DATA EMBEDDING USING PREDICTIVE SYNTAX ELEMENTS IN SCALABLE CODED VIDEO

PANG LIE LIN

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2021

DATA EMBEDDING USING PREDICTIVE SYNTAX ELEMENTS IN SCALABLE CODED VIDEO

PANG LIE LIN

THESIS SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2021

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: ***PANG LIE LIN***

Registration/Matric No.: ***17042311/1 (WHA 140044)***

Name of Degree: ***DOCTOR OF PHILOSOPHY***

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):

DATA EMBEDDING USING PREDICTIVE SYNTAX ELEMENTS IN SCALABLE CODED VIDEO

Field of Study: ***VIDEO CODING***

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date: 22 July 2021

Subscribed and solemnly declared before,

Witness's Signature Date:

Name:
Designation:

DATA EMBEDDING USING PREDICTIVE SYNTAX ELEMENTS IN SCALABLE CODED VIDEO

ABSTRACT

With the rapid advancement in digital technologies, video has emerged as one of the most effective and popular communication media. While videos are increasingly adopted for various purposes including entertainment, dashboard camera recording, and recently education, many issues related to video arise, including unauthorized or illegal use of video content, tampered video content, and packet loss due to network congestion or bandwidth fluctuation. As a result, various proposals are put forward to manage videos, and one of them is data embedding. Essentially, data embedding inserts data into a video to serve a specific purpose, including proof of ownership via watermark, covert communication in steganography, as well as error concealment and authentication via fragile watermark. This work focuses on embedding data into video coded in the format of Scalable High Efficiency Video Coding (SHVC), which plays an important role in adaptive video streaming applications. The scalability feature is essential, notably when considering the heterogeneity in transmission and decoding devices. However, the underlying architecture of this scalable video coding standard differs from the previous scalable video coding standards. A study is therefore required to explore the possibility to embed data into SHVC encoded video. Specifically, the predictive syntax elements in the SHVC compressed domain are analyzed. To increase embedding capacity, payload bits are embedded in all scalable coded layers. Conceptually, coding a video frame using more (smaller) prediction blocks leads to more embedding opportunities due to the availability of the syntax elements. Therefore, an adjustable control parameter is introduced to guide the quad-tree partitioning process to permit more split blocks. The proposed technique achieves encouraging payload with slight bit rate overhead. When a predictive syntax element is utilized for data embedding, the encoder calculates and encodes the prediction error caused by data embedding as a portion of the prediction residual and reconstruct the video as similar as possible to the original video. Hence, the manipulation of predictive syntax elements will not cause drift-error. In view of this, both intra- and inter-coded blocks are jointly utilized for data embedding without compromising the perceptual quality. During encoding, the prediction modes and partitioning depth are assessed by rate distortion optimizer, and only the one with the best rate distortion cost is adopted for coding. A data embedding framework named CUPSEED, which automatically selects different prediction elements for data embedding, is then proposed to manage multiple data embedding venues and coding layers. The proposed framework outperforms existing data embedding techniques in the SHVC compressed domain in term of embedding capacity. It manages to achieve higher payload while preserving the perceptual quality with minimal bit rate variation.

Keywords: data embedding, SHVC, predictive syntax element, threshold controlled, CUPSEED

PEMBENAMAN DATA DI DALAM VIDEO BERKOD BOLEH SKALA DENGAN MENGGUNAKAN ELEMEN SINTAKS RAMALAN

ABSTRAK

Dengan kemajuan pesat dalam teknologi digital, video telah muncul sebagai salah satu meida komunikasi yang paling berkesan dan popular. Dengan video yang semakin digunakan untuk pelbagai tujuan termasuk hiburan, rakaman kamera papan pemuka, dan baru-baru ini Pendidikan, banyak isu yang berkaitan dengan video telah timbul, termasuk penggunaan kandungan video tanpa kebenaran/ menyalahi undang-undang, perubahan ke atas kandungan video, dan kehilangan paket yang disebabkan oleh kesesakan rangkaian atau jalur lebar yang tidak stabil. Justeru itu, pelbagai cadangan telah dikemukakan untuk menguruskan video, dan salah satunya adalah pembenaman data. Data boleh dibenam ke dalam video untuk mencapai tujuan seperti menjadi sebagai bukti pemilikan melalui tera air, komunikasi secara sulit, penyembuhan kesilapan dan pengesahan video menggunakan tera air yang rapuh. Kerja ini memberi tumpuan kepada membenamkan data ke dalam video yang dikodkan dalam format Pengekodan Video Kecekapan Tinggi Berskala (SHVC), yang memainkan peranan penting dalam aplikasi penstriman video penyesuaian. Ciri-ciri kebolehskalaan dalam SHVC amat penting memandangkan penghantaran video ke pelbagai jenis peranti penghantaran dan pengekodan. Walau bagaimanapun, seni bina asas standard video berkod boleh skala ini berbeza daripada standard video berkod boleh skala sebelumnya. Oleh itu, kajian diperlukan untuk meneroka kemungkinan untuk membenamkan data ke dalam video yang dikodkan dengan SHVC. Khususnya, elemen sintaks ramalan dalam domain mampatan SHVC dianalisi. Demi meningkatkan muatan, bit muatan dibenam ke dalam semua lapisan yang dikodkan secara boleh skala. Pengekodan bingkai video menggunakan lebih banyak blok ramalan yang kecil boleh meningkatkan peluang untuk membenam data ke dalam elemen sintaks. Oleh itu, parameter kawalan yang boleh dilaras disarankan untuk membimbing proses pembahagian quad-pokok untuk membenarkan lebih banyak blok kecil dikodkan. Teknik yang disarankan ini dapat mencapai muatan yang menggalakkan dengan sedikit peningkatan dalam overhed kadar bit. Apabila elemen sintaks ramalan digunakan untuk pembenaman data, pengekod mengira dan mengekod ralat ramalan yang disebabkan oleh pembenaman data sebagai sebahagian daripada baki ramalan dan membina semula video supaya video ini mencapai keserupaan dengan video asal yang mungkin. Oleh yang demikian, manipulasi elemen-elemen sintaks ramalan tidak akan menyebabkan kesilapan hanyut. Memandangkan ini, kedua-dua blok ramalan secara intra- dan antara-bingkai digunakan bersama untuk pembenaman data tanpa menjejaskan kualiti perseptual. Semasa pengekodan, mod ramalan dan kedalaman pembahagian dinilai oleh pengoptimum herotan kadar, dan hanya mod ramalan dan kedalaman pembahagian yang mempunyai kos herotan dengan kadar yang terbaik akan digunakan untuk pengekodan. Rangka kerja pembenaman data bernama CUPSEED, telah dicadangkan untuk memilih and menguruskan pelbagai elemen sintaks ramalan yang berbeza daripada semua lapisan pengekodan secara automatik untuk pembenaman data. Rangka kerja yang dicadangkan ini telah mencapai muatan yang lebih tinggi berbandingkan dengan teknik pembenaman data yang sedia ada dalam domain mampat SHVC sementara mengekalkan kualiti video dan variasi kadar bit yang minimum.

Kata kunci: pembenaman data, SHVC, elemen sintaks ramalan, kawalan ambang, CUPSEED

ACKNOWLEDGEMENTS

When I started with my research journey after a gap of 20 years, I was possessed with the fear and fright of whether I would be able to cope with my responsibilities, daily tasks and the research.

I am extremely grateful to my supervisor Associate Professor Wong Kok Sheik for his patient, effort, and consistent support. His generous guidance, help, valuable comments, feedback, and suggestions toward improving my research work in these years are much appreciated. The enthusiasm he has for research has motivated me throughout the tough times of my pursuit of Ph.D.

My sincere thanks to my supervisor Dr. Nizam for all his support, assistance, comments and guidance in thesis writing.

My thanks are extended to those who have rendered their supports in my PhD research journey especially the reading group members from MSPIH and Monash University, Malaysia campus.

I would like to express my gratitude to my parents for their unconditional support and love. I special thank with love to my sisters and brothers for all kind of supports they have provided.

Lastly, to my little girls Jia Yan, Jia Xuan and Jia Sheng, appreciate your understanding and caring through the years. You all are my wonderful girls who always be independent to ease me task and enable me to focus on my research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

A_0	:	bottom left.
A_1	:	left.
B_0	:	top right.
B_1	:	top right.
B_2	:	top left.
C_0	:	bottom right.
C_1	:	center.
D_2	:	$d = 2$.
D_4	:	partitioning depth for 4×4 block.
D_i	:	partitioning depth level i.
M	:	payload data.
N	:	the length of payload data.
RDC	:	Rate Distortion Cost.
V	:	original video.
V'	:	processed video.
Ψ	:	the set of all possible intra and inter prediction modes.
Υ	:	threshold.
Υ_0	:	$\Upsilon = 0\%$.
Υ_5	:	$\Upsilon = 5\%$.
Υ_{10}	:	$\Upsilon = 10\%$.
γ	:	embedding cost.
κ_Ω	:	First key utilized to select/skip blocks for data embedding.
κ_ϕ	:	Second key utilized to select/skip blocks for data embedding.
λ	:	Lagrange multiplier.
$\mathcal{D}$	:	distortion between the original input and reconstructed signals.
$\mathcal{P}_d$	:	RDC for block at partitioning depth level d.
$\mathcal{R}$	:	the number of bits spent on coding.
μ	:	mean intensity of a given video frame.
ψ	:	coding mode applicable to a coding block.
ρ	:	the number of embedded payload bits.
σ	:	standard deviation of all the pixel values of a video frame.
τ	:	threshold.
ϑ	:	intra prediction mode.
ξ	:	threshold to limit the number of bits to be embedded.

b_e	:	the number of bits spent on coding the processed video sequences.
b_o	:	the number of bits spent on coding the original video sequences.
c	:	current layer.
d	:	partitioning depth level.
f	:	input frame.
f'	:	input frame.
h	:	height of a video.
k	:	reference layer.
m	:	payload bit.
n_d	:	the total number of split blocks at partitioning depth level d.
p	:	parity bit.
w	:	width of a video.
3G	:	Third Generation.
ABV	:	above.
AI	:	All Intra.
ALL	:	IPM+MVP+MVD+MRG+BSZ.
AMVP	:	Advanced Motion Vector Prediction.
AV1	:	AOMedia Video 1.
AVC	:	Advanced Video Coding.
AVI	:	Audio Video Interleave.
BD-BR	:	Bjøntegaard Delta of bit rate.
BD-PSNR	:	Bjøntegaard Delta of PSNR.
BL	:	base layer.
BR_m	:	bit rate after embedding with payload bits.
BR_o	:	bit rate before embedding with payload bits.
BRV	:	Bit Rate Variation.
BSZ	:	Block Structure.
CB	:	Coding Block.
Cb	:	blue chrominance component.
CD-ROM	:	compact disc read-only memory.
CGS	:	Color Gamut Scalability.
CPU	:	central processing unit.
Cr	:	red chrominance component.
CTB	:	Coding Tree Block.
CTU	:	Coding Tree Unit.
CU	:	Coding Unit.
CUPSEED	:	Combined Use of Predictive Syntax Elements to Embed Data.
CurrBlk	:	Current Block.

dB	:	decibels.
DCT	:	Discrete Cosine Transform.
DPB	:	Decoded Picture Buffer.
DST	:	Discrete Sine Transform.
DVD	:	Digital Versatile Disc.
EL	:	enhancement layer.
fps	:	frames per second.
GB	:	gigabyte.
GHz	:	gigahertz.
GOP	:	Group of Pictures.
HD	:	High Definition.
HDR	:	High Dynamic Range.
HDTV	:	High Definition Television.
HEVC	:	High Efficiency Video Coding.
HVS	:	human visual system.
IEC	:	International Electrotechnical Commission.
ILR	:	Inter-layer Reference.
IoT	:	Internet of Things.
IPM	:	intra prediction mode.
ISO	:	International Standards Organization.
ITU-T	:	International Telecommunication Union-Telecommunication Standardization Sector.
IVG	:	IPM+MVP+MVD+MRG.
JCT-VC	:	Joint Collaborative Team on Video Coding.
JVT	:	Joint Video Team.
kbps	:	kilobits per second.
LDB	:	Low Delay B.
LDP	:	Low Delay P.
LFT	:	left.
LSB	:	Least Significant Bit.
MB	:	Macroblock.
Mbps	:	Megabytes per second.
MFM	:	Motion Field Mapping.
MPEG	:	Motion Picture Expert Group.
MPEG-1	:	Motion Picture Expert Group Phase 1.
MPEG-2	:	Motion Picture Expert Group Phase 2.
MPEG-3	:	Motion Picture Expert Group Phase 3.
MPEG-4	:	Motion Picture Expert Group Phase 4.
MPM	:	Most Probable Mode.
MSE	:	Mean Squared Error.
MV	:	Motion Vector.

MVA	:	MVP+MVD.
MVD	:	Motion Vector Difference.
MVG	:	MVP+MVD+MRG.
MVP	:	Motion Vector Predictor.
NTSC	:	National Television Standards Committee.
ORI	:	Original.
PAL	:	Phase Alternating Line.
PB	:	Prediction Block.
PC	:	personal computer.
PCM	:	Pulse Code Modulation.
POC	:	Picture Order Count.
PSE	:	Predictive Syntax Element.
PSNR	:	Peak Signal-to-Noise Ratio.
PU	:	Prediction Unit.
QP	:	Quantization Parameter.
RAM	:	random-access memory.
RDO	:	Rate Distortion Optimizer.
RGB	:	red green blue.
RPL	:	Reference Picture List.
SC	:	SubCommittee.
SD	:	Standard Definition.
SHM	:	SHVC Test Model.
SHVC	:	Scalable High Efficiency Video Coding.
SNR	:	Signal-to-Noise Ratio.
SSIM	:	Structural Similarity Index.
SVC	:	Scalable Video Coding.
TB	:	Transform Block.
TMVP	:	Temporal Motion Vector Predictor.
TU	:	Transform Unit.
TV	:	Television.
U	:	blue – luminance.
V	:	red – luminance.
VCEG	:	Video Coding Experts Group.
VoD	:	Video-on-Demand.
WG	:	Working Group.
Y	:	luminance component.

CHAPTER 1: INTRODUCTION

Nowadays, we rely on video technology for various purposes, including dashcam recording/surveillance video as evidence in the event of an incident, movie streaming for entertainment, endoscopic video for health care, to name a few. As a result, several issues associated with video arise, including ineffective search of video content, unauthorized utilization, tampered/forged video content and packet loss. Data embedding is typically utilized as one of the solutions to address the aforementioned issues. This chapter presents a brief introduction to data embedding, followed by an overview of SHVC (Boyce, Ye, Chen, & Ramasubramonian, 2016), the scalable extensions of the High Efficiency Video Coding (HEVC) standard (Sullivan et al., 2012). Then, research motivation, problem statements, research questions, objectives and scopes of this research are described. Next, the limitations and contributions of this research are summarized. Lastly, an outline of the organization of this thesis is presented.

1.1 Introduction to Data Embedding

Data embedding is a mechanism which inserts invisible or inaudible information (message) into digital object (carrier medium or host signal) such as text, image, audio and video without introducing perceptual distortion to the host signal. The inserted information can only be extracted from the host signal by using the bespoke extraction algorithm which is known to the authorized user. The main requirement of the design of a data embedding algorithm is that the semantics of the host signal must be preserved. In addition, the algorithm should have the ability to manage the trade-off among several criteria, including: imperceptibility of the inserted information, perceptual quality, bit rate (or file size) variation, and payload capacity that can be carried by the host signal. Data embedding techniques are considered as imperceptible (invisible) when the data is perceptually imperceptible to human eyes. Good imperceptibility also suggests high fidelity of the output content, where the original content and its processed counterpart (which contains embedded data) are perceptually similar.

On one hand, video has become an effective communication medium. It continues to gain popularity with the rise of social media. It is accessible to anyone with internet access. Besides that, the Internet of Things (IoT) connects network-enabled cameras with other devices and IP-based systems allow users to monitor, access and control devices /objects remotely. As a result, several issues associated with video arise, including ineffective search of video content, unauthorized utilization, tampered/forged video content and packet loss. However, as more videos are in circulation, it is essential to have different approaches to manage and protect the video contents as well as resolving the piracy issues. Therefore, this research focuses on data embedding, which can be the solution for applications such as error concealment, hyper-linking in video, steganography, tracking copyrighted videos, identifying tampered videos, and other general administrative purposes. Some of the common applications of data embedding are described as follow:

1. Error concealment: Error concealment based on data embedding techniques (D. Xu, Wang, & Shi, 2014; Usman et al., 2019) are designed to reduce playback interruption caused by abrupt bandwidth changes or packet losses when transmitting video over an error-prone or congested network. Missing data can be retrieved by utilizing the correlation of the predicted and redundant information in temporally and / or spatially collocated blocks in the video by using data embedding technique.
2. Hyper-linking in video: Video hyperlinking (Hao, Ngo, & Huet, 2020) assists in exploring large video repositories more efficiently. A hyper-linked video combines other videos by using a non-linear information structure, which allows a user to make choices based on the content of the video as well as to navigate between videos

and other hypermedia elements based on the user's interests. To serve this purpose, relevant information can be embedded into a video.

3. Steganography: Steganography is the art and science of hiding information into digital media for covert communication so that the hidden information is imperceptible to everyone except the intended recipient (Fridrich, 2009). The processed video (embedded with secret data) must preserve the perceptual quality and exhibit similar statistics with respect to the original (unprocessed) content. In addition, a steganographic technique must also allow it to carry large payload capacity.
4. Watermarking: Watermarking serves a number of purposes which includes copyright protection, intellectual property rights protection and ownership identification (Buhari, Ling, Baskaran, & Wong, 2016). The basic requirement of digital watermarking is that when the carrier signal is copied or transferred, the watermark must also be carried along. The inserted watermark must be robust against a variety of attacks aiming to destroy or remove the watermark while preserving the usability of the host. In the case of invisible watermarking, the embedded watermark should be imperceptible. On the other hand, for the case of blind watermarking, the embedded watermark can be extracted without the need to refer to the original unprocessed content.
5. Authentication: Authentication is a process to check the integrity and originality of a host content. It is an important application because videos are shared and transmitted across heterogeneous network environments. They are vulnerable to various forms of attack (e.g., trimming, cropping, re-compressing) due to the availability of powerful computer and video editing tools (Muchmore, 2021). Hence, a video needs to be authenticated to ensure its source is trustworthy and its content can be verified to be genuine. When realizing authentication by using data embedding, the forged

content/region is identified by validating the authentication information embedded into the host content. In addition, it often offers the ability to detect and localize tampered region (Tew, Wong, Phan, & Ngan, 2018).

6. Video annotation and management: Data annotation can facilitate the processes involved in various application, including machine learning, information searching, retrieval, and categorization (Yulin Wang, 2004). For instance, hash codes are inserted into a video for efficient searching and retrieval purposes. As one of the fastest growing technologies, artificial intelligence (AI) and machine learning requires a huge volume of data to train the machine learning model of interest. Ground truth, or more generally, metadata, can be inserted into a video as the training data to assist in automated computer vision-based applications such as object tracking and localization. There are also security concerns over adversarial videos aiming to corrupt the training process (Jiang, Ma, Chen, Bailey, & Jiang, 2019), which can be addressed by data embedding to ensure that only genuine training videos are utilized. In addition, information such as comment, reference, index, tags (e.g., time, location and activity) can also be embedded to a video to enrich the content and to improve the efficiency of video management.

1.2 Introduction to SHVC

HEVC and its extensions were developed by the Joint Collaborative Team on Video Coding (JCT-VC) of ISO/International Electrotechnical Commission (IEC) Joint Technical Committee 1 SC 29/WG 11 MPEG and International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) Q6/16 Video Coding Experts Group (VCEG) (Boyce et al., 2016). HEVC is reported to save 50% of bit rate as compared to H.264 while delivering the same perceptual video quality. HEVC has enabled a variety of applications and solutions, including digital storage media, video streaming, video conferencing, and television broadcasting. To cater for more application scenarios, the extensions of HEVC were explored and one of them is SHVC. Specifically, the SHVC is the scalable extension of HEVC for delivering high resolution content to a mix of clients of different display resolutions and computational power, transmitted through unknown network condition and different transmission bandwidths. SHVC provides support in spatial, Signal-to-Noise Ratio (SNR), bit depth, temporal, hybrid codec and color gamut scalability. Scalability refers to the property where a video bit stream can be partially decoded by discarding parts of the bit stream with a graceful degradation in video quality according to the condition of transmission environment, capabilities of the receiving device as well as bit rate, format and power adaptation in heterogeneous network environments. With the SHVC codec, a high definition video can be encoded in multiple-layers, which consists of a base layer (BL) and multiple enhancement layers (ELs). The receivers can select and decode different number of layers of the transmitted video which match the receiver's characteristics and in accordance with the processing constraints and limitation of both the network and output devices. In this research, SHVC is considered as the host signal to embed data, which can be utilized to serve multiple purposes as detailed in Section 1.1.

1.3 Research Motivation

Although existing research has been carried out on data embedding, most studies in the field focused on earlier video coding standards. Specifically, among the existing literature, there is a lack of data embedding algorithm designed for SHVC. Therefore, this research investigates into different syntax elements in the SHVC coding structure for data embedding purposes. This research is motivated by the following:

1. In recent years, the digital revolution in smartphone and telecommunication has

brought numerous changes in our daily life. Video recording and sharing have gained popularity and become more feasible, where video has become an effective communication medium. Particularly, video of higher resolution such as 4K needs to be managed and transmitted in a more efficient way. In addition, the recent pandemic due to the situation of Covid-19 has further accelerated the adoptation of video conferencing technology for online meeting to ensure business continuity. Besides that, virtual classrooms are conducted for teaching and learning purposes and more lecture/tutorial videos are recorded and shared to make teaching and learning possible while teachers and students are physically apart. As video becomes an increasingly important tool for communication, productivity and now education, there is increasing concern over the problems related to video. Some of the common problems include tampered video content, unauthorized/illegal use of video content, and packet loss due to network congestion or bandwidth fluctuation. On the other hand, the computer or cloud administrator may need additional information to handle encrypted video (e.g., transcode or compress the encrypted video), linking related videos, annotation/labelling, managing ownership as well as other legal aspects of the video, to name a few of the tasks. Hence, there is an urgent need for new approaches to address these problems and effectively manage huge volume of video files. For the aforementioned purposes, data embedding, which inserts some data into the video, has gained increased research interest.

2. Nowadays, drone, surveillance and dashcam video recording have been widely deployed for monitoring, preventing and tracing crimes or nuisances purposes. These videos play a crucial role as evidence of wrong-doing. However, there are doubts concerning the reliability of the recordings and whether or not the video recordings can be used as evidence in legal procedures. Therefore, the originality

and integrity of these video recordings are of the highest priorities. In addition, it is useful to have some fast-tracking and control mechanisms to facilitate the monitoring and searching processes. These requirements have motivated researchers to design and develop video-based data embedding schemes for integrity verification and efficient searching/retrieval purposes.

