Error Control Strategies in H.265|HEVC Video Transmission

By

Taha Tareq Alfaqheri

A thesis submitted for the degree of

Doctor of Philosophy

in

Electronic and Computer Engineering

School of Engineering Design and Physical Sciences

Brunel University London

February 2019

Abstract

With the rapid development in video coding technologies in the last decade, high-resolution video delivery suffers from packet loss due to unreliable transmission channels (time-varying characteristics). The error Resilience approaches at channel coding level are less efficient to implement in real time video transmission as the encoded video samples are in variable code length. Therefore, error resilience in video coding standard plays a vital role to reduce the effect of error propagation and improve the perceived visual quality. The main work in this thesis is to develop an efficient error resilience mechanism for H.265|HEVC video coding standard to reduce the effects of error propagation in error-prone conditions. In this thesis, two error resilience algorithms are proposed. The first one is Adaptive Slice Encoding (ASE) error resilience algorithm. The concept of this algorithm is to extract and protect the most active slices in the coded bitstream based on the adaptive search window. This algorithm can be applied in low delay video transmission with and without using a feedback channel. It is also designed to be compatible with reference coding software manual (HM16) for H.265|HEVC coding standard. The second proposed algorithm is a joint encoder-decoder error resilience called Error resilience based on Supplemental Enhancement Information (ERSEI) algorithm. A feedback message status is used from the decoder to notify the encoder to start encoding clean random-access picture adaptively based on the decoded picture hash message status from the decoder. At the same time, the decoder will be notified to start the error concealment process whilst waiting to receive correct video data. A recovery point message from the decoder feedback channel is used to update the encoder with error messages.

In this thesis, extensive experimental work, evaluation, and comparison with state-of-the-art related algorithms have been conducted to evaluate the proposed algorithms. Furthermore, the best trade-off between the coding efficiency of the proposed error resilience algorithms and error resilience performance has been considered at the design stage. The experimental work evaluation includes both encoding conditions, i.e. error-free and error-prone. The results achieved from the experiments show significant improvements, in (Y-PSNR) results and subjective quality of the decoded bitstream, using the proposed algorithm in error-prone conditions with a variety of packet loss rates. Moreover, experimental work is conducted to test the algorithms complexity in terms of required processing execution time at both encoding and decoding stages.

Additionally, the video coding standard performance for both H.264|AVC and H.265|HEVC coding standards are evaluated in error-free and error-prone environments. For ASE algorithm and when compared with improved region of interest (IROI) and region of interest (ROI) algorithms, a significant improvement in visual quality was the most obvious finding from the obtained results with PLRs of 2-18 (%).

For ERSEI algorithm and when compared with the default HM16 with pixel copy concealment and motion compensated error concealment (MCEC) techniques, the evaluation results indicate clear visual quality enhancement under different packet loss rates PLRs (1,2 6, 8) %.

Author's Declaration

I wish to state that I authored all the work in this thesis. The author's views expressed in this thesis are written to enhance Error Control Strategies in HEVC|H.265 Video Transmission. Brunel University is now authorised to make this thesis electronically available to the public.

Signature: ……………………………

Acknowledgements

All praise to Almighty Allah for giving me blessings and strength to complete my research and rewarding my hard work. I would like to heartily thank my Supervisor Professor Abdul Hamid Sadka. I owe him a great debt of gratitude for his great support and guidance during my research work journey. I feel very fortunate to have him as my educator and supervisor. I would also like to express my deep sense of gratitude to my Research Development Advisor Dr Lu Gan for her efforts and help offered me on my periodic reviews. Furthermore, I must sincerely thank all Brunel University staff especially the Post-graduate office and the Research Students for their great support and guidance.

I am extremely grateful to The Ministry of Higher Education and Scientific Research in Iraq for offering me this great opportunity to carry out my research and pursue my academic path that I always wanted. I do appreciate their support.

Last but not least, I would like to say thank you very much to my parents for their endless support and prayers, my brothers and sister, my dear wife and her family, and my daughter. I would have never achieved this without being embraced with your support and love.

Dedication

I want to dedicate this work to my parents, brothers and my dear sister, my wife and my daughter for their love and support.

Contents

List of Abbreviations

2D	2 Dimensional
3D	3 Dimensional
AMV	Average Motion Vector
AMVP	Advanced Motion Vector Prediction
ARQ	Automatic Repeat Request
ASE	Adaptive Slice Encoding
ATM	Asynchronous transfer mode
AVC	Advanced Video Coding
BD-BR	Bjontegaard Delta Bit Rate
BER	Bit Error Rate
bps	Bit per second
CABAC	Context-adaptive binary arithmetic coding
CIF	Common Intermediate Format
CPB	Coded Picture Buffer
CTB	Coding Tree Block
CTU	Coding tree unit
CV	Curve Vector
DCT	Discrete Cosine Transform
DASH	Dynamic Adaptive Streaming over HTTP
DP	Data Partitioning
DPB	Decoded Picture Buffer
DST	Discrete Sine Transform
EC	Error Concealment
ECSEI	Error Concealment Supplemental Enhancement Information
ER	Error resilience
EREC	Error Resilience Entropy Coding
ERSEI	Error Resilience based on Supplemental Enhancement Information
FEC	Forward Error Correction
FMO	Flexible Macroblock Ordering
GOP	Group Of Pictures
GPM	Gray Projection Method
HEVC	High Efficiency Video Coding
HM	HEVC Test Model
HRD	Hypothetical Reference Decoder
IEC	International Electrotechnical Commission
IROI	Improved Region Of Interest
ISO	International Standardisation Organisation
ITU-T	International Telecommunication Union-Telecommunication sector
JCT-VC	Joint Collaborative Team on Video Coding

JM	Joint test Model
Layer ID	Layer identification
MANE	Media Aware Network Element
MB	Macroblock
MCEC	Motion Compensated Error Concealment
MOS	Mean Opinion Score
DCR	Degradation Category Rating
MDC	Multiple Description Coding
MMS	Multimedia Messaging service
MPEG	Moving Picture Experts Group
CNN	Convolutional Neural Network
MVC	Multiview video Coding standard
NAL	Network Abstraction Layer
NS3	Network Simulator Version 3
PDA	Personal digital Assistant
PLR	Packet Loss Rate
POC	Picture Order Count
POCS	Projection Onto Convex Set
PPS	Picture Parameter Set
PSNR	Peak Signal to Noise Ratio
PU	Prediction Unit
QCIF	Quarter Common Intermediate Format
QoE	Quality of Experience
QoS	Quality of Service
R6bits	Reserved 6 bits
RAM	Random Access Memory
RD	Rate Distortion
ROI	Region Of Interest
RPS	Reference Parameter Set
RTP	Real-Time Transport Protocol
RVLC	Reversible Variable Length code
SAO	Sample Adaptive Offset
SPS	Sequence Parameter Set
SSCQE	Single Stimulus Continuous Quality Evaluation
SVC	Scalable Video Coding standard
UDP	User Datagram Protocol
UHD	Ultra-High Definition
VCL	Video Coding Layer
VLC	Variable Length Code
VPS	Video Parameter Set
VQM	Video Quality Metric
W-CDMA	Wideband Coding standard Division Multiple Access
WPP	Wavefront Parallel Processing
YUV	luma component (Y) and two chrominance (UV) component

List of Figures

Chapter One

1 Introduction

This chapter is organised as follows. Section 1.1 presents a background about the importance of video error resilience in video communications. A problem statement is introduced in section 1.2. In section 1.3, the main aim and objectives of the research study are summarised and discussed in this section, followed by a research contribution in section 1.4.

1.1 Background and Motivation

High quality video delivery is engaging research area nowadays in the multimedia communication sector. The rapid growth of video data traffic is expected to continue. The dramatic progress in the development of mobile communication systems has increased the demand of end users for video streaming applications [1]. A market report which was produced by Cisco in 2013 [2] stated that 57% of mobile data usage was for video streaming services, and in 2017 this was expected to grow to up to 69% [2]. Video traffic is expected to increase from 55% to 72% of the total world mobile data traffic between years 2014 and 2019 as demonstrated in Figure 1.1 [1]. Moreover, in 2021, video data traffic is expected to be more than 78% of the total world mobile data traffic in 2021 [3].

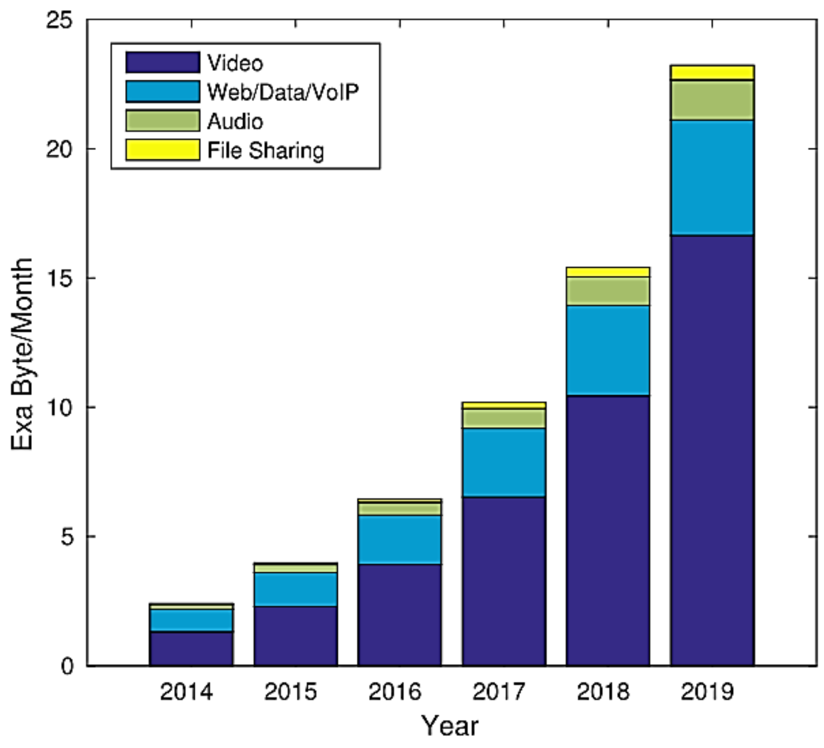


Figure 1.1 Global Mobile data traffic [1]

Under these circumstances, this yearly growth in video data traffic will cause congestion in the traditional video communication system as a result of limited bandwidth capacity and time delay [1]. Furthermore, employing retransmission mechanisms such as Automatic repeat request (ARQ) causes unacceptable delay in real time video transmission applications. Such error control mechanism makes it difficult to implement due to network flooding considerations. In that case, even implementing channel error control mechanisms, will lead to loss or corrupt of the sensitive compressed video bitstream at the receiver end, and in reality makes it difficult to recover [4]. Above all, issue has become more popular than before with the advancement of manufacturing technology of portable smart devices such as smartphones, laptops and tablets that support Ultra High Definition (UHD) video contents [4]. The increasing demand for using high definition video transmission with different applications restricts transmission channel bandwidth significantly. It is important to realise, when sending highly compressed video content in real time manner, with variable length codes on unreliable (time-varying characteristics) channels, a single bit error leads to severe visual quality degradation. Additionally, in some cases this leads to making the received video bitstream undecodable at the at the decoder side [5].

At present, the most efficient video coding standard is called High Efficiency Video Coding (H.265|HEVC) standard. The compression performance of this standard is double the coding efficiency of the previous H.264|AVC video standard [6]. The H.265|HEVC relies on flexible coding units called Coding Tree Units (CTUs) ranging from (16×16) pixels to (64×64) pixels size [6]. Higher coding unit size supports higher coding efficiency. However, The flexibility in coding units sizes in H.265|HEVC coding standard comes at the cost of the difficulty in the Coding standard adaptation with the previously implemented video coding error resilience techniques [7]. As a result, the implementation of efficient error resilience techniques either at encoding stage or jointly between the encoder and decoder are essential to reduce the effect of visual error propagation at H.265|HEVC decoder end. Therefore, in this thesis, we focus on proposing an efficient error resilience algorithm to reduce the effect of errors on the decoded visual quality. A default reference software for H.265|HEVC video coding standard, in addition to the related state of the art algorithms are chosen as a benchmark for performance evaluation and comparison of the proposed error resilience algorithms.

1.2 Problem statement

The main concern of this thesis is how to reduce the effects of different packet loss rates on the decoded visual quality in the H.265|HEVC video coding standard. The increasing requirements of end-to-end guaranteed video delivery for real-time transmission with UHD resolution video urged video coding experts in H.265|HEVC video coding standard to develop highly efficient video coding tools [8]. The H.265|HEVC video coding standard is the latest generation of collaborative work between International Telecommunication Union-Telecommunication Sector (ITU-T) and the International Standardization Organisation / International Electrotechnical Commission (ISO/IEC) organisation bodies [9].

The primary target of both organisations is to increase bitrate saving to more than fifty percent compared to the previous H.264|AVC coding standard [9], [10]. The high bitrate saving was achieved by adding new coding features to support more efficient coding for video resolution and to make it more friendly to parallel processing applications [9]. However, both professional bodies do not suggest efficient error resilience or concealment approaches to mitigate received error effects on decoded visual quality [11]. The main focus of this thesis is to improve decoded visual quality at the decoder side in error prone conditions in light with optimising the proposed work with coding efficiency and complexity overheads. Figure 1.2 demonstrates a user case scenario using H.265|HEVC error resilience tools as well as error concealment tools at the decoder side.

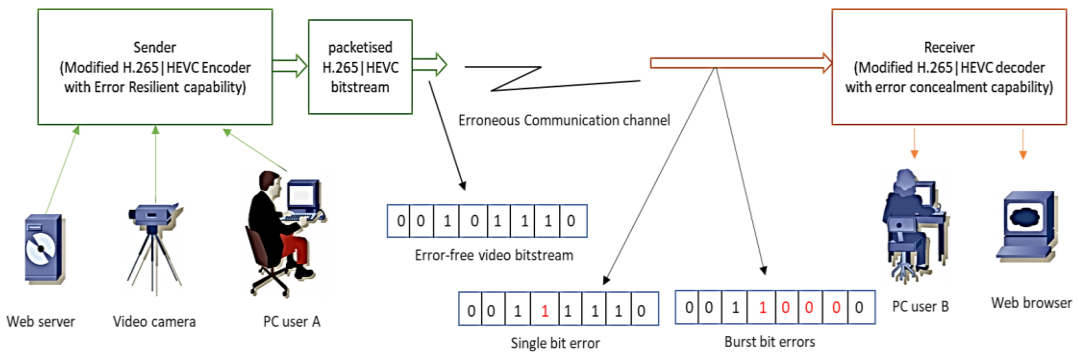

Sender
(Modified H.265|HEVC Encoder with Error Resilient capability)
packetised H.265|HEVC bitstream
Erroneous Communication channel
Receiver
(Modified H.265|HEVC decoder with error concealment capability)
Web server
Video camera
PC user A
0 0 1 0 1 1 1 0
Error-free video bitstream
0 0 1 1 1 1 1 0
Single bit error
0 0 1 1 0 0 0 0
Burst bit errors
PC user B
Web browser

Figure 1.2 H.265|HEVC error resilience implementation scenario

Thus, the primary research objective is to propose efficient error resilience algorithms compatible with H.265|HEVC video coding standard. The purpose of these algorithms is to reduce the effects of corrupted video data on the H.265|HEVC decoded video quality.

1.3 Aim and Objectives

The research study aims to design efficient error resilience algorithms based on the H.265|HEVC video coding standard system to reduce the effects of error propagation in error-prone conditions. The proposed algorithms should be optimised on finding the best trade-off between coding efficiency and error resilience performance in error-free and error-prone environments. The research objectives are listed below:

1. Produce efficient error resilience algorithms for H.265|HEVC video coding standard.
2. Evaluate the proposed error resilience algorithms with the relative state-of the-art work.
3. Propose H.265|HEVC video evaluation platform to support objective and subjective visual quality assessments in both error-prone and error-free conditions.
4. Evaluate the computational complexity of the proposed algorithms using the default reference software as a benchmark.
5. Analysis and comparison of coding performance of the current H.265|HEVC and previous H.264|AVC video coding standards in error-free and error-prone settings with various spatial resolutions.

1.4 Research contribution

The main achievements of this research are;

- Proposing error resilience algorithms in H.265|HEVC coding standard to reduce the effect of spatial and temporal error propagation under different packet loss rates.
- Optimising the proposed algorithms to achieve the best trade-off between encoding bitrate and error resilience performance in error-free and error-prone conditions.
- Comparing the decoded visual quality when using the same error-prone conditions in H.265|HEVC and H.264|AVC video coding standards with different video resolutions.
- Evaluating encoding/decoding error sensitivity of the proposed error resilience algorithms, and comparing and analysing the achieved execution time results with the default reference software (HM16).

1.5 Author publications

In this thesis, the research of H.265|HEVC Error resilience work has been that I have achieved has been addressed.

The paper that is published:

- Taha Alfaqheri and Abdul Hamid Sadka. "Low delay error resilience algorithm for H.265|HEVC video transmission ", RTIP-D-19-00175R2, DOI: 10.1007, Journal of Real-Time Image Processing, October 2019.

The papers currently being prepared for publication:

- Taha Alfaqheri and Abdul Hamid Sadka. "coding standard performance of H.265|HEVC and H.264|AVC coding standards in error-prone and error-free conditions ".
- Taha Alfaqheri and Abdul Hamid Sadka. "Joint Encoder-Decoder error resilience algorithm for H.265|HEVC video coding standard ".

1.6 Thesis outline

This thesis is organised into six chapters.

- Chapter One presents an introduction to the research work.
- Chapter Two provides review study for the commonly used error resilience techniques in the current and previous video coding standards. Encoding error resilience techniques at the encoder side and error concealment techniques as well as interactive error resilience techniques are described and discussed.
- Chapter Three describes H.265|HEVC video coding standard and its relative video coding tools which used to enhance the coding standard error resilience. Furthermore, a comparative study with H.264|AVC coding standard is also discussed highlighting coding performance in error prone conditions. Additionally, the evaluation tools and methodologies used in this research work is described. Then, the design and implementation of the adaptive slice encoding algorithm is presented.
- Chapter Four presents the proposed encoding error resilience work within H.265|HEVC video standard. Previous proposed related error resilience algorithms are described and evaluated with the proposed error resilience encoding work. The chapter begins with a detailed review of the error resilience tools used at the video encoder side. The related video coding tools such as frame partitioning and region of interest extraction tools are presented and discussed in detail. The proposed Adaptive slice encoding algorithm is described in detail. Finally, the chapter presents experimental evaluation results using frame by frame assessment method and objective metric analysis with its discussions. Finally, the chapter presents conclusion on the research work findings.
- Chapter Five presents the proposed joint encoder-decoder H.265|HEVC error resilience algorithm. The chapter defines the required encoding parameters set tools to be used to improve decoded visual quality at the receiver side. Then, the H.265|HEVC picture management concept is highlighted and discussed. The proposed joint encoder-decoder error resilience algorithm is described in detail followed by the obtained evaluation results with technical discussions. At the end of the chapter, conclusion of the research work is presented.
- Chapter six summarises and concludes the thesis highlighting the main research achievements and limitations together with future work recommendations.

1.7 Thesis Scope

In this research, error resilience techniques at the encoder or joint encoder-decoder error robustness techniques are the main approaches used to measure and reduce the corrupted H.265|HEVC encoded bitstream. The scope of this thesis is described as follows:

- A literature survey for the most relevant state of art error resilience techniques implemented in current (H.265|HEVC) and previous video coding standards, in addition to a detailed theoretical study on H.265|HEVC video coding algorithms highlighting a controversy in the H.265|HEVC error resilience study.
- Design of error resilience algorithms that can be implemented efficiently in the H.265|HEVC coding standard system.
- Evaluate the proposed encoder and joint encoder-decoder based error resilience algorithms in terms of the decoded visual quality and coding efficiency.
- Evaluate the complexity of the proposed algorithms in terms of encoding and decoding execution time.
- An evaluation platform that conforms to the H.265|HEVC bitstream structure, for evaluating and validating the proposed algorithms.
- Coding standard comparison studies for evaluating the performance of the H.265|HEVC video standard compared to H.264|AVC coding standard in error-free and error-prone conditions with different video resolutions.

Chapter Two

2 Literature Review

This chapter presents a general review of error Resilience approaches for robust video coding. The chapter is organised as follows. Section 2.1 provides introduction to robust video coding standards systems. Section 2.2 presents a brief review of previous video coding standard systems. The effect of spatial and temporal error propagation on the decoded visual quality is discussed in section 2.3. Section 2.4 discusses the basic concept of using error resilience in error-prone environments. A full review of the employed encoder-based error resilience algorithms is presented in section 2.5, while section 2.6 presents the most common error concealment techniques at the decoder side. Section 2.7 reviews the joint encoder-decoder error resilience techniques used in video coding standards. The latest error control techniques employed in H.265|HEVC video coding standards are presented in section 2.8 and section 2.9. A conclusion of the chapter is presented in section 2.10.

2.1 Introduction

Over the past two decades, there has been a dramatic increase in developing highly efficient video coding to support available and future visual communication systems. The non-guaranteed quality of service in real time video delivery presents a challenging task for both academia and industry sectors. Therefore, error resilience approaches at video coding layer is a major area of interest within the field of video communication systems.

In fact, the increase in video coding efficiency comes at the cost of increasing inter prediction and motion compensation processes. These mainly involved in increasing the number of temporal redundant information [12]. In addition to the high requirements of computation complexity, a highly compressed bitstream means more redundant video information encoded. As a result, compressed video content becomes more sensitive to channel bit errors. As a result, transmitting a highly compressed video bitstream in unreliable transmission channel leads to degradation of the perceived decoded visual quality or failure in the decoding process for the whole video sequence if errors hit the sensitive encoded data such as slice header [11]. Figure 2.1 demonstrates a video transmission issue on the received video quality when using an unreliable wireless channel.

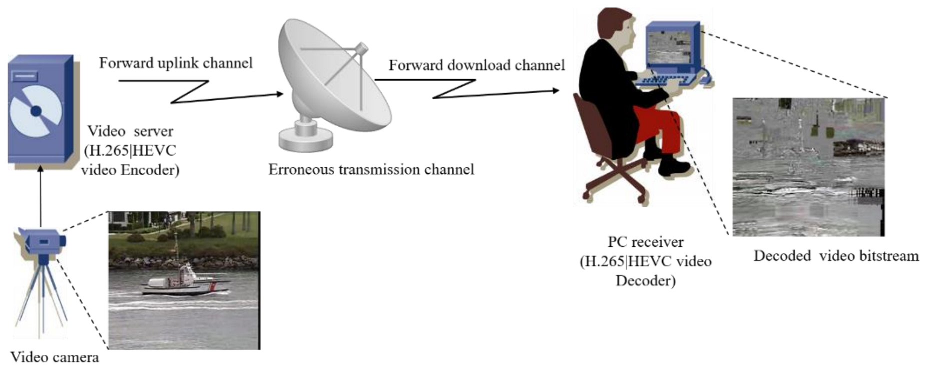


Figure 2.1 video transmission issue scenario

There are two main error control categories to reduce the effects of transmission errors on perceived visual quality. The first one employs traditional data error control methods which use lossless channel coding tools in data recovery such as ARQ error control schemes. However, implementing such error recovery tools in compressed video delivery is less efficient because the nature of the compressed bitstream is of a variable-length code, which makes error recovery of decoded video contents very challenging task.

To minimise the effects of the transmission errors efficiently at the video decoder side, the video error control can be divided into three approaches; forward error recovery, error concealment, and interactive error recovery approaches.

In forward error recovery approach, the video encoder takes the full responsibilities to insert redundant error resilience codes and makes the coded bitstream more robust against errors.

The second approach is error concealment techniques in which the decoder is responsible to conceal the errors spatially and temporally. The spatial error concealment employs correctly received information using interpolation techniques on the surrounded macroblocks. Or in case of complete macroblock information is lost, a simplest and most common concealment techniques are replacing the lost macroblocks with its same location in the previously decoded macroblocks. The other error concealment approach at the decoder is called temporal error concealment techniques which extrapolate the correctly received motion vectors from the current and previous decoded frames [13].

The third video error control approach is using joint encoder-decoder error resilience techniques. The encoding settings are adaptively change according to the received feedback update about network conditions or decoding process. In this approach, a backword feedback channel is used from the decoder to the encoder sides in the error recovery process.

2.2 Video Coding Standards (MPEGX, H.26X)

In general, developing digital video coding techniques plays an important role to support real-time video transmission applications and broadcast delivery [14]. The goal of developing video coding tools is to remove temporal and spatial redundant information from its original video contents. Nowadays, two main leading organisations are responsible for developing video coding standards namely; International Telecommunication Union Telecommunication sector (ITU-T) and International Standard Organisation /International Electrotechnical Commission (ISO/IEC) [15]. Both ISO/IEC and ITU-T organisations are aiming to increase encoding bit rate saving keeping the same perceived video quality. Some developers and researchers are working hard inside the JCT-VC team to maximise the compression ratio in video coding standards, while other researchers are focusing on improving the error robustness of video transmission in different network environments (as error robustness in H.265|HEVC is out of scope of the compression standard itself) [16].

The primary goal of video coding standard standardisation is to impose restrictions on conformance of the video bitstream and its syntax elements. These conformance restrictions are applied on the decoding video process to ensure the video coding standard compatibility with different decoding devices [17]. Figure 2.2 shows the scope of the bitstream conformance for video standardisation for the ITU-T and ISO/IET organisations [17].

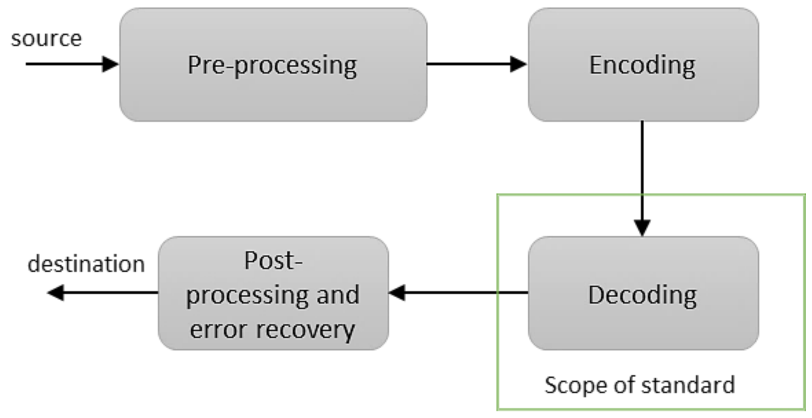


Figure 2.2 H.265|HEVC video standard scope

Additionally, both organisations are involved in collaborative work to generate MPEG-2/H.262, AVC/H.264 and H.265|HEVC video coding standards. Figure 2.3 demonstrates video coding standards generations starting from 1990 (H.261) and ending with the most recent H.265|HEVC video coding generation.

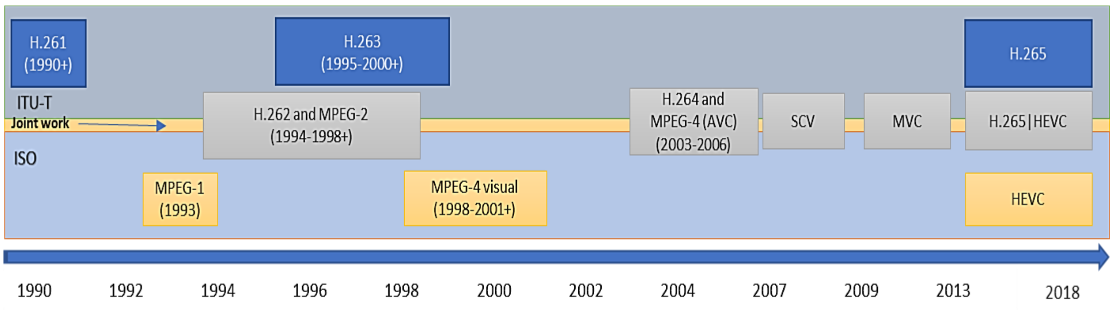


Figure 2.3 Video coding standards generations for ITU-T and ISO

The first video compression standard was originally developed by ITU-T body in 1984, then revised in 1986. During the late 1980s and beginning of 1990s, the two bodies worked in parallel targeting low delay video delivery applications and video broadcasting services. The first video coding standard is generated by Video Coding Experts Group (VCEG) created by ITU-T [18]. Then, in 1994, the ITU-T and ISO/IEC jointly worked on producing (H.262/MPEG-2) Video coding standard [14]. Between 1995 and 1996, the ITU-T organisation produced H.263 video coding standard. The primary objective of the standard was to support low bit rate video conferencing applications [19]. After that in 1998, the ISO/IEC organisation produced MPEG-4 visual coding standard [20]. In 2009, joint collaborative work started between both originations to produce .264/MPEG-4 Advanced Video Coding (AVC) and the standard was finalised in 2003.[21]. The developing standard work was extended in 2009 to support high definition video applications such as satellite TV broadcasting applications and real-time video delivery in internet and mobile networks [22].

A summary of the main video applications for each MPEGX and H.26X standards are summarised below:

- H.261:

H.261 is the first video standard member in H.26X series. The standard mainly developed to support Integrated Services Digital Network (ISDN) networks. It also supports audio-visual services at data rate of p×64 kb/s, where p is in the range between

1 and 30 (number of B-channels of ISDN) with bit rate ranging (40-2000)(Kbps) with bit depth chroma sampling 4:2:0 [23].

- H.262/MPEG-2:

The H.262/MPEG-2 produced in 1994 by VCEG from ITU-T and MPEG from ISO/IEC. Then the standard approved in 1995 [14]. The main applications of this video coding standard are: Satellite TV, DVD optical disc format, interlaced video for supporting NTSC, PAL, and SECAM TV systems. However, it does not support video applications bit rate lower than 1 Mbps [14] [24].

- H.263:

H.263 was produced to support low bit rates video compression for telephony and video conferencing applications. The first version of the standard was approved in 1996. In 1997, an error tracking tools were included in the standard to increase data robustness against data loss. However, it supports only limited picture sizes (up to 16CIF). Then, it extended into subsequent versions (H.263+ and H.263++). In H.263+, the video standard was improved in its error robustness capability by adding several annexes to the standard. The added annexes include: Slice Structured mode, Improved PB-frames mode, Reference picture resampling, and Independent Segment Decoding mode. In 2000, the standard was further extended (H.263++) with additional recommendation document to include: Data-partitioning at slice level, and enhanced reference picture selection mode [24].

- H.264|AVC:

The first H.264|AVC standard version was standardised in 2003. The main intent of using the standard was to double the bit rates saving of the previous H.263 standard keeping the same visual quality. The main improvements added to this standard were weighted prediction and motion estimation, as well as in-loop filtering and motion compensation process [25][26]. More advanced coding tools were included in this standard to improve coding efficiency compared to earlier video coding standards including H.261 [27], H.262 (MPEG-2), and H.263 [19]. It was also improved with more flexibility with multimedia services such as Multimedia messaging services (MMS) and wireless and mobile networks [26].

- H.265|HEVC:

The H.265|HEVC is the most recent video coding standard with capability to compress raw video data to half the size of the compressed file using H.264|AVC coding standard . The first meeting of Joint Collaborative Team on Video Coding (JCT-VC) between ITU and ISO bodies was held in January, 2010 and issued a call of proposal for future video standard work [28]. The first H.265|HEVC standard draft was approved for public in January, 2013, in ISO/IEC as MPEG-H Part 2, and in ITU-T as H.265 recommendations. The H.265|HEVC uses the same hybrid encoding architecture of previous standards. It has been designed to address two issues in the previous standards; increase video resolution applications and parallel processing capabilities. Further standard technical details will be discussed in Chapter Three.

2.3 Error propagation effects on video quality

Transmission errors in video bitstream can be classified into two types; random bit error and erasure error [29]. Random bit error results from deficiencies in physical channels which lead to bit insertion, bit deletion, and bit reversal [29]. Such errors could be corrected using video coding methods based on the contents of the received damaged video information [29]. Erasure error results from packet loss in packet networks with physical channels defects or system failure such as storage media [29].
All video coding standards produced by ITU and ISO bodies starting from H.261 standard are using hybrid encoding architecture to efficiently compress video data. A hybrid video coding tools uses intra and inter-frame prediction with 2D transform signal to generate a residual signal [25]. At the encoding stage, each frame is partitioned into a similar square shape of blocks. The coding standard system uses a decoder loop system; the loop signal allows the encoder and decoder to synchronise the same prediction signal as demonstrated in Figure 2.4.

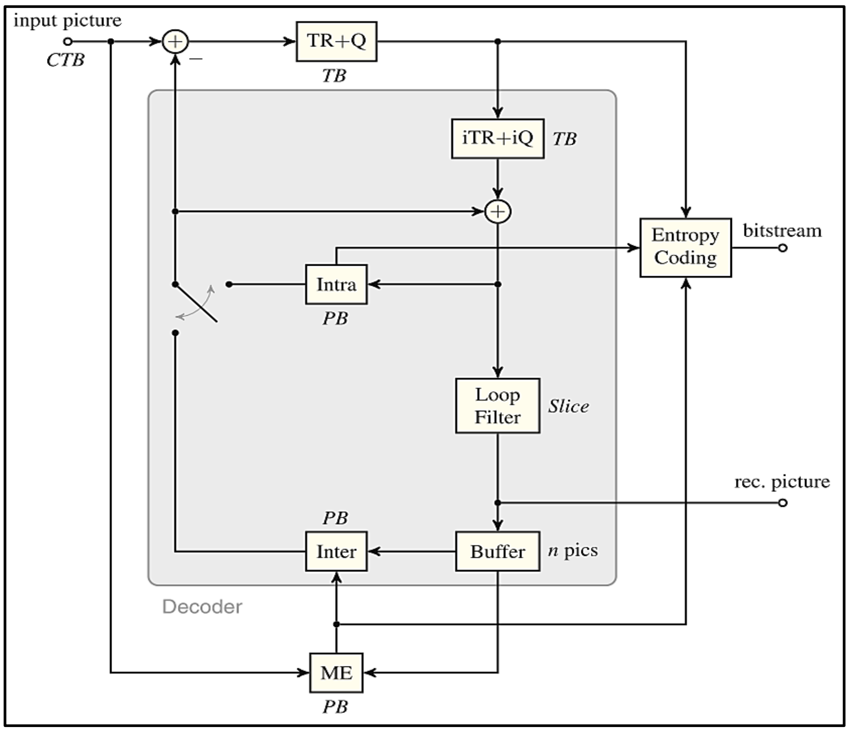


Figure 2.4 Hybrid block-based video coding block diagram [6]

Therefore, due to the nature of intra and inter prediction process, transmitting compressed video data on erroneous channels can have a detrimental effect on decoded video quality. This is because compressed video bitstream is encoded with variable length code (VLC) with high temporal and spatial predictions [30]. A single-bit error can corrupt a whole temporal prediction data block. It can also result in loss of synchronisation of the upcoming video samples at the decoder side [30] [31]. Thus, a small injected bit error can lead to loss of synchronisation of the upcoming video samples at the decoder side resulting in severe degradation in the perceived visual quality [31].
When transmitting a video bitstream with high compression rates, a compressed motion video data will be vulnerable to the loss of video information [32]. Figure 2.5 demonstrates a real visual example of temporal error propagation in the corrupted frame at time (t), the error effect is persistent with the same level of visual quality degradation on the next frame at a time (t+1). For this reason, motion compensation is the main contributor for error propagation.

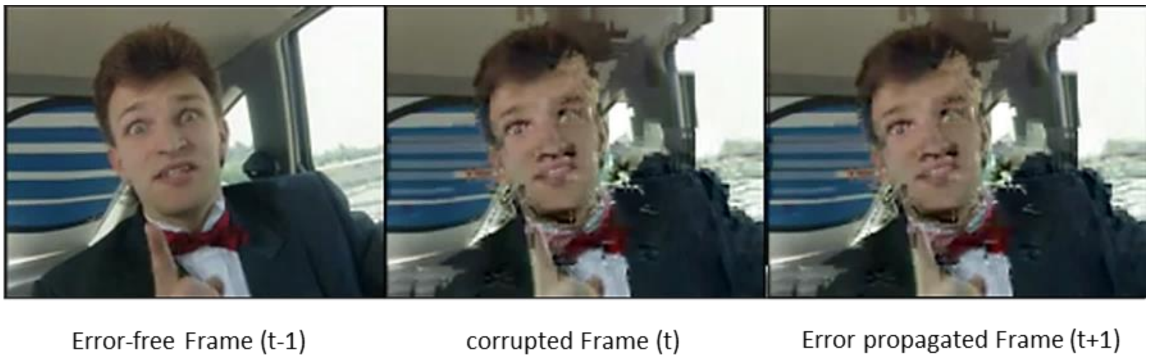


Figure 2.5 Effect of Error propagation on visual quality

To put it in another way, motion compensation is the main process used in the video compression. If motion compensated information is corrupted with errors, the corrupted motion data will propagate to all dependent motion compensated information [33].
One bit error can cause not only the decoder to drop a corrupted slice segment data. It can also leads to lose loss of synchronisation for the whole packet which increases error propagation effects on perceived visual quality [33]. A real visual example can be depicted in Figure 2.6 [34]. A single bit error hits sensitive video data related to the front players and background area. Accordingly, these injected errors cause severe damage to the decoded frame and as a result the damaged areas will be temporarily frozen for several seconds until receiving correct decoding update from the encoder.

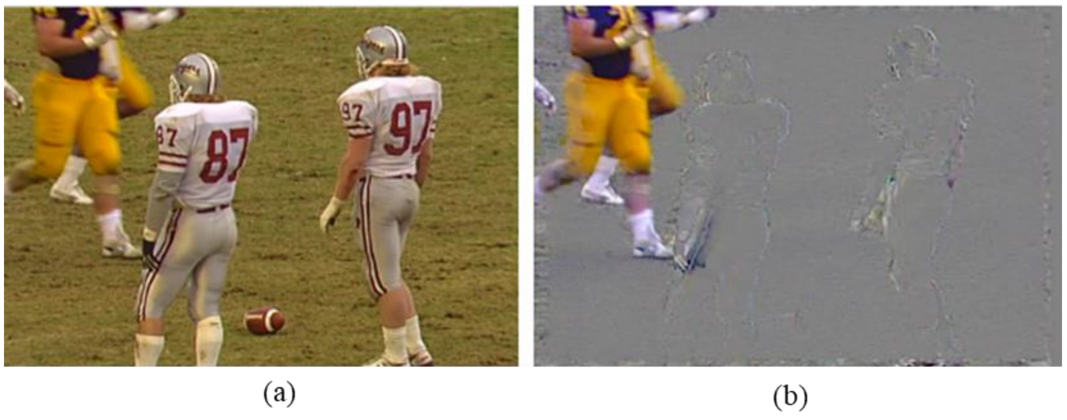


Figure 2.6 Error propagation effect caused by one-bit error
(a) original frame, (b) Erroneous frame [34]

In general, video coding standards with highly efficient temporal and spatial dependencies lead to reducing coding robustness against transmission errors.
As shown in Figure 2.7, block number 15 in the encoding frame at time (t) temporarily depends on four blocks (8, 9, 14, and 15) including the one in the same location of the

previously encoded frame at (t-1). If any bit error occurred in any of the four blocks, errors will affect all the corresponding neighbour blocks.

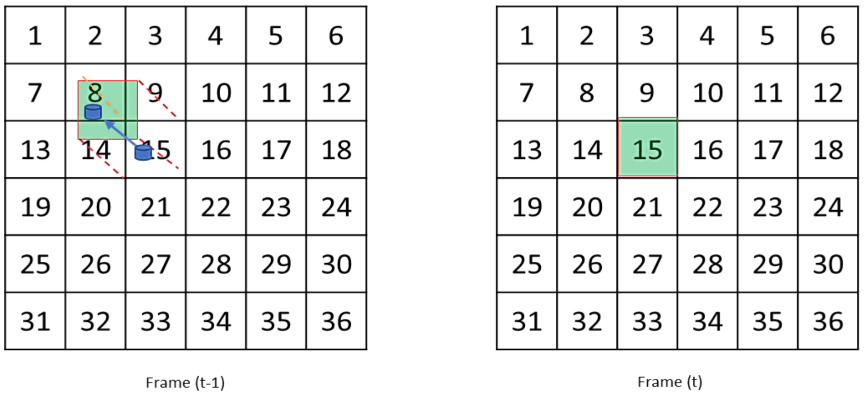


Figure 2.7 error propagation during the motion compensation process

Figure 2.8 features a simple illustration of the effect of spatiotemporal error propagation on decoded visual quality [32]. As it is depicted in the figure, the error occurred in frame 3. Visual quality degradation spread not only spatially due to the motion-compensated prediction, but also through temporal error propagation [35], [32].

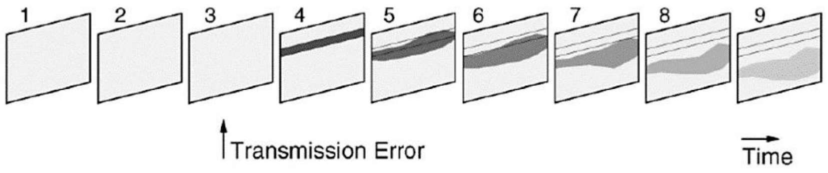


Figure 2.8: Spatiotemporal error propagation concept [32]

Other concealment techniques use switching between temporal and spatial methods based on correctly received Discrete Cosine Transform blocks and motion compensation vectors data [36].

One solution is to reduce the effect of error propagation by inserting cyclic resynchronisation words. Once resynchronisation between the encoder loop and local decoder is restored with a new error-free synchronisation word, the prediction errors will stop [33].

2.4 Error Resilience in Video Transmission

As mentioned earlier in the previous section, due to using unreliable transmission channels, a decoded video suffers from visual quality degradation. Thus, to keep the perceived visual quality at acceptable level in error-prone conditions, three video error

Resilience approaches can be implemented at various video processing stages [29]. Firstly, Forward Error Correction (FEC) technique where the encoder plays the primary role in improving error resilience of the compressed video bitstreams. Secondly, decoder-based error concealment approaches whereby the decoder plays the primary role in concealing the received damaged or lost macroblocks. Error concealment in this method relies on the previously received decoded video frames. The concealment process can involve temporal error concealment or spatial error concealment or switching between temporal and spatial. Thirdly, using error resilience algorithms at video encoder and decoder sides for more robust compressed video bitstream [37], [38]. The three approaches will be described in the following subsections.

2.5 Encoder based Error Resilience approaches

In general, because real time video communication systems are very sensitive to time delay, retransmission of corrupted slice segments does give acceptable results. Hence, one solution is to encode correction codes within encoded bitstream to help to recover errors or reduce its effects at the decoder side. Despite the fact that encoding error resilience techniques is very efficient solution to reduce error propagation in real time video applications, it comes at the cost of increasing processing complexity and bit overhead cost with a slight reduction in video coding efficiency. This section gives a review of the state-of-the-art encoding error resilience approaches used in video communication systems today.

2.5.1 Inserting synchronisation words technique

The concept of inserting synchronisation words is to divide a large redundant signal into smaller independently decodable video segments. In such a way, the error propagation can be reduced significantly [39]. A synchronisation word can be identified by unique designed codeword encoded within the coded bitstream. The addition of synchronisation words or markers can be divided locally (in each frame raw) with predefined row intervals. In this case, a synchronisation codeword will be added at the end of each scan line. Alternatively, it can be added at cyclic bits intervals (once every 256 bits). In this case, a video with high motion activity needs higher cyclic intervals than a video with low motion activity.

Then, at the receiver side, the decoder can detect the end of each block without requiring update knowledge from subsequent or preceding decoded block [32]. After checking a predefined number of decoded blocks, the decoder looks for the available synchronisation words [32]. However, if there is a bit error during the bitstream decoding process, the decoder will lose synchronisation, then the decoder should wait for a while until receiving the next synchronisation word [40]. Therefore, a large amount of correct data will be discarded at the synch word decoder as shown in Figure 2.9. Moreover, inserting periodic synchronisation words into the compressed bitstream results in a reduction of the coding efficiency due to the increased bit overhead [39].

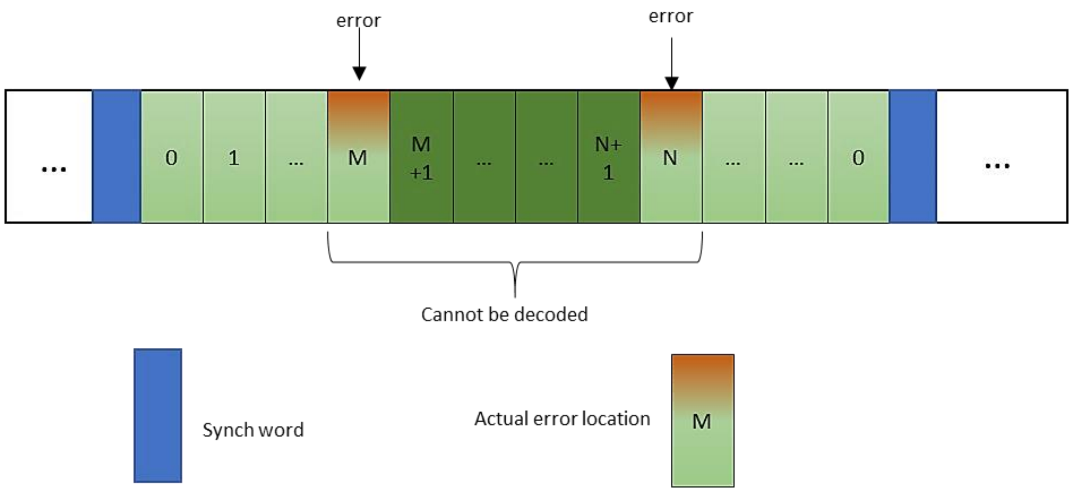


Figure 2.9 Inserting additional synchronisation words to a video data segment adapted from [40]

2.5.2 Data Partitioning (DP) Technique

A compressed video bitstream consists of various syntax elements. In error prone conditions, these syntax elements have different impacts on perceived visual quality. Therefore, depending on the importance of different bitstream elements, the encoder employs data partitioning on the video data to be protected with different levels of Unequal Protection Schemes (UEP) [41],[42]. The video data with high error sensitivity (i.e. motion data slice headers) is sent with a higher level of protection [41] [43]. Usually, UEP uses Forward Error Correction schemes at channel level.

One drawback of this technique is the inability to handle errors generated from the rapidly changing transmission channels conditions such as mobile channels [31].

The first time of using data partitioning in error resilience was in developing H.263video coding standard [44]. The video data are classified into different groups according to its relevant importance to the decoding process. In H.264|AVC coding standard, the DP

concept is further improved to be grouped into three main data partitions. The DP process is activated by nal−ref−idc (NRI) with two-bit codeword in its NAL units header to support different sub stream priority [45]. In H.265|HEVC coding standard, the DP encoding tools is not included in the design of NAL unit header structure [28].
In H.264|H.264|AVC, there are mainly three different data partitioning parts as described in the following [43]:

- Partition A: includes highest important compressed data such as slice header and motion vector information, which could lead to frame freezing or undecodable video frames in case such information are corrupted.
- Partition B: contains the intra coded video data (I mode) or transform coefficients therefore it will cause temporal error propagation until receiving correct intra coded video data.
- Partition C:contains inter coded video data (B or P mode) that are less error-sensitive than A and B data partitions [43].

Overall, DP technique in single channel transmission work effectively alongside with UEP protection schemes in such a way the most important partition is protected with high level of channel protection codes. In order to reduce error effect in case of multi-channel transmission, a DP technique with (Multiple Description Coding) schemes work effectively for higher error resilience, as it is discussed later in section (2.5.4).

2.5.3 Flexible Macroblock Ordering (FMO)

Another popular encoding error resilience technique named Flexible Macroblock Ordering (FMO) which was implemented for the first time in H.264|AVC coding standard [46]. In this technique, each macroblock is organised into specific slice group in predefined pattern. Different encoding patterns can be arranged by grouping each MB into slice group (SG) [46]. This technique aims to reduce the spatial error propagation at frame level by reordering the coding MBs and spreading the errors into larger regions [46]. So that, the human visual system cannot recognise the visual artefacts at the decoder. In H.264|AVC coding standard, there are six FMO types numbered with (0-5). When FMO is disabled, type 0 is activated refers to the default H.264|AVC encoding raster scanning setting, a horizontal scan pattern in every raw is used to encode each slice segment. Type 1 represented by checkerboard pattern with slice group 0 and 1. Type 2 employs one or more rectangular shaped slice groups alongside background slice group. MBO types (3-5) are flexible in shape with every picture, as demonstrated in Figure 2.10.

S.K. Im and A.J. Pearmain exploit FMO to offer a new classification algorithm for prioritised video transmission [47]. They employed new optimisation algorithm to classify different slice groups according to its importance using vulnerary factor for each macroblock [47]. Further research study revealed the finding of the weighting sum of the motion vectors to reduce the complexity of the macroblocks classification process [48].

