitrate overhead while conserving the real-time characteristics of video transmission.

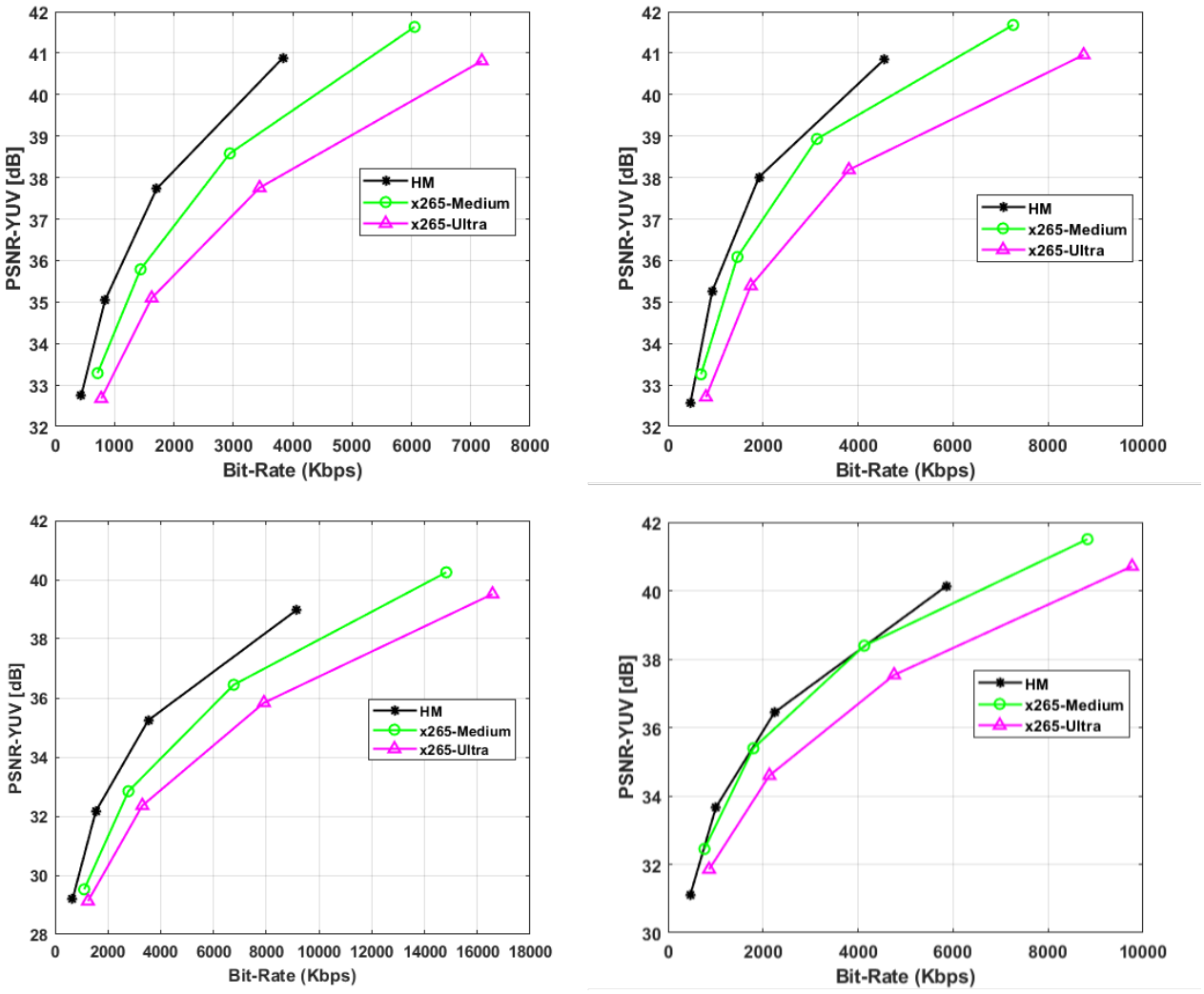


Figure 3.4 – Rate-Distortion curves for HM, x265-M and x265-U regarding class C video sequences: (left upper): BasketBall-Drill, (left lower) PartyScene, (right upper): BQMall, (right lower) RaceHorses

Table 3.1 – Coding performance comparison of HM, x265-M (Medium) and x265-U (Ultra)

CODEC/ ANCHOR	**HM**	**X265-M**	**X265-U**
HM	/	-21,69	-44,44
X265	29,25	/	-28,98
X265U	81,36	40,98	/

Table 3.2 – Error resilience performance of HM, x265-M and x265-U for 1% loss.

	HM	**X265-M**	**X265-U**	**HM**	**X265-M**	**X265-U**
	PSNR			**SSIM**		
BasketballDrill	28.5492	25.9405	26.3774	0.930725	0.9263	0.915675
BQMall	19.692625	21.157225	21.03135	0.805675	0.8429	0.82995
PartyScene	22.8789	22.6703	20.70705	0.79855	0.82685	0.751525
RaceHorses	19.345725	19.515225	18.263825	0.79715	0.804775	0.76155
Average	**22.6166125**	**22.3208125**	**21.59490625**	**0.833025**	**0.85020625**	**0.814675**

3.3.2.1 Error Resilience Evaluation

Tables 3.2 and 3.3 present the quality of the end video compared to the original sequences. The quality is represented in terms of PSNR and SSIM values obtained from averaging four QP points for each video sequence i.e., QP22, QP27, QP32 and QP37. For more detail about the SSIM metric refer to 4.1.

According to Tab. 3.2, on average, we can observe that the sensitivity to network errors of HM and x265-M bitstreams is the same for a 1 percent loss of the total information. The result is noticed for both PSNR and SSIM values. However, there is only a slight difference. The gap difference between HM and x265-M is noticed for BasketballDrill. Indeed, when analyzing the error propagation graphs of BasketballDrill depicted in Fig. 3.5, we can see that the gap difference between the implementations is larger for low-bitrates (QP37). When comparing HM to x265-U we can see that HM provides better quality in terms of PSNR, in which the difference is by about 1 dB.

From Tab. 3.3, where the loss is 10%, we can see that HM provides better quality than x265-M by 0.6 dB in terms of PSNR values. While it provides enhanced quality by 1.14 dB compared to x265-U. Regarding SSIM, the same conclusions about the gain of HM compared to the optimized versions can be observed as well. However, small gains are seen for SSIM values between HM and x265-M with differences in the third digit value. while HM outperforms x265-U by 0.0438 in terms of SSIM.

Figure 3.6 shows the error propagation of the three implementations. Based on the PSNR plots, HM is clearly providing improved quality than both x265-M and x265-U. Conversely, SSIM plots show that for high bitrates HM and x265-M provide the same quality. while for low bitrates, all implementations provide the same error-resilient performance.

In general, based on the obtained results and under the chosen test conditions, we can say that HM,

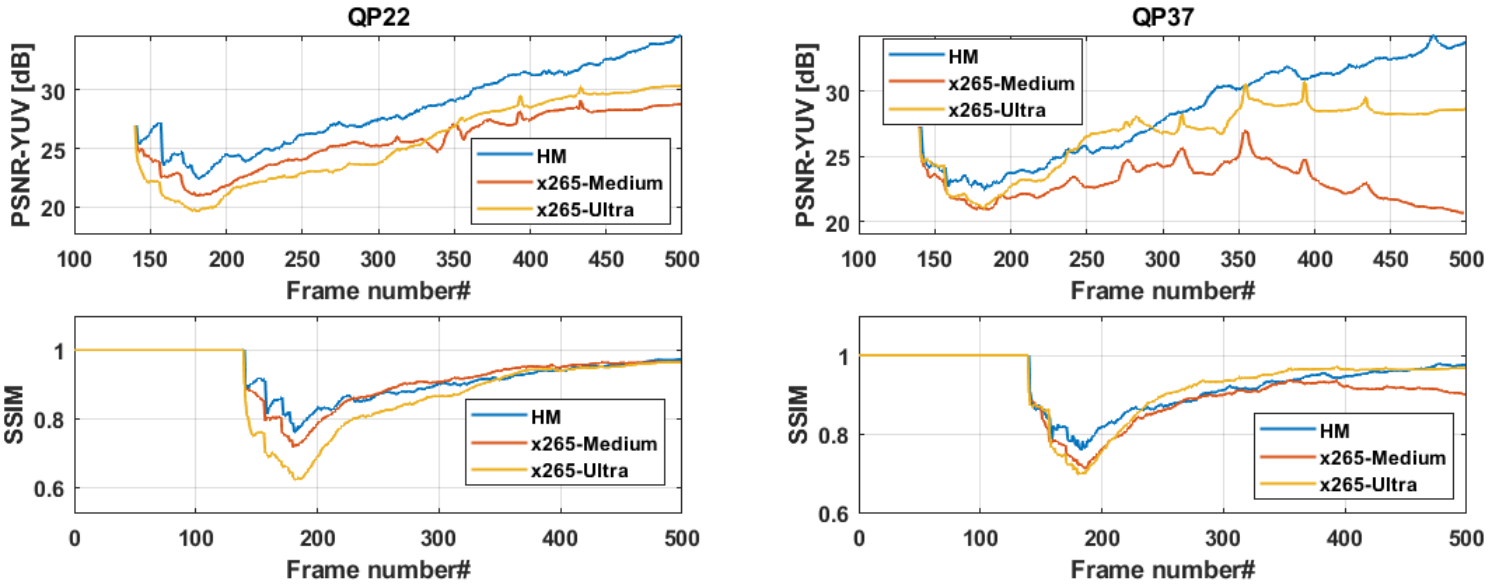


Figure 3.5 – Error propagation plots for BasketballDrill 1% loss

Table 3.3 – Error resilience performance of HM, x265-M and x265-U for 10% loss.

	HM	**X265-M**	**X265-U**	**HM**	**X265-M**	**X265-U**
	PSNR			**SSIM**		
BasketballDrill	21.7036	19.887275	20.02575	0.729975	0.70505	0.666625
BQMall	16.727725	16.50085	16.3163	0.6601	0.663375	0.6275
PartyScene	19.335875	19.04435	18.115275	0.6224	0.63715	0.579275
RaceHorses	18.77715	18.487725	17.5336	0.7008	0.706675	0.66365
Average	**19.1360875**	**18.48005**	**17.99773125**	**0.67831875**	**0.6780625**	**0.6342625**

x265-M and x265-U provide the same error resilience performance. According to Tab. 3.4, which presents the overall results, there exist slight differences between the PSNR values for different implementations. However, these changes are due to the nature of video content and the amount of motion each implementation applies when encoding. For the SSIM results, the difference is negligible. Hence, we can conclude that different implementations provide the same error resilience performance.

The reason behind the obtained results is that all different codecs implement the same HEVC core techniques for compression. In addition, the optimization part does not affect the error resilience performance as was the case with the compression efficiency performance. Another reason is that we did not use the RTP/UDP/IP packetization. This means that when making losses to the frames and concealing the spatial information at the decoder, we are only deleting the temporal information i.e., side effect of frame copy concealment. Hence, the same temporal information is deleted regardless of the size of the frame. On the other hand, when packetizing using RTP, we would have different RTP packets that include different number of frames. For instance, an RTP packet encapsulating HM bitstreams will contains more information than RTP packet containing x265-M bitstreams. Thus,

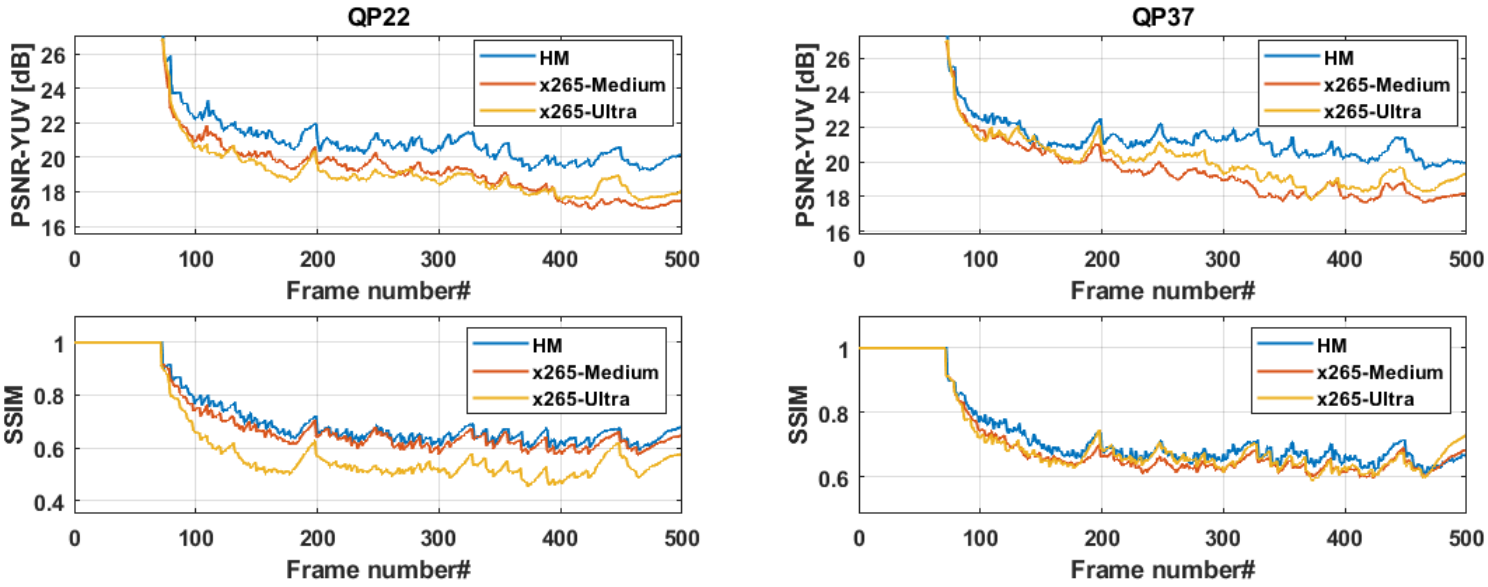


Figure 3.6 – Error propagation plots for BasketballDrill 10% loss

Table 3.4 – The overall error resilience gain of HM, x265-M and x265-U in lossy environments.

	HM vs x265-M		**HM vs x265-U**		**X265-M vs x265-U**	
	Delta PSNR	**Delta SSIM**	**Delta PSNR**	**Delta SSIM**	**Delta PSNR**	**Delta SSIM**
1%	0.2958	-0.01718125	1.02170625	0.01835	0.72590625	0.03553125
3%	0.68873125	-0.00598125	1.26704375	0.0266125	0.5783125	0.03259375
5%	0.7824	0.0014	1.2677125	0.0393125	0.4853125	0.0379125
10%	0.6560375	0.00025625	1.13835625	0.04405625	0.954125	0.043025
Average	**0.605742188**	**-0.005376563**	**1.173704688**	**0.032082812**	**0.685914062**	**0.037265625**

making HM less resilient compared to x265-M.

In a nutshell, our evaluations of error resilience for the different codec implementations have revealed that code optimizations do not affect the robustness of HEVC bitstreams to network errors. However, a different study that encapsulates the bitstreams into RTP packets might lead to different conclusions. Indeed, these findings provide valuable insights into the relative performance of the different codecs and can inform the selection and design of video coding systems.