3. SHVC encodes a high definition video into a single bit stream with multiple resolutions or qualities. It emerges as an efficient video coding solution for adaptive video streaming applications. The scalability feature is essential, notably when considering the heterogeneity in transmission environment and decoding devices. Many applications such as Video-on-Demand (VoD) streaming, broadcasting applications, smart surveillance and monitoring system have benefited from this technology. It allows adaptive video streaming according to the available network bandwidth, decoding and display capabilities. For example, it allows users at different places to monitor an area in real-time at different display resolution or frame rate depending on the network condition and decoding devices constraints. Although there are various data embedding methods, there are new opportunities in SHVC. The design architecture of SHVC is based on HEVC and it enables inter layer reference picture as prediction references for EL (i.e., reference layer can be fully reconstructed). This differs from the architecture of the previous standards, namely, Motion Picture Expert Group (MPEG) Phase 1 (MPEG-1), MPEG-2, MPEG-4, H.264/AVC (Advanced Video Coding), and its scalable extension, H.264/SVC (Scalable Video Coding) which does not have reference layer fully reconstructed (i.e., spatial intra-picture prediction in the reference layer cannot be predicted from the corresponding reconstructed inter-picture prediction coded spatial neighboring blocks). Hence, it is not straightforward to adopt the existing data embedding algorithm to the SHVC standard. Among the

existing literature, limited data embedding techniques designed for scalable coded video, especially none for the recent finalized SHVC standard, has motivated the research to develop a data embedding technique using SHVC encoded video.

1.4 Problem Statements

According to a 2019 report by an American networking equipment company Sandvine, video contributes to more than 60% of the internet traffic for delivering data from internet service providers to consumers (Cullen, 2019). The increased demand for video streaming over the internet has brought many issues that continue to draw researchers' attention, which include the following:

1. Format dependent design: Data embedding technique depends heavily on the standard adopted to encode the content. The earlier scalable video coding standard (i.e., H.264/SVC) adopts a single-loop coding strategy, which requires only partial decoding and reconstruction of reference layers for decoding frame for higher coding layer. In contrast, SHVC adopts multi-loop decoding structure, which differs from H.264/SVC. On one hand, this multi-loop coding structure approach improves the rate distortion performance, which only requires high level syntax modifications to the encoder and decoder in all scalable coded layers. On the other hand, this approach needs to retain the predictive information in all reference layers for inter layer prediction. A full reconstruction of video frames in all layers has increased the computational complexity of SHVC as compared to the previous video coding standards. Although some researches have been performed on data embedding in scalable coded video, the underlying architecture of SHVC differs from the previous scalable video coding standard. Therefore, when a new format is published, new data embedding techniques are required. A study is hence required to investigate

into the potential venues that can be utilized to embed data in SHVC.

2. Payload: For different multi-disciplinary applications of data embedding, the payload requirement usually varies. Applications such as watermarking, authentication and fingerprinting may require relatively lower embedding capacity. In contrast, higher payload capacity is required for applications such as error concealment, integrated video content management, steganography and video annotation. With the ever-increasing video resolution such as 4K, the efficiency of the video content search and retrieval system is particularly important for managing a huge volume of video files each of large size, which requires high embedding capacity. A varying embedding capacity requirement (i.e., varying size payloads) is yet to be considered, which possess the challenge of defining a global control parameter to allocate rooms for embedding capacity. Therefore, it is necessary to find a solution to increase embedding capacity according to the payload requirements of the applications in SHVC. However, embedding capacity has been mostly restricted to the availability of the syntax elements utilized for data embedding in a video.

3. Balancing trade-off: An increased embedding payload usually causes more quality distortion, which is noticeable, and vice versa. In particular, this distortion is not desired for applications dealing with highly sensitive data (e.g., diagnostic medical, military and evidence for legal prove). On the other hand, SHVC aims to reduce the video file size so that less bandwidth and storage are required. However, data embedding might cause bit stream size increment. Therefore, the proposed data embedding technique is required to consider not only the embedding capacity, but also the increase of file size so that a balanced trade-off can be found. A balancing trade-off among embedding payload, perceptual quality and bit stream size increment is crucial for applications such as error concealment, integrated video

content management, video hyper-linking, steganography, and video annotation.

1.5 Research Questions

Based on the issues highlighted in Section 1.4, this research aims to address the following research questions:

RQ1. What are the potential venues that can be utilized to embed data in SHVC coded video?

RQ2. How to increase embedding capacity according to the payload requirements of the applications?

RQ3. How to maintain minimum distortion and bit rate overhead with a better rate-distortion performance for data embedding purpose?

1.6 Research Objectives

This research work aims to design and develop data embedding techniques in SHVC that fulfill the following features: (a) maintain SHVC format compliant, (b) achieve high payload, (c) maintain video perceptual quality, and minimize bit rate overhead.

The corresponding research objectives of this thesis can be summarized as follows:

1. To investigate into potential venues in SHVC coding structure for data embedding purposes, focusing on their characteristics and impact to ensure that bit stream after data embedding is still format compliant when they are perturbed for data embedding purposes;
2. To exploit the multi-layer coding mechanism, block partitioning structure and predictive syntax elements to increase the embedding capacity while maintaining perceptual quality of the video, and;

3. To create an integrated framework for managing data embedding in multiple predictive syntax elements, and to assess the embedding cost for each predictive syntax element so that it can automatically select different predictive syntax elements to achieve optimal payload and image quality with minimum bit rate variation.

1.7 Research Scopes and Limitations

The scope of data embedding is wide in general. In order to conduct research efficiently and achieve the research objectives within the approved time frame, some restrictions are inevitable and they are summarized below:

1. To explore data embedding opportunities using predictive syntax elements in the compressed domain of SHVC, good programming skill is required for manipulating the SHVC reference software (i.e., SHM-12.0). Due to the complexity of the scalable coding standard, it requires long hours of reading and editing to understand the architectural design and constraints imposed by SHVC.
2. Six video sequences with resolution 1280×720 and 832×480 coded in two spatial layers are considered for the encoding/decoding tests using different configuration settings. Due to large file size, they require large memory and hard disk space to process and store the processed videos.
3. The experiments are conducted by using a PC with 2.10 GHz CPU and 8GB of RAM running on a 64-bit Windows 10 operating system. The SHVC video encoding process is time consuming. For example, $2\times$ spatial scalability for a resolution of 832×480 takes more than 24 hours to encode a video of 4 seconds at 50 frames per second (fps), i.e., a total of 200 frames. This requires a powerful machine to process the entire video sequence when using various quantization parameter (QP) settings within a reasonable time frame. Hence, the length of the video sequence is

set to 200 frames (which is the maximum length for one of the sequence - *BlueSky*, rounded to hundreds).

4. This research does not cover the implementation of the proposed data embedding techniques for deployment in actual applications.

1.8 Research Contributions

This research makes the following contributions:

1. Design a threshold-controlled embedding technique to achieve a trade-off between embedding capacity (payload) and bit rate variation;
2. Realize high embedding capacity by the combined use of intra and inter predictive syntax elements without compromising the video quality, and;
3. Create an integrated data embedding framework which selects and manages different predictive syntax elements for data embedding in scalable coded video.

1.9 Thesis Organization

This thesis is structured into eight chapters as shown in the following.

- Chapter 1 introduces the subject matter and background of problem(s) being studied.
- Chapter 2 encompasses a review of the literature related to the video coding standard and data embedding techniques in video. In addition, the key video coding features in the SHVC standard are highlighted in this chapter.
- Chapter 3 describes the research methodology applied in this research.
- Chapter 4 presents data embedding technique using motion vector predictor in spatial scalable coded video.
- Chapter 5 proposes an association between merge mode and payload bits in SHVC compressed video.

- Chapter 6 presents the utilization of the intra and inter prediction modes, as well as threshold-controlled block splitting for data embedding.
- Chapter 7 puts forward a data embedding framework to jointly utilize multiple predictive syntax elements in order to achieve high payload.
- Chapter 8 provides a brief summary of this research study and illustrates the significance of this study. Besides that, suggestions are put forward for further research.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

In this chapter, an overview of the development of video coding standards is presented. The coding tools and features of HEVC and SHVC are then discussed and compared to the previous standards, namely H.264/AVC and SVC. After that, YUV color space is briefly introduced to ease the discussion of the coding features. Next, the state-of-the-art data embedding methods in encoded video (compressed domain) are reviewed. Subsequently, the limitation of data embedding in video is succinctly identified and to seek for opportunities to realize data embedding in this scalable coded video. Lastly, the knowledge gaps for data embedding in SHVC is discussed.

2.2 Overview of Video Coding Standards

Video technology has evolved from analog to digital, where many of the basic concepts of video coding such as motion estimation and compensation, transformation and entropy coding were developed in the 1970s and 1980s. The first digital video coding standard H.120 was developed in 1984. MPEG-1 and MPEG-2 were standardized in early 1990s. MPEG-4 was then developed in the late 1990s after H.263 standardization. H.264/AVC and its scalable extension, SVC, were published in 2003 while Google released the VP8 and VP9 video coding format in 2008 and 2012, respectively. In 2013, HEVC standard was published. The SHVC standards was then finalized in October 2014. In addition, other video coding formats include Audio Video Interleave (AVI), Theora, and AOMedia Video 1 (AV1) were also developed over the years. These digital video coding standards have been developed to cover a wide range of applications, and they achieve encouraging compression. The timeline of these video coding standards is shown in Figure 2.1.

The following sections review some of these video coding standards in details.

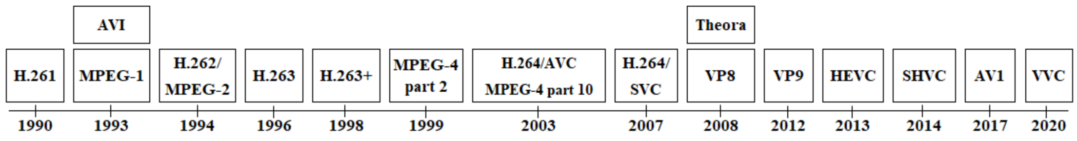


Figure 2.1: Timeline for various video coding standards.

2.2.1 MPEG-1 (H.261)

H.261 is the first widespread practical video coding standard developed by ITU-T in early 1990s. MPEG-1 was then built on H.261 to get better quality. It supports compression of video with 352 × 288-pixel resolution for PAL video at 25 fps and 352 × 240-pixel resolution for NTSC video at 30 fps. It compresses a video into a data stream of 1.5 Mbps (Westwater & Furht, 1997). It is mainly intended for stored interactive video applications in CD-ROM. However, it only supports progressive scan (non-interlaced) picture. In this standard, motion vectors (MVs) are coded losslessly.

2.2.2 MPEG-2 (H.262)

MPEG-2 video coding standard was jointly developed by the MPEG of the ISO and IEC in year 1994. It serves as an extension of MPEG-1 video to support interlaced video coding. It is widely used for the transmission of standard definition (SD) and high definition (HD) TV signals over satellite, cable, and terrestrial emission, and for DVD storage. It supports compression of broadcast television (704 × 576 at 30 fps) and HDTV (1920 × 1152 at 60 fps) (Westwater & Furht, 1997). MPEG-3 was originally designed for HDTV standard but the standard was subsequently merged into MPEG-2.

2.2.3 H.264/AVC (MPEG-4 part 10)

H.264/AVC video coding standard was developed by the Joint Video Team (JVT) consisting of experts from ITU-T VCEG of ITU-T and ISO/IEC MPEG in 2003 (Wiegand, Sullivan, Bjontegaard, & Luthra, 2003). It significantly improves coding efficiency in comparison to the previous video coding standards. It provides flexibility for effective use of applications over a wide variety of networks and systems. It supports HDTV broadcasting, video streaming (e.g., YouTube), video conferencing over fixed and wireless networks, and over different transport protocols, HD-DVD and Blu-Ray disc video format, etc. It is the most commonly used format for the recording, compression, and distribution of video content.

2.2.4 H.264/SVC

H.264/SVC is the scalable extension of the H.264/AVC standard. It encodes a high-quality video bit stream into one or more subset of bit streams, i.e., one BL and multiple ELs. The BL provides the essential information, while the ELs preserve detailed information for rendering at a higher resolution. It provides a mean of readily adapting encoded bit stream to meet decoding device constraints and network condition to allow graceful degradation in lossy transmission environments as well as bit rate, format, and power adaptation. H.264/SVC supports temporal, spatial and quality scalability (Schwarz, Marpe, & Wiegand, 2007). In spatial and temporal scalable coded video, subsets of the bit streams represent the source content with a reduced picture size (spatial resolution) or frame rate (temporal resolution), respectively. In a quality (fidelity or SNR) scalable coded video, the sub stream provides the same spatio–temporal resolution as the entire bit stream, but with a lower quality.

2.2.5 HEVC (MPEG-H Part 2, H.265)

HEVC was designed as a successor to the H.264/AVC by the JCT-VC, which is a partnership between the ITU-T VCEG and the ISO/IEC MPEG standardization organizations in year 2013. HEVC is more complex than its prior standards, and it is designed based on parallel processing architectures. It achieves a compression ratio up to 50% in comparison

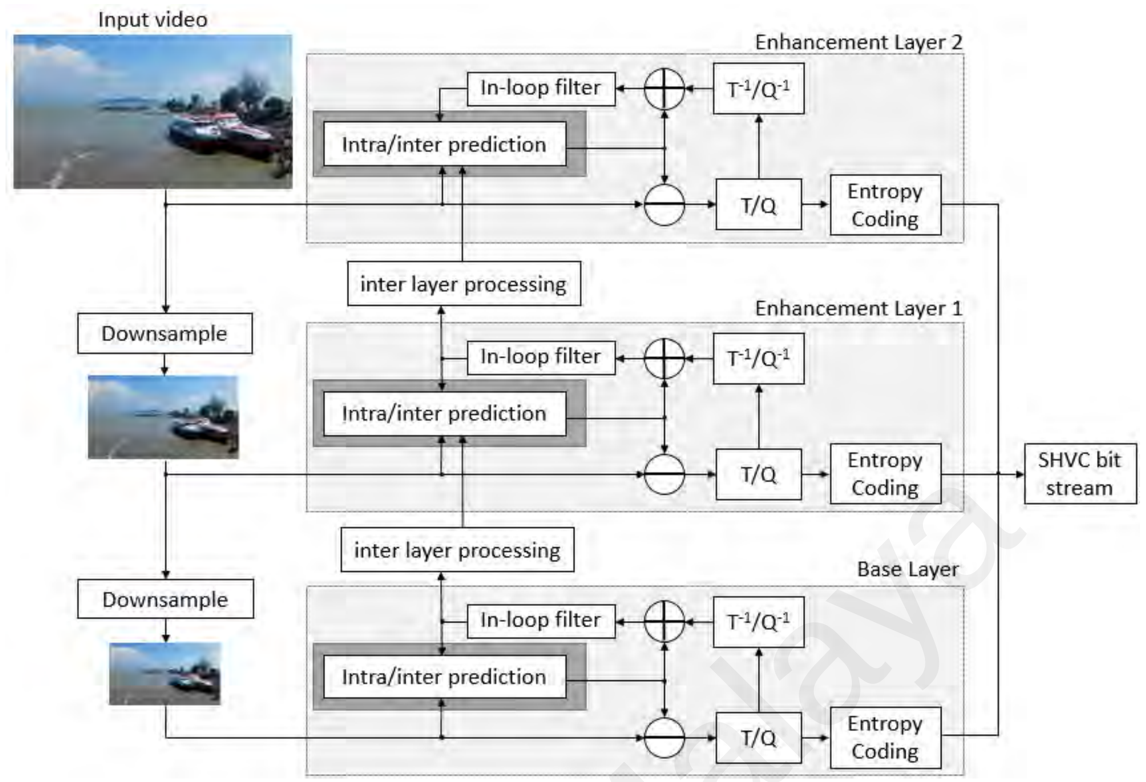


Figure 2.2: Simplified block diagram of a spatial scalability encoding structure of SHVC.

to the H.264/AVC standard while maintaining the same perceptual quality, and it supports high resolution videos such as ultra HDTV up to 8k with frame rates up to 120 fps and video with high dynamic range (HDR) (Sze, Budagavi, & Sullivan, 2014). It is widely utilized for applications which include broadcast of HDTV signals over various transmission systems, video content acquisition and editing systems, camcorders, security applications, internet and mobile network video, Blu-ray discs, and real-time conversational applications such as video chat, video conferencing, and telepresence systems.

2.2.6 Scalable High Efficiency Video Coding (SHVC)

SHVC is an extension of the HEVC standard, which was finalized in October 2014. In addition to the temporal scalability included in the first version of HEVC, SHVC provides support for spatial, SNR, bit depth, hybrid codec and color gamut scalability functionalities, as well as combinations of any of these. The SHVC architecture has been designed as a high-level syntax-only changes to allow the reuse of existing decoder components (Boyce et al., 2016). Figure 2.2 depicts the spatial scalability encoding structure for 2 ELs. In spatial scalability, a picture is represented in different spatial resolutions. Specifically, during encoding, SHVC downsamples each frame from an input (i.e., high resolution) video to produce frames with progressively decreasing resolution. The lowest resolution frame serves as the BL, and it is encoded based on the HEVC standard. This encoded BL is then decoded and enlarged (through some image processing techniques) to match the next resolution considered during the up-sampling process. The differences between the decoded-enlarged and the original frames are coded as the residual information in an EL, and this process is repeated until the full resolution frame is coded. During decoding, BL provides the lowest resolution video, while an EL, when decoded and added onto the enlarged BL, provides a video of higher resolution. In this layered approach, the spatial scalable extension allows the same video content with different resolutions to be coded into a single bit stream as illustrated in Figure 2.2.

Table 2.1 summarizes the features of the aforementioned video coding standards. Since the color representation used by SHVC is YUV, the following section briefly describes YUV color space.

2.3 YUV Color Space

YUV color space is more efficient in representing visual perception, and it can be coded at reduced bandwidth as compared to RGB color space, which is defined as a combination of pure red, green, and blue lights. Therefore YUV is generally used in video processing. The YUV color space comprises of 3 components: luminance (Y) and two chrominance (U and V) components. Y represents brightness and can be computed as a weighted sum of red, green and blue components (i.e., $Y = 0.299 \times R + 0.587 \times G + 0.114 \times B$), while chrominance components represent colors and are formed by subtracting luminance from blue and from red (i.e., $U = 0.492 \times (B - Y)$ and $V = 0.877 \times (R - Y)$. YUV format is

Table 2.1: The features of various video coding standards.

Video coding standards	Features
MPEG-1 (H.261) In 1993	Support compression for 352 × 288 for PAL video at 25 fps, and 352 x 240 for NTSC video at 30 fps; Compress a video into a data stream of 1.5 Mbps; Developed for storage on CD-ROM; Support video conferencing over ISDN; Support only progressive scan picture; Lossless MV.
MPEG-2 (H.262) In 1994	Support interlaced video coding; Support SD, HDTV and video on DVD;
H.263/H.263+ 1996/1998	Improve quality of H.261 at lower bit rate; Support video conference and telephony over 3G wireless network.
H.264/AVC (MPEG-4 part 10) In 2003	Support compression of video with 704 × 576 at 30 fps; Support HDTV broadcasting, video streaming and conferencing; Support HD-DVD and Blu-Ray discs up to 4k.
H.264/SVC In 2007	Encode multiple bit streams in a file; Support temporal, spatial and SNR scalability; Adapt encoded video stream to meet output device constraints and network condition.
HEVC (MPEG-H Part 2, H.265) In 2013	50% bit rate saving for same video quality compare to H.264/AVC; Support parallel processing and temporal scalability; Apply more prediction modes and block sizes; Support Ultra HDTV and HDR video up to 8k at 120 fps; Support Blu-ray discs, and real-time conversational applications.
SHVC In 2014	Support temporal, spatial, SNR, bit depth, hybrid codec, and color gamut scalability; Syntax design is common to all layers; Impose high-level syntax changes only.

used in the compressed video due to the physical characteristics of human visual system (HVS). Human eyes are more sensitive to brightness as compared to color. Therefore, reducing the sampling of chroma (i.e., video-encoded chrominance) can basically keep the similar visual effect. YUV 4:4:4 format has equal sample for each component. The chroma can be adequately sampled at half the sample rate of the luma (i.e., video-encoded luminance) without being noticed by the HVS. Therefore, YUV 4:2:2 and 4:2:0 formats are generally used. YUV 4:2:2 means the chrominance components are sampled horizontally at half the rate of the luminance while YUV 4:2:0 means the Cb (blue chrominance) and Cr (red chrominance) components are sub-sampled at a factor of 2 in the vertical horizontal directions.

2.4 Coding Tools and Features of HEVC and SHVC

The SHVC architecture is designed based on the core coding tools in the HEVC specification with additional scalability features and syntax elements (Boyce et al., 2016). Here, the coding tools and features of HEVC and SHVC are described and compared to the previous standards, i.e., H.264/AVC and H.264/SVC, respectively.

2.4.1 Coding Tools and Features of HEVC

In addition to the features inherited from H.264/AVC standard, HEVC contains more advanced coding tools and features to improve coding and compression efficiency. The video coding employs block-based spatial and temporal prediction, followed by transform coding of the residual and entropy coding of quantized transform coefficients and other coding parameters. Some of the coding tools and features of HEVC are highlighted as follows.

2.4.1.1 Coding Structure

One of the features contributing to the high compression efficiency of HEVC and its extensions is the quad-tree-based variable block size coding structure, which is recursively generated by following the quad-tree structure. First, the largest coding block size is expanded to 64×64 in HEVC as compared to 16×16 in H.264/AVC. Specifically, each frame is partitioned into multiple blocks called coding tree units (CTUs). A CTU consists of a $N \times N$ luma coding tree block (CTB) and the corresponding $(N/2) \times (N/2)$ chroma CTBs. The value of N can be 16, 32, or 64. A CTU is conceptually corresponding to a Macroblock (MB), which consists of a 16×16 luma block and the corresponding chroma blocks in video coding standards prior to HEVC. Each CTU is then split into multiple coding units (CUs). A CU consists of a luma coding block (CB) and the corresponding chroma CBs, together with the associated syntax. The size of a CB can range from the CTB

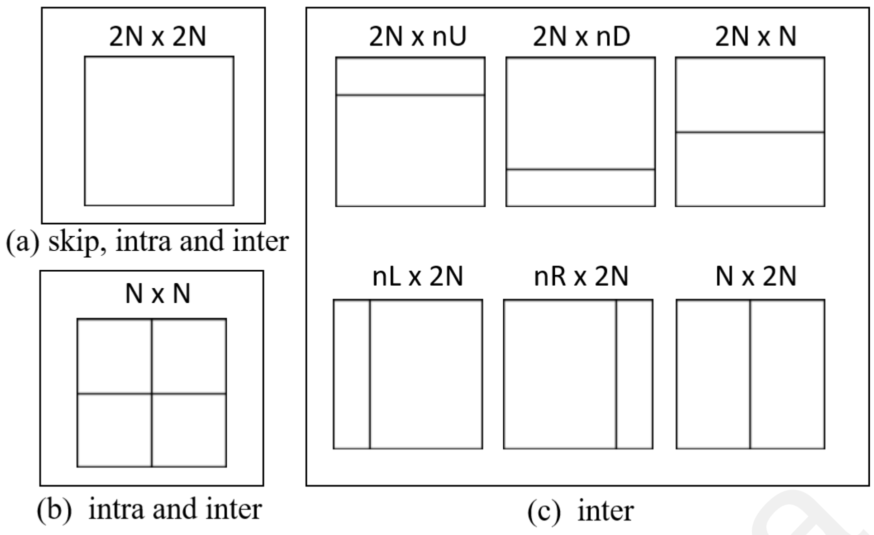


Figure 2.3: Block partitioning structure in HEVC.

size to a minimum size (8 × 8 luma block or larger) as specified by a syntax element. Each CU in turn may contain multiple prediction units (PUs), and a tree of transform units (TUs). A PU consists of a luma prediction block (PB) and two chroma PBs, together with the associated syntax elements. The size of PU may range from 4 × 4 to 32 × 32 pixels for intra prediction, while the size ranges from 8 × 4 or 4 × 8 to 64 × 64 for motion-compensation. The prediction residual is coded as one or multiple smaller transform blocks (TBs) with size ranges from 32 × 32 down to 4 × 4 in a separate quad-tree structure to compress the energy of prediction residual. Similar to H.264/AVC, integer discrete cosine transform (DCT) is employed by HEVC, except for the 4 × 4 intra picture prediction residuals, which uses an integer approximation of the discrete sine transform (DST).