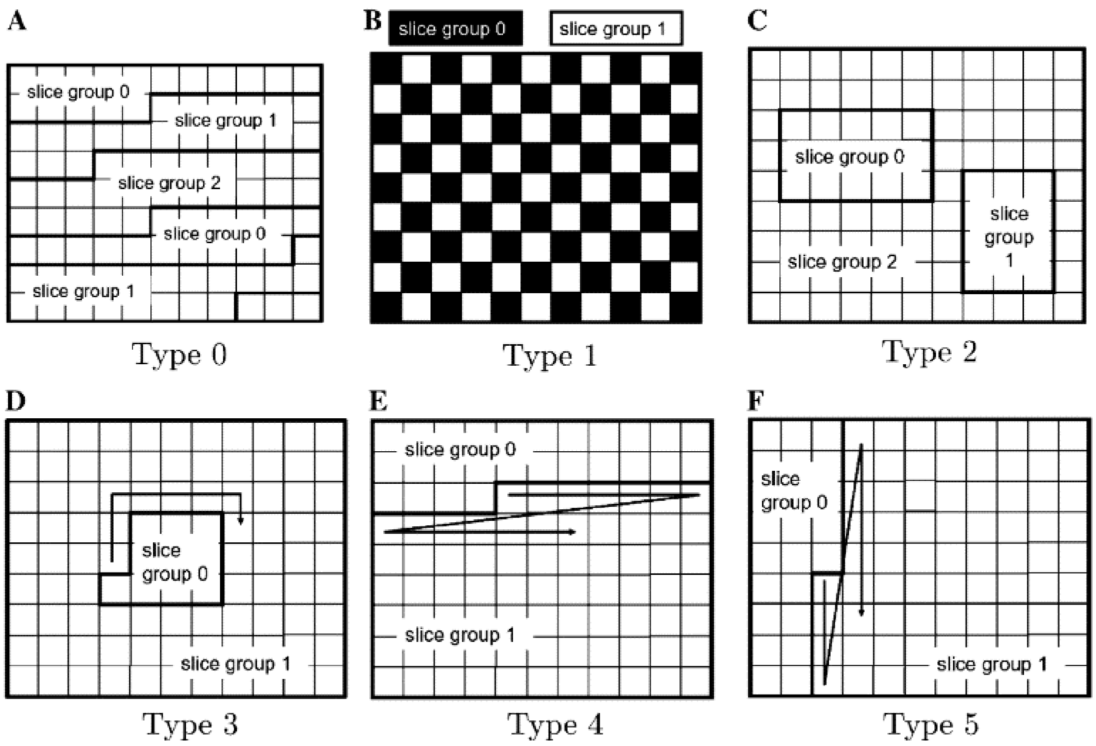


Figure 2.10 Flexible Macroblock Ordering (FMO) types

The authors in [49] reported the possibility of using FMO tool in regions of interest (ROI) applications. As an illustration, a rectangular slice groups represented by ROI area which consists of the main important frame parts to be encoded with higher quality while other areas are encoded with lower quality. A real example of FMO implementation can be demonstrated in Figure 2.11.

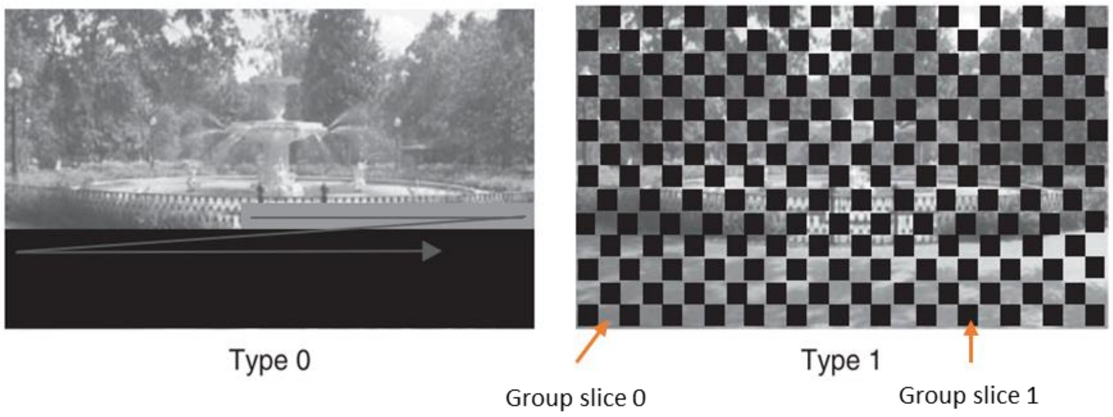


Figure 2.11 Slicing group maps for Flexible macroblock ordering (FMO) [46]

In type 1, each frame is encoded with two slice groups with checkerboard encoding scan pattern. The encoded pattern in type 1 gives better error recovery results than type 0 by increasing the probability of finding correctly received neighboured macroblocks [46]. However, the FMO technique does not support H.265|HEVC encoded bitstream because its basic coding unit, i.e. coding tree units (CTUs) are flexible in size [45].
The macroblocks ordering styles in H.264|AVC can be grouped into (8) slice groups. In a study of testing objective visual quality with (5%) packet loss rates over IP network, the achieved quality gain in average was 1.9 dB (as shown in Figure 2.12) [49].

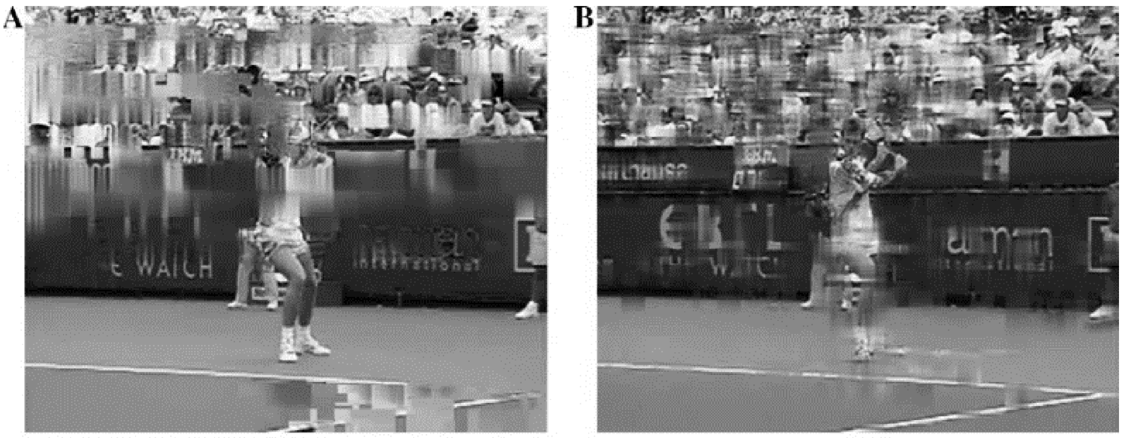


Figure 2.12 Stefan sequence at 11th frame subjected to 5% PLR [49]
A) without FMO, B) with FMO

According to their achieved results, the error resilience with activation FMO tool can stop propagated errors along the whole row for the reconstructed frames at slice level. Additionally, it helps to reduce the flickering effects at the decoding stage with achieving more pleasant visual quality.

2.5.4 Multiple Description Coding (MDC)

In this technique, the video encoder generates several subsets of encoded bitstreams named Descriptions (D) with the same importance to enhance error resilience [50]. The encoded descriptions are transmitted in different physical channels. In lossy network environments, when some of descriptions are corrupted with errors, the decoder is still able to decode the correctly received descriptions at the cost of lower perceived visual quality. An example of video transmission system using MDC is demonstrated in Figure 2.13 [51]. The video encoder in the figure generates four descriptions (D1-D4) using four separate channels.

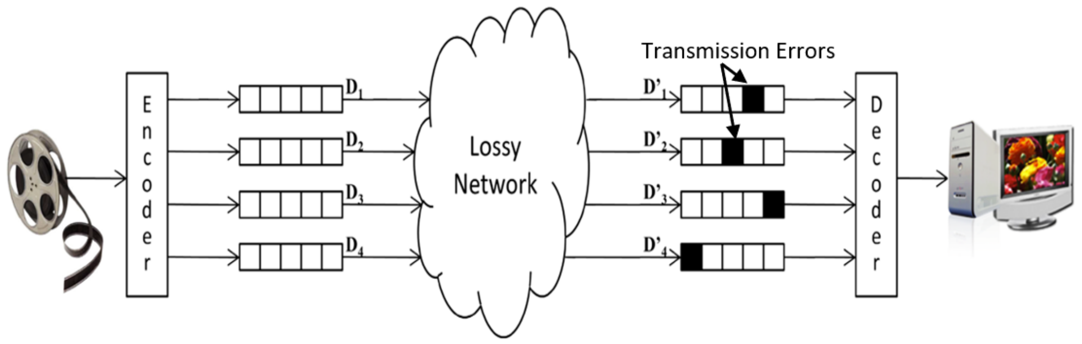


Figure 2.13 Error resilience using Multiple Description Coding (MDC) [51]

In MDC implementation, the probability of losing the same video data in separate channels is very low. Therefore, at the MDC decoder side, in case of receiving corrupted descriptions at specific time (referred as D' in the figure), the other correctly received descriptions are used in recovery process corrupted information [52]. Each subset of video stream is independently decodable [51]. In case of receiving all descriptions without errors, the MDC decoder mixes and decodes all the sub streams together with achieving higher visual quality [50].

2.5.5 Error Resilience Entropy Coding (EREC)

The EREC is a technique that maps the variable video codes into fixed-length codes [53]. This technique is first implemented by Algra in 1992 [54]. The length of the fixed codes is controlled by target video coding efficiency [53]. In the mapping process, additional synch words with fixed-length codes are inserted into compressed bitstream [53]. However, using entropy coding in error resilience can increase the compression overhead which negatively affect on the coding efficiency of video standard [53]. The fixed length coding algorithm can be calculated by the following equation [40]:

$$T - \sum_{i=1}^{N} b_i \geq 0 \qquad \text{Eq.(2.1)}$$

Where (T) is a total segment length. This must be transmitted in a high protection channel. (N) is the number of divided fixed-length segments, and (b_i) is the variable-length code. The N Value should be known in the decoder [40].

The conversion of the variable to the fixed-length process is demonstrated in Figure 2.14. N consists of six stages defined by predefined EREC slots at the encoder [33] , [53]. The VLC blocks are placed in the EREC structure [54]. The conversion process starts by allocating each variable-length block to a fixed EREC slot [54]. As shown in Figure 2.14, In stage 1, if the VLC length (b) is less than EREC slots (s), then the slots are encoded with full length. In stage 2, the unused EREC slots are filled with closest variable length code (b). In the last stage, each available data block that yet to is encoded is allocated to the available EREC space [33] , [54].

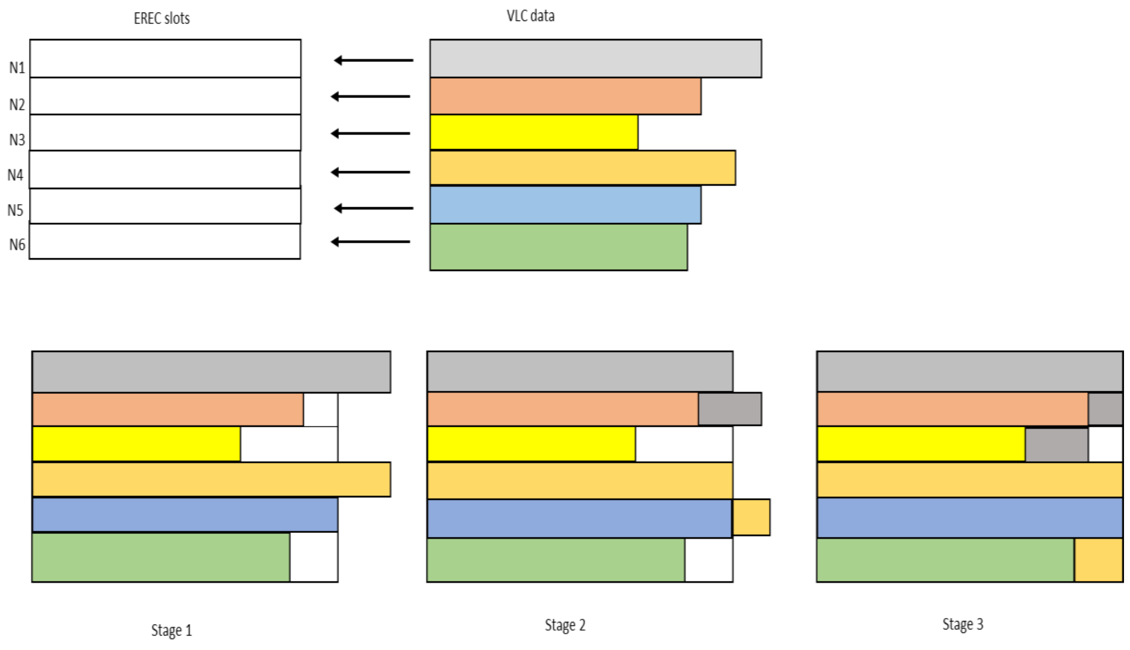


Figure 2.14 Error resilience using fixed-length entropy coding (EREC) adapted from [33]

2.5.6 Reversible Variable Length Codes (RVLC)

The RVLC technique was first used in 1997 by Wen and Villasenor to improve the error resilience of H.263+ video coding [55]. In this technique, the video decoder employs a fixed length codes capable to be decoded in two directions i.e. forward and backward directions. The fixed length code includes binary bits (0 and 1) in the bitstream [39].Thus, the two way (forward and backward directions) decodable code allows the decoder to more accurate error location compared to ERFC technique [56]. The RVLC performance comes at the cost of additional bit overhead ranging 2-3% which decreases the video coding efficiency [55]. In general, The bitrate overhead of the RVLC technique depends on the design of a set of variable length codes [57].

In this technique, once the decoder read the bitstream and detects an error, the decoding process starts in reverse direction using fixed length code with specific design [56]. Figure 2.15 is illustration example of error resilience mechanism in RVLC. Suppose that the decoder has received erroneous video segment.

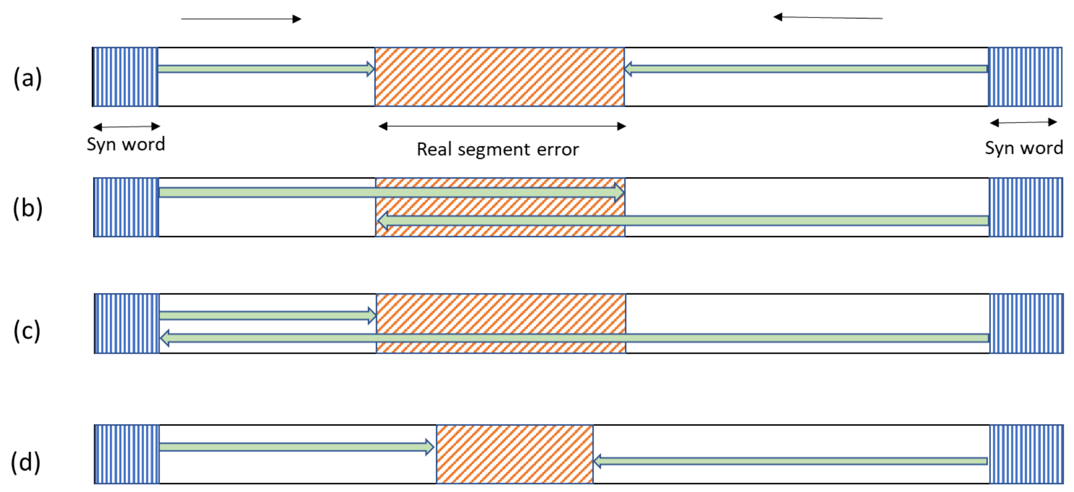


Figure 2.15 Reversible Variable Length Codes process
a) Starting and ending erroneous segment (b) Crossed error points (c) One direction error detection (d) Discarded segment after error isolation

Once the decoder detects the beginning of the erroneous video segment in forward decoding process, the decoder starts to decode the bitstream in backword direction as demonstrated in the figure depending on the next synchronisation word [57].

2.6 Decoder based Error Resilience approaches

Error concealment techniques are used to reduce perceived visual quality deterioration at a minimum level of the erroneous decoded video bitstream [58]. The error concealment process employ correctly received neighbouring information of the missing blocks to improve the subjective and objective visual quality of the damaged block [59]. As this type of error recovery use zero redundancy at video encode, error concealment is one of the most common approaches to reduce channel impairments with affecting video coding efficiency. Thus, the missing video information are recovered based on the correctly intra and inter decoded information at the decoder. There are two methods to hide the corrupted errors at the decoder, namely spatial (intra) concealment, and temporal (inter) concealment methods [60]. In spatial concealment, the corrupted decoded blocks are concealed by employing the surrounding blocks of the missing block [58]. In temporal concealment, the correct decoded information in previous frames are mainly involved in the recovery process of corrupted coding units [58]. In other words, this method involves replacing the erroneous area with its corresponding previous decoded frame [61].

In this section, the most popular video error concealment techniques on the video decoder side are reviewed.

2.6.1 Smoothing Filter Technique

Smoothing filter technique employs both temporal and spatial information of correctly received surrounding blocks to conceal the damaged blocks [78]. The smoothing technique includes estimating the missing frequency components of damaged blocks, in a way that the resulting recovered block is as smooth as possible [78]. This technique use a constrained energy minimisation approach [38]. The lost DCT coefficients are recovered by selecting minimum distance of spatially and temporally neighbouring blocks [38].

In 1993, Wang and his colleagues proposed the first smoothing filter method to recover a packet loss in image transmission over Asynchronous Transfer Mode (ATM) networks [79]. Their proposed approach was depending on using smoothness property to conceal the damaged block. This is obtained by minimising the variation of coded sample across the block boundary [38]. This technique is useful to conceal the DC and low-frequency components [38]. In the same year, Zhu and his colleagues made some improvements on the previous technique using adaptive interpolation by using temporal, spatial, and frequency domains [78]. The weighting values of the adaptive interpolation method depend on motion video contents and the area of the damaged block [78]. In 2006, W. Kung et al. presented a novel method for sharp or discontinuities edge concealment in luminance values [62]. Kwok and Sun made some enhancements on the smoothness filtering approach to include minimisation of edge variations by using adaptive edge measurements between the edges of the adjacent damaged block [63]. However, the success of this method depends on the correctness of the edge detection method.

2.6.2 Projecting Onto Convex Sets (POCS) technique

In the previous error concealment technique, the decoder uses energy minimisation method to minimise the erroneous affected visual areas at the decoder.
In Projecting onto Convex Sets (POCS) technique, the missing block information at the decoder is recovered by deriving Convex Sets of neighbouring blocks from smooth area (isotropic) or directional (areas containing edges) using spatial correlation from pixels in the surrounded missing blocks [64]. This is achieved using block transform coder. In 1995, the POCS technique was proposed by Sun H and Kwok, it designed to support block-based video coding standards [64]. They used only intra mode for interpolation by spatially correlating the damaged block with pixels of surrounded neighbouring blocks to recover the missing blocks [64]. As shown in Figure 2.16, a corrupted block is recovered by using eight bordering blocks including the damaged block. The derivation of the convex set is divided into directional edges areas and isotopically flat areas [64].

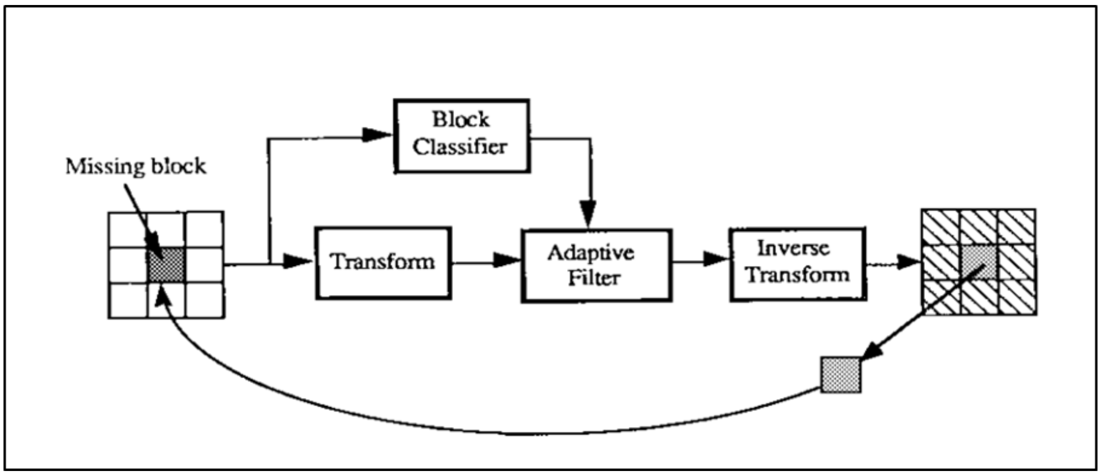


Figure 2.16 Illustration of adaptive POCS iterative restoration process [61]

At the first stage, the corrupted block is sent to the edge existence test by using the Sobel operator [61]. The block is categorised as either a monotone block (no visible edge orientations) or an edge block. The quantisation process starts at the edge orientation of the eight directions in an equally spaced form ranging from 0 to 180 degrees. Finally, the two projection operators are applied to the combined block [91]. The POCS technique is iterative. This means there are two projection operations applied to conceal the missing block in the Fourier transform domain [65]. The first one depends on the edge classification of the output value. The second one implements the truncation on the output value in the range between 0 and 255 [65]. These two operations are applied sequentially until there is no further change in the block values [65],[64].

2.6.3 Interpolation Technique

This technique interpolates the coefficients of spatially adjacent blocks depending on the smoothness feature in the reconstructed frame [59]. Compared to the previous concealment techniques, this technique is simpler in terms of computation processing cost and give better visual quality results at the decoder [59]. For example, a linear interpolation is used to predict the missing samples of the lost blocks. Hemami and Meng proposed an algorithm to reconstruct a missing block based on received transform coefficients at the decoder. Their proposed algorithm is based on exploiting the correctly received transform coefficients in adjusting blocks. A linear combination of the same coefficients in available adjacent blocks is used to determine the structure of the damaged blocks [59]. They considered worst case scenario when all the surrounded blocks are lost at the decoder side [59]. To overcome this issue, they used the difference values of minimum spatial interpolation to recover the received blocks in error concealment process [66] [59]. For getting more accurate estimation of lost blocks, Sign and Fazel enhanced the perceived video quality in the concealing process including smoothness property based on partial transform coefficients [66]. Their simulation results show a significant improvement in visual quality for recovering low frequency components [66]. However, the same results can be achieved when inserting (0) values when recovering frequency components.

2.7 Interactive Error Resilience approaches

For many years, starting from 1996, video delivery based on ARQ schemes could enhance video quality with higher coding efficiency gain compared with FEC schemes [67]. There are a number of the proposed error control algorithms which adopt ARQ schemes to improve the received video quality in ATM networks [68]. However, the increased required time in the ARQ schemes leads to sever distraction service in real-time video delivery applications. For instance, in video conference applications, the minimum recommended time delay should be no more than 400ms [69].

The Interactive Error Resilience combines both error resilience techniques at encoder and decoder sides to provide joint error recovery collaboration. This technique requires a reliable feedback channel to keep the encoder updated about the corrupted coding units. So that, the encoder can avoid involving the corrupted data in prediction process [70].

Feedback error control methods work jointly between the encoder on one side and transmission channel or decoder or both on the other side. A simplest interactive error resilience technique is enforcing the encoder to encode the coming coding unit with intra refresh at predefined refresh cycles. This type of approaches reduces temporal error propagation significantly at the cost of increasing processing complexity compared with encoding and decoding error control standalone approaches. To overcome high bit overhead resulting from error resilience encoding approaches. Furthermore, to get best the trade-off between coding efficiency and error resilience performance, once the network condition is improved, the encoder decreases the additional redundant information at encoding stage such as intra refresh cycles [32]. A great deal of previous research into interactive error resilience approaches has focused on adaptive intra coding based on update signal from the network and decoder conditions. The authors in [71] increase the bit rate saving through encoding only selective sensitive blocks in intra mode. This approach is further enhanced using an adaptive intra map generation for each encoded frame [71].

In these approaches, a decoder in interactive error resilience decides which part of the received bitstream is to be encoded as INTRA mode using a feedback channel, and which part requires to be concealed [32]. Thus, feedback-based error resilience techniques minimise the use of INTRA mode and therefore maintain higher coding efficiency on hostile channels [32]. In H.263 coding standard system, feedback channels used to increase error robustness by updating the encoder about corrupted MB’s location [72].

For multi-point video communications, Wada in 1989 proposed a method known as Wada’s selective recovery method to improve decoded visual quality [73]. This method reduces temporal error propagation by marking all corrupted macroblocks using a single-bit flag in order to avoid involving them in the interframe prediction process [73]. However, Wada’s method only considers temporal error propagation without taking into considerations the spatial error propagation at the frame level which negatively affects the perceived visual quality.

P. Haskell and D. Messerschmitt worked mainly on reducing temporal error propagation using motion vector resynchronisation in Asynchronous transfer mode (ATM) networks [74]. Their proposed algorithm based on inserting synchronisation codewords in motion compensated data with predefined intervals at the cost of slight increase in bitrate [74]. They proposed conditional resynchronisation mechanism called (Conditional Leaky Difference) [74]. This mechanism involves switching between temporal and spatial error concealment process [74].

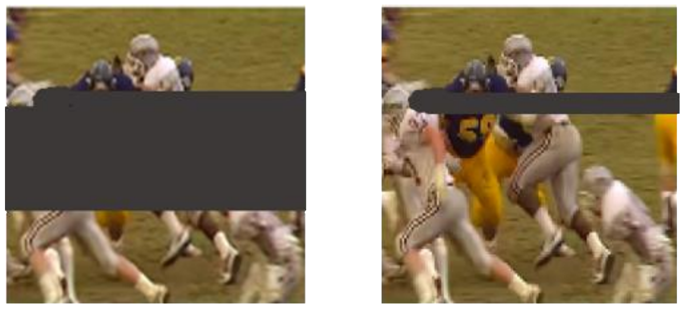
A considerable amount of literature has been published on interactive error control approach using feedback signal. The encoder side is updated about network conditions [75][32]. These studies include using feedback update spinal via reliable channel mechanism to update the encoder about the transmission channel conditions [29]. B. Girod and N. Farber have focused on interactive error control techniques [32][75]. They used acknowledgement information provided by a feedback channel to keep the encoder updated with network delay conditions [32]. One way to deal with these errors is to use a retransmission mechanism such as Automatic Repeat Request (ARQ) [7]. However, this solution does not support real-time video transmission applications in addition to subjecting bitstream with high error rates.

2.8 H.265|HEVC Error resilience techniques

With the development of video coding tools and video communication services, a demand to transmit a robust video stream in erroneous network conditions has increased [76]. Moreover, the growth of supporting high video resolution applications in mobile, teleconferences, and video on demand services motivates video coding experts and researchers to enhance the coding efficiency of previous coding standards to meet market expectations [77]. The H.265|HEVC coding standard experts are working hard on new H.265|HEVC coding standard versions to a variety of advanced multimedia technologies [76].

For instance, the first version of H.265|HEVC standard produced in April 2013, it supported only three coding profiles with limited adaptation to commercial video applications [76]. Every year a new H.265|HEVC coding standard version is produced with more coding standard extension coding profiles. In 2014, version 2 support coding profiles; Multiview and scalable coding applications. In 2015, H.265|HEVC coding standard version 3 was introduced to support 3D coding profiles [78]. However, the two standardisation bodies do not take into consideration the error resilience aspects in codec development stages. Moreover, the most challenging video transmission environments are due to wireless channel environments and heterogeneous networks [79]. Furthermore, high compressed bitstream suffers from transmission channels with time-varying characteristics. For instance, a heterogeneous network in mobile communications could cause a multipath fading channel. As the nature of compressed video is of variable length codes, the transmitted bitstream is very sensitive to multipath propagation delays, bitstream error and packets losses, especially when dealing with real-time video communication applications like video teleconferencing.

Including error tools at the encoder affect negatively on video coding standard efficiency and implementation cost. In order to reduce the redundant video coded data, the video coding standard needs to optimise the best trade-off between bandwidth channel requirements and employed video coding standard applications [13]. During the last two decades, many researchers have been working on improving H.265|HEVC video coding standard to be more robust against errors. G. Kulupana et al. proposed a motion estimation method based on concealment of past and future coded pictures in H.265|HEVC video transmission [80]. Figure 2.17 and Figure 2.18 show practical examples of spatial error resilience by inserting repetitive synchronisation words at slice level. Figure 2.17 (a) shows a spatial error propagation effect on visual quality within the frame, and Figure 2.17 (b) shows inserting a repetitive intra slice refresh using preselected slice segments groups.



(a) Spatial propagated error within the frame

(b) Error resilience by inserting repetitive synchronisation words

Figure 2.17 Spatial error resilience effects
(a) Spatial error propagation at slice level; (b) Inserting receptive sync words

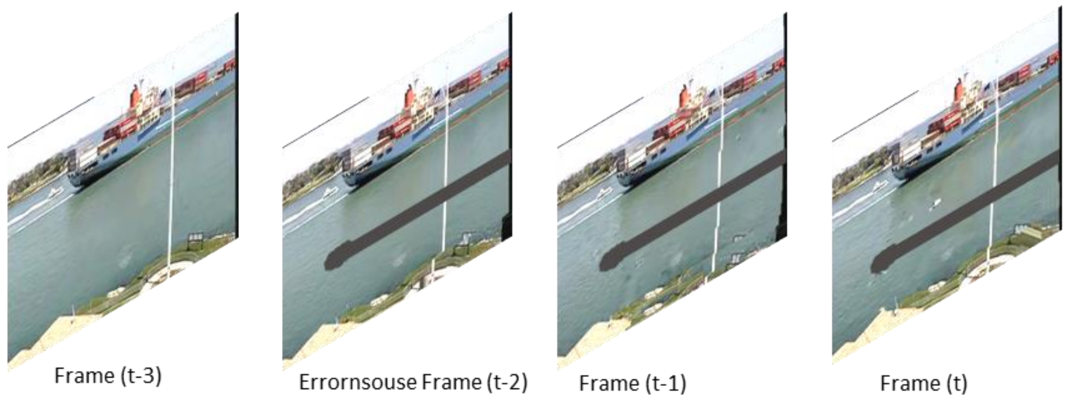


Figure 2.18 temporarily error propagation after decoded frame at time (t-2)

In general, the injected errors types on video compressed bitstream that directly affect the decoded visual quality at the end users can be summarised below [81]:

- Loss of bitstream header information:
  When a single-bit error hits header information, the whole encoded video sequence will be discarded or freeze at the decoder.
- Lost synchronisation word:
  When a synchronisation word is lost, the decoder cannot locate the actual errors and considers the following bits as undecodable, so will wait to receive next synchronisation word from the encoder. Because the video coded data is of variable length, the decoder is unable to locate the actual bit errors.
- Loss of video motion data:
  If the motion data are corrupted, the injected error will propagate temporally. Furthermore, the decoder will use incorrect motion vector and prediction block which lead to displacement some of the reconstructed areas.
- Loss of video data:
  When a sampled video data is corrupted with bit errors, the error effect will not only cause loss of synchronisation but also loss of spatial or temporal error propagation between decoded frames.

In a hybrid ARQ schemes, when the transmission channel is in good condition (error free), this means no retransmission is required which increases the transmission data rate effectively.

On the other side, a hybrid ARQ error control scheme in poor channel conditions such as wireless channel, the erroneous video data is retransmitted at the cost of reducing the effective data rates and increasing the processing time delay. It is worth noting, such error control schemes suffer from burst errors at the decoders because most cases result in unacceptable waiting leading to lose of synchronisation at decoder process. Therefore, the hybrid ARQ schemes does not guarantee receiving high visual quality for end to end low delay video applications.

Figure 2.19 is a simple block diagram showing end to end wireless video delivery system based on hybrid ARQ error control scheme. In the first stage, the video encoder compresses the input raw video. In the second stage, a compressed video is prepared for video delivery. In this stage, the compressed video data is partitioned and packetized into appropriate slice segments suitable for more error robustness in unreliable video transmission scenarios. The adaptive rate control estimates the available channel bandwidth through using reliable update channel at low delay constraints. If the receiver side detected an error, a retransmission process is triggered for the corrupted received packets [82]. Then, the encoder adapts the encoding setting accordingly.

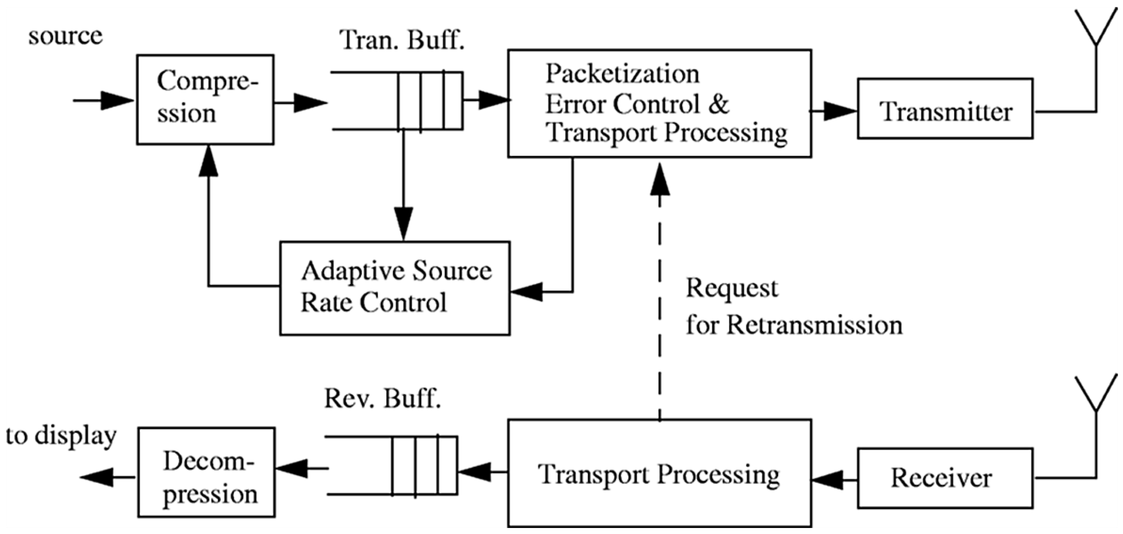

source
Compre-ssion
Tran. Buff.
Packetization Error Control & Transport Processing
Transmitter
Adaptive Source Rate Control
Request for Retransmission
Rev. Buff.
to display
Decom-pression
Transport Processing
Receiver

Figure 2.19 Wireless video transmission system with ARQ error control mechanism [82]

To overcome the limited bandwidth in commercially available channel, a scalable video coding (SVC) is the most common end to end video delivery solution used in heterogeneous networks. This technique helps to overcome the limitation of using multi-rate coding [83]. As shown in Figure 2.20, the SVC encodes a single bitstream into multi bitstreams with different video resolutions and frame rates [15].

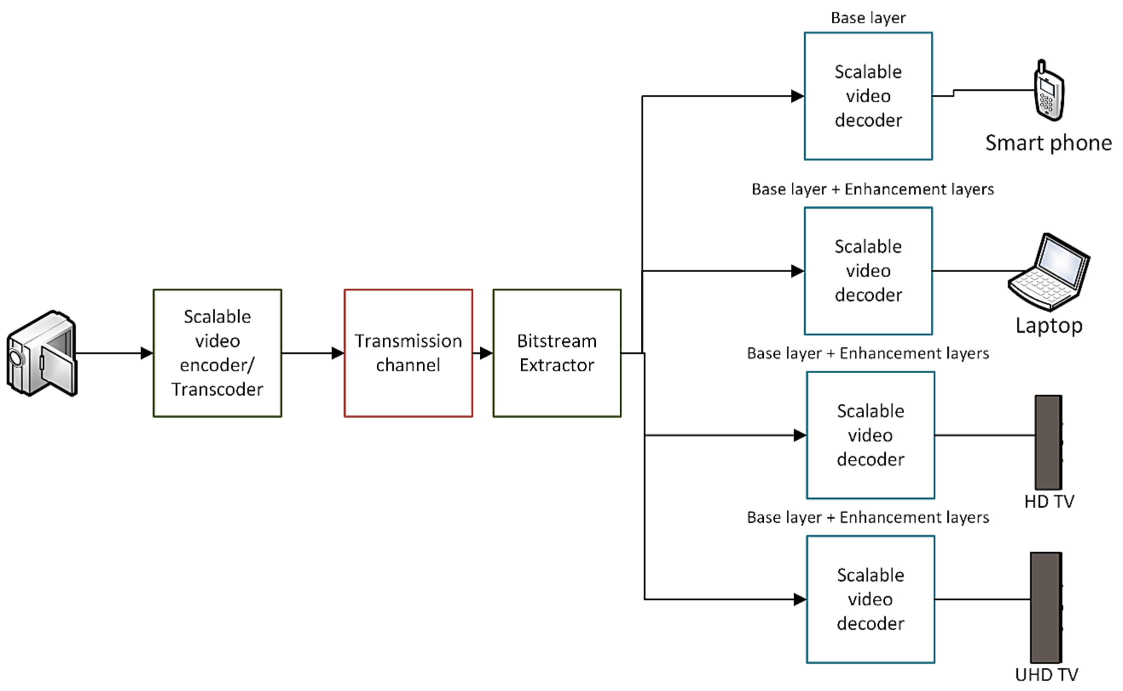


Figure 2.20 End to end video delivery system based on scalable video coding

At the decoding stage, according to the currently available network speed and bandwidth channel capabilities, the SVC decoder down-samples the bitstreams into different video resolutions applications and frame rate settings [83].

2.9 H.265|HEVC error control using feedback update

In general, low delay video application such as video conference applications require special attention to the design of error control using reference picture management method. In unreliable networks, the encoder with receiving feedback capabilities usually receives acknowledgement signal from the decoder with a delay channel (in milliseconds). Basically, in video communication system, there are two types of acknowledgement signals at slice level. These acknowledgement signals are transmitted from the decoder to the encoder [70]. The first type is called a positive acknowledgement (ACK) signal which is responsible for sending acknowledgement signal to the encoder indicating the correctly received slice. The reference frame is chosen depending on the ACK signal update received from the decoder. If the encoder did not receive an ACK signal at a predefined interval, the encoder assumes an error has occurred and an intra coding must be applied to resynchronise the decoder and terminate error propagations. The second feedback signal type is sending negative acknowledgement (NACK) signals by the decoder to notify the encoder that an error or loss has been occurred to the received bitstream. In addition to the acknowledgement of the correctly received slice, the addresses of the corrupted parts are signalled back to the encoder. At the same time, the reference picture buffer is updated accordingly every time receiving the acknowledgement signal. On the other hand, the decoder can apply an error concealment technique to reduce the temporal-spatial error propagation in case an error occurred.

2.10 Conclusions

As the nature of the compressed video is of variable length codes, a single bit error can lead to loss of synchronisation in the decoding process. This could lead to loss not only in spatial error propagation within frame level but also propagates to the following frames. Therefore, the demand of developing highly efficient video coding tools to support high quality video compression applications is expected to continue. Accordingly, transmitting highly redundant video data in unreliable channels with time limited bandwidth is a challenging task.

This chapter reviewed the state of art video error control approaches to enhance video quality in erroneous transmission channels. Overall, this literature review strengthens the idea of proposing efficient error control tools support the current H.265|HEVC video coding. Furthermore, the study has raised important question on how to achieve the best encoding balance between coding efficiency, encoding and decoding processing complexity at high error resilience performance.

At the beginning the process of the video transmission system is highlighted emphasising the channel errors effects on perceived visual quality at the decoder side. A brief review of the video coding standards is described. The review is focusing on the main aim for each coding tools developed for the current and previous video coding standards.

The error control approaches aiming to reduce the effects of spatial and temporal error propagation on perceived video quality are discussed highlighting the pros and cons of each error control approach. The literature survey further features the state of the art encoder-based error resilience techniques. Then the chapter features the decoder-based error concealment techniques aimed at mitigating the effect of channel errors at decoder side using zero redundancy by temporal and spatial correctly received information. The other error control approach is called interactive encoding error resilience approach where the encoder adapts the encoding process based on receiving feedback update signals from reliable transmission channel. The encoder becomes fully aware about the current network condition, as a result making the video encoder source more adaptive to time-changing network conditions and up to date with decoding process.

As H.265|HEVC compressed video has high redundant information, a single bit error could lead to corruption of larger decoded areas than previous video standards. Therefore, decoder-based error concealment techniques will not able to work efficiently alone without collaboration with video encoder side.

Therefore, it is obvious that developing a suite of video error resilience algorithms is necessary to handle and mitigate the transmission errors at the video decoder. These error resilience algorithms should be able to support different video applications scenarios.

Chapter Three

3 High Efficiency Video Coding (H.265|HEVC) Standard

The aim of this chapter is to highlight the key technical and theoretical concepts in addition to evaluating the H.265|HEVC coding performance in error-free and error-prone conditions. This chapter is organised as follows. In section 3.1, the chapter begins by a brief introduction to H.265|HEVC video coding standard, Section 3.2 discusses H.265|HEVC bitstream construction. Section 3.3 described the main coding tools to increase video coding efficiency. Intra and Inter prediction processes are described in details in sections 3.4 and 3.5, respectively. Section 3.6 discusses the reference picture set in H.265|HEVC codec highlighting the three referencing types. Deblocking and Sample Adaptive Offset (SAO) filters are described in details in Section 3.7. In Section 3.8, the H.265|HEVC video coding complexity is described in details. Section 3.9 presents the produced coding profiles in the three video standard versions. Section 3.10 presents video quality evaluation methodologies used in the experimental work, and Section 3.11 presents the used framework setup in the experimental work. Section 3.12 provides comparison study on bit rate savings of H.265|HEVC compared with H.264|AVC coding standards. Section 3.13 presents video coding comparison study between H.265|HEVC and H.264|AVC standards in error-free and error-prone environments. Section 3.14 focuses on encoding error sensitivity analysis in terms of VCL and non-VCL data in addition to motion vector fields sensitivity against random errors. Section 3.15 provides technical comparison study in video coding tools between H.265|HEVC and H.264|AVC coding standards. Section 3.16 summarises and concludes this chapter.

3.1 Introduction

The advancement in the manufacturing of high-performance electronic devices and their display technologies, such as mobile phones, tablets, and smart televisions devices resulted in increased demands of low delay with ultra-high-resolution video content delivery. Furthermore, most of the commercially available displays support spatial resolution up to 4K (7668×4320) resolution [84]. Such high-resolution display capabilities can consume most of the available bandwidth in conventional networks. Hence, highly efficient video coding tools to support high-resolution video delivery is necessary to meet the user requirements. At the present, the most recent video coding standard is High-efficiency video coding standard (H.265|HEVC) standard [7].

The H.265|HEVC video standard is a result of continuous hard work from video coding experts and researchers to enhance the coding efficiency of the previous video Coding standard standards, as well as, to support two main applications: increase support for parallel processing applications and meet end-users demand for ultra-high-definition video delivery [7]. Hence, the main target of developing H.265|HEVC standard was to double the coding efficiency of H.264|AVC video coding standard. This means keeping the same video quality at half encoding bit rate [7]. The H.265|HEVC standard produced by Joint Collaborative Team on Video Coding standard (JCT-VC) from International Electrotechnical Commission (IEC) and International Organization for Standardization (ISO) organisations [9]. There are three coding standard versions generated from both main organisations. The first version published in January 2013, and the specification was formally standardised in April 2013 [28]. The second version published in October 2014 [85]. The third and most recent H.265|HEVC video standard was version 3 which was published in April 2015 [4].

3.2 Bitstream construction

The bitstream generation from H.265|HEVC coding standard system is based on using the same hybrid video coding concept used in the previous video coding standards produced by ITU and ISO bodies. To explain, a hybrid video coding system employs more than one encoding components to compress raw video samples. A raw video sequence is compressed by removing extra spatial and temporal video information at the first stage aiming to keep the best balance between encoding bitrate and the perceived visual quality at the decoder. This is achieved by using efficient encoding tools using intra and inter predictions process for removing spatial and temporal redundant video samples. Figure 3.1 shows the main video encoding stages in H.265|HEVC coding standard starting from picture partitioning stage of the raw video sequence ending to the generation of H.265|HEVC bitstream [86].

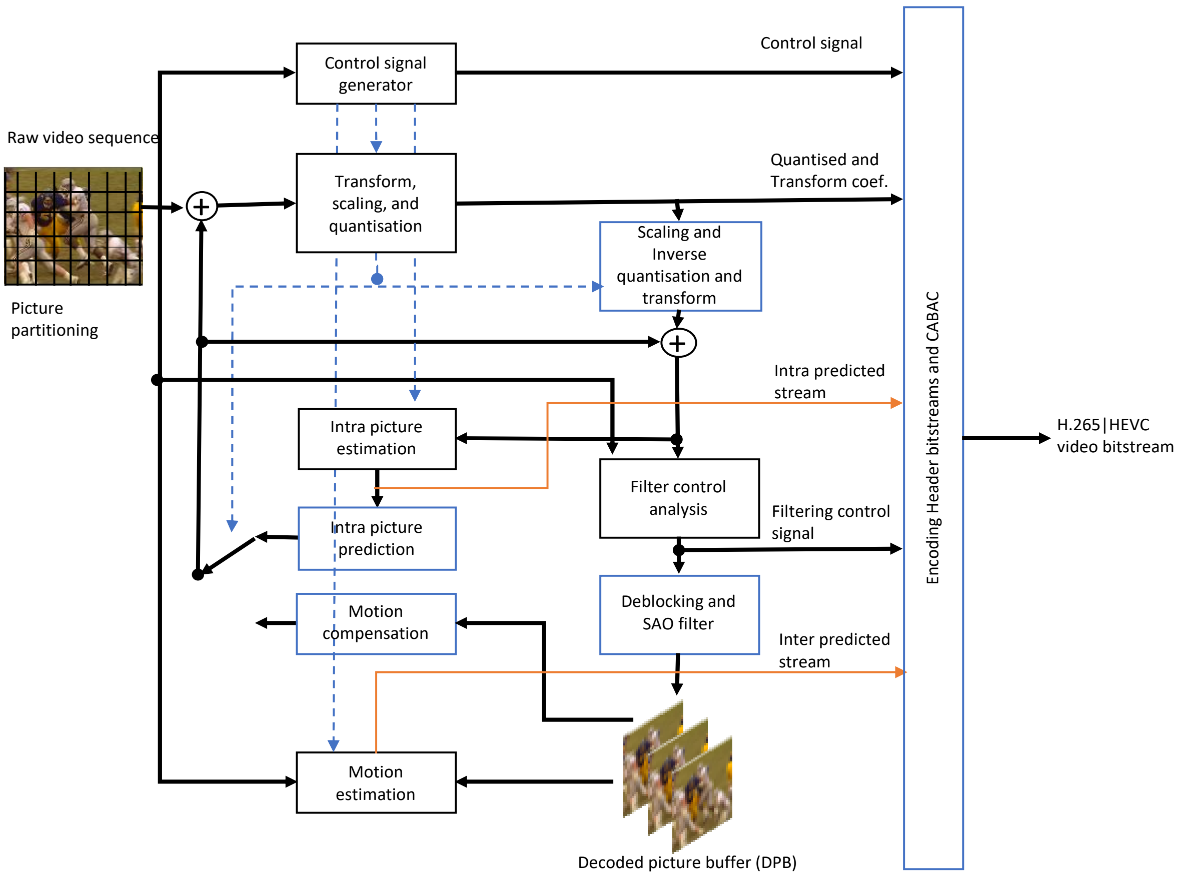


Figure 3.1 H.265|HEVC coding standard block diagram, adapted from [9]

The input picture at the encoder is firstly partitioned into small blocks sizes. A residual signal (error signal) is generated as a difference between the input signal and the prediction signal (intra or inter signal) [86]. Then, the generated residue signal is transformed using Discrete Cosine Transformation (DCT) and Discrete Sine Transformation (DST). After that, the output transformed signal is quantised and scaled. At this stage, the obtained generated signal from the transformation process has few frequency components mostly in low frequency components. Correspondingly, after scaling and quantisation stage, the residue signal can be encoded with fewer representation bits.

A resulted output signal then filtered using In-loop filters. There are two In-loop filters called Deblocking Sample Adaptive Offset (SAO) filters. The deblocking filter is used to reduce and smooth the generated artefacts from block based transform and quantisation process [86]. After deblocking filter, the output signal is adaptively filtered at frame level using SAO filter. The adaptive SAO filter process depends on generated offset values from lookup tables to be sent alongside with control signal to the CABAC entropy coder. The Context-adaptive binary arithmetic coding (CABAC) is binary entropy method used in H.265|HEVC video coding standard. The filtered signal then stored in decoded picture buffer to be used later in inter prediction process (Figure 3.1). Additionally, the CABAC entropy coder encode the output residual signal together with prediction signal (intra or inter predicted signal) and encoding control signal [86]. The control signals include necessary encoding information about the generated video bitstream such as inter and intra prediction modes, pictures order numbers, and other important encoding information [86]. In other words, at the decoder side, a reverse encoding process will be applied at the video bitstream.

3.3 Video Coding Tools

The performance of HEVC video standard has been improved more than the previous standard version (H.264), this is done by enhancing some existing tools such as CABAC coding tools. Besides, some of them have been added to the standard for two main purposes; enhancing the compression efficiency for increasing needs to use 4K video resolution in everyday consumer electronics and its transport integration, and reducing complexity by using parallel processing tools [6]. In addition to that transmission, a high-resolution video transmission shows an urgent challenge due to limited bandwidth channels capacity. All the mentioned reasons have motivated the researchers who work in the field of video coding systems to develop a suite of tools capable of increasing coding efficiency keeping the same required bit rate of the previous standard, i.e. H264|AVC. The leading organisations ITU and ISO worked on generating a reference software called HM [87] with specifications coding document [28]. The reference software tool is used for testing conformance of new algorithms and research purposes and internal committee development work. The performance of HEVC video standard has been improved more than the previous standard version (H.264). This is done by enhancing some existing tools such as CABAC coding tools. In the next subsections, the main efficient video coding tools are illustrated.