Based on our compression efficiency and error resilience experiments, we can say that x265-M "Medium" is the best software implementation that can be used to transmit high and ultra-high videos in real-time. However, research to improve HEVC robustness to network errors is a major concern. In this regard, due to the fact that x265 and HM provide the same error resilience performance, and for test purposes, we chose to implement our error resilience method on top of HM. Moreover, it is due to the fact that HM implements all the building blocks of HEVC without any optimizations which will allow for deeper analysis.

In fact, studying the robustness of new HEVC-implemented methods to network errors should be performed in order to effectively enhance its error resilience. Hence the following section delves into analyzing the error robustness of different HEVC methods.

3.4 The error resilience performance of new methods introduced to HEVC

In this section, we would like to analyze the error resilience performance of new methods introduced to HEVC using HM as our main implementation. Indeed, after sending an HEVC bitstream through error-prone networks, the decoder may not decode certain frames. This depends mainly on the portion of the bitstream affected and the effectiveness of the Error Concealment (EC) technique to recover packet losses. According to [99], by default, the HM decoder provides a frame copy error concealment technique. This technique will help recover the loss of a complete frame. However, the HM decoder does not support slice copy concealment. Thus, in the event of a slice loss, the decoder will crush and stops working. Conversely, the HEVC decoder implemented within FFMPEG supports both frame copy and slice copy error concealment techniques. Therefore, before analyzing the error resilience efficiency of methods implemented within the HM encoder, we would like to study the limitations of HM decoder and FFMPEG decoder.

3.4.1 The effectiveness of different HEVC decoder implementations

Based on our experiments while working with HM and FFMPEG decoders we noticed that they provide different characteristics in lossy environments as shown in Tab. 3.5.

However, to select the appropriate decoder, some questions are needed to be answered:

— Due to the fact that FFMPEG provides additional error concealment methods, is it a good choice to rely on it instead of than HM?

— Are there any differences in the performance of the error concealment methods implemented within FFMPEG

— What are the main causes of the missing frames?

Table 3.5 – HM and FFMPEG decoder characteristics

Characteristics	**HM Decoder [65]**	**FFMPEG Decoder [101]**
End video quality	Good	Moderate
Decode all frames	Yes	No X
Frame Concealment	Yes	Yes
Slice Concealment	No X	Yes

To answer these questions, we conducted a study following the diagram illustrated in Fig. 3.7. In this experiment we encoded a 720p test sequence with different encoding parameters. First, we represented each frame as one slice. This means that each frame contains only one Nal Unit header.

Second, we divided the 720p frame into 40 slices. In other words, we put 6 Coding Tree Units (CTUs) into one slice and represented a whole frame as 40 slices. Note that each slice has its own Nal Unit header. To this end, we obtain two different representations of the bitstreams for the same coded video. We can say that we encapsulated the same video information with two different packetization formats. After that, we sent each of the aforementioned bitstreams separately via an error-prone environment. For each bitstream, we altered the information based on three different scenarios. First, we deleted the whole slice. Second, we only deleted the header of several slices and finally we altered to slices content. For each time, we decoded the resulting bitstream with a different error concealment technique as shown in Fig. 3.7. The results of the conducted work are provided in the following subsections.

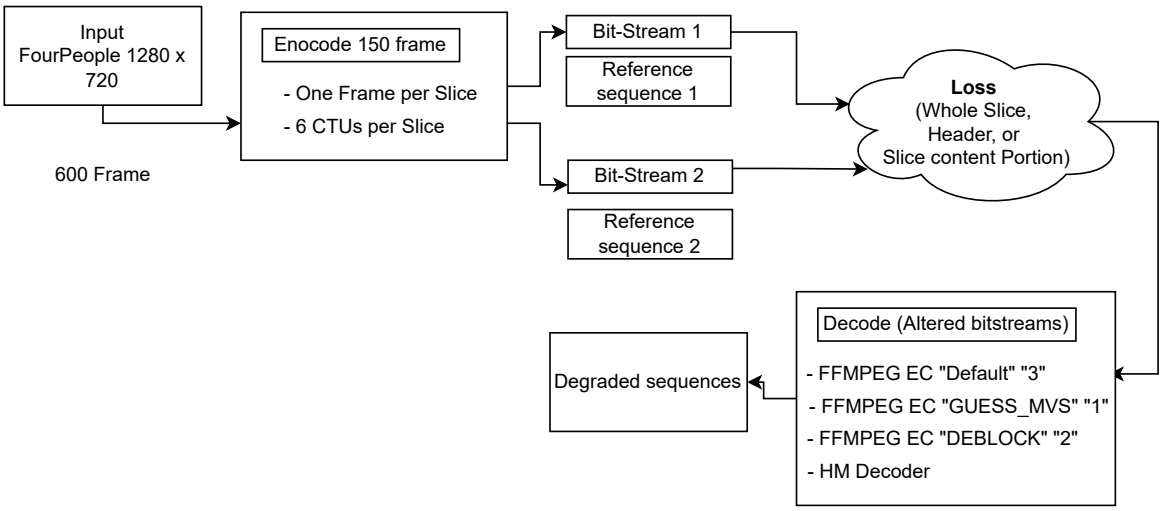


Figure 3.7 – Diagram of the conducted work for HM and FFMPEG error concealment.

3.4.1.1 Phase one: One slice Per frame

After dropping frame number 3 or the header of frame number 3, we noticed that FFMPEG was not able to conceal the lost frame. However, the frames following frame number 3 are successfully decoded for both cases. Note that error propagation is seen within the decoded frames. Figure 3.8 presents a snapshot of the immediate frame following the lost frame decoded using FFMPEG. Indeed, the decoded frames are of poor quality as depicted in Fig. 3.8.

Note that the same results are seen with all different error concealment techniques implemented within FFMPEG. However, when decoded using the HM decoder. We saw that HM conceals the lost frame POC 3 by replacing it with its preceding frame i.e., frame POC 2. Indeed, the frame following frame 3 is illustrated in Fig. 3.9. As can be seen, the frame provides high quality compared to that decoded

Figure 3.8 – Snapshot of frame 4 in the event of Whole Fame Loss/Header Loss decoded with FFMPEG all EC.

using FFMPEG. The only altered information is within the moving regions i.e., hands. Note that, HM crushes when given as an input the bitstream with deleted header of the frame number 3.

Figure 3.9 – Snapshot of frame 4 in the event of Whole Fame Loss decoded with HM frame copy.

When deleting a portion of the slice i.e., portion of frame number 3, we notice that HM stops working and does not decode any of the following frames. Conversely, when decoding using FFMPEG, it conceals the lost portion perfectly. Figure 3.10 Depicts a snapshot of frame number 3 after a successful decoding. In this case, FFMPEG decodes all of the subsequent frames in high quality.

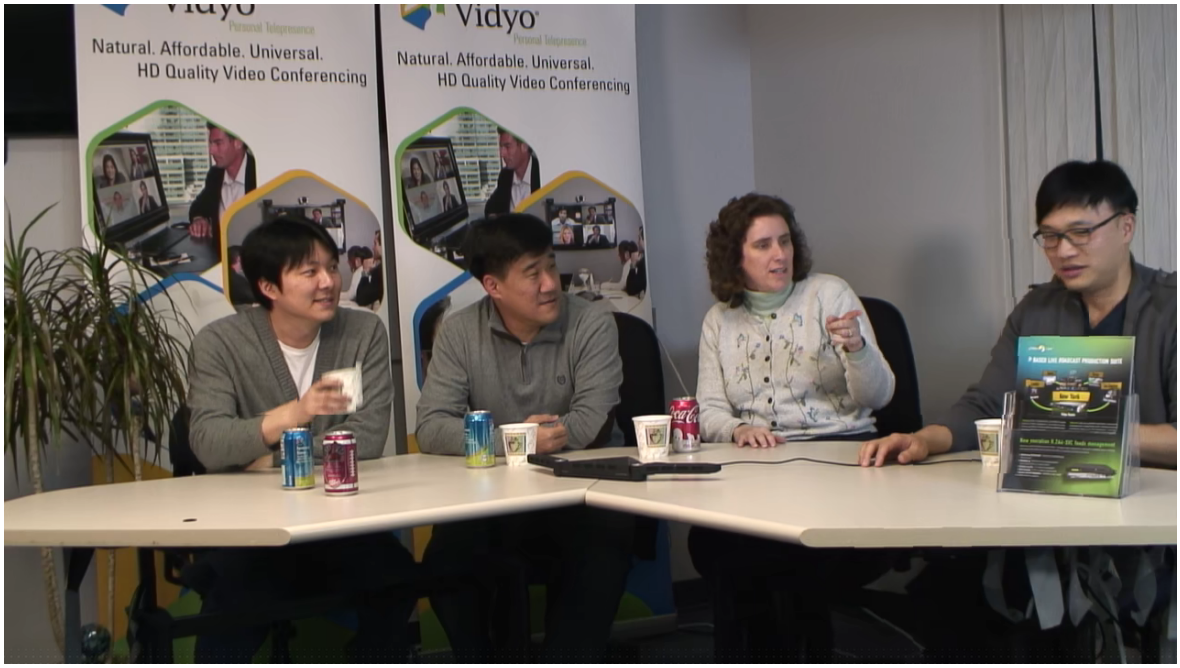


Figure 3.10 – Snapshot of frame 3 in the event of slice portion loss decoded with FFMPEG all EC.

Based on these experiments, we conclude the following point:

— FFMPEG doesn't drop packets when the loss appears in the content of the videos.

— Header loss in FFMPEG equals to whole Frame loss.

— All EC methods implemented in FFMPEG provide the same performance.

— HM crushes when any loss happens to the bitstream other than whole frame loss.

3.4.1.2 Phase two: 6 CTUs Per Slice

In this work, we represent each frame as a collection of slices each with its own header. This will help decode each slice independently in a lossy environment leading to an increase in the error robustness of the bitstream. The only drawback of this type of encoding is the bitrate overhead induced by the header information and the decreased compression efficiency across the borders of the slices. In our experiment, we introduced losses to two types of slices, "slice number one" and "slice number n". The first slice at the beginning of the frame is "slice number one" and any other slice is "slice number n". We distinguish between these two slice types due to the fact that they provide different performances as will be described next.

For FFMPEG decoding, when deleting the header of "slice one" within frame number 3 or the deleting the "whole slice", we noticed that the complete frame was dropped and not concealed. Indeed, the same results were noticed for all error concealment techniques implemented within FFMPEG. Figure 3.11, shows frame 4 snapshot which is the frame following the lost one.

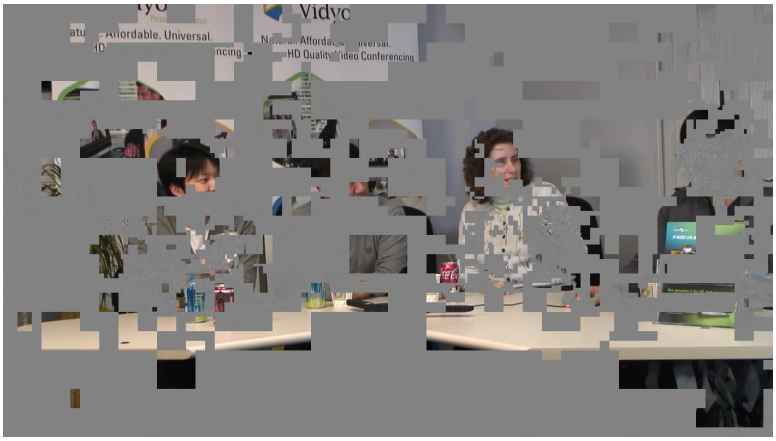


Figure 3.11 – Snapshot of frame 4 in the event of slice Header Loss/ whole Slice loss decoded with FFMPEG all EC.

As can be seen from Fig. 3.11. The loss of the first slice results in the same outcome as when we deleted the whole frame i.e., poor quality of the following frames. However, when deleting portion of the "slice one", the slice was concealed and the frame number 3 was successfully decoded. Figure 3.12, illustrated the frame with a portion of slice number one deleted.

Figure 3.12 – Snapshot of frame 3 in the event of slice portion content loss decoded with FFMPEG all EC.

In the case where we alter both header and whole slice loss for "slice number 2", we can see from Fig. 3.13 that the whole slice is deleted and replaced by a green block. The important thing to observe here, is that frame 3 was not completely dropped as was the case with "slice number 1".

Figure 3.13 – Snapshot of frame 3 in the event of slice Header Loss/ whole Slice loss decoded with FFMPEG all EC.

When deleting a portion of "slice number 2", we can see from Fig. 3.14 that the slice was concealed with some degrees of errors within the content. The second thing noticed in this case, was that different error concealment techniques implemented in FFMPEG provide different qualities.

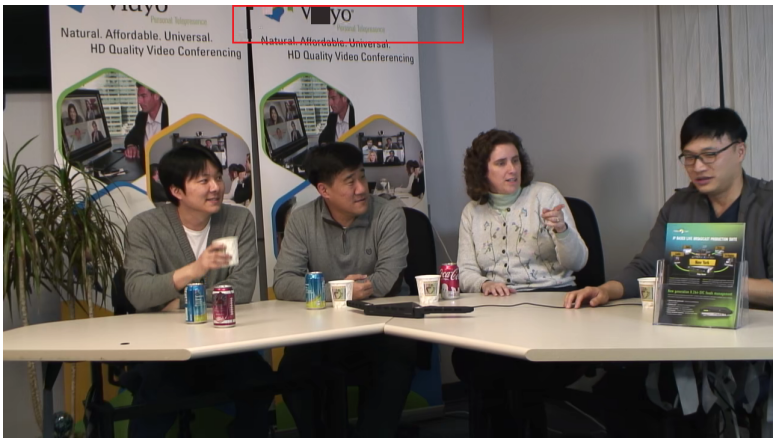


Figure 3.14 – Snapshot of frame 3 in the event of slice portion content loss decoded with FFMPEG all EC.

Based on the second phase experiments, we conclude the following point:

— FFMPEG drop frames when encounter loss in the header of the first slice (when one frame equals to several slices) or the header of the whole frame (when one frame equals one slice).

— When encoding using several slices, the first slice at the beginning of each frame should be protected in order to prevent from whole frame loss issue.

— FFMPEG does not conceals slices when the loss appears in the header.

— FFMPEG conceals slices or frames when the loss appears within the content.

— Different EC methods implemented in FFMPEG provide different performance when the lost slice is successfully concealed.

Consequently, we can say that FFMPEG can be utilized as a temporal solution that provides an acceptable quality. The only condition to provide good quality with FFMPEG is that you need to represent the video using a number of slices and to ensure that the loss appears within its content. Second, implementing slice copy error concealment within HM decoder will be the perfect solution to work with HEVC in a lossy environment.

3.4.2 Activating TMVP in HEVC under Erroneous channels

Temporal Moving Vector Prediction (TMVP) is a new method developed for HEVC to gain more compression efficiency compared to H.264 [3]. In this method the motion vector of the current block is predicted or copied from the temporal co-located block. Indeed, using TMVP will increase the temporal dependency between adjacent frames [3]. Consequently, the loss of motion information will affect the decoding process as it uses the motion information of the previous block to predict the current block. Based on this fact, we would like to study the effect of losing the motion information and the usage of TMVP on the reconstructed quality of HEVC decoded bitstream. To do so, we chose to work with BasketballDrill test sequence from class C and BQSquare from Class D. For simplicity, we only used 240 frames for these videos. However, for the prediction pattern, we used two different Low-delay P configurations, IPPP structure and IPPI32. The former includes only one Intra frame and the latter includes Intra-refresh cycles each 32 frames. Indeed, in our experiment, we made restrictions to motion compensation tools. First, we increased the temporal dependency between frames by activating the use of TMVP (default HEVC settings). Second, we deactivated the TMVP tool to study its effect on the error propagations. Note that, all the tests were conducted using HM 16.20 version with one QP value equals to 22. The conducted experiments are illustrated in Fig. 3.15.