The block splitting types in HEVC is illustrated in Figure 2.3. Similar to prior standards, each CU in HEVC can be classified into three categories: skipped CU, inter coded CU, and intra coded CU. As shown in the figure, only $2N \times 2N$ is allowed for the skipped CU. Two block splitting types, i.e., $2N \times 2N$ and $N \times N$ are supported for the intra coded CU, while all block splitting types are supported for inter coded CU.

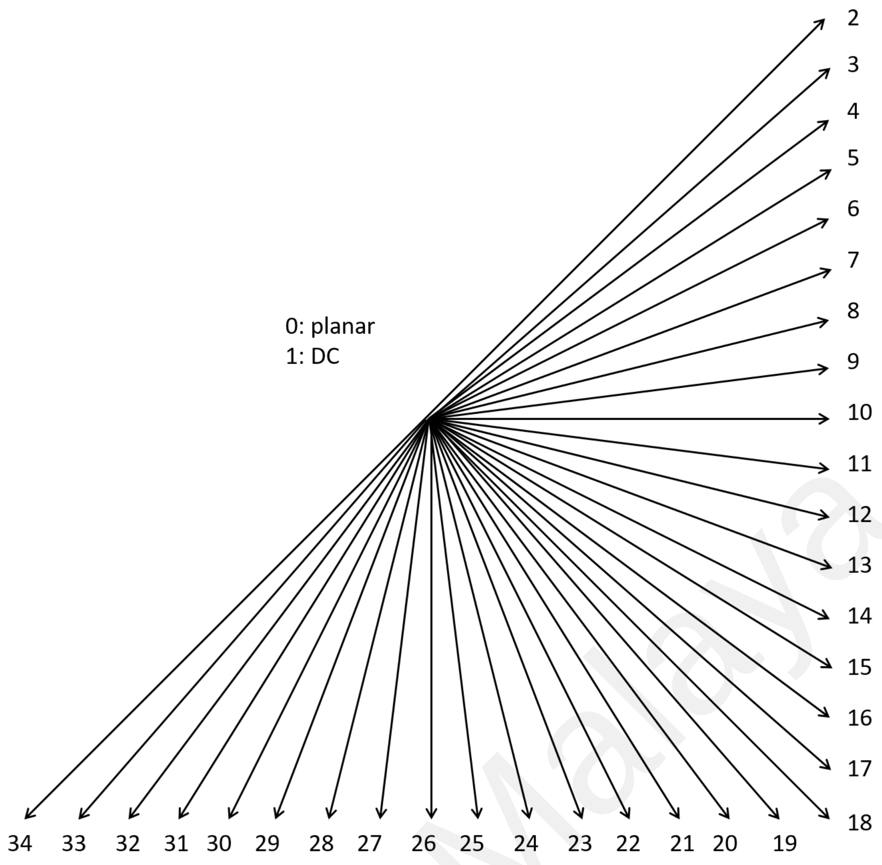


Figure 2.4: Intra prediction modes. Reproduced from (Sullivan et al., 2012).

2.4.1.2 Intra-picture Prediction

One of the breakthroughs in video coding technology is the introduction of more precise and efficient prediction techniques. Within a video frame, neighboring blocks usually have similar textures that are highly correlated. Intra-picture prediction exploits the spatial redundancy within a frame. During intra-picture prediction, the current block is predicted by using the decoded spatial-predicted information of its neighboring blocks. For luma channel, HEVC supports up to 33 directional prediction modes in order to improve intra-picture prediction performance as compared to H.264/AVC, which only utilizes 8 intra directional prediction mode besides planar and DC prediction. The planar mode of HEVC was improved in order to preserve continuities along the block edges. It is derived from the average values of two linear predictions using four corner reference blocks to prevent discontinuities along the block boundaries. DC mode is derived from the average value of reference blocks, and it is efficient for predicting a flat surface. The intra prediction modes of HEVC is illustrated in Figure 2.4. Three most probable modes (MPMs) are derived based on the modes of the neighboring blocks to predict the intra prediction mode instead of a single most probable mode in H.264/AVC. Specifically, intra prediction mode of the one above (ABV) and the one to the left (LFT) of the current PU are included in MPM if these PUs are available and coded by using intra prediction mode. If a neighboring PU is not coded as intra or it is coded by using pulse code modulation (PCM) mode, the PU is coded in DC mode. When the above neighboring PU originates from a different CTU, ABV is set to DC mode.

During encoding, the RD optimizer selects the best intra prediction mode among the candidates. Here, the best refers to the one coded with the least rate distortion cost (RDC). A flag is used to specify whether the selected intra prediction mode is an element of MPMs. If the intra prediction mode matches one of the elements in MPMs, the flag is set to 1 and only the index in MPM is encoded. Otherwise, the flag is set to 0 and a 5-bit fixed length code is utilized to encode the intra prediction mode.

2.4.1.3 Inter-picture Prediction

Within a video frame, adjacent blocks and co-located blocks in adjacent frames (i.e., used as reference picture) are likely corresponding to the same moving object with similar motion. Therefore, the motion information of its adjacent blocks and/or co-located blocks in adjacent frames can be used to remove temporal redundancy. Similar to the previous video coding standards, HEVC employs motion compensated prediction as the main technique for removing temporal redundancy among adjacent blocks or frames. However, HEVC is more complex than the previous video coding standards with the increase of inter prediction modes and the range of block sizes. To increase the efficiency and

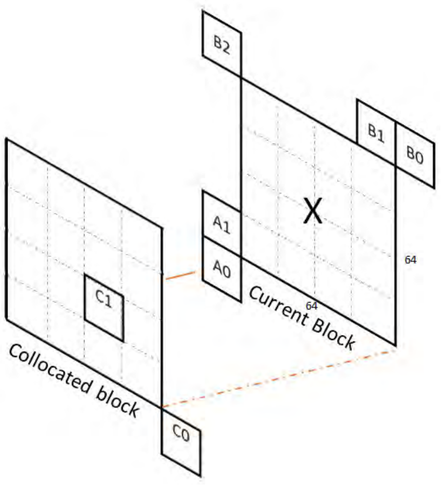


Figure 2.5: MVP candidates (Pang & Wong, 2019).

coding performance of MV prediction, two new coding tools, i.e., advanced motion vector prediction (AMVP) and merge mode are introduced in the coding standard.

During AMVP, motion vector predictor (MVP) candidates are derived from adjacent PBs (spatial) and/or a co-located PB (temporal) in the reference picture, depending on availability. In order to reduce bit rate required to code the predicted MV, the index of the best MVP candidate, i.e., the one having the optimal cost, is selected for encoding. Then, motion vector difference (MVD) is calculated as $MVD_i = MV_i - MVP_i$, where $i \in \{x, y\}$ for the horizontal (x) and vertical (y) components of MV, respectively. Essentially, MVD captures the difference between the MV of the best MVP candidate and the predicted MV of the current block. Then, both MVP index of the best candidate and MVD are coded and transmitted to the decoder together with the residual information. In H.264/AVC, the MV components are differentially coded by using either median or directional prediction from neighboring blocks. The MVP is derived from the medium value of MVs of the spatially adjacent blocks, i.e., block on left, top and top right and/or co-located block from adjacent frame without any RDC evaluation and no MVP is signaled to the decoder (Wiegand et al., 2003).

Quad-tree structure is used for block based partitioning, which allows a video be coded by using sub-blocks of various sizes. However, one of the drawback of quad-tree splitting is that it results in too many small partitioning blocks, which leads to redundant signaling and ineffective borders. This drawback is addressed by the concept of *block merging* (Sze et al., 2014). Adjacent blocks usually contain the same moving object or they have similar motion background. It is likely that these blocks can be predicted by using the same MV. Hence these blocks can be merged together for collective consideration and predicted by using a single MV. When a block is merged to its neighboring block, its MV is derived from its neighboring blocks.

HEVC includes a merge mode to derive motion information from spatially and/or temporally adjacent blocks and/or reference frame. Merge mode is introduced so that a merged region shares the same motion information. Subsequently, less bits are required for signaling the prediction information. Here, motion information is obtained directly from the previously encoded blocks and no MVD information is signalled to the decoder which leads to great saving on bit rate. The merge mode is conceptually similar to the skip-mode and direct-mode in H.264/AVC. However, there are two important differences. First, it transmits index of the merge candidate which is used to derive the MV. Second, it explicitly identifies the reference picture list (RPL) and reference picture index, whereas the direct mode in H.264/AVC assumes that these RPL and reference picture index have some predefined values. The merge mode is similar to the MVP candidate (which comprises of five spatial and two temporal candidates) as depicted in Figure 2.5. Merge mode improves the coding efficiency and save up to 20% for the class E sequences (video conferencing content with the static background) (Sze et al., 2014). Besides that, it also has caused an increased amount of blocks with zero quantized residual which are efficiently coded by using skip CUs (Helle et al., 2012). Note that skip-mode is a special type of merge mode, where the only difference is that skip-mode has zero MV.

2.4.2 Coding Tools and Features of SHVC

In addition to the coding tools from HEVC, SHVC encoder uses inter-layer prediction to improve coding efficiency. The decoded picture at the lower layer is upsampled and the corresponding texture and motion prediction are up-scaled, where they are in turn utilized as a reference picture in predicting ELs. Besides that, SHVC supports a richer set of scalability, which include bit depth, color gamut scalability (CGS) and the hybrid codec scalability as well as the combination in addition to temporal, spatial, SNR inherited from SVC. (Boyce et al., 2016).

The major technical difference between H.264/SVC and SHVC is the multi-loop structure adopted in SHVC that makes predictive information in all the reference layers available for inter layer prediction. The multi-loop design reduces the implementation complexity and improves coding efficiency at the cost of higher computations and memory accesses as well as encoding complexity. In contrast, H.264/SVC adopts a single-loop decoding strategy which requires only partial decoding and reconstruction of reference layers for full decoding picture in upper layer of an H.264/SVC bitstream. Furthermore, H.264/SVC EL MB syntax and decoding process differs from a single layer H.264. A single-layer H.264 codec needs rigorous modifications to handle features introduced in H.264/SVC. Instead, SHVC adopts a coding architecture that only requires high-level syntax changes to the single-layer HEVC standard to accommodate the new features introduced in SHVC.

2.4.2.1 Inter-Layer Texture Prediction

In spatially scalable video coding, video is coded into multiple layers, including one BL at the lowest resolution, and one or more ELs with higher resolutions. The spatial co-located blocks in different layers correspond to the same area of a frame. Hence, the BL (or a lower layer) of a scalable coded video can be used as a reference layer for predicting the video frame in EL. Specifically, a PU in EL can be predicted from predicted information of the co-located PU in the reference layer. In order to enable inter-layer texture prediction, a reference index corresponding to the upsampled reference layer picture is inserted into the EL reference picture list (RPL) and is marked as a long-term reference picture (i.e., reference pictures that have been scaled according to the resolution difference). The upsampled picture can then be utilized as a reference picture similar to inter-picture prediction.

2.4.2.2 Inter-Layer Motion Prediction

SHVC predicts motion information for PUs in EL using the prediction information from a co-located block or frame coded at a lower resolution. The upsampled motion information from the reference layer is inserted into the EL RPL during inter picture prediction in the EL. The predictive information such as prediction modes, reference indices, MVs and reference picture order counts (POCs) of the co-located PU are required for the inter-layer motion prediction. Here, the motion field mapping (MFM) process is performed to obtain the MV of the inter-layer reference picture as a collocated video frame for temporal motion vector predictor (TMVP) derivation based on the compressed motion field of the lower resolution reference layer picture such that the mapped MVs from the reference layer can be used to predict the MVs in the EL. The MFM enables TMVP on inter layer reference (ILR) without low level modification and reduces the memory required for inter-layer motion prediction. Inter-layer prediction provides better video quality as compared to inter picture prediction in certain situation. For example, when temporal distance is large, the reconstructed video in BL is relatively high quality, and it offers predicted video frame with better quality. To explore inter-layer redundancy, the MV between a PU of the current EL picture and its co-located block in the ILR picture is determined from the motion estimation while the MVP is obtained by the AMVP process. When ILR picture is selected as the reference picture of the EL, the MVs pointing to the

ILR picture is set to 0 to improve prediction efficiency. The computationally expensive operations, including fractional sample interpolation and motion estimation processes, can be skipped when the current PU makes reference to the ILR picture for prediction. Furthermore, this could significantly reduce the encoding and decoding complexity.

2.5 Data Embedding in Video Compressed Domain

The design of data embedding process depends heavily on the properties of the host video as well as its application. Various data embedding techniques have been proposed for the video such as AVI (McGowan, 2004), MPEG-2, MPEG-4, H.264/AVC and HEVC for different purposes, including watermarking, steganography, error correction, as well as additional managerial features (Tew & Wong, 2014). The embedding process can be taken place at various stages, including during spatial and temporal prediction, transformation, quantization, as well as entropy encoding. The state-of-the-art data embedding in video compressed domain includes the techniques as detailed in the following subsections.

2.5.1 Intra Prediction Mode-based Data Embedding

Hu et al. (Hu, Zhang, & Su, 2007) propose to modify the prediction directions in 4×4 intra prediction modes in H.264/AVC video, where specified mapping rules are set to associate prediction modes and data bits. Yang et el. (G. Yang, Li, He, & Kang, 2011) propose to embed two bits into every three I4-blocks intra-prediction modes of H.264/AVC coded video by using matrix encoding. However, the number of intra directional prediction modes in H.264/AVC are limited and any changes to the directional prediction mode affect the encoding efficiency. The number of intra prediction modes has increased from 9 in H.264/AVC to 35 in HEVC, leading to more opportunities for data embedding. Hence, the utilization of intra prediction mode for data embedding in HEVC results in smaller bit rate variation and lesser video quality distortion as compared to H.264/AVC. For example, Wang et al. (J. Wang, Wang, Xu, & Li, 2015) and Xu et al. (J. Xu, Wang, Huang, Wang, & Xu, 2015) map data bits to intra-prediction mode of the smallest blocks in HEVC coded video. Sheng et al. then utilize the difference of two consecutive intra prediction modes (i.e., directions) or a pair of continuous planar or DC mode of the smallest PBs to embed data (Sheng, Wang, Pei, & Wang, 2016). The bit rate overhead for I-frame is slightly higher for (Sheng et al., 2016) due to the increase in the prediction error when the prediction direction is shifted away from the optimal direction to realize 3-bit embedding. Mareen et al. propose to reduce the distortion due to data embedding by either increasing or reducing the intra prediction mode by one so that the processed video is imperceptible to HVS (Mareen, De Praeter, Van Wallendael, & Lambert, 2018). Recently, Saberi et al. select the smallest PUs of intra-coded block for data embedding into the intra prediction modes. In their work, a random function is utilized in selecting CTU as data carrier (Saberi, Ramezanpour, & Khorsand, 2020). The perceptual quality degradation for the intra prediction mode is less than 0.2 dB (Hu et al., 2007; Mareen et al., 2018; Saberi et al., 2020), which is negligible. On the other hand, the bit rate overhead caused by using the intra prediction mode to embed data in (Mareen et al., 2018) is capped at 4%. In general, modifying an intra prediction directional mode in HEVC to its adjacent directional mode has minimal impact on the video quality, although the file size is slightly increased. Therefore, this technique has been widely exploited to embed data in the literature. In most proposed techniques, only the smallest intra-coded blocks are manipulated for data embedding. However, the number of the smallest PBs depends on the video content. When there are more fine or heterogeneous regions, more small PBs are coded, and vice versa. While intra prediction mode is an attractive technique to embed data, there is a situation where data cannot be embedded into a block. It is because all prediction modes and coding block structure are evaluated to achieve better compression and only the one with the optimal RDC is selected for coding. When an intra prediction mode is manipulated, the RDC might be higher than other prediction modes or block sizes, and hence the coding pipeline reduces the number of intra-coded blocks that can be utilized for data embedding. Therefore, the availability of the embedding venue greatly restricts the applications of data embedding that require high payload such as private communication and general administrative purposes.

2.5.2 Motion Vector-based Data Embedding

The inter-picture prediction syntax elements utilized for removing temporal redundancy are also commonly exploited to embed data. For example, Nguyen et al. (Nguyen, Tay, & Deng, 2006) and Xu et al. (C. Xu, Ping, & Zhang, 2006) propose to embed data into a H.264/AVC video during inter prediction by exploiting the magnitude of both the horizontal and vertical components of MV. The least significant bit (LSB) of either the x or y component of MV is modified for embedding payload bit. The proposed methods offered low embedding capacity. On the other hand, the difference between the phase angles for MV pairs are manipulated by Fang et al. (Fang & Chang, 2006) to embed data. When the phase angle difference does not satisfy the embedding condition, one of the MVs is replaced by a qualified MV. The embedding process is restricted to large MVs (magnitude) with the justification that changes to MVs with larger magnitude lead to less noticeable distortion. Wang et al. propose to embed data by exploiting MVs and mode selection in H.264/AVC compressed video (P. Wang, Zheng, & Ying, 2008). Aly then associates data to MVs with high prediction error in MPEG-2 compressed video (Aly, 2011). It is designed with the argument that distortion caused by data embedding will be less obvious when the prediction error is large. However, selecting MVs with high prediction error does not always lead to minimum quality degradation. Subsequently, data embedding may fail when other prediction mode (e.g., intra prediction) has a lower RDC. Specifically, it is observed that the number of MVD is relatively low as compared to other syntax elements, and the perceptual quality degradation is less than 0.06 dB while the bit rate overhead incurred is less than 1%. To suppress distortion, matrix encoding is employed by Hao et al. in their work (Hao, Zhao, & Zhong, 2011) to embed data, where either the horizontal or vertical component of large MVs (with respect to a threshold) are selected to embed data. More recently, Van et al. (Van, Praeter, Wallendael, Cock, & de Walle, 2015) exploit MVD at different partitioning depth levels to increase the embedding opportunity in HEVC. The video quality degradation is < 3 dB when MVD in all different block partitioning depth levels are manipulated for data embedding. Similarly, Yang et al. manipulate the selected MV components in the smallest PUs for HEVC (J. Yang & Li, 2018). Although the number of modification to MV components is reduced, the payload capacity is improved. However, the embedding capacity is limited to the number of coded MVs, which depends on the content of the video sequence. Generally, videos with more non-static or dynamic scenes are coded by using more MVs, which can offer more embedding opportunities, and vice versa. Similar to the intra prediction mode, there is situation where syntax element exploited for data embedding is simply unavailable; for example, the MV is non-existence in an intra predicted block, hence the MV cannot be exploited. In addition, there are unexploited syntax elements (e.g., MV predictor and block merging) for data embedding purposes.

2.5.3 Block Partitioning Structure-based Data Embedding

Instead of dealing with MV/MVD, Tew et al. manipulate PB size based on some predefined mapping rules in HEVC coded video to embed data (Tew & Wong, 2014). Moveover, Shanableh associates the payload bit to split flag, which is utilized in HEVC to indicate whether a block is split into smaller blocks. Specifically, a weighting model is introduced to predict split decision of a block. If the payload bit to be embedded is ‘1’, both flags must be identical, otherwise the coded split flag must be different from the predicted flag (Shanableh, 2018). In another work, Yang et al. divide block partition modes into three groups to represent different data bits (Y. Yang, Li, Xie, & Zhang, 2019). It is noteworthy that the embedding capacity for block partitioning-based approach is higher as compared to the former methods because it is applicable to both intra and inter-coded blocks. However, the embedding capacity still depends on the texture and content of the video. Specifically, a video frame with more smooth regions tends to be coded by using larger blocks. Hence, such frame usually carries lower payload, and vice versa. It is observed that the quality degradation for prediction block structure method falls in the range of [1.0, 2.3] dB while the bit rate overhead falls in the range of [5, 5.5] (Y. Yang et al., 2019). Manipulating coding block structure enables more payload capacity at the cost of slightly higher quality degradation and bit rate overhead. This is because more bits are spent in coding sub-optimal block and prediction mode, which indirectly increases the bit stream size due to higher prediction error and additional signaling to the decoder. Since the non-optimal mode and size are coded after manipulating the coding structure, the bit rate variation is higher and the quality of the video is affected.

2.5.4 Coefficient-based Data Embedding

Data embedding in transform and quantification domain includes the work of (Noorkami & Mersereau, 2008; Swati, Hayat, & Shahid, 2014; Chang, Chung, Chen, Lin, & Lin, 2014; Tew & Wong, 2014; Van et al., 2015; Dutta & Gupta, 2016; Buhari et al., 2016; Amiri, Amiri, & Meghdadi, 2019; Singh, Nigam, Singh, & Elhoseny, 2020; Vybornova, 2020). Recently, Vybornova embedded an image into the coefficients of Discrete Wavelet Transform for video authentication purposes (Vybornova, 2020). Embedding data into the mid-frequency and high-frequency AC coefficients usually results in extra bit rate overhead while embedding data in low-frequency coefficients leads to perceptual quality degradation. Manipulating the value of DC coefficient causes changes to the entire block of pixel values. Changes to more transform coefficients may result in artifact and being perceptible and notably degradation. It is observed that the average quality degradation for coefficient-based methods (Noorkami & Mersereau, 2008; Buhari et al., 2016) falls in the range of $[0.2, 4.8]$ dB while the bit rate overhead falls in the range of $[1.4, 5.0]$ percents. Therefore, modification to coefficients must be well-planned in order to minimize the distortion drift and propagation error.

2.5.5 Discussion on Existing Data Embedding Methods

The existing data embedding in video compressed domain is summarized in Table 2.2. Among the methods, either payload is low or quality degradation with increased bit rate overhead are encountered after data embedding. Data embedding strategies vary depending on the coding tools, syntax elements and the underlying architecture of the video coding standard. Undoubtedly, the conventional data embedding methods proposed for HEVC can be adopted to SHVC. However, there are additional features and tools in SHVC that can be exploited to improve the work. For instance, the payload can be increased by using multi-coded layers. In addition, video quality degradation introduced by data embedding can be minimized by using inter layer prediction. The data embedding works designed for scalable coded video include (Shanableh, 2012), (Buhari et al., 2016), (Amiri et al., 2019) and (Sun et al., 2021). Shanableh proposed to associate message bits with the QP parameters in a multi layer of a MPEG-2 coded video (Shanableh, 2012). The payload was increased due to the number of layers applied for data embedding. Buhari et al. embedded data into the DCT coefficients that satisfy some predefined complexity model (Buhari et al., 2016). Limited number of data bits were inserted to the DCT coefficient of each scalable coded layer with slight quality degradation and bit rate overhead. Amiri et

Table 2.2: Existing data embedding in video compressed domain.