3.3.1 Network Abstraction Layer (NAL)

The main aim of using NAL units in both H.265|HEVC and H.264|AVC video coding standards is to make the encoded video more network friendly and more adaptable to

various transport systems [87]. In other words, introduction of NAL units concept helps to map video samples representations into different transport layers including MPEG transport stream standard. In addition to supporting Real-time Transport Protocol (RTP) for audio and video contents transport, it also provides adaptation for bitrate streaming technique such as Dynamic Adaptive Streaming over HTTP (DASH). Figure 3.2 shows simple diagram demonstrating NAL units transmission process between the encoder and the decoder. The bitstream mapped in NAL units [88]. As shown in the figure, after encoding the input raw frames with H.265|HEVC encoder, the encoded bitstream is transmitted in NAL units. The transmitted encoded video is sent or stored in form of a bit sequence of encoded NAL units (with an integer number of bytes).
At the receiver end, the H.265|HEVC decoder extract the fixed size header NAL units (two bytes in H.265|HEVC coding standard) to classify NAL units contents.

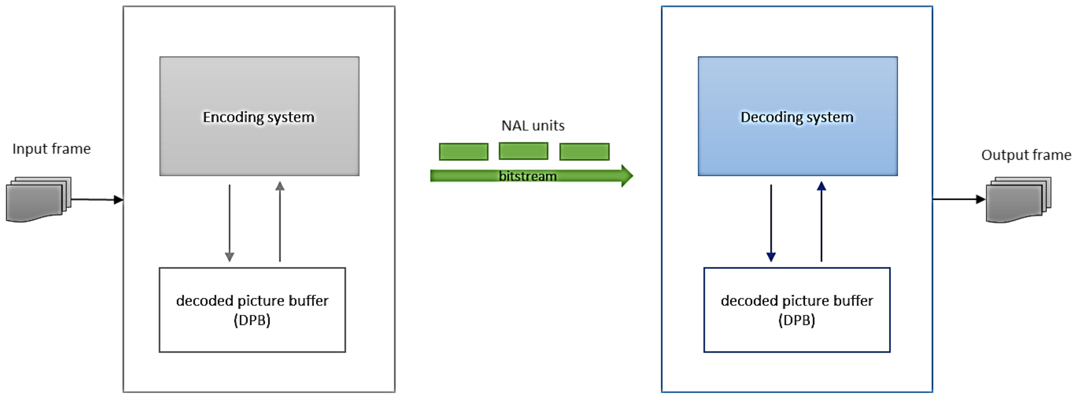


Figure 3.2 NAL units representation in H.265|HEVC coding standard system adapted from [88]

In general, each NAL unit contains a NAL unit header (two bytes length), and the remaining parts of NAL units include payload video data. The NAL unit header identifies the purpose of the upcoming payload data [45]. The NAL units are divided into two main classifications; 1) Video coding layer (VCL-NAL) units, and 2) Non-Video Coding Layer (Non-VCL-NAL) units. The VCL-NAL units includes compressed video samples representation in form of group of video slice segments [87]. The two bytes NAL unit header includes description for the following NAL unit’s data contents (for both Non-VCL-NAL and VCL-NAL). Correspondingly, it extract the bitstream structure properties of the payload video information [87]. The Non-VCL-NAL type is the most sensitive part as it includes the encoding parameters sets; video parameter set

(VPS), sequence parameter sequence (SPS) and a picture parameter set (PPS) in addition to the supplemental enhancement information (SEI) messages [87]. The VCL-NAL unit contains video slice syntax elements which are necessary for decoding allocated region in the decoded frame [28]. Each group of VCL-NAL units construct one Access Unit (AU) [8]. Figure 3.3 shows NAL unit header in H.265|H.265|HEVC video coding standard.

F	NAL type						layer ID						T ID		
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
Byte 1								Byte 2							

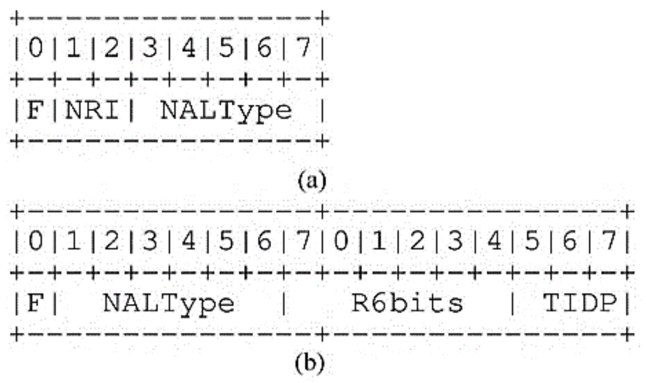
Figure 3.3 Network Abstraction Layer (NAL) header in H.265|HEVC standard adapted from [87]

The starting bit of each NAL header in the H.265|HEVC standard is set to “0” bit and it is called forbidden bit for compatibility purposes with different transport protocols. The second part of the NAL unit header is denoted as NAL type, which includes (1 to 6) bits of the Byte 1 header. The NAL type field helps to inform the decoder that the following payload data is needed to be extracted or discarded.

The third header part (layer ID) contains six bits which is reserved for the next six bits (starting from bit 7 in byte 1 ending to bit 4 of byte 2). In the first version of H.265|HEVC, the six bits are set to (0) values as it was reserved for future coding standard extensions [78]. The H.265|HEVC Version 2 supports 21 range extensions, one Multiview and two Scalable profiles [78]. While in the third standard version, the 6bits field is adapted to describe the depth level and temporal view layers in the 3D extension profile [78].

The fourth header part referred to as (T ID) contains 3 bits (Temporal identification number) which is responsible for temporal layer scalability ranging from layer (0) up to layer (6). If the current NAL unit is at higher layer and discarded, then it identifies the lower temporal sublayer of the encoded bitstream [24][28].

Comparing with previous H.264|AVC video coding standard, the NAL unit header length is extended into two bytes length. The extra added bits are to support the more future video standard developments with more extended profiles. These extended profiles include conformance of H.265|HEVC bitstream to Multiview and 3D video applications [87]. Figure 3.4 shows NAL unit header structure comparisons for H.264|AVC and H.265|HEVC video coding standards.



```
+---------------+
|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+
|F|NRI| NALType |
+---------------+
```

(a)

```
+---------------+---------------+
|0|1|2|3|4|5|6|7|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F|  NALType  |   R6bits  | TIDP|
+---------------+---------------+
```

(b)

Figure 3.4 Comparison of NAL unit header structures
a) H.264|H.264|AVC coding standard system b) H.265|HEVC coding standard system [45]

Similar to H.264|AVC coding standard , the starting bit in NAL header should be (0) to avoid conflicting with the start code of MPEG-2 systems [45]. The two bits nal−ref−IDC (NRI) are removed in H.265|HEVC coding standard because they are already included in NAL types making it unnecessary to reserve two additional bits in the NAL header [45]. The number of bits in H.265|HEVC NAL type is increased by one bit to meet the requirements of the designed NAL types which include 64 types [28]. Some of NAL type IDs are still reserved for future usage [24].

3.3.2 Picture partitioning

The main aims of picture partitioning in H.265|HEVC can be summarised below. Firstly, to increase error resilience robustness by allow encoded slices to be independently decoded. For instance, in case of error injected on one encoded slice, the following independently encoded slices will not be affected at the decoder. Secondly, to increase support to the encoded bitstream to adapt different maximum transmission unit (MTU) sizes. The third advantage is using optional partitioning scheme to support parallel processing applications [89]. More details about the picture partitioning types and process will be discussed later in H.265|HEVC complexity (section 3.8).

3.3.3 Block partitioning

The main block partitioning difference between H.265|HEVC coding standard and the previous standards is in using basic processing unit structure. The basic processing unit in H.265|HEVC is flexible in size. The H.265|HEVC coding standard use quadtree structure, so each frame is divided into Largest Coding Units (LCUs), and then the coding units are further partitioned into smaller units called Coding Tree Units (CTUs). Therefore, this high flexibility with blocks partitioning size contributes to increase the H.265|HEVC coding efficiency [16]. The quadtree structure is ranging from (64×64) up to (8×8) block size. The decision of the block size is adaptively selected. The H.265|HEVC coding standard can achieve high bit rate saving with high-quality video resolutions due to using larger block sizes compared with only (16×16) Macroblock with fixed size in previous coding standards [90].

There are four different block names in H.265|HEVC partitioning process: coding tree unit, coding unit, a prediction unit, and transform unit [91]. Each unit has its related coding blocks named; coding tree block (CTB), coding block (CB), prediction block (PB), and transform block (TB) [91]. An example of frame partitioning can be demonstrated in Figure 3.5.

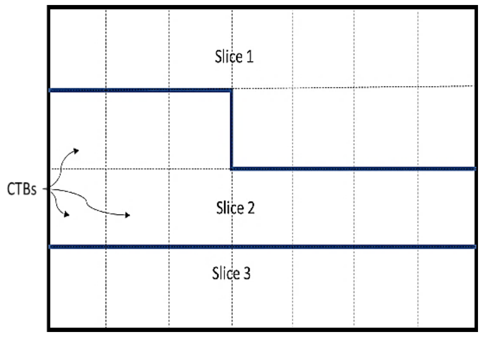


Figure 3.5 Example of frame partitioning into multiple slices and CTBs

As in H.264|AVC coding standard system, each frame is divided into one or more slices. On the other side, each slice in H.265|HEVC contains a group of CTUs [92]. As a result, the flexibility of CTUs numbers inside each slice insures the encoded NAL units meets the channel MTU size requirement.

In the following, the process steps of H.265|HEVC block partitioning starting at frame level are explained [93];

1 Partition of each frame into square shape blocks with varied sizes Coding Tree Blocks (CTBs).
2 A Group of CTBs combined to be in one slice or more than one for each frame.
3 Each slice consists of one or multi-slice segments.
4 Each slice segment consists of; Coding Blocks (CB), Transform Blocks (TB), Prediction Blocks, and syntax elements.
5 The organisation of each slice blocks can be represented as decodable blocks samples at the decoder.

Each H.265|HEVC frame includes a single or multi encoded slice [16]. The slice consists of a group of CTUs encoded in raster scan order [16]. The flexible coding unit size can be divided recursively from large to small sizes (see Figure 3.6); the largest coding units (LCU) with a size of 64×64 with coding unit (CU) depth of (CU=level 0), CTB size (32×32) and CU depth (level 1), (16×16) with CU depth (level 2), and the smallest CUB size is (8×8) with CU depth (level 3) [16]. Figure 3.6, illustrates an example of frame partitioning into multi CTUs in a recursive way with the largest CTU of 64×64, and the smallest size of 8×8.

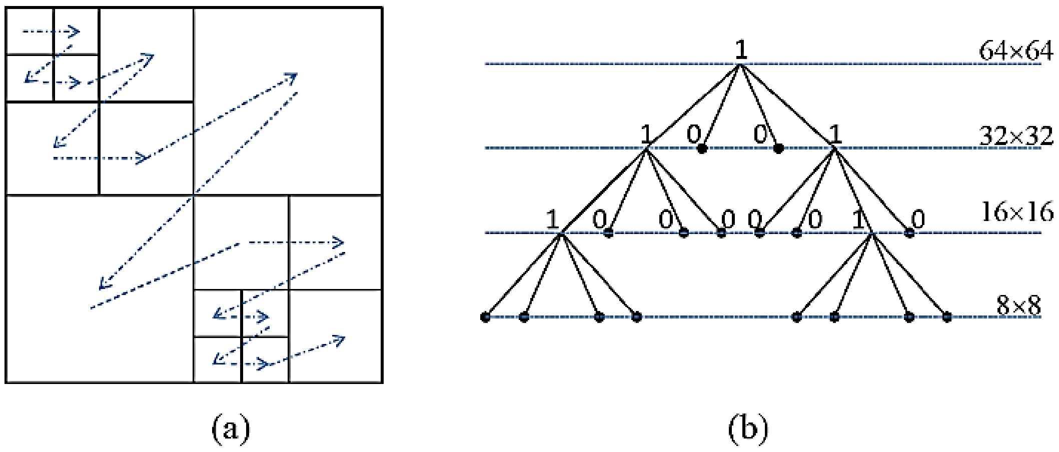


Figure 3.6 CTUs partitioning in H.265|HEVC [16];
(a) CTU raster scan processing, (b) Coding tree structure

Figure 3.6 (a) demonstrates CTBs partitioning in raster scan with different sizes. Figure 3.6 (b) shows the corresponding levels for the demonstrated partitioned frame [16]. The frame partitioning levels are represented by binary numbers. Binary (0) represent a leaf node which refers to the smallest coding unit. Whereas a binary (1) represent a non-leaf node refers to quadrant nodes. The decision of CTB partitioning depends on the coding standard rate-distortion optimisation [46]. In other words, when sharp texture edges are detected at frame level, the splitting process of CTUs will continue to partition larger block into smaller blocks as depicted in Figure 3.7. The recursive partitioning CTUs will stop until reaching a high correlation between adjacent coding blocks.

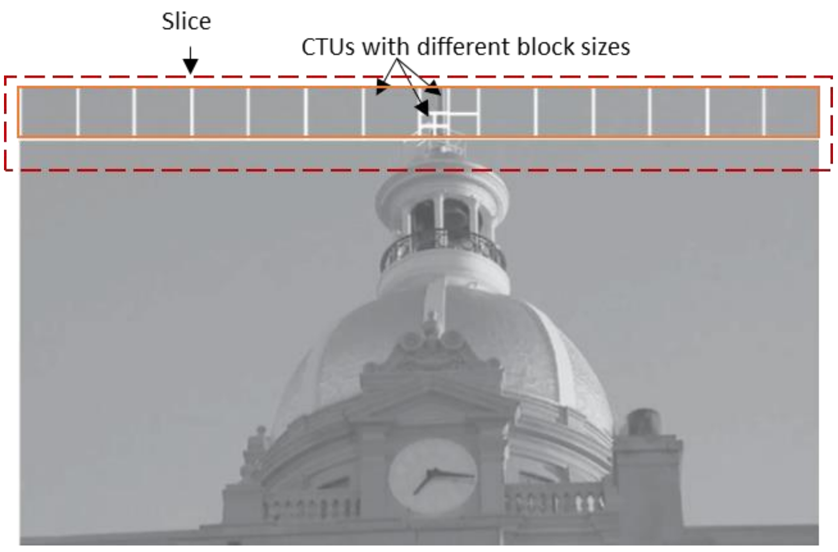


Figure 3.7 Coding Tree Block partitioning for real picture adapted from [46]

The CTU consist of one luma CTB and two chroma CTBs with their corresponding syntax elements as demonstrated in (Figure 3.8). The sizes of CTBs are: 16×16, 32×32, or 64×64. The larger the size of luma CTBs, the better the encoding bit saving [9].

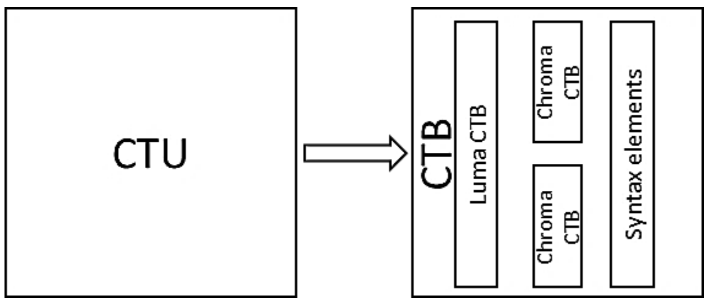


Figure 3.8 Coding Tree Unit representation in H.265|HEVC standard

The lower levels of CTU contain; Transform Blocks (TBs) and Prediction Blocks (PBs) [46]. The TBs contain the generated transformed and quantised residue signals. The PBs are predicted signals generated from spatial or temporal predictions process. The prediction mode is chosen at coding unit (CU) level. Furthermore, when grouping luma PBs alongside with chroma PBs with their prediction syntax elements, they construct one Prediction Unit (PU) [46]. The inter (temporal) and intra (spatial) prediction modes in H.265|HEVC video standard will be cover later in this chapter.

3.4 Intraprediction

The main aim of intraprediction process is to remove the spatial correlation in each frame. In H.265|HEVC coding standard, this is achieved by removing the spatial redundancy between CTBs at frame level [94]. To increase the coding efficiency, the H.265|HEVC video standard includes more intra modes compared with previous coding standards. The intra prediction units of chroma components support 5 block sizes ranging ($4 \times 4 - 64 \times 64$). The smaller block size, the higher support with texture details representations compared to larger the block sizes. Furthermore, each intra prediction unit supports 35 prediction modes. The prediction mode include: planner (mode 0), DC (mode 1), and directional modes (modes 2-34) [95]. The intra prediction process of the current prediction unit starts from pixel prediction depending on the closest pixels of the neighbour Top and left directions with the same texture details of the closest neighbour blocks [94]. The two non-directional prediction modes i.e. DC (flat mode) and planner (or surface fitting) predictions modes are employed to increase the coding efficiency when encoding smooth areas. To increase visual quality of H.265|HEVC video coding standard, the angles or directional modes are increased to 33 modes compared with 8 modes in H.264|AVC video coding standard. This increase in directional modes allows the video encoder to be adaptable to represent more texture details. Figure 3.9 shows an example of directional prediction. The figure shows samples predictions in four different directional modes. The reference samples are located from the last row of the top neighbour block and the last column of the left of the current block which are involved directly in the prediction process [95].

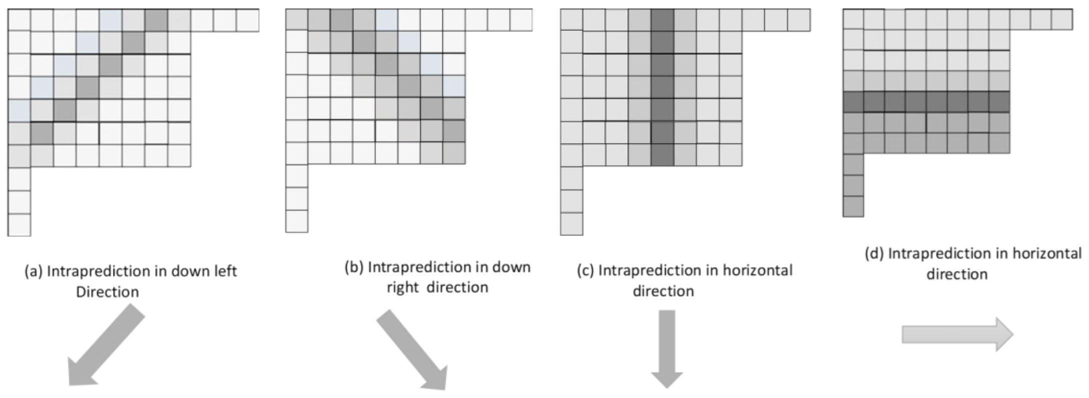

(a) Intraprediction in down left Direction
(b) Intraprediction in down right direction
(c) Intraprediction in horizontal direction
(d) Intraprediction in horizontal direction

Figure 3.9 Directional intra prediction process

In general, more prediction modes increase the video coding efficiency at the cost computational complexity. In H.265|HEVC coding standard, the angular prediction modes can be depicted in Figure 3.10 [96]. At prediction unit level, the video encoder selects the best suitable prediction mode depending on previously decoded video samples. The prediction process employs reference samples from top, top left, and left of its neighbour blocks.

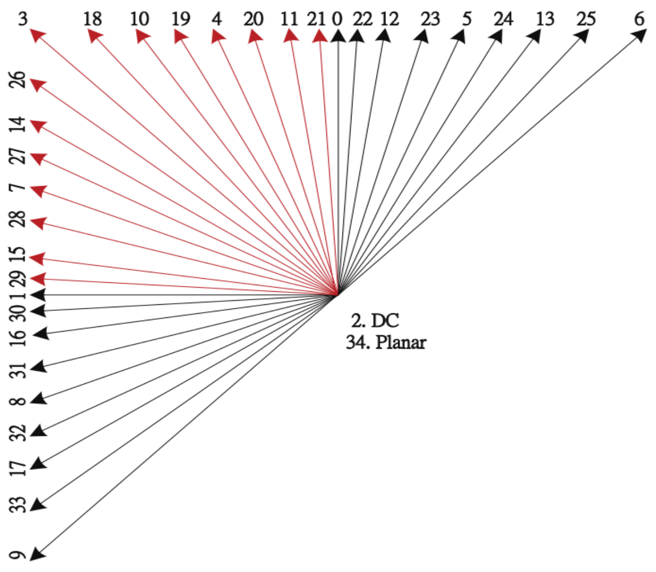


Figure 3.10 Intra prediction modes in H.265|HEVC standard [96]

It is worth noting, the prediction process works independently from the transform block (TB) [97]. At the slice boundary level, there are no reference samples to be employed in the intra prediction process [97]. To reduce the spatial error propagation effects in H.265|HEVC video coding standard, in the decoding process, the missing samples are replaced with the closest neighbour reference blocks available [97]. In contrast to H.264|AVC, intra_DC is enforced to be used in decoding process [98].

3.5 Interprediction

In general, the inter prediction in hybrid video coding standards contributes to improve coding efficiency significantly due to removing the temporal redundancy between video frames. In inter prediction, a block-based motion compensation is used to estimate the current frame from one or more previously encoded frames [99]. In H.265|HEVC coding standard, a block matching algorithm is commonly employed in motion estimation process [44]. For motion vector coding, a new coding tool called Advanced Motion Vector Prediction (AMVP) is employed to support merge mode and differential coding in H.265|HEVC coding standard [87]. The inter prediction of H.265|HEVC coding standard system employed at prediction blocks (PB) level while decision of prediction mode is selected at higher level i.e. coding unit (CU) level [100] , [101].

The basic idea of inter prediction process is that a portion of moving frames is selected to be suitable for interprediction process. The difference between the moving objects is small, and this difference can be inter predicted using motion vectors and error signal. The motion vector can be represented in X and Y coordination values and it is referred to $MV(x, y)$ as shown in Figure 3.11.

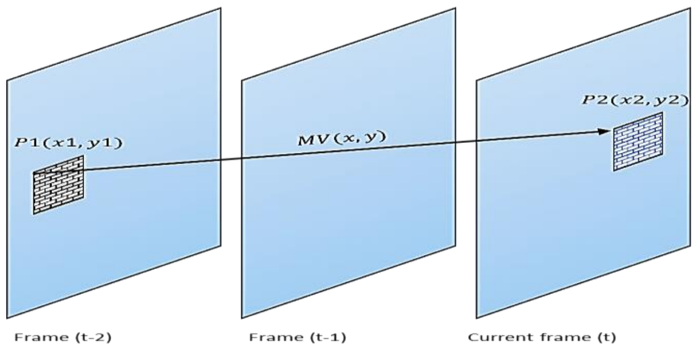


Figure 3.11 Motion compensation Prediction (MCP) concept

The figure demonstrates moving block referred to as point P1 and located at $(x1, y1)$ with time interval (t-2) from currently encoded frame. A predictor blocks with motion vector $MV(x, y)$ is inter-coded with displacement of Δx and Δy values to the current block. The related motion predictors values of referenced frames at $\Delta T = 2$ are stored decoded picture buffer (DPB) [102]. The motion vector accuracy of the moving object is defined by sub samples fraction values.

For inter prediction mode, the motion vectors are generated from two prediction types: Bi- prediction and uni- prediction types. In bi-prediction, a combination of two sets motion vectors are employed to generate final motion compensation prediction.

One set presents prediction in forward direction (previously encoded frames) and other set presents reverse direction (future frames) and can be denoted as $MV1(x,y)$ and $MV2(x,y)$. In both prediction types, the motion vectors can be generated using more than one reference picture stored in two reference picture lists i.e. (list 0) and (list 1). In case of using more than one reference picture, there are two possible scenarios; the first one is using motion vector averaging methods, and the second one is using a weighted motion vector prediction method [103]. Concerning H.265|HEVC coding standard, the minimum size of bi-prediction blocks is limited to 8 ×4 and 4×8 to meet the memory limit requirement. Example of using bi-prediction coding using I and B frame types only can be demonstrated in Figure 3.12 [103]. The prediction process depends on previous and future encoded frames. The selected referenced pictures are stored in (list 0) and (list 1).

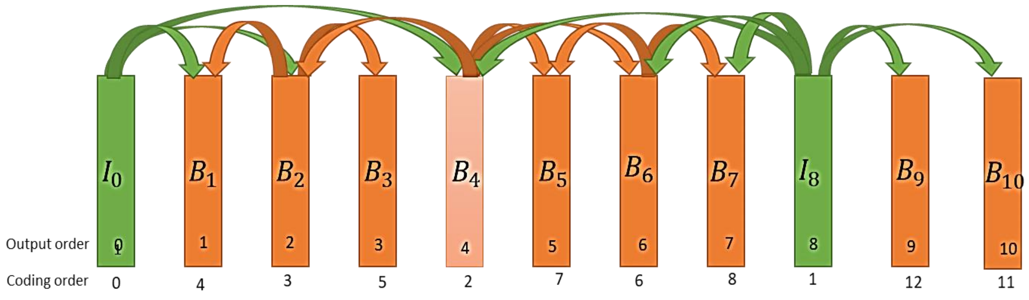

I_0
B_1
B_2
B_3
B_4
B_5
B_6
B_7
I_8
B_9
B_{10}
Output order 0 1 2 3 4 5 6 7 8 9 10
Coding order 0 4 3 5 2 7 6 8 1 12 11

Figure 3.12 Bi-prediction coding scheme

The Uni-prediction encode motion data $MV(x,y)$ is based on calculation of previously encoded frames. This is achieved using one or more reference pictures encoded with indexed ID and stored in (list 0), as shown in Figure 3.13.

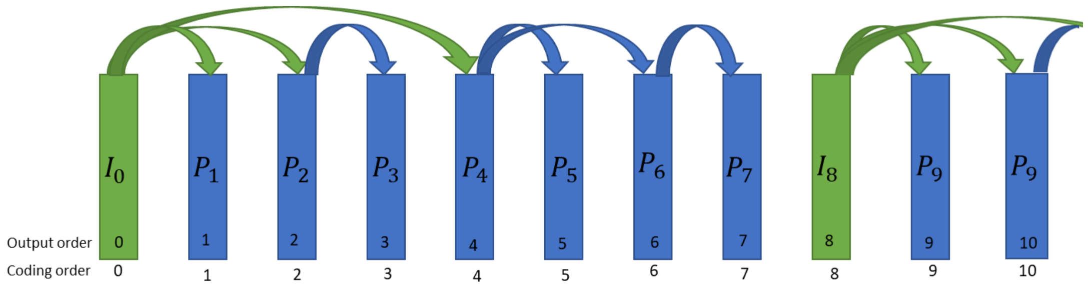


Figure 3.13 Uni-prediction coding scheme using I and P frame types

Figure 3.14 shows an example of using four referenced frames in Bi prediction type with different picture order count numbers (POC_N).

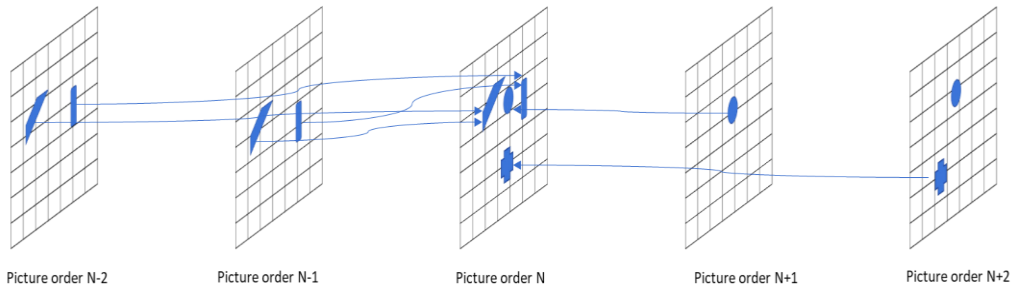


Figure 3.14 Bi predictions using four referenced frames

In bi prediction, the motion vectors are predicted using previous and future (I or P frames). In uni-directional prediction, the motion vectors are predicted depending on previously (P or I frames) [102]. The decision of taking inter or merge modes is taken at the coding unit (CU) level. All PUs should be in the same mode at CU level [104].

For deriving motion vector predictors accuracy, same as in H.264|AVC video coding standard, the H.265|HEVC coding standard support accuracy with $(1/4th)$ distance between Luma samples [105]. For chroma samples, the prediction accuracy between chroma samples depends on the encoding format used in chroma samples. In case of 4:2:0 coding format, the distance between the samples will be at $(1/8th)$ of samples [105]. Figure 3.15 shows an example of inter prediction process at fractional position (non-integer) for pixel value located at (x=0.25, y=0.75). Firstly, 8-tap kernel filter called Finite Impulse Response (FIR) filter interpolates the reference luma samples in the horizontal direction. Then, 4- tap FIR filter kernel is employed in vertical direction to interpolate chroma samples [102]. These two filters contribute to improve filtering process of high frequency components.

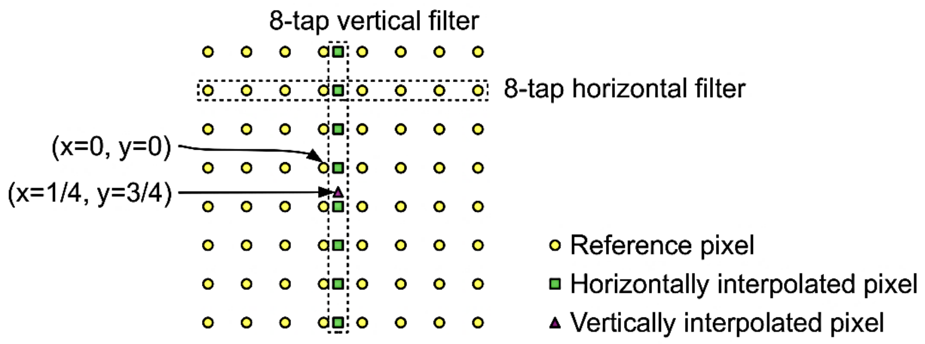


Figure 3.15 H.265|HEVC Interpolation process at pixel level [102]

3.5.1 Advanced Motion Vector Prediction (AMVP)

In general, H.265|HEVC coding standard uses the same motion vector prediction concept used in previous hybrid video coding standards, the motion vector can be represented by a difference of (x) and (y) displacement (shift) values of the current block from selected reference. The difference of displacement values called motion vector predictor (MVP). The process of generating motion vectors is usually correlated with other motion vectors of the current or previous encoded frames [99].
The calculation of Motion Vector Difference in horizontal and vertical directions can be calculated as in equations (3.1) and (3.2):

$$\mathrm{MVD(x)} = \triangle\, \mathrm{x} - \mathrm{PB(x)} \qquad \text{Eq.(3.1)}$$

$$\mathrm{MVD(y)} = \triangle\, \mathrm{y} - \mathrm{PB(y)} \qquad \text{Eq.(3.2)}$$

Where $\triangle$ x and $\triangle$ y are the displacement values in (x) and (y) directions, respectively, between the current block pixel and referenced block pixel. The PB(x) and PB(y) are x and y values of prediction block located in reference pictures list and addressed at $PB(x)$ and $PB(y)$, respectively. In H.265|HEVC coding standard , the motion vector is generated from motion vector predictor and motion vector difference [87]. The selection of the candidate predictor process includes all available four selection candidates (Figure 3.16) based on advanced motion vector prediction (AMVP) [87]. The AMCP is introduced to support the new flexible partitioning feature in H.265|HEVC coding standard. The motion information is derived from the selected candidate list based on merging operation mode [87]. The merging mode employs the same shared prediction motion vectors across large frame areas with more accurate motion vector selection [87].

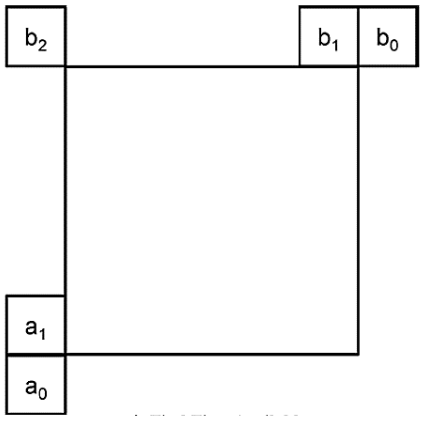


Figure 3.16 Motion vector derivation in H.265|HEVC standard [87]

In H.264|AVC coding standard , the motion vector derivation process in direct mode depends on finding the average of selected in three surrounding blocks referred to A, B, and C blocks as demonstrated in Figure 3.17 [6].

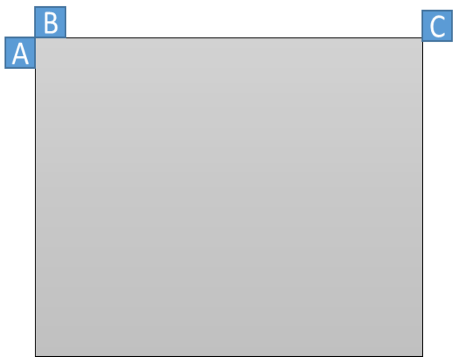


Figure 3.17 Motion vector derivation in H.264|AVC standard

3.6 Reference Picture Sets (RPS)

A reference picture management in H.265|HEVC video coding standard use a new concept called Reference Picture Sets (RPS) [106]. The fundamental idea of using RPS is to manage multiple reference pictures at decoded picture buffer (DPB) in more efficient way [45][6] [106]. A motion prediction and motion vectors are mainly involved in RPS process [107]. The main aim of using RPS concept is to improve reference picture management by enhancing motion data prediction accuracy [6].

In reference marking process, the RPS use three classes of decoded pictures, which are stored at the decoded picture buffer (DPB) [45]; short-term, long-term reference picture classes, and unused for reference picture. Both short-term and long-term decoded pictures are used in interprediction process. While the unused reference picture will not be included in referencing process [45]. In H.265|HEVC coding standard system, each slice header contains a full list of prediction picture indicators. This list includes a Picture order count (POC) which identifiers inter frame prediction order at DPB in decoder side. The RPS is keeping the DPB updated by signalling the POC of the selected references [107]. RPS is updated at slice header level, this feature help to improve bitstream robustness against errors. This referencing feature is different from the previous coding standards. For instance, in H.264|AVC coding standard a change of selected reference set (marking of current pictures) is updated in DPB before decoding the current frame [45].

3.7 Loop Filters

Due to block-based prediction process and quantisation process, a hybrid video coding system suffers from generating discontinuities of video signal across block borders. Furthermore, at high quantisation values, a blackness effect increases significantly [108]. Thus, to reduce a blackness effect in H.265|HEVC coding standard system, a deblocking filter aims to filter these visual effects. In 2012, Norkin et.al. worked on enhancing deblocking filter capable of reducing the visible artefacts at block boundaries [108]. There are two filter types employed in H.265|HEVC coding standard; 1) Deblocking filter, 2) Sample Adaptive Offset (SAO). The deblocking filter gives more accurate detection of available artefacts at block boundaries [108].

Next, the obtained signal is passed to another filter called Sample Adaptive Offset (SAO). The SAO filter aims to improve the accuracy of reconstructed video signal. The HEVC deblocking filter has lower computational complexity and better parallel processing capabilities compared to the H.264/AVC deblocking filter [108].

In ASO filter, the encoder uses lookup table to adaptively categorise different offset values on the reconstructed bitstream video samples[109]. The loop filters are implemented before saving the pictures in decoded picture buffer and after inverse quantisation process in both decoder and the encoder sides [45]. The primary targets of introducing a new loop filter design are to meet the user requirements to be more friendly with parallel processing applications in addition to reducing hardware complexity [108].

3.8 H.265|HEVC Complexity

The H.265|HEVC video coding standard is produced to meet multiple objectives. These design objectives are high coding efficiency and more friendly to parallel applications. However, these improvements come at the cost of increasing coding standard complexity. Large part of computational complexity is due to quad tree partitioning process. This recursive coding units partitioning process in H.265|HEVC coding standard consumed the largest processing part (reached to 80% of overall encoding time in HM 3) [110].

Therefore, the H.265|HEVC coding standard provides two main encoding modes depending on video applications used. These encoding modes are high efficiency and low complexity modes. The high efficiency encoding mode provides high bit rate gain at the cost of high computational complexity while low complexity encoding mode provides best trade of between computational complexity and coding efficiency [11]. In recent years, there has been an increasing amount of literature on reducing computation complexity of H.265|HEVC coding standard. In 2013, computational complexity analysis on H.265|HEVC coding standard was conducted by F. Bossen and his colleagues. The study aims to reduce H.265|HEVC decoder complexity for real time applications. However, the encoding process is several times more complex than H.264|AVC coding standard [111]. In the same year, S. Ma et al. proposed low complexity rate distortion optimisation. Their obtained results show that the encoding time has been reduced to 30% compared to H.265|HEVC reference encoder [112]. In 2014, G. Correa et. al. proposed scalable H.265|HEVC encoder to support high resolution real time video applications [113]. To reduce computational complexity, the authors proposed adaptive method to control the shapes of prediction units and maximum tree depth in each coding tree block. The achieved results report that the encoding processing time scaled down to 50% with PSNR quality loss (1.41 dB) [113]. In 2015, Y. Zhang et. al. proposed machine a learning method to reduce computational complexity of bit depth allocation at CU level. The achieved experimental results show that the computational complexity reduced by 51.45 % on average at increase in bit rate to only 1.98% [114]. In 2018, M. Xu. et. al. proposed deep learning approach to reduce H.265|HEVC encoding complexity. The proposed approach is based on using convolution neural network (CNN) and Long Term and Short Term Memory (LSTM) network to predict CU partitioning at inter and intra prediction process [110].

According to JCT-VC common test conditions [115], there are mainly three encoding configuration settings; Low delay B slices, All intra (AI), and random access (RA). In low delay-B configuration, the first frame is encoded with intra frame type and the following frames are encoded as redundant frames with bi-directional B-frames which give higher coding efficiency and coding delay than redundant frames with uni-prediction P-frames [116], [45]. Furthermore, the bi-direction prediction in general is one of the main contributing factors of increasing complexity of video coding standard hardware, in addition it requires higher decoding buffer capabilities.

For AI encoding configuration, intra mode is used to encode the whole video sequence. This encoding type gives low encoding time but requires very high encoding rates.

For RA encoding configuration, the encoded video frames are organised in hierarchical B structure. This mode gives higher compression efficiency than other encoding modes. However, it is not suitable for low delay applications because it requires more processing for reorganising the decoding pictures order at the far end decoder.

For reducing computational complexity and hardware implementation cost at H.265|HEVC video decoder, there are three coding tools that have been added and these coding tools can be illustrated as follow [45], [89]:

3.8.1 Tiles

A slice can be represented by a group of CTUs that can be decoded independently as depicted in Figure 3.18 (a). A tile includes a group of coding tree units arranged in rectangular areas, in way that each frame is divided into horizontal and vertical boarders as seen in Figure 3.18 (b). The main aim of using tiles in picture partitioning is to support parallel processing applications due their capabilities to synchronise multi threat process. Furthermore, tile partitioning can provide random access to local regions of the video pictures of HEVC bitstream [45]. However, the tile coding structure does not support error robustness against errors in H.265|HEVC coding standard. The scanning order in tile partitioning starts from the top left of each divided tile [76].

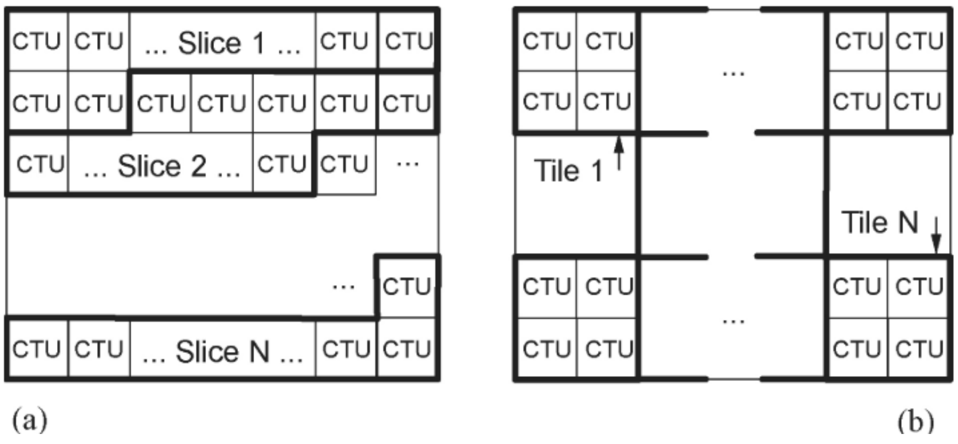


Figure 3.18 Picture partitioning using a) Slices, b) Tiles [76]

3.8.2 Wavefront Parallel Processing (WPP)

The Wavefront Parallel Processing (WPP) is an optional coding partitioning feature to support high level parallel processing applications [117]. The WPP tool has two main benefits. One is to reduce the computation processing speed of the coding standard system, and the second benefit is to support Maximum Transfer Unit (MTU) size matching [45] [117]. The process of encoding CTUs with activation of WPP can be demonstrated in Figure 3.19.

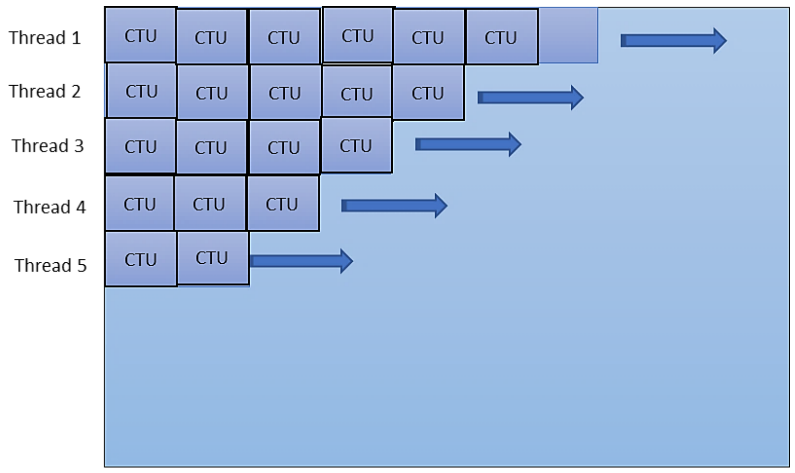


Figure 3.19 Partitioning of CTUs in Wavefront Parallel Processing (WPP)

When WPP is activated at the encoder, the WPP partitioning process start by dividing each frame into rows of CTUs. Then, each CTU row is processed independently. Between each CTU row, a delay of two consecutive CTUs in the second row is used [102], [118]. The purpose of using slices in WPP is to provide higher compression gain at the same time reducing visual artefacts that can be produced using tiles [118].

3.9 H.265|HEVC profiles

The primary role of introducing different coding profile is to get a maximum benefit of interoperability between different devices. For example, streaming video applications and video broadcast services [119]. Thus, the coding profile can be defined as restrictions of video coding standard implementation on specific video applications. These restrictions are defined coded bitstream combabilities to video decoder side. There are different hardware capabilities of video decoders. These capabilities are defined decoder design in terms of minimum hardware requirements, computational complexity cost, and video coding efficiency, in addition to error resilience tools.

In H.265|HEVC coding standard, the first version which was published in 2013 has three profiles: Main still picture, Main, and Main 10. All profiles in the first version supports video applications with 4:2:0 Chroma format only and bit depth 8-10 bits per sample [9]. The second version of H.265|HEVC coding standard was approved in 2014. There are many video coding extensions were added in this version mainly to support Scalable Video Coding, and Multiview Video Coding (MVC). For instance, it includes one Multiview profile, two scalable extension profiles, and 21 range extension profiles. Also, the bit depth of chroma sampling is increased to include 4:4:4 and 4:2:2 bit depth formats [9] [120]. The third version of H.265|HEVC coding standard is produced in first quarter of 2016. The main objectives of this version are to support video applications with 3D coding and Screen Content Coding (SCC) profile extension [12]. The screen content extension profiles mainly produced to support related wireless video display applications. Furthermore, the encoded bitstream in the third standard version conforms with both real and computer-generated contents with coding of 4:4:4 chroma format. Further details on the extended profiles for the third H.265|HEVC version can be summarised in Table 3.1 [12].

Table 3.1 Application scenarios for extended video coding standard profiles

Application scenario	Usage
Digital video broadcasting	Applications with 4:2:2 chroma sample at 10 bits per sample
Professional camera video capturing	Supports applications with 4:4:4 chroma format
High Dynamic Range (HDR) compression	Support applications with up to 16 bits per sample
Improved lossless compression	To be used with medical imaging and content preservation.
Screen Content Coding (SCC)	Supports end user devices with wireless display capabilities for 4:4:4 chroma format with up to 10 bits per sample.

3.10 Video Quality Evaluation

Video quality characteristics are affected during passing video signal into a chain of processing components related to video encoding /decoding and video transmission. The processed signals may suffer from video quality degradation in one or all signal processing complements which gives low quality of experience to the end users. Thus, evaluating the perceived visual quality is an important task to test the coding efficiency of a specific video coding system. In general, there are two video quality evaluation methods: objective and subjective evaluations methods.

3.10.1 Objective Evaluation Method

The most widely used objective quality metric is Peak Signal to Noise Ratio (PSNR) metric, which can emulate the perceived video quality as observed by human visual system. The PSNR metric has been widely used in image and video processing measurements due to being relatively simple in implementation [121]. Furthermore, a PSNR quality metric considers to be one of the most reliable indicator of visual quality variations in video development algorithms in both industry and academia. In addition, it is used as a reference benchmark for video quality metric measurements in developing video coding standards context [122]. The Y-PSNR calculation has come to be used by finding signal energy and noise energy. For each frame, a pixel in Luminance Y component of the reference frame (signal energy) is compared with a processed frame pixel (noise energy). The PSNR objective metric can be calculated as logarithmic scale as in equation (3.3) [121].

$$\text{Y\_PSNR} = 10log_{10}\left[\frac{(2^n-1)^2}{\frac{1}{x}\sum_i\sum_j\left(Y_{ref}(i,j)-Y_{prc}(i,j)\right)^2}\right] \qquad \text{Eq.(3.3)}$$

Where $(2^n - 1)$ is the square of the peak signal value and (n) refers to number of bits per pixel in luminance component, $Y_{ref}(i,j)$ represents the pixel values of the referenced image and $Y_{prc}(i,j)$ represents the pixel values of the processed frame and X is a total number of pixels in the frame [123].

3.10.2 Subjective Evaluation Method

The subjective quality is a highly interpretative assessment method where perceived visual quality in human visual system is a main factor in video quality evaluation. One of the most common method used to evaluate perceived visual quality subjectively called Degradation Category Rating (DCR) [124]. The DCR method use 10 score points (11 levels) for conducting subjective evaluation. A lowest quality level is (0) and highest value is (10). The average rating score is called Mean Opinion Score (MOS). However, this method requires high manpower, costly setup environment, and spending a great deal of evaluation time. More details about method implementation can be found in [124] and [125]. The other method is called frame by frame quality assessment, in this method a reference frame (unprocessed frame) is compared with the processed frame using frame by frame subjective assessments. in this thesis, the quality assessment evaluation is processed based on third party tool called MSU Quality Measurement Tool version 11 [126]. This openly available tool allows to evaluate multiple processed video frames at once. Furthermore, it evaluates the geometry mismatch between the original frame and the processed frame.

3.11 Video framework Evaluation Setup

A significant amount of research work has been spent to develop a video evaluation framework that support most video coding standards. A most popular open source evaluation platform is Testing video Transmission framework [127]. It is an evaluation platform proposed to provide more realistic video testing simulation for video coding standards. Another proposed video evaluation framework called Evalvid framework [128]. Evalvid framework provide objective metrics for Quality of Service (QoS) assessment of video quality delivery. These evaluation metrics are the PSNR and the fraction of decodable frames [129]. The literature on proposing video evaluation platforms has highlighted several reliable open source video evaluation platforms. However, all the mentioned evaluation platforms do not support the H.265|HEVC bitstream structure. A newly developed video quality framework to support H.265|HEVC video bitstream is presented in this chapter. More details about quality evaluation tools functionalities and process overview are described in the following section.

3.11.1 Video Quality Evaluation Process

A structure of evaluation platform with its main tools are demonstrated in Figure 3.20. A raw video data in YUV format is firstly encoded with video encoder; the generated output video bitstream is in binary format (.bin). The encoders can accept any resolution with video source of YUV format and 8-bit depth sample. A Transmitted NAL units are traced with sender tracer tool, the identification numbers of NAL units are traced and recorded in sender log file. The output bitstream file which generated from the testbed network can be used as input to be subjectively evaluated video decoder side. In the same time, it will be used objective evaluation process. The same with the receiver tracer tool, it records all the received NAL units at the receiver side passing through (a simulated network or real network environment). The output both log files from NAL evaluator tool are compared between transmitter and receiver sides. In this tool, each NAL unit header ID is parsed and checked with sender log file. A NAL header checker tool is responsible for checking the transmitted and received NAL unit header information, in addition, it is responsible to make sure the generated files are decodable at video decoder side. For injecting a variety of packet loss rates (PLRs) into the targeted bitstream, a modified version of Network abstraction layer (NAL) loss software [130] is utilised to support the proposed evaluation platform. As the NAL unit structure of H.265|HEVC coding standard is different from previous standard, the main modifications are focused to support the new NAL unit structure (VCL-NAL and non-VLC NAL).