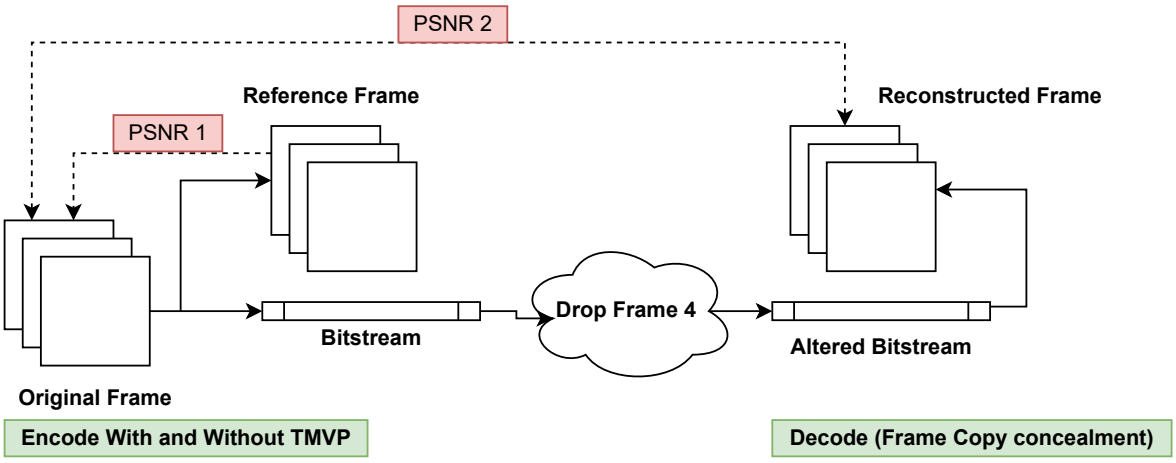


Figure 3.15 – Diagram for the conducted experiments with and without TMVP.

After calculating the PSNR between the original frame and the reference frames for both cases i.e., with and without TMVP, we noticed small decrease in compression efficiency for the frames encoded without using TMVP. Indeed, when analyzing the PSNR map between the two resulting bitstreams from both the experiments, we noticed that the difference in the compression efficiency is critical in the active areas. In other words, the PSNR reduction was mostly observed within the active areas or where there is a great number of moving vectors. Figure 3.16 shows the PSNR Map between two resulting bitstreams with and without TMVP. Note that the green color indicates small PSNR values while the pink color refers to high PSNR values. It is clearly seen that the green color is surrounding the blocks containing more moving vectors (red lines).

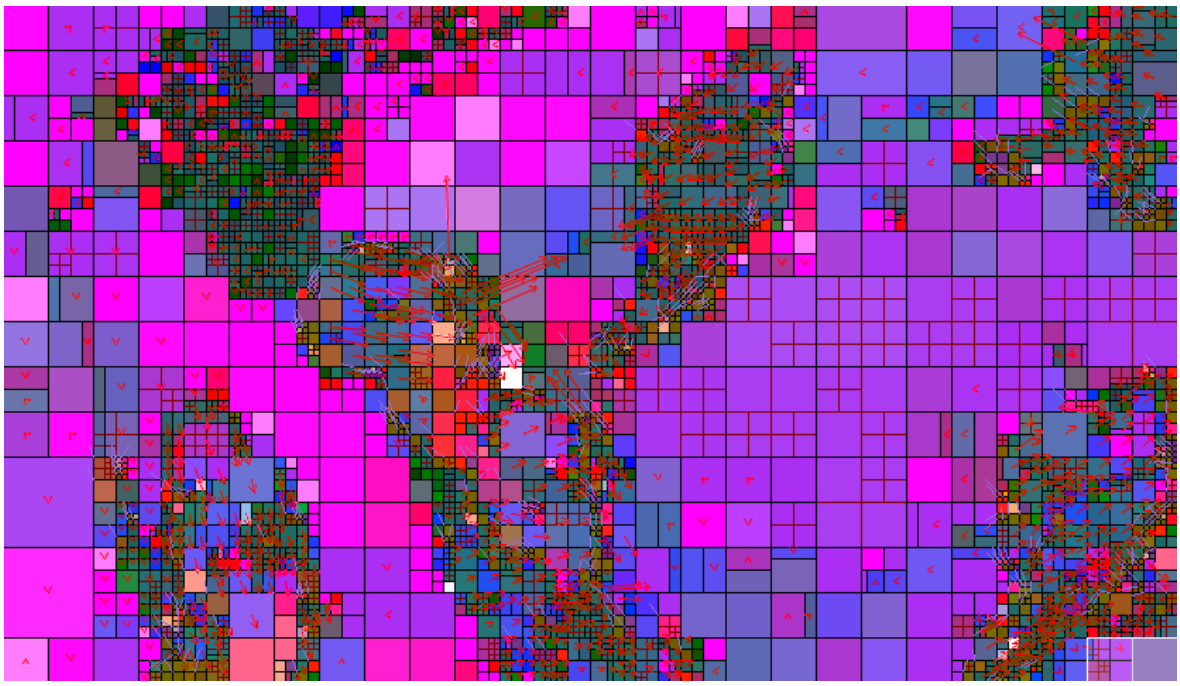


Figure 3.16 – PSNR map difference between the reference frame encoded using TMVP and the reference frame encoded without TMVP for BasketballDrill.

When dropping frame 4 for both bitstreams i.e., with and without TMVP, for all test sequences, we obtained the error propagation graphs depicted in Fig. 3.17. Note that, we used frame copy error concealment technique implemented within the HM decoder to obtain the altered videos. Using frame copy error concealment with zero motion vector compensation [102] will lead to the loss of the motion information only. However, based on Fig. 3.17, it is clearly observed that activating TMVP leads to a devastating effect in the event of frame losses. In contrast, deactivating the TMVP truncates the error propagation resulted from the high temporal dependency between frames imposed by TMVP. Based on this fact, we can conclude that deactivating TMVP will increase the error robustness of HEVC bitstreams in a lossy environment. Another way to truncate the error propagation is to use Intra refresh methods. This can be clearly observed in Fig. 3.17 as well. For instance, for BasketballDrill plot

without-intra-refresh, we can observe that the error propagates from the dropped frame until the final image. While with-Intra-refresh we can see that at the first appearance of an Intra coded frame, the error propagation is truncated and the final video quality is improved.

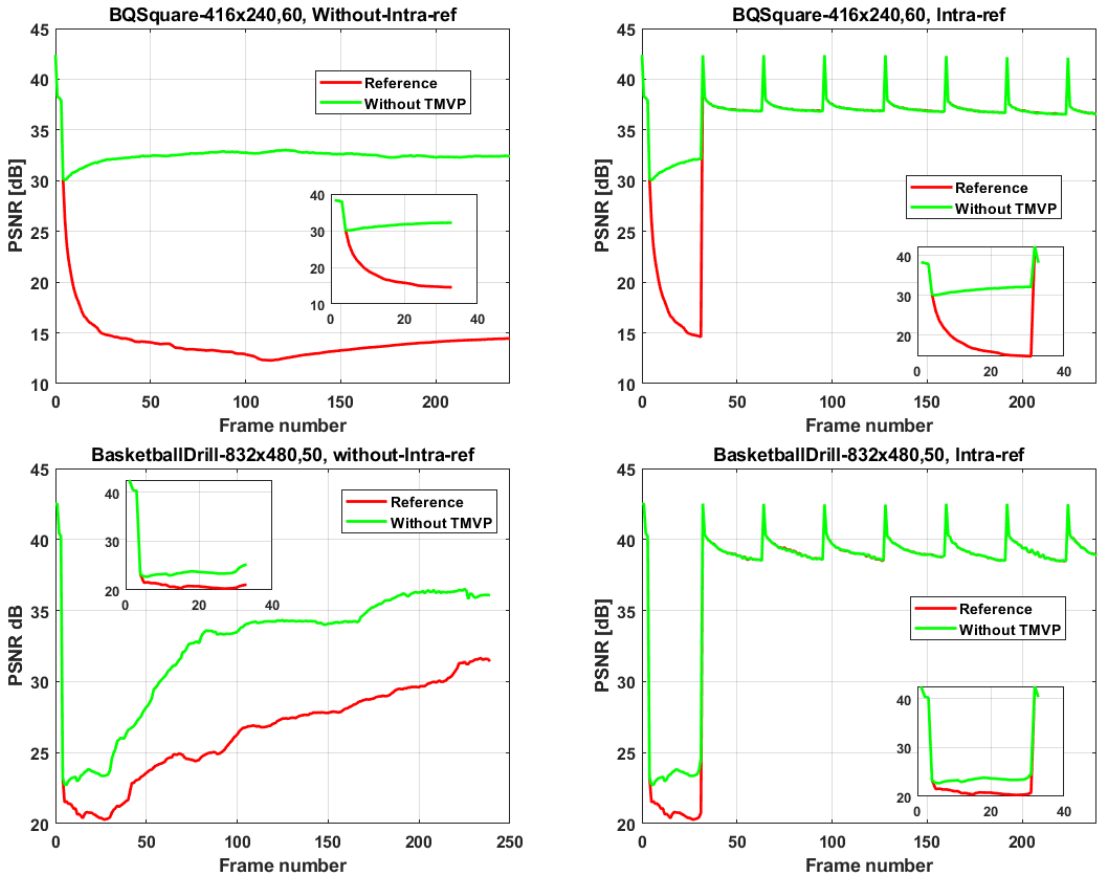


Figure 3.17 – Error propagation curves when frame 4 is lost.

In a nutshell, based on the conducted experiments, we can say that activating TMVP can improve the compression efficiency at the expense of a decreased error resilience performance in error-prone channels. Thus, deactivating the TMVP tool is the best solution to enhance HEVC error resilience performance.

3.5 Conclusion

In this chapter, we first provided different simulation and emulation platforms for modelling packet loss scenarios. After that, we conducted a comparative study of HM and x265 in terms of error resilience. Our study showed that both codecs provide similar error resilience performance in the event of packet loss. However, when comparing HM and FFMPEG in terms of decoding and error concealment, we found that HM decoding provides better quality compared to FFMPEG. Finally, we explored the use of TMVP in error-prone environments and found that it can be effective in improving the robustness of

video transmission under such conditions. Overall, our studies in this chapter provide insights into the performance and effectiveness of different methods for HEVC video transmission and evaluation in the presence of errors. Moreover, we conclude that error resilience is a major concern in HEVC-based systems. Thus, in the following chapter, we will present our proposed method for enhancing the robustness of HEVC bitstreams in error-prone environments. This method builds upon the findings of our studies in this chapter and aims to improve the error resilience of HEVC-based systems.

Chapter 4

Proposed Approach to Enhance HEVC Robustness in a Real-time video transmission

4.1 Introduction

The HEVC standard has gained widespread adoption due to its ability to achieve high levels of compression efficiency. Indeed, the increased efficiency is achieved through the use of advanced tools such as quadtree block structures, motion compensation and enhanced algorithms for parallel encoding and decoding. However, it makes the encoded bitstreams more susceptible to errors during transmission over wireless networks. This will result in the degradation of video quality due to temporal and spatial error propagations. To address this issue, various error resilience techniques have been proposed, including those implemented at the source level, at the channel level, and those that require interaction between the source encoder and decoder. In this chapter, we propose an approach that presents a reference picture selection-based error-resilient method for improving the robustness of HEVC-encoded bitstreams over hostile channels. The proposed algorithm aims to reduce temporal error propagation and improve end video quality by selecting reference pictures based on error status and decreasing the temporal dependency between adjacent frames. In fact, this chapter is based on our article that was published online [103]. At first, the chapter discusses the motivation behind the proposed work alongside the research gap that exists within previously published works. After that, section II provides a detailed description of the proposed algorithm. Next, section III describes the experimental setup. In this section, the used configurations and all the required tools to conduct our studies are elaborated. The implications of the results are discussed in section IV. Finally, section V concludes the chapter.

4.2 Motivation

To enhance the error resilience performance of the HEVC video coding standard, several works have been conducted targeting different techniques [25, 3, 26, 57, 58, 104, 105, 106]. The error resilience of motion compensation techniques was extensively evaluated in the following works. In Ref. [25], the authors studied the loss effect of motion information on temporal error propagation as well as on entropy decoding failure. Their work shows that although Temporal Moving Vector Predictor (TMVP) is crucial to improve the coding performance, it decreases the robustness of HEVC. In addition, it may cause immediate failure in the decoding of the successive frames as well as in the frame being decoded. To overcome this issue, they propose to periodically deactivate the use of TMVP at the frame level. Based on the previous work and to make HEVC more resilient to network errors, the authors in Ref. [3] reduce the temporal dependency at the block level by making restrictions on the use of temporal Moving Vector (MV) candidates. Moreover, they provide a comparative study between HEVC and H.264/AVC for the moving vector loss and its influence on the perceived quality. In Ref. [26] the authors proposed a method to convey the selected moving vectors as redundant side information to the decoder. This approach will help mitigate the issue of motion vector mismatch prediction that occurs in the decoder due to packet losses. Furthermore, it improves the motion-copy concealment method implemented at the decoder to enhance the reconstructed quality.

More recently researchers are interested in the Reference Picture Selection (RPS) algorithm as a solution for the error resilience problem of HEVC. In Ref. [57], a hierarchical-P RPS based on the hierarchical-P coding structure is proposed. In this algorithm, the authors implemented a bock-based Region of Interest (ROI) method that considers the moving regions as ROI. The extracted important areas will be protected against errors by an Intra-prediction encoding. Moreover, in this work, the authors manage to decrease the complexity of the non-ROI regions by limiting the CU splitting depth level. However, they yield a significant gain in terms of objective quality metrics compared to the conventional hierarchical-P RPS algorithm.

Other researchers [58], implemented an error resilience algorithm based on ROI. In their work, they represented the most active slices as an ROI. Nevertheless, after the ROI is extracted, it will be encoded using Intra Mode and thus decreasing the error propagation at the slice level. Frame differencing method, as well as the greyscale projection method [104], is used to conduct the ROI extraction. However, based on the fact the visual human system is sensitive to active areas as well as to the central part of images, they further split the non-ROI region. The quality of ROI regions and non-ROI regions is decrease gradually based on their importance. They yield, increased end video quality compared to primary works of Ref. [57, 105, 106]. Indeed, in both Ref. [57, 58], they impose Inter-frames to be encoded using Intra coding at the moving region in rate-constrained conditions. Note that, despite the high performance of HEVC rate control techniques in achieving target bitrates, they have not been accurately designed for Intra frames. Although the afore-mentioned Ref. [57, 58] works made

adjustments to the Rate control algorithm implemented in the HEVC, they still have bitrate fluctuation in their results.

All previously mentioned works aim to achieve better error resilience by considering the new technology of H.265/HEVC. However, the TMVP has been studied for different general loss concepts, while, the use of Temporal Moving Prediction to mitigate the effect of a long delay in the RPS algorithm due to network delay has not been investigated for lossy networks. In this chapter, we would like to exploit the use of new motion estimation techniques instead of Intra-refresh to mitigate error losses due to RPS feedback delay.