Venue	Video Standard	Remarks	Reference
Intra prediction mode	H.264/AVC HEVC	• Limited intra directional prediction modes in H.264/AVC affect the coding efficiency; • The number of intra prediction modes has increased from 9 in H.264/AVC to 35 in HEVC, leading to more opportunities for data embedding; • Minimal impact on the video quality at the cost of larger file size; • If the prediction direction is shifted further away from the optimal direction, the bit rate overhead will be increased; • Data embedding opportunity is reduced when other prediction mode or block size achieves better compression. • It is content-dependent.	(Hu et al., 2007) (G. Yang et al., 2011) (J. Wang et al., 2015) (J. Xu et al., 2015) (Sheng et al., 2016) (Mareen et al., 2018) (Saberi et al., 2020)
Motion vector	MPEG-2 H.264/AVC HEVC	• Relatively low embedding capacity; • Embedding capacity depends on the number of MVs; • The perceptual quality degradation and bit rate overhead incurred is insignificant; • Compete with other prediction mode to achieve coding efficiency; • It is content-dependent.	(Nguyen et al., 2006) (C. Xu et al., 2006) (Fang & Chang, 2006) (P. Wang et al., 2008) (Aly, 2011) (Hao et al., 2011) (Van et al., 2015) (J. Yang & Li, 2018)
Block partitioning structure	HEVC	• Applicable to both intra and inter picture prediction prediction modes; • Frame with more homogeneous regions carries lower payload, and vice versa; • Higher embedding capacity at the cost of higher quality degradation and bit rate overhead; • Increase in bit stream size is due to higher prediction error and additional signaling to the decoder; • It is content-dependent.	(Shanableh, 2012) (Tew & Wong, 2014) (Y. Yang et al., 2019)
Coefficient	H.264/AVC H.264/SVC HEVC	• Dependency between video frames has to be taken into consideration to reduce the propagation error; • Changes to more transform coefficients may result in artifact, and notably degradation; • Manipulating DC coefficient may cause changes to the entire block of pixels values; • Changes to mid-frequency and high-frequency AC coefficients usually results in extra bit rate overhead; • Changes to mid-frequency AC coefficients leads to perceptual quality degradation; • It is not content-dependent.	(Noorkami & Mersereau, 2008) (Swati et al., 2014) (Chang et al., 2014) (Tew & Wong, 2014) (Van et al., 2015) (Dutta & Gupta, 2016) (Buhari et al., 2016) (Amiri et al., 2019) (Singh et al., 2020) (Vybornova, 2020) (Sun, Wang, Huang, & Chen, 2021)

al. manipulated coefficients of frequency sub-bands in a H.264/SVC coded video so that the embedding visual artifact occurs in the region of high texture and brightness contrast (Amiri et al., 2019). Similar work is found in (Sun et al., 2021). However, these methods are designed for an older generation of scalable video codec, namely H.264/SVC and MPEG-2. Furthermore, SHVC is different from SVC in term of the architectural design (i.e., multi-loop coding architecture and EL coding applies high level syntax changes). As a result, these data embedding methods cannot be applied directly in SHVC coded video.

Most existing methods are only able to embed data into blocks coded in either intra picture or inter picture prediction mode. Besides that, depending on the availability of the existing syntax elements, the conventional methods may not be able to fulfill the requirements of applications that require high embedding capacity.

Regardless of the application of interest, it is crucial to minimize the degradation in perceptual quality caused by embedding data into the video. The requirement on quality is particularly important in steganography, which aims to secretly embed a large payload (Abdulla, 2015). Other applications have different requirements, for example, the survival of the embedded data as in the application of watermarking (Tew & Wong, 2014), quick data extraction for continuous authentication (Wong, Chan, & MaungMaung, 2020), and reversibility for handling sensitive content or rare artwork (Guan & Wu, 2020). However, increased payload usually leads to image distortion or quality degradation. Abdulla et al. propose to decompose pixel intensity values into 16 bit-planes to increase the payload. The proposed scheme achieves a good trade-off between payload and image quality (Abdulla, Sellahewa, & Jassim, 2014). In addition, bit-plane(s) mapping technique is invented to increase similarity between the binary secret image and the LSB plane of the cover image for reducing changes due to data embedding while maintaining payload (Abdulla, Sellahewa, & Jassim, 2019). On the other hand, Konyar et al. utilize a matrix encoding-based approach to achieve the same goal (Konyar, Akbulut, & Öztürk, 2020). The strengths and weaknesses of the matrix encoding are identified to achieve a trade-off among payload, increase in bit rate, and video quality. The ability to maintain high perceptual quality of the video is particularly important for high resolution video such as 4K and annotated endoscopic video for health care applications. Hence, while increasing the embedding capacity, it is also essential to ensure that the perceptual quality of the manipulated video can be maintained. On the other hand, the bit rate overhead and increased file size may cause transmission delay. Therefore, a balanced trade-off among increase in embedding capacity, perceptual quality as well as bit rate overhead needs to be achieved.

2.6 Prospective Research Scope

Nowadays, the use of scalable coded video, which saves the cost for storing and distributing multiple non-scalable video over the internet depending on the heterogeneous networks condition, decoding capability and display resolution, has drawn more attention. While the SHVC is gaining popularity, the literature to date only contains a handful of data embedding methods for SHVC coded video, although many syntax elements and encoding tools of SHVC can be utilized for data embedding purposes. This has motivated this study to investigate into the coding structure of SHVC which has more advanced features and higher-efficiency coding tools that can be utilized for data embedding.

Specifically, data embedding design based on the previous video coding standards (e.g., H.264, SVC) and HEVC can be improved and adopted to SHVC. For instance, the embedding capacity can be maximized by using multi-coded layers. Besides that, video quality degradation introduced by data embedding can be minimized using inter layer prediction and include lower layer decoded video frame as reference layer when data embedding takes place in upper ELs. The correlation of the co-located block can also be utilized for embedding data, which can further preserve the video quality and reduce bit rate overhead. With these significant differences, this research aims to design and develop data embedding methods based on the architecture of SHVC to increase embedding capacity while preserving the video quality with reasonable bit rate overhead.

Manipulating video for data embedding usually degrades the quality of the video. Limiting the embedding capacity can minimize video degradation, and vice versa. However, there are applications (e.g. military and legal applications) which require a high payload with good perceptual quality. The great challenge here is to achieve high embedding capacity while maintaining imperceptibility simultaneously. Most existing data embedding works focus on transform domain to increase the payload. However, based on the extensive literature search, the concept of utilizing multiple predictive syntax elements in both intra and inter prediction modes simultaneously for data embedding has not been investigated as to date. Hence, this research plans to investigate into the feasibility of utilizing multiple predictive syntax elements of the SHVC codec to achieve higher embedding capacity while reducing the impact to the quality and bit rate overhead.

Manipulating video in transformation and quantization domains to embed data bit may result in error propagation if the embedding strategy and venues are not well-planned. The distortion is more severe when a distorted video frame is utilized as reference for coding video. Many techniques has been proposed to eliminate the distortion due to propagation error. For example, data embedding in the DCT domain is usually performed on the lower or the mid-band frequencies, as higher frequencies are lost when the image is compressed (Buhari et al., 2016). However, when more payload is required, quality degradation is unavoidable or only limited payload capacity is available. Hence, it is important to seek for a solution to meet the requirement of high data embedding capacity while preserving the visual quality.

Different data embedding applications require different payload capacity, typically varying from low (e.g. access control) to high payload high (e.g., finger printing and authentication application). All these applications require for a good trade-off among payload capacity, better visual quality and minimal bit rate overhead than those of state-of-the-art schemes. Generally, coding a frame using more (smaller) blocks implies that more syntax elements are available for manipulation to embed data. More syntax elements can be obtained by forcing the encoder to encode a frame by using small blocks instead of large blocks to improve embedding capacity as long as the bit rate overhead is within an allowable limit.

2.7 Summary

In this chapter, an overview of the international standardization of video coding technology is provided. The coding tools and features of HEVC and SHVC are studied and compared to the previous standards, namely H.264/AVC and SVC. Subsequently, the state-of-the-art data embedding methods in video compressed domain are reviewed. Specifically, the architectural design of HEVC is different from previous video coding standards in a number of respects. Varying sizes block partitioning structure, additional prediction modes and tools are adopted to achieve better compression. In addition to the above-mentioned features, SHVC adopts multi-loop structure to retain predictive information for inter-layer prediction at the cost of increased computational complexity. The knowledge gaps in data embedding in video is then succinctly identified. In particular, data embedding in prediction modes or block partitioning structure depends on the availability of the syntax elements and coded block size. Data embedding opportunity is also reduced when other prediction mode or block partitioning structure achieves a better coding efficiency. Generally, increased payload may affect the quality of the video with increased file size, which is unfavorited for storage and transmission. Similarly, manipulating more transformed coefficients may cause artifact and notably quality degradation as well as propagation of error if the embedding process is not well-planned. Lastly, the discussion to close the knowledge gaps for data embedding in the scalable coded video is provided for future research work.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Introduction

This research study explores the potential venues that can be utilized to embed data in SHVC coded video and seeks solutions to increase embedding capacity while maintaining minimum distortion and bit rate overhead with a better rate-distortion performance for data embedding purposes. This chapter discusses the research methodology adopted for conducting this research. Generally, this research study is structured into four phases which includes: literature reviews, exploration on data embedding venues, design and implementation of data embedding techniques, and the performance evaluation and analysis, as illustrated in Figure 3.1.

The relevant topics of the state-of-the-art data embedding techniques and video coding standards were reviewed. Two main areas, namely, (a) recent video compression standards, and (b) state-of-the-art data embedding in compression video, were studied to gain knowledge of video coding principles, as well as data embedding concepts and techniques. The state-of-the-art data embedding techniques in video were then compared, classified and summarized according to the video coding standards and domains. Subsequently, critical analysis and evaluation were performed for each method to identify the strength and limitation, as well as the contribution of each method to the body of knowledge of this domain. Then, the issues and problems in the reviewed state-of-the-art methods were discussed in order to seek the knowledge gaps and identify areas that require further investigation and research.

3.2 Exploration of Data Embedding Venues

Although similar principles are used in the design of video coding, video compression mechanism has changed considerably, and has evolved enormously over time, taking

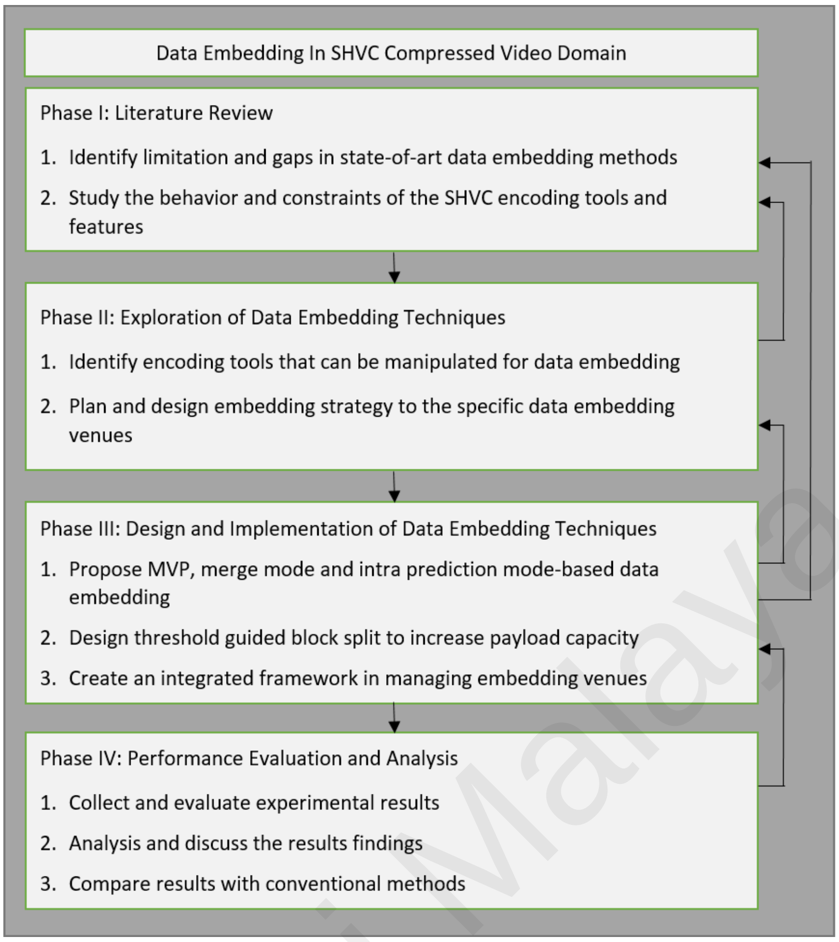

Data Embedding In SHVC Compressed Video Domain
Phase I: Literature Review
1. Identify limitation and gaps in state-of-art data embedding methods
2. Study the behavior and constraints of the SHVC encoding tools and features
Phase II: Exploration of Data Embedding Techniques
1. Identify encoding tools that can be manipulated for data embedding
2. Plan and design embedding strategy to the specific data embedding venues
Phase III: Design and Implementation of Data Embedding Techniques
1. Propose MVP, merge mode and intra prediction mode-based data embedding
2. Design threshold guided block split to increase payload capacity
3. Create an integrated framework in managing embedding venues
Phase IV: Performance Evaluation and Analysis
1. Collect and evaluate experimental results
2. Analysis and discuss the results findings
3. Compare results with conventional methods

Figure 3.1: Research methodology.

the advantage of the increasing computational power of the digital system. Hence, a study on the SHVC codec and recent video coding standards is essential to gain the latest development and concepts used in video coding and its features. It allows us to gain a deeper understanding of the encoder functionalities and behaviour so to seek for data embedding opportunities. Recall that an encoder may select between intra-picture prediction and inter-picture prediction for block-shaped regions of each video frame to remove spatial or temporal redundancy. The predicted information is stored in predictive syntax elements together with the prediction residuals as the difference between the predicted and actual image, which are then compressed using quantization, transformation and entropy coding. In the process, the predictive syntax elements, constraints imposed on values of the syntax elements, semantics (i.e., the scope, restrictions and conditions that are imposed on the syntax elements) associated with the syntax structures, and encoding algorithms were identified and analyzed to explore the possibility of utilizing these predictive syntax elements for data embedding purposes. After each predictive syntax element has been identified, embedding using bit mapping rule and replacement that can be applied to the identified data embedding venue was determined. A data embedding strategy and plan was then put forward after considering the constraints of the encoding tools. A well-planned embedding strategy and venue selection for data embedding is essential to achieve imperceptibility as well as to minimize variation in bit rate. Further consideration was the feasibility of the data extraction at the decoder end. The video bit stream file contains entropy coded predictive syntax elements and residuals as well as other prediction information. It is also required to ensure that the bit stream file is format compliant with the video coding standard and decodable by using SHVC decoder while the embedded payload bits can be extracted from these predictive syntax elements at the receiver end during video decoding.

3.3 Design and Implementation of Data Embedding Techniques

Data embedding techniques were designed for each embedding venue in compressed domain and realized by using the SHVC reference software (Joint Collaborative Team on Video Coding, 2017). Specifically, the predictive syntax elements of intra- and inter-picture prediction of SHVC, which include motion vector predictor, merge mode and intra prediction mode in all spatial scalable coded layers of SHVC were considered for manipulation. Bit mapping rule and replacement were applied to embed data bit to the identified venue. Optimization strategy was designed in order to increase the payload at an acceptable bit rate overhead. First, a threshold guided block splitting process was proposed to facilitate more small blocks for data embedding. Then, a data embedding framework was designed to select and manage multiple data embedding venues and coding layers.

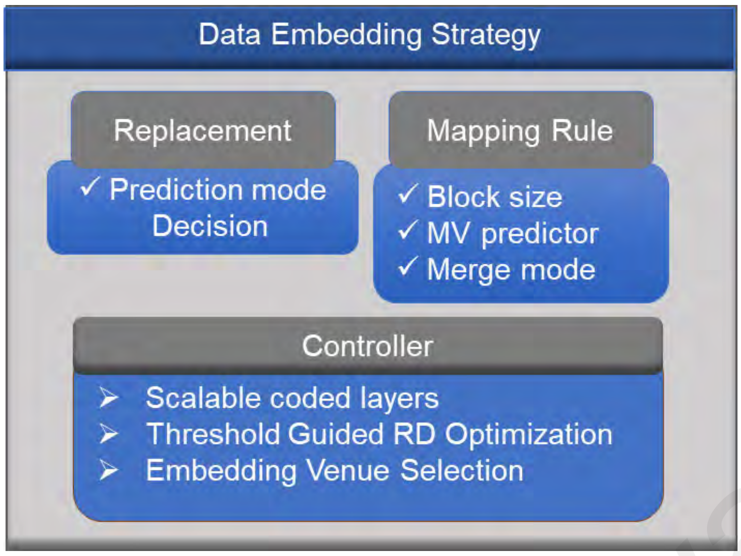
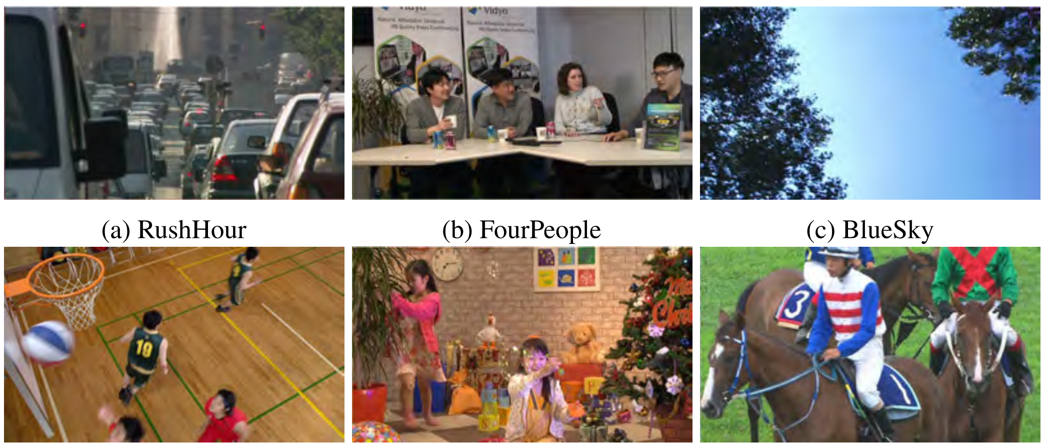


Figure 3.2: Data embedding strategy.

The data embedding strategy used to realize the data embedding is depicted in Figure 3.2.

In the process, the required software and test data set (i.e., video test sequence) were identified. Specifically, two types of data were involved. First, it was the video test sequences, which were obtained from the Universitat-Hannover (Universität-Hannover, 2013) collection and the Derf (Xiph.org, 2013) collection. Six standard video test sequences as shown in Figure 3.3 were considered to evaluate the performance of the proposed data embedding methods. The second input was the data to be embedded. For experiment purposes, a pseudo-random number generator was employed to generate a sequence of binary numbers (i.e., 0's and 1's), which was then embedded as the payload. The experiments were conducted by using a PC with AMD Ryzen 5 3500U 2.10 GHz CPU and 8 GB of RAM running on a 64-bit Windows 10 operating system. All results were collected by processing the first 200 frames (which is the maximum length for one of the sequence - *BlueSky*, rounded to hundreds) in each video test sequence.


(a) RushHour (b) FourPeople (c) BlueSky

(d) BasketballDrill (e) PartyScene (f) RaceHorses

Figure 3.3: Video test sequences considered for experiments.

3.4 Video Test Sequence Selection

Selecting video test sequence has been a challenging task in this research study with the absence of the powerful computer in running the experiment and to collect experimental results within a limited time frame. It is noted that more video test sequences should be considered to better reflect the performance of the proposed method and to make the arguments more convincing. However, it takes a long period of time to encode these video test sequences, which are high in resolution hence large in file size. The selected videos should have different levels of spatio-temporal activities (Winkler, 2012; ITU-T, 2008). Therefore, the video test sequences considered in the experiments are carefully selected so that they are sufficiently diverse, i.e., having basic simple to complex scenes (i.e., spatial activity consisting of smooth region of varying size), motion of various speeds as well as static and dynamic background. The selected video test sequences include *RushHour*, *FourPeople*, *BlueSky*, *BasketballDrill*, *PartyScene* and *RaceHorses*. The characterization of video content for the selected video sequences is described as follows:

RushHour: A sequence showing a street in a city with heavy traffic. There are many vehicles captured at various distances and displayed as smooth regions of different sizes. The camera stands still and most motions are related to the moving vehicles. The background is not completely static due to the air ripples caused by heat.

FourPeople: This is a scene with four people sitting and passing brochure from one to another. There are areas rich in textures such as the bunting in the background, but also some smoother areas such as the wall and tables. The camera is stationed in a fixed position. The background is also static. Most motions are related to the hands and heads of the four people for passing brochure.

BlueSky: This is a very high contrast sequence showing dark leaves of a tree with bright blue sky and relatively smooth rotating motion. Some areas are very smooth (blue sky) while some areas are very complex with high contrast borders between the tree and the sky.

BasketballDrill: This is a sequence with much motion, where several basketball players are performing shooting drills. The basketball players are running and turning quickly while the camera fixed and the background is static. The background is relatively rich in details and textures.

PartyScene: A scene with few children, a pile of colorful Christmas gifts, a complex brick background, and a Christmas tree on the left. Two children are running around a tree on the right hand side, a girl is blowing bubbles, while a duck toy is dancing to the left and right. This sequence has small objects with complex motion and much fine texture across the whole frame. The camera pans to the direction of the girl who is blowing bubbles.

RaceHorses: This is a sequence with few people riding on horses and moving around. It is a dynamic and motion-filled video, which contains high-frequency details. The camera is panning to focus on the moving object. The background is not static and the foreground has a mixture of smooth and detail region.

3.5 Performance Evaluation and Analysis

Generally, an encoder aims at encoding video at an optimal level with the trade-off between bit rate and perceptual quality. After a video is embedded with data bits, it might cause slight bit rate variation and video quality degradation. This research study has adopted the quantitative methodology to examine the relationship among payload capacity embedded in different venues, the impact to bit rate variation, and perceptual quality. Besides that, the embedded data bits are extracted at the decoder end for verification purposes. The assessment methods and evaluation metrics utilized to measure the quality of the signal and bit rate variation were identified and detailed in the following subsections. The experimental results were analyzed and compared with the state-of-the-art data embedding methods in term of bit rate variation, perceptual quality and payload capacity. The findings were discussed and explained. Finally, conclusions of this research study was drawn from the experimental results and findings.

3.5.1 Video Quality

While subjective video quality analysis is measured by using HVS, objective video quality analysis is commonly measured by using the PSNR and Structural Similarity Index Metrics (SSIM). Mathematically, PSNR is a logarithmic representation of Mean Squared Error (MSE). The MSE is the cumulative squared error between the compressed and the original video. Generally, a higher value of PSNR indicates that the video has higher quality. The mathematical equation for MSE is defined in the following:

$$MSE = \frac{1}{hw}\sum_{i=1}^{h}\sum_{j=1}^{w}(V'_{i,j} - V_{i,j})^2, \tag{3.1}$$

where h and w are the height and the width of the video, V is the original video and V' is the processed video.