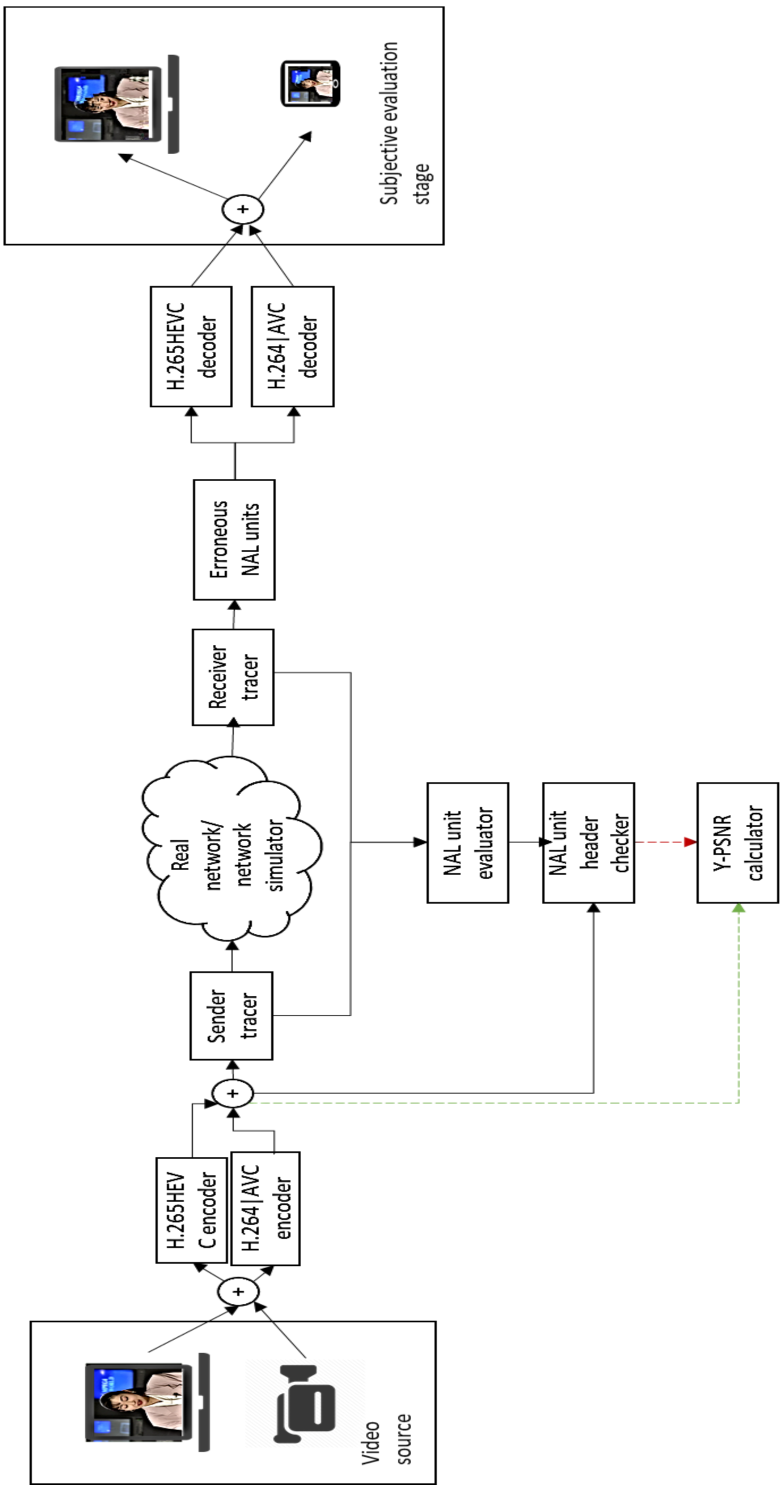


Figure 3.20 Overall video evaluation framework

3.11.2 Hardware and Third-Party Tools Requirements

This section reports the implemented hardware and software used in experimental work. The video coding standard is compiled with Microsoft Visual studio package. The used programming language is C++. The PC hardware specifications used to encode video test sequences are Intel Core i7 2.3GHz four-core, NVIDIA GeForce GT 750M, with 16GB RAM- memory. The other two PCs are used with hardware and software specifications: Linux operating system (Linux Ubuntu LTS version 16.04 OS with minimum x86_64: gcc version 4.8). These two PCs are used for streaming and testing purposes of encoded video bitstream in error-prone environments. A network simulator version 3 (NS3) is implemented in the proposed evaluation platform to simulate the streamed video with different network environments. For NS3 implementation inside windows operating system environment, an open source tool named (Cygwin) version 1.7 should be installed on a Windows operating system.

3.12 Average bitrate saving comparison study

This section evaluates the video quality performance of the latest two video coding standards i.e. (H.264|AVC and H.265|HEVC) in error-free and error-prone environments. The evaluation study includes objective and frame by frame assessments. Moreover, this study will include the average bit rate saving for different video resolutions from low-resolution QCIF (176x144) resolution up to 4K (3840 x 2160) resolution.

3.12.1 Video encoding configurations

In this section, the encoding settings of reference software for both video coding standards are presented. For H.265|HEVC coding standard, a reference software called HEVC Test Model (HM) version 16 [131] is chosen in the evaluation study whereas for H.264|AVC, a reference coding software called Joint Test Model (JM) version 19 [132] is selected.

For both video coding standard s, a random-access configuration is selected to achieve highest possible quality than low pass encoding settings. The main profile is selected for H.265|HEVC coding standard and high profile is chosen as it is the highest efficient profile in H.264|AVC coding standard. The video sequences characteristics are reported in [Appendix A]. the encoding configuration settings for both video coding standards are reported in Table 3.2.

Table 3.2 Encoding configurations for bitrate savings experiment

Parameter Name	description
Profile	Main
Encoding GOP size	8 pictures
Filtering	Enabled
Search range	64 block size
bit depth	8 bits
Intra mode	1st frame in each GOP
Fast merging decision	Enabled
Number of Reference frames	4 frames
(max/min) transform unit size	32/4 (unit size)
Asymmetric Motion Partitioning (AMP)	enabled
Hierarchical B frames	4 frames

This comparison study aims to calculate the average bit rate saving for H.265|HEVC and H.264|AVC video coding standard s with frame resolution ranging from low (QCIF) to high (UHD) resolutions. video sequences are selected. A full list of selected video sequences with its characteristics can be found in [Appendix A].

A bit rate saving is calculated based on Bjøntegaard-Delta bit-rate (BD-BR) measurement [133].

The calculation process is demonstrated in Figure 3.21. If the result is in minus, it means there is a bit rate saving [133]. The experiments were conducted using the same encoding settings. An average Y-PSNR vs Bit rates values is taken for more than one video test sequence with same video resolution.

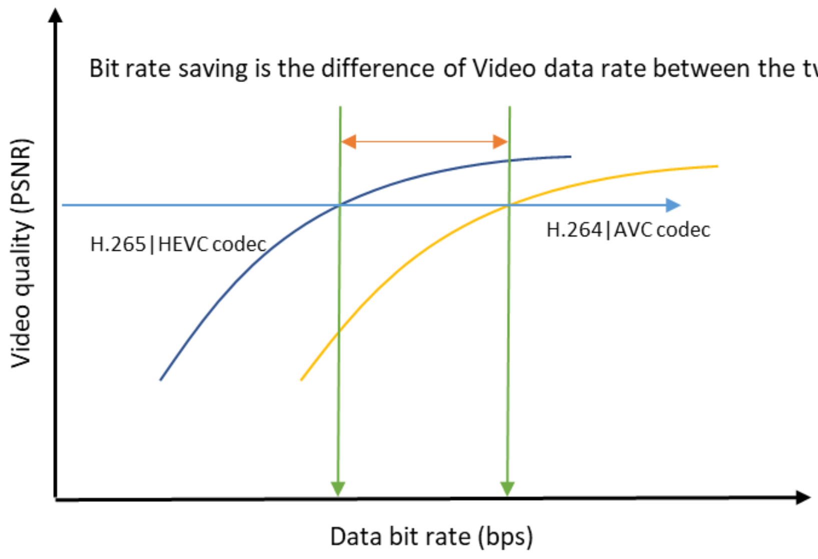

Bit rate saving is the difference of Video data rate between the two codecs
H.265|HEVC codec
H.264|AVC codec
Video quality (PSNR)
Data bit rate (bps)

Figure 3.21 Bjøntegaard-Delta bit-rate (BD-BR) measurement using same PSNR values

As shown in Figure 3.22. the negative percentage value for each video resolution category represents the average bit rate saving percentage in H.265|HEVC coding standard using HM reference software compared to H.264|AVC coding standard using JM reference software. It can be noted that the bit rate saving increases with increasing the video frame resolution. It is apparent from the line chart that the H.265|HEVC coding efficiency works better with higher resolutions, which is one of the primary coding standard targets [134].

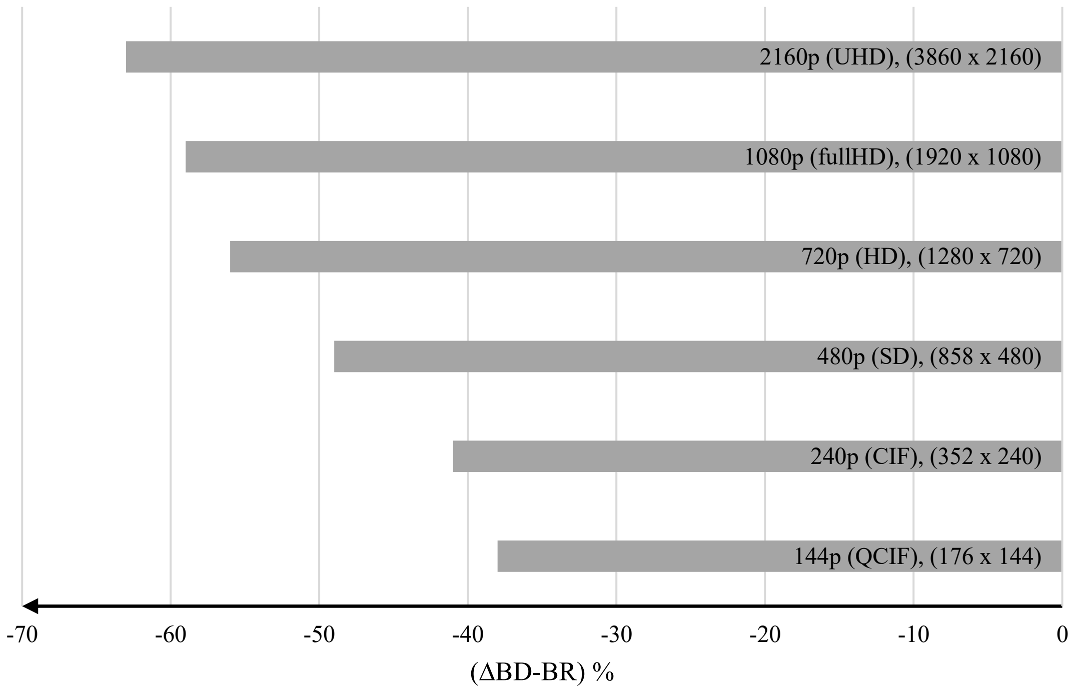


Figure 3.22 Average bit rate saving for H.265|HEVC compared with H.264|AVC coding standards

To sum up, the coding efficiency of H.265|HEVC increases proportionally with increasing video spatial resolution. Three facts can explain the high bit rate gain. Firstly, H.265|HEVC coding standard employs larger coding units with flexible sizes of up to (64x64) compared to H.264|AVC coding standard unit size of (16x16). Secondly, using advanced motion vector production in the compressed video samples in the interprediction process. Thirdly, the intra prediction modes increase up to 33 directional modes compared to only eight directional modes in H.264|AVC coding standard. These factors improve the coding efficiency specialty in high spatial resolutions and bitrate saving compared to the previous coding standard.

3.13 Codecs Comparison Study in Error-Free and Error-Prone Conditions

This study aims to analyse the coding standard performance of H.265|HEVC and H.264|AVC video coding standards in two transmission environments: without injecting errors to the encoded bitstream (error-free condition) and with injecting errors at various bit error rates (error-prone conditions). Two video test sequences are selected in this comparison study with two video resolutions (QCIF and CIF resolutions), more sequence details are reported in [Appendix A]. Akiyo video sequence has low motion activity which classified according to its motion activity and texture details as Class A. Another selected video sequence named (silent) is classified as class B which has higher motion activities and texture details.

Furthermore, the study includes testing coding standard performance with different encoding bit rates (in kbps). Each video sequence has 300 frames, with display aspect ratio (4:3). For error-prone evaluation experiments, and to achieve fair study each video test sequence is repeated 30 times. Then, the average Y-PSNR values of the test results are recorded.

3.13.1 Video Encoding Configurations

In this section, the encoding settings for H.265|HEVC and H.264|AVC coding standards are reported in Table 3.3 and Table 3.4, respectively. To get the highest possible quality results, a random-access encoding configuration is selected rather than using low pass encoding configuration.

Table 3.3 H.265|HEVC Encoding settings for error-free and error prone experiments

Parameter Name	description
Profile	Main
Encoding GOP size	8 frames
Filtering	Enabled
Search range	64 block size
bit depth	8 bits
Intra mode	1st frame in each GOP
Fast merging decision	Enabled
Number of Reference frame	4 frames
(max/min) transform unit size	32/4
Asymmetric Motion Partitioning (AMP)	enabled
Hierarchical B frames	4 frames

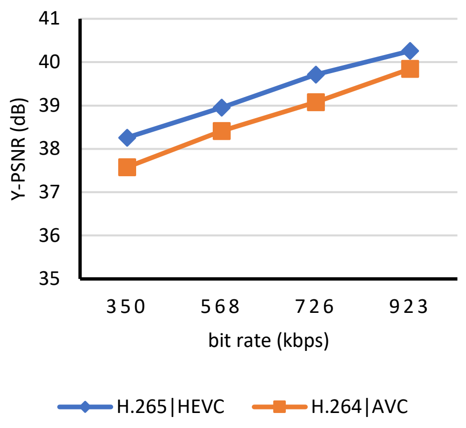
Table 3.4 H.264|AVC Encoding settings for error-free and error prone experiments

Parameter Name	value
Profile	High
Encoding GOP size	8 frames
Intra mode	1st frame in each GOP
Filtering	Enabled
Search range	64 block size
bit depth value	8 bits
Hierarchical B frames	4 frames
No. of Reference picture	4 frames
Search range	64 block size

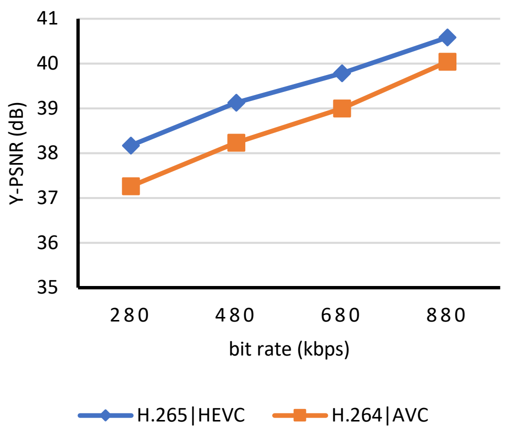
The main profile is selected for H.265|HEVC coding standard whereas a high profile is chosen to achieve highest coding efficiency in H.264|H.264|AVC video coding standard. A rate-distortion optimisation for both video coding standards is enabled. The video sequences characteristics are reported in Appendix A. In the evaluation study, HM16 reference software [131] is chosen to encode H.265|HEVC video bitstream while for H.264|AVC coding standard a reference software JM19 [132] is selected during the experimental work.

3.13.2 Objective Evaluation

This section presents the objective comparisons results between H.265|HEVC and H.264|AVC
Video coding standards in both error-free and error-prone settings. For both coding standard s, the same hardware and software are used for evaluation work are selected for encoding video test sequences. Best possible efforts have been spent to keep nearly the same encoding parameters. Firstly, the two test sequences are compared with different bit rates ranging from 280 kbps to 923 kbps in error-free environment. The obtained results are depicted in Figure 3.23 and Table 3.5.



(a) Akiyo sequence, CIF resolution



(b) Akiyo sequence, QCIF resolution

Figure 3.23 Video quality vs Bit rates comparison in error free condition

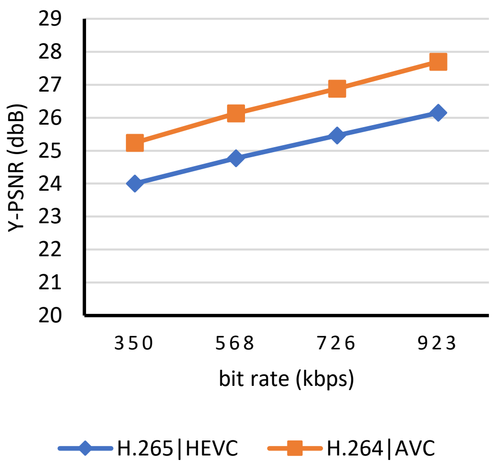
Table 3.5 Video quality vs Bit rates in error-free conditions Akiyo (CIF resolution)

Bitrate (kbps)	H.265|HEVC	H.264|AVC	Y-PSNR difference
350	38.261	37.579	0.681
568	38.952	38.412	0.539
726	39.712	39.080	0.631
923	40.259	39.844	0.414
Average	39.296	38.729	0.566

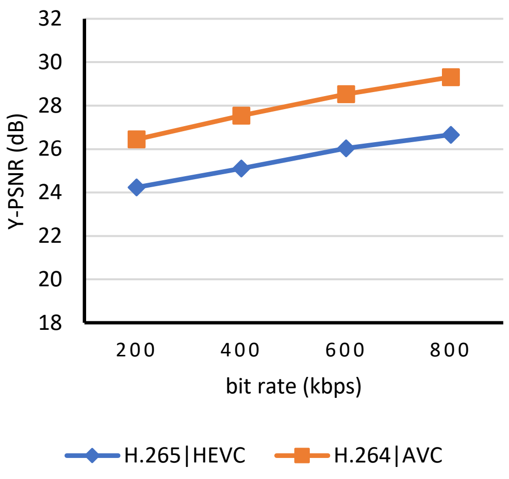
The results show that visual quality PSNR results have improved with an average Y-PSNR gain of 1.5 dB and 1.9 dB for the decoded H.265|HEVC sequences with QCIF and CIF spatial resolutions, respectively.

Figure 3.24 compares both coding standard s performance in error-prone environments subjected with random bit errors (1×10^{-4}) %. Each test is repeated 30 times and recorded the average (Y-PSNR) for the same encoded bit rates.

Looking at error prone figure, a high-quality degradation is observed in the H.265|HEVC decoded video sequences compared to H.264|AVC decoded sequences under the same error conditions. More objective comparison details can be depicted in Table 3.6.



(a) Akiyo sequence, CIF resolution



(b) Akiyo sequence, QCIF resolution

Figure 3.24 Video quality vs Bit rates for the HM16 and JM19 reference software with BER (1x10^-4) %

Table 3.6 Video quality vs Bit rates for Akiyo sequence in error-prone conditions (CIF resolution), BER=(1x10^-4) %,

Bitrate (kbps)	H.265|HEVC	H.264|AVC	Y-PSNR difference
350	24.003	25.242	-1.238
568	24.773	26.130	-1.356
726	25.462	26.883	-1.421
923	26.147	27.697	-1.550
Average	25.096	26.488	-1.391

Both coding standard s do not give acceptable visual quality regarding Y-PSNR results. However, On average the Y-PSNR degradation in H.265|HEVC decoded frames is (-2.45 dB) less than decoded H.264|AVC bitstream.

Looking at the figures in Table 3.5 and Table 3.6, we can find that the H.265|HEVC coding standard performance in error-prone conditions is worse than the H.264|AVC coding standard performance.

To sum up, the experimental results reveal that at low encoding bit rates (in several kbps), the H.265|HEVC coding standard performance suffers higher than H.264|AVC coding standard because the motion compensation mechanism in H.265|HEVC standard employs more data dependency which make more vulnerable to errors and resulted in losing decoding synchronisation.

3.13.3 Subjective Evaluation

This section includes the achieved subjective quality results in terms of frame by frame assessments comparative study. Figure 3.25 shows the achieved comparisons results for silent video test sequence (CIF resolution) encoded at 400 kbps. The extracted frames are randomly selected.

The erroneous decoded frames from both coding standard s are compared with original frames (error free frames). The middle column in the figure represents the decoded frames using reference software (JM14.2), the right column represents the decoded frames using reference software (HM16).

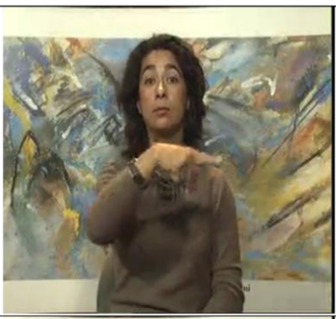
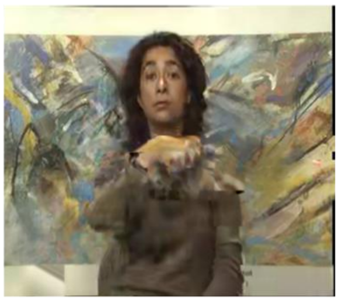
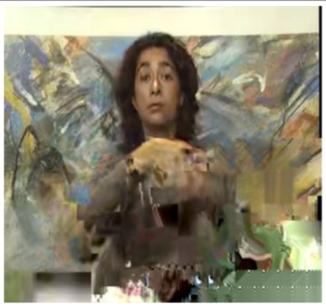
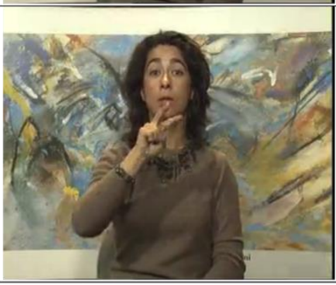
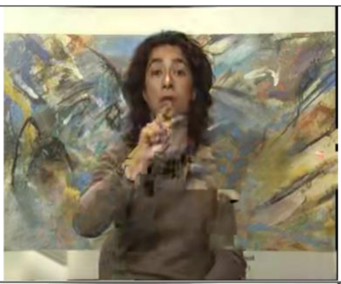
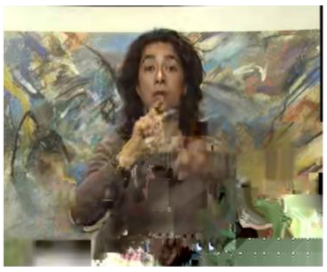
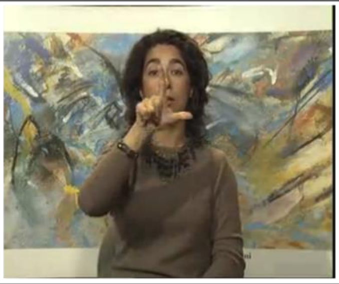
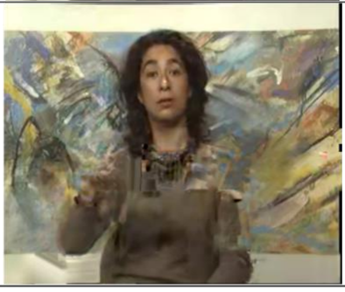
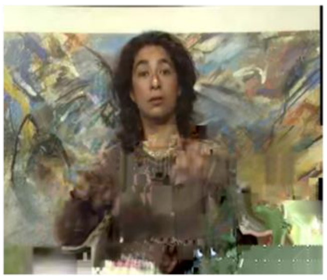
Frame number	Original frame	H.264|AVC codec (JM14.2)	H.265|HEVC Codec (HM16)
Frame 80th			
Frame 95th			
Frame 115th			

Figure 3.25 Frame by frame quality assessment between H.264|AVC and H.265|HEVC in error-prone condition

As it can be seen, the visual quality in both decoded frames suffered from losing part of the woman body. However, in H.265|HEVC decoded frames, a larger frame area is suffered from spatial error propagation compared with H.264|AVC decoded frames.
The achieved results of this study show that the video quality degradation generated from H.265|HEVC coding standard is higher than quality degradation level with H.264|AVC coding standard. The most obvious finding to emerge from the subjective and objective analysis is that the encoded motion information in H.264|H.264|AVC coding standard has less temporal and spatial redundant informational than H.265|HEVC coding standard. It means the spatial and temporal redundant data are employs on larger set of motion vector predictors to increase the coding performance. However, this increase in coding performance will affect on the perceived visual quality when dealing with errors. Another main reason is that the basic coding unit in H.265|HEVC coding standard is flexible in sizes is flexible in size with Largest coding unit size (16x16, 32x32, 64x64). This coding feature leads to cause sever distortion of the decoded visual quality.

3.14 Error Sensitivity Evaluation in H.265|HEVC Coding standard system

In this evaluation study, encoding sensitivity is evaluated with different bit error rates in H.265|HEVC coding standard. The evaluation work includes two compressed video types: motion data and NAL unit's data. This work is done using a reference software manual version HM16.06 for H.265|HEVC coding standard. The experimental work is conducted in error-free and error-prone environments using different Bit Error Rates (BERs).

3.14.1 Encoding Configuration Parameters

In error sensitivity codec evaluation, a reference software HM 16 is chosen in the experimental work. A video sequence named Akiyo with CIF resolution is selected as the input video test sequence with frame rate of 30 fps [135].
A bit error rates are selected at BER $(0,4,8,10) \times (10^{-4})\%$. A random packet loss rate is generated with different seeds. At the encoding stage, the main profile is employed to encode video test sequences as a typical profile for most consumer devices. The same encoding setting reported in Table 3.3 is employed. The test sequence is encoded at 26

slices per frame. The sampling colour information is 4:2:0 in (YUV format) with 8-bits per sample.

3.14.2 H.265|HEVC error sensitivity evaluation process

The input video bitstream is encoded using the same configuration setting in both testing environments, i.e. error and error-prone. A Y-PSNR is the basic objective metric used in objective evaluation. The overall process block diagram can be demonstrated as shown in Figure 3.26. The objective PSNR calculations are done in (YUV format). After encoding a raw video test sequence, the encoded bitstream is saved as an error-free bitstream. The same encoded bitstream is injected with different bit error rates. Each test condition is tested after the decoding stage.

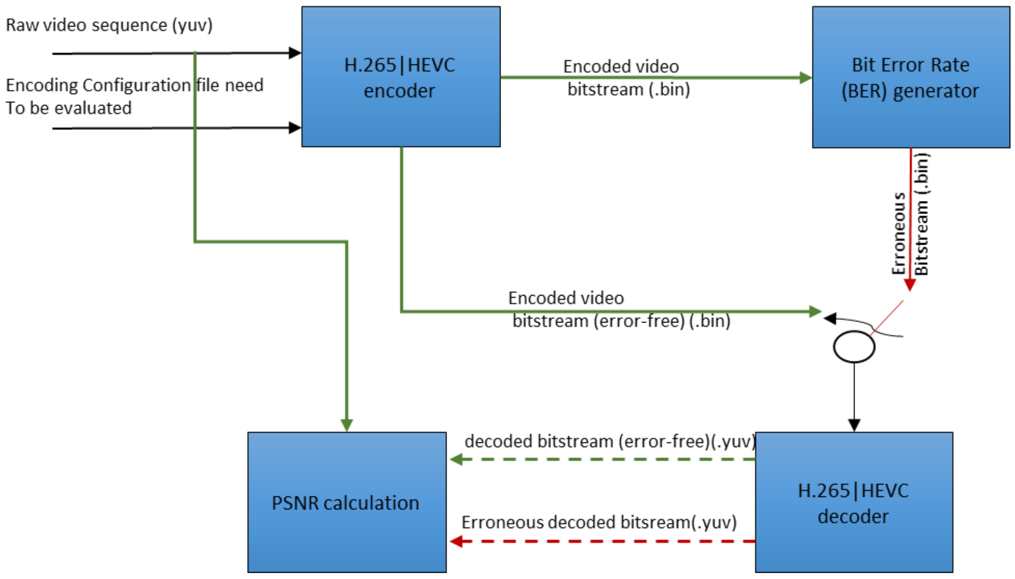

Raw video sequence (yuv)
Encoding Configuration file need To be evaluated
H.265|HEVC encoder
Encoded video bitstream (.bin)
Bit Error Rate (BER) generator
Erroneous Bitstream (.bin)
Encoded video bitstream (error-free) (.bin)
decoded bitstream (error-free)(.yuv)
Erroneous decoded bitsream(.yuv)
PSNR calculation
H.265|HEVC decoder

Figure 3.26 Error sensitivity evaluation process diagram

3.14.3 Error sensitivity on NAL units

The main purpose of using NAL units in both H.265|HEVC and H.264|AVC coding standards is to support various video transmission systems [25]. The NAL units are divided into two main parts; non-VCL NAL units and VCL-NAL units. The VCL-NAL units data which represents about 95% of the total encoded video data includes only video samples representation without data control. The second part is non-VCL NAL units data which contains the most sensitive shared header control information [45].

The primary objective of this experimental work is to study the effects of applying different BERs on the two NAL units' parts generated from H.265|HEVC coding standard. Figure 3.27 shows the effect of injecting various BERs on the two NAL units data types. The blue line refers to achieved objective quality when various BERs are injected into NAL units (evenly distributed to video sequence). The red line represents the VCL-NAL units. The grey line refers to the achieved objective quality when 5% of the total injected BERs are injected on encoded sensitive information (non-VCL NAL units).

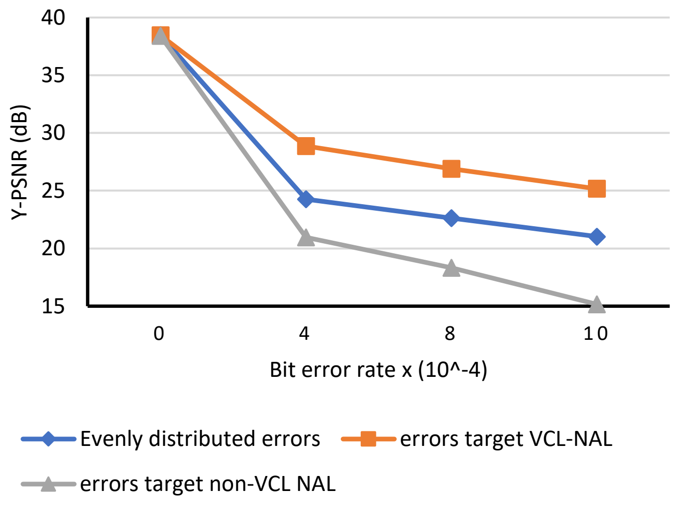


Figure 3.27 Encoding error sensitivity for non-VCL NAL and VCL-NAL units in error-prone conditions with different BERs

From comparison objective results (Figure 3.27), it is shown that when 5% of overall BERs are applied on non-VCL NAL unit, the obtained objective quality is (4.33 dB) higher compared than when injecting BERs in evenly distributed (normal) on all NAL units.

For gray line representation, when 5% of the injected bit errors is applied on non-VCL NAL units., the quality is reduced by (8.83dB) compared to evenly distributed BERs injection,

What can be clearly seen in this figure is the steady decline for both VCL and NON-VCL NAL units. What is striking is that the NON-VCL NAL data suffers from high degradation in Y-PSNR values. As the NON-VCL NAL data contain a shared information for more than one frame which effects on the reconstruction quality. On the other side, a VCL-NAL data contain video samples representation which effect only on the current frame at slice level.

3.14.4 Motion data sensitivity

In this experiment, we mainly focused on finding the effects of losing motion data information on perceived visual quality. The motion data includes prediction data derived from spatial and temporal surrounding information such as motion vector fields (predicters candidate set). Same encoding configuration settings reported in Table 3.3 are used in the experimental work.

Figure 3.28 and Figure 3.29 show an achieved objective and subjective results, respectively. Figure 3.28 shows two lines, the red line with square shapes refers to BER (10^{-4}) on all coded video data. The blue line with rhombus shapes refers to injecting (5%) of BER (10^{-4}) on encoded motion data.

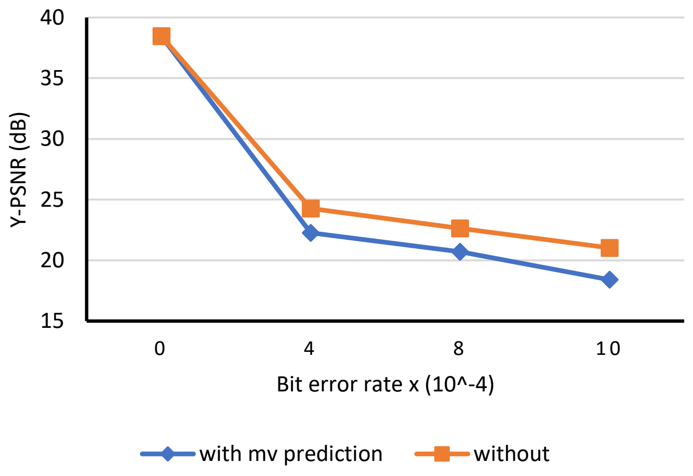


Figure 3.28 Objective results of error sensitivity to motion vector prediction data

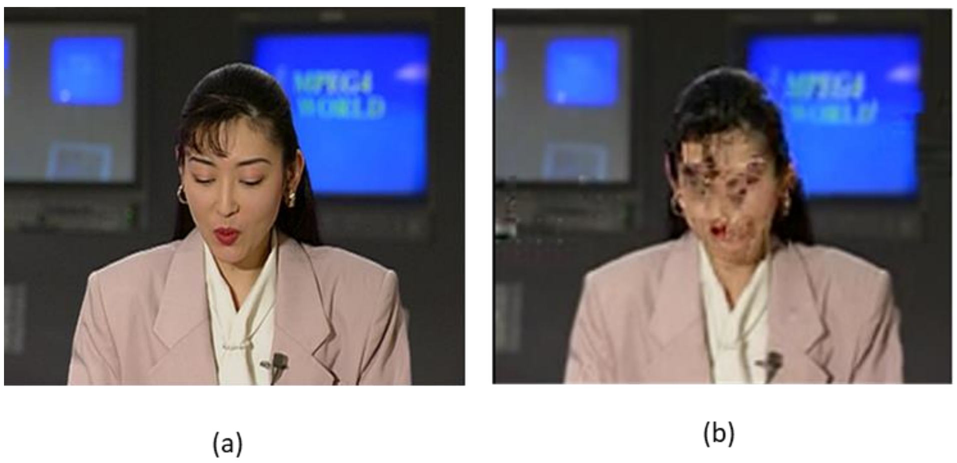


Figure 3.29 Perceived visual quality effect on Akiyo video sequence
(a) decoded frame, error-free, (b) (5%) decoded frame with BER (10^{-4}) on head part

It can be noticed from the achieved results that the propagated spatial error in the head part results from the inter prediction errors which is highly dependent on the motion vector fields. As a result, the reconstructed picture suffers from severe degradation in large frame areas.

3.15 Comparison between HEVC-H.265 and H.264 video standards

A technical coding tools comparison between H.265|HEVC and H.264|AVC video coding standards can be summarised in Table 3.7.

Table 3.7: Video coding comparison between H.265|HEVC and H.264|AVC standards

	H.265|HEVC	H.264|AVC
Names	ISO/IEC 23008-2 MPEG-H Part 2, ITU-T H.265	ITU-T H.264, ISO/IEC MPEG-4 AVC standard
Published date	2013 [136]	2003 [137]
Coding efficiency	Its coding efficiency increased about 50% higher compared to its predecessor H.264/MPEG-4 AVC coding standard [138].	Achieved an increase about 50% in coding efficiency compared to its predecessor H.262/MPEG-2 coding standard.
Design	It is designed to support parallel processing applications and enhance video quality at the same bit rate with increased advanced display technology. Furthermore, supporting 2K and 4K video delivery in conventional networks [111].	It is designed to support both low- and high bit-rate video coding in order to accommodate various transport layers and storage media [17].
Motion compensation block size	64 × 64, 64 × 48, 64 × 32, 64 × 16, 48 × 64, 32 × 64, 16 × 64, 32 × 32, 32 × 24, 32 × 16, 32 × 8, 24 × 32, 16 × 32, 8 × 32, 8 × 8, 8 × 4, 4 × 8 [76]	4×4, 4×8, 8×4, 8×8, 16×8, 8×16, 16×16 [76]
Format range extensions Extension	The 2nd version published in 2014 and approved in 2015 [6]. Scalable coding extensions (SHVC), and multi-view extensions (MV-HEVC) [139]. 3D-HEVC extensions for 3D video were finished at beginning of 2015. Screen content coding (SCC) extension introduced in 2016 [140].	Fidelity Range Extensions (FRExt) with its prominent High profile, the Scalable Video Coding (SVC) extension and finally, the Multiview Video Coding (MVC) extension. (Mainly from 2003 to 2009) [137]
Entropy coding	Employ only Context-adaptive binary arithmetic coding (CABAC) [76]	Employ CABAC and Context-adaptive variable-length coding (CAVLC) [137]

Coding Unit	Macroblocks structure based on flexible sub-partitioning structures named Coding Tree Unit (CTU) with larger block size up to (64×64) and smallest size (16×16) [9].	Macroblocks structure based on fixed block size (16x16) [137].
NAL unit types	Extended to 64 NAL types, some of them reserved for future use of standard developments [6]	32 NAL unit types [141]
Video Quality	Support up to 8k UHD (8192×4320) [9]	Support Up to 4K UHD (4096×2304) [58]
Frame Rate	Work on frame rate up to 300 fps [136]	Work on frame rate up to 59.94 fps [136]
Parallel processing tools	Use tiles and Wavefront parallel processing (WPP) to independently encode/decode video bitstream [9].	Does not support parallel processing architecture [9].
Weaknesses	Several times more complex than H.264|H.264|AVC coding standard implementation [111].	Required high bit rates for high-resolution video applications [142].
Intra prediction mode	Use two non-directional modes :DC (flat mode) and planner (or surface fitting), and 33 directional modes [9].	Use nine intra modes, one DC intra mode, and eight directional modes [58].

3.16 Conclusions

This chapter of dissertation is divided into two main parts. The first part presents a detailed technical study of H.265|HEVC video coding standard. The process of encoding/decoding video bitstream of the standard is described and highlighted with technical key differences from the previous video coding standards. High level video coding syntax of H.265|HEVC coding standrd focusing on picture partitioning process, transport interface and NAL structure. Moreover, the main coding tools improvements that contribute to increase video coding efficiency is explained.

Same hybrid video coding main components of previous coding standards are retain in H.265|HEVC video coding standard. The main coding components include temporal and spatial predictions components, transform and quantisation in addition to filtering and entropy coding components. Thus, the coding process in these main components made a small significance difference in H.265|HEVC bitstream construction i.e. NAL unit structure. However, the major improvement in coding efficiency is due to introducing quad tree structure using coding tree units as a basic coding unit in the standard. In addition to block flexible partitioning concept, new tools have been introduced called

AMVP to support merging operation mode and flexible macroblock ordering which helps to increase coding efficiency significantly in interprediction process. Furthermore, the number of intra prediction modes (directional modes) has increased to 33 compared to 8 direction modes in H.264|AVC standard which contribute on enhancing video resolution as well. The block partitioning In H.265|HEVC coding standard has high flexibility with various block sizes. Therefore, the previously implemented error concealment and resilience algorithms in previous standards do not support the current H.265|HEVC bitstream structure.

Additionally, as NAL unit header structure amended to include more NAL units, some of NAL unit header bits have been removed from H.265|HEVC coding standard which were necessary to support codec error robustness such as flexible micro-frame ordering (FMO) and arbitrary slice ordering (ASO). Furthermore, the increase in video coding efficiency comes at the cost of increasing computational complexity due to involving more complex motion compensation processes than previous coding standards.

The second main part of this chapter reports and discusses the achieved experimental work evaluation of the coding standard. The experimental work includes comparison study on video coding performance between H.265|HEVC and H.264|AVC coding standards in error free and error prone conditions. The purpose of this study was to determine the effects of various packet loss rates on perceived visual quality for both coding standards using nearly similar encoding settings. Furthermore, the experimental work study set up out to include encoding error sensitivity are reported and discussed using latest reference software for coding standard. The aim of this study was to explore the effects of different encoded components (NAL unit types and motion vector fields) encoded in H.265|HEVC coding standards on decoded perceived visual quality. The study has identified the effects of motion vector prediction data and video coding layer samples and non-video coding layer samples encoded in bitstream and subjected to error prone conditions with various BERs. Furthermore, a structure of H.265|HEVC bitstream, i.e. the NAL units and the encoded motion data are evaluated. The error sensitivity study helps to identify the most sensitive compressed data to encode in higher protected channel, and to encode the less error sensitive encoded data in enhancement layer.

Chapter Four

4 Error Resilience based on Video Encoder

This chapter presents proposed error resilience algorithm based on adaptive H.265|HEVC video encoder. This chapter is organised as follows. Section 4.1 presents an overview of encoder-based error resilience tools that protect region of interest to reduce error effects on decoded visual quality. Sections 4.2 discusses the state-of-the-art techniques to extract and protect the important frames areas used in H.265|HEVC video standard. In Section 4.3, a proposed error resilience based on adaptive slice encoding (ASE) algorithm is described. Section 4.4 covers the evaluation experimental setup including hardware and software requirements. Network testbed setup, and coding configuration settings. Section 4.5 presents the achieved evaluation results of the proposed ASE algorithm in terms of objective results, frame by frame assessment, and rate distortion performance. A computational complexity of the ASE algorithm is evaluated and compared with the default reference software standard. Further, experimental work on various network congestion loads and video processing delays are conducted on evaluation performance of the ASE algorithm. Finally, a summary and chapter conclusion are presented in Section 4.7.

4.1 Introduction

A highly compressed video bitstream is more vulnerable to errors in time varying channels [143]. Multimedia protocols such as User Datagram Protocol (UDP) and Real-time transport protocol (RTP) are employed to support multimedia content delivery in efficient way. However, much of instability of using UDP in compressed video delivery can be attributed to the nature of the compressed video is very sensitive to time delay as it is coded with variable length. This compressed video nature causes the received packets at video decoder to be dropped in many situations when decoder buffer reaches its maximum limit. On the other side, RTP does not have a reliable mechanism to timely deliver video packets in sequence order. Therefore, the delivery delay becomes worse when the buffer is full of data packets with time-varying channels conditions this scenario called stochastic distribution. Figure 4.1 shows the adaptive data transmission system solution to overcome time-varying channels conditions issues.

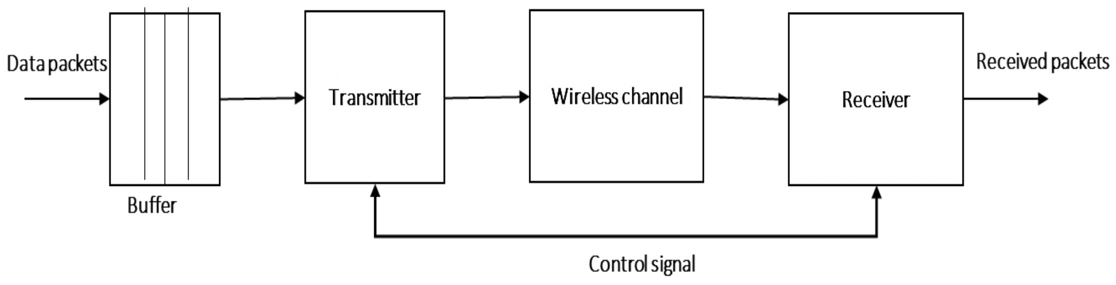


Figure 4.1 Time-varying Communication System [143]

In this system, the encoding video parameters are configured according to buffer capacity and channel conditions received which updated via receiving control signal [143].

Therefore, Video error control strategies is one of the practical solutions to reduce bit error effects on the transmitted video stream. One of the main video encoding requirements at low delay or conversational video applications is to reduce the number of reference frames to a minimum level. This low delay video processing requirements can be achieved by reducing the number of previously used future reference frames at the motion process.

The first conducted work to select a group of MBs to be encoded with intra refresh in H.264|AVC video coding was by Hoaming Chen et al [144]. Their proposed error resilience coding scheme based on adaptive intra refresh. The error resilience coding scheme selects the important regions depending on the used network conditions (packet loss rates) and video motion information. The refresh cycle sizes ranging in (4, 8, 16). A selected area depends on the PLR value in the feedback channel. When receiving low PLR values i.e. (10^{-4}), the cycle size will be selected with smaller values i.e. (4). In contrast to high error-prone environments i.e. (10^{-1}), the refreshing cycles sizes will be increased to obtain a balance between error resilience and H.264|AVC coding efficiency performance.

In this piece of research, proposed error resilience algorithm is presented to be implemented at H.265|HEVC encoder side. Experimental work has been conducted to evaluate the ASE algorithm with reference software and related state of art algorithms.

4.2 Encoding error resilience using region of interest extraction

This section presents a literature review of related state-of-the-art work of proposed error resilience algorithms for low delay video delivery applications based on Region Of Interest (ROI) extraction approaches. There is relatively small body of literature that is concerned with using ROI appraoch at video encoder to imrove error resilience at H.265|HEVC video coding standard. In these studies, the encoded moving areas are considered as important regions need to be pretected against transmission errors. In 2015, the authors proposed error resilience algorithm based on generating activity map, the moving regions are segmented into blocks and based on the maximum depth level of CTUs, they calculate the moving objects activities to be considered as ROI regions and protected them against errors at video encoder side [145]. In 2016, they improved error resilience algorithm by utilising encoding bit rate control in a region of interest extraction process and enhancing perceived video quality in error prone conditions at slight increase in bit rate overhead [146]. The obtained objective quality results presented by the authors show that a significant improvement of 0.88 dB was achieved compared with H.265|HEVC reference selection method with injecting Packet Loss Rate (PLR) of (5%) [146]. The moving region extraction methods of these studies are based on proposed work by Hai-Miao Hu et. al. in 2012 [147]. This work (ROI based rate control scheme) aims to improve the coding efficiency of H.264/AVC coding standard by allocating more encoding bit budget to moving regions and improve the perceived quality for ROI area at the cost of non-ROI visual quality. In 2008, Yang Liu et. al. proposed H.264|AVC video communication system based on resource allocation [148]. The system aims to reduce computational complexity in H.264|AVC standard and support conversational video applications.

However, as in H.265|HEVC coding standard, the partitioning of coding tree units is flexible in size, the previous ROI extraction process suffers from inaccurate selectivity of moving regions which effects on directly on perceived visual quality and coding efficiency. Therefore, an adaptive slice encoding (ASE) algorithm is developed and proposed based on understanding of previous related work. A general use case of the proposed ASE algorithm work can be demonstrated in Figure 4.2.

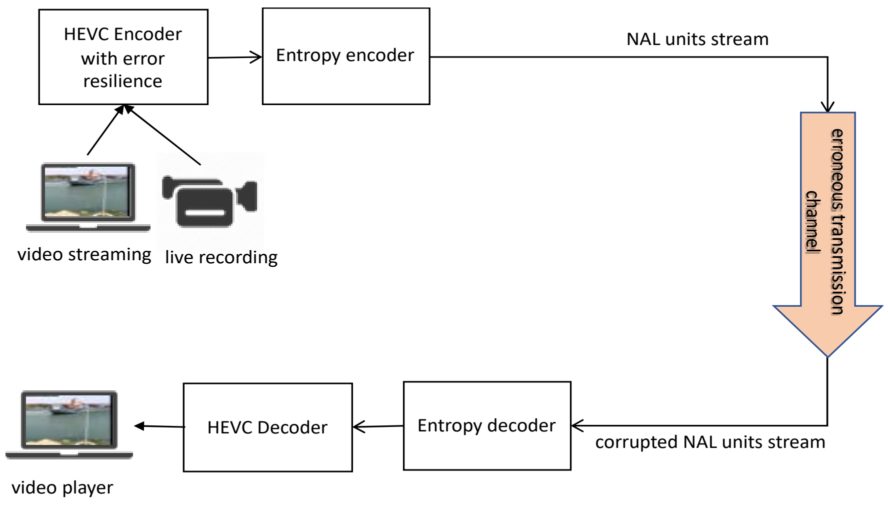


Figure 4.2 Use case scenario of ASE error resilience algorithm

One of the most challenging tasks in the region of interest extraction process in H.265|HEVC coding standard is how to keep computation complexity at minimum level. Another challenge is how to extract accurate ROI in low delay processing constraints such as conversational video communication.

Rate control is responsible for calculating the best trade-off between image quality and the required bit rate. Most of video coding systems, in general, are lossy systems, so it is important to keep a bit rate saving at highest level and at the same time maintain perceived visual quality to various quality levels according to targeted video applications.

The moving extraction process starts after the motion estimation stage, in which quantisation parameters are adjusted accordingly. There is a dilemma between region segmentation and motion estimation priorities. Quantisation parameters (QP) need to be adjusted before the rate-distortion optimisation (RDO) process start. On the other hand, motion information is generated after RDO and QP are generated before RDO process. However, in the moving region extraction process, the QP needs to be adjusted based on motion information which is considered in the literature as a ROI.

The design of ROI method should take into consideration the encoding computation processing. To reduce codec complexity overhead, the developers who work in the field are spending more bit rate budget and encoding processing power on parts of video frames that human visual system pays more attention that other parts such as the face of broadcaster news. Therefore, in real time video communication, and the extraction process needs to take into considerations: frame texture details, skin colour, object motion speed. These ROI extraction requirements lead to make rate control adaptation at the encoder more challenging task in real-time processing applications. To solve motion information and adjustments of QP dilemma, researchers in [149] proposed the motion differencing method. In this method, the motion vector of each macroblock is compared with other macroblocks of the previous frame. However, this method does not give acceptable results when dealing with fast-moving objects. In [148], the authors proposed a method to distinguish the importance of each macroblock using Mean Absolute Difference (MAD) method between the current and previous macroblocks [150]. Further, the authors in [151] and [152] achieved high accurate extraction results when dealing with relatively low motion activities with stable video background. However, when evaluation slight movement in moving object in background area (e.g. temporal changing in lighting conditions or camera zooming) leads to inaccurate selection of ROI with disastrous visual effects [152].