In order to provide an error-resilient system, here we propose an algorithm that is based on the reference picture selection technique (RPS) [107] implemented with different strategies to decrease the temporal dependency between adjacent frames. An example of a used strategy is to fully deactivate the use of temporal motion vector candidates. RPS is an interactive-based recovery technique, in this approach, the Decoded Picture Buffer (DPB) within the encoding loop will be updated regularly according to the error state signaled through a feedback channel from the decoder. With this approach, no error propagation is available due to transmission. Rather to inadequate error concealment methods as well as large delays. In other words, RPS suffers when the trip delay is high especially when the error concealment method is not adequate. More details about the proposed approach can be found in the following section.

4.3 Proposed solution

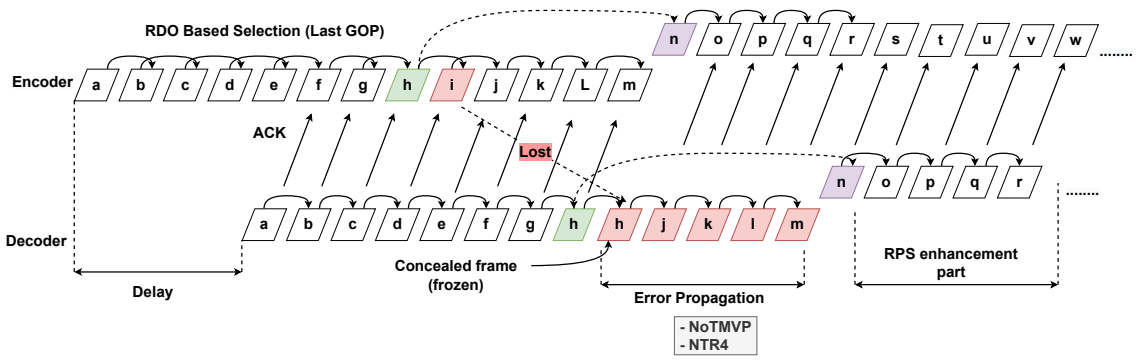


Figure 4.1 – The overall diagram of our proposed approach.

Our proposed system is depicted in Fig. 4.1. First, the video sequence is supposed to be encoded using the regular configuration pattern IPPP. This coding structure is useful for bitrate-constrained applications that require a trade-off between low delay and low complexity [108]. After that, the encoded bitstream will be sent to the decoder through the unreliable network. When the decoder detects an error, it will conceal the lost frame. To this end and after a small delay i.e., two frames delay

in a perfect condition, the encoder will be informed to change the reference pictures of the frames following the altered one in the decoder to avoid referencing it. Despite that, error propagation still occurs between the concealed frame and the frame with the changed references as shown in red in Fig. 4.1. As it is self-evident, the error propagation is due to the mismatch prediction of the temporal information by the error concealment as well as the round-trip delay.

Note that the process of changing the references in the encoder is guaranteed using the Reference Picture Selection (RPS) algorithm. However, to mitigate the effect of error propagation found within RPS we propose two methods. The first is to fully deactivate the use of TMVP, which yields good error-resilient performance when used with simple coding structures such as IPPP as mentioned in Ref. [25]. Second, the No-TMVP Refresh (NTR4) frames are used. A detailed description of the used strategies will be discussed in the following paragraphs.

4.3.1 Proposed solution for the RPS method

The RPS works in two modes, positive Acknowledgment (ACK) signals and negative Acknowledgment (NACK) signals. In our case, we used the ACK based RPS mode as it provides better performance than the NACK mode [105]. In the former system (ACK), the encoder receives ACK signals via a backward feedback channel whenever a frame is correctly received by the decoder. When the frame is lost, the decoder doesn't send any signal for that frame. Consequently, the encoder will adjust its reference frames based on the error status. On the other hand, in NACK the decoder will inform the encoder about the lost frames.

Our RPS proposed approach is quite similar to the one presented in Ref. [57]. Indeed, it exploits multiple references and the appearance of errors to mitigate the error propagation due to error concealment error at the decoder side. By default, when an error happens, the decoder applies the frame-copy error concealment technique at the decoder to conceal the lost frames.

However, the drawback of the copy concealment is that the motion information will be lost. Hence, this leads to error propagation at the decoder side due to motion information mismatch prediction for the subsequent frames. Moreover, when the trip delay is large, the number of infected frames will be high.

On one hand, when feedback channel updates are available i.e., perfect condition and at the first appearance of errors at the encoder, the RPS checks if the reference pictures of the current frame in the reference picture set are located after the lost frame. If it is the case, the algorithm will refer to the previously encoded frames, thus avoiding referring to the altered ones. This is achieved by selecting three of the last pictures in the Group of pictures (GOPs) that precede the erroneous frame and the immediate previous picture. Note that due to the Rate-Distortion Optimization (RDO) and the used coding structure (IPPP), the final picture in the GOP contains the largest bit budget.

At this stage, the encoder will signal the new reference pictures for the frame being encoded to the

decoder via a local Reference Picture Set at the Slice header, refer to Algo. 1. Consequently, the decoder will avoid utilizing the predefined reference picture sets, thus preventing referencing from erroneous frames. Moreover, the encoder hints error status to the subsequent frames being encoded. In a nutshell, in this case, the error will only propagate to a small number of frames (two) and the RPS algorithm will have a great impact on the end-user quality.

On the other hand, when the delay is significant or there are no feedback channel updates, the error will propagate to a great number of frames located after the concealed frame until when the RPS algorithm is triggered. i.e., When applying the RPS algorithm, the quality will be enhanced after the trip delayed frame numbers. In this case, we take advantage of the TMVP by decreasing the temporal dependency. In other words, when real-time is crucial and feedback channel updates are absent, our method will increase the error resilience by decreasing the extensive use of the temporal dependency introduced in the HEVC.

Algorithm 1: The Proposed algorithm

```
Data: Deactivate TMVP with a chosen strategy
Data: Read feedback channel Updates
if ACK signal is detected then
    ErrorDetected = false
else
    ErrorDetected = true
    TripDelay = POCcurrent - POClost
    for all Slices in Current Frame do
        Get all Reference Pictures
        for all Reference Pictures in Slice do
            if POCReferencePicture ≥POCLost then
                update Reference Picture in the DPB
            end
        end
        Apply Reference Pictures
        Signal to slice header the new Reference Picture Set
        Signal Error Information to Subsequent Frames
    end
end
```

4.3.2 Proposed approach to decrease the temporal dependency

As mentioned in previous sections, HEVC has increased the temporal dependency between frames leading to decreased robustness against errors. Based on this fact, we use an RPS implementation with

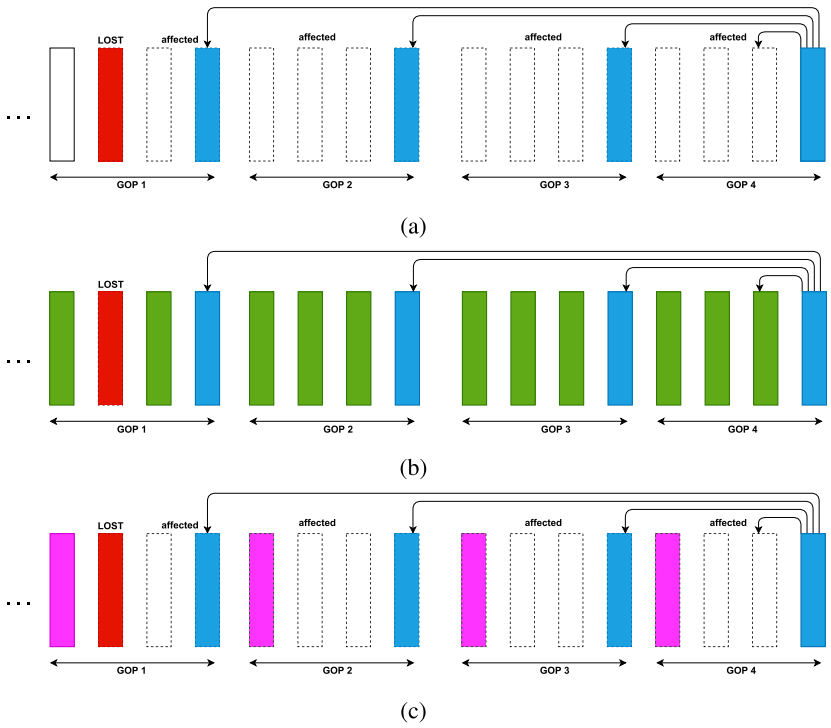


Figure 4.2 – Decreased temporal dependency strategies. (a) TMVP for all frames (b) No-TMVP for all frames (light green) (c) NTR4 frames at the beginning of each GOP (Pink).

TMVP to mitigate the temporal propagation due to the Trip Delay (TD) in an RPS algorithm.

Since the moving vectors of the current frame will be encoded using motion vectors of the previously encoded frame, the loss of this frame will affect the decoding of subsequent frames that refer to it and thus leading to error propagation which is not desirable. Note that such a scenario would never happen in H.264/AVC standard as it doesn't support the use of temporal motion vector candidates. Hence, one solution is to totally disable the relation between these moving vectors as shown in Fig. 4.2b i.e., light green frames. In this research, we refer to these frames as No-TMVP frames. With this strategy, we will increase the number of independent decodable moving vectors and thus truncating the error propagation at the MV level. However, this will lead to decreased Rate-Distortion (R-D) performance as less reference blocks are used to predict or encode MVs.

To compensate for the R-D performance without penalizing the error resilience we suggest the use of No-TMVP Refresh (NTR4) frames. These frames represent the first pictures of each GOP. In the IPPP coding structure, the GOP size is fixed to be 4 frames as shown in Fig. 4.2c.

To further illustrate the proposed approach, in Fig. 4.2a, when the RED frame is lost within the GOP1 and due to the activation of TMVP, the error will propagate to all subsequent GOPs and all the frames will be affected. Indeed, all the reference pictures of the last frame in GOP4 will be affected as well. However, when the Temporal candidate is deactivated as in Fig. 4.2b, meaning that all the Moving

vectors are independently decodable, we will not have the propagation of the error between Moving Vectors in different frames. In addition, all the reference frames of the blue picture in GOP4 will not be affected. Following the same scenario, when the RED frame is lost in GOP1 as seen in Fig. 4.2c, the error propagation will be contained within GOP1. Despite that fact, the blue frame in GOP4 will be partially affected. With this strategy, we are slightly decreasing the error resilience to compensate for the compression efficiency. Note that, according to Ref. [25] deactivating the TMVP partially is better than fully deactivating it.

4.4 Evaluation setup

To allow the replication of our work, this section describes the conduction of the test experiments. We present the selected test video sequences, the used software and the encoding configurations. The main goal is to test the performance of our proposed approach under unreliable medium using different coding configurations.

4.4.1 Dataset

To assess the impact of video characteristics on our algorithm, we used a variety of different content with different degrees of texture and temporal details. According to Ref. [78], the selected videos shall cover the maximum Spatial Information (SI) and Temporal information (TI) plan i.e., SI/TI plan. This

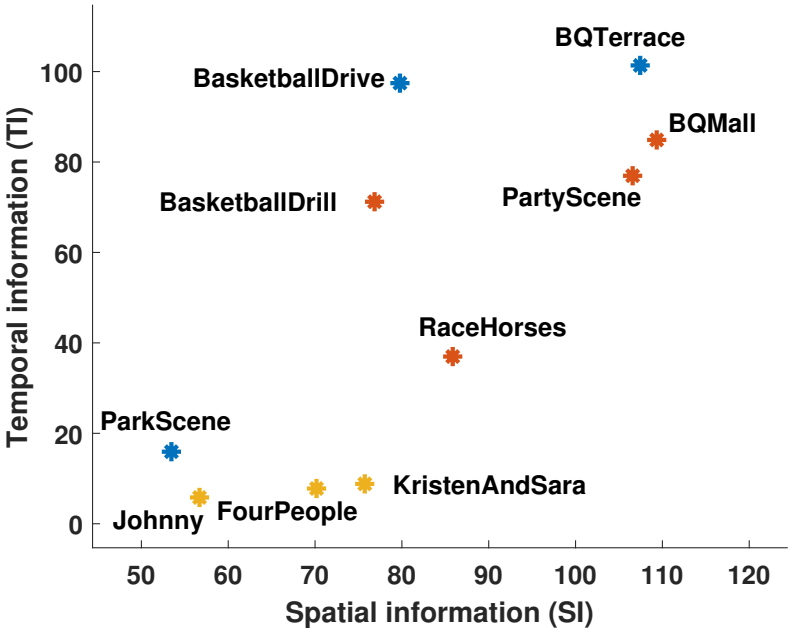


Figure 4.3 – Spatial and temporal information plot of the test sequences.

Table 4.1 – Selected Test Video Sequences.

Class	Video Sample	Content Description	Frame Rate (fps)
C 832×480	Basketball Drill	Moderate texture and rapid change of the players (High Motion)	50
	BQMall	High texture and high translational movement (rapid change)	60
	Party Scene	High texture background and high motion (with zoom)	50
	Race Horses	Medium motion and moderate texture	30
E 1280×720	FourPeople	Four people doing HD video conferencing (arms moving)	60
	Johnny	Still background and low local motion	60
	KristenAndSara	Still background and moderate local motion	60
B 1920×1080	BasketballDrive	Moderate texture, camera following the player	50
	BQTerrace	High Motion, Camera tilts at an angle and then focuses on the road	60
	ParkScene	Low movement and low spatial complexity	24

plan shows the temporal and spatial complexity distribution of all the samples, refer to Fig. 4.3. Class C, Class E and Class B video sequences are appropriate for evaluating low-delay applications [77]. In these classes, videos are of WVGA (832×480), 720p (1280×720) and Full-HD 1080p (1920×1080) spatial resolution, respectively and of different frame rates with 10 seconds length. Table 4.1, provides a description of the used test sequences.

4.4.2 Configuration setup

In order to conduct the experiments and to produce an HEVC compliant bitstream that is robust to network errors, we implemented our algorithm on top of HEVC Test Model (HM) 16.20 [65]. The test sequences were encoded using Low-Delay (LD-P) coding structure. It was selected from the configuration files provided by the HM. Compared to other coding structures, such as Random-Access (RA) and All-Intra (AI), LD-P prediction pattern provides good performance for real-time applications as it only relies on past information to encode videos. Moreover, it guarantees the delay requirements in delay-sensitive video transmission.

Considering the importance of rate control, especially for applications constrained by bandwidth requirements such as real-time video transmission, we encoded the videos using different targeted bitrates. Indeed, scholars in Ref. [109] recommend using the following values to conduct rate-constrained tests i.e., 384 Kbps, 512 Kbps, 768 Kbps, 1200 Kbps and 2000 Kbps.

In addition to the aforementioned configurations and regarding the RPS algorithm, we guaranteed the availability of 4 pictures in the Decoded Picture Buffer as well as 4 active pictures to be used as a

reference in the configuration file. To maintain the efficiency of the HM software in terms of encoding and decoding run-time, we did not use extra C++ functions. Rather, we used the built-in classes.

4.4.3 Test conditions

To evaluate the proposed algorithm in an error-prone-like environment, we used the NAL unit Loss Simulator [82], which utilizes a predefined loss pattern to apply errors to the bitstream. However, it does not work directly with HEVC and it requires some modifications.

To introduce errors to our bitstreams in a packetized format, we represented each frame or each slice as one packet. After, the loss of a frame, the decoder manages to conceal this lost data by applying frame copy error concealment [99]. The concealment strategy is used to avoid failure in the decoding stage.