Then, the PSNR is calculated between the original and processed video. The PSNR is defined by using the following equation:

$$PSNR = 10\log_{10}\frac{255\times 255}{MSE}. \tag{3.2}$$

It is defined in decibels (dB). The quality variation, $\Delta PSNR$ is defined as

$$PSNR_o - PSNR_m, \tag{3.3}$$

where $PSNR_o$ and $PSNR'_m$ are the video coding quality before and after embedding with payload bits, respectively. The smaller the value of Δ*PSNR*, the more similarity between the processed and original video.

The SSIM is another method for predicting the perceived quality of videos and measuring the similarity between two video images (Z. Wang, Bovik, Sheikh, & Simoncelli, 2004). The measurement of video quality is based on an initial uncompressed or distortion-free video as a reference. It computes the mean, variance and co-variance of small patches within a frame and combines the measurements into a distortion map. This metric calculates the structural similarity index between two video frames based on three components: luminance, contrast, and structure. Let f and f' are the two input frames being compared. Firstly, the luminance of each signal is compared. The luminance is measured by taking the average of all pixel values in a video frame. It is estimated as the mean intensity of a given video frame and denoted by μ and is computed as

$$\mu_f = \frac{1}{N}\sum_{i=1}^{N} f_i. \tag{3.4}$$

The luminance comparison is then defined as the function $l(f, f')$ in term of μ_f and $\mu_{f'}$

as shown in the following expression:

$$l(f, f') = \frac{2\mu_f \mu_{f'} + C_1}{\mu_f^2 + \mu_{f'}^2 + C_1}, \tag{3.5}$$

where C_1 is a constant included to ensure stability when the value of $(\mu_f^2 + \mu_{f'}^2)$ is very close to 0. C_1 is defined as

$$C_1 = (K_1 L)^2, \tag{3.6}$$

where L is the dynamic range of the pixel values (it is set to 255 for standard 8-bit images), and K_1 is a small constant.

Next, the contrast is measured by using the standard deviation (square root of variance) of all the pixel values of a video frame. It is denoted by σ and represented by the following formula:

$$\sigma_f = (\frac{1}{N-1} \sum_{i=1}^{N} (f_i - \mu_f)^2)^{\frac{1}{2}}. \tag{3.7}$$

The contrast comparison is then defined as a function $c(f, f')$ for comparing σ_f and $\sigma_{f'}$ as shown in the following expression:

$$c(f, f') = \frac{2\sigma_f \sigma_{f'} + c_2}{\sigma_f^2 + \sigma_{f'}^2 + C_2}, \tag{3.8}$$

where C_2 is defined as

$$C_2 = (K_2 L)^2, \tag{3.9}$$

for K_2 is a small constant.

Subsequently, the structural index is measured. The input signal is divided by its standard deviation (denoted as σ) to obtain unit standard deviation which is computed as:

$$\frac{f - \mu_f}{\sigma_f}. \tag{3.10}$$

The structural comparison is conducted after luminance subtraction and variance normalization. It is defined as a function $s(f, f')$ to compare the two signal in term of unit standard deviation as expressed below:

$$s(f, f') = \frac{\sigma_{ff'} + C_3}{\sigma_f \sigma_{f'} + C_3}, \tag{3.11}$$

where $\sigma(ff')$ is defined as

$$\sigma(ff') = \frac{1}{N-1} \sum_{i=1}^{N} (f_i - \mu_f)(f'_i - \mu_{f'}). \tag{3.12}$$

Finally, these three comparison functions, namely, $l(f, f')$, $c(f, f')$ and $s(f, f')$, are combined to produce the similarity index between signals f and f', which is given by

$$SSIM(f, f') = [l(f, f')]^{\alpha} \cdot [c(f, f')]^{\beta} \cdot [s(f, f')]^{\eta}, \tag{3.13}$$

where $\alpha > 0, \beta > 0, \eta > 0$ are parameters utilized to denote the relative importance of the three components. When $\alpha = \beta = \eta = 1$ and $C_3 = C_2/2$, this results in a specific form of SSIM:

$$SSIM(f, f') = \frac{(2\mu_f \mu_{f'} + C_1)(2\sigma_{ff'} + C_2)}{(\mu_f^2 + \mu_{f'}^2 + C_1)(\sigma_f^2 + \sigma_{f'}^2 + C_2)}. \tag{3.14}$$

Generally, k_1 is set to 0.01 and k_2 is set to 0.03. The SSIM values between two video frames falls in the range of $[0, 1]$. A value of 1 indicates that the two video frames are very similar or identical while a value of 0 indicates that the two video frames are very different.

3.5.2 Bit Rate Variation (BRV)

Bit rate is the amount of data required to encode a second of video. Bit rate affects both the quality and file size of a streaming video. The higher the bit rate the higher the quality of the video and the larger the file size. When payload bits are embedded into a video sequence, additional bits are required to carry embedded data hence the increase in the bit rate. The bit rate variation, *BRV* is defined as

$$\frac{BR_m - BR_o}{BR_o} \times 100\%, \tag{3.15}$$

where BR_o and BR_m are the bit rate before and after embedding with payload bits, respectively.

3.5.3 Bjøntegaard Metric

In general, the Bjøntegaard metric is used to measure the coding efficiency of a processed video in comparison with the original video by taking multiple quality points or bit rates, which are encoded by using different quantization parameters. Here, it is adopted for this specific application i.e., data embedding. The Bjontegaard metric, which includes Bjøntegaard delta of bit rate (BD-BR) and Bjøntegaard delta of PSNR (BD-PSNR) are computed to analyze the bit rate overhead and video quality degradation (Bjøntegaard, 2001). It enables the comparison of rate distortion curves in terms of the average bit rate savings/overhead for the same video quality (e.g., PSNR), as well as the average PSNR gain/loss for a given bit rate range (e.g., BR) for the same bit rate, with respect to the original video. It is calculated between two rate distortion curves. In general, a decrease in BD-BR and an increase in BD-PSNR indicates that the processed video has better performance.

3.6 Summary

This chapter has outlined the research methodology adopted in this research. The research study was conducted by firstly examining the state-of-the-art data embedding techniques, followed by video coding tools and features of the recent video coding standards. In the process, the knowledge gaps and video coding tools that can be utilized for data embedding were identified. A study was conducted to examine the feasibility of associating targeted predictive syntax elements with payload bits. A data embedding strategy and plan were then put forward after considering the constraints of the encoding tools. Then, the research focused on designing and developing data embedding algorithms. The assessment methods and evaluation metrics that are commonly applied in the data embedding domains were identified. Subsequently, video test sequences were selected and experiments were conducted to demonstrate the effectiveness of the proposed data embedding technique. The video test sequences considered in the experiments were carefully selected to ensure that they are sufficiently diverse, i.e., having complex scenes (i.e., spatial activity) and motion of various speeds. Next, the experimental results were verified, analysed and discussed. The experimental results were compared with the state-of-the-art data embedding methods in term of bit rate variation, perceptual quality and payload capacity. Finally, conclusions of this research study are drawn from the experimental results and findings.

CHAPTER 4: A DATA EMBEDDING TECHNIQUE FOR SPATIAL SCALABLE CODED VIDEO USING MOTION VECTOR PREDICTOR

4.1 Introduction

Video compression algorithms attempt to reduce the number of bits required to encode video content by exploiting both the temporal and spatial redundancies. One of the unique syntax elements in video coding is MV, which is introduced for inter-frame prediction to remove temporal redundancy among neighboring frames. Neighboring blocks in a video frame usually contain similar motion data. The MV of the neighboring blocks and frames can be utilized as a motion predictor to reduce the amount of data required for coding MV of a coding block. The syntax element motion vector predictor (MVP) was introduced in the HEVC and its extensions to improve the predictive coding of MVs of a coding block by using the neighboring motion information. Specifically, the difference between the MVs of the coding block and the selected MVP is minimized to improve the coding efficiency.

This chapter presents a data embedding technique for SHVC video compressed domain by manipulating the syntax elements related to MVs. Although MVs have been considered in existing works, the MVP and multi-layer natural of the scalable video codec have not been considered. At the time of this writing, this is the first data embedding technique based on MV for SHVC (Boyce et al., 2016). Data is embedded into multiple coded layers of a video by manipulating the MVP. Specifically, the MVP index specifying the MVP candidate of the current block is associated with data bit. Furthermore, to improve embedding capacity, more syntax elements are obtained by forcing the encoder to encode a frame by using more small blocks instead of big blocks. The PBs with the smallest size is utilized for data embedding as long as the bit rate overhead is below a predefined threshold. The main contribution of this work is the development of MVP-based threshold-controlled split and new embedding technique to achieve a good trade-off between payload and bit rate variation while preserving the video quality. Experimental results suggest that higher payload is achieved when smaller PBs are coded at a slightly higher bit rate, with negligible degradation in perceptual video quality. In the best case scenario, 118.9 kbps are embedded into the *PartyScene* video test sequence while PSNR drops by 0.06 dB with a bit rate overhead of 4.72%.

4.2 Proposed Data Embedding Technique

For a video sequence, the spatial and temporal correlations are high between neighboring blocks. Specifically, from the perspective of temporal correlation, MVs from neighboring blocks are highly correlated because these blocks are often estimated by using similar motion information. Hence, the motion information for a block can be predicted from that of the neighboring blocks, which can be utilized for the removal of temporal redundancy between neighboring blocks and frames. Particularly, a MVP is derived from the decoded MVs of the spatially neighboring blocks or from a scaled MV of a co-located-temporally-neighboring block to reduce the magnitude of the MV to be coded. The MVD is encoded for storage and transmission purposes, where MVD captures the difference between the predicted MV and the current MV. The calculation of both MVD horizontal (denoted by x) and vertical (denoted by y) components are expressed as $MVD_x = MV_x - MVP_x$ and $MVD_y = MV_y - MVP_y$, respectively. Recall that a coding block has several MVs from its neighboring blocks and frames as potential MVP candidates as detailed in Chapter 2. The best MVP candidate, i.e., the one having the optimal cost, will be selected and encoded.

In this work, the motion correlation of the neighboring blocks is exploited and the selection of MVP is manipulated to embed data by associating the MVP indices to the payload bits. Subsequently, the PB structure is manipulated to achieve higher embedding capacity. Conceptually, partitioning a video frame into more (smaller) prediction blocks leads to more embedding opportunities due to the availability of the syntax elements.

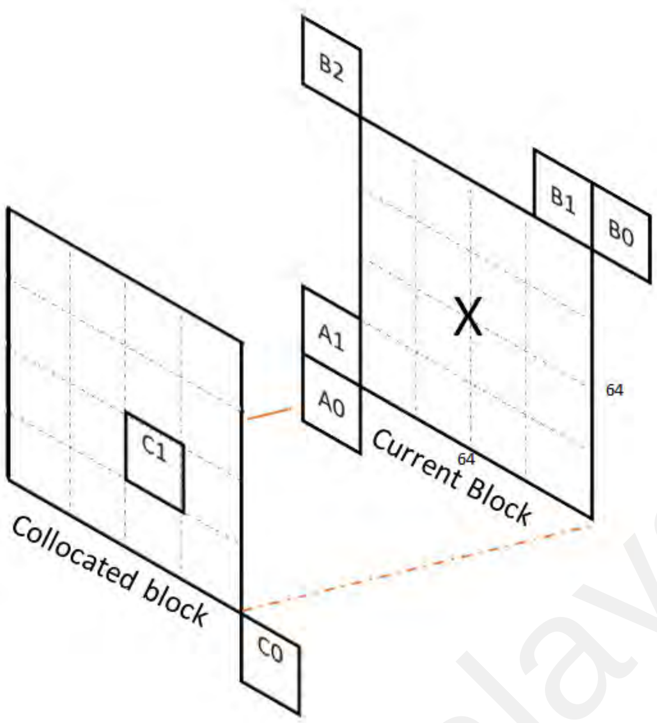


Figure 4.1: MVP candidates.

Therefore, an adjustable control parameter is introduced to guide the quad-tree partitioning process to permit more split blocks. Specifically, when the cost for coding smaller blocks is below a predefined threshold, blocks of smaller size are coded so that there are more MV syntax elements available for data embedding purposes. The following subsections detail the manipulation of MVP and PB size.

4.2.1 Manipulation of Motion Vector Predictor

During inter-picture prediction, two MVP candidates are shortlisted from the neighboring blocks or collocated blocks from the reference picture (Sze et al., 2014). The MVP candidates for block X, including A_0, A_1, B_0, B_1 and B_2 from the neighboring blocks, and C_0 and C_1 of the collocated blocks from the reference picture, are shown in Figure 4.1. Here, the parity bit of MVP indices are exploited to represent the payload bit m to be embedded. The selection of MVP candidate depends on the value of m, which is either '0' or '1'. Instead of using the MVP determined by the encoder, motion information (viz., index's parity bit) of the MVP candidate matching the value of m is selected for coding. Note that the selected MVP is either the best or second best candidate with minimum cost for inter-picture prediction. To increase payload, data bits are embedded in all scalable coded layers. It is noteworthy that although the proposed data embedding technique is designed for spatial scalability video coding, it can be applied to handle different type of scalability, including temporal and quality.

In EL, ILR picture is utilized to explore inter-layer redundancy. When ILR picture is selected as the reference picture, the MVs pointing to the ILR picture are usually set to zero in order to suppress the computational complexity. However, in this work, the zero MV setting (viz., $REF\_IDX\_ME\_ZEROMV$) is disabled so that MVPs will be coded, and data can be embedded in a similar manner as in BL. After the final MVP is determined, MVD is calculated based on the finalized MVP and MV of the current block. The proposed data embedding algorithm is executed simultaneously during the layered encoding process. On the other hand, for extracting the embedded data, only the parity bit of the MVP indices in each layer are required, hence data extraction can be carried out without much overhead.

4.2.2 Manipulation of PB Size

To improve the embedding capacity, the PB size is manipulated. During encoding, the Rate Distortion Optimizer (RDO) is employed to determine a good trade-off between the visual quality and the amount of bits spent on coding as a RDC function. Then, RDC for different PB sizes are compared to identify the PB size having the minimal cost and only the one with the best rate distortion cost is adopted for coding. The encoder calculates $RDC = \mathcal{D} + \lambda\mathcal{R}$, where $\mathcal{D}$ is the distortion between the original input and reconstructed signals, while $\mathcal{R}$ is the compression rate which represents the number of bits spent on coding. The parameter λ is the Lagrange multiplier used for Lagrangian optimization.

Recall that the motion prediction block size ranges from 4×8 and 8×4 luma samples to 64×64 luma samples, where block of 4×8 and 8×4 are the smallest block sizes that can be utilized for motion prediction. For each CTU, RDC of all possible block sizes are calculated in a quad-tree structure. The calculated RDC for each depth level d is compared with the accumulated RDC for the split blocks from depth level $d + 1$. For example, for maximum depth $d_{\max} = 3$, $d = 0, 1, 2$ and 3 corresponds to block sizes of $N = 64, 32, 16$ and 8, respectively. The accumulated cost at depth level $d = 3$ is compared with the cost at depth level $d = 2$. The blocks at depth level d (either 2 or 3 here) having the minimum cost will be selected for coding. Then the cost at depth level $d = 2$ is compared to the cost at depth level $d = 1$. The comparison process continues until all block sizes are compared and coded, which ends at the root (i.e., $d = 0$) of the quad-tree.

A threshold τ is introduced so that when the scaled RDC for bigger block size (denoted by $(1 + \tau) \times \mathcal{P}_d$ is larger than the accumulated RDC of split blocks (denoted by $\mathcal{P}_{d+1}$), the split blocks are coded instead of coding block with the size suggested by RDO. Here,

$$\mathcal{P}_d = \sum_{i=0}^{n_d - 1} RDC_d(i), \tag{4.1}$$

where n_d is the total number of split blocks at partitioning depth level d. Note that having more (smaller) blocks implies that more syntax elements are available for manipulation to embed data. The threshold-controlled embedding can be utilized to maximize the payload by adjusting the value of τ so that more small blocks are coded at slightly higher cost. An example of block splitting depending on the predefined threshold τ is illustrated in Figure 4.2. Assuming that the blocks with the best RDC (grey color) are supposed to be coded. When a threshold a is utilized to guide the block splitting, the RDC for blocks at different partitioning depth levels are computed and compared during the encoding

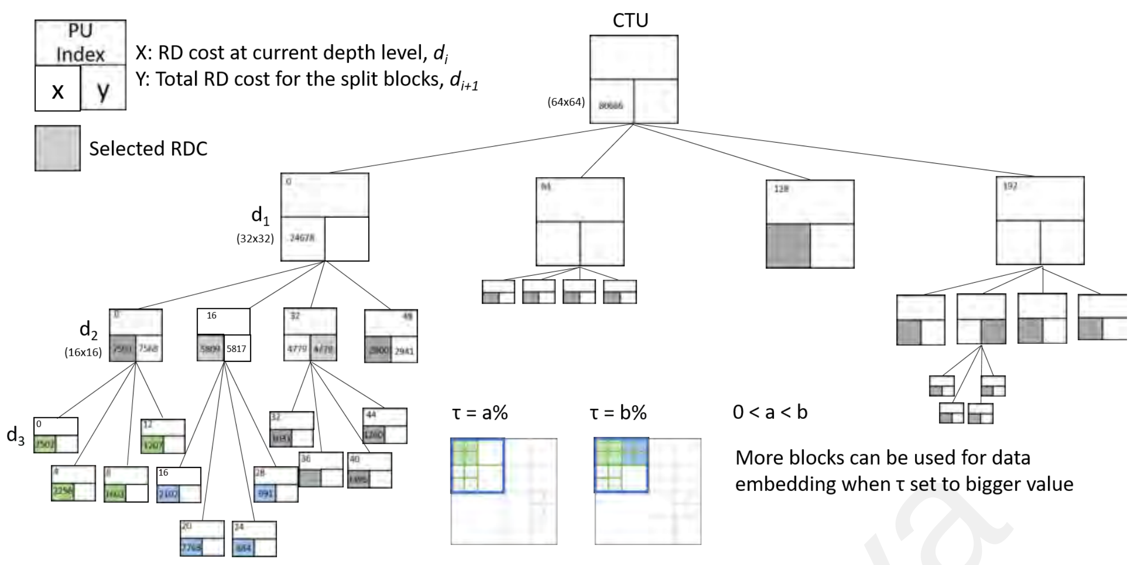

PU Index
X: RD cost at current depth level, d_i
Y: Total RD cost for the split blocks, d_{i+1}
Selected RDC
CTU
(64x64)
d_1
(32x32)
d_2
(16x16)
d_3
τ = a%
τ = b%
0 < a < b
More blocks can be used for data embedding when τ set to bigger value

Figure 4.2: Example of threshold guided block splitting.

Table 4.1: Increment in bit rate (%) after embedding data.

Sequence	**Frame Rate (Hz)**	**Original (kbps)**	τ **= 0%** (%)	τ **= 5%** (%)	τ **= 10%** (%)	**MVD** (%)
RushHour	30	6,687	0.42	6.75	17.58	0.86
FourPeople	60	13,353	0.12	7.76	13.06	0.34
BlueSky	24	13,754	0.23	3.01	4.78	0.89
BasketballDrill	50	10,437	0.30	5.66	10.22	0.75
PartyScene	50	27,058	0.29	2.81	4.72	0.80

process. The RDC for blocks with PU index = 0 at partitioning depth level d_3 is better than the RDC at partitioning depth level d_2, hence the blocks at partitioning depth level d_3 (green color) are coded. By adjusting the threshold to a larger value b, more split blocks are coded as shown in the block with PU index = 16 (blue color).

4.3 Experimental Results

The proposed data embedding technique is implemented by modifying the SHVC reference software SHM-12.0 (Joint Collaborative Team on Video Coding, 2017). Five standard video test sequences (i.e., *RushHour*, *FourPeople*, *PartyScene*, *BlueSky* and *BasketballDrill*) are utilized together with different threshold values. Their frame rates are recorded in Table 4.1, while their resolutions are recorded in Table 4.2. Experiments are conducted by using two spatial layers, with a group of pictures (GOP) of 4. The QPs for BL and EL are set to 22 and 20, respectively. The remaining parameters are set to the SHM default *Low Delay P (LDP)* configuration. The random bits, i.e., sequence of 0's and 1's, are embedded as the payload. To analyze video quality degradation and bit rate overhead, the PSNR, SSIM (Z. Wang et al., 2004) and bit rate are computed based on the first 200 frames of each video sequence.

4.4 Discussion and Analysis

To examine the experimental results, bit rate overhead, quality distortion and payload capacity are measured and compared as in the following subsections.

4.4.1 Variation in Bit Rate

The bit rate of the original and the processed video by the proposed technique are recorded in Table 4.1. As suggested in Table 4.1, when $\tau = 0\%$, the overhead ranges from 0.12% (*FourPeople*) to 0.42% (*RushHour*). When $\tau = 5\%$ and 10%, the lowest overhead are 2.81% and 4.72%, respectively, as observed in *PartyScene*. For $\tau = 5\%$, the highest bit rate overhead is 7.76% (*FourPeople*) while for $\tau = 10\%$, the highest bit rate overhead is 17.58% (*RushHour*). The average bit rate increment is $\sim$ 0.27%, $\sim$ 5.2% and $\sim$ 10.07% for $\tau = 0, 5$ and 10%, respectively. For comparison, bit rate for the video processed by the *MVD*-based data hiding technique (Van et al., 2015) is shown in Table 4.1. *MVD*-based basically manipulates the differences between MVs directly to embed data. Experimental Results suggests that the overhead ranges from 0.34% to 0.89%, where the average overhead is $\sim$ 0.73%, which is slightly higher as compared to the proposed technique with $\tau = 0\%$. One of the reasons leads to such outcome is due to the estimated MVs remain the same (i.e., MV = MVP + MVD) in the *MVP*-based data embedding technique. In contrast, *MVD*-based technique might cause changes to the estimated MVs.

Table 4.2: Video quality degradation in term of PSNR (dB) and SSIM for the processed videos.

Video Sequence	PSNR (10^{-2}dB)				SSIM (10^{-4})			
	τ = 0%	τ = 5%	τ = 10%	MVD	τ = 0%	τ = 5%	τ = 10%	MVD
RushHour								
BL (960 × 540)	1.00	0.00	3.00	2.00	0.00	0.00	1.00	0.00
EL (1280 × 720)	0.00	4.00	1.00	1.00	0.00	0.00	0.00	0.00
FourPeople								
BL (640 × 360)	1.00	0.00	0.00	1.00	0.00	0.00	0.00	1.00
EL (1280 × 720)	0.00	0.00	0.00	1.00	0.00	0.00	0.00	1.00
BlueSky								
BL (640 × 360)	1.00	3.00	7.00	4.00	1.00	0.00	0.00	1.00
EL (1280 × 720)	1.00	3.00	7.00	3.00	0.00	0.00	1.00	0.00
BasketballDrill								
BL (416 × 240)	1.00	1.00	2.00	2.00	0.00	0.00	0.00	0.00
EL (832 × 480)	1.00	1.00	4.00	1.00	0.00	0.00	0.00	3.00
PartyScene								
BL (416 × 240)	1.00	2.00	6.00	2.00	0.00	0.00	0.00	0.00
EL (832 × 480)	0.00	3.00	6.00	1.00	0.00	0.00	0.00	5.00

Therefore, additional bits are required to code the prediction error, which leads to the higher bit rate overhead.