4.3 Error resilience based on Adaptive Slice Encoding (ASE) Algorithm

The aim of the proposed algorithm is to reduce error propagation at slice level. An adaptive encoding algorithm is introduced at video encoder to encode and protect the most active slices. Therefore, it is one of the practical ways to support low delay video delivery applications. A general review process of ASE algorithm can be described in the flowing.

A coded video sequence is represented as a series of Access Units (AUs) in sequential order with shared sequence parameter. Each access unit is represented by a group of NAL units. A prefix code of access unit delimiter is used to identify the start of new AU in NAL unit bitstream. A primary encoded AU contains a group of VCL NAL units which includes one or multiple slices. These slices represent real video samples data. A redundant coded picture is encoded as additional VCL NAL units. These additional VCL samples are used in the case when the original or primary video samples are lost or corrupted. In this case, the decoder will parse the contents of the correctly received data to recover the corrupted video samples. In error-free conditions, the decoder will discard received additional redundant video data. At the end of each video coded sequence, a non-VCL NAL unit is encoded to indicate the end of the NAL units bitstream.

The concept of the proposed algorithm is illustrated in Figure 4.3. Suppose the ship in the video sequence is the most important area that requires protection against errors. This algorithm will extract the active slice, i.e. the ship which is the active slice area, then encoding the active areas with intra mode. More details on ASE algorithm implementation is described in the following subsections.

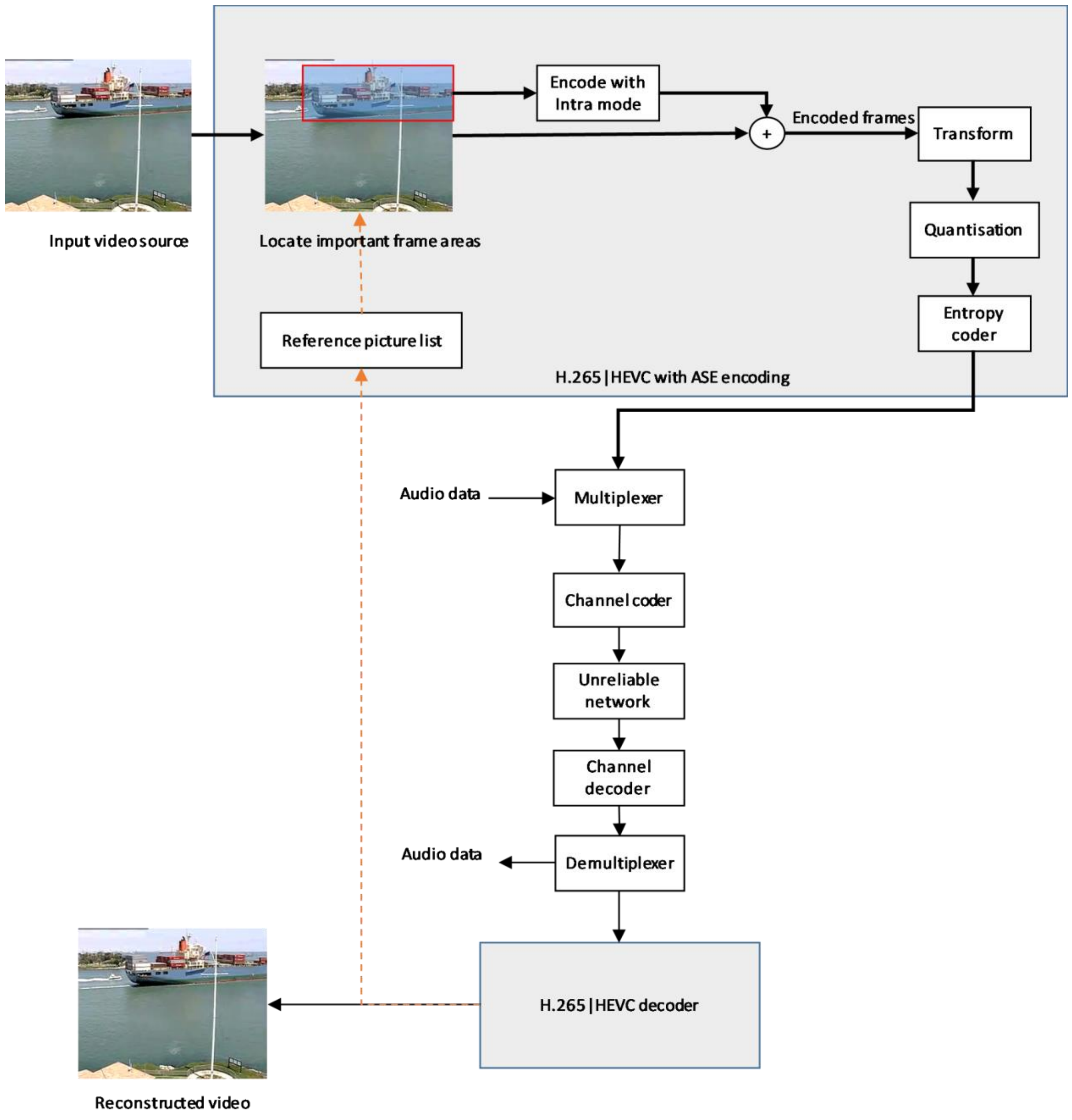


Figure 4.3 Proposed ASE working concept

During the encoding process, the independent slice segment header identifies the address of its exact location at picture level. The identification number refers to count number (ctb) identification in fixed scanning order. The objective of ASE algorithm is to reduce the temporal error propagation by encoding most sensitive and important coding units (CUs) in the selected slices with intra coding mode. In the following subsections, further details are provided on how the activation map is generated. Further, a rate control mechanism for the subdivided frame regions is presented as well.

4.3.1 Area of Interests protection

The proposed algorithm is described as follows: At the first stage, a slice level differencing method is implemented on the current and previous frames. The active area consists of change in content with new texture information. A moving slice with high texture information considered as highly important slice need to be protected against transmission errors. Each slice is mapped with its gray scale representation. The current and previous encoded slices are mapped into projection curves of row (CV_n^x) and column (CV_n^y), respectively, where (n) refers to slice number. Then, each slice in the current frame is projected into 1-Dimensional vector. The one-dimensional gray scale representation (L_n) of selected slice area L(x,y) and slice number (n) with frame number (P) can be calculated in equation (4.1) and equation (4.2):

$$\mathrm{L_n(x)} = \sum_{\mathrm{x}} \mathrm{L(x,y)} \qquad \text{Eq.(4.1)}$$

$$L_n(y) = \sum_{y} L(x,y) \qquad \text{Eq.(4.2)}$$

Where L_n is gray scale values for frame number (P). The average values of $L_n(x)$ and $L_n(y)$ are calculated based on the number of the calculated gray samples rows (r) refers to ($\mathrm{L_{avn}(x)}$) and columns (c) refers to ($\mathrm{L_{avn}(y)}$) , respectively, as defined in equation (4.3) and equation (4.4):

$$\mathrm{L_{avn}(x)} = \frac{\sum_{\mathrm{x}} \mathrm{L_n(x)}}{\mathrm{r}} \qquad \text{Eq.(4.3)}$$

$$L_{avn}(y) = \frac{\sum_{y} L_n(y)}{c} \qquad \text{Eq.(4.4)}$$

Then, the averaged 1-D projected curves are normalised using equation (4.5) and equation (4.6):

$$CV_n^x = L_n(x) - L_{avn}(x) \qquad \text{Eq.(4.5)}$$

$$CV_n^y = L_n(y) - L_{avn}(y) \qquad \text{Eq.(4.6)}$$

Where (CV_n^x) and (CV_n^y) represent the 1-dimensional projected curves for the slice number (n). For better ROI extraction performance with motion activities in background area, a generated intra refresh map is calculated based on Gray-Scale Projection (GPM) method.

The GPM extraction method used in image stabilities applications because of its simplicity in process implementation and at the same time achieve high accuracy of moving objects selectivity [153]. The calculation of 1D-curve vector for the current and previous slices can be demonstrated in Figure 4.4.

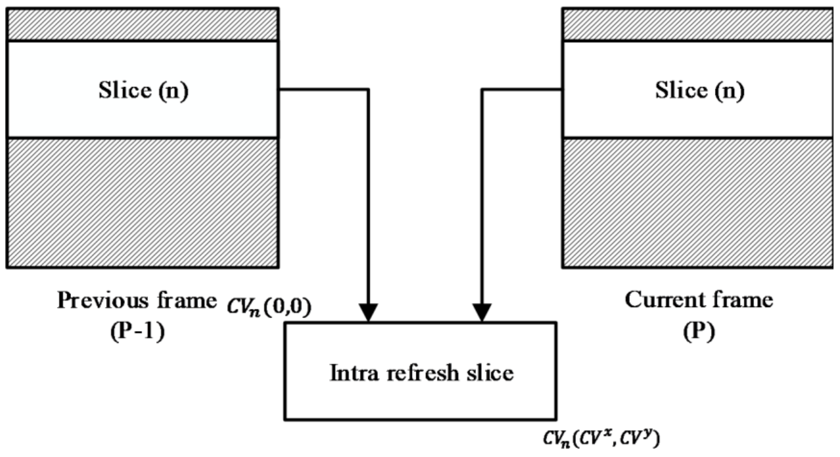


Figure 4.4 Generation refresh map

A cross correlation between current and previous slice is then calculated [153]. The Difference Vector $DV_n(p)$ for each slice is then measured based on equation (4.7):

$$\mathrm{DV_n(p)} = \frac{1}{256}\sum_{(i,j)\in p}^{TS} \left| L_n(i,j) - L_{n-1}(i + CV_n^x\,, j + CV_n^y)\right| \qquad \text{Eq.(4.7)}$$

Where $L_n(i,j)$ and $L_{n-1}(i,j)$ are the luma samples representation for the current (n) and previous slice (n-1), (p) is frame number, and (TS) is the total number of encoded slices per current frame. The maximum cross correlation searching block area of normalised projection curves between the processed slices in current and previous frames can be calculated as in equation (4.8):

$$\text{Searching block area} = \frac{\text{number of }(\mathrm{CU}_{\mathrm{level1}}^{\mathrm{p}}) + \text{number of }(\mathrm{CU}_{\mathrm{level1}}^{\mathrm{p-1}})}{2} \qquad \text{Eq.(4.8)}$$

Where (CU_{level1}^{p}) and (CU_{level1}^{p-1}) are the encoded units with block size (32x32) at coding level 1 for the current and previous frames, respectively. The equation (4.8) is optimised from trials and errors to get best trade-off between the encoding processing delay (an additional computational cost which resulted from the motion estimation calculations) and error resilience performance. The difference vector $DV_n(p)$ representation can be demonstrated in Figure 4.5.

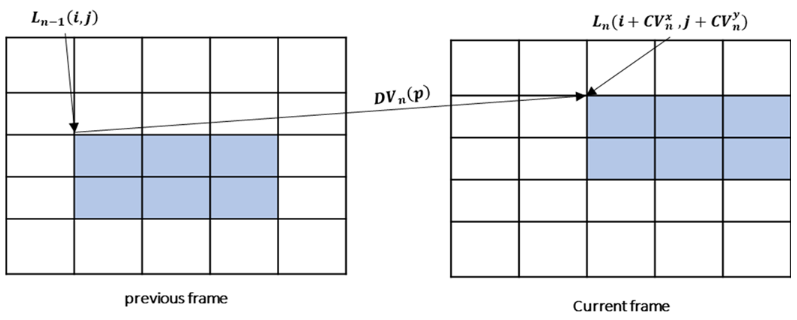


Figure 4.5 Difference vector calculation in ASE algorithm

4.3.2 Subdivision of non-active area

In general, people pay more attention to moving objects in the foreground due to the nature of the human visual system. Additionally, people focus more in the middle area of the display [151]. To get the best trade-off between H.265|HEVC video coding efficiency and perceived visual quality, the identified non-active area from area of interest protection process in previous section is further subdivided into a high textured area which contains a high stationary spatial detail, and a passive (or flat) area which includes a fixed background area with lowest texture details. A simple subdivision example is demonstrated in Figure 4.6. The decoded video quality is reduced in a gradual way from high important areas passing to textured area (transition area) and ending to passive areas, respectively

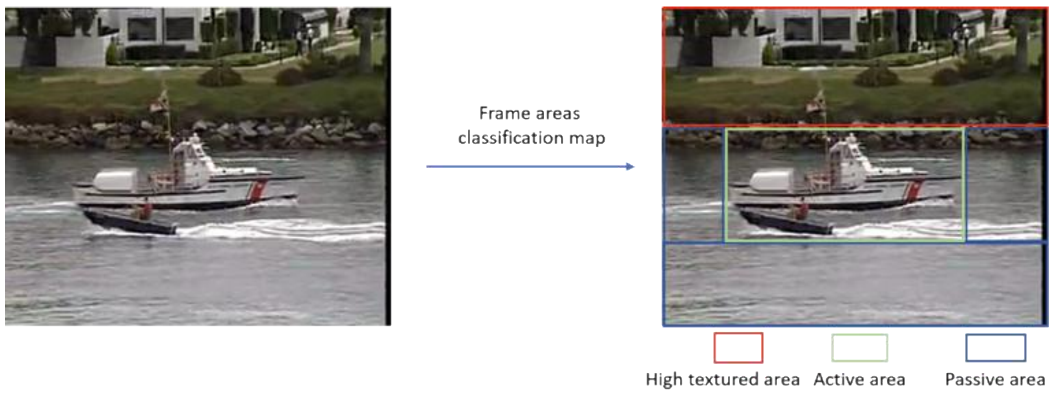


Figure 4.6 Divisions areas in ASE algorithm.

Weighting factor called Adaptive Modified Gray Projection ($AMGP_w$) is defined based on GPM method in [153]. The main aim of AMGP weighting factor is to achieve more accurate selection slice areas. In ASE algorithm, there are three predefined weighting values allocated for partitioning process of Active, high textured, and passive areas. The preselected three values are chosen based on the trial-and-error experiments to be optimised with intra coding refresh of the proposed algorithm. In the experimental work, different weighting factors ranging (0.1 to 0.9) were objectively evaluated to obtain best rate control optimisation with ASE implementation. Furthermore, the optimal AMGP value will be selected depending the slice location at frame level.

The selected weighting factors are achieved during the trial and errors experimental work on a modified HM16.06 +ASE encoder. Due to limited space, one selected video test sequence is depicted in Figure 4.7. The video test sequence is Akiyo and encoded with frame rate (25 fps). The figure shows the effects of various weighting factor (AMGP) values on objective quality in terms of Y-PSNR values.

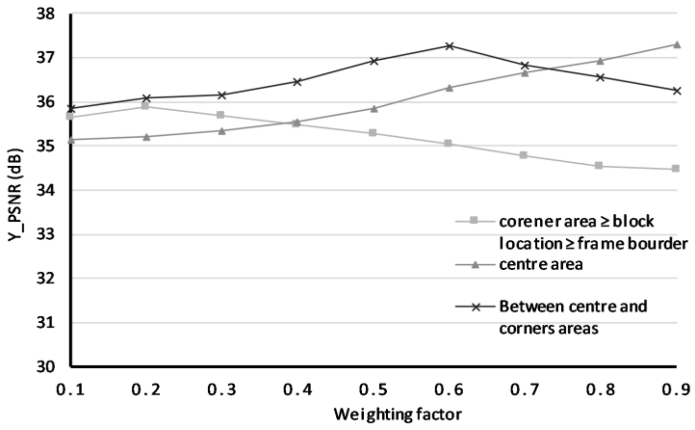


Figure 4.7 Weighting factors three for three ASE areas

To optimise the proposed ASE algorithm with coding standard efficiency, the adaptive AMGP values are allocated at different frame areas. As mentioned before, the frame content complexity is divided into three main areas in the proposed ASE algorithm. The decision of allocating weighting factor depends on the encoded frame area's sizes, which is proportional to the frame dimensions. As the natural human visual system focuses more on the central frame area, this means the probability of the active areas to be encoded will be high (0.9). A lower probability will be in the transition area between the central and corner areas with weighting value of (0.6). The corner area will be allocated the lowest weighting value of (0.2). A weighting factor is assigned for each frame region according to equation (4.9):

$$\mathrm{AMGP_w} = \begin{cases} 0.9, & \text{If the block location} \leq \text{bounding box of centre frame area} \\ 0.2, & \text{If the block location} \geq \text{corners frame area} \\ 0.6, & \text{Otherwise} \end{cases} \qquad \text{Eq.(4.9)}$$

The extraction process of active areas is mainly depending on two calculated values; the Difference Vector $\mathrm{DV_n(p)}$ and the weighting factor of the current frame. The active slice map in the current frame (p) is generated according to equation (4.10):

$$\mathrm{AMGP}_n(p) = \begin{cases} 1, \text{if} & \mathrm{AMGP_w} \times \mathrm{DV_n(p)}/\mathrm{average}[\mathrm{DV_n(p)}] > \mathrm{AMGP_{th}} \\ 0, & \text{Otherwise} \end{cases} \qquad \text{Eq.(4.10)}$$

Where $\mathrm{DV_n(p)}$ is the difference vector and $\mathrm{AMGP_w}$ is the weighting factor for the currently encoded frame using equation (4.9). Finally, the encoding unit in the active map is encoded with intra mode. Hence, the partitioning's factor depends on the current location of the encoded macroblock and the generated reference vector. A weighting value means higher probability of encoding the current block with intra mode.

4.3.3 Non-Active Areas selection

As discussed earlier, a non-active area for each frame is further divided into two regions according to video content features. A further subdivision region contributes to ensuring a perceived visual quality at frame transition level from high quality regions (active areas) to lower quality (passive or high flat region areas). Furthermore, it helps to allocate larger bit budget for active areas and lower bit budget to non-active regions for spending more encoding bits to important frame areas. A Mean Absolute Difference (MAD) calculation between the current and previous frames is used to split non-active areas from high textured areas. In this work, a (0.35) value is selected as a threshold point for generating high textured map as defined in the following equation (4.11).

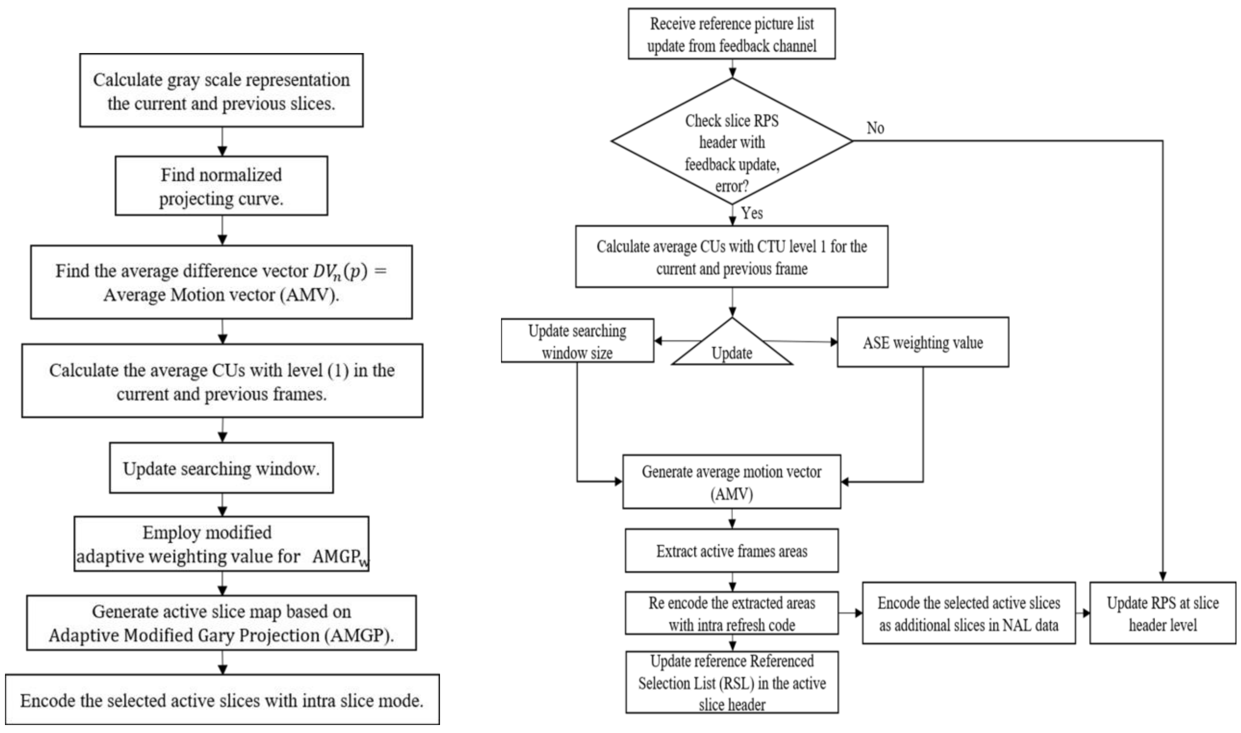
$$\mathrm{H_n(p)} = \begin{cases} 1, & \text{if } \mathrm{H_n(p-1)} < \text{Threshold} \\ 0, & \text{Elsewhere} \end{cases} \qquad \text{Eq.(4.11)}$$

Where $H_n(p-1)$ is macroblock in the previous frame. Then, the remaining map areas are extracted and encoded as lowest complex areas (passive areas).

4.3.4 ASE Algorithm Implementation

At each slice header, a full set of reference picture list is extracted at the decoded picture buffer (DPB). To identify whether the current slice is suitable to be used in the prediction process or not, an RPS data at the slice header is compared with the referenced pictures at DPB.

For error detection and recovery purposes, a feedback channel from the decoder is used to notify the encoder about the occurred errors. The H.265|HEVC coding standard use flag named (used−by−curring−pic−X−flag). The encoder parses the slice header and checks the flag activation [45]. At the decoder side, a slice header RPS is checked against available reference pictures list at the DPB. If there is an update from RPS at the slice header but is not available at DPB, it will consider this slice as not used in the current prediction process. However, if the flag is activated, then the current slice is intended to be used in the prediction process but there is loss or corruption in the reference pictures at decoder side. Figure 4.8 (a) shows flowchart of implementation ASE algorithm without receiving feedback signal.


Calculate gray scale representation the current and previous slices.
Find normalized projecting curve.
Find the average difference vector $DV_n(p)$ = Average Motion vector (AMV).
Calculate the average CUs with level (1) in the current and previous frames.
Update searching window.
Employ modified adaptive weighting value for AMGP$_w$
Generate active slice map based on Adaptive Modified Gary Projection (AMGP).
Encode the selected active slices with intra slice mode.
Receive reference picture list update from feedback channel
Check slice RPS header with feedback update, error?
No
Yes
Calculate average CUs with CTU level 1 for the current and previous frame
Update searching window size
Update
ASE weighting value
Generate average motion vector (AMV)
Extract active frames areas
Re encode the extracted areas with intra refresh code
Encode the selected active slices as additional slices in NAL data
Update RPS at slice header level
Update reference Referenced Selection List (RSL) in the active slice header

(a) ASE algorithm without feedback update channel

(b) ASE algorithm with feedback update channel

Figure 4.8 Adaptive Slice Encoding flowchart

4.3.5 ASE Algorithm with Feedback Update

The proposed error resilience algorithm is further extended to work with the video coding process based on the acknowledgement (ACK) system. The H.265|HEVC coding system requires a feedback channel to locate a damaged slice. A flowchart of the proposed algorithm with feedback channel implementation is shown in Figure 4.8 (b). To obtain more accurate error localisation, the segment header information of the corrupted slices is sent back via a feedback channel to encoder side. This header information contains the most recent update about the reference picture list which includes the address of the most recent erroneous slice.

4.3.6 Rate control adaptation of the proposed algorithm

The challenging task in region of interest extraction implementation is how to keep the computation complexity at minimum level with low delay constraint.
In this algorithm, the encoder is optimised to achieve best trade-off between the number of intra coded slices per frame and the coding efficiency target. A frame is divided into a passive or flat area and high texture or complex area. In HM16 reference software, a lambda rate control is used to optimise the encoding bit rate (bit allocation budget) and video quality (target quantisation parameters) [154]. The encoding bit rate is adjusted based on target bit rate and picture buffer size for each Group Of Pictures (GOP). Then, the encoder allocates required encoding bit budget at LCU level. Depending on the calculated target bit rate, a number of bit per pixel (bpp) is measured depending on the following rate-distortion equation (4.12):

$$\lambda = \alpha . bpp^{\beta} \qquad \text{Eq.(4.12)}$$

Where bpp is bit per pixel, α and β are predefined parameters values. Once λ is calculated, a QP (delta quant) value can be obtained from equation (4.13) and quantisation step size from equation (4.14).

$$\mathrm{P} = 4.2 \ln\lambda + 13.7 \qquad \text{Eq.(4.13)}$$

$$Q_{step} = 1 + 2^{\frac{1}{6}} \qquad \text{Eq.(4.14)}$$

4.4 Experiments Setup

In this section, hardware and software tools and encoding video configurations are described. The experimental work includes testing the ASE algorithm with different error prone conditions, encoding bi rates, in addition to the computational complexity of the modified video reference HM16 software. Pre-selected standard video test sequences are chosen in the experiments. The video sequences characteristics which used in the evaluation work are presented in Table 4.1.

Table 4.1 Characteristics of the test video sequence

No.	Sequence name	Resolution	number of frames	video class
1.	Hall Monitor	QCIF, CIF	300	A
2.	Mobile	QCIF, CIF	300	A
3.	Bus	CIF	150	B
3.	Container	QCIF, CIF	300	A
4.	Grandma	QCIF	870	A
5.	Akiyo	QCIF, CIF	300	A
6.	Miss America	QCIF	150	A
7.	Bridge-close(far distance)	QCIF, CIF	2001	A
8.	Mother and daughter	QCIF, CIF	300	A
9.	News	QCIF, CIF	300	
10.	Bridge (far)	QCIF, CIF	2101	A
11.	Bridge-close (near distance)	QCIF, CIF	2001	A
12.	Coastguard	QCIF, CIF	300	B
13.	Claire	QCIF	494	A
14.	Carphone	QCIF	382	B
15.	Highway	QCIF, CIF	2000	A
16.	Salesman	QCIF	449	B
17.	Silent	QCIF, CIF	300	B
18.	Suzie	QCIF	150	B

The test sequences are in raw format (YUV) with colour space format 4:2:0. They are classified into two groups according to their video textured detail and motion activity speed. Class A: represent video sequences with low texture details and slow-motion activity. Class B: represent video sequences with high texture details and high-motion activity.

4.4.1 Hardware and Software testbed setup

Three PCs are used in experimental work. Two PCs implemented as PC video server (Dell T410 Power Edge server, CPU: Quad-core 2.35GHz, RAM: 16GB, operating system: Microsoft Windows 10). and video receiver (Dell XPS, CPU: Intel Core i5-7200

@2.5GHz, RAM: 8GB, operating system: Microsoft Windows 10). Open source network simulator version 3 NS3 is installed on separated PC (HP Compac 8200, CPU: Core i5-2500s, RAM: 8GB, Operating system: Ubuntu server 15.04).

4.4.2 Network testbed setup

The NS3 is chosen to be installed on Linux operating system. The Long-Term Evolution (LTE) module is embedded within the NS3 environment. To embed NS3 simulator software with LTE network module and integrate them to a real physical Ethernet interface, a Hardware In Loop (HIL) platform in [155] is employed. Each node in NS3 / LTE network is connected using Carrier-Sense Multiple Access (CSMA). LTE gateway SGW/PGW uses a point to point internet connection. Figure 4.9 shows an overall network interface connection for LTE implementation to stream encoded H.265|HEVC video bitstream with MP4 container format to multiple clients.

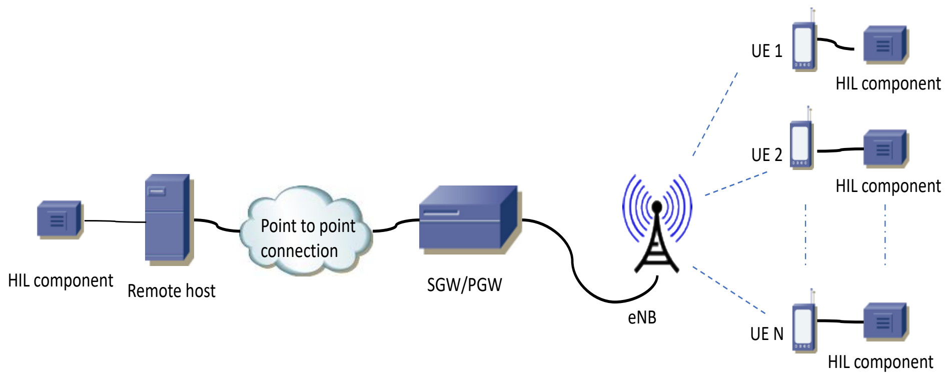


Figure 4.9 Long-Term Evolution (LTE) network testbed

Table 4.2.reports LTE network configuration settings used in the experimental work. The open-source cross-platform multimedia player (VLC) is used to stream the video test sequences at the sender side. At the receiver, the VLC player used to visualise the perceived visual quality. The end to end video evaluation platform is demonstrated in Figure 4.10.

Table 4.2 LTE network parameters

Network parameter	**Value**
Physical Layer Profile	Orthogonal frequency-division multiple access (OFDMA), Full-frequency-division duplexing (FDD) Downlink: 5 MHz Uplink: 1.4 MHz
Path loss model	Friis free space model
User equipment (UE) Noise Figure	8dB
Antenna type	Isotropic antenna
Transmission mode	Single input single output
Base station (eNB) operating power / noise figure	50 dbm/ 4dB

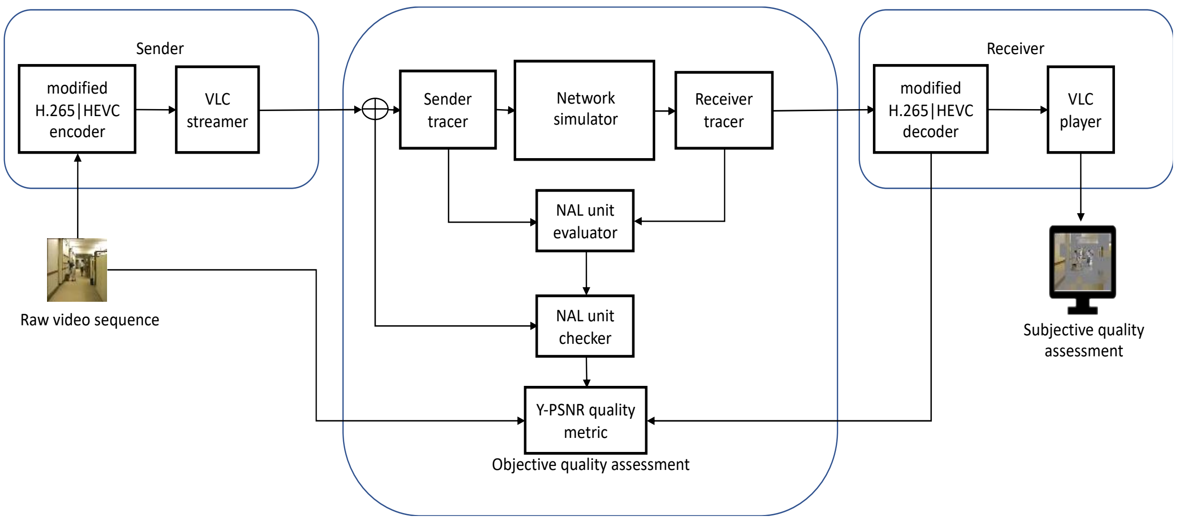


Figure 4.10 Real-time video streaming evaluation framework

4.4.3 Performance evaluation setup

This section discusses the quality evaluation process and encoding configuration setting for evaluation ASE algorithm in error-free and error-prone conditions in addition to computational complexity study. In these experiments, various packet loss rates are injected into encoded video bitstreams. Packet loss rates were generated using evaluation platform described in (Figure 3.20) in Section (3.11.1). To achieve fair evaluation, each selected video sequence is repeated 30 times with different error generator seeds. Then, average Y_PSNR values are recorded. All video sequences were randomly injected with PLRs ranging (2-18) % using packet loss rate generator software. All tests are carried out with GOP size (8) and low delay P configuration, other encoding configuration settings are reported in Table 4.3.

Table 4.3 Encoding settings for modified HM 16.06+ ASE and HM 16.06

Encoding parameter	**Configuration**
Profile name	Main profile
Encoding GOP size	8 frames
Rate control	Disabled
Filtering	Enabled
Search range	64 blocks
bit depth	8 bits
Intra mode	1st frame in each GOP
(max/min) transform unit size	32/4
Largest coding unit (LCU) size	(64x64) pixels
Frames to be encoded (-f)	Various frames, (90-2001) frames
Input file	Video test sequences [apendix A]
SourceWidth (-wdt) x SourceHeight (-hgt)	Test sequences characteristics are reported in Appendix A

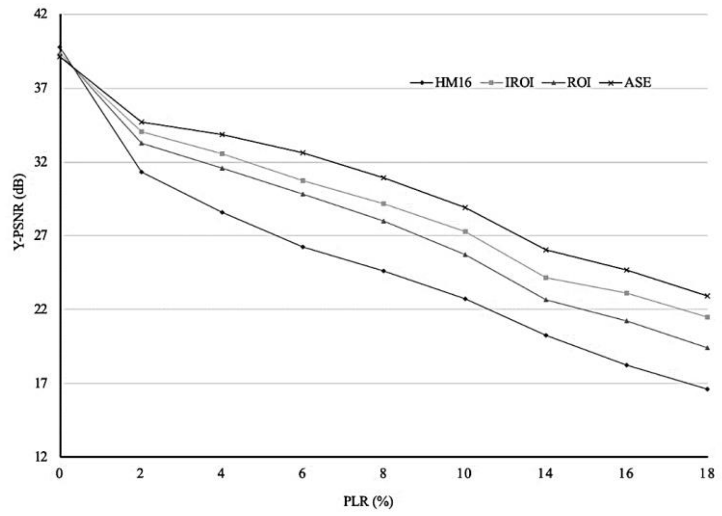
The process involves comparing the performance of ASE algorithm with default reference software (HM16.06), ROI algorithm, and IROI algorithm. The default reference software (HM16.06) produced by Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T and ISO/IET organisations. ROI error resilience algorithm is Region-based error resilient scheme for HEVC video transmission ROI method [145]. IROI error resilience algorithm is Improved Region of Interest (IROI) based on rate control adaptation algorithm [146].

4.5 Results and Discussions

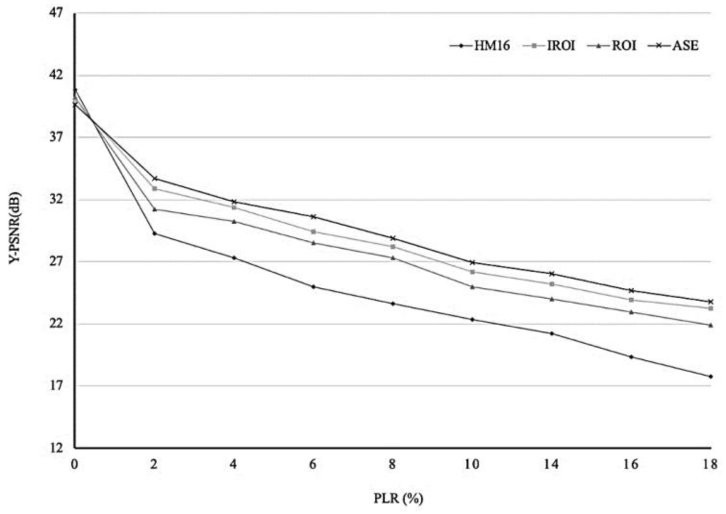
This section presents quality evaluation comparisons for the proposed ASE algorithm. The objective quality comparisons begin with testing the proposed ASE algorithm and another related state of art algorithms which are a region of interest (ROI) in [145] and improved region of interest (IROI) in [146], in addition to default reference encoder (HM16) for H.265|HEVC coding standard .

4.5.1 Objective Results in Error-Free and Error-Prone Conditions

Three test video test sequences are chosen to evaluate the proposed ASE algorithm, namely Coastguard, Hall Monitor, and Mobile sequences. Coastguard sequence is classified as class B represent with high textured details and motion activities, and the other two sequences are classified as class A (fewer texture details and slower motion object) [Appendix A]. The selected video tests sequences are encoded with encoding settings reported in (section 4.4.3). For error prone evaluation, all video sequences were randomly injected with packet loss rate PLRs ranging (2-18) %. using packet loss rate generator software. Figure 4.11 shows the achieved objective quality results for three video test sequences in error free and error prone conditions.

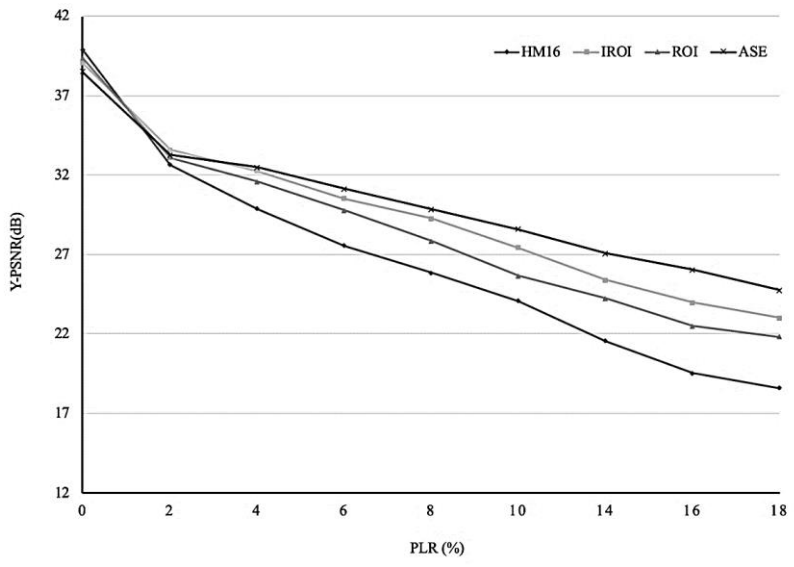


(a) Coastguard



(b) Hall monitor

Figure 4.11 Proposed ASE algorithm evaluation with various Packet Loss Rates



(c) Mobile

Figure 4.11 Proposed ASE algorithm evaluation with various Packet Loss Rates

The averaged Y-PSNR values with different packet loss rates for the three video sequences are reported in Table 4.4, Table 4.5, and Table 4.6.

Table 4.4 Video quality vs PLRs for Coastguard sequence (CIF resolution)

PLR (%)	HM16.06	IROI	ROI	ASE	ASE vs HM16	ASE vs ROI	ASE vs IROI
0	39.802	39.226	39.423	39.1	-0.702	-0.323	-0.126
2	31.337	34.052	33.251	34.712	3.375	1.461	0.66
4	28.575	32.566	31.557	33.846	5.271	2.289	1.28
6	26.255	30.732	29.834	32.61	6.355	2.776	1.878
8	24.593	29.15	28.019	30.916	6.323	2.897	1.766
10	22.745	27.268	25.708	28.913	6.168	3.205	1.645
14	20.255	24.151	22.675	26.039	5.784	3.364	1.888
16	18.241	23.108	21.268	24.664	6.423	3.396	1.556
18	16.605	21.499	19.44	22.942	6.337	3.502	1.443
Average	**23.575**	**27.815**	**26.469**	**29.330**	**5.754**	**2.861**	**1.514**

Table 4.5 Video quality vs PLRs for Hall sequence (CIF resolution)

PLR (%)	HM16.06	IROI	ROI	ASE	ASE vs HM16	ASE vs ROI	ASE vs IROI
0	40.848	39.926	40.275	39.652	-1.196	-0.623	-0.274
2	29.293	32.853	31.187	33.712	4.419	2.525	0.859
4	27.331	31.385	30.27	31.846	4.515	1.576	0.461
6	24.961	29.433	28.501	30.61	5.649	2.109	1.177

8	23.651	28.18	27.299	28.916	5.265	1.617	0.736
10	22.357	26.201	24.994	26.913	4.556	1.919	0.712
14	21.241	25.226	24.01	26.039	4.798	2.029	0.813
16	19.378	23.894	22.949	24.664	5.286	1.715	0.77
18	17.738	23.22	21.922	23.742	6.004	1.82	0.522
Average	**23.243**	**27.549**	**26.391**	**28.305**	**5.061**	**1.913**	**0.756**

Table 4.6 Video quality vs PLRs for Mobile sequence (CIF resolution)

PLR (%)	HM16.06	IROI	ROI	ASE	ASE vs HM16	ASE vs ROI	ASE vs IROI
0	39.892	39.056	39.372	38.501	-1.391	-0.871	-0.555
2	32.644	33.597	33.085	33.269	0.625	0.184	-0.328
4	29.882	32.253	31.599	32.477	2.595	0.878	0.224
6	27.562	30.533	29.764	31.123	3.561	1.359	0.59
8	25.852	29.267	27.848	29.85	3.998	2.002	0.583
10	24.089	27.418	25.666	28.591	4.502	2.925	1.173
14	21.563	25.406	24.249	27.058	5.495	2.809	1.652
16	19.548	23.992	22.506	26.029	6.481	3.523	2.037
18	18.612	23.01	21.81	24.743	6.131	2.933	1.733
Average	**24.969**	**28.184**	**27.065**	**29.142**	**4.173**	**2.076**	**0.958**

For Coastguard sequence, the objective results of ASE algorithm in error-prone conditions outperforms the default HM16 software, ROI, and IROI algorithms by (5.754 dB), (2.861 dB), and (1.514dB), respectively.

In the same error conditions, for video sequences class A, the improvements of Y-PSNR for Hall monitor video sequence are (5.061 dB), (1.913 dB), and (0.756 dB) for default HM16, ROI, and IROI algorithms, respectively. For Mobile video sequence, the (Y-PSNR) improvements under the same error conditions are (4.173 dB), (2.076 dB), and (0.958 dB) for default HM16, ROI, and IROI algorithms, respectively.

Table 4.7 shows the Y-PSNR results for all the three video sequences (Coastguard, Hall, and Mobile sequences) in CIF resolution. The average (Y-PSNR) of ASE algorithm with different PLRs has improved by (4.521db), (2.283db), and (1.076db) compared to HM16 for H.265|HEVC coding standard, ROI, and IROI algorithms, respectively.

In error-free conditions (PLR=0%), Y-PSNR of the ASE algorithm is reduced by (-1.096 dB; HM16), (-0.605dB; ROI), and (-0.318dB; IROI).

Table 4.7 ASE performance comparisons in terms of Y-PSNR (dB) vs PLRs

Sequence	**Error-free condition Y-PSNR values (dB)**			**Error-prone Condition Y-PSNR values (dB)**		
	ASE vs HM16	ASE vs ROI	ASE vs IROI	ASE vs HM16	ASE vs ROI	ASE vs IROI
Coastguard	-0.702	-0.323	-0.126	5.754	2.861	1.514
Hall	-1.196	-0.623	-0.274	3.636	1.913	0.756
Mobile	-1.391	-0.871	-0.555	4.173	2.076	0.958
Average	-1.096	-0.605	-0.318	4.521	2.283	1.076

The proposed ASE algorithm is further evaluated using 18 video sequences with different video motion complexity and texture information. The selected test sequences are in CIF and QCIF resolutions with different frame numbers. The characteristics of video test sequences are listed in [Appendix A]. A packet loss rate is generated randomly at PLR=4%. The encoding setting for experimental work comparisons is reported in section 4.4.3. Table 4.8 and Figure 4.12 show the objective results for 18 video test sequences.

Table 4.8 ASE objective evaluation results of Eighteen video tests injecting with PLR=4%

Sequence number	Sequence name	HM16.06	ROI	IROI	ASE	Video Class
1	Hall monitor	29.613	31.554	33.542	35.294	A
2	mobile	28.943	34.261	35.021	36.054	A
3	container	32.054	34.264	35.492	36.558	A
4	grandma	33.264	35.304	36.164	36.949	A
5	akiyo	32.062	33.942	34.265	35.724	A
6	miss-america	30.406	34.584	35.452	36.144	A
7	bridge-close (far distance)	29.543	34.261	35.021	35.854	A
8	Mother-daughter	29.124	33.825	35.194	36.142	A
9	News	32.654	34.804	35.797	36.015	A
10	Bridge-far	32.164	34.262	35.942	36.745	A
11	Bridge-close(near distance)	31.264	33.859	34.664	35.215	B
12	Coastguard	31.123	33.562	35.028	36.784	B
13	Claire	27.622	32.627	33.209	34.624	B
14	Carphone	25.154	29.594	30.943	32.549	B
15	highway	28.663	32.28	33.705	34.45	B
16	Salesman	30.264	33.085	34.482	35.552	B
17	silent	28.294	33.235	34.723	35.264	B
18	Suzie	27.142	32.162	34.285	35.416	B

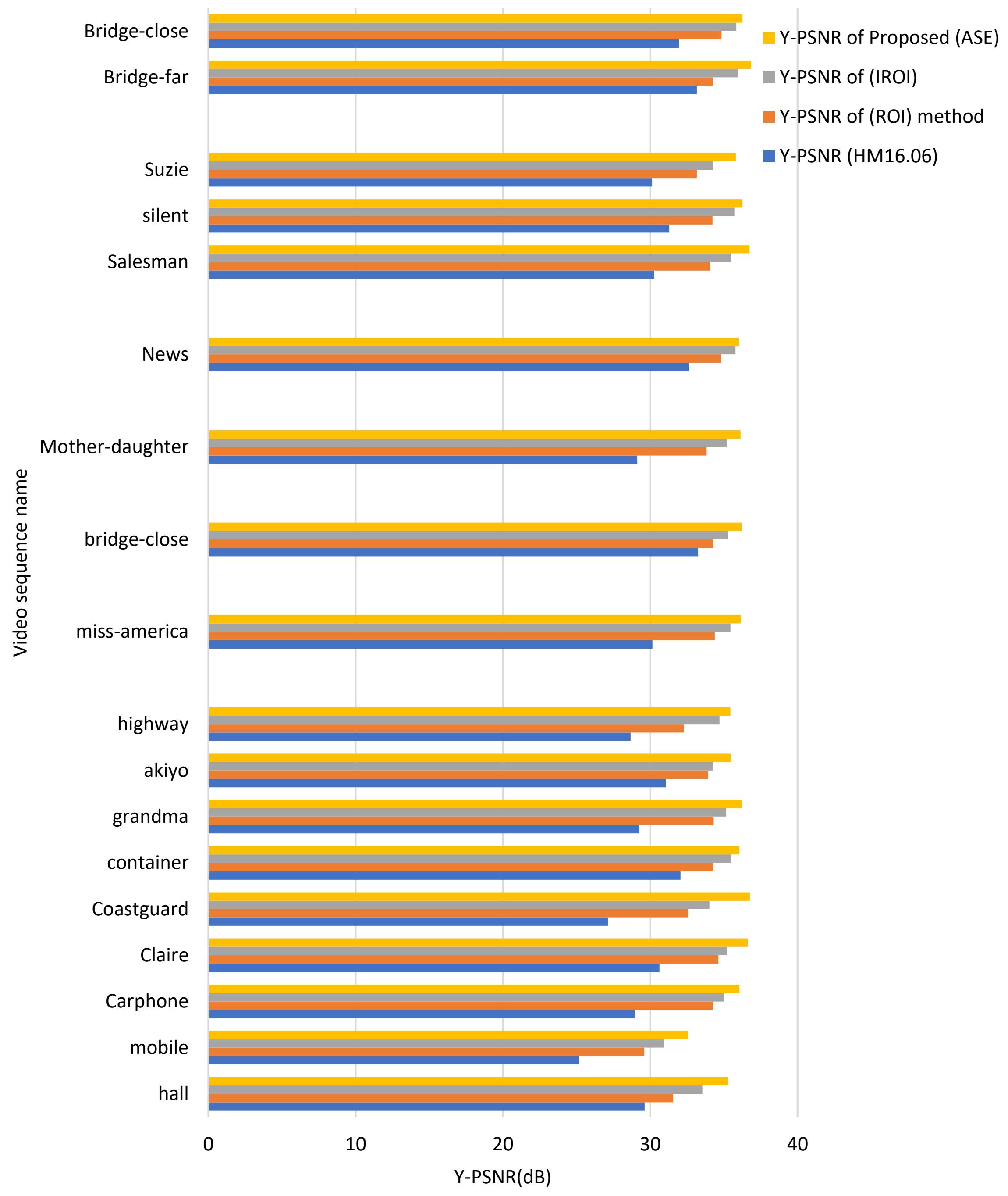


Figure 4.12 Objective Evaluation for ASE algorithm with 18 video test sequences in error-prone condition

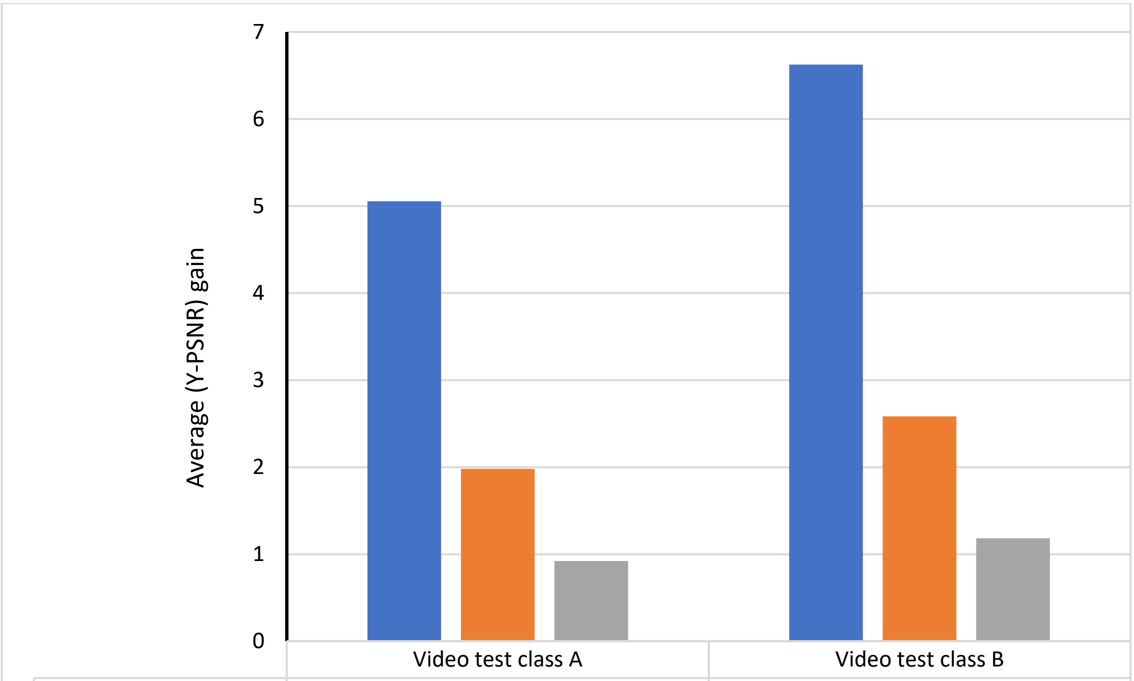
Table 4.9 and Table 4.10 present the average Y-PSNR gain achieved by the ASE algorithm using the same encoding and packet loss simulation described in (section 4.4.3). The injected errors are generated randomly with PLR=4%. Table 4.9 reports the obtained quality results in terms of Y-PSNR for video sequences that have less motion activity and texture details (Class A). Table 4.10 shows the experimental results obtained for evaluating the ASE algorithm applied to high complexity test video (video test sequences class B) using the same error condition (PLR=4%) used with class A video sequences. Figure 4.13.