In fact, with this scenario, only the motion information will be lost. In other words, the motion information will be removed and it will not be available on the decoder side. Consequently, we will produce error propagation due to the mismatch prediction of the motion information that in turn is due to the inadequate performance of the frame copy concealment strategy.

Note that, in our system, backward feedback channel updates are assumed and the RPS algorithm will adjust its reference frames based on this information. Indeed, to guarantee the appropriate reference pictures at the DPB, we used a bitstream analyzer known as Elecard Analyzer [110]. A round trip delay of eight frames is imposed on our simulated framework. This was an arbitrary choice by the authors of this paper as the RPS algorithm is supposed to be implemented with a trip delay of two frames.

4.5 Results and Discussions

This section presents the discussion of the obtained results from encoding the test video sequences. Note that all the tests were carried out on a PC with an AMD Ryzen 7 2700X 8 Core CPU, 3.7 GHz and a RAM of 16GB. In the first experiments, we studied the effect of using different Intra refresh updates when rate control is activated in an error-free environment. Then, we compared the best resulting configuration with different strategies to combat network errors in a lossy environment.

Indeed, for our experiments, we relied on two well-known full-reference assessment metrics, Peak Signal to Noise Ratio (PSNR) [79] and Structural Index Similarity (SSIM) [100]. *SSIM* is define using Eq. 4.1

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4.1}$$

Where μ_x and μ_y are the mean luminance and σ_x and σ_y are the standard deviation. σ_{xy} is the covariance between x and y. The positive constants are $C1$ and $C2$.

Table 4.2 – PSNR Results for the Default algorithm with different intra-refresh rates.

	Target Bitrate (kbps)	Average Achieved bitrate (kbps)	Average PSNR (dB)
	384	385.5382	30.24
HMRPS-LP-Default	512	513.5384	31.19
	768	769.5558	32.59
	1200	1201.5712	34.17
	2000	2001.557	36.06
Average	/	/	**32.85**
	384	513.6952	26.62
HMRPS-LPI4- Default	512	625.0688	27.53
	768	882.8426	29.07
	1200	1320.634	30.84
	2000	2045.3608	32.63
Average	/	/	**29.34**
	384	428.0616	27.59
HMRPS-LPI8- Default	512	544.1922	28.61
	768	797.5916	30.29
	1200	1219.7052	31.99
	2000	2018.1732	33.92
Average	/	/	**30.48**
	384	398.5358	28.72
HMRPS-LPI16- Default	512	523.055	29.86
	768	776.6008	31.36
	1200	1207.5096	33.02
	2000	2008.529	35.04
Average	/	/	31.60
	384	385.5362	30
HMRPS-LP'IS'- Default	512	513.5406	30.98
	768	769.5466	32.39
	1200	1201.5452	33.99
	2000	2001.568	35.91
Average	/	/	**32.65**

4.5.1 Experimental results for Intra-refresh updates

The performance of the default RPS algorithm is compared with four different coding structures as shown in Tab. 4.2. HMRPS-LPI-default is the default RPS algorithm with the TMVP being fully activated and using the default coding structure IPPP, where the different versions LPI4, LPI8, LP16 and LPIS denote different Intra refresh cycles. Intra refresh is inserted regularly each 4, 8, 16, frames and at each second respective for LPI4, LPI8, LPI16 and LPIS. In this subpart, for in-depth analysis, we only rely on class C video sequences.

As can be seen from Tab. 4.2, when calculating the average PSNR values and the achieved bitrates for all the sequences, the more we add Intra frames in a video sequence, we notice less performance of the rate control algorithm in achieving the targeted bitrate. In other words, take for instance the LPI4, which introduces an Intra-frame at each existing GOP, we can see that it provides the least quality in

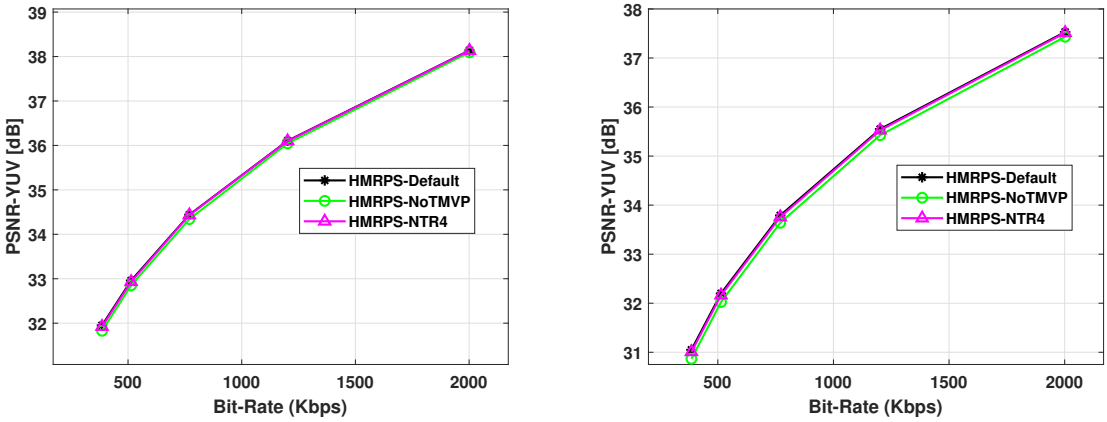


Figure 4.4 – PSNR R-D curves in error free environment (a) BasketballDrill; (b) BQmall.

terms of PSNR values and provides high bitrates related to the targeted ones. This is due to the fact that the HEVC rate control algorithm allocates a high bitrate budget for those Intra coded frames and uses high Quantization Parameter (QP) values for the rest of the pictures in the video sequence. The same results can be observed for the LPI8 and LPI16 configurations with increased R-D performance when using less Intra updates.

However, for the LPIS, where we introduced an Intra-frame at each second, we can see that the R-D performance is almost the same as when we used the default coding structure IPPP. Note that, the achieved bitrates have almost the exact values as the targeted bitrates. In addition, we can see that we have only a difference of 0.20 dB in PSNR. Indeed, the LPIS coding structure seems to preserve the time characteristics such as that of the low delay IPPP structure.

It is worth mentioning that, the same results are observed for the HMRPS-NoTMVP as well as for HMRPS-NTR4 algorithms for different Intra refresh updates. Consequently, the same conclusions can be made. However, based on this study, in the following experiments, we will use only the LPIS structure rather than the default IPPP structure.

4.5.2 Experimental results for the proposed strategies

In this subsection, the two proposed methods namely, HMRPS-NoTMVP and HMRPS-NTR4 are compared with the default HMRPS-Default algorithm. Note that, the LPIS prediction structure is used. In this work, we compared the compression efficiency of the three algorithms as well as their error resilience performance in an error-prone environment. For the error-prone conditions, we dropped the frame with Picture Order Count (POC) 20. However, as discussed earlier high trip delay of 8 frames is imposed on our framework. With this scenario, the error will propagate until the RPS algorithm is activated in POC 28. In order to analyse the impact of different sequences from different classes on the

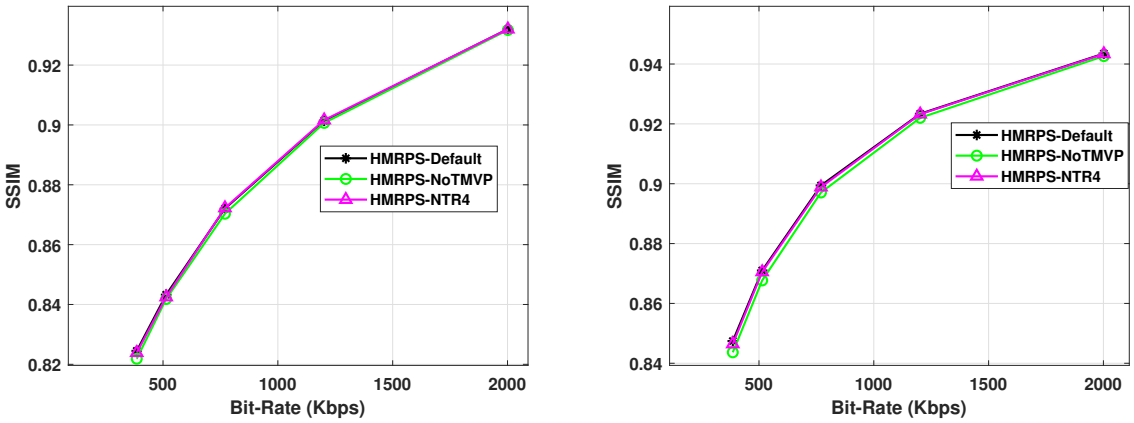


Figure 4.5 – SSIM R-D curves in error free environment (a) BasketballDrill; (b) BQmall.

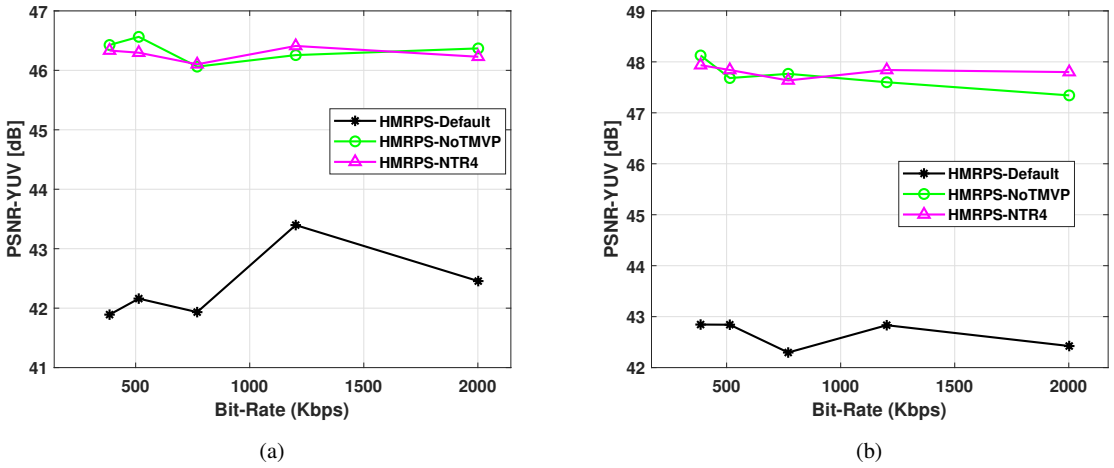


Figure 4.6 – PSNR R-D curves when frame 20 is dropped (a) BasketballDrill; (b) BQmall.

proposed methods, all the test sequences presented in Tab. 4.1 were included in the test.

The results of the compression efficiency for different video sequences are depicted in Fig. 4.4 and Fig. 4.5. As can be seen from both figures, fully deactivating the TMVP provided the least amount of quality in comparison to the NRT4 and the default algorithm. Although the difference is almost negligible and the graphs are almost overlapping, we can see that NTR4 provide the same performance as that of the Default algorithm. Note that, the same conclusions were observed for PSNR as well as for SSIM plots regardless of the class category of the test sequences. Hence, we can conclude that the deactivation of TMVP in either case i.e., fully or periodically. does not affect the compression performance in rate constrained conditions using the LPIS structure.

On the contrary, when we expose the resulted bitstreams to even one frame loss as shown in Fig. 4.6 and Fig. 4.7, we can see that default RPS algorithm which yields significant gain in terms of

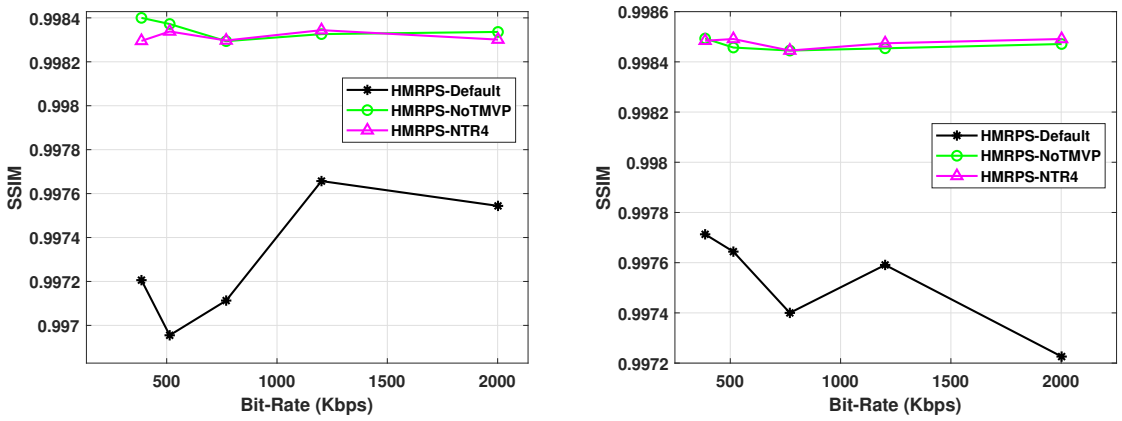


Figure 4.7 – SSIM R-D curves when frame 20 is dropped (a) BasketballDrill; (b) BQmall..

compression efficiency, provides the least performance in the event of whole frame loss. Whereas the RPS algorithm with decreased temporal dependency using both strategies NTR4 and NoTMVP provide enhanced performance regarding the HEVC robustness against errors. Indeed, the NTR4 and NoTMVP provide almost the same performance in terms of PSNR and SSIM in the event of whole frame loss. With these strategies, we can effectively enhance the visual quality and truncate the error propagation leading to enhanced perceived quality. Note that, the same results are observed for all the video test sequences that contain different degrees of temporal and spatial complexity. However, for some videos such BQTerrace, the gap difference between graphs is larger than for other videos due to their different content characteristics.

Table 4.3 and Tab. 4.4 provide detailed results of all the selected video sequences from three different class resolutions. Indeed, Tab. 4.3, provides the average PSNR and SSIM gains of the proposed strategies in regard to the default algorithm. First, the gain is obtained by calculating the difference between the average results of the proposed strategies and the default method for each video sequence. After that, the mean of the difference for all the test sequences in each class category is calculated and denoted as PSNR and SSIM gain for that class as in Ref. [57]. Note that, the average value corresponding to each video sequence is obtained by calculating the mean value of five different bitrate levels as depicted in Tab. 4.4.

Based on the presented results from Tab. 4.3 and Tab. 4.4, for all the different classes, the proposed RPS methods achieve better PSNR and SSIM values compared to the default RPS algorithm. Indeed, from the viewpoint of the average PSNR metric, RPS-No-TMVP and RPS-NTR4 provide a maximum gain of about 6.13 dB and 5.97 dB respectively for the Full-HD test sequences. However, for 720p video sequences, the proposed methods achieve average PSNR improvements of about 5.20 dB and 5.24 dB respectively. Moreover, for 480p videos, we notice that 4.72 dB and 4.69 dB gains are achieved by RPS-No-TMVP and RPS-NTR4 respectively. It is worth mentioning that the proposed methods

Table 4.3 – Average PSNR and SSIM gains for compared performance of proposed strategies.