4.4.2 Video Quality

Table 4.2 shows the difference in quality between the original and processed videos in terms of PSNR and SSIM. Results in Table 4.2 suggest that the manipulation of PB size degrades the perceptual video quality insignificantly in both BL and EL for all video sequences. The PSNR drops by $\leq$ 0.01% when the threshold τ is set to 0%. The PSNR degradation ranges from 0.01 to 0.04 dB when the threshold τ is set to 5%. In the worst case scenario, PSNR drops by 0.04 dB for EL of the sequence *RushHour*. On the other hand, in the best case scenario, the perceptual quality for the sequence *FourPeople* is almost the same as the original decoded video sequence for both τ = 5% and 10%. In general, results suggest that when τ is set to a larger value, the quality is still maintained, while PSNR decreases < 0.1 dB. For *MVD*, results suggest that the PSNR degradation ranges from 0.01 to 0.04 dB, where the average PSNR drops by ~ 0.02 dB. When comparing between the proposed *MVP*- and the existing *MVD*-based techniques, mixed results are

Table 4.3: Comparison of embedding capacity (kbps).

Sequence		τ **= 0%**	τ **= 5%**		τ **= 10%**		**MVD**	
		(kbps)	**(kbps)**	(%)	**(kbps)**	(%)	**(kbps)**	(%)
RushHour	BL	8.71	18.91	117	21.93	152	8.31	−5
	EL	3.39	27.93	723	123.26	3533	3.13	−8
	Total	12.10	46.84	287	145.18	1100	11.44	−5
FourPeople	BL	4.40	7.44	69	8.64	97	3.96	−10
	EL	10.31	32.13	212	50.66	391	9.16	−11
	Total	14.71	39.58	169	59.30	303	13.12	−11
BlueSky	BL	7.84	13.08	67	15.06	92	6.60	−16
	EL	26.04	51.93	99	60.28	131	22.03	−15
	Total	33.88	65.01	92	75.34	122	28.63	−15
BasketballDrill	BL	6.30	8.78	39	9.42	49	5.62	−11
	EL	16.17	32.26	99	44.76	177	14.27	−12
	Total	22.48	41.04	83	54.18	141	19.88	−12
PartyScene	BL	11.03	16.84	53	17.97	63	9.53	−14
	EL	45.51	85.94	89	100.95	122	38.23	−16
	Total	56.53	102.78	82	118.92	110	47.76	−16

observed for different videos and parameter settings. Nonetheless, both techniques show comparable video quality after embedding data, and the degradation is negligible. One of the reason for perceptual quality of the processed video can be maintained is due to the coding decision is made after data embedding. In addition, the prediction error caused by data embedding is calculated as a part of the prediction residual. Therefore, the video can be reconstructed as an approximation of the original video.

4.4.3 Payload

Table 4.3 records the increment of payload for both techniques, using the payload achieved by $\tau = 0$ as the reference value. The capacity consistently increases when τ increases, and as expected larger τ leads to more dramatic increment. The best result is achieved by the *RushHour* sequence, where an increment up to $3,533\%$ is observed. A potential reason for this is that most PBs in *RushHour* are originally coded in larger size with slightly lower RDC. When a PB is split into 4 smaller PBs, the number of manipulable syntax elements also increases by 4. Furthermore, the increment observed in EL is larger than that of BL, and this observation is consistent across all video test sequences. This is because more blocks are split and coded using smaller size in EL. It is apparent that the proposed *MVP*-based technique outperforms the *MVD*-based technique for all settings of τ.

All in all, the proposed *MVP*-based technique is able to consistently embed more data while causing less bit rate overhead, with similar quality degradation when compared with the *MVD*-based technique. More data can be embedded by increasing τ, but at the expense of more distortion and larger bit rate overhead.

4.5 Summary

A technique to embed data during inter-picture prediction in SHVC is proposed. The parity bit of the MVP indices in multi-layer scalable coded video are manipulated to embed data. A threshold is also introduced to control the coded PB size, which in turns influences the payload. Experimental results suggest that the achieved payload for the proposed technique is encouraging, where about 118.9 kbps can be embedded with a drop of 0.06 dB in PSNR and bit rate overhead of 4.72%. As future work, the aim is to suppress bit rate overhead due to data embedding, and investigate into different ways to combine the proposed MVP and other data embedding techniques.

CHAPTER 5: MERGE MODE-BASED DATA EMBEDDING IN SHVC COMPRESSED VIDEO

5.1 Introduction

The advanced merge block coding tool introduced in SHVC enables motion data to be inherited from a spatial or temporal neighboring block/frame. The introduction of merge mode has improved the coding performance by sharing MVs with neighboring blocks. Specifically, only one set of MVs is required to be coded and transmitted for neighboring blocks that have similar motion. From the comparison conducted by Helle et al. (Helle et al., 2012) on blocks predicted by the different prediction mode, block merging yielded a RD performance gain ranging from −6% to −8% BD rate. It was observed that an average of 76% of the test sequences were coded with block merging, which is significantly more frequent than other prediction modes. The total number of CUs coded with block merging increases when the value of the QP increases. Hence, the potential of data embedding using merge mode is encouraging, and it can be utilized to complement the existing MV-based data embedding for improving the embedding capacity.

This chapter explores merge mode-based data embedding technique for SHVC compressed video. Inter-prediction block merging candidate selection decision is manipulated to embed data based on some pre-defined mapping rules. The main contribution of this work is utilizing merging block predictive syntax element introduced in HEVC and its extension to increase payload capacity while maintaining perceptual quality of the video. Experimental results suggest that encouraging payload is achieved at the expense of slight bit rate increment and negligible degradation in perceptual video quality. In the best case scenario, the sequence *PartyScene* can embed 84.45 kbps with an average 1.1% bit rate overhead for the Low Delay B (LDB) configuration.

5.2 Proposed Data Embedding Technique

In this work, the inter-prediction block merging candidate selection decision is manipulated based on some pre-defined mapping rules. Recall that spatial and temporal neighboring blocks or frames containing object with similar motion are likely to be predicted by using the same MV. Hence, merge or skip mode in SHVC reuses motion information from these neighboring blocks. Specifically, a list of merge candidates is generated from these spatial and temporal neighboring blocks or frames. Then, a flag is used to indicate whether motion information from the neighboring block is coded for a particular prediction block. When merge or skip mode is coded, an index is utilized to indicate the candidate block where motion data is derived from and no MV is coded for the prediction block. For example, given a scene with a moving object in the foreground, the foreground object can be grouped into several regions, with each having similar motion. Each region with similar motion can be predicted using same MV. Instead of coding MV for each block in a region, only one MV and flags which used to indicate the merge mode is required to be coded, which improves the coding efficiency. An illustration of possible merge block is depicted in Figure 5.1, where the merge blocks are marked with blue borders.

To embed data, members in merging candidate list are divided into two groups, where one is associated with bit ‘0’ and the other associated with bit ‘1’. An example of the mapping rule is shown in Figure 5.2. Naturally, RDO decides the best merge candidate block from the candidate list for coding. To determine the merge candidate block for the current block (*CurrBlk*), only the neighboring blocks which are mapped to the payload bit m (dependent on mapping rule) are considered, where the one with the best RDC is coded instead of the merge candidate block determined by RDO. For instance, if the merging block decided by RDO is A_0 and $m = 1$, then the proposed scheme will force the RDO to

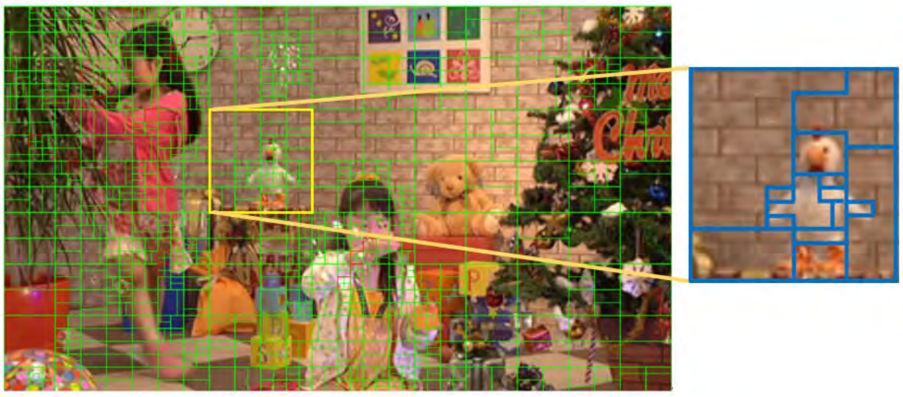


Figure 5.1: Quad-tree partitioning and merging block structure.

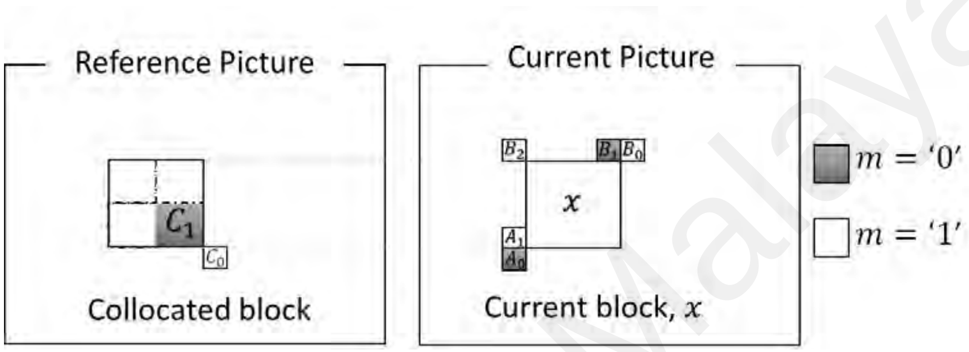


Figure 5.2: Example of mapping rule for merge candidates.

recalculate the required RDC for the neighboring block A_1, B_0, B_2 and C_0, then chooses the best candidate that results in the lowest cost.

The proposed merge mode-based technique follows the steps below to embed data into a SHVC video and the payload bits are embedded in all scalable coded layers:

1. Initialize best RDC to MAX_{double}.
2. Construct a list of merging candidates from spatial, temporal and inter-layer candidates as shown in Figure 5.2.
3. For each merging candidate, check if the candidate block is mapped to m. If YES then go to step 4, otherwise proceed to the next merging candidate.
4. Perform motion compensation and residual prediction, then calculate RDC.
5. If the RDC is less than the best RDC, then set the current candidate as the best merge candidate.

6. Repeat step 3∼ 5 until the best merge candidate is selected for the PU.

During decoding, the embedded data can be extracted from the prediction blocks based on the same predefined mapping rule for the merge mode candidates. The advantage of manipulating merge block for data embedding is that the bit rate overhead can be kept to the minimum while payload can be increased because the number of blocks associated with skip or merge CUs is > 50% as compared to other prediction mode (Helle et al., 2012). Furthermore, the perceptual quality of the processed video can be maintained without drift-error.

5.3 Experimental Results

The proposed data embedding technique is implemented by modifying the SHVC reference software SHM-12.0 (Joint Collaborative Team on Video Coding, 2017). Five standard video test sequences (i.e., *RushHour* @ 30 Hz (960×540, 1280×720), *FourPeople* @ 60 Hz and *BlueSky* @ 24 Hz (640 × 360 , 1280 × 720), and *BasketballDrill* @ 50 Hz and *PartyScene* @ 50 Hz (416 × 240, 832 × 480)) are utilized for evaluation. Experiments are conducted using two layers spatial scalability with GOP size of 4. The remaining parameters are set to the SHM's default scalable configuration for LDP and LDB. A pseudo-random number generator is employed to generate a sequence of binary numbers (i.e., 0's and 1's), which is then embedded as the payload. To analyze video quality degradation and bit rate overhead, the PSNR, SSIM (Z. Wang et al., 2004) and bit rate are computed for each video sequence.

5.4 Discussion and Analysis

For comparison purpose, the experimental results are compared with the *MVP*-based (Pang & Wong, 2019) and *MVD*-based (Van et al., 2015) data embedding technique. Note that (Pang & Wong, 2019) is based on MVP, and it manipulates the indices of

Table 5.1: Increment in bit rate (%) after embedding data.

Video Sequence	Frame Rate (Hz)	LDP	LDB	MVD	MVP
RushHour	30	1.35	2.37	0.86	0.42
FourPeople	60	0.97	2.44	0.34	0.12
BlueSky	24	0.83	1.48	0.89	0.23
BasketballDrill	50	1.10	2.05	0.75	0.30
PartyScene	50	0.65	1.11	0.80	0.29

motion vector predictor candidates of the current block while the MVD-based technique manipulates the difference between motion vectors to embed data.

5.4.1 Bit Rate Variation

The impact of data embedding to the bit rate for the proposed merge mode-based data embedding (labelled as LDP and LDB) and the previous method (Pang & Wong, 2019) is shown in Table 5.1. It is observed that the bit variation falls in the ranges of $[0.7, 1.1]$ and $[1.1, 2.4]$ percents for the LDP and LDB configurations, respectively. The increased in bit rate is due to neighboring blocks associated to the data bits, which has slightly higher RDC, is selected for coding. The bit variation is higher for LDB because it manipulates additional candidates in bi-prediction slices. The sequence *PartyScene* has the lowest bit variation while the largest variation is observed in *RushHour*. Specifically, the sequence *PartyScene* is coded with more merge modes than other sequences because it contains more fine details with similar motion, which are more cost-effectively to be coded by using smaller block sizes and merge mode. On the other hand, the sequence *RushHour* is originally coded in larger block sizes. Modifying the merging candidate for larger blocks generally causes higher RDC. Particularly, it is observed that more blocks are coded using *MVD* or intra-picture prediction mode for the sequence *RushHour*. Results suggest that the overhead for *MVD*-based and *MVP*-based falls in the range of $[0.3, 0.9]$ and $[0.1, 0.4]$, respectively, which is slightly lower when compared to the proposed technique. This is mainly due to their low embedding capacity as suggested in Table 5.3. Another reason is

Table 5.2: Video quality degradation in term of PSNR (dB) and SSIM for the processed videos.

Video Sequence	**PSNR (10^{-2}dB)**				**SSIM (10^{-4})**			
	LDP	**LDB**	**MVD**	**MVP**	**LDP**	**LDB**	**MVD**	**MVP**
RushHour								
BL (960×540)	1.00	1.00	2.00	1.00	0.00	0.00	0.00	0.00
EL (1280×720)	1.00	2.00	1.00	0.00	0.00	0.00	0.00	0.00
FourPeople								
BL (640×360)	2.00	1.00	1.00	1.00	1.00	0.00	1.00	0.00
EL (1280×720)	2.00	1.00	1.00	0.00	0.00	0.00	1.00	0.00
BlueSky								
BL (640×360)	1.00	0.00	4.00	1.00	0.00	0.00	1.00	1.00
EL (1280×720)	1.00	1.00	3.00	1.00	0.00	0.00	0.00	0.00
BasketballDrill								
BL (416×240)	1.00	1.00	2.00	1.00	1.00	1.00	0.00	0.00
EL (832×480)	1.00	2.00	1.00	1.00	0.00	1.00	3.00	0.00
PartyScene								
BL (416×240)	0.00	1.00	2.00	1.00	0.00	0.00	0.00	0.00
EL (832×480)	1.00	0.00	1.00	0.00	0.00	0.00	5.00	0.00

due to the *MVD*-based and *MVP*-based data embedding techniques only manipulate the blocks of the smallest size.

5.4.2 Video Quality

Table 5.2 shows the difference in quality between the original and processed videos in terms of PSNR and SSIM. Results suggest that the proposed technique embeds data without compromising the perceptual quality in both BL and EL for all video sequences. The PSNR drops by $\leq 0.04\%$ for all cases, while the drop in SSIM value is insignificant. In the best case scenario, the perceptual quality for the sequence *PartyScene* drops at most by 0.01 dB. On the other hand, in the worst case scenario, PSNR drops by 0.02 dB for EL in the sequence *FourPeople*. For *MVD* and *MVP*-based techniques, results suggest that the PSNR falls in the ranges of $[0.01, 0.05]$ dB and $[0.00, 0.01]$ dB, respectively. When comparing between the proposed technique and *MVP*-based technique, mixed results are observed for different videos and parameter settings. Nonetheless, both techniques show comparable video quality after embedding data, and the degradation is negligible.

Table 5.3: Comparison of embedding capacity (kbps).

Video Sequence		LDP	LDB	MVD	MVP
	BL	18.1	22.2	8.3	8.7
RushHour	EL	21.0	35.6	3.1	3.4
	Total	39.1	57.7	11.4	12.1
	BL	19.9	25.1	4.0	4.4
FourPeople	EL	64.0	71.4	9.2	10.3
	Total	83.8	96.5	13.1	14.7
	BL	11.5	15.3	6.6	7.8
BlueSky	EL	43.5	50.8	22.0	26.0
	Total	54.9	66.1	28.6	33.9
	BL	11.8	13.8	5.6	6.3
BasketballDrill	EL	35.2	39.8	14.3	16.2
	Total	47.0	53.6	19.9	22.5
	BL	17.1	18.4	9.5	11.0
PartyScene	EL	66.8	66.0	38.2	45.5
	Total	83.9	84.4	47.8	56.5

5.4.3 Payload

Table 5.3 records the payload for both the LDP and LDB configurations. Results suggest that the proposed merge mode-based data embedding technique achieves higher payload when compared to the *MVP*- and *MVD*-based data embedding techniques at the expense of some bit rate overhead. This is because there are more prediction blocks associated with skip or merge mode, which can provide more syntax elements for manipulation. Besides that, skip or merge mode in all the blocks size are utilized for data embedding. In contrast, only the smallest blocks are manipulated in the *MVP*- and *MVD*-based techniques. In the best case scenario, the sequence PartyScene embeds up to 84.45 kbps with the perceptual quality degradation 0.01 dB and average 1.1% bit rate overhead for the LDB configuration. A potential reason is that there are many fine regions in the video sequence resulted in more small blocks being coded. It is apparent that the proposed merge mode-based technique outperforms the *MVD*-based and *MVP*-based techniques in term of payload capacity. All in all, the proposed technique is able to consistently embed more data while causing slight bit rate overhead, with similar quality degradation when compared with the *MVD*-based and *MVP*-based techniques, which offers lower embedding capacity.

5.5 Summary

A merge mode-based data embedding technique is proposed for SHVC video. The members in merging candidate list are divided into two groups, where one is associated with bit ‘0’ and the other associated with bit ‘1’. Merging candidate is chosen based on the predefined mapping rules to embed data without significantly compromising the perceptual video quality. Experimental results show that payload can be increased at the expense of higher bit rate as compared to MVD-based and MVP-based techniques. In the best case scenario, the sequence *PartyScene* can embed up to 83.9 kbps of payload with an average 0.7% bit rate overhead for the LDP configuration, while the same video can embed up to 84.5 kbps with an average 1.1% bit rate overhead for the LDB configuration. As future work, the plan is to jointly utilize the other predictive syntax elements in SHVC to further improve of payload and refine this work for actual application.

CHAPTER 6: DATA EMBEDDING USING PREDICTIVE ELEMENTS AND THRESHOLD-CONTROLLED BLOCK SPLITTING IN SHVC VIDEO

6.1 Introduction

In a spatial scalable coded video, spatially co-located blocks at different coding layers code a scene at different resolutions. Specifically, the texture information in the decoded BL (used as a reference layer) is up-sampled to match the EL (coding layer) resolution. Therefore, the syntax elements from spatially co-located blocks, although differ from layer to layer, are expected to be similar to a certain extent. Inter layer prediction coding tool in SHVC exploits the correlation of the inter coded layers to improve coding efficiency in the EL. Within a CTU, there are mixture of inter- and intra-coded blocks coded at the optimal cost. Although data embedding can be achieved with MV and merge mode-based syntax elements, these approaches are not applicable to intra-coded blocks. Furthermore, the inter layer correlation in the scalable coded video is not fully exploited for data embedding purposes.

In this chapter, intra prediction mode data embedding method is put forward to complement the inter prediction-based methods using spatial scalable coded video. Although the use of inter-layer prediction, together with prediction modes and partitioning decision based on PU size and frame type increases the complexity of the encoder, the video perceptual quality of the video after data embedding can be preserved. Specifically, the encoder calculates the prediction error based on the original pixel value and predicted value using the modes associated with payload bits, therefore it can eliminate the drift-error. Here, data embedding using both intra prediction mode and MVP in each spatial scalable coded layer is put forward with adjustable rate-distortion guided split block to increase embedding capacity. This particular study makes the following contributions: (a) payload bits are associated to the intra mode and inter-mode predictor indexes in all scalable coded layers of a SHVC video without drift-error, and; (b) adopt an adjustable rate-distortion guided block splitting technique to allow more small blocks for data embedding purpose. Experiments are conducted to verify the performance of the proposed technique. Results suggest that the proposed technique can effectively embed data into spatial scalable coded video while maintaining the video quality. In the best case scenario, the sequence *PartyScene* can embed up to 516.9 kbps of payload with an average 7.0% bit rate overhead for the LDP configuration, while the same video can embed up to $1,578.6$ kbps with an average 2.9% bit rate overhead for the All Intra (AI) configuration.

6.2 Proposed Data Embedding Technique

In this work, predictive syntax elements in both intra- and inter-picture prediction are jointly utilized for data embedding purposes. The intra prediction mode and MVP in all scalable coded layers are utilized for data embedding. When a predictive syntax element is utilized for data embedding, the coding decision will be changed and results in different prediction error/residual. The encoder will then calculate and encode the prediction error caused by data embedding as a part of the prediction residual and reconstruct an approximation of the original video. Since data embedding is performed during the encoding process, and the prediction error is always calculated based on the predictive syntax elements after it is associated with the data, hence no drift-error will occur. Without loss of generality, the descriptions are based on a 2-layer setting in a scalable coded video. A simplified flow of data embedding processes is depicted in Figure 6.1.

6.2.1 Data Embedding

Let ϑ denote the intra prediction mode, p denote the parity of the intra prediction mode, and m denote the payload bit to be embedded. During intra prediction in BL, the payload is embedded into the smallest luma PB, viz., 4×4, by manipulating the prediction

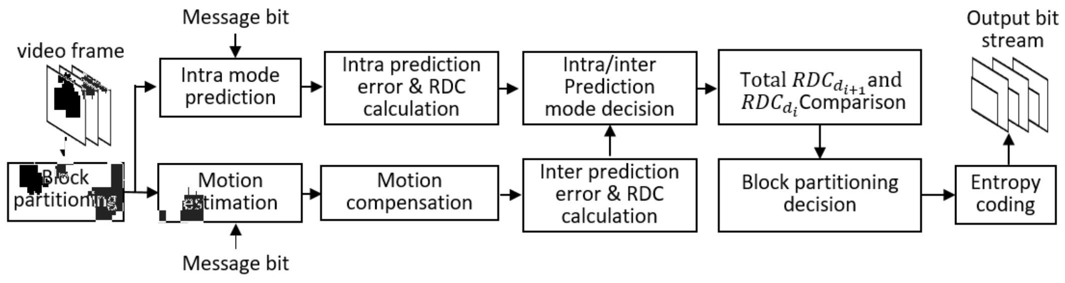

Message bit
video frame
Intra mode prediction
Intra prediction error & RDC calculation
Intra/inter Prediction mode decision
Total $RDC_{d_{i+1}}$ and RDC_{d_i} Comparison
Output bit stream
Block partitioning
Motion estimation
Motion compensation
Inter prediction error & RDC calculation
Block partitioning decision
Entropy coding
Message bit

Figure 6.1: Flow of processes in the proposed data embedding technique.