Table 4.9 Objective quality results for class A video test sequences [Appendix A]

No.	**Sequence name**	**ASE vs HM16**	**ASE vs ROI**	**ASE vs IROI**
1	hall	5.681	3.74	1.752
2	mobile	7.111	1.793	1.033
3	container	4.31	2.1	0.872
4	grandma	3.685	1.645	0.785
5	akiyo	3.4	1.52	1.197
6	miss-america	6	1.754	0.692
7	bridge-close (far distance)	6.311	1.593	0.833
8	Mother-daughter	7.018	2.317	0.948
9	News	3.361	1.211	0.218
10	Bridge-far	4.581	2.483	0.803
11	Bridge-close (near distance)	3.951	1.356	0.551
Average		5.037	1.955	0.880

Table 4.10 Objective quality results for class B video test sequences [Appendix A]

Sequence number	**Sequence name**	**HM16**	**ROI**	**IROI**	**ASE**	**ASE vs HM16**	**ASE vs ROI**	**ASE vs IROI**
1	hall	29.613	31.554	33.542	35.294	5.681	3.74	1.752
2	mobile	28.943	34.261	35.021	36.054	7.111	1.793	1.033
3	container	32.054	34.264	35.492	36.364	4.31	2.1	0.872
4	grandma	33.264	35.304	36.164	36.949	3.685	1.645	0.785
5	akiyo	32.062	33.942	34.265	35.462	3.4	1.52	1.197
6	miss-america	30.144	34.39	35.452	36.144	6	1.754	0.692
7	bridge-close (far distance)	29.543	34.261	35.021	35.854	6.311	1.593	0.833
Average		30.984	34.066	35.141	36.021	5.037	1.955	0.880



	Video test class A	Video test class B
🟦 ASE vs HM16 (Y-PSNR)	5.054	6.625
🟧 ASE vs ROI (Y-PSNR)	1.979	2.584
⬜ ASE vs IROI (Y-PSNR)	0.921	1.181

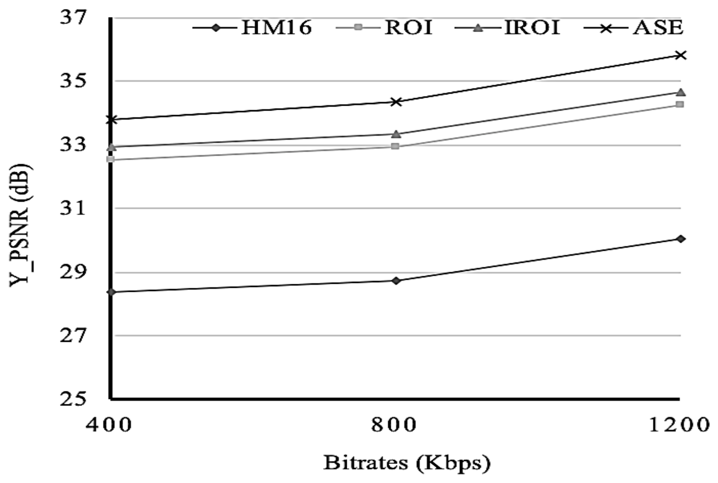
Figure 4.13 Performance of ASE algorithm with different encoded video complexity

It is observed from Table 4.9, Table 4.10, and Figure 4.13, that for video test sequences class A, the ASE algorithm performance is higher than ROI, IROI and HM16 with (Y-PSNR) gain of (0.921 dB, 1.979 dB, and 5.054 dB), respectively. For video test sequences class-B, the improvements are significantly higher compared to video test sequences class-A with achieved (Y-PSNR) gains of (2.584 dB, 1.181 dB, and 6.625 dB), respectively.

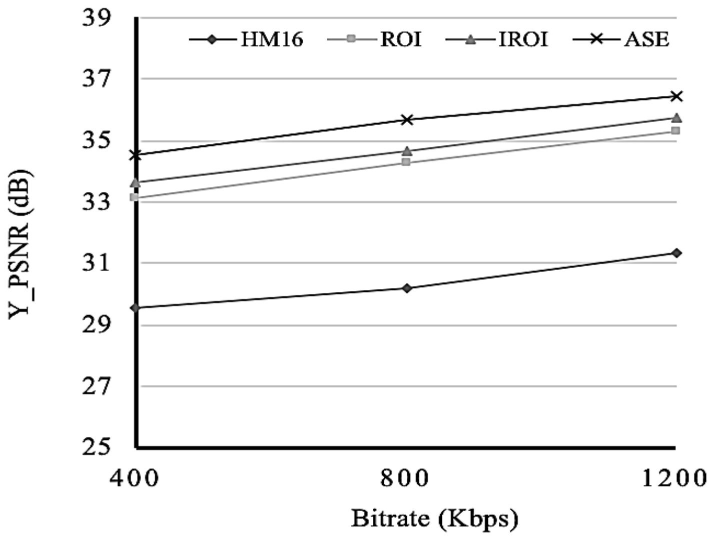
To sum up, from the obtained test results, it can be noticed that the performance of the proposed ASE algorithm is less effective in error free conditions. This is due to the fact that the complex processing part the proposed algorithm depends on the dividing each frame into three different areas. The divided areas are depending on their contents complexity which they are necessary in error resilience performance for protecting important areas and achieving best balance between coding deficiency and error resilience performance. The accuracy of the encoding bit rate allocation is lower than the default rate control used with the other compared algorithms.

4.5.2 Rate distortion performance

Further evaluation tests are performed to measure the effectiveness of the proposed ASE algorithm under erroneous condition with injecting BER (1×10^{-5}) in various encoding bitrates (in several kbps). The input video test sequences are in CIF resolution. Three video test sequences are selected (Coastguard sequence; class B, Hall sequence; class A, Mobile; class A). All sequences are encoded with 300 frames using (HM16, MHM+ASE algorithm, MHM+ROI algorithm, and MHM+IROI algorithm). The achieved objective results are shown in Figure 4.14.

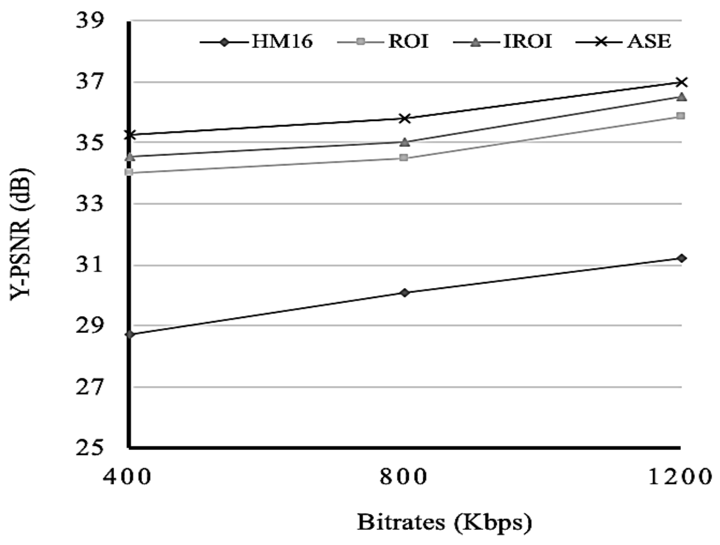


(a) Coastguard



(b) Hall monitor

Figure 4.14 Rate distortion performance for ASE evaluation with BER (10^(-5))



(c) Mobile

Figure 4.14 Rate distortion performance for ASE evaluation with BER (10^{-5})

Looking at Figure 4.14, it can be noticed that the performance of ASE algorithm increases proportionally with increasing encoding bit rate consistently. Furthermore, the video quality is severely degraded in all video sequences even when increasing encoding bit rates. The figure also indicates that the ASE algorithm outperforms the ROI and IROI algorithms. For lower complexity motion videos (Hall monitor and mobile class A sequences) (figure (b) and (c)), IROI algorithm produces better results with encoding slow-motion video in error-free conditions.

4.5.3 Frame by Frame Quality Assessment

The proposed ASE algorithm is further evaluated subjectively using frame by frame quality assessment. In the quality assessment process, the unprocessed frames is compared side by side with the processed frames as demonstrated in Figure 4.15. Coastguard video test sequence with CIF resolution is selected and encoded with (30 fps) in quality assessment test. Figure 4.16 shows randomly selected three frames in raw formats (before compression).

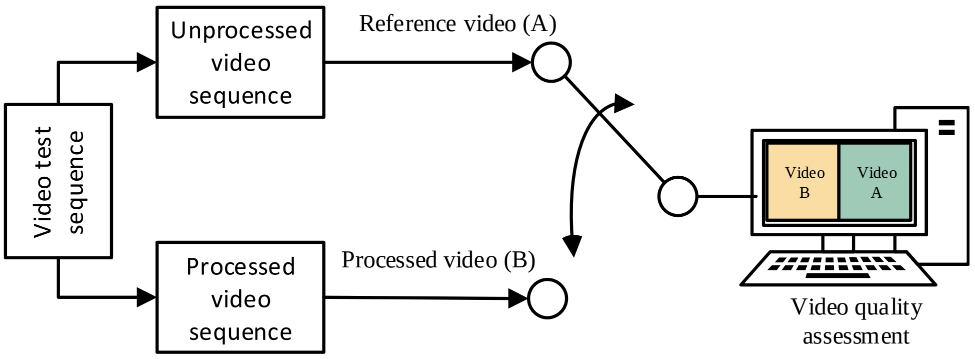


Figure 4.15 Frame by frame video quality assessment

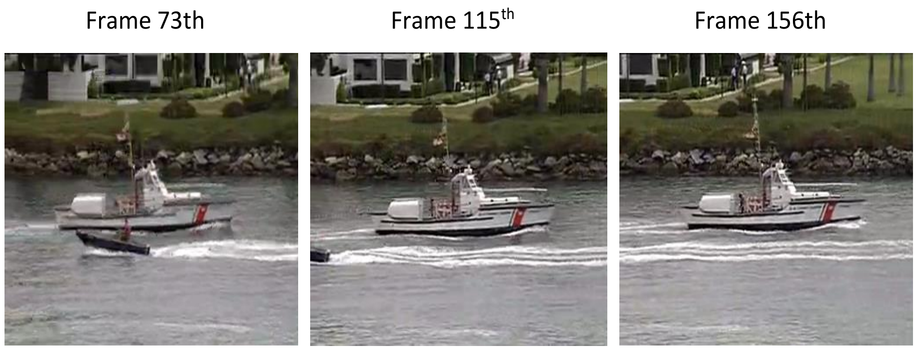


Figure 4.16 Selected error-frames for Coastguard video sequence

PLR of 2% is injected into coded bitstream (Coastguard sequence). Figure 4.17 shows frame by frame visual quality assessment of three randomly selected frames for HM 16.06, ROI, IROI, and ASE algorithms in error prone condition at PLR of 2%.

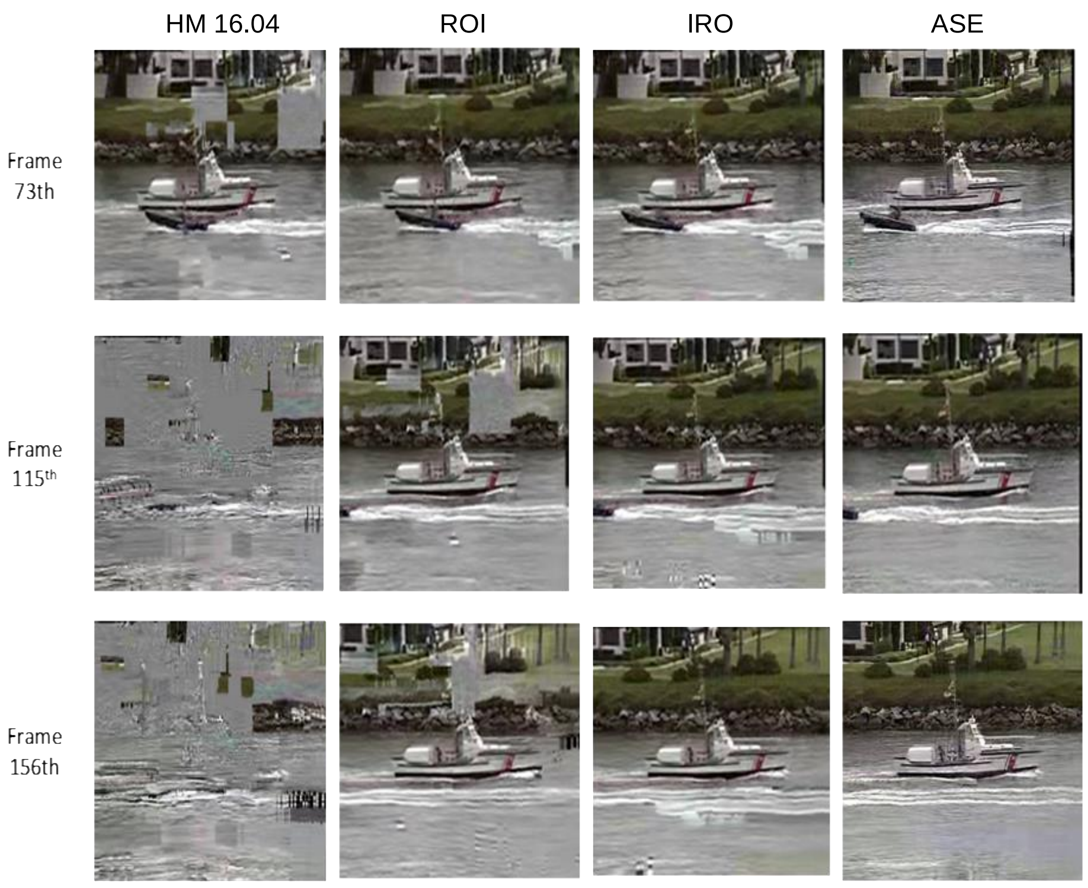


Figure 4.17 Subjective frame by frame quality assessment for ASE, ROI, and IROI algorithms

It can be revealed that two of decoded frames (115th and 156th) with reference software HM 16.06 are suffer from high visible artefacts which results in fail to achieve accepted visual quality. The reconstructed frames with ROI and IRO algorithms are improved significantly with slight small artefacts on some frame's areas. Using same erroneous conditions with ASE algorithm, it is noticed that the important area which is the ship in this case is preserved higher visual quality in addition to higher smoothness in water area compared with ROI and IROI algorithms. From the achieved quality assessment results, it is clearly that the proposed ASE algorithm select and protect the important areas more accurately with keeping higher quality detail by spending higher bit budget on these areas. Additionally, the ASE algorithm successfully reconstructed erroneous frames without visible blackness effects.

4.6 Computation complexity

The aim of this work is to determine the impact of encoding/decoding computational complexity at video encoder and decoder sides for ASE algorithm. It is worth noting that the reference software (HM) is mainly used for developing H.265|HEVC video coding algorithms without taking into accounts support real time video encoding applications. Although the HM reference software suffers from slow speed of execution of encoding and decoding process. However, some attempts have been made during various HM versions generations. These attempts aim to reduce the computational complexity in terms of required processing time, hardware, and software implementations.

In the experimental work, we measured the encoding/decoding processing time for the proposed algorithm, then the recorded processing time are compared with standard default reference software HM16. In this work, we evaluated ASE algorithm using low delay-B configuration mode and input test sequences are encoded with QP at (32). The video test sequences are encoded using same encoding setting reported in Table 4.3 Section (4.4.3). The video sequences characteristics used in complexity evaluation work are reported in Table 4.11.

The computational processing tests are carried out on PC (Dell T410 Power Edge) with CPU: Quad-core 2.35GHz and RAM: 16GB. The operating system (Microsoft Windows 10) is installed on the PC hardware.

Table 4.11 Characteristics of the test video sequence

No.	Sequence name	Resolution	number of frames	video class
1.	Hall Monitor	QCIF, CIF	300	A
2.	Mobile	QCIF, CIF	300	A
3.	Bus	CIF	150	B
3.	Container	QCIF, CIF	300	A
4.	Grandma	QCIF	870	A
5.	Akiyo	QCIF, CIF	300	A
6.	Miss America	QCIF	150	A
7.	Bridge-close (far distance)	QCIF, CIF	2001	A
8.	Mother and daughter	QCIF, CIF	300	A
9.	News	QCIF, CIF	300	
10.	Bridge (far)	QCIF, CIF	2101	A
11.	Bridge-close (near distance)	QCIF, CIF	2001	A
12.	Coastguard	QCIF, CIF	300	B
13.	Claire	QCIF	494	A

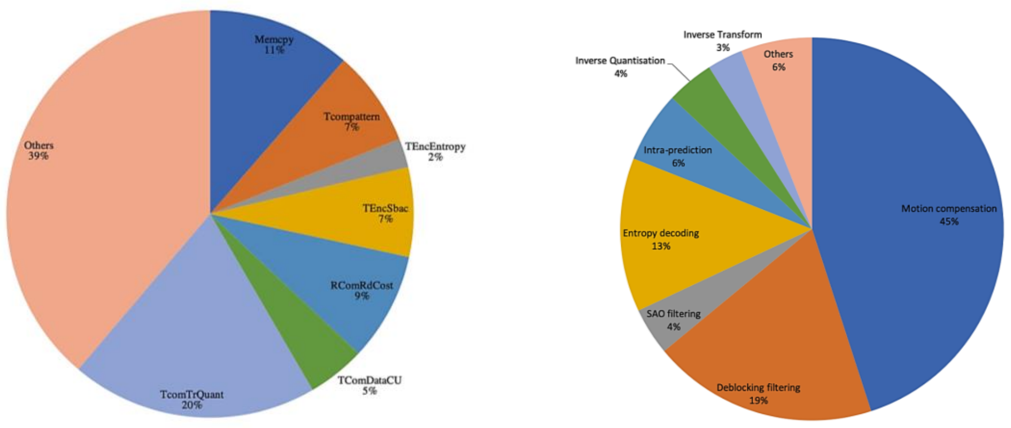
14.	Carphone	QCIF	382	B
15.	Highway	QCIF, CIF	2000	A
16.	Salesman	QCIF	449	B
17.	Silent	QCIF, CIF	300	B
18.	Suzie	QCIF	150	B

For complexity evaluation, ASE algorithm and HM16.06 Reference software are compared in terms of encoding and decoding execution processing time for each video test sequence, the achieved results are reported in Table 4.12.

Table 4.12 Computational complexity of ASE algorithm compared to HM16.06 Reference software

		Number of frames	**Encoding time (seconds)**		**Decoding time (seconds)**	
			HM16	**HM16+ASE**	**HM16**	**HM16+ASE**
1	Hall monitor	300	575	664	6.5	7.4
2	Mobile	300	491	646	6.8	7.8
3	Container	150	275	298	4.3	5.2
4	Grandma	300	548	686	5.8	6.6
5	Akiyo	870	956	1232	9.3	10.2
6	Miss-America	300	507	653	5.8	6.3
7	Bridge-close (far distance)	150	249	367	4.2	5.4
8	Mother-daughter	2001	1283	1492	16.3	17.4
9	News	300	501	647	6.2	7.1
10	Bridge-far	300	498	622	5.7	6.4
11	Bridge-close (near distance)	2101	1355	1498	15.2	16.4
12	Coastguard	2001	1294	1435	14.3	15.1
13	Claire	300	485	588	5.8	6.6
14	Carphone	494	684	809	7.3	8
15	Highway	382	538	653	6.6	7.6
16	Salesman	2000	1385	1509	16.5	17.5
17	Silent	449	649	844	8.4	9.2
18	Suzie	300	473	586	6.5	7.3
Average (sec)			708	846	8.4	9.3

Figure 4.18 presents the spent processing excution time of the selected eighteen sequences at (a) video encoder side and (b) video decoder side. In the figure, the overall averaged execution times are reported in percentage for ASE algorithm compared to HM 16.06 reference software.



(a) Average Encoding time distribution (b) Average Decoding time distribution

Figure 4.18 Computation complexity of ASE algorithm compared to reference software HM 16.06

Figure 4.19 shows the average increase in percentage of encoding and decoding processing time with ASE algorithm compared to HM16 reference software. It is noticed that when using ASE algorithm, the encoder consumes more time than at the decoder side. The additional computation time in modified HM16 encoder has arisen from rate control adaptation for encoding different areas with different encoding bit rates. Furthermore, it comes from allocating different quantisation parameters at LCU levels. Additionally, it comes also from picture partitioning process of splitting different sub regions areas. As mentioned before, this process includes differencing method at frame level which add additional computation processing at encoding stage.

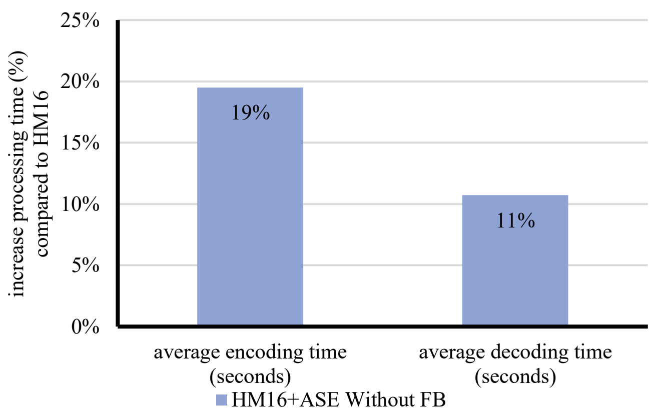


Figure 4.19 Average increase encoding/decoding processing compared to HM16

For additional computations at the decoder, the high amount of processing time spent on parsing the redundant slices which leads to an increase in reference sample generation process at decoded picture buffer (DPB). Furthermore, an additional part of the decoding process is spent on checking operation for slice boundaries, in addition to the reference sample generation of intra slices prediction process. Moreover, the increase time in both encoding and decoding process compared with HM software has risen from adding a set of C++ classes for error resilience implementation.

4.6.1 Network Congestion and Video Processing Delay

The proposed algorithm is further evaluated by streaming the encoded video bitsream into LTE network with various network loads. The number of the end users is ranging from 10 to 30 users per one base station. The experimental work is conducted using LTE network evaluation platform and LTE configuration settings described in the network testbed setup Section (4.4.2). A frame copy concealment is used at the decoder to avoid failure in the decoding process. Two main objectives are targeted in this experimental work. The first one is to show the effect of various client's numbers on objective quality in LTE network using shared bandwidth. The second one is the video start-up time at the decoder, as it is a critical factor for meeting the user's quality of experience requirements [156]. The start-up time is defined as the required time that decoder buffer needs to be processed before displaying the decoded pictures.

The authors in reference [156] recommended that the start-up video delay in video streaming applications should be not more than 2 seconds. In this experimental work, we chose 1000 ms and 500 ms use cases. Our algorithm is integrated with video evaluation platform described in Section (3.11.1). Based on the reported encoding configuration settings and network testbed settings. Eighteen video sequences were selected and the average for each tested number of clients group (10,20,30 end users) are recorded. Each test is repeated 10 times to get more reliable verifications. Average Y-PSNR results for the preselected test sequences are recorded for evaluating network load behaviour on objective visual quality. A network load is categorised into three levels; light (10 users), medium (20 users), and heavy load (30 users). Figure 4.20 and Figure 4.21 show achieved results in terms of Y-PSNR at the decoder from testing three user groups using shared transmission bandwidth on the objective visual quality.

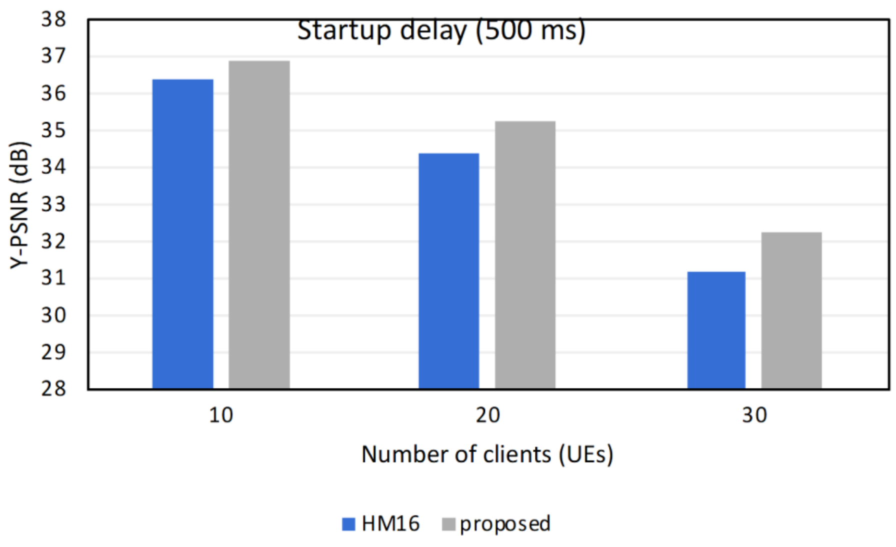


Figure 4.20 ASE video quality performance with start-up delay (500ms)

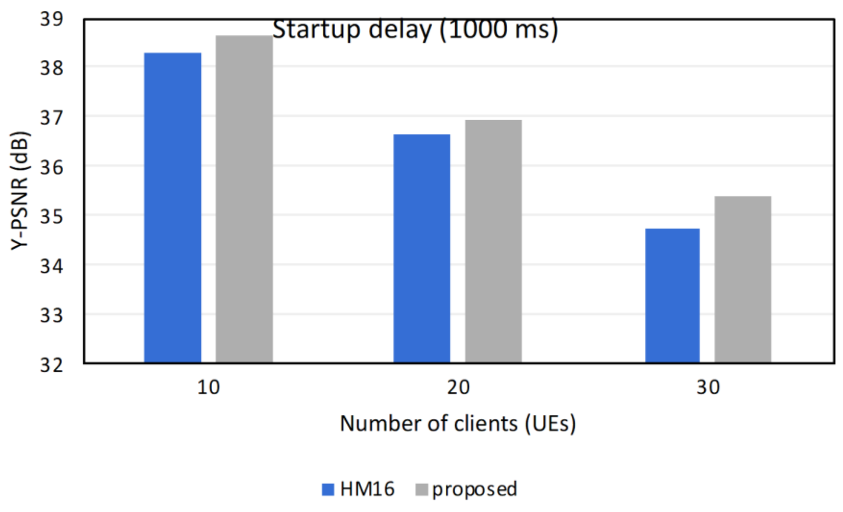


Figure 4.21 ASE video quality performance with start-up delay (1000ms)

The most interesting finding is that when we increase the number of clients, the proposed algorithm outperforms the default reference software. Moreover, when the encoded video sequences are streamed with high network load (30 users), the probability of increasing dropped packets is increased significantly due to network congestion. As a result, the objective video quality is deteriorated further at a higher network load with a shared bandwidth network.

4.7 Conclusions

This chapter presents an efficient H.265|HEVC error resilience algorithm to support low delay video delivery applications. The experimental results reveal a significant improvement in quality performance of decoded video with the proposed algorithm in error-prone conditions. The ASE algorithm is compared with the most state of art related algorithms: the region of interest (ROI) and improved region of interest (IROI). However, the ASE algorithm was limited with achieving high coding efficiency compared with the default reference software. The findings indicate that in some video decoding cases, the subdivision process into three different areas affect on the reconstructed quality with high textured details. Furthermore, the modified rate control suffers from bad rate allocation in the transition areas.

The comparison study is conducted in error free condition and erroneous environment with various PLRs of (0-18) (%). The most apparent finding from the obtained results with PLRs of (2-18) (%) is a significant improvement in visual quality with an average (Y-PSNR) gain of (1.076dB) compared to IROI, while in ROI algorithm the quality gain was (2.283dB). The ASE algorithm was also compared with the default video H.265|HEVC coding standard HM16.06 + slice copy method, and the gain was (4.5 dB).The experimental results reflect a significant improvement in terms of the quality performance of decoded video with the proposed algorithm in error-prone conditions. This improvement, however, compromises visual quality in error-free conditions. The objective visual quality loss in error free-environment for HM16.06 + slice copy method, ROI, and IROI was (-1.09 dB), (-0.6 dB), (-0.31dB), respectively. The current findings indicate that the proposed algorithm can provide visual quality improvements in error-prone conditions at the cost of slight (Y-PSNR) loss in error-free conditions.

In terms complexity evaluation, the proposed ASE algorithm is evaluated in terms of computational complexity. Both encoding and decoding excution times are recorded. The achieved results are compared with the default reference HM 16 as a benchmark. It was found that the encoding and decoding processing times are increased by (19%) and (11%), respectively. this increase in excution time comes from implementation of adaptive partitioning approach which mainly contribute to increase required processing power at the encoder side. Furthermore, the error detection and parsing process at slice segments level increase the computation processing at the decoder side.

The proposed algorithm is evaluated in LTE simluated network with various network loads, the achievd results show that the ASE algorithm can tolerate more than default reference software HM 16 under same limited bandwidth conditions. This study strengthen the idea of optomising the codec error resilience with real time computation processing to get more practical solutions in video transmission such as 4K video broacasting services.

Chapter Five

5 Joint Encoder-Decoder Error Resilience

This chapter presents proposed error-resilience algorithm depending on collaboration between both video encoder and decoder sides. The chapter is organised as follows. Section 5.1 gives a brief overview of H.265|HEVC coding tools relevant to the proposed the proposed work. Section 5.2 presents technical description of H.265|HEVC parameter sets highlighting its features to support error resilience implementations. Section 5.3 describes reference picture management concept in H.265|HEVC coding standard to improve its error robustness. Section 5.4 describes the mechanism of using H.265|HEVC error concealment to improve the perceived visual quality at H.265|HEVC decoder. The proposed error-resilience algorithm based on joint encoder-decoder using Supplemental Enhancement Information is described in Section 5.5. The experimental setup to evaluate the proposed algorithm is reported in Section 5.6. Section 5.7 presents the achieved quality evaluation results in terms of objective (Y-PSNR) and subjective (frame by frame assessments). Section 5.8 covers computation complexity analysis study of the proposed work. Finally, Section 5.9 summarises and concludes the chapter.

5.1 Introduction

As it mentioned before, one of the main best practical solutions to recover the received errors at the decoder side and accordingly improve the perceived visual quality is to use joint video encoder and decoder error control approach. In this research study, proposed algorithm called Error Resilience based Supplemental Enhancement Information is presented and described. The implementation of the proposed (ERSEI) algorithm is divided into two parts: encoder and decoder parts.

In H.265|HEVC coding standard, the Supplemental Enhancement Information (SEI) are high level syntax that generated optionally within NAL data. It is defined as non VCL type with NAL unit header types (PREFFIC_SEI_NUT) and (SUFFIX_SEI_NUT).

Some SEI messages are inherited from the previous standard which are used to support temporal video scalability in previous H.264|H.264|AVC coding standard. It is also used in network adaptation for different temporal layer switching points. New SEI syntax messages have been added to H.265|HEVC standard produced for future video coding extensions like multiview, 3D coding, and scalable video coding extensions. In addition to error detection and tracking purposes [87]. In general, as SEI provide metadata about bitstream construction, it is optional and does not restrict the decoding process of H.265|HEVC coding standard. Hence, it coded within NAL units bitstream. IT is worth noting that the new added feature in H.265|HEVC compared to H.264|AVC standard is that it can be encoded as suffix which can follows the VCL data. Another very important feature that can be used in error resilience called decoded picture hash SEI message. This message defines three hash calculation types including: CRC, MD5, and checksum calculation methods. These messages can be used to inform the encoder side about decoded picture status at the decoder side. in this research, the encoder is modified to support newly introduced SEI message in H.265|HEVC coding standard. Furthermore, the decoder is modified to conform and read the modified encoded SEI messages. The ERSEI algorithm relies on receiving feedback update from the decoder side using reliable transmission channel. A modification on both H.265|HEVC encoder and decoder does not influence the decoding process, and it is backward compatible to older standard versions.

The most common H.265|HEVC video transmission system involves adding a media aware tool named Media Aware Network Element (MANE) which is a signalling tool between video transmitter and receiver [45]; Figure 5.1 demonstrates a video network scenario using MANE tool. This tool sends a feedback control signal about the network connectivity status to the video receiver [45]. The H.265|HEVC bitstream contains NAL unit header information which contains information about bitstream coding standard profiles and levels [87]. The MANE tool makes a smart decision adaptation depending on encoded metadata, i.e. profiles and levels [45]. Based on the collected data from the bitstream metadata and network connectivity status, a MANE decides to employ a local recovery and redundancy coding tool on the oncoming video bitstream [45].

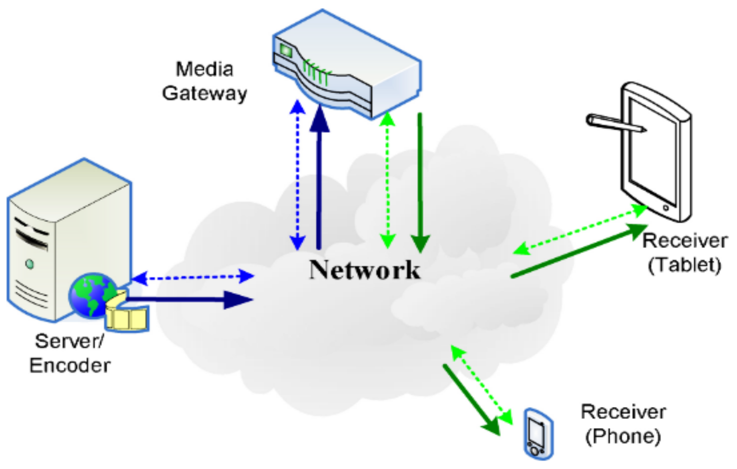


Figure 5.1 Video transmission scenario for H.265|HEVC coding standard [45]

The strength of ERSEI algorithm is that because the error resilience redundant data are encoded within SEI non-VCL NAL units, these redundant information at the decoder side or in the media gateway can be easily accessible without spending more computational power on parsing each slice header in the encoded bitstream. In general, a ERSEI algorithm is defined as joint error resilience-concealment mechanism developed to be used in error-prone environments for delivering more robust coded H.265|HEVC bitstream, a use case scenario of ERSEI can be depicted in Figure 5.2.

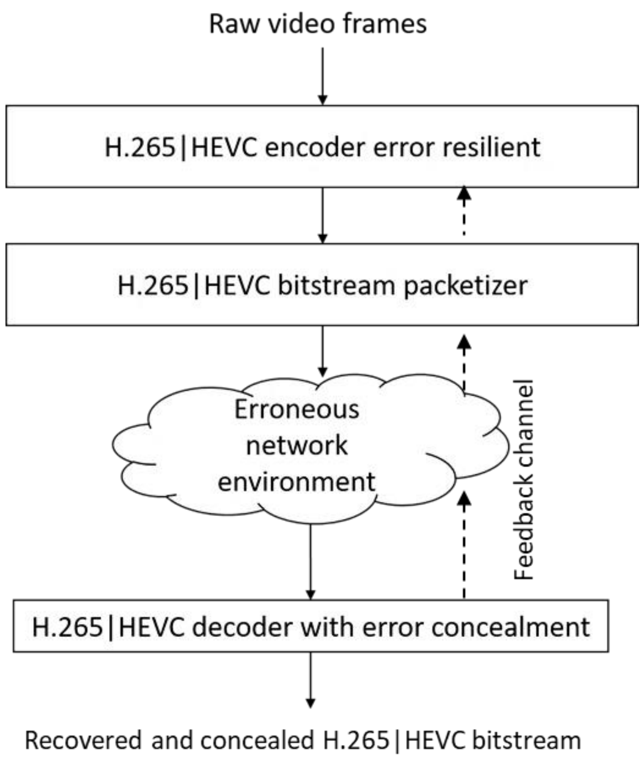

Raw video frames
H.265|HEVC encoder error resilient
H.265|HEVC bitstream packetizer
Erroneous network environment
Feedback channel
H.265|HEVC decoder with error concealment
Recovered and concealed H.265|HEVC bitstream

Figure 5.2: joint error resilience-concealment mechanism for H.265|HEVC bitstream

5.2 Parameters Sets and Error Resilience

In general, the H.265|HEVC video coding standard inherits same parameter set concept from H.264|AVC video coding standard. The main aim of introducing parameter set concept in H.265|HEVC video coding standard is to overcome RTP transmission issues of corrupting picture header or sequence headers in coded video sequence[45]. Furthermore, it support improve synchronisation and share important encoding syntax elements between encode and decoder using in bound or out bound control signal in reliable channels [88].

In H.265|HEVC video coding standard, two encoding parameter sets are inherited from H.264|AVC coding standard which are Sequence Parameter Set (SPS) and Picture Parameter Set (PPS). A new parameter set called Video parameter set (VPS) is introduced in H.265|HEVC video coding standard. The VPS consists of syntax elements used to share different layers and sub-layers of the compressed video sequence [88]. This feature help to improve coding efficiency of the coding standard. The activation process of parameter set in H.265|HEVC bitstream is summarised in the following prargraph.

A slice header contains shared information between slice segments. This shared information differs from slice to slice at frame level. A reference selection list is updated in each slice header and signals explicitly. The slice header information of each frame is stored in picture parameter set (PPS). The PPS data is stored in a sequence parameter set (SPS). Finally, a video parameter set contains shared information of the PPS and SPS [88]. Further demonstration of the interconnection of the three parameters set is shown in Figure 5.3.

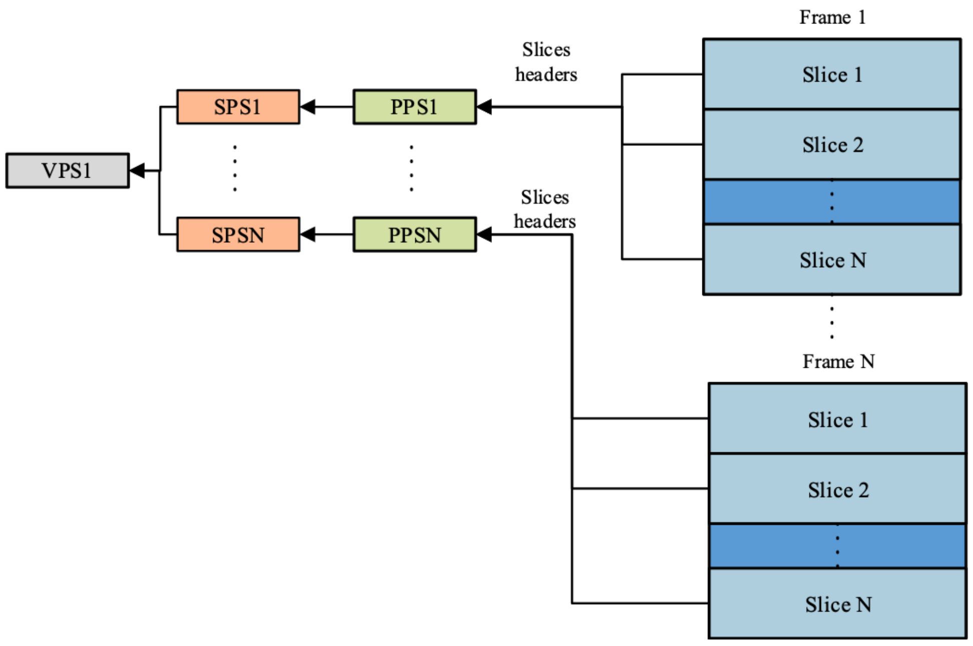


Figure 5.3 Activation parameters sets for H.265|HEVC coding standard

5.3 Error Resilience and Reference Picture Management

Multi reference pictures concept used in the first time in motion compensation process in extended version of Annex U in H.263+ video coding standard. Then, the flexible referencing concept further extended to include control and manage the decoded frames at decoder buffer with identification reference order for each reference picture [157].

The goal of using the referencing management concept at decoder buffer is the coding efficiency due to increase temporal prediction performance. Furthermore, it has been implemented to improve encoding error resilience by reducing temporal dependency and thus reducing error propagation by detecting and tracking the lost or corrupted reference pictures [158].

A new referencing management system called Hypothetical Reference Decoder (HRD) is used in both H.264|AVC and H.265|HEVC video coding standards. The main difference between the HDR and previous referencing management systems is in reference picture marking process [76]. Vast improvements are accomplished to make sure that each slice header is updated full referencing picture data for the current and future frames [159]. In this referencing management system, there are three marking pictures types: unused for reference (not used in future as a referenced picture), used for

short term reference (for the most recent and updated referenced picture), and used for long term reference (to be used for the current and future pictures). The HRD is implemented at decoded picture buffer (DPB) [43]. The H.264|AVC coding standard system employs control signal named (framNum) for tracking and checking the availability of referenced pictures at DPB [43]. Furthermore, it can be used to buffer a non-referenced decoded pictures parsing MMCO commands [43]. In addition, sliding window mechanism is used in H.264|AVC coding standard for tracking and error handling purposes [76]. Another referencing mechanism called
Memory Management Control Operation (MMCO) [76]. In this referencing mechanism, an explicit control signal is used to mark more than one referenced picture [76].
It is worth to note, Both H.264|AVC and H.265|HEVC video coding standards use Hypothetical Reference Decoder (HRD) model to manage the decoded frame between the Coded Picture Buffer (CPB) and Decoded Picture Buffer (DPB) and improve coding efficiency. The HDR buffer model block diagram can be demonstrated in Figure 5.4 [45].

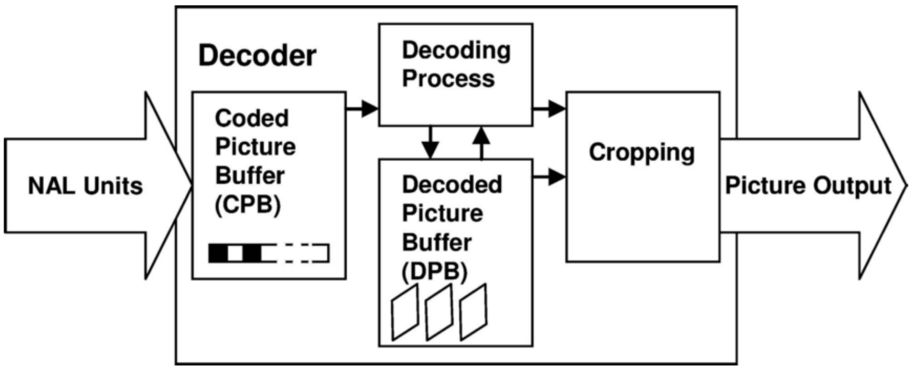


Figure 5.4: Hypothetical Reference Decoder (HRD) buffer model [45]

The reference pictures marking process can be depicted in Figure 5.5 for H.264|AVC coding standard on the left side and for H.265|HEVC on the right side [43]. As can be seen from the figure, the main difference between the codecs is that referencing update changes to the current decoded frame. In another way, vast improvements are added to H.265|HEVC standard to make sure that each slice header has a full update of a referencing picture list for the current and future frames. On the other hand, in H.264|AVC referencing marking process, only referencing changes to the current frame only [159]. Another difference is in the priority of reference marking according to frame decoding position [159]. In H.264|AVC, the referencing marking update is triggered after decoding the currently frame as demonstrated in Figure 5.5 [45]. The figure shows the difference in output and removal of reference pictures process between H.264|AVC and H.265|HEVC video coding standards.

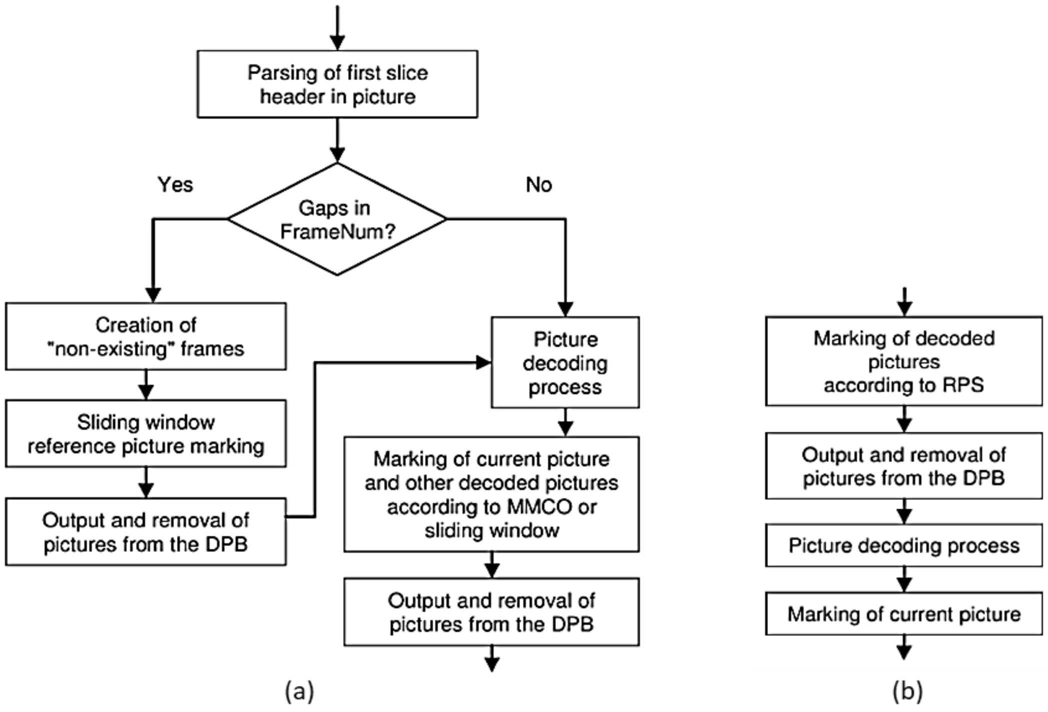

Parsing of first slice header in picture
Yes
Gaps in FrameNum?
No
Creation of "non-existing" frames
Sliding window reference picture marking
Output and removal of pictures from the DPB
Picture decoding process
Marking of current picture and other decoded pictures according to MMCO or sliding window
Output and removal of pictures from the DPB
(a)
Marking of decoded pictures according to RPS
Output and removal of pictures from the DPB
Picture decoding process
Marking of current picture
(b)

Figure 5.5 Reference pictures marking [45];
(a) H.264|AVC coding standard, (b) H.265|HEVC coding standard

For referenced pictures identification purposes, frame number ID known as (frameNum) is used in H.264|AVC coding standard [76].

In H.265|HEVC coding standard, the FrameNum is omitted and replaced with Picture Order Count (POC) number. At slice header level, a full set of reference picture list is received in regularly from DPB [76]. To identify the current frame will be employed in the prediction process or no, a RPS information at slice header will be compared with referenced pictures list at DPB. For error detection and recovery purposes, a feedback channel from the decoder is used to notify the encoder about the occurred errors. The H.265|HEVC coding standard use a flag named (used−by−curring−pic−X−flag) [76]. The encoder parses the slice header and checks the flag activation [45]. At the decoder side, a slice header RPS is checked against the available reference pictures at the DPB [45]. If there is an RPS data at the slice header but it is not available at DPB, it will be considered as this slice will not be used in the prediction process [45]. However, if the flag is activated, then the encoded current slice is intended to be used in the prediction process but it is lost or corrupted. The decoder should take action to recover or conceal the corrupted slice which is available in referenced pictures. Such errors could lead to severe degradation in video quality for the current and future inter-coded slices such as B-slices. Therefore, a suitable action for error concealment at the decoder should be applied which is out of scope of H.265|HEVC video coding standard.

5.4 H.265|HEVC Error Concealment Techniques

Error concealment process in video processing can be defined as employing spatial and temporal dependent data to conceal the erroneous decoded video blocks. The aim of these techniques is to reduce temporal and spatial error propagation on the perceived visual quality at the decoder. As in previous coding standard standards, the error concealment techniques are employed at the video decoder side [75]. There are two concealing approaches to hide the decoded erroneous video samples at H.265|HEVC decoder; temporal and spatial error concealment approaches [80]. A temporal error concealment process employs correctly received decoded motion data to reduce perceived visual quality at zero redundancy. This approach contributes to reduce decoded temporal error propagation [173]. The other concealing approach is using spatial error concealment, in this approach the damaged blocks are recovered by using interpolation techniques on correctly received neighbour blocks [76].