		Δ PSNR (dB)		**Δ SSIM**	
Class	**Sequence**	**No TMVP Vs Default**	**NTR4 Vs Default**	**No TMVP Vs Default**	**NTR4 Vs Default**
	BasketballDrill	3.97	3.91	0.00105	0.00102
	BQMall	5.05	5.16	0.000949	0.00096
C	PartyScene	6.38	6.29	0.003852	0.003824
(832 × 480)	RaceHorses	3.47	3.41	0.0043536	0.0042618
	Averages	**4.72**	**4.69**	**0.00255135**	**0.002517**
	FourPeople	4.38	4.50	0.0001032	0.0001076
E	Johnny	4.94	4.94	0.0003578	0.0003522
(1280 × 720)	KristenAndSara	6.27	6.29	0.0003512	0.0003556
	Averages	**5.20**	**5.24**	**0.000270733**	**0.0002718**
	BasketballDrive	4.56	4.31	0.0017402	0.0016794
B	BQTerrace	10.42	10.24	0.003685	0.003624
(1920 × 1080)	ParkScene	3.41	3.36	0.001209	0.001186
	Averages	**6.13**	**5.97**	**0.0022114**	**0.0022114**

achieved the lowest PSNR gains with 480p video sequences. As for the greatest gains, they were obtained with high resolution videos.

Regarding the video content, as shown in Tab. 4.3, the least performance is obtained with RaceHorses and ParkScene video sequences. These videos are shown to have low frame rate (30 and 24 respectively) and low to moderate spatial complexity. Indeed, the decreased gain is obtained regardless of their class category and regardless of their Temporal Information. In contrast, videos with high fps and high SI/TI, have the largest PSNR gain. These sequences are BQTerrace, PartyScene and BQMall. One exception is KristenAndSara sequence which yields a gain of about 6.27 dB with moderate spatial complexity and moderate local motion.

According to Tab. 4.3, the proposed approaches provide a maximum gain of about 10.42 dB and 10.24 dB in terms of PSNR for RPS-No-TMVP and RPS-NTR4 respectively for the BQTerrace sequence. In the same category (Class B, 1080p), we can see that basketballdrive yields a gain of only 4.56 dB compared to BQTerrace. This decrease in PSNR gain can be attributed the fact that basketballdrive contains lower SI and lower frame rate than BQTerrace. On the other hand, it is clearly observed that the low frame rate (24) and the low SI/TI information are the essential factors that lead to decreased gain in PSNR for ParkScene sequence. Indeed, the 3.41 dB gain for ParkScene is seen to be the least gain in this category.

As illustrated in Fig. 4.3, the 720p class video sequences contain low to moderate Spatial Information. In fact, they have the same frame rate and the same Temporal Information. However, based on Tab. 4.3, we can see that KristenAndSara which contains the highest SI among the test sequences has the largest gain. To this end, we can say that Spatial Information has a great impact on the PSNR gain as was the case with class B video sequences. The same conclusions can be observed with Partyscene and Basketballdrill video sequences of class C (480p), in which they have the same fps and the same

Table 4.4 – PSNR and SSIM Results for all the algorithms when frame 20 is dropped

Video Sequences	Target Bitrate (kbps)	PSNR (dB)			SSIM		
		Default	**No TMVP**	**NTR4**	**Default**	**No TMVP**	**NTR4**
	384	41.890	46.42	46.33	0.997206	0.9984	0.998295
	512	42.16	46.56	46.29	0.996955	0.998372	0.998338
BasketballDrill	768	41.93	46.06	46.10	0.997113	0.998294	0.998297
832 × 480 **(fps=50)**	1200	43.39	46.25	46.41	0.997657	0.998326	0.998344
	2000	42.45	46.36	46.22	0.997544	0.998336	0.998301
Averages	/	**42.36**	**46.33**	**46.27**	**0.997295**	**0.9983456**	**0.998315**
	384	42.84	48.12	47.93	0.997713	0.998493	0.998484
	512	42.84	47.68	47.84	0.997644	0.998457	0.998491
BQMall	768	42.29	47.76	47.63	0.9974	0.998445	0.998445
832 × 480 **(fps=60)**	1200	42.83	47.59	47.83	0.997591	0.998454	0.998474
	2000	42.42	47.34	47.79	0.997226	0.998471	0.998491
Averages	/	**42.64**	**47.70**	**47.80**	**0.9975148**	**0.998464**	**0.998477**
	384	44.45	50.26	49.87	0.994791	0.998397	0.998332
	512	44.13	49.94	49.84	0.994802	0.998271	0.998237
PartyScene	768	44.02	50.45	50.58	0.99453	0.998506	0.998501
832 × 480 **(fps=50)**	1200	43.74	50.52	50.46	0.994504	0.998482	0.998487
	2000	43.48	50.57	50.53	0.994394	0.998625	0.998584
Averages	/	**43.96**	**50.35**	**50.26**	**0.9946042**	**0.9984562**	**0.9984282**
	384	33.51	37.40	37.28	0.987244	0.991231	0.99107
	512	33.57	37.35	37.24	0.986932	0.99126	0.990949
RaceHorses	768	33.79	37.23	37.24	0.986634	0.990663	0.99071
832 × 480 **(fps=30)**	1200	33.95	37.11	37.05	0.985866	0.9907	0.990619
	2000	33.99	37.07	37.04	0.986059	0.990649	0.990696
Averages	/	**33.76**	**37.23**	**37.17**	**0.986547**d	**0.9909006**	**0.9908088**
	384	58.48	60.63	60.83	0.99979	0.999819	0.999817
	512	57.43	61.30	61.07	0.999764	0.999832	0.999845
FourPeople	768	56.94	61.71	61.99	0.999756	0.999854	0.999862
1280 × 720 **(fps=60)**	1200	56.64	62.05	61.75	0.99973	0.999857	0.999855
	2000	55.66	61.38	62.00	0.999657	0.999851	0.999856
Averages	/	**57.03**	**61.41**	**61.53**	**0.9997394**	**0.9998426**	**0.999847**
	384	56.33	61.53	61.40	0.999321	0.999749	0.999719
	512	58.61	61.66	61.67	0.999602	0.999769	0.999759
Johnny	768	57.26	62.22	62.41	0.99942	0.999771	0.999794
1280 × 720 **(fps=60)**	1200	56.79	62.50	62.57	0.999381	0.999811	0.9998
	2000	56.96	62.78	62.61	0.999379	0.999792	0.999792
Averages	/	**57.19**	**62.14**	**62.13**	**0.9994206**	**0.9997784**	**0.9997728**
	384	53.43	59.53	59.75	0.999194	0.99955	0.999574
	512	54.21	59.94	59.90	0.99925	0.999594	0.999588
KristenAndSara	768	54.33	60.17	60.20	0.999314	0.999589	0.99959
1280 × 720 **(fps=60)**	1200	53.59	60.64	60.62	0.999262	0.999611	0.999612
	2000	54.52	61.15	61.08	0.99921	0.999642	0.999644
Averages	/	**54.01**	**60.29**	**60.31**	**0.999246**	**0.9995972**	**0.9996016**
	384	40.62	45.67	45.71	0.99563	0.997515	0.997384
	512	40.42	45.30	44.75	0.995561	0.99732	0.997207
BasketballDrive	768	40.60	44.80	44.54	0.995589	0.997252	0.997245
1920 × 1080 **(fps=50)**	1200	40.44	44.78	44.69	0.995472	0.997183	0.997165
	2000	40.46	44.80	44.39	0.995513	0.997196	0.997161
Averages	/	**40.51**	**45.07**	**44.81**	**0.995553**	**0.9972932**	**0.9972324**
	384	46.38	54.80	54.93	0.997669	0.999453	0.999131
	512	44.70	55.65	54.06	0.996389	0.999515	0.999425
BQTerrace	768	43.48	53.11	53.56	0.995301	0.998953	0.999034
1920 × 1080 **(fps=60)**	1200	42.83	54.06	54.23	0.994404	0.999242	0.999258
	2000	42.63	54.49	54.44	0.99425	0.999275	0.999285
Averages	/	**44.01**	**54.42**	**54.24**	**0.9956026**	**0.9992876**	**0.9992266**
	384	48.27	51.81	51.67	0.997074	0.998298	0.998242
	512	47.55	51.66	51.69	0.996977	0.998274	0.99828
ParkScene	768	48.29	51.49	51.44	0.99712	0.998225	0.998203
1920 × 1080 **(fps=24)**	1200	48.23	51.26	51.24	0.996996	0.998183	0.998182
	2000	47.99	51.15	51.09	0.996896	0.998128	0.998086
Averages	/	**48.07**	**51.47**	**51.43**	**0.9970126**	**0.9982216**	**0.9981986**

Figure 4.8 – Frame by frame visual quality evaluation, frame 27 snapshot of all the test sequences: (a) default, (b) No-TMVP, (c) NTR4

TI. According to the fact that Partyscene contains higher Spatial Information, it yields 6.38 dB while Basketballdrill yields 3.97 dB. Note that, the least performance in this category is obtained by the RaceHorses sequence, this is due to the fact that RaceHorses contains moderate SI and low frame rate.

Considering the obtained PSNR results from Tab. 4.3, Tab. 4.4 and the visual results from Fig. 4.8, we can say that in our case, the SSIM fails and therefore the PSNR metric is more reflective of the human visual system in the event of packet loss.

Based on the carried experiments, we can conclude that the most important factors that affect our proposed algorithms are:

First, the frame rate (fps) and second, the Spatial Information. Moreover, as discussed earlier, the increase in video resolution and the temporal information increase the gain for our proposed methods.

The visual quality of the proposed algorithms is further evaluated in this section. When comparing the three algorithms in an error-prone environment as depicted in Fig. 4.8, we can see that the TMVP usage has a devastating effect on the end video quality. However, the proposed strategies (No-TMVP and NTR4) yield the same visual quality with a small difference based on the amount of motion information found in the video sequences.

As is clearly observed, the video content has a substantial effect on the chosen algorithm. However, our

proposed algorithms yield good performance for content with high motion and high complex scenes.

4.6 Conclusion

In this chapter, a Reference Picture Selection algorithm with decreased temporal dependency is proposed to improve the end video quality. Indeed, two modes for decreasing the TMVP are tested: fully deactivating the TMVP as well as a periodical deactivation. To the best of our knowledge, this is the first work that studies the effect of this particular combination. The proposed methods yield a significant gain in terms of quality demonstrated by the PSNR and SSIM values. The results obtained, show that deactivating the TMVP will not affect the compression efficiency, however, it will provide significant gain in the event of frame loss of about 4.72 dB, 5.20 dB and 6.13 dB for 480p, 720p and 1080p resolutions respectively.

As future works, the best strategies that exist for decreasing the temporal dependency can be tested with the RPS algorithm. In addition, studying the vulnerability of losing the moving vectors in newer standards such as VVC is crucial as it introduces new motion estimation tools.

Conclusion and Future Works

This thesis has analyzed the theoretical background of video compression and error resilience, specifically for the HEVC video coding standard. Through comparative evaluations in Chapters 2 and Chapter 3, the compression efficiency of different software implementations for a variety of video coding standards and the error resilience performance of HEVC were analyzed and compared. In Chapter 4, a novel approach to enhance the error resilience of real-time HEVC video transmission was proposed, consisting of the combination of three error-resilient methods. Through the conducted research in our thesis, the following conclusions have been drawn.

Based on the experiments presented in chapter 2, we saw that each newly born standard indeed provides enhancements in coding performance according to its predecessors. However, newly developed codecs cannot be used for certain applications that require constrained and limited resources. For instance, even though VVC and AV1 are newly born codecs that provide bitrate saving according to HEVC, we found that HEVC is the best choice for real-time video surveillance applications, more precisely, x265-medium implementation. In chapter 2, we compared different codecs for a variety of test sequences and we used similar coding configurations. Based on this study we noticed that the multi-pass encoding mode is not suitable for low-delay configurations and real-time video applications. However, single pass encoding mode is desirable for such applications specifically when addressing delay-bounded transmission such as real-time video surveillance. The error resilience of HEVC is further analyzed in chapter 3. In this chapter, we compared the error resilience of two prominent HEVC implementations i.e., HM and x265. Based on the obtained results, our study concludes that both implementations provide similar performances. Moreover, this chapter benchmarks the efficiency of HEVC newly developed methods in error-free and erroneous events using different transmission platforms. Results from this work show that the most responsible methods for the coding performance gain led to decreased performance in the event of packet loss. An example of this is the temporal motion vector prediction tool. The conducted experimental studies in Chapters 2 and 3 provide insights into the performance and effectiveness of different methods for HEVC video transmission, taking into account the chosen implementation and the type of video content. With this, a comprehensive overview of video coding standards addressing video compression as well as video transmission is provided.

Indeed, the works presented in previous chapters enabled us to propose a method that enhances the robustness of HEVC bitstreams to network errors. In other words, our proposed method in chapter 4 builds upon the findings of our studies found in chapter 2 and chapter 3. In the final chapter of our thesis, we proposed an error resilience method based on combining source coding and selective encoding using feedback channel updates. This strategy enhances error resilience and is suitable for delay-bounded applications such as real-time video surveillance. Indeed, we used three different techniques namely: Intra-refresh, decreased temporal dependency using TMVP and Reference Picture Selection (RPS) methods. RPS and Intra-Refresh are our main error resilience methods. However,

TMVP is used to improve the RPS performance. In other words, we enhanced the RPS performance using TVMP making it more suitable for low-delay applications. We proposed to decrease the temporal dependency using two strategies: fully and periodically. The results of the study indicate that the proposed approach in Chapter 4 effectively enhances the error resilience of real-time HEVC video transmission, leading to improved end video quality. This work has made a valuable contribution to the field of digital video transmission and has the potential to improve the overall user experience for video surveillance over error-prone networks.

In the future, it would be interesting to further evaluate the proposed approach under a realistic video surveillance framework with various types of video content. Additionally, the proposed approach could be extended to other video coding standards, such as AV1 and VVC, to explore its generality and effectiveness. Another interesting direction for future work is to integrate the proposed approach with other error resilience techniques, such as forward error correction, to provide an even more robust solution for real-time video transmission over error-prone networks.

Bibliography

[1] Psannis, K. HEVC in wireless environments. *Journal Of Real-Time Image Processing*. **12**, 509-516 (2016,8), https://doi.org/10.1007/s11554-015-0514-6, [doi:10.1007/s11554-015-0514-6]

[2] Cisco Visual Networking Index: Forecast and Trends 2017–2022,"`https://twiki.cern.ch/twiki/pub/HEPIX/TechwatchNetwork/HtwNetworkDocuments/white-paper-c11-741490.pdf`",[Online; Accessed 2022-05-15]

[3] Carreira, J., De Silva, V., Ekmekcioglu, E., Kondoz, A., Assuncao, P. & Faria, S. Dynamic motion vector refreshing for enhanced error resilience in HEVC. *2014 22nd European Signal Processing Conference (EUSIPCO)*. pp. 281-285 (2014,9)

[4] Wang, Y., Wenger, S., Wen, J. & Katsaggelos, A. Review of error resilient coding techniques for real-time video communications. *IEEE Signal Processing Magazine*. **17**, 61-82 (2000)

[5] Nightingale, J., Wang, Q. & Grecos, C. HEVStream: a framework for streaming and evaluation of high efficiency video coding (HEVC) content in loss-prone networks. *IEEE Transactions On Consumer Electronics*. **58**, 404-412 (2012)

[6] Sullivan, G., Ohm, J., Han, W. & Wiegand, T. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1649-1668 (2012,12), [doi:10.1109/TCSVT.2012.222119]

[7] Oztas, B., Pourazad, M., Nasiopoulos, P. & Leung, V. A study on the HEVC performance over lossy networks. *2012 19th IEEE International Conference On Electronics, Circuits, And Systems (ICECS 2012)*. pp. 785-788 (2012,12), [doi:10.1109/ICECS.2012.6463542]

[8] Yang, Y. & Zhu, Q. Error control and concealment for video communications: a review. *Proceedings Of The IEEE*. **86**, 974-997 (1998)

[9] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S. & Hannuksela, M. RTP payload format for high efficiency video coding (HEVC). (2016)

[10] Schwarz, H., Marpe, D. & Wiegand, T. Overview of the scalable video coding extension of the H. 264/AVC standard. *IEEE Transactions On Circuits And Systems For Video Technology*. **17**, 1103-1120 (2007)

[11] Helle, P., Lakshman, H., Siekmann, M., Stegemann, J., Hinz, T., Schwarz, H., Marpe, D. & Wiegand, T. A scalable video coding extension of HEVC. *2013 Data Compression Conference*. pp. 201-210 (2013)

[12] Y. Wang, J. Ostermann, & Y.-Q. Zhang, Video processing and communications. *Prentice Hall Upper Saddle River* 2002, vol. 5.