MPM

6	5	7
[0]	[1]	[2]

Candidate list

6	5	7	4	8
[0]	[1]	[2]	[3]	[4]

MPM

31	1	0
[0]	[1]	[2]

Candidate list

7	26	32	12	1
[0]	[1]	[2]	[3]	[4]

Figure 6.2: Examples of MPM and candidates with top performances.

mode. During intra coding, the RD optimizer determines the best intra prediction mode to be coded. When the payload bit to be embedded differs from the parity of the best prediction mode (i.e., $m \neq p$), the intra prediction mode with the opposite parity bit having the best RDC from the candidate list is selected in place of the originally recommended one. In other words, the prediction mode associated to the opposite parity bit is selected, and the bit stream is updated accordingly. For illustration purposes, consider the examples illustrated in Figure 6.2. In Example (a), MPM is a member of the candidate list. If $m = 0$, then there is no difference between the parity bit of the best candidate selected by RD optimizer and the message bit. Therefore, no changes are required, viz., mode 6 is selected and its index with reference to MPM (i.e., 0) is coded. On the other hand, if $m = 1$, then prediction mode 5, which is the best candidate with parity matching m, is selected. Hence, the index 1 in MPM is coded. On the other hand, consider Example (b) where MPM is not a subset of the candidate list. Suppose we want to embed $m = 0$ using the prediction mode. Then mode 26 will be treated as if it is the top performing candidate because its parity (i.e., 0) matches m. On the other hand, to embed $m = 1$, mode 7 is treated as if it is the best candidate because its parity matches the message bit. In both cases of Example (b), none of these prediction modes is found in MPM. Hence, a 5-bit fixed-length codeword is coded for each case. Based on the statistics compiled by Chernyak (Chernyak, 2014), the percentage of MPM being selected as the optimal prediction mode ranges from 55% to 72%. Therefore, there are opportunities to embed data using MPM in SHVC video. When the modes in MPM used for data embedding, it does not alter the number of bits required for coding as only the index of MPM is coded.

Based on the selection process detailed above, there can be a situation where none of the candidates in the list having the parity that matches the message bit. In such a situation, the encoder is made to predict and calculate the RDC for each intra prediction mode with its parity matching the message bit. Then, the candidate list is updated. In most cases, the second best prediction direction deviates slightly from the best intra prediction mode ϑ_{best}, viz., either $\vartheta_{best} + 1$ or $\vartheta_{best} - 1$. For example, when a payload bit m to be embedded in BL differs from the parity bit of the intra prediction mode p, the intra prediction direction angle is either increased or decreased to the next angle. The selection of optimum mode ϑ_{Best} is formulated below:

$$\vartheta_{\text{Best}} = \arg\min_{p=m}\{\text{RDC}(\vartheta, p)\}. \tag{6.1}$$

In spatial scalable coded video, spatially co-located blocks at different layers correspond to the same area within a slice (image). An example of spatially co-located blocks is shown in Figure 6.3, where the small boat is coded with 12 and 48 CU's in BL and EL, respectively. Since spatially co-located blocks refer to the same region in the same scene in different layers, they share similar textural information. Therefore, the syntax elements

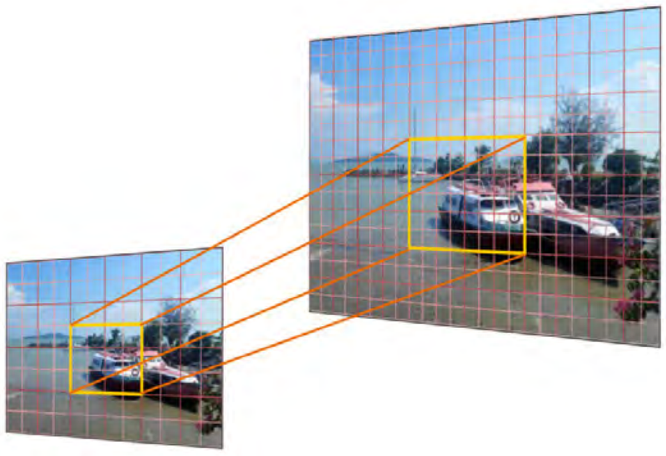


Figure 6.3: Illustration of spatially co-located blocks in the BL and EL.

from the spatially co-located blocks, although differ from layer to layer, are expected to be similar to a certain extent. Here, the spatial correlation of the spatially co-located blocks in BL and EL are exploited to embed data.

Inter-layer prediction in SHVC uses the reconstructed video signal from a reference layer (denoted by EL_k, for $k < c$ where $BL = EL_0$) to predict the current EL_c. It is introduced to improve the coding efficiency of the ELs. During inter-layer prediction, the co-located reconstructed blocks (viz., visual information) from the reference layer EL_k are tagged with reference indices in the (RPLs) in addition to other prediction information in the current encoding layer EL_c. In order to achieve inter-layer prediction, EL requires the reconstructed pictures from the BL Decoded Picture Buffer (DPB), which includes the reconstructed texture samples, and other prediction information (Boyce et al., 2016). Therefore, BL prediction information is available and can be retrieved from DPB of the reference layer for inter-layer prediction.

Exploiting this property, data embedding in ELs makes use of the inter-layer correlation of intra-picture prediction information for the spatially co-located blocks across layers. The intra prediction mode of the spatially co-located reconstructed block in reference layer EL_k is included as a candidate for the intra prediction mode in the EL_i. When the block is selected (i.e., when it is of size 4×4) for data embedding, the inter-layer intra prediction mode is set with the highest priority and labelled as the top performer in the candidate list so that it will be selected for coding. Recall that the intra prediction mode assumes an integer value in the interval of $[0, 34]$, hence up to 5 bits can be embedded, although a threshold parameter ξ can be introduced to limit the number of bits to be embedded. To embed data, the selected bits from intra prediction mode, namely $b_{\xi-1}b_{\xi-2}\cdots b_0$ is utilized to perform the bitwise-XOR operation with the message bits m_i from the message segment M, i.e., $b'_i = b_i \oplus m_i$. The resulting values b'_i are converted into decimal value. Although the data embedding results in additional prediction errors, encoder calculates prediction error /residual to encode the residual signal of a block by taking the difference between the original and predicted (embedded with payload bits) value to create almost identical compression artifacts without drift-error. The prediction error is then transformed, quantized and entropy coded. The proposed embedding in predictive domain has an advantage in term of selecting the venue for data embedding as compared to transform domain which apply a variation of LSB modification on quantized transform coefficients (Chang et al., 2014; Tew & Wong, 2014; Buhari et al., 2016; Gui & Xue, 2017; Long, Peng, & Li, 2018; Liu, Zhao, Liu, Feng, & Liu, 2018). The coefficient modification must be minimized in order to minimize the distortion as changes to more transform coefficients may result in artifact and being perceptible and notably degradation.

Naturally, if a spatial scalable encoded video contains more layers, then more bits can be embedded. Similarly, if both intra and inter-coded blocks are jointly utilized as data carrier, it provides more embedding capacity. Therefore, the MVP is jointly utilized as payload carrier. Recall that having more split blocks implies more embedding opportunities with the availability of the syntax elements. Therefore, an adjustable rate-distortion guided block splitting process is applied to increase the embedding capacity. The utilization of

MVP for data embedding and rate-distortion guided block splitting is detailed in Chapter 4. Note that visual quality degradation in the processed video can be kept to the minimum by restricting data embedding to the smallest blocks (i.e., 4×4 in intra mode as well as 8×4 and 4×8 in inter mode) while skipping the rest. When one follows such approach, the total number of PBs coded in the smallest size will directly affect the embedding capacity. However, larger block size may yield better bit rate but lower embedding capacity, and vice versa. Hence, there is a trade-off among embedding capacity, quality, and bit rate. Here, a random key can be utilized to select or skip blocks to distribute payload bits across the entire video.

6.2.2 Data Extraction

At the receiver's end, the embedded data can be extracted from the selected predictive syntax elements during decoding, by using the same key used during encoding, which are only known to the authorized parties. Firstly, the embedded data is retrieved from the intra prediction mode by checking the *parity* of the intra prediction mode (i.e., $0, 1, \cdots, 34$) in the identified block in the BL. In the EL, the spatial decoded mode of the spatially co-located block from the BL and the coded mode is utilized to derive the embedded data by using bitwise-XOR operation. Similarly, the parity bits of the MVP indices are extracted from the inter-coded blocks in all coded layers.

6.3 Experimental Results

The SHVC reference software SHM-12.0 (Joint Collaborative Team on Video Coding, 2017) is modified to implement the proposed data embedding technique. Here, the QP parameter settings are adopted as suggested in SHM test conditions (Seregin & He, 2014), i.e., QP = 22, 26, 30, and 34. Experiments are conducted by using two layers of spatial scalability for LDP and AI settings with GOP structure of 4, i.e., IPPPIPPP$\cdots$.

The remaining parameters are set to the SHM default configuration. Six standard video test sequences (i.e., *RushHour*, *FourPeople*, *PartyScene*, *BlueSky*, *BasketballDrill* and *RaceHorses*) from (Universität-Hannover, 2013; Xiph.org, 2013) are considered to evaluate the performance of the proposed data embedding technique. A pseudo-random number generator is employed to generate a sequence of binary numbers (i.e., 0's and 1's), which is then embedded as the payload by using both the intra- and inter-prediction techniques. Unless specified otherwise, all available embedding venues are utilized, i.e., embedding at the maximum rate.

Using SHVC compressed video as the baseline, the processed videos (containing embedded payload) are evaluated. It is verified that the bit streams can be decoded by SHVC decoders and the embedded data can be extracted correctly for all settings. All results are collected by processing the first 200 frames in each video test sequence. It is crucial to note that the AI setting is considered to capture the performance behaviour of embedding data using intra prediction mode, while the LDP setting is for embedding data using both intra prediction mode and motion vector prediction.

6.4 Discussion and Analysis

Manipulating video to carry additional payload bits may cause reduced coding efficiency and additional bit rate overhead, as well as affecting the visual quality of the processed video. The bit rate overhead and the perceptual quality is affected by payload capacity, video content as well as the venue/domain perturbed for data embedding. A higher embedding capacity (more payload bits) might lead to increment of bit rate overhead (e.g., due to additional prediction errors) and more severe visual quality degradation (e.g., due to drift propagation error), and vice versa. Hence, the experimental results in term of bit rate overhead, quality distortion and embedding capacity are measured and compared in the following subsections.

Table 6.1: Bit rate overhead (%) for the processed video.

Sequence	**QP**		**LDP**			**AI**	
		Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}
RushHour	22	0.4	6.5	21.0	0.3	3.3	8.1
@30 Hz	26	0.1	3.9	16.5	0.1	2.6	6.8
BL(960×540)	30	0.1	3.7	13.4	0.0	2.1	5.9
EL(1280×720)	34	0.0	4.6	13.0	0.0	2.1	5.7
	Average	0.1	4.3	15.2	0.1	2.4	6.3
FourPeople	22	1.3	8.9	16.3	1.2	3.7	6.7
@60 Hz	26	1.5	7.3	13.9	1.2	3.6	6.5
BL(640×360)	30	1.5	6.4	12.1	1.0	3.3	6.5
EL(1280×720)	34	0.3	6.0	13.3	0.5	3.1	6.6
	Average	1.2	6.8	13.3	1.0	3.5	6.6
BlueSky	22	0.5	4.6	8.0	0.6	3.3	6.2
@24 Hz	26	0.7	5.5	10.1	0.6	3.5	6.6
BL(640×360)	30	0.8	6.0	11.4	0.4	3.8	7.5
EL(1280×720)	34	0.5	7.2	15.2	0.0	4.3	9.0
	Average	0.7	6.0	11.6	0.4	3.8	7.6
BasketballDrill	22	0.8	7.0	13.2	1.1	3.9	7.1
@50 Hz	26	0.9	7.4	15.1	1.0	3.7	7.0
BL(416×240)	30	0.9	7.9	16.4	1.0	3.7	7.2
EL(832×480)	34	0.6	8.9	18.1	0.8	3.5	7.5
	Average	0.8	7.8	15.9	0.9	3.6	7.1
PartyScene	22	0.7	4.3	7.0	0.9	4.1	2.9
@50 Hz	26	1.1	5.7	9.7	1.1	3.4	4.9
BL(416×240)	30	1.2	7.2	13.4	1.2	3.9	5.9
EL(832×480)	34	0.6	9.9	19.1	0.9	4.6	7.6
	Average	1.0	7.2	13.3	1.0	4.0	6.0
RaceHorses	22	0.4	4.3	8.7	0.2	5.0	7.8
@30 Hz	26	0.5	5.3	11.1	0.3	4.9	8.3
BL(416×240)	30	0.5	6.5	14.9	0.3	5.2	9.5
EL(832×480)	34	0.4	9.8	20.7	0.0	5.9	11.6
	Average	0.5	6.8	14.8	0.3	5.2	9.6

6.4.1 Variation in Bit Rate

Let Υ_0, Υ_5 and Υ_{10} refer to the setting when Υ = 0, 5 and 10%, respectively. Table 6.1 records the bit rate before and after data embedding by using the proposed technique. Results suggest that, for both the LDP and AI configurations, the bit rate increases as Υ increases. This is an expected results because larger Υ permits more split blocks to be coded (hence increasing embedding capacity), instead of larger blocks which are more cost effective. The lowest average bit variation is observed in *BlueSky* for LDP configuration and *PartyScene* for AI configuration. A reason for such outcome is that these sequences are originally coded with 80 ∼ 90% of small blocks at partitioning depth level $d = 3$ in both layers. On the other hand, the largest average bit variation is observed in *BasketballDrill* for LDP configuration and *RaceHorses* for AI configuration. A reason for such outcome is due to large homogeneous region in both video sequences. These smooth regions can be effectively coded by using larger blocks, but instead they are partitioned into smaller blocks for increasing embedding capacity. Based on these observations, it is concluded that: (a) video sequence with textured scene offers higher payload, and; (b) the total number of the smallest sized PB to be coded can be controlled by using Υ so that just sufficient capacity is offered to accommodate the data, without over-supplying, which leads to higher bit rate.

6.4.2 Video Quality

The effect of data embedding on video quality is reported in Table 6.2. In all cases, the average PSNR degradation is less than 0.17 dB and 0.13 dB for the LDP and AI configurations, respectively. In terms of SSIM (Z. Wang et al., 2004), the degradation is less than 0.0027 and 0.0022 for the LDP and AI configurations, respectively. The sequence *PartyScene* experiences slightly greater degradation due to its high embedding capacity for all Υ values considered. On the other hand, the lowest average quality degradation is observed in the sequence *BlueSky* and *RushHour* for AI configurations, respectively. It is also observed that video coded by using AI configuration is slightly inferior in terms of video quality. It is mainly caused by the significantly higher payload carried by the all-intra coded sequence. Nonetheless, the results for both configuration settings show comparable video quality after embedding data, and these outcomes suggest that the manipulation of PB size has negligible impact on the video quality for all video sequences.

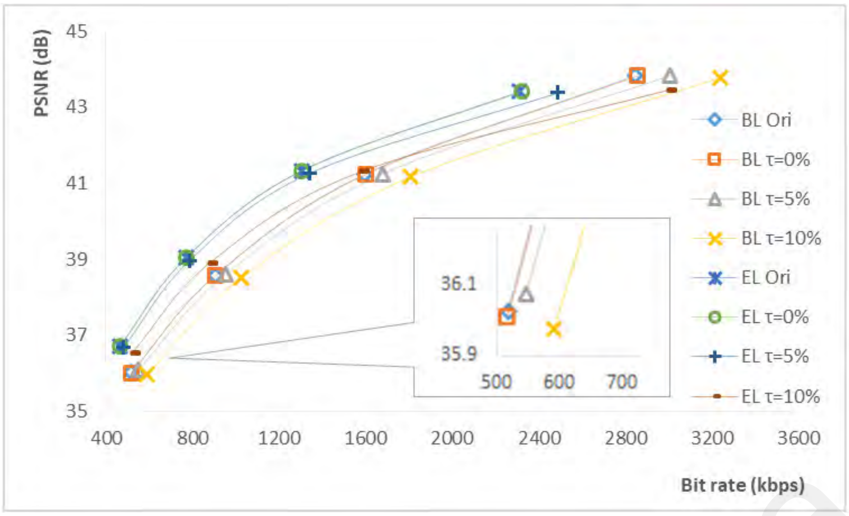
To visualize the performance of the proposed technique, the RD curves comparing the original (compressed) and processed videos for the LDP and AI configurations are plotted in Figure 6.4 and Figure 6.5, respectively. Results suggest that data embedding leads to a slight drop in coding efficiency. The graphs also suggest that implementing the proposed technique at higher QP causes greater loss in coding efficiency. Generally, coding

Table 6.2: Visual quality degradation in term of PSNR (dB) and SSIM for the processed video.

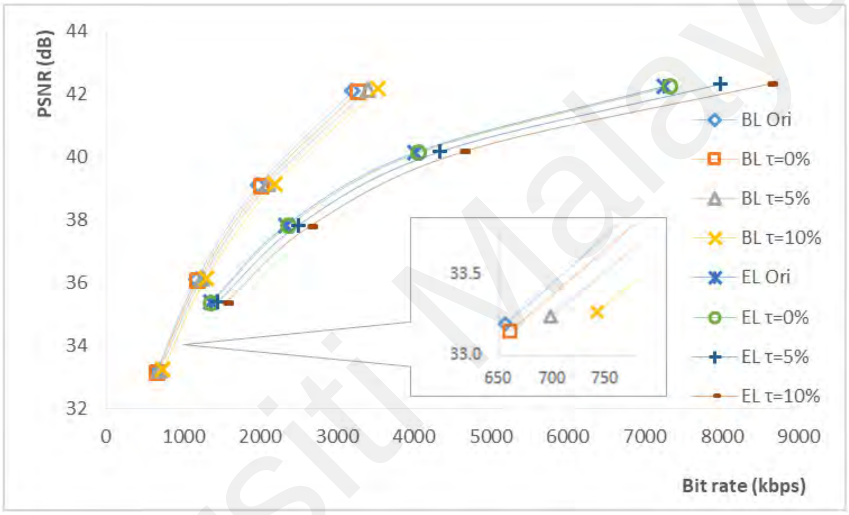
			PSNR (10^{-2})						**SSIM(10^{-4})**					
Sequence	QP		**LDP**			**AI**			**LDP**			**AI**		
			Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}
RushHour	22	BL	0.6	0.0	5.4	3.5	0.0	1.3	0.2	0.0	0.3	0.5	0.0	0.0
		EL	0.0	3.2	0.0	1.1	0.0	0.0	0.0	1.0	0.0	0.1	0.0	0.0
	26	BL	0.6	0.0	7.1	3.6	0.0	0.0	0.0	0.0	1.2	0.9	0.0	0.0
		EL	0.8	8.0	2.8	1.3	0.0	0.0	0.2	3.4	2.1	0.1	0.0	0.0
	30	BL	0.7	0.0	7.9	2.7	0.0	0.0	0.2	0.0	3.1	0.7	0.0	0.0
		EL	0.0	6.9	13.1	1.1	0.0	0.0	0.1	4.3	10.6	0.2	0.0	0.0
	34	BL	1.6	0.0	5.3	1.8	0.0	0.0	2.1	0.0	0.9	0.0	0.0	0.0
		EL	0.2	2.2	16.4	0.7	0.0	0.0	1.0	2.7	19.7	0.0	0.0	0.0
FourPeople	22	BL	3.9	0.0	0.0	10.3	6.4	8.1	1.4	0.0	0.0	3.0	1.3	1.4
		EL	0.2	0.0	0.0	2.5	0.3	2.5	0.0	0.0	0.0	0.2	0.0	0.0
	26	BL	2.6	0.0	0.0	9.9	6.3	8.6	1.6	0.0	0.0	5.3	2.6	3.0
		EL	0.6	0.0	0.0	3.3	0.3	2.8	0.1	0.3	0.1	0.3	0.0	0.0
	30	BL	4.1	0.0	0.0	10.4	4.5	4.8	3.0	0.0	0.0	9.9	2.1	0.1
		EL	1.4	0.3	3.9	3.5	0.0	1.6	0.0	1.0	2.2	0.4	0.0	0.0
	34	BL	4.4	0.0	0.0	9.2	2.1	1.6	6.4	0.0	0.0	13.2	0.0	0.0
		EL	2.5	0.8	1.1	3.8	0.0	0.0	0.3	1.6	2.2	0.9	0.0	0.0
BlueSky	22	BL	1.0	0.5	5.9	6.3	0.0	1.1	0.5	0.0	0.0	2.7	0.4	0.1
		EL	0.2	0.0	4.6	2.8	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0
	26	BL	0.4	0.0	4.0	5.2	1.8	8.4	0.4	0.0	0.0	4.1	0.4	0.9
		EL	0.0	0.0	2.8	3.2	0.0	3.1	0.0	0.0	0.0	0.4	0.0	0.0
	30	BL	0.4	0.0	5.7	5.2	3.3	12.0	0.6	1.5	2.3	6.6	1.1	2.2
		EL	0.0	0.0	6.7	3.6	0.0	3.4	0.0	0.0	0.0	0.8	0.0	0.0
	34	BL	0.4	0.0	0.2	5.7	2.1	9.5	0.3	0.0	1.7	10.0	0.6	2.2
		EL	0.2	0.0	0.0	4.3	0.0	0.0	0.0	0.0	0.0	1.2	0.0	0.0
BasketballDrill	22	BL	2.9	0.0	3.5	7.4	5.6	7.7	2.5	0.3	4.1	4.4	4.1	5.9
		EL	1.1	0.4	5.6	4.1	1.4	1.4	0.0	0.0	1.1	1.4	0.0	0.0
	26	BL	3.0	1.4	3.9	6.3	4.7	6.6	3.5	1.7	5.7	6.1	6.0	6.2
		EL	0.4	0.9	4.4	4.4	1.9	1.7	0.0	0.4	3.3	1.9	1.1	0.3
	30	BL	0.4	0.0	1.8	7.1	4.2	6.2	2.9	0.0	0.0	8.6	5.7	7.2
		EL	0.9	0.9	3.0	3.9	0.5	1.1	1.5	0.9	0.0	2.3	0.0	0.9
	34	BL	1.5	0.0	0.0	7.0	3.4	3.3	3.1	0.0	0.0	9.9	2.6	0.0
		EL	0.1	0.0	1.7	3.8	0.0	0.0	0.0	0.0	0.0	1.9	0.0	0.0
PartyScene	22	BL	1.9	0.0	4.0	13.0	7.1	6.2	0.6	0.0	2.8	4.8	2.1	2.6
		EL	0.7	0.0	2.0	6.4	0.0	0.0	0.0	0.0	0.4	1.0	0.0	0.0
	26	BL	1.3	0.1	4.1	12.2	5.3	6.2	1.2	1.3	7.5	8.3	3.4	4.0
		EL	0.0	0.0	1.4	5.5	0.0	0.0	0.0	0.0	1.2	1.8	0.0	0.0
	30	BL	1.5	0.6	4.6	10.7	3.2	3.7	2.0	7.3	20.6	15.0	4.1	4.0
		EL	0.5	0.0	1.0	5.4	0.0	0.0	0.0	0.0	1.3	2.7	0.0	0.0
	34	BL	3.8	0.0	4.1	9.9	1.2	0.7	11.3	8.7	27.0	21.6	5.4	4.6
		EL	0.9	0.0	0.0	6.1	0.0	0.0	0.0	0.0	0.0	5.2	0.0	0.0
RaceHorses	22	BL	1.3	0.3	6.1	7.4	0.0	1.8	1.3	0.0	2.5	3.8	0.3	0.0
		EL	0.1	2.6	5.5	4.2	0.0	0.0	0.0	0.7	0.6	0.3	0.0	0.0
	26	BL	1.4	1.1	6.0	6.7	1.0	3.7	8.2	3.9	10.0	4.7	0.9	2.4
		EL	0.0	1.4	5.7	4.1	0.0	0.0	0.6	0.4	2.7	0.9	0.0	0.0
	30	BL	1.5	0.0	0.7	6.0	0.7	0.6	0.0	0.0	0.0	7.4	0.2	0.0
		EL	0.3	1.0	4.0	4.1	0.0	0.0	0.0	6.8	8.0	1.7	0.0	0.0
	34	BL	1.1	0.0	0.0	5.6	0.0	0.0	0.0	1.6	6.1	16.1	4.5	0.0
		EL	0.2	0.0	0.0	4.7	0.0	0.0	0.0	0.0	0.0	5.2	0.0	0.0

efficiency is content-dependent and may differ for different content for a given QP value.