The performance of temporal error concealment approaches depends on correctly received encoded motion vectors.

A conventional method to conceal lost block is to replace it the previous decoded blocks. If motion vector of the damaged block is corrupted, a zero value is placed for the corrupted motion vector which means copy the same block location of previous frame [80], [160]. This method gives good quality results with concealing errors of video with stable areas [80]. However, with moving areas videos, a perceived visual quality suffers from noticeable freezing areas for moments. Another concealment technique includes locating the corrupted blocks with its motion vectors [160]. Then replace them with average motion vectors of the surrounding erroneous blocks [160]. This technique gives good results in flat (smooth) areas only. However, it gives unsatisfactory results when concealing sharp edges areas.

Recent studies have been conducted conceal corrupted or lost blocks in H.265|HEVC video coding standard. These studies employed a correctly received motion data to conceal the corrupted decoded frames. Y. Chang et al. proposed an error concealment algorithm to conceal corrupted blocks using block merging and residual energy. The proposed algorithm identify a reliable motion vector based on received residual energy, then the unreliable motion vectors are merged to create new motion vector to be used for error concealment at the decoder [161]

T. Lin et al. proposed temporal error concealment algorithm based on block partitioning decision. Their proposed algorithm extrapolates the motion vector of previous frames based on block partitioning decision.

5.5 Error resilience based on Supplemental Enhancement Information (ERSEI) algorithm

The design of the proposed ERSEI algorithm is divided into two parts; encoder and decoder sides. Both the encoder and decoder sides are work jointly to improve error resilience performance of H.265|HEVC coding standard. Figure 5.6 demonstrates the ERSEI encoding process using pseudo code.

At the encoder side, the encoder starts encoding a current frame based on receiving feedback signal with (Decoded picture hash) message. The feedback signal should be transmitted in out of band reliable channel. The (Decoded picture hash) messages is updated at both picture and video bitstream (NAL units) levels. The video contents in VCL-NAL unit type is checked to decide whether the current NAL unit is referenced picture type or non-referenced. Depending on comparison results, the encoder activates control signal to send encoded clean random-access picture for referenced NAL units. If the received NAL unit type is non-referenced picture, the encoder will encode and send Error Concealment Supplemental Enhancement Information (ECSEI) message to the decoder and inform the decoder side to start error concealment process.

```
//H.265|HEVC Encoder modifications
Input: NAL units Video data
Algorithm:
Step 1: parse the encoded Non-VCL header information
Step 2: Check the status of the decoded picture hash (DPH) message
        If DPH=error
        Begin
                ERSEI flag = 1;
                Check VCL type in the current NAL unit
                If VCL is Referenced type
                        Begin
                                Read recovery point message
                                 Encode Clean Random Access Picture in the located random access point
                        Else
                                Encode ECSEI message
                        End
        End
Step 3: Produce robust video data
```

Figure 5.6 Pseudo Code of ERSEI algorithm at the encoder side

At the decoder side, if there is no error in the received Decoded Picture Hash (DPH) message and no ECSEI message, this means that the transmission channel has error-free condition and the H.265|HEVC encoder will pass ERSEI algorithm checking process (Figure 5.7).

If DPH message received with error flag and ECSEI is activated, there two approaches to recover corrupted data depending on the current VCL-NAL unit type. If VCL-NAL unit type is referenced, then the decoder will decode the received random access picture from the encoder. If VCL-NAL unit type is non referenced type, the decoder will start spatial error concealment on the corrupted VCL data.

```
//H.265|HEVC Decoder modifications
Input: NAL units Video data
Algorithm:
Step 1: Check the status of the decoded picture hash (DPH) message
Step 2: If DPH=error
        Begin
                Update the encoder about the corrupted data using reliable feedback channel
        Else
        If ECSEI message is activated
                If VCL-NAL is referenced
                        Decode random access picture
                Else
                Start spatial error concealment process
                End
        End
Step 3: Decode video data
```

Figure 5.7 Pseudo Code ERSEI algorithm at the decoder side

5.5.1 H.265|HEVC Encoder Modifications

Figure 5.8 demonstrate the implementation of the proposed ERSEI algorithm at the encoder side. the encoding process is described as follows. The encoding process depends on receiving regular update signal from decoder side. If there is no indication of error message, the encoder will pass other checks and start encoding the video bitstream normally. If there is an error from (Decoded picture hash) message, then the encoder will use ERSEI algorithm.

In case of an error transmission occurred, a recovery point SEI and DPH messages are used to update the encoder about a corrupted slice segment location in VCL-NAL video units and ERSEI update messages. Once the encoder is notified about the error location. The encoder will then check the VCL-NAL unit type (referenced or non-referenced). If a VCL-NAL type is encoded as a referenced picture, it means the temporal encoded frame will be corrupted and interprediction motion data will be incorrectly used at the encoder side. Furthermore, motion vector fields include motion predictors of other dependent slice segments will be incorrectly decoded and causes error propagation. Consequently, all the dependent inter picture prediction will be incorrectly decoded at the decoder. To reduce error propagation at the decoder, error recovery message in the non-VCL NAL unit is used. The main aim of using error recovery message in H.265|HEVC coding standard is to precisely locate corrupted video segment and keeps the encoder in synchronisation with the decoder side. On the other hand, if the VCL-

NAL type is a non-referenced picture, ECSEI message will be encoded at the first bit of non-VCL NAL unit. This is to notify the H.265|HEVC decoder to take error concealment action.

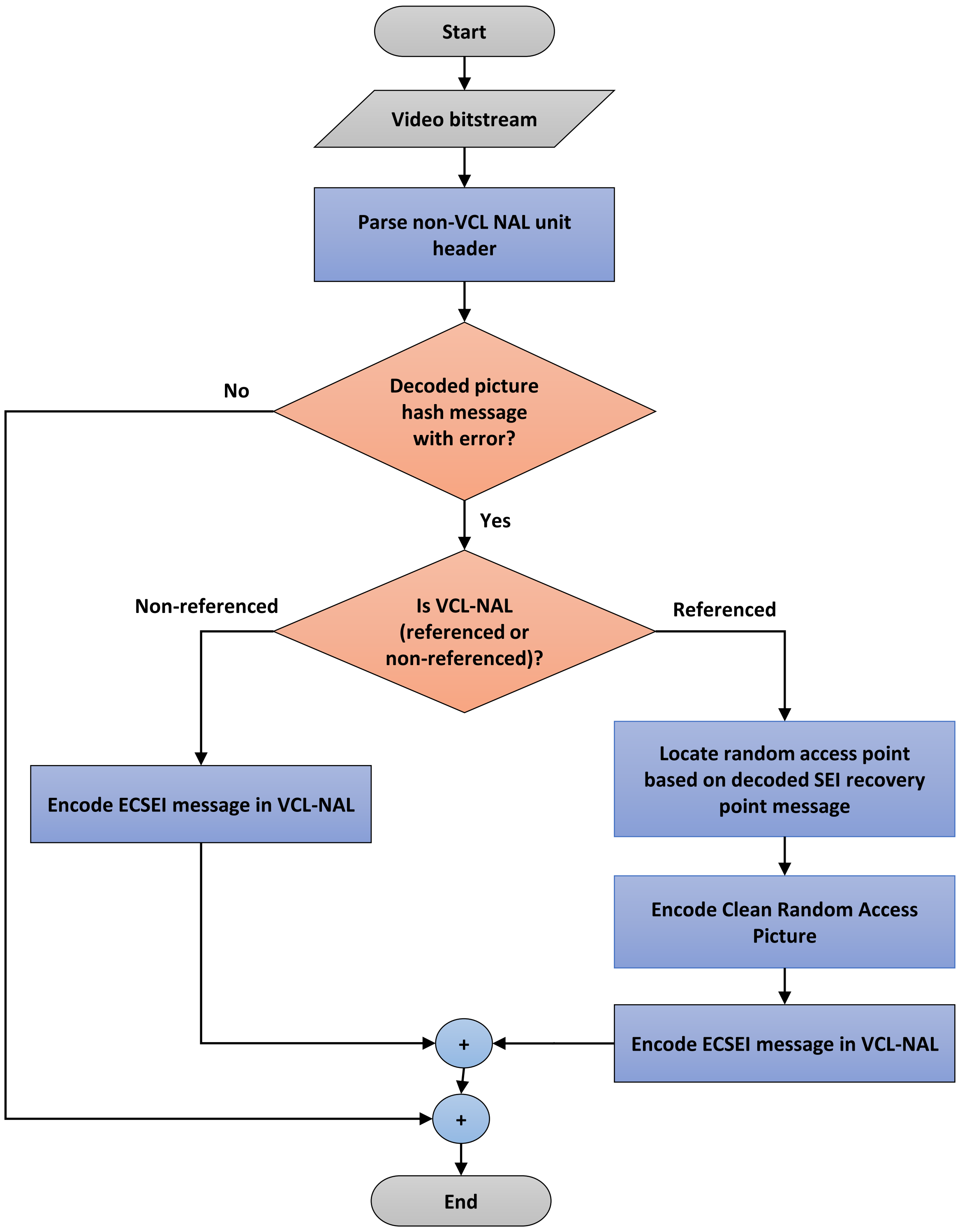

Start
Video bitstream
Parse non-VCL NAL unit header
Decoded picture hash message with error?
No
Yes
Is VCL-NAL (referenced or non-referenced)?
Non-referenced
Referenced
Encode ECSEI message in VCL-NAL
Locate random access point based on decoded SEI recovery point message
Encode Clean Random Access Picture
Encode ECSEI message in VCL-NAL
+
+
End

Figure 5.8 Flowchart of ERSEI algorithm at H.265|HEVC Encoder

5.5.2 H.265|HEVC Decoder Modifications

In this section, a decoder modification of proposed ERSEI algorithm are discussed. Figure 5.9 shows a flowchart of ERSEI decoding process.

In this decoding, the H.265|HEVC decoder parse the received encoded NAL units at Video Coding Sequence (VCS) level. A received bitstream at the decoder contains one or several VCSs. Each VCS consists of non-VCL NAL and VCL-NAL units. The decoder will check first the received DPH message from update signal. If there is an error in DPH message, an error recovery message will be encoded to the transmitted bitstream to notify the encoder about the location of the corrupted slice segments. If there is no error, the decoder will do another check called ECSEI checking process. In this process, the ECSEI message with (00111100) value will be checked after the first bit of the current non-VCL NAL unit. If the ECSEI is activated, then the decoder will check if VCL data is referenced or non-referenced types. If the VCL data is referenced, the decoder will start decoding the coming Access unit as Random Access Picture and stops the error propagation at the decoder side.

If the VCL is non-referenced type, the detected erroneous slice segments at slice headers will be concealed at the decoder side.

The concealment process at the decoder is based on interpolation technique to conceal the currently decoded frames proposed in [58]. The concealment of the corrupted slice samples can be defined as in the following equation [58]:

$$\text{Interpolated pixel } (p_0) = \frac{\sum_{i=1}^{PB} d_i p_i}{\sum_{i=1}^{PB} d_i} \qquad \text{Eq.(5.1)}$$

Where p_i is the pixel value in the adjacent intra predicted blocks, and d_i is the closest distance of the surrounded pixels at the borders of corrupted or missing blocks [58]. To obtain the best balance between decoding processing complexity and perceived video quality, a spatial interpolation concealment method is used. The error concealment action will insure to reduce the visual artefacts resulted from spatial error propagation while waiting to receive a new clean random-access picture. The coming sections reports the experimental work setup and the achieved evaluation results.

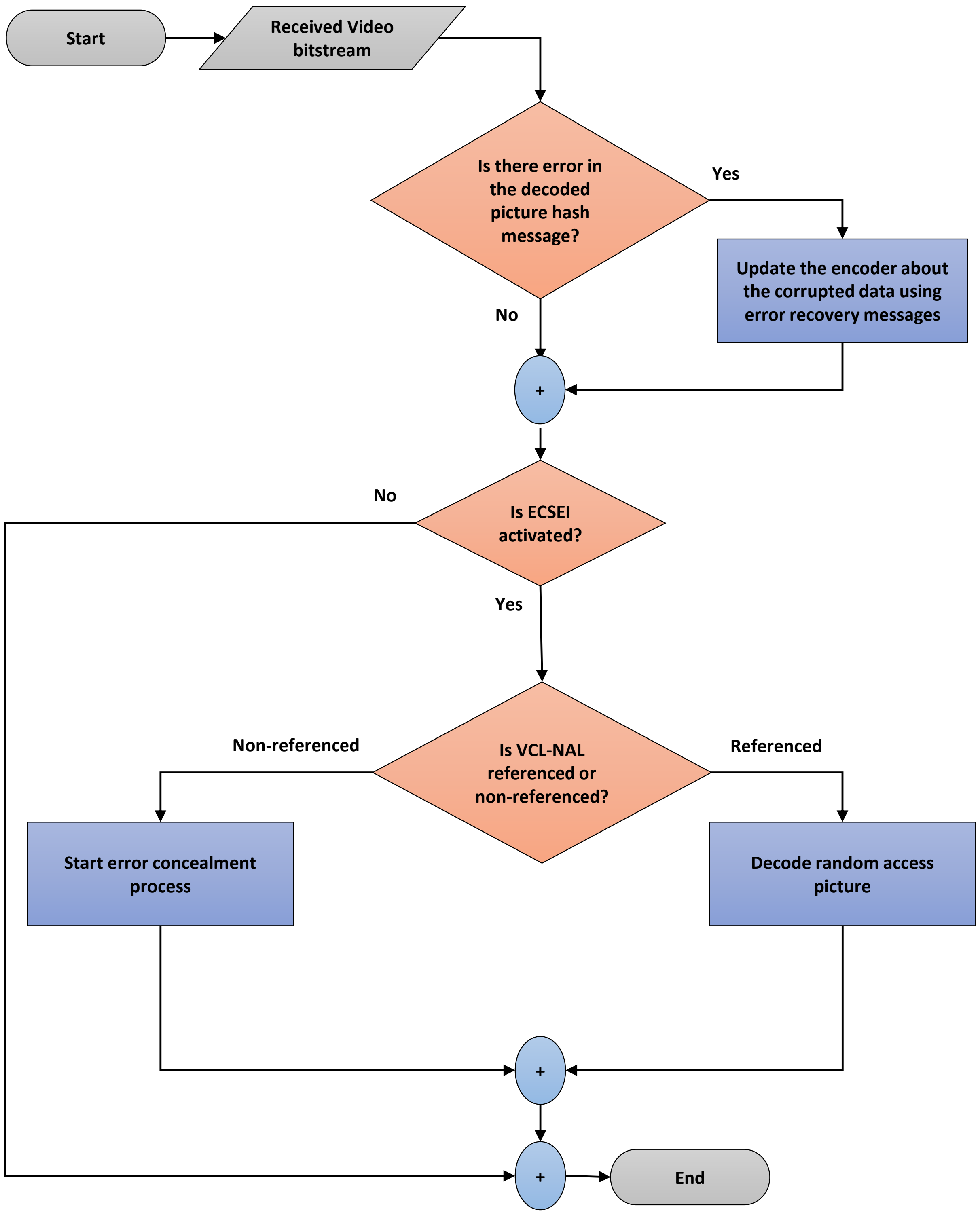

Start
Received Video bitstream
Is there error in the decoded picture hash message?
Yes
Update the encoder about the corrupted data using error recovery messages
No
+
No
Is ECSEI activated?
Yes
Non-referenced
Is VCL-NAL referenced or non-referenced?
Referenced
Start error concealment process
Decode random access picture
+
+
End

Figure 5.9 Flowchart of ERSEI algorithm at H.265|HEVC Decoder

5.6 Experiments Setup

This section reports the used encoding settings and testing conditions for evaluating the proposed ERSEI algorithm. During the experimental setup, the proposed ERSEI algorithm is implemented modified version of reference software HM16. A default reference software HM16 with pixel copy method and Motion Compensated Error Concealment (MCEC) algorithm proposed in [161] are used as a bench marks in evaluation process. The input video test sequences are reported in [Appendix A]. For evaluation the proposed ERSEI algorithm with various PLRs, a video quality evaluation testbed described in Section 3.11.1) is implemented with packet loss generator software [130]. For achieving fair objective evaluation results, each video test sequence is repeated 30 times with different PLR seed and the averaged Y_PSNR values are recorded. All video test sequences are encoded using same encoding setting reported in Table 5.1.

Table 5.1 Encoding configuration setting for ERSEI evaluation work

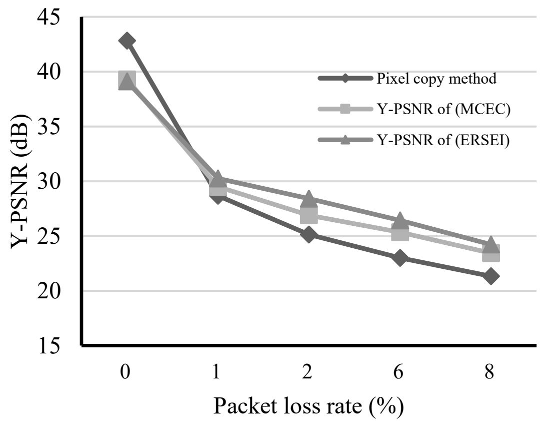
Encoder parameter	configuration
Profile	Main
Encoding GOP size	8 frames
Rate control	Disabled
Filtering	Enabled
Search range	64 block size
bit depth	8 bit depth
Intra mode	1st frame in each GOP
(max/min) transform unit size	32/4 block size
Largest coding unit (LCU) size	64x64 block size
Frames to be encoded (-f)	Listed in Appendix A
Input file	Listed in Appendix A
SourceWidth (-wdt) x SourceHeight (-hgt)	Listed in Appendix A
Quantisation parameter	28 value
LCU/slice	20 (unit)/slice
Intra period	1/20 (P-frame)

5.7 Experiments and discussions

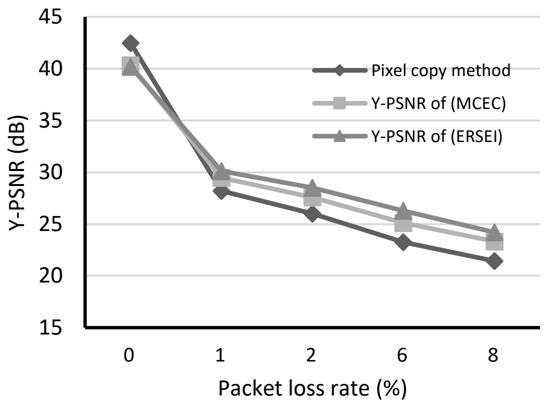
5.7.1 Objective evaluation

This section presents achieved results of objective evaluation work on the proposed (ERSEI) algorithm. Both error-free and error-prone conditions are reported in experimental work. The obtained objective results in error-prone conditions are averaged Y-PSNR values with 30 test runs encoded with 300 frames in each test video sequence. Figure 5.10 shows achieved objective evaluation results for (Carphone, Mobile, and News) video sequences. All video test sequences were randomly injected with packet loss rate PLRs ranging (0-8) (%).

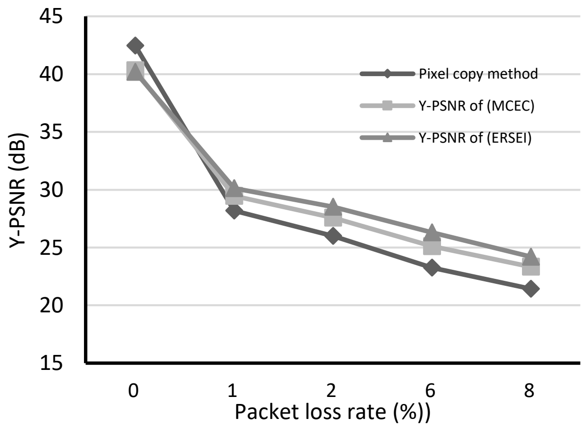
The average Y-PSNR values for ERSEI algorithm compared to HM16 with pixel copy and MCEC algorithm with various PLRs are reported in Table 5.2, Table 5.3, and Table 5.4.



(a) Carphone video sequence



(b) Mobile video sequence



(c) News video sequence

Figure 5.10 Achieved objective results of ERSEI algorithm compared to HM16 using pixel copy and MCEC algorithm with various PLRs

Table 5.2 Video quality in terms of Y-PSNR (dB) vs PLRs for Carphone sequence

PLR (%)	pixel copy vs MCEC	ERSEI vs MCEC
0	**-0.631**	**-0.246**
1	1.661	0.683
2	2.556	1.465
6	2.810	1.125
8	2.168	0.847
average	**2.298**	**1.030**

Table 5.3 Video quality in terms of Y-PSNR (dB) vs PLRs for Mobile sequence

PLR (%)	pixel copy vs MCEC	ERSEI vs MCEC
0	**-0.522**	**-0.188**
1	1.572	0.762
2	3.265	1.516
6	3.430	1.099
8	2.897	0.785
Average	**2.791**	**1.040**

Table 5.4 Video quality in terms of Y-PSNR (dB) vs PLRs for News sequence

PLR (%)	pixel copy vs MCEC	ERSEI vs MCEC
0	**-0.671**	**-0.127**
1	1.929	0.653
2	2.533	0.945
6	3.052	1.186
8	2.777	0.871
Average	**2.573**	**0.914**

The discussion of the objective results begins with evaluating the effectiveness of the proposed ERSIE algorithm in error-free and error-prone conditions.

In error-prone conditions, the objective results are reported in previous tables from taking the average Y-PSNR for four different PLRs (1,2,6,8) in (%).

It can be seen from the obtained results in error prone conditions. For Carphone sequence, the objective results of ERSEI algorithm outperforms the default HM16 software with pixel copy and MCEC algorithms by (2.298 dB) and (1.030 dB), respectively. For News sequence, the objective results outperform the default HM16 software with pixel copy and MCEC algorithms by (2.573 dB) and (0.914 dB), respectively. In the same error conditions, for Mobile sequence, the objective results of ERSEI algorithm outperforms the default HM16 software with pixel copy and MCEC algorithms by (2.791 dB) and (1.040 dB), respectively.

Table 5.5 reports the overall averaged Y-PSNR values for all three video sequences. It can be found that the Y-PSNR of ERSEI algorithm with different PLRs has improved by (2.554 dB), and (0.995 dB) compared to pixel copy and MCEC algorithms, respectively. In error-free conditions (PLR=0%), Y-PSNR of the ERSEI algorithm is reduced by (-0.608 dB; pixel copy), (-0.187 dB; MCEC).

Table 5.5 ERSEI evaluation in terms of Y-PSNR vs PLRs

Sequence name	**Error-free condition (Y-PSNR)**		**Error-prone condition (Y-PSNR)**	
	pixel copy vs ERSEI	**MCEC vs ERSEI**	**pixel copy vs ERSEI**	**MCEC vs ERSEI**
Carphone	-0.631	-0.246	2.298	1.030
Mobile	-0.522	-0.188	2.791	1.040
News	-0.671	-0.127	2.573	0.914
Average	-0.608	-0.187	2.554	0.995

In conclusion, the proposed algorithm in error-prone conditions exceeds an HM 16 using a pixel copy method and Motion compensated error concealment (MCEC) method by (2.554 dB) and (0.995 dB), respectively.

Therefore, based on corrupted video data condition, an error concealment action will be applied to reduce error spatial propagation.

In terms of evaluation the proposed ERSEI algorithm with different encoding bit rate (Kbps), Figure 5.11 presents obtained objective results from injecting PLR=4% into video test sequence (News) encoded with various bit rates. The evaluation includes testing the ERSEI, HM16 with pixel copy, and MCEC algorithms.

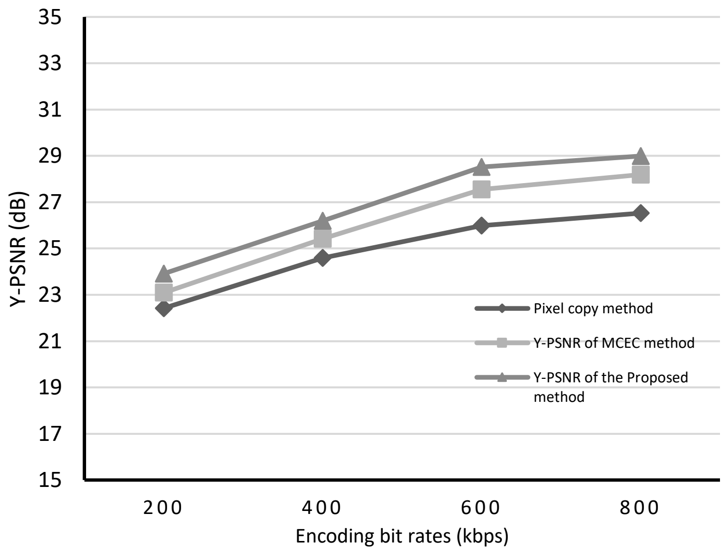


Figure 5.11 Y-PSNR vs Encoding bit Rates for ERSEI, HM 16 with pixel copy, MCEC algorithms

The figure highlights the consistency of the proposed algorithm in terms of objective video quality Y-PSNR over other two algorithms. This consistency indicates that in the modified SEI messages proposed ERSEI algorithm contribute to enhance decoded objective quality with increasing encoding bit rates more than the default reference software HM 16 in erroneous conditions. This benefit comes at the cost of reducing coding efficiency due to increasing the number of encoded random-access pictures.

5.7.2 Frame by Frame Quality Assessment

The proposed ERSEI algorithm is further evaluated using subjective quality assessment. Pre-selected frames are extracted from the raw video test sequence for quality assessment. A PLR with 2% is injected into Carphone, News, and Mobile test sequences. The selected input sequences are encoded with 30 fps at CIF resolution. The number of encoded frames in the experimental work are 300 frames. In this evaluation, the video test sequence is encoded with both ERSEI algorithm and the MCEC algorithm [161]. Randomly preselected decoded video frames are extracted from the erroneous bitstreams. The frame by frame assessments for the (Carphone, News, and Mobile) video sequences are shown in Figure 5.12, Figure 5.13, and Figure 5.14, respectively. From the subjectibe assessment results. It can be concluded the following.

One interesting finding in Figure 5.12 is that ERSEI algorithm successfully preserved most of the man body textured details. MCEC algorithm, on the other hand, is less effective in terms of preserving the important edges details. This is because in MCEC algorithm, some of generated motion vectors are incorrectly considered as reliable and lead to incorrect displacement of prediction units. Another important finding was that we can also see in the same figure that some slice segments in the shoulder and mouth areas are displaced when encoded using the ERSEI algorithm.

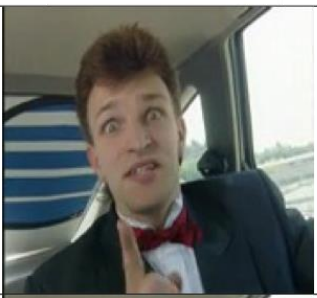
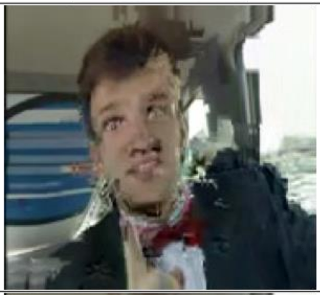
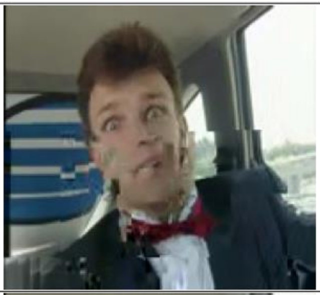
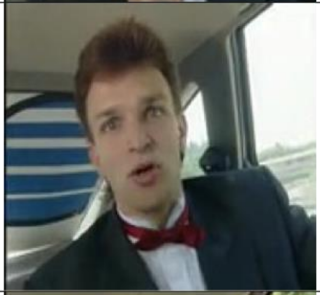
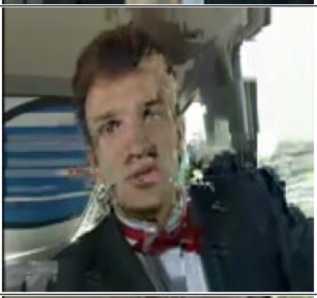
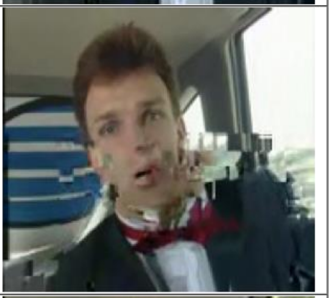
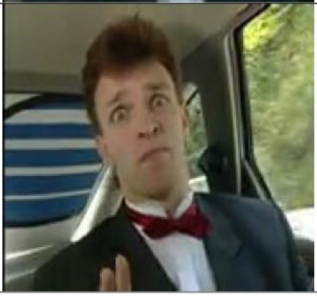
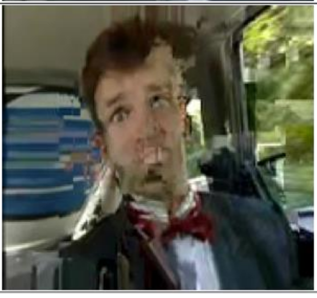
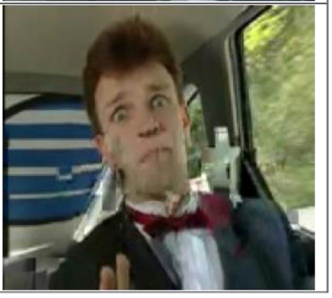
Frame number	Raw frame	MCEC	ERSEI
Frame 125th			
Frame 155th			
Frame 212th			

Figure 5.12 Subjective comparison of ERSEI compared to MCEC algorithms for Carphone sequence

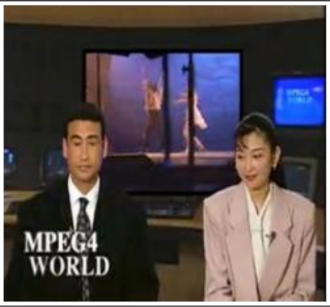
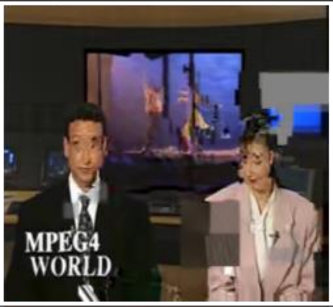
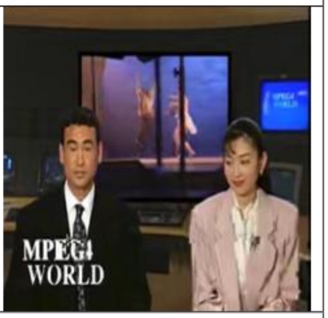
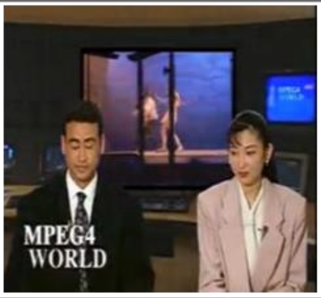
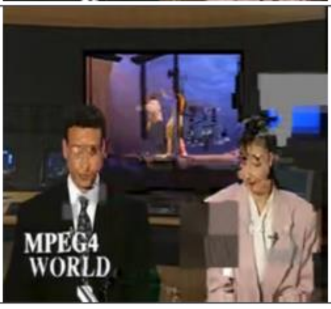
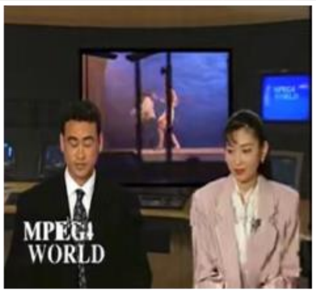
Frame number	Raw frame	MCEC	ERSEI
Frame 200th			
Frame 220th			

Figure 5.13 Subjective comparison between ERSEI and MCEC algorithms for News sequence

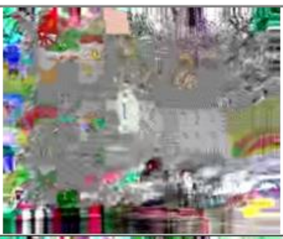
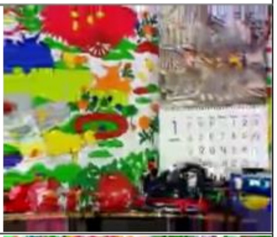
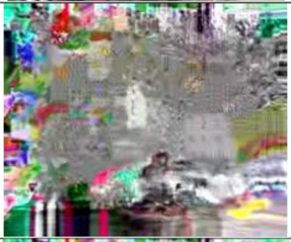
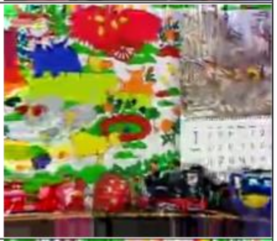
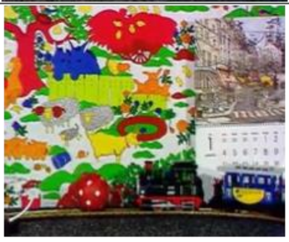
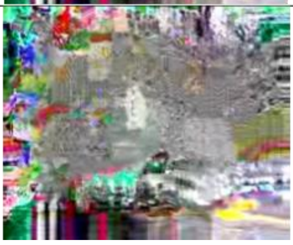
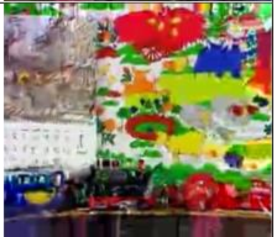
Frame number	Raw frame	HM16	ERSEI
120			
150			
170			

Figure 5.14 Subjective comparison between ERSEI and HM 16 for mobile sequence

From the frame by frame subjective companion result in Figure 5.14. It can be seen in that the reference encoding software for H.265|HEVC coding standard has failed to preserve any clear texture details in Mobile test sequence. From this point we can conclude that some slice segments failed to be concealed using pixel copy method in the fast-moving objects. Further, a significant improvement in visual quality was achieved when using the modified version of HM reference software compared to the default HM16 software in error-prone condition.

5.8 Computation complexity

This section focuses on evaluating the proposed ERSEI algorithm in terms of computation complexity. The experimental work includes comparisons between the modified reference software HM 16 with ERSEI algorithm and default reference software as a benchmark using low delay encoding configuration. In this experimental work, both encoding and decoding execution times are recorded using Intel Core i7-7200 CPU processor with 2.7GHz. The installed RAM was 8 GB. As mentioned earlier, the main purpose of using HM reference software is to provide common reference implementation for video coding developers. As both HM16 and modified HM16 with ERSEI algorithm are written in C++ language, under these circumstances, the consumed processing time for encoding and decoding classes are recorded and evaluated. The experimental work conducted for taking an average of the consumed encoding time of eight video sequences reported in Table 5.6.

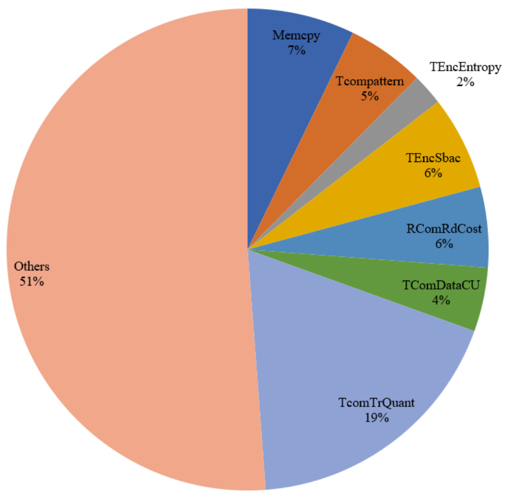
Table 5.6 video sequences (CIF resolution)

Sequence name	**Frame number**
Akiyo	300
Bridge (close)	2001
Bridge (far)	2101
Bus	150
Container	300
Coastguard	300
Flower	250
Foreman	300

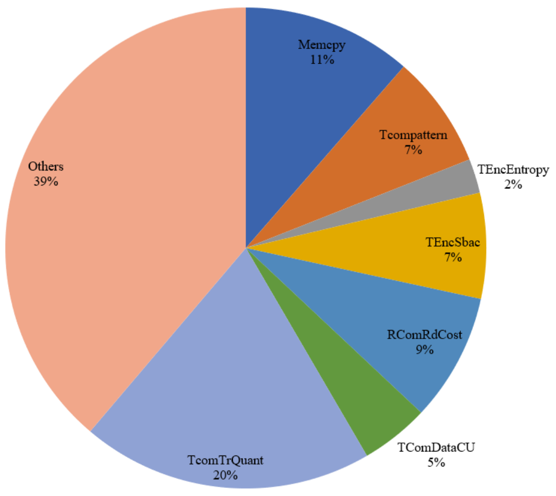
It is worth noting that, although the HM16 is implemented with high level language (C++) which may inversely affects the coding standard performance processing speed, the refence software developers have made significant improvements and considerations to obtain more ideal and fair coding standard evaluation [119]. The average time is recorded for each encoding and decoding class for both cases with and without ERSEI implementation.

5.8.1 Encoding complexity

This section presents computational complexity evolution for ERSEI algorithm compared with default HM 16. Furthermore, computational complexity of the ERSEI algorithm is evaluated with ASE algorithm. In general, the encoding complexity depends on different encoding settings in H.265|HEVC video coding standard. There are three main encoding settings according to the common test conditions which are AI, RA, and LB encoding settings. It should be noticed that during the experimental work, the parallel wavefront processing tool is disabled in this experimental work alongside with scaling and prediction metrics. The fast encoding setting is enabled with activation (fastsearch) encoding setting in both HM16 and modified HM 16 reference software. To calculate the execution time for each encoding component (writing in C++ class), the encoder code is profiled according to complexity evaluation method reported in [111]. Figure 5.15 shows the percentage of the encoding distribution time for different main encoding classes without ERSEI implementation in (Figure 5.15 (a)) and using Modified HM 16 (with ERSEI implementation) in (Figure 5.15 (b)).



(a) Without ERSEI error resilient algorithm



(b) With ERSEI algorithm

Figure 5.15 Encoding time distribution for ERSEI algorithm

From reported results, it can be noticed that a significant increase (3 %) of encoding processing time in (RComRdCost) class where rate- distortion optimization takes place. Important to realise that with the additional encoded bits overhead, the complexity of bit rate optimisations is increased trying keep the coding efficiency optimised with error resilience implementation.

In Modified HM16, the highest consumed processing time spent on quantisation encoding class (TcomTrQuant) (20 %) while in the HM16 was (19 %). The averaged consumed time spent on the other encoding classes is small. The entropy coding class was same as the default reference software. It should be noticed that the encoding memory requirements for the modified HM 16 software with error resilience is increased with additional 4 % of encoding processing time, whereas in coding unit's computation process with additional (1 %). For the other encoding classes, the encoding processing time is distributed with (39 %) with modified HM16 while with the default HM16 software (51 %) of consumed encoding processing time.

The obtained encoding processing is further compared with ASE error resilience algorithm using same encoding settings. A low delay-B configuration mode is used. A quantisation parameter is selected with (32). Table 5.7 and Figure 5.16 report the average execution time of eighteen video test sequences for both ERSEI and ASE algorithms compared to HM16 reference software (without error resilience).

Table 5.7 Encoding time of ERSEI algorithm with different video sequences

Sequence name		Frame number	Encoding time (seconds)		
			HM16	**HM16+ASE**	**HM16+ERSEI**
1	Hall monitor	300	575	664	697
2	Mobile	300	491	646	682
3	Container	150	275	298	342
4	Grandma	300	548	686	725
5	Akiyo	870	956	1232	1279
6	Miss-America	300	507	653	704
7	Bridge-close (far distance)	150	249	367	411
8	Mother-daughter	2001	1283	1492	1524
9	News	300	501	647	682
10	Bridge-far	300	498	622	679
11	Bridge-close (near distance)	2101	1355	1498	1540
12	Coastguard	2001	1294	1435	1492
13	Claire	300	485	588	625
14	Carphone	494	684	809	869
15	Highway	382	538	653	691
16	Salesman	2000	1385	1509	1564
17	Silent	449	649	844	896
18	Suzie	300	473	586	625
Average (sec)			708	846	890

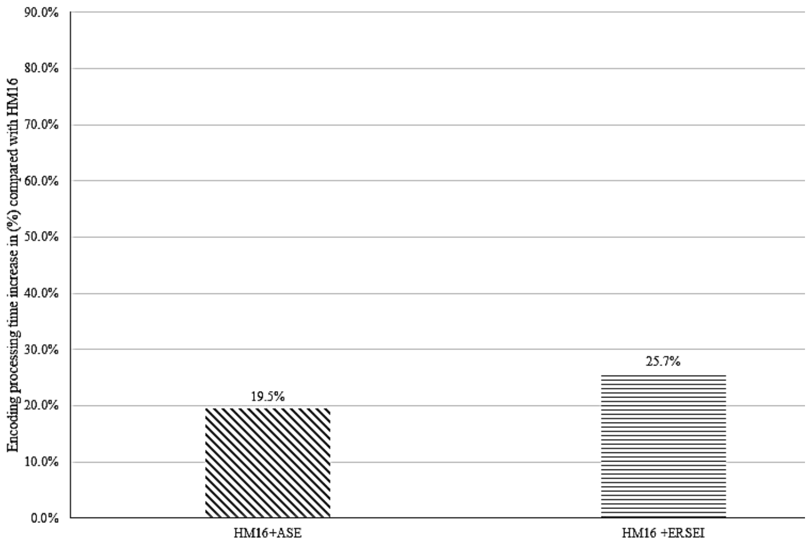


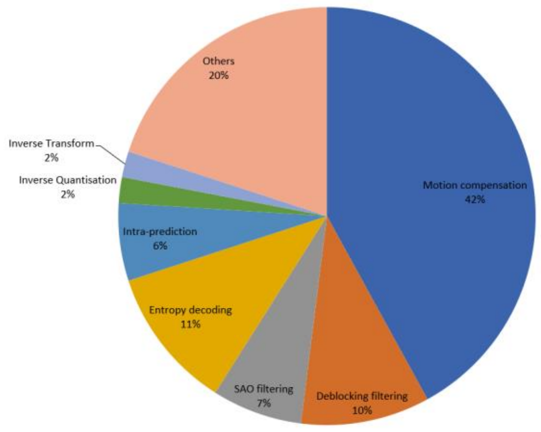
Figure 5.16 Encoding Computational complexity comparisons between ASE and ERSEI algorithms

It can be concluded from the obtained results is that the significant increase in encoding processing time for both proposed algorithms comes from several encoding components. However, the dominant encoding component come from (RComRdCost) class where rate- distortion optimization takes place. As the rate distortion optimisation process in the proposed works requires more processing power to parse each encoding video block and calculate its actual encoding bit cost.

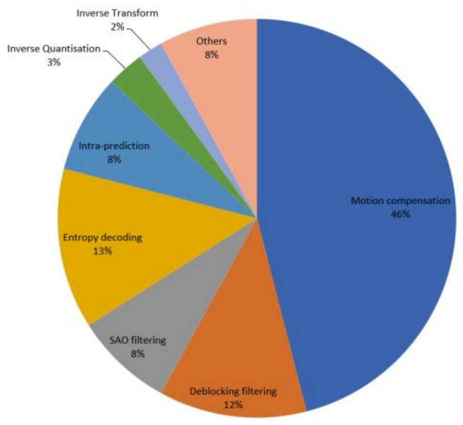
5.8.2 Decoding complexity

As with previous complexity section, this section presents complexity evaluation at the decoder side for the ERSEI algorithm first. The default reference decoder (HM 16) is used as a benchmark in this study. In second evaluation part, the decoding execution process of ERSEI algorithm is compared with decoding complexity of ASE algorithm using same encoding settings which are reported in previous section (5.8.1).

The achieved comparison results for ERSEI algorithm (Modified HM 16 decoder) compared with HM 16 reference software are reported in Figure 5.17. The decoding excution times are recorded by taking the averaged of the eight video test sequences (Table 5.6).



(a) Without ERSEI error resilient algorithm



(b) With ERSEI error resilient algorithm

Figure 5.17 Average Decoding time distribution for ERSEI algorithm

As it can be seen from decoding distributed time figure, the dominant decoding component in both cases (without and with ERSEI implementation) was the motion compensation process referred by (Motion compensation class). The computational complexity of ERSEI algorithm implementation is increased by (4%) compared with default reference HM 16 decoder (without error resilience implementation). For

Deblocking filtering, Entropy decoding, and intra-prediction classes, the decoding execution time with ERSEI implementation is increased by 2% compared with the default HM16 software. Regarding to SAO filtering and Inverse Quantisation classes, the decoding execution time in these classes (with ERSEI algorithm implementation) is increased by (1%).

Table 5.8 and Figure 5.18 report the achieved decoding execution times ERSEI algorithm compared to reference software HM16 and ASE algorithm.

Table 5.8 Decoding time of ERSEI algorithm compared to HM16 and ASE algorithm

Sequence name		**Number of frames**	**Decoding time (seconds)**		
			HM16	**HM16+ASE**	**HM16 +ERSEI**
1	Hall monitor	300	6.5	7.4	9.2
2	Mobile	300	6.8	7.8	9.6
3	Container	150	4.3	5.2	7.4
4	Grandma	300	5.8	6.6	8.4
5	Akiyo	870	9.3	10.2	11.8
6	Miss-America	300	5.8	6.3	8.2
7	Bridge-close (far distance)	150	4.2	5.4	7.1
8	Mother-daughter	2001	16.3	17.4	19.3
9	News	300	6.2	7.1	9.5
10	Bridge-far	300	5.7	6.4	8.2
11	Bridge-close (near distance)	2101	15.2	16.4	18.4
12	Coastguard	2001	14.3	15.1	16.9
13	Claire	300	5.8	6.6	8.5
14	Carphone	494	7.3	8	7.9
15	Highway	382	6.6	7.6	9.3
16	Salesman	2000	16.5	17.5	19.3
17	Silent	449	8.4	9.2	11.5
18	Suzie	300	6.5	7.3	9.2
Average (sec)			8.4	9.3	11.1

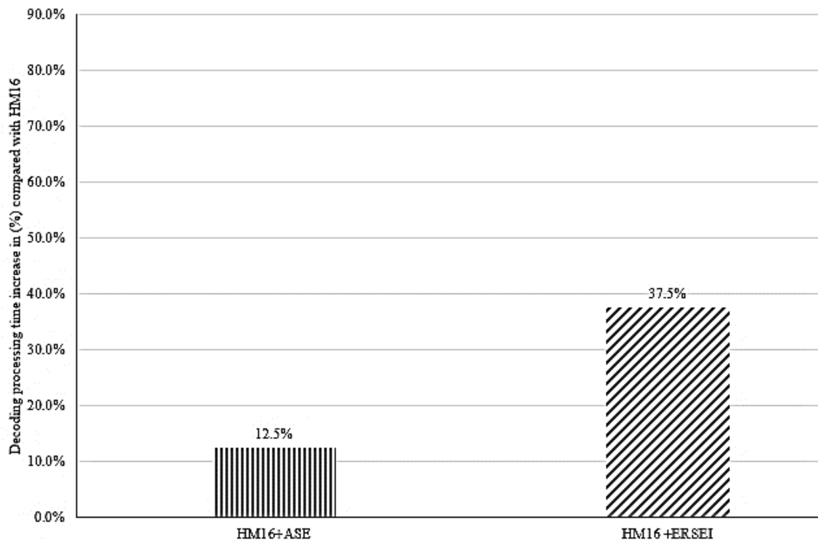


Figure 5.18 Decoding Computational complexity comparisons between ASE and ERSEI algorithms

As it shown from the figure, the decoding processing time for ERSEI algorithm reaches more than double compared with ASE decoding process. The reason of significant increase in decoding processing time is because SEI update messages in ERSEI algorithm are parsed and decoded at both access unit level and higher level i.e. coded video sequence. This decoding mechanism enforce the modified decoder to spend higher computational decoding processing.

Another important point is as timing SEI buffering periods is increased in ERSEI decoder side output delay of the decoded picture buffer is increased which adversely affect the decoding complexity process.

5.9 Conclusions

This chapter presents to improve error robustness of H.265|HEVC video coding standard using interactive encoder-decoder error resilience algorithm. The present study was designed to employ modified syntax elements of Supplemental Enhancement Information SEI messages. The modified SEI messages are encoded within non-VCL NAL units to improve perceived visual quality at H.265|HEVC decoder. The chapter presents proposed error resilience algorithm called Error Resilience based on Supplemental Enhancement Information (ERSEI). Depending on decoding conditions at the decoder, the encoder setting is adaptively changed based on received Error Concealment Supplemental Enhancement Information (ECSEI) and Decoded Picture Hash (DPH) messages. a feedback message status is used to update decoding error conditions. Based on a received error messages, the encoder will encode a current video NAL units with clean random-access pictures. In the same time will notify the decoder by sending developed ECSEI message that a necessary error concealment action is needed. The evaluation results for the proposed ERSEI algorithm indicate an evident visual quality enhancement under different packet loss rates PLRs (2, 4, 6, 8)%. The (Y-PSNR) gain is (2.554 dB) and (0.995 dB)) when compared to the modified HM16 with pixel copy concealment technique and MCEC, respectively. However, in error-free condition, there is small degradation in visual quality of (-0.608 dB) and (-0.187 dB) for pixel frame copy and (MCEC) methods, respectively. These findings reveal that the ERSEI can be the cornerstone for future studies on utilisation of supplemental enhancement messages in joint encoder-decoder error resilience approach.