[13] Chakareski, J., Han, S. & Girod, B. Layered coding vs. multiple descriptions for video streaming over multiple paths. *Proceedings Of The Eleventh ACM International Conference On Multimedia*. pp. 422-431 (2003)

[14] Lee, Y., Kim, J., Altunbasak, Y. & Mersereau, R. Layered coded vs. multiple description coded video over error-prone networks. *Signal Processing: Image Communication*. **18**, 337-356 (2003)

[15] Goyal, V. Multiple description coding: Compression meets the network. *IEEE Signal Processing Magazine*. **18**, 74-93 (2001)

[16] Wang, Y., Reibman, A. & Lin, S. Multiple description coding for video delivery. *Proceedings Of The IEEE*. **93**, 57-70 (2004)

[17] Kazemi, M., Shirmohammadi, S. & Sadeghi, K. A review of multiple description coding techniques for error-resilient video delivery. *Multimedia Systems*. **20**, 283-309 (2014)

[18] Majid, M., Owais, M. & Anwar, S. Visual saliency based redundancy allocation in HEVC compatible multiple description video coding. *Multimedia Tools And Applications*. **77**, 20955-20977 (2018)

[19] Duong, D. Distributed Coding Based Multiple Descriptions for Robust Video Transmission over Error-Prone Networks. *Proceedings Of The 2020 International Conference On Computer Communication And Information Systems*. pp. 24-28 (2020)

[20] Duong, D. New H. 266/VVC Based Multiple Description Coding for Robust Video Transmission over Error-Prone Networks. *2020 International Conference On Advanced Technologies For Communications (ATC)*. pp. 155-159 (2020)

[21] Zeeshan, M. & Majid, M. HEVC compatible perceptual multiple description video coding for reliable video transmission over packet networks. *Telecommunication Systems*. **76**, 63-83 (2021)

[22] Zhang, Y. & Gu, B. Error-resilient coding by convolutional neural networks for underwater video transmission. *Journal Of The Franklin Institute*. **358**, 9307-9324 (2021)

[23] Wang, F., Chen, J., Zeng, H. & Cai, C. Spatial-frequency HEVC multiple description video coding with adaptive perceptual redundancy allocation. *Journal Of Visual Communication And Image Representation*. **88** pp. 103614 (2022)

[24] Cong, H., Van, X. & Dinh, D. HEVC Compatible Multiple Description Coding for Robust Video Transmission over Lossy Networks. *VNU Journal Of Science: Computer Science And Communication Engineering*. (2022)

[25] Li, B., Xu, J. & Li, H. Parsing robustness in high efficiency video coding-analysis and improvement. *2011 Visual Communications And Image Processing (VCIP)*. pp. 1-4 (2011)

[26] Carreira, J., Ekmekcioglu, E., Kondoz, A., Assuncao, P., Faria, S. & De Silva, V. Selective motion vector redundancies for improved error resilience in HEVC. *2014 IEEE International Conference On Image Processing (ICIP)*. pp. 2457-2461 (2014)

[27] Zou, J. & Li, B. Rate Control in HEVC: A Survey. *Signal And Information Processing, Networking And Computers: Proceedings Of The 5th International Conference On Signal And Information Processing, Networking And Computers (ICSINC)*. pp. 578-583 (2019)

[28] Zhang, R., Regunathan, S. & Rose, K. Video coding with optimal inter/intra-mode switching for packet loss resilience. *IEEE Journal On Selected Areas In Communications*. **18**, 966-976 (2000)

[29] Yang, H. & Rose, K. Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in H. 264/AVC. *IEEE Transactions On Circuits And Systems For Video Technology*. **17**, 845-856 (2007)

[30] Liao, Y. & Gibson, J. Rate-distortion based mode selection for video coding over wireless networkswith burst losses. *2009 17th International Packet Video Workshop*. pp. 1-10 (2009)

[31] Wan, S. & Izquierdo, E. Rate-distortion optimized motion-compensated prediction for packet loss resilient video coding. *IEEE Transactions On Image Processing*. **16**, 1327-1338 (2007)

[32] Xiao, J., Tillo, T., Lin, C. & Zhao, Y. Joint redundant motion vector and intra macroblock refreshment for video transmission. *EURASIP Journal On Image And Video Processing*. **2011**, 1-11 (2011)

[33] Yang, H. & Rose, K. Optimizing motion compensated prediction for error resilient video coding. *IEEE Transactions On Image Processing*. **19**, 108-118 (2009)

[34] Shu, H. & Chau, L. Intra/inter macroblock mode decision for error-resilient transcoding. *IEEE Transactions On Multimedia*. **10**, 97-104 (2007)

[35] Li, B., Nanjundaswamy, T. & Rose, K. An error-resilient video coding framework with soft reset and end-to-end distortion optimization. *2017 IEEE International Conference On Image Processing (ICIP)*. pp. 1910-1914 (2017)

[36] Wang, T., Li, F. & Cosman, P. H. 265/HEVC video coding over lossy networks: Flexible or fixed mode in one CTU?. *IEEE Access*. **6** pp. 71279-71284 (2018)

[37] Wang, T., Li, F., Qiao, X. & Cosman, P. Low-Complexity Error Resilient HEVC Video Coding: A Deep Learning Approach. *IEEE Transactions On Image Processing*. **30** pp. 1245-1260 (2020)

[38] Kulupana, G., Talagala, D., Arachchi, H. & Fernando, A. End user video quality prediction and coding parameters selection at the encoder for robust HEVC video transmission. *IEEE Transactions On Circuits And Systems For Video Technology*. **29**, 3367-3381 (2018)

[39] Kazemi, M., Ghanbari, M. & Shirmohammadi, S. Intra coding strategy for video error resiliency: behavioral analysis. *IEEE Transactions On Multimedia*. **22**, 2193-2206 (2019)

[40] Kazemi, M. In favor of fully intra coding for HEVC video transmission over lossy channels. *Signal, Image And Video Processing*. **15**, 165-173 (2021)

[41] Kazemi, M., Iqbal, R. & Shirmohammadi, S. A selective intra-coding approach for multiple description video coding. *2016 IEEE International Symposium On Multimedia (ISM)*. pp. 301-306 (2016)

[42] Kazemi, M., Iqbal, R. & Shirmohammadi, S. Joint Intra and Multiple Description Coding for Packet Loss Resilient Video Transmission. *IEEE Transactions On Multimedia*. **20**, 781-795 (2018,4), Conference Name: IEEE Transactions on Multimedia

[43] Schierl, T., Hannuksela, M., Wang, Y. & Wenger, S. System layer integration of high efficiency video coding. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1871-1884 (2012)

[44] Sjoberg, R., Chen, Y., Fujibayashi, A., Hannuksela, M., Samuelsson, J., Tan, T., Wang, Y. & Wenger, S. Overview of HEVC high-level syntax and reference picture management. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1858-1870 (2012)

[45] Wenger, S. H. 264/avc over ip. *IEEE Transactions On Circuits And Systems For Video Technology*. **13**, 645-656 (2003)

[46] Wang, Y., Skupin, R., Hannuksela, M., Deshpande, S., Drugeon, V., Sjöberg, R., Choi, B., Seregin, V., Sanchez, Y., Boyce, J. & Others The high-level syntax of the versatile video coding (VVC) standard. *IEEE Transactions On Circuits And Systems For Video Technology*. **31**, 3779-3800 (2021)

[47] Lambert, P., De Neve, W., Dhondt, Y. & Walle, R. Flexible macroblock ordering in H. 264/AVC. *Journal Of Visual Communication And Image Representation*. **17**, 358-375 (2006)

[48] Piñol, P., Martinez-Rach, M., Garrido, P., Lopez-Granado, O. & Malumbres, M. Error resilient coding techniques for video delivery over vehicular networks. *Sensors*. **18**, 3495 (2018)

[49] Luby, M. "Broadcast Delivery of Multimedia Content to Mobile Users". (Qualcomm, Technical Report,2012)

[50] Luby, M.; Shokrollahi, A.; Watson, M.; Stockhammer, T.; Minder, L. RaptorQ Forward Error Correction Scheme for Object Delivery; IETF RMT Working Group, RFC 6330, IETF: Santa Clara, CA, USA, 1 August 2011. Available online: "`https://www.rfc-editor.org/rfc/pdfrfc/rfc6330.txt.pdf`", (accessed on 25 January 2023).

[51] Garrido Abenza, P., Malumbres, M., Piñol, P. & López-Granado, O. Source coding options to improve HEVC video streaming in vehicular networks. *Sensors*. **18**, 3107 (2018)

[52] Shannon, C. A mathematical theory of communication. *The Bell System Technical Journal*. **27**, 379-423 (1948)

[53] Vembu, S., Verdu, S. & Steinberg, Y. The source-channel separation theorem revisited. *IEEE Transactions On Information Theory*. **41**, 44-54 (1995)

[54] Lin, J., Zhang, Y., Li, N. & Jiang, H. Joint Source-Channel Decoding of Polar Codes for HEVC-Based Video Streaming. *ACM Transactions On Multimedia Computing, Communications, And Applications (TOMM)*. **18**, 1-23 (2022)

[55] Ma, Z. & Sun, S. Research on HEVC screen content coding and video transmission technology based on machine learning. *Ad Hoc Networks*. **107** pp. 102257 (2020)

[56] Liu, Z., Ishihara, S., Cui, Y., Ji, Y. & Tanaka, Y. JET: Joint source and channel coding for error resilient virtual reality video wireless transmission. *Signal Processing*. **147** pp. 154-162 (2018)

[57] Maung Maung, H., Aramvith, S. & Miyanaga, Y. Hierarchical-P reference picture selection based error resilient video coding framework for high efficiency video coding transmission applications. *Electronics*. **8**, 310 (2019)

[58] Alfaqheri, T. & Sadka, A. Low delay error resilience algorithm for H. 265| HEVC video transmission. *Journal Of Real-Time Image Processing*. **17** pp. 2047-2063 (2020)

[59] Bross, B., Wang, Y., Ye, Y., Liu, S., Chen, J., Sullivan, G. & Ohm, J. Overview of the versatile video coding (VVC) standard and its applications. *IEEE Transactions On Circuits And Systems For Video Technology*. **31**, 3736-3764 (2021)

[60] Chen, Y., Murherjee, D., Han, J., Grange, A., Xu, Y., Liu, Z., Parker, S., Chen, C., Su, H., Joshi, U. & Others An overview of core coding tools in the AV1 video codec. *2018 Picture Coding Symposium (PCS)*. pp. 41-45 (2018)

[61] Wiegand, T., Sullivan, G., Bjontegaard, G. & Luthra, A. Overview of the H. 264/AVC video coding standard. *IEEE Transactions On Circuits And Systems For Video Technology*. **13**, 560-576 (2003)

[62] Wien, M. High efficiency video coding. *Coding Tools And Specification*. **24** (2015)

[63] Mansri, I., Doghmane, N., Kouadria, N., Harize, S. & Bekhouch, A. Comparative evaluation of VVC, HEVC, H. 264, AV1, and VP9 encoders for low-delay video applications. *2020 Fourth International Conference On Multimedia Computing, Networking And Applications (MCNA)*. pp. 38-43 (2020)

[64] "`https://jvet.hhi.fraunhofer.de/`, [Online; Accessed 2023-01-26].

[65] Hm reference software.16.20,"`https://hevc.hhi.fraunhofer.de/`", [Online; Accessed 2023-01-26].

[66] "`http://x265.org`", [Online; Accessed 2023-01-26].

[67] Jm reference software. 19,"`http://iphome.hhi.de/suehring/tml/`", [Online; Accessed 2023-01-26].

[68] "`https://www.videolan.org/developers/x264.html`", [Online; Accessed 2023-01-26].

[69] Grois, D., Marpe, D., Nguyen, T. & Hadar, O. Comparative assessment of H. 265/MPEG-HEVC, VP9, and H. 264/MPEG-AVC encoders for low-delay video applications. *Applications Of Digital Image Processing XXXVII*. **9217** pp. 207-216 (2014)

[70] Řeřábek, M., Hanhart, P., Korshunov, P. & Ebrahimi, T. Quality evaluation of HEVC and VP9 video compression in real-time applications. *2015 Seventh International Workshop On Quality Of Multimedia Experience (QoMEX)*. pp. 1-6 (2015)

[71] Yousfi, R., Omor, M., Damak, T., Ayed, M. & Masmoudi, N. JEM-post HEVC vs. HM-H265/HEVC performance and subjective quality comparison based on QVA metric. *2018 4th International Conference On Advanced Technologies For Signal And Image Processing (ATSIP)*. pp. 1-4 (2018)

[72] Grois, D., Nguyen, T. & Marpe, D. Performance comparison of AV1, JEM, VP9, and HEVC encoders. *Applications Of Digital Image Processing XL*. **10396** pp. 68-79 (2018)

[73] Laude, T., Adhisantoso, Y., Voges, J., Munderloh, M. & Ostermann, J. A comprehensive video codec comparison. *APSIPA Transactions On Signal And Information Processing*. **8** pp. e30 (2019)

[74] Ohm, J., Sullivan, G., Schwarz, H., Tan, T. & Wiegand, T. Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC). *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1669-1684 (2012)

[75] U. Joshi, "Senior developer," Google Groups AV1 Team, 2019.

[76] Hong, D., Horowitz, M., Eleftheriadis, A. & Wiegand, T. H. 264 hierarchical P coding in the context of ultra-low delay, low complexity applications. *28th Picture Coding Symposium*. pp. 146-149 (2010)

[77] Bossen, F. & Others Common test conditions and software reference configurations. *JCTVC-L1100*. **12** (2013)

[78] ITU-T Subjective Video Quality Assessment Methods for Multimedia Applications. (International Telecommunication Union,2008,4)

[79] Huynh-Thu, Q. & Ghanbari, M. Scope of validity of PSNR in image/video quality assessment. *Electronics Letters*. **44**, 800-801 (2008)

[80] Bjontegaard, G. Calculation of average PSNR differences between RD-curves. *ITU SG16 Doc. VCEG-M33*. (2001)

[81] "`https://www.ultraedit.com/`, [Online; Accessed 2023-01-26].

[82] S. Wenger, Nal unit loss software, "`http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=4373`", Document JCTVC-H0072. 2012, [Online; Accessed 2023-01-26].

[83] S. Wenger, Proposed Error Pattern Files for JCT-VC, Document JCTVC-G150. 2011.

[84] Nal unit loss software, "`https://github.com/IslemMansri/LossTest`", [Online; Accessed 2023-01-26].

[85] Haßlinger, G. & Hohlfeld, O. The Gilbert-Elliott model for packet loss in real time services on the Internet. *14th GI/ITG Conference-Measurement, Modelling And Evalutation Of Computer And Communication Systems*. pp. 1-15 (2008)

[86] Chamitha de Alwis (2022). Generate Packet Loss Patters using Gilbert Elliot Model, "https://www.mathworks.com/matlabcentral/fileexchange/38554-generate-packet-loss-patters-using-gilbert-elliot-model", MATLAB Central File Exchange. Retrieved December 28, 2022

[87] "https://www.nsnam.org/, [Online; Accessed 2023-01-26].

[88] "https://github.com/lte-sim, [Online; Accessed 2023-01-26].

[89] Klaue, J., Rathke, B. & Wolisz, A. Evalvid–a framework for video transmission and quality evaluation. *Computer Performance Evaluation. Modelling Techniques And Tools: 13th International Conference, TOOLS 2003, Urbana, IL, USA, September 2-5, 2003. Proceedings 13*. pp. 255-272 (2003)

[90] Piro, G., Grieco, L., Boggia, G., Capozzi, F. & Camarda, P. Simulating LTE cellular systems: An open-source framework. *IEEE Transactions On Vehicular Technology*. **60**, 498-513 (2010)

[91] "https://github.com/IslemMansri/shell-script-/blob/main/MySim.sh", [Online; Accessed 2023-01-26].

[92] Jassal, A. H. 265/HEVC video transmission over 4G cellular networks. (University of British Columbia,2016)

[93] Sohi, R. Performance Comparison of H. 264/AVC and HEVC Standards Over LTE Networks. (Simon Fraser University,2015)

[94] MacAulay, A., Felts, B. & Fisher, Y. Whitepaper–ip streaming of mpeg-4: Native rtp vs mpeg-2 transport stream. *Envivio Inc Whitepaper On Powering MPEG-4 Applications From Mobile To HD*. (2005)

[95] Schierl, T., Gruneberg, K. & Wiegand, T. Scalable video coding over RTP and MPEG-2 transport stream in broadcast and IPTV channels. *IEEE Wireless Communications*. **16**, 64-71 (2009)

[96] Wang, Y., Wenger, S. & Hannuksela, M. Common conditions for SVC error resilience testing. *ISO/IEC JTC1/SC29/WG11 And ITU-T SG16 Q*. **6** pp. 7-10 (2005)

[97] Panayides, A., Pattichis, M., Pantziaris, M., Constantinides, A. & Pattichis, C. The battle of the video codecs in the healthcare domain-a comparative performance evaluation study leveraging VVC and AV1. *IEEE Access*. **8** pp. 11469-11481 (2020)

[98] Nguyen, T., Wieckowski, A., Bross, B. & Marpe, D. Objective Evaluation of the Practical Video Encoders VVenC, x265, and aomenc AV1. *2021 Picture Coding Symposium (PCS)*. pp. 1-5 (2021)

[99] Liu, C., Ma, R. & Zhang, Z. Error concealment for whole frame loss in HEVC. *Advances On Digital Television And Wireless Multimedia Communications: 9th International Forum On Digital TV And Wireless Multimedia Communication, IFTC 2012, Shanghai, China, November 9-10, 2012. Proceedings*. pp. 271-277 (2012)

[100] Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions On Image Processing*. **13**, 600-612 (2004)

[101] FFMPEG decoder,"https://www.ffmpeg.org/download.html", [Online; Accessed 2023-01-26].

[102] Katsaggelos, A., Galatsanos, N., Girod, B. & Färber, N. Error-resilient standard-compliant video coding. *Signal Recovery Techniques For Image And Video Compression And Transmission*. pp. 175-197 (1998)

[103] Mansri, I., Kouadria, N., Doghmane, N., Harize, S. & Bekhouch, A. Reference picture selection with decreased temporal dependency for HEVC error resilience. *Journal Of Visual Communication And Image Representation*. **90** pp. 103724 (2023)

[104] Ren, G., Li, P. & Wang, G. A novel hybrid coarse-to-fine digital image stabilization algorithm. *Information Technology Journal*. **9**, 1390-1396 (2010)

[105] Maung, H., Aramvith, S. & Miyanaga, Y. Region-of-interest based error resilient method for HEVC video transmission. *2015 15th International Symposium On Communications And Information Technologies (ISCIT)*. pp. 241-244 (2015)

[106] Maung, H., Aramvith, S. & Miyanaga, Y. Improved region-of-interest based rate control for error resilient HEVC framework. *2016 IEEE International Conference On Digital Signal Processing (DSP)*. pp. 286-290 (2016)

[107] Fukunaga, S., Nakai, T. & Inoue, H. Error resilient video coding by dynamic replacing of reference pictures. *Proceedings Of GLOBECOM'96. 1996 IEEE Global Telecommunications Conference*. **3** pp. 1503-1508 (1996)

[108] Hong, D., Horowitz, M., Eleftheriadis, A. & Wiegand, T. H.264 hierarchical P coding in the context of ultra-low delay, low complexity applications. *28th Picture Coding Symposium*. pp. 146-149 (2010,12), [doi:10.1109/PCS.2010.5702445]

[109] Joint Call for Proposals on Video Compression Technology,ITU-T SG16/Q6 document VCEG-AM91 and ISO/IEC MPEG Document N11113, ITU-T and ISO/IEC JTC 1. 2010

[110] Elecard streameye studio, "https://www.elecard.com/products/video-analysis/streameye-studio, [Online; Accessed 2023-01-26].

[111] Yang, J., Yin, Z. & Tang, T. Adaptive Intra Refresh for Screen Content Video Transmission in Dynamic Network. *Mobile Networks And Applications*. (2023,1,25), https://doi.org/10.1007/s11036-023-02101-1

[112] ITU-T Rec H.265 SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS Infrastructure of audiovisual services – Coding of moving video. (International Telecommunication Union,2019)

[113] Kim, I., Min, J., Lee, T., Han, W. & Park, J. Block partitioning structure in the HEVC standard. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1697-1706 (2012)

[114] Misra, K., Segall, A., Horowitz, M., Xu, S., Fuldseth, A. & Zhou, M. An overview of tiles in HEVC. *IEEE Journal Of Selected Topics In Signal Processing*. **7**, 969-977 (2013)

[115] Rodriguez-Sanchez, R. & Quintana-Ortıi, E. Tiles-and WPP-based HEVC decoding on asymmetric multi-core processors. *2017 IEEE Third International Conference On Multimedia Big Data (BigMM)*. pp. 299-302 (2017)

[116] Lainema, J., Bossen, F., Han, W., Min, J. & Ugur, K. Intra coding of the HEVC standard. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1792-1801 (2012)

[117] Nguyen, T. & Marpe, D. Performance analysis of HEVC-based intra coding for still image compression. *2012 Picture Coding Symposium*. pp. 233-236 (2012)

[118] Sjoberg, R., Chen, Y., Fujibayashi, A., Hannuksela, M., Samuelsson, J., Tan, T., Wang, Y. & Wenger, S. Overview of HEVC High-Level Syntax and Reference Picture Management. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1858-1870 (2012,12), [doi:10.1109/TCSVT.2012.2223052]

[119] Helle, P., Oudin, S., Bross, B., Marpe, D., Bici, M., Ugur, K., Jung, J., Clare, G. & Wiegand, T. Block merging for quadtree-based partitioning in HEVC. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1720-1731 (2012)

[120] Sole, J., Joshi, R., Nguyen, N., Ji, T., Karczewicz, M., Clare, G., Henry, F. & Duenas, A. Transform coefficient coding in HEVC. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1765-1777 (2012)

[121] Saxena, A. & Fernandes, F. DCT/DST-based transform coding for intra prediction in image/video coding. *IEEE Transactions On Image Processing*. **22**, 3974-3981 (2013)

[122] Sze, V. & Budagavi, M. High throughput CABAC entropy coding in HEVC. *IEEE Transactions On Circuits And Systems For Video Technology*. **22**, 1778-1791 (2012)

[123] "`https://github.com/Telecommunication-Telemedia-Assessment/SITI`, [Online; Accessed 2023-01-26].

[124] "`https://github.com/slhck/siti`, [Online; Accessed 2023-01-26].

[125] "`https://github.com/NabajeetBarman/SI-TI`, [Online; Accessed 2023-01-26].

[126] Mercat, A., Mäkinen, A., Sainio, J., Lemmetti, A., Viitanen, M. & Vanne, J. Comparative rate-distortion-complexity analysis of VVC and HEVC video codecs. *IEEE Access*. **9** pp. 67813-67828 (2021)

Appendix A

HEVC Video Dataset (Test Sequences)

Generally, in the design stage of video codecs, the corresponding institutions start with selecting test sequences that are appropriate for benchmarking purposes [62]. Therefore, during the development of HEVC, the Joint Collaborative Team on Video Coding (JCT-VC) defines common test conditions as described in [77]. In addition, it provides a video database with a set of 24 video samples of various content and various resolutions. Indeed, the JCT-VC specifies a class terminology for the videos, from class A to class F. The spatial resolutions in these classes range from (416 $\times$ 240) to (2560 $\times$ 1600). While the temporal resolutions are 24 fps, 30 fps, 50 fps and 60 fps with a 10s sequence length for each video. The selection of relevant categories enables us to investigate the various effects elicited in real-world scenarios. Indeed, the chosen spatial resolution for videos is highly related to the desired video application. Thus, class A video sequences with a resolution of (2560 $\times$ 1600) are appropriate for editing applications or when evaluating the efficiency of ultra-high definition broadcasts. Note that class A videos are cropped regions from ultra-high resolution. However, Class B sequences are used for HDTV applications with a resolution of (1920 $\times$ 1080). As stated in [74, 62], Class C (832 $\times$ 480) and class D (416 $\times$ 240) video sequences are appropriate when evaluating mobile applications. However, for video surveillance, it is advised to work with resolutions greater or equal to (832 $\times$ 480p). Indeed, for real-time applications in general (low-delay configuration) and according to the test conditions provided by the JC-TVC [77], all classes except class A can be used for testing low-delay configuration with an emphasis on class E and Class E' for low-delay and interactive applications [74]. Note that, the resolution for classes E and E' is (1280 $\times$ 720) pixels. For the class F category, videos are of different sizes namely: (832 $\times$ 480), (1024 $\times$ 768), and (1280 $\times$ 720). This class is devoted to evaluating video applications in which the actual data is not captured using a camera but is computer-generated (screen content applications).

Owing to the fact that the selected test sequences should be general and cover most cases, the ITU recommended the use of Spatio-Temporal information [78] to describe the video content nature i.e., the amount of spatial complexity as well as the amount of temporal complexity available in a scene.

Furthermore, the ITU recommends that the selected test sequences shall cover the maximum SI/TI plan. Indeed, the spatial and temporal complexity of content can be calculated using the metrics: Spatial Information (SI) and Temporal Information (TI). Usually, the SI will be high when there is a high amount of spatial detail in a scene. In other words, when high textures are in the scene the SI will be high. On the other hand, the same can be applied to the TI; for high-motion scenes or when a high amount of temporal changes in a video are available, the TI will be high.

The SI and TI can be calculated based on Eq. (A.1) and Eq. (A.3) respectively [78].

$$SI = max_{time}\left\{std_{space}\left[Sobel\left(F_n\right)\right]\right\} \tag{A.1}$$

Where F_n is the frame (luminance plane) at time n. After filtering each frame using the *Sobel* filter, the standard deviation for that particular result (filtered frame pixels) is calculated. Finally, the maximum std value of the time series for F_n frames will be taken as the actual SI.

The Temporal Information TI, is calculated upon the motion difference $M_n(i,j)$.

$$M_n(i,j) = F_n(i,j) - F_{n-1}(i,j) \tag{A.2}$$

Where $M_n(i,j)$ is the difference between the collocated pixel values $F_n(i,j)$ and $F_{n-1}(i,j)$ from two successive frames F_n and F_{n-1}. Accordingly, the TI is computed as the maximum std value of the time series for the std_{space} of $M_n(i,j)$ as follows:

$$TI = max_{time}\left\{std_{space}\left[M_n(i,j)\right]\right\} \tag{A.3}$$

However, to calculate the SI/TI for a given video sequence according to ITU-T P.910 [78], Pierre Lebreton, Werner Robitza and Steve Göring developed a command-line-based tool for windows using the C++ language [123]. Other Python and MATLAB Versions can be found in [124, 125].

Figure A.1 depicts the distribution of all the test sequences on the spatiotemporal SI/TI plan. While Tab. A.1 shows the video sequences according to their resolution, frames count, frame rate, bit-depth and the intended class for specific applications.

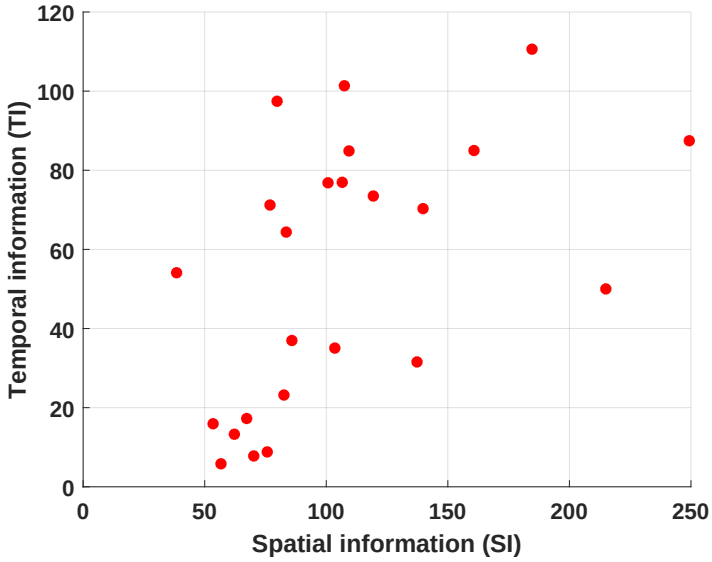


Figure A.1 – The spatial and temporal information indices of the test sequences

Table A.1 – All the test video sequences.

Class	Sequence name	Resolution	Frames count	Frame rate	Bit depth
A	Traffic	2560x1600	150	30fps	8
	PeopleOnStreet	2560x1600	150	30fps	8
	Nebuta	2560x1600	300	60fps	10
	SteamLocomotive	2560x1600	300	60fps	10
B	Kimono	1920x1080	240	24fps	8
	ParkScene	1920x1080	240	24fps	8
	Cactus	1920x1080	500	50fps	8
	BQTerrace	1920x1080	600	60fps	8
	BasketballDrive	1920x1080	500	50fps	8
C	RaceHorses	832x480	300	30fps	8
	BQMall	832x480	600	60fps	8
	PartyScene	832x480	500	50fps	8
	BasketballDrill	832x480	500	50fps	8
D	RaceHorses	416x240	300	30fps	8
	BQSquare	416x240	600	60fps	8
	BlowingBubbles	416x240	500	50fps	8
	BasketballPass	416x240	500	50fps	8
E	FourPeople	1280x720	600	60fps	8
	Johnny	1280x720	600	60fps	8
	KristenAndSara	1280x720	600	60fps	8
E’	Vidyo1	1280x720	600	60fps	8
	Vidyo3	1280x720	600	60fps	8
	Vidyo4	1280x720	600	60fps	8
F	BaskeballDrillText	832x480	500	50fps	8
	ChinaSpeed	1024x768	500	30fps	8
	SlideEditing	1280x720	300	30fps	8
	SlideShow	1280x720	500	20fps	8