Furthermore, the experimental work on the proposed ERSEI algorithm is extended to include evaluation study of computational complexity at both encoder and decoder sides. The reference software HM 16 is used as a benchmark in experimental work. From the achieved results, it can be concluded that the execution processing times are increased by (25.7 %) at encoder side and (37.5 %) at the decoder side. The rate distortion optimisation process at the modified encoder side consumed more processing power than default reference software HM 16. More processing power is spent on allocating encoding bit rate cost in encoding process.

The increase in decoding process comes from employed complex parsing process at frame and video sequence levels which requires high decoding processing time than default reference software HM 16.

It is worth noting, that the ERSEI algorithm apply error concealment or encoding error resilience in adaptive way depending on error corruption level. This adaptive approach helps to keep coding efficiency at the highest level during ERSEI algorithm implementation.

As far as error resilience in video transmission is concerned, we found that ERSEI algorithm is more efficient compared to the state of art error resilience and the default reference software HM16 algorithms.

Furthermore, comparing with ASE complexity performance, the execution time increased (25%) at decoding side and (6.2 %) at encoding side. The results of this performance complexity study indicate that despite the error resilience performance of ERSEI algorithm gives promising results in terms of visual quality and coding efficiency, one issue of the complexity study was that the ERSEI algorithm as well as ASE algorithm are implemented using high level language (C++) which is not practical for computational complexity analysis.

These findings in this piece of work reveal that the ERSEI algorithm can be the cornerstone for future studies on utilisation of supplemental enhancement messages in joint encoder-decoder error resilience approach.

These findings suggest that in general the optional SEI messages can be used efficiently to reduce perceived decoded visual artefacts at H.265|HEVC decoder at slight increase in coding efficiency.

Chapter Six

6 Conclusions and Future Work

In this chapter, Section 6.1 provides general summary for the whole thesis. Section 6.2 presents general conclusions for the conducted research work in this thesis. Section 6.3 presents future work recommendations.

6.1 General Summary

Nowadays, ultra-high definition video coding has been a fascinating subject in the field of modern video communication systems especially in 4K and 8K video resolutions. However, there have been some challenging issues related to the transmission of high-quality video contents in the conventionally available networks. Some of these challenges include limited bandwidth capacity with increasing number of end users besides channel interference and fading impact [162]. Moreover, sending compressed video bitstream using unreliable transmission channels such as wireless channels can result in receiving delays and video packet loses at the decoder [11]. These factors lead to temporal error propagation into future decoded frames (encoded with P or B frames), or spatial error propagation within the frame level. In the worst scenario, such errors lead to th received video bitsream completely undecodable. In this piece of research, we aim to present an efficient error resilience algorithm to reduce the effects of injected various packet loss rates on perceived visual quality at H.265|HEVC decoder side.

This thesis is organised as follows, Chapter Two presents a review of the proposed state of the art error control algorithms in video transmission. Chapter Three describes H.265|HEVC video coding standard highlighting its main video coding tools in addition to technical comparison study with H.264|AVC video coding standard. Furthermore, it presents an evaluation comparison study between H.265|HEVC and H.264|AVC video coding standards in both error-free and error prone environments with various video sequences characteristics. Further, the video quality evaluation platform which used in experimental work is described in details. Additionally, it presents encoding error sensitivity evaluation study for H.265|HEVC coding standard system. In Chapter Four, a proposed adaptive encoding error resilience algorithm is presented. The proposed algorithm called Adaptive Slice Encoding (ASE). This algorithm can be used in low delay video applications without using a feedback channel. The main concept of this algorithm is to extract the most active slices inside the coded bitstream based on the adaptive searching window. The identified active slices are then encoded with intra mode. The algorithm is designed to be compatible with reference software manual (HM16) for H.265|HEVC video coding standard. An evaluation work has been conducted in terms of injecting various packet loss rates, encoding bit rates, and computational complexity. Chapter Five presents interactive encoder-decoder error resilience algorithm. The proposed algorithm called Error Resilience based on Supplemental Enhancement Information (ERSEI) algorithm. In this algorithm, a feedback message status is used to update decoding error conditions. Based on a received error messages, the encoder will encode a current video NAL units with clean random-access pictures. In the same time will notify the decoder by sending developed ECSEI message that a necessary error concealment action is needed. At the decoder side, the video bitstream is parsed for Decoded picture hash and ECSEI messages, and ERSEI will be activated based on both messages. The ERSEI algorithm is evaluated in terms of objective using Y-PSNR metric and subjective using frame by frame quality assessment. Furthermore, computational complexity analysis work has been conducted for testing RESEI performance with default reference software as a benchmark. Both proposed algorithms are designed and optimised to find the best trade-off between a coding efficiency and error resilience performance in both error-free and error-prone environments.

6.2 General Conclusions

This section summarises the experimental work results:

6.2.1 Adaptive Slice Encoding (ASE) Algorithm

The novality of this algorithm lies in automatically selecting the most active frame regions and protecting them against transmission errors at the cost of a tolerable increase in the encoding bit rate overhead and computational encoding and decoding complexity. The proposed work also took into consideration the coding efficiency by subdividing the non-active regions into flat and high textured areas. The saving of the bit budget in non-active areas is also being considered in this study through spending a larger portion of the available bit budget on active frame areas in addition to achieving best trade-off between the coding efficiency and error resilience performance.

The ASE algorithm is evaluated in several simulation scenarios. Firstly, the experimental work was conducted in error- free and error-prone environments by injecting various packet loss rates PLRs ranging (2-18) %. The obtained results revealed that the proposed algorithm yields a Y-PSNR gain of (4.52 dB) over the HM16 reference software, and outperforms the state-of-the-art algorithms ROI and IROI by (2.28 dB) and (1.07 dB), respectively. However, in error-free conditions, the proposed algorithm suffered from Y-PSNR loss gain of (-1.09 dB) compared with default reference software HM16.

With respect to the coding complexity, it was the first study tooks into considerations the computation complexity in encoding error resilience implementation. The encoding and decoding processing time of the tested video sequences are analysed and reported in terms of computational complexity. The processing time results of the proposed algorithm showed that the encoding and decoding time of the proposed work increased by 19% and 11%, respectivily. The increase in processing time came from several factors. These factors include processes: encoding data partitioning in addition to slice boundaries detection and reference sample generations at the encoder side, and parsing on reduntant slices at the decoder side.

The algorithm is further investigated to evaluate the percieved visual quality with different number of the end users using LTE network with shared bandwidth. From the achieved results, it is shown that with high network loads (increasing number of end users), the ASE algorithm more adaptable with high dropped packet percentage than HM16 reference software in same limited bandwidth conditions.

With Start-up video play delay (0.5 second and 1 second), in Long-term Evolution LTE network. The obtained results showed that when the start-up delay increases (0.5 to 1 second) at the decoder, the objective decoded video quality remarkably increases (1 dB on average). One interesting finding is that as start-up delay of the proposed work is within accepted delay for real time applications. The ASE algorithm without implementing feedback update channel can be used in low delay video delivery applciations. Another important finding is that in high PLRs, the percieved visual quality is higher that H.265|HEVC refernce software and other related state of the art algorithms

6.2.2 Error resilience based on Supplemental Enhancement Information (ERSEI) Algorithm

The proposed algorithm employs supplemental enhancement information to encode robust bitstream against channel errors. The proposed algorithm is compared with default reference software of H.265|HEVC encoding version (HM16.06) and state-of-the-art related algorithm called Motion Compensation Error Concealment (MCEC). The evaluation work of the proposed work includes objective and subjective quality evaluations in error-free and error prone conditions. Additionally, the computational complexity at both encoder and decoder sides are evaluated and compared the default reference software. The most obvious experimental findings from the evaluation of ERSEI algorithm are summarised in the following paragraphs. The achieved evaluation results show significant improvements in terms of Y-PSNR metric and frame by frame quality assessment. In terms of objective results with subjecting various PLRs to the encoded videos, the ERSEI achieves Y-PSNR gain of (2.5 dB) compared to the codec reference software and (0.9 dB) compared to MCEC algorithm. However, this increase comes at the cost slight decrease in coding efficiency. In error-free conditions, the Y-PSNR gain is decreased by less than (0.61 dB) compared with reference software.

Moreover, the evaluation work of ERSEI algorithm is further extended to include complexity analyses of ERSEI algorithm performance in terms of computation complexity. The complexity evaluation work includes comparisons study of encoding and decoding execution times between ERSEI, ASE, and default reference software HM 16. From the achieved complexity analysis results, it was found the execution time increase (25.7 %) at the encoder side and (37.5 %) at the decoder side compared with reference software HM 16. Furthermore, comparing with ASE complexity performance, the execution time increased (25%) at decoding side and (6.2 %) at encoding side. The results of this performance complexity study indicate that despite the error resilience performance of ERSEI algorithm gives promising results in terms of visual quality and coding efficiency, one issue of the complexity study was that the ERSEI algorithm as well as ASE algorithm are implemented using high level language (C++) which is not practical for computational complexity analysis.

These findings in this piece of work reveal that the ERSEI algorithm can be the cornerstone for future studies on utilisation of supplemental enhancement messages in joint encoder-decoder error resilience approach.

6.2.3 H.265|HEVC Video Evaluation Platform

Video quality evaluation platform is proposed to support H.265|HEVC video bitstream evaluation. The platform supports subjective and objective video quality measurement methodologies. Furthermore, A proposed platform that supports real and simulated networks is presented.

6.2.4 Video Coding Standard Comparison Study between H.265|HEVC and H.264|AVC Coding Standards

The latest H.264|AVC and H.265|HEVC video coding standard standards in error-free and error-prone environments are evaluated using the same environmental setup. The study calculates the average bit rate saving for different video resolutions from low-resolution QCIF (176x144) resolution up to 4K (3840 x 2160) resolution. The findings indicate that H.265|HEVC coding standard is less efficient in bit rate saving at lower resolution, i.e. QCIF (176 x 144) resolution (38%). The coding standard efficiency increases consistently with increasing video resolution. The maximum bitrate saving when encoding (UHD) (3860 x 2160) resolution can reach up to (63%).

6.2.5 H.265|HEVC Encoding Error Sensitivity Evaluation

This involves evaluating the effect of decoded visual quality using reference software (HM16) for H.265|HEVC. The study includes injecting random bit error rate on encoded motion data and different NAL unit types (VCL and non-VCL NAL units). The results show that the visual quality decreases by (2dB) when injecting 5% of total generated errors on motion prediction data compared to the evenly distributed errors on all video sequence. On the other hand, when targeting only 5% of the total errors on non-VCL units, the visual quality decreases by (8.83 dB). Several conclusions can be extracted from this study. First, the H.265|HEVC coding standard has introduced a new prediction mechanism using advanced motion vector prediction (AMVP) which derives motion data based on several prediction blocks and reference pictures. The highly efficient inter prediction tools make motion data very sensitive to bit errors. A single bit errors twill causes he visual quality to drop dramatically. Second, the network abstraction layer (NAL) units support different network protocols including VCL and non-VCL units. A VCL NAL units contain a video sample representation. Therefore, if there is a bit error in VCL-data, it will not affect the whole decoding process. Non-VCL NAL, on the other hand, contains shared metadata for more than one frame. In non-VCL NAL, if there is a single bit error, the error will propagate to the subsequent decoded frames or the bitstream will be undecodable at the decoder.

6.3 Future Work

Transmitting a highly compressed video with high resolution is still an area of interest in research at present. Improving error resilience with developing video coding standard is very important to achieve the best trade-off between bit rate saving and coding standard error robustness. This is the focus of this study. Some areas have been identified for future improvements.

- Implementing a Gilbert-Elliott model with the proposed algorithm for providing real time quality service estimation. The model will be mainly involved in automatically adjusting the encoding parameters. Future studies on error resilience based on adaptive encoding in H.265|HEVC coding standard are therefore recommended. Further investigations are required to evaluate the impact of ASE algorithm implementation on power consumption on the decoder side.
- Further research is required to reduce computational complexity for ASE and ERSEI algorithms.
- Optimisation of encoded H.265|HEVC bitstream robustness based on cyclic encoding refresh of Supplemental Enhancement information.
- To extend the proposed algorithms to include (spatial and quality) scalability or Multiview coding in H.265|HEVC extension, where a video parameter set (VPS) should be included as a sensitive video data.
- The H.265|HEVC transmission system based on Media Aware Network elements (MANEs) can be enhanced to support temporal scalable bitstream. This might be achieved by utilisation of the active video and sequence parameters in NAL unit headers. The aim of this method is to increase error robustness and decrease computation complexity at the decoding stage; a full processing diagram is demonstrated in Figure 6.1

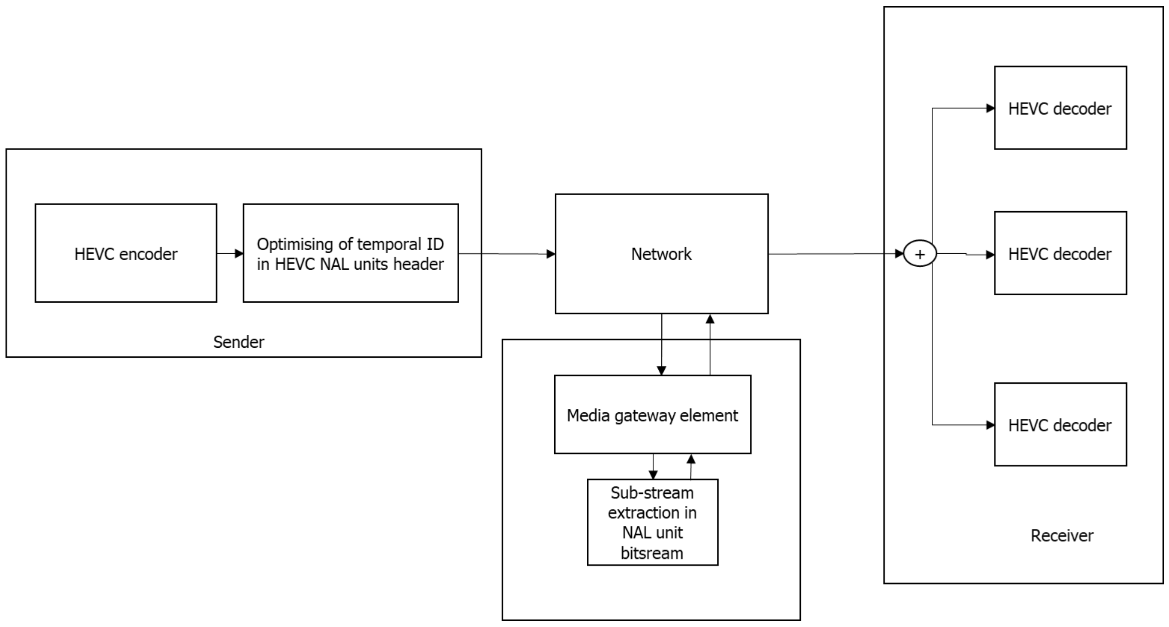

HEVC encoder
Optimising of temporal ID in HEVC NAL units header
Sender
Network
Media gateway element
Sub-stream extraction in NAL unit bitsream
+
HEVC decoder
HEVC decoder
HEVC decoder
Receiver

Figure 6.1 Enhanced H.265|HEVC transmission system

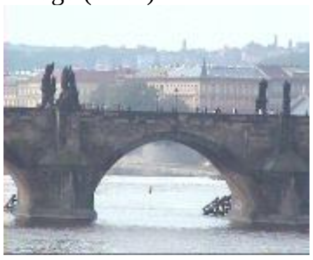
Appendix A: Video test sequences characteristics

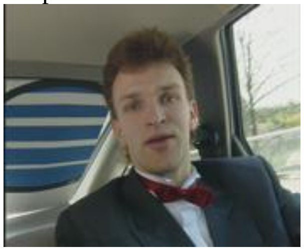
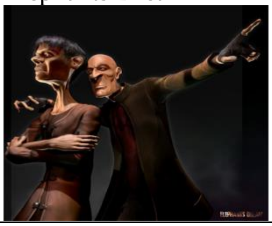
Various standard video test sequences were selected with different resolution and frame rates. All the selected sequences are in raw format with colour space (YUV) and chroma format 4:2:0 using three main components, i.e. Y, Cr, and Cb. There is no header information that exists in video test sequences. All the test video sequences are classified according to their texture video information and motion activity speed. Mainly they divided into two classes:

Class A: video sequence with low texture details and slow-motion activity

Class B: video sequence with high texture details and high-motion activity

Table 9 Characteristics of the test video sequence [163]

No.	Sequence name	Resolution	Number of frame	Class
1	Akiyo	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	A
2	Bridge (close) 	352 x 240 (CIF), (176 x 144) QCIF	2001 frames /(4:3)	A
3	Bridge (far)	352 x 240 (CIF), (176 x 144) QCIF	2101 frames /(4:3)	A

4	Bus	352 x 240 (CIF)	150 frames /(4:3)	B
5	Carphone 	(176 x 144) QCIF	382 frames /(4:3)	B
6	Claire	176 x 144) QCIF	494 frames /(4:3)	A
7	Coastguard	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	B
8	Container	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	A
9	Elephants Dream 	352 x 240 (CIF), (1920x1080)	15691 frames /(4:3)	B

10	Flower	352 x 240 (CIF)	250 frames /(4:3)	A
11	Foreman	352 x 240 (CIF)/30fps, (176 x 144) QCIF	300 frames /(4:3)	B
12	Grandma	(176 x 144) QCIF	870 frames /(4:3)	A
13	Hall Monitor	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	A
14	Highway	352 x 240 (CIF), (176 x 144) QCIF	2000 frames /(4:3)	A
15	Miss America	(176 x 144) QCIF	150 frames /(4:3)	A

16	Mobile	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	A
17	Mother and daughter	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	A
18	News	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	B
19	Paris	352 x 240 (CIF)	1065 frames /(4:3)	A
20	Salesman	(176 x 144) QCIF	449 frames /(4:3)	B
21	Silent	352 x 240 (CIF), (176 x 144) QCIF	300 frames /(4:3)	B
22	Stefan	352 x 240 (CIF)	90 frames /(4:3)	B

23	Suzie	(176 x 144) QCIF	150 frames /(4:3)	B
24	Tempete	352 x 240 (CIF)	260 frames /(4:3)	B
25	Waterfall	352 x 240 (CIF)	260 frames /(4:3)	B
26	Football (b) [164]	352 x 240 (CIF)/30fps, (176 x 144) QCIF/15fps	260 frames /(4:3)	B
30	Ice	352 x 240 (CIF)/30fps, 704x480 (4CIF)/60fps,	480 frames /(4:3)	B

Table 10 High resolution video test sequence [165]

No.	Video sequence name	Resolution/frame rate	Number of frames	remarks
1	**Elephants_dream**	480p (SD), (858 x 480)	15691 frames (16|9)	Class: B
2	**old_town_cross**	720p (1280 x 720), 1080p (1920 x 1080), and 2160p (3860 x 2160)/ (50 fps)	500 frames (16|9)	Class: A Camera recorder: ARRI ArriFlex 765 System for 65mm, 5 perf, film

References

[1] Y. Huo, C. Zhou, J. Jiang, and L. Hanzo, "Historical information aware unequal error protection of scalable HEVC/H.265 Streaming over free space optical channels," *IEEE Access*, vol. PP, no. 99, 2016.

[2] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012-2017," 2013.

[3] D. Webster, "Cisco Visual Networking Index (VNI)," *Glob. Forecast Updat.*, p. 6, 2017.

[4] S. Liu, X. Xu, S. Lei, and K. Jou, "Overview of HEVC extensions on screen content coding," *APSIPA Trans. Signal Inf. Process.*, vol. 4, no. January 2014, pp. 1–12, 2015.

[5] S. Chen and H. Leung, "A temporal approach for improving intra-frame concealment performance in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 3, pp. 422–426, 2009.

[6] M. Wien, *High Efficiency Video Coding: Coding Tools and Specification*. Springer, 2014.

[7] J. Nightingale, Q. Wang, and C. Grecos, "HEVStream: A framework for streaming and evaluation of high efficiency video coding (HEVC) content in loss-prone networks," *IEEE Trans. Consum. Electron.*, vol. 58, no. 2, pp. 404–412, 2012.

[8] B. Bross *et al.*, "HEVC performance and complexity for 4K video," in *2013 IEEE Third International Conference on Consumer Electronics ¿ Berlin (ICCE-Berlin)*, 2013, pp. 44–47.

[9] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[10] R. Yousfi, M. Ben Omor, T. Damak, M. A. Ben Ayed, and N. Masmoudi, "JEM-post HEVC vs. HM-H265/HEVC performance and subjective quality comparison based on QVA metric," *2018 4th Int. Conf. Adv. Technol. Signal Image Process. ATSIP 2018*, pp. 1–4, 2018.

[11] K. E. Psannis, "HEVC in wireless environments," *J. Real-Time Image Process.*, vol. 12, no. 2, pp. 509–516, 2015.

[12] D. Flynn *et al.*, "Overview of the range extensions for the HEVC standard: Tools, profiles, and performance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 4–19, 2016.

[13] M. Usman, X. He, K. M. Lam, M. Xu, S. M. M. Bokhari, and J. Chen, "Frame Interpolation for Cloud-Based Mobile Video Streaming," *IEEE Trans. Multimed.*, vol. 18, no. 5, pp. 831–839, 2016.

[14] "H.262: Information technology - Generic coding of moving pictures and associated audio information.," *TU-T and ISO/IEC JTC*. [Online]. Available: http://www.itu.int/rec/T-REC-H.262. [Accessed: 30-Jul-2015].

[15] Y. Ye and P. Andrivon, "The scalable extensions of HEVC for ultra-high-definition video delivery," *IEEE Multimed.*, vol. 21, no. 3, pp. 58–64, 2014.

[16] I. K. Kim, J. Min, T. Lee, W. J. Han, and J. H. Park, "Block partitioning structure in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1697–1706, 2012.

[17] G. B. T. Wiegand, G.J. Sullivan, "Overview of the H.264/AVC video coding

standard," *Circuits Syst. Video Technol.*, vol. IEEE Trans, 2013.

[18] "Draft revision of recommendation H.261: Video codec for audiovisual services at p × 64 kbit/s," *Signal Process. Image Commun.*, vol. 2, no. 2, pp. 221–239, Aug. 1990.

[19] K. Rijkse, "H. 263: Video coding for low-bit-rate communication," *Commun. Mag. IEEE*, vol. 34, no. 12, pp. 42–45, 1996.

[20] "ISO/IEC 14496-2:1999 - Information technology -- Coding of audio-visual objects -- Part 2: Visual." [Online]. Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=25034. [Accessed: 30-Jul-2015].

[21] "ITU-T H.264." [Online]. Available: http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=6312&lang=en. [Accessed: 30-Jul-2015].

[22] "Recommendation ITU-T H.264: Advanced Video Coding for Generic Audiovisual Services," Tech. Rep., ITU-T, 2003."

[23] T. Von Roden, "H.261 and MPEG1-a comparison," in *Conference Proceedings of the 1996 IEEE Fifteenth Annual International Phoenix Conference on Computers and Communications*, 1996, pp. 65–71.

[24] J. R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards-including high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, 2012.

[25] T. Wiegand, "Overview of the H. 264/AVC video coding standard," *... Syst. Video ...*, vol. 13, no. 7, pp. 560–576, 2003.

[26] G. J. Sullivan, "Improved lossless intra coding for H.264/MPEG-4 AVC," *IEEE Trans. Image Process.*, vol. 15, no. 9, pp. 2610–2615, Sep. 2006.

[27] D. Itu-T, "VIDEO CODEC FOR AUDIOVISUAL SERVICES AT p × 64 kbits: ITU-T Recommendation H.261," *Int'l Telecommun. Union,(May 2, 1996)*, vol. 261, 1996.

[28] J. Ohm and G. J. J. Sullivan, "High Efficiency Video Coding: The Next Frontier in Video Compression," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 152–158, Jan. 2013.

[29] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communications: {A} review," *Proc. IEEE*, vol. 86, no. 5, pp. 974–997, 1998.

[30] Y. Wang and S. Wenger, "Error resilient video coding techniques," *IEEE Signal Process. Mag.*, vol. 17, no. 4, 2000.

[31] R. Talluri, "Error-resilient video coding in the ISO MPEG-4 standard," *IEEE Commun. Mag.*, vol. 36, no. 6, pp. 112–119, Jun. 1998.

[32] B. Girod and N. Farber, "Feedback-based error control for mobile video transmission," *Proc. IEEE*, vol. 87, no. 10, 1999.

[33] M.-T. Sun, "Error resilience Coding," in *Compressed Video Over Networks*, 2000, p. 584.

[34] F. Almasalha, R. Hasimoto-beltran, and A. A. Khokhar, "Partial Encryption of Entropy-Coded Video Compression Using Coupled Chaotic Maps," *Entropy*, no. October, 2014.

[35] N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video*, Prentice-H. Prentice Hall, 1984.

[36] H. Sun and J. Zdepski, "Adaptive error concealment algorithm for MPEG compressed video," in *Applications in Optical Science and Engineering*, 1992, pp. 814–824.

[37] K. Illgner and D. Lappe, “Mobile multimedia communications in a universal telecommunications network,” 1995, pp. 1034–1043.

[38] Y. Wang and Q. Q. F. Zhu, “Error control and concealment for video communication: a review,” *Proc. IEEE,* vol. 86, no. 5, pp. 974–997, 1998.

[39] “ISO/IEC JTC/SC29/WG11 MPEG97/N1646: ‘Description of error resilient core experiments,’ Bristol meeting, April 1997.”

[40] Abdul H. Sadka, *Compressed Video Communications*. Guildford,UK: Wiey, 2002.

[41] J. M. Boyce, “Packet loss resilient transmission of MPEG video over the Internet,” *Signal Process. Image Commun.*, vol. 15, no. 1, pp. 7–24, 1999.

[42] Y. Wang, S. Panwar, S. P. Kim, and H. L. Bertoni, *Scalable video coding with multiscale motion compensation and unequal error protection*. New York: Plenum, 1996.

[43] S. Wenger, “H.264/AVC over IP,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 645–656, 2003.

[44] I. E. G. Richardson, *Video codec design : developing image and video compression systems*. West Sussex, England: Wiley, 2002.

[45] R. Sjoberg *et al.*, “Overview of HEVC high-level syntax and reference picture management,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1858–1870, 2012.

[46] B. Bing, *Next-generation video coding and streaming*, First Edit. JohnWiley & Sons, 2015.

[47] S. K. I. and A. J. P. Pearmain, “Error resilient video coding with priority data classification using H.264 flexible macroblock ordering,” *IET Image Process.*, vol. 1, no. 2, pp. 197–204, 2007.

[48] K. Tan and A. Pearmain, “An improved FMO slice grouping method for error resilience in H.264/AVC,” in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010, pp. 1442–1445.

[49] P. Lambert, W. De Neve, Y. Dhondt, and R. Van De Walle, “Flexible macroblock ordering in H.264/AVC,” *J. Vis. Commun. Image Represent.*, vol. 17, no. 2, pp. 358–375, 2006.

[50] Y. Wang, a R. Reibman, and S. Lin, “Multiple Description Coding for Video Delivery,” *Proc. IEEE*, vol. 93, no. 1, pp. 57–70, 2005.

[51] M. Kazemi, S. Shirmohammadi, and K. H. Sadeghi, “A review of multiple description coding techniques for error-resilient video delivery,” *Multimed. Syst.*, vol. 20, no. 3, pp. 283–309, Jun. 2014.

[52] V. K. Goyal, “Multiple description coding: Compression meets the network,” *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 74–93, 2001.

[53] R. Llados-Bernaus and R. L. Stevenson, “Fixed-length entropy coding for robust video compression,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 6, pp. 745–755, 1998.

[54] T. Algra, “Fast and efficient variable-to-fixed-length coding algorithm,” *Electron. Lett.*, vol. 28, no. 15, p. 1399, 1992.

[55] W. Jiangtao and J. D. Villasenor, “Reversible variable length codes for robust image and video transmission,” *Conf. Rec. Thirty-First Asilomar Conf. Signals, Syst. Comput. (Cat. No.97CB36136)*, vol. 2, pp. 973–979, 1997.

[56] F. Eryurtlu, A. H. Sadka, and A. M. Kondoz, “Error robustness improvement of video codecs with two-way decodable codes,” *IEEE/IET Electron. Lett.*, vol. 33, no. I, p. 558, 1997.

[57] S. Dogan, A. H. Sadka, and A. M. Kondoz, “ERROR-RESILIENT TECHNIQUES FOR VIDEO TRANSMISSION OVER WIRELESS CHANNELS,” *Cent. Commun. Syst. Res. (CCSR), Univ. Surrey*, vol. Guildford, 2000.

[58] G. Gennari and G. a Mian, “A robust H. 264 decoder with error concealment capabilities,” in *European Signal Processing Conference*, 2004, pp. 649–652.

[59] S. S. Hemami and T. Y. Meng, “Transform coded image reconstruction exploiting interblock correlation,” *IEEE Trans. Image Process.*, vol. 4, no. 7, pp. 1023–7, Jan. 1995.

[60] B. Girod and N. Farber, “Feedback-based error control for mobile video transmission,” *Proc. IEEE*, vol. 87, no. 10, 1999.

[61] H. Sun and W. Kwok, “Concealment of damaged block transform coded images using projections onto convex sets,” *IEEE Trans. image Process.*, vol. 4, no. 4, pp. 470–7, Jan. 1995.

[62] W. Y. Kung, C. S. Kim, and C. C. J. Kuo, “Spatial and temporal error concealment techniques for video transmission over noisy channels,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 7, pp. 789–802, 2006.

[63] W. Kwok and H. Sun, “Multi-directional interpolation for spatial error concealment,” *IEEE Trans. Consum. Electron.*, vol. 39, no. 3, pp. 455–460, 1993.

[64] E. P. Bahl and B. Girod, “Special section on wireless video,” *IEEE Communications Magazine, vol. 36*, pp. 92–151, 1998.

[65] D. J. Costello and M. J. Miller, “Automatic-repeat-request error-control schemes,” *IEEE Commun. Mag.*, vol. 22, no. 12, pp. 5–17, Dec. 1984.

[66] Y. Wang, Q.-F. Zhu, and L. Shaw, “Maximally smooth image recovery in transform coding,” *IEEE Trans. Commun.*, vol. 41, no. 10, pp. 1544–1551, 1993.

[67] Hang Liu and M. El Zarki, “Performance of video transport over wireless networks using hybrid ARQ,” in *Proceedings of ICUPC - 5th International Conference on Universal Personal Communications*, vol. 2, pp. 567–571.

[68] H. Xie, P. Narasimhan, R. Yuan, and D. Raychaudhuri, “Data link control protocols for wireless ATM access channels,” in *Proceedings of ICUPC ’95 - 4th IEEE International Conference on Universal Personal Communications*, pp. 753–757.

[69] R. V. Cox and P. Kroon, “Low bit-rate speech coders for multimedia communication,” *IEEE Commun. Mag.*, vol. 34, no. 12, pp. 34–41, 1996.

[70] F. Wang, W. Wu, Y. Lou, and A. Yang, “An adaptive H.264 video protection scheme for video conferencing,” *IEEE Vis. Commun. Image Process.*, pp. 1–4, 2011.

[71] N. Mukawa, H. Kuroda, and T. Matsuoka, “An interframe coding system for video teleconferencing signal transmission at a 1.5 Mbit/s rate,” *IEEE Trans. Commun.*, vol. 32, pp. 280–287, 1984.

[72] “Control Protocol for Multimedia Communication, ITU-T Rec- ommendation H.245, 1996.”

[73] M. Wada, “Selective recovery of video packet loss using error concealment,” *IEEE J. Sel. Areas Commun.*, vol. 7, no. 5, pp. 807–814, Jun. 1989.

[74] P. Haskell and D. Messerschmitt, “Resynchronization of motion compensated video affected by ATM cell loss,” in *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1992, vol. 3, pp. 545–548 vol.3.

[75] B. Girod and N. Farber, “Feedback-based error control for mobile video

transmission," *Proc. IEEE*, vol. 87, no. 10, 1999.

[76] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[77] J. Nightingale, Q. Wang, C. Grecos, S. Member, and S. Goma, "The Impact of Network Impairment on Quality of Experience (QoE) in H . 265 / HEVC Video Streaming," *IEEE Trans. Consum. Electron.*, vol. 60, no. 2, pp. 242–250, May 2014.

[78] W. Zhu, W. Ding, J. Xu, Y. Shi, and B. Yin, "Screen content coding based on HEVC framework," *IEEE Trans. Multimed.*, vol. 16, no. 5, pp. 1316–1326, 2014.

[79] M. Wang, S. Member, K. N. Ngan, H. Li, and S. Member, "An Efficient Frame-Content Based Intra Frame Rate Control for High Effciency Video Coding," *IEEE SIGNAL Process.*, vol. 22, no. 7, pp. 896–900, 2015.

[80] G. Kulupana, D. S. Talagala, H. K. Arachchi, and A. Fernando, "Error resilience aware motion estimation and mode selection for HEVC video transmission," *2016 IEEE Int. Conf. Consum. Electron. ICCE 2016*, pp. 85–86, 2016.

[81] W. li and J. Xu, "Confident Recovery for Lost Motion Vectors in Video Transmission," *Procedia Environ. Sci.*, vol. 11, pp. 263–269, 2011.

[82] H. L. El Zarki, "Adaptive source rate control for real-time wireless video transmission," *Mob. Networks Appl.*, vol. 103, no. 3, pp. 239–248, 1998.

[83] J. Lee and K. Kang, "SVC-aware selective repetition for robust streaming of scalable video," *Wirel. Networks*, vol. 21, no. 1, pp. 115–126, 2015.

[84] "Recommendation ITU-T H.265," 2013. [Online]. Available: http://handle.itu.int/11.1002/1000/11885-en?locatt=format:pdf&auth.

[85] M. Uhrina, M. Malicek, and M. Vaculik, "Subjective Video Quality Assessment of H.265 Compression Standard for Full HD Resolution," *Adv. Electr. Electron. Eng.*, vol. 13, no. 5, pp. 545–551, 2015.

[86] J. Lainema, F. Bossen, W. J. Han, J. Min, and K. Ugur, "Intra coding of the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1792–1801, 2012.

[87] Frank Bossen, D. Flynn, K. Sharman, and K. Sühring, "HM Software Manual," SG16 WP3 and ISO/IEC JTC1/SC29/WG1, 2016.

[88] R. Sjöberg and J. Boyce, "HEVC High-Level Syntax," in *High Efficiency Video Coding (HEVC): Algorithms and Architectures*, 2014, pp. 13–48.

[89] D. R. Tobergte and S. Curtis, "H.265 (HEVC) Bitstream to H.264 (MPEG 4 AVC) bitstream transcoder," THE UNIVERSITY OF TEXAS AT ARLINGTON, 2015.

[90] J. Melorose, R. Perroy, and S. Careas, "RTP Payload Format for H.265/HEVC Video," *Netw. Work. Gr. Internet*, vol. 1, no. May, pp. 1–100, 2015.

[91] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An Effective CU Size Decision Method for HEVC Encoders," *IEEE Trans. Multimed.*, vol. 15, no. 2, pp. 465–470, Feb. 2013.

[92] A. Torres, P. Pi??ol, C. T. Calafate, J. C. Cano, and P. Manzoni, "Evaluating H.265 real-time video flooding quality in highway V2V environments," *IEEE Wirel. Commun. Netw. Conf. WCNC*, pp. 2716–2721, 2014.

[93] M. A. Saleh, H. Hashim, N. Tahir, and E. Hisham, "Review for High Efficiency Video Coding (HEVC)," *IEEE Conf. Syst. Process Control*, no. December, pp. 141–146, 2014.

[94] A. Tabatabai, J. Xu, and O. Nakagami, "Intra_Block Copying Enhancements For

HEVC in Range-Extension (Rext),” US2015/0049813 A1, 19-Feb-2015.

[95] G. Correa, P. Assuncao, and L. Agostini, “Fast coding tree structure decision for HEVC based on classification trees,” *Analog Integr. Circuits Signal Process.*, vol. 87, no. August, pp. 129–139, 2016.

[96] Y. H. Chiu, K. L. Chung, W. N. Yang, C. H. Lin, and Y. H. Huang, “Universal intra coding for arbitrary RGB color filter arrays in HEVC,” *J. Vis. Commun. Image Represent.*, vol. 24, no. 7, pp. 867–884, 2013.

[97] M. P. Sharabayko, O. G. Ponomarev, and R. I. Chernyak, “Intra Compression Efficiency in VP9 and HEVC,” *Appl. Math. Sci.*, vol. 7, no. 137, 2013.

[98] K. E. Psannis and Y. Ishibashi, “Efficient error resilient algorithm for H.264/AVC: Mobility management in wireless video streaming,” *Telecommun. Syst.*, vol. 41, no. 2, pp. 65–76, 2009.

[99] J. Xiantao, “Study on key technologies for video coding standard beyond H . 265 / HEVC,” University of Tokushima, Japan, 2016.

[100] “High efficiency video coding. ITU-T Rec. H.265 (HEVC). http://www.itu.int/rec/T-REC-H. 265/en (2013).”

[101] B. Bross, P. Helle, H. Lakshman, and K. Ugur, “Inter-Picture Prediction in HEVC,” in *High Efficiency Video Coding (HEVC): Algorithms and Architectures*, Springer, 2014, pp. 113–140.

[102] V. Sze, M. Budagavi, and G. J. Sullivan, *High Efficiency Video Coding (HEVC) Algorithms and Architectures*. Switzerland: Springer, 2014.

[103] K. Muller *et al.*, “3D high-efficiency video coding for multi-view video and depth data,” *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3366–3378, 2013.

[104] E. G. Mora, M. Cagnazzo, and F. Dufaux, “AVC to HEVC transcoder based on quadtree limitation,” *Multimed. Tools Appl.*, vol. 76, no. 6, pp. 8991–9015, 2017.

[105] J. Vanne, M. Viitanen, and T. D. Hamalainen, “Efficient Mode Decision Schemes for HEVC Inter Prediction,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 9, pp. 1579–1593, Sep. 2014.

[106] J. Carreira, V. De Silva, E. Ekmekcioglu, A. Kondoz, P. Assuncao, and S. Faria, “Dynamic motion vector refreshing for enhanced error resilience in HEVC,” *Eur. Signal Process. Conf.*, pp. 281–285, 2014.

[107] B. Li, J. Xu, and H. Li, “Parsing robustness in High Efficiency Video Coding - analysis and improvement,” in *2011 Visual Communications and Image Processing (VCIP)*, 2011, pp. 1–4.

[108] A. Norkin *et al.*, “HEVC deblocking filter,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1746–1754, 2012.

[109] K. Yang, S. Wan, Y. Gong, H. R. Wu, and Y. Feng, “A Novel SAO-based Filtering Technique for Reduction in Temporal Flickering Artifacts in H.265/HEVC,” *Circuits, Syst. Signal Process.*, vol. 35, no. 11, pp. 4099–4128, 2016.

[110] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan, “Reducing complexity of HEVC: A deep learning approach,” *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5044–5059, 2018.

[111] F. Bossen, B. Bross, K. Suhring, and D. Flynn, “HEVC complexity and implementation analysis,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, 2013.

[112] S. Ma, S. Wang, S. Wang, L. Zhao, Q. Yu, and W. Gao, “Low complexity rate distortion optimization for HEVC,” *Data Compression Conf. Proc.*, vol. 73, no. 1, pp. 73–82, 2013.

[113] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Complexity scalability for real-time HEVC encoders," *J. Real-Time Image Process.*, pp. 1–16, 2014.

[114] Y. Zhang, S. Kwong, and X. Wang, "Machine Learning-Based Coding Unit Depth Decisions for Flexible Complexity Allocation in High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2225–2238, 2015.

[115] F. Bossen, "Common test conditions and software reference configurations (JCTVC-L1100)," Geneva, m28412, 2013.

[116] T. Zhao, Z. Wang, and S. Kwong, "Flexible Mode Selection and Complexity Allocation in High Efficiency Video Coding," *IEEE J. Sel. Top. Signal Process.*, vol. 7, no. 6, pp. 1135–1144, Dec. 2013.

[117] H. Schwarz, T. Schierl, and D. Marpe, "Chapter 3 Block Structures and Parallelism Features in HEVC," in *High Efficiency Video Coding (HEVC): Algorithms and Architectures*, 2014, pp. 49–90.

[118] M. Zhou, V. Sze, and M. Budagavi, "Parallel tools in HEVC for high-throughput processing," vol. 8499, pp. 849910-849910–13, 2012.

[119] M. Viitanen, J. Vanne, T. D. Hamalainen, M. Gabbouj, and J. Lainema, "Complexity analysis of next-generation HEVC decoder," *IEEE Int. Symp. Circuits Syst.*, pp. 882–885, 2012.

[120] S. Sector and O. F. Itu, "ITU-T Recommendation H.265 High Efficiency Video Coding," *ITU Telecommun. Stand. Sect.*, vol. 265, no. Series H: Audiovisual and Timedeia Systems, p. 609, 2015.

[121] M. Ghanbari, *Standard codecs: Image compression to advanced video coding*, Third. Institution of Engineering and Technology, 2011.

[122] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electron. Lett.*, vol. 44, no. 13, p. 800, 2008.

[123] I. E. G. Richardson, *Video Coding Design Developing Image and Video Compression Systems*, (3)Edition. The Robert Gordon University, UK: John Wiley & Sons Ltd, 2011.

[124] T. K. Tan *et al.*, "Video quality evaluation methodology and verification testing of HEVC compression performance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 76–90, 2016.

[125] "Subjective video quality assessment methods for multimedia applications, interactive test methods for audiovisual communications," ITU-T Study Group 12, P.910, 1996.

[126] "MSU Graphics and Media Lab," 2014. [Online]. Available: http://www.compression.ru/video/quality_measure/vqmt_download.html. [Accessed: 16-Aug-2014].

[127] V. Ukani and J. Bhatia, "Testing Video Transmission in NS2 using the Evalvid Framework," *Open Gurus*, pp. 83–87, 2015.

[128] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid–A Framework for Video Transmission and Quality Evaluation," *13th Int. Conf. TOOLS 2003, Urbana, IL, USA, Sept. 2-5, 2003. Proc.*, no. September, pp. 255–272, 2003.

[129] C. H. Ke, C. K. Shieh, W. S. Hwang, and A. Ziviani, "An evaluation framework for more realistic simulations of MPEG video transmission," *J. Inf. Sci. Eng.*, vol. 24, no. 2, pp. 425–440, 2008.

[130] S. Wenger, "Nal Unit Loss software." JCTVCH0072, JCT-VC Document, 2012.

[131] F. Bossen, D. Flynn, K. Sharman, and K. Sühring, "JCTVC-Software Manual," 16.6, 2015.

[132] K. Sühring, “H.264/AVC JM Reference Software,” 2015. [Online]. Available: http://iphome.hhi.de/suehring/tml/. [Accessed: 30-Sep-2015].

[133] “G. Bjøntegaard, ‘Calculation of average PSNR differences between RD-curves’, ITU-T Q.6/SG16 VCEG 13th Meeting, Document VCEG-M33, Austin, USA, Apr. 2001.”

[134] J. L. Lin, Y. W. Chen, Y. W. Huang, and S. M. Lei, “Motion vector coding in the HEVC Standard,” *IEEE J. Sel. Top. Signal Process.*, vol. 7, no. 6, pp. 957–968, 2013.

[135] L. Song, X. Tang, W. Zhang, X. Yang, and P. Xia, “The SJTU 4K Video Sequence Dataset,” Klagenfurt, Austria, 2013.

[136] D. Grois, D. Marpe, A. Mulayoff, B. Itzhaky, and O. Hadar, “Performance comparison of H.265/MPEG-HEVC, VP9, and H.264/MPEG-AVC encoders,” in *2013 Picture Coding Symposium (PCS)*, 2013, pp. 394–397.

[137] “H.264 : Advanced video coding for generic audiovisual services,” *International Telecommunication Union*. [Online]. Available: http://www.itu.int/rec/T-REC-H.264-201201-S. [Accessed: 01-Aug-2015].

[138] P. Seeling and M. Reisslein, “Video traffic characteristics of modern encoding standards: H.264/AVC with SVC and MVC extensions and H.265/HEVC,” *Sci. World J.*, vol. 2014, no. 3, 2014.

[139] G. J. Sullivan, J. M. Boyce, Y. Chen, J. Ohm, C. A. Segall, and A. Vetro, “Standardized Extensions of High Efficiency Video Coding (HEVC),” *IEEE J. Sel. Top. Signal Process.*, vol. 7, no. 6, pp. 1001–1016, 2013.

[140] J. Xu, R. Joshi, and R. A. Cohen, “Overview of the emerging HEVC screen content coding extension,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 50–62, 2016.

[141] “RTP payload format for scalable video coding. IETF RFC6190. http://tools.ietf.org/html/ rfc6190 (2011). Accessed 14 Jan 2016.”

[142] R. Garcia and H. Kalva, “Subjective evaluation of HEVC and AVC/H.264 in mobile environments,” *IEEE Trans. Consum. Electron.*, vol. 60, no. 1, pp. 116–123, Feb. 2014.

[143] A. T. Hoang and M. Motani, “Cross-layer adaptive transmission: Optimal strategies in fading channels,” *IEEE Trans. Commun.*, vol. 56, no. 5, pp. 799–807, 2008.

[144] H. Chen, C. Zhao, M. T. Sun, and A. Drake, “Adaptive intra-refresh for low-delay error-resilient video coding,” *J. Vis. Commun. Image Represent.*, vol. 31, pp. 294–304, 2015.

[145] H. M. Maung, S. Aramvith, and Y. Miyanaga, “Region-of-interest based error resilient method for HEVC video transmission,” *2015 15th Int. Symp. Commun. Inf. Technol. Isc. 2015*, pp. 241–244, 2016.

[146] H. Maung, S. Aramvith, and Y. Miyanaga, “Improved region-of-interest based rate control for error resilient HEVC framework,” *Int. Conf. Digit. Signal Process. DSP*, pp. 286–290, 2017.

[147] H. M. Hu, B. Li, W. Lin, W. Li, and M. T. Sun, “Region-based rate control for H.264/AVC for low bit-rate applications,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 11, pp. 1564–1576, 2012.

[148] Y. Liu, Z. G. Li, and Y. C. Soh, “Region-of-interest based resource allocation for conversational video communication of H.264/AVC,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 1, pp. 134–139, 2008.

[149] H. Li, Z. Wang, H. Cui, and K. Tang, “An improved ROI-based rate control

algorithm for H.264/AVC," *Int. Conf. Signal Process. Proceedings, ICSP*, vol. 2, 2007.

[150] Y. Liu, Z. G. Li, S. Member, and Y. C. Soh, "A Novel Rate Control Scheme for Low Delay Video Communication of H . 264 / AVC Standard," vol. 17, no. 1, pp. 68–78, 2007.

[151] D. Y. Wang and S. X. Sun, "Region-based rate control and bit allocation for video coding," *Int. Conf. Apperceiving Comput. Intell. Anal.*, vol. 8, no. 1, pp. 147–151, 2008.

[152] H. Song and C.-C. J. Kuo, "A Region-Based H.263+ Codec and Its Rate Control for Low VBR Video," *IEEE Trans. Multimed.*, vol. 6, no. 3, pp. 489–500, Jun. 2004.

[153] G. Ren, P. Li, and G. Wang, "A novel hybrid coarse-to-fine digital image stabilization algorithm," *Inf. Technol. J.*, vol. 9, pp. 1390–1396, 2010.

[154] B. Li, H. Li, L. Li, and J. Zhang, "λ domain Rate Control Algorithm for High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 23, no. 9, pp. 3841–3854, 2014.

[155] A. Fouda *et al.*, "Real-Time Video Streaming over NS3-based Emulated LTE Networks," *Int. J. Electron. Commun. Comput. Technol.*, vol. 4, no. 3, pp. 659–663.

[156] S. S. Krishnan and R. K. Sitaraman, "Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs," *IEEE/ACM Trans. Netw.*, vol. 21, no. 6, pp. 2001–2014, 2013.

[157] H. S. Jung, R. C. Kim, and S. U. Lee, "Error-resilient video coding using long-term memory prediction and feedback channel," *Signal Process. Image Commun.*, vol. 17, no. 8, pp. 597–609, 2002.

[158] W. Xu and Y. Chen, "Error resilience video coding parameters and mechanisms selection with End-to-End rate-distortion analysis at frame level," *Multimed. Tools Appl.*, vol. 75, no. 4, pp. 2347–2366, 2016.

[159] R. Sjöberg and J. Samuelsson, "Absolute Signaling of Reference Pictures," JCTVC-F493, 6th Meeting, Turin, Italy, 2011.

[160] Q. Peng and C. Zhu, "Block-Based Temporal Error Concealment for Video Packet Using Motion Vector Extrapolation," pp. 10–14, 2002.

[161] Y. L. Chang, Y. A. Reznik, Z. Chen, and P. C. Cosman, "Motion compensated error concealment for HEVC based on block-merging and residual energy," in *20th International Packet Video Workshop*, 2013.

[162] Cisco company, "Cisco Global Cloud Index: Forecast and Methodology, 2012–2017," 2013.

[163] Arizona State University and Aalborg university, "Video Trace Library." [Online]. Available: http://trace.eas.asu.edu/. [Accessed: 20-Apr-2019].

[164] *Image Processing Research ECSE Department, Rensselaer Polytechnic Institute*. Troy, NY 12180, USA.

[165] Institute for Telecommunication Sciences (ITS), "Video Quality Research," *3141012-300*, 2008. [Online]. Available: https://media.xiph.org/video/derf/. [Accessed: 17-Mar-2016].