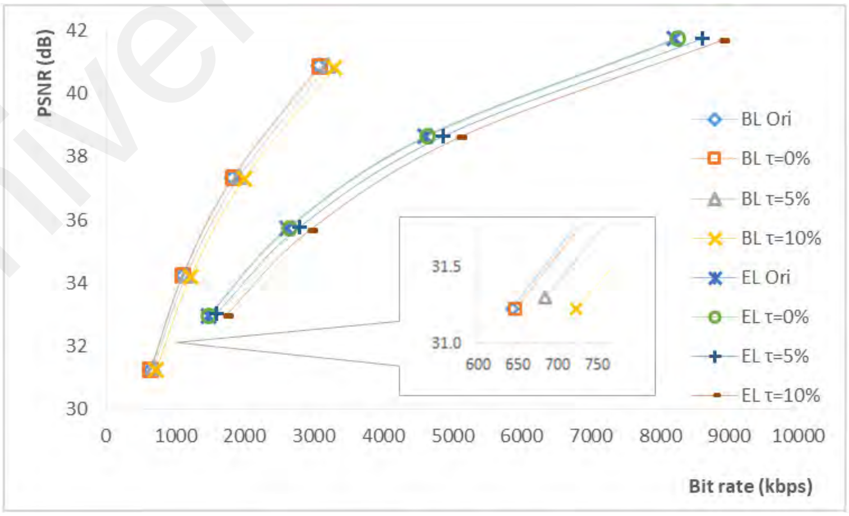
Hence, the rate distortion curves vary depending on the video content. Particularly, motion,



(a) RushHour

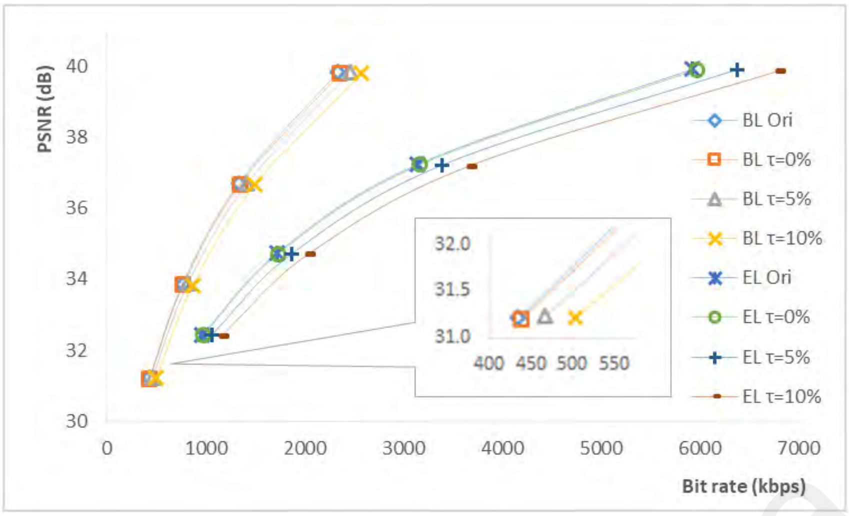


(b) FourPeople

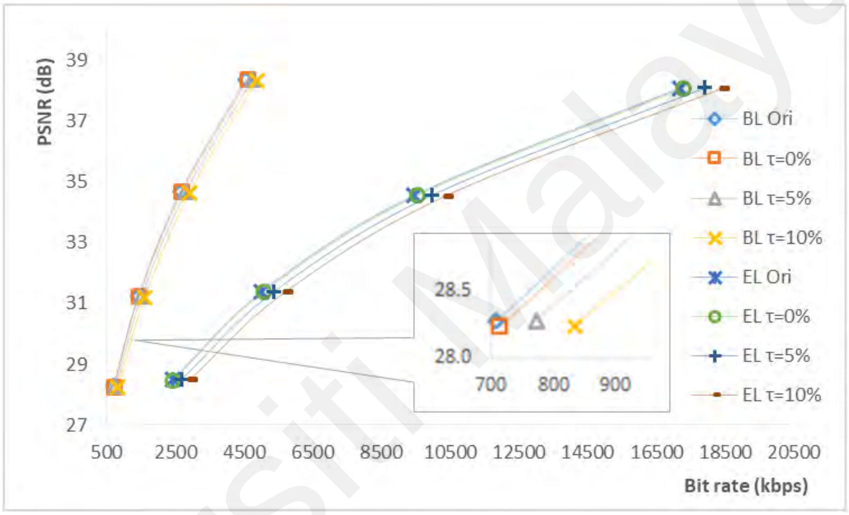


(c) BlueSky

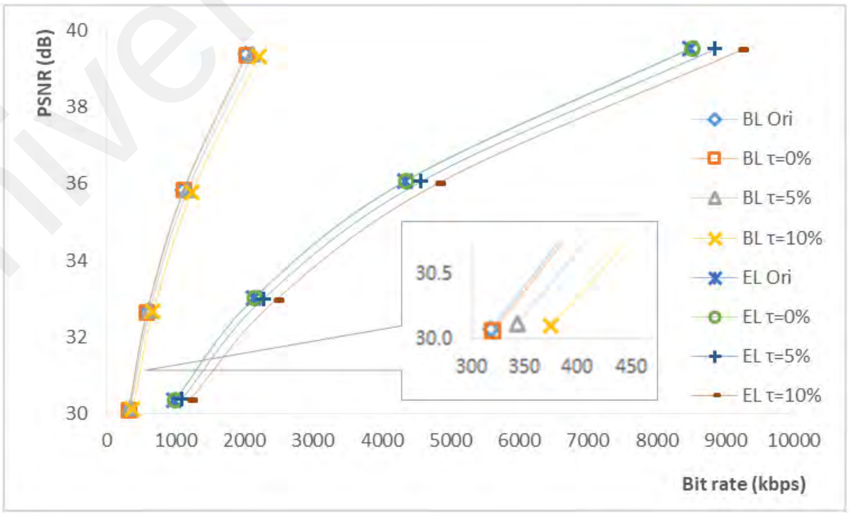
Figure 6.4: Rate distortion curve for the original compressed video (ORI) and proposed technique using LDP configuration.



(d) BasketballDrill

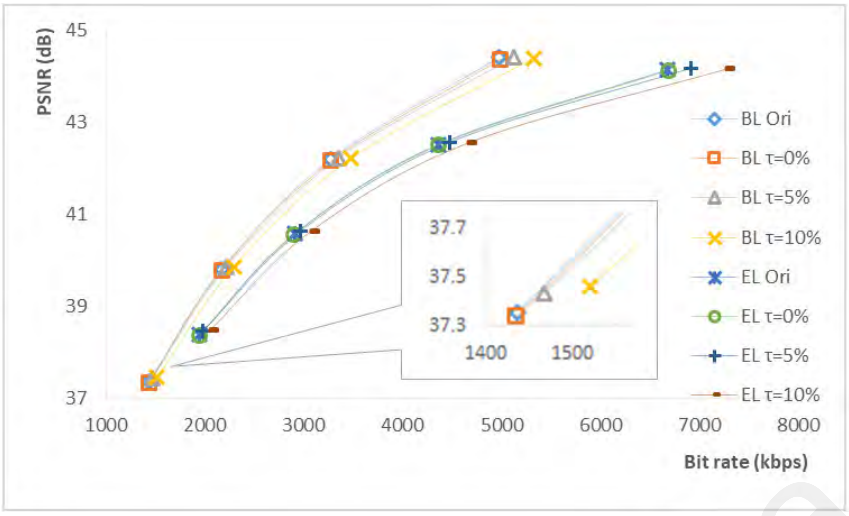


(e) PartyScene

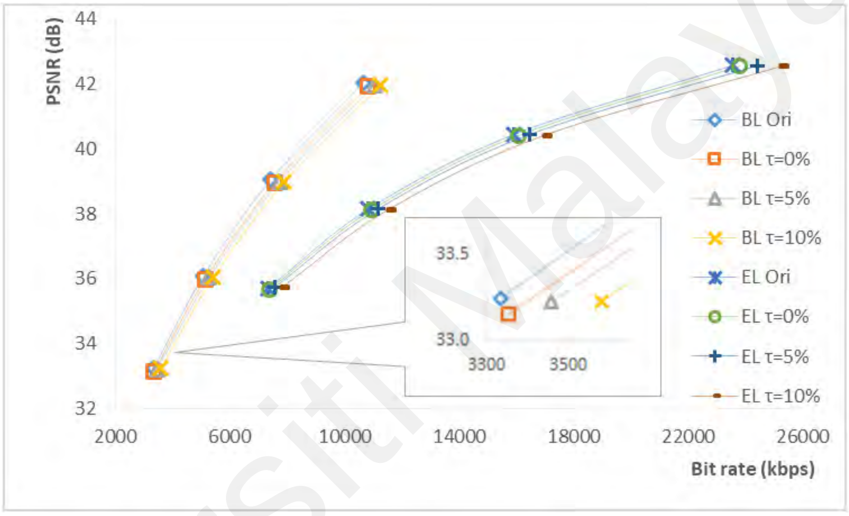


(f) RaceHorses

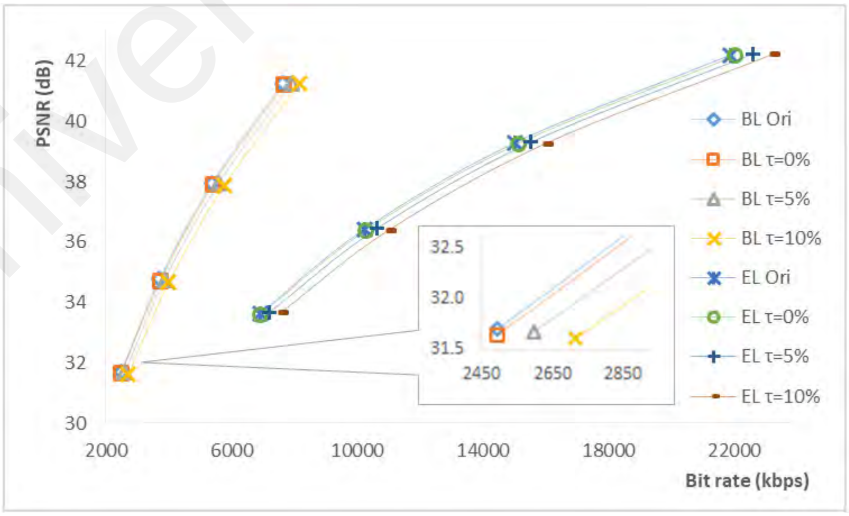
Figure 6.4: (Continued).



(a) RushHour

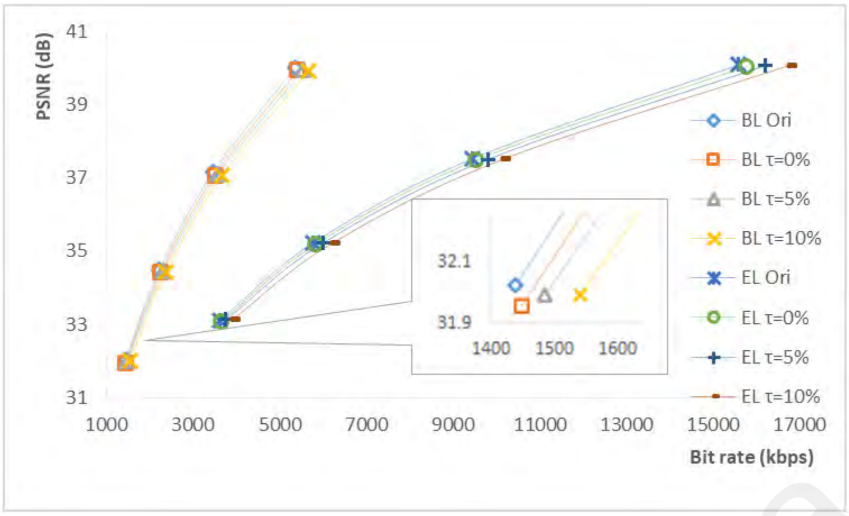


(b) FourPeople

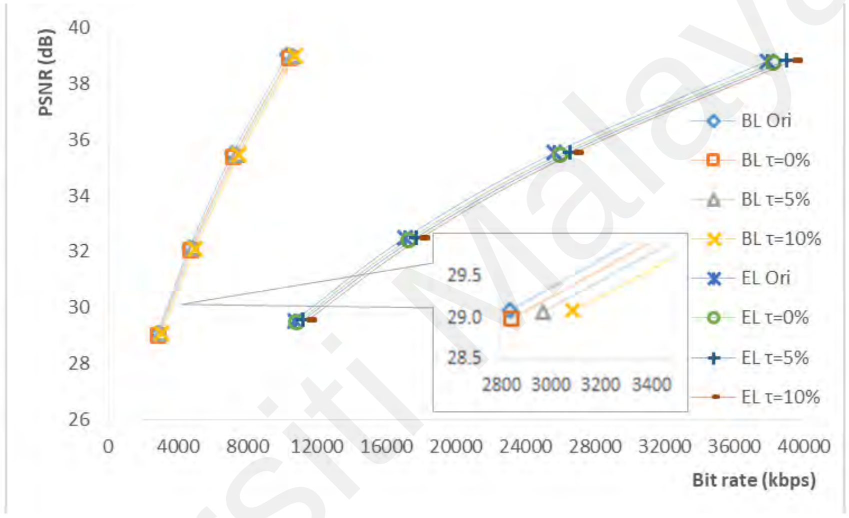


(c) BlueSky

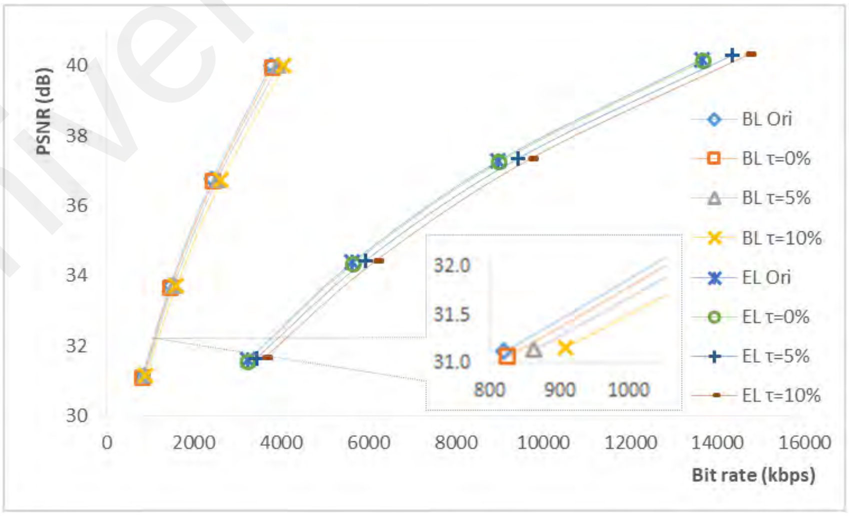
Figure 6.5: Rate distortion curve for the original compressed video and proposed technique using AI configuration.



(d) BasketballDrill



(e) PartyScene



(f) RaceHorses

Figure 6.5: (Continued).

Table 6.3: Comparison in term of BD-BR and BD-PSNR for the processed video.

	BD-BR (%)						**BD-PSNR(dB)**					
Sequence	**LDP**			**AI**			**LDP**			**AI**		
	Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}	Υ_0	Υ_5	Υ_{10}
RushHour	0.26	4.76	17.63	0.44	1.42	5.35	0.01	0.20	0.70	0.02	0.07	0.27
FourPeople	1.91	6.23	12.30	2.11	3.86	7.13	0.10	0.29	0.55	0.14	0.26	0.47
BlueSky	0.71	5.17	11.36	1.01	3.73	7.99	0.04	0.28	0.61	0.08	0.30	0.63
BasketballDrill	1.13	7.42	15.83	2.02	4.10	7.58	0.05	0.33	0.68	0.11	0.22	0.40
PartyScene	1.34	6.23	12.04	2.17	3.87	5.84	0.07	0.32	0.60	0.16	0.29	0.43
RaceHorses	0.73	5.99	13.89	1.31	4.39	8.53	0.03	0.27	0.61	0.08	0.26	0.49
Average	1.01	5.97	13.84	1.51	3.56	7.07	0.05	0.28	0.63	0.10	0.23	0.45

texture, activity and noise level in a video influence video compression efficiency. A video sequence with more static scenes (e.g., *FourPeople*) achieves higher quality when it is coded by using the same QP value. A video sequence with fine and complex textures with local motion (e.g., *PartyScene*) generate higher bit rate as compared to other sequences when it is coded by using the same QP value. The curves of *BlueSky* and *RushHour* show that higher PSNR value is generated. On the other hand, more dynamic motion in the video sequence (e.g., *RaceHorses*) requires more bits to provide the same video quality in comparison to video sequences with less dynamic motion (e.g., *BasketballDrill*). It is observed that the rate distortion curves for the video sequence *RushHour* in Figure 6.4 for 1.5x spatial scalability using the LDP configuration exhibit different characteristic. It is mainly due to more large blocks being coded in EL, which yields > 50% bit rate saving in EL, while most of the inter coded blocks are coded by using merge and skip modes (i.e., 4× more) for 1.5× spatial scalability as compared to 2× spatial scalability for the LDP configuration.

To further analyze the video quality degradation and bit rate overhead, Bjøntegaard Delta of bit rate (BD-BR) and Bjøntegaard Delta of PSNR (BD-PSNR) are computed (Bjøntegaard, 2001). The average bit rate overhead and visual quality degradation caused by data embedding are recorded in Table 6.3. The results suggest that the average bit rate overhead for the LDP and AI configurations fall in the ranges of $[1.01, 13.84]$ and $[1.51, 7.07]$ percents, respectively. On the other hand, the average quality degradation

Table 6.4: Embedding capacity (kbps) for the proposed embedding technique using LDP configuration.

Sequence	**QP**		Υ_0			Υ_5			Υ_{10}	
		BL	**EL**	**Total**	**BL**	**EL**	**Total**	**BL**	**EL**	**Total**
RushHour	22	15.7	6.7	22.4	42.9	36.1	79.1	79.5	138.8	218.3
	26	6.4	2.3	8.7	19.0	12.0	31.1	35.2	45.8	81.0
	30	2.3	0.7	3.0	7.7	4.4	12.2	15.7	17.5	33.2
	34	0.7	0.2	0.8	2.6	1.3	3.9	6.1	6.5	12.6
FourPeople	22	43.6	79.7	123.3	78.7	182.4	261.1	106.6	296.7	403.2
	26	28.7	49.2	77.9	54.0	107.0	161.0	78.3	178.1	256.4
	30	14.9	24.7	39.6	31.0	56.6	87.6	48.2	99.8	148.0
	34	5.0	8.1	13.1	12.1	20.7	32.8	22.1	43.4	65.6
BlueSky	22	21.5	78.9	100.4	47.5	174.1	221.6	66.3	242.4	308.7
	26	14.1	47.1	61.2	35.5	120.4	155.9	54.4	185.0	239.3
	30	7.5	22.3	29.8	25.4	78.7	104.0	41.1	126.8	167.9
	34	3.5	7.9	11.5	13.3	35.1	48.3	29.6	82.9	112.6
BasketballDrill	22	23.0	67.6	90.6	44.8	141.8	186.6	65.6	218.5	284.1
	26	14.8	34.9	49.7	28.7	72.2	100.9	44.1	113.0	157.1
	30	8.4	18.8	27.2	17.4	38.8	56.1	27.9	64.6	92.5
	34	3.5	7.9	11.4	8.2	18.6	26.9	14.9	33.5	48.4
PartyScene	22	43.9	190.5	234.4	86.7	341.2	427.9	108.6	408.4	516.9
	26	30.7	142.1	172.8	63.7	255.4	319.1	86.8	316.1	402.9
	30	17.4	91.2	108.6	39.5	172.7	212.2	59.5	225.5	285.0
	34	6.6	37.8	44.4	17.9	85.2	103.1	31.4	127.9	159.4
RaceHorses	22	17.7	44.6	62.3	36.1	128.9	165.0	54.9	184.8	239.6
	26	12.2	32.1	44.3	24.2	88.0	112.2	38.7	131.7	170.4
	30	6.9	18.6	25.5	13.9	51.0	64.9	22.5	81.8	104.3
	34	3.2	8.3	11.4	6.9	21.5	28.5	12.0	35.6	47.6

for LDP and AI configurations fall in the ranges of $[0.05, 0.63]$ dB and $[0.10, 0.45]$ dB, respectively.

6.4.3 Embedding capacity

The embedding capacity of each video using the proposed technique is recorded in Table 6.4 and Table 6.5 for the LDP and AI configurations, respectively. Generally, video sequences with complex texture or fast-moving scene (e.g., *PartyScene* and *BasketballDrill*), which are predicted by using more small PBs, can offer more rooms for data embedding, and vice versa (e.g., *RushHour*). In the best case scenario, the sequence *PartyScene* embeds 516.9 kbps with an average 7.0% bit rate overhead for the LDP configuration, while it embeds $1,578.6$ kbps with an average 2.9% bit rate overhead for the AI configuration. On the other hand, the sequence *RushHour*, which contains only a few fine detailed textures and

Table 6.5: Embedding capacity (kbps) for the proposed embedding technique using AI configuration.

Sequence	**QP**		Υ_0			Υ_5			Υ_{10}	
		BL	**EL**	**Total**	**BL**	**EL**	**Total**	**BL**	**EL**	**Total**
RushHour	22	28.3	41.9	70.1	81.7	139.3	221.0	163.4	311.5	474.8
	26	15.7	21.2	37.0	44.5	69.3	113.8	96.0	163.2	259.1
	30	7.1	9.3	16.3	21.2	30.4	51.6	48.8	77.4	126.2
	34	2.6	3.2	5.8	8.7	11.7	20.4	22.0	32.8	54.8
FourPeople	22	175.8	401.4	577.3	290.9	828.9	1,119.8	376.1	1,251.0	1,627.1
	26	138.7	304.7	443.4	235.5	602.1	837.7	315.5	921.0	1,236.5
	30	95.5	202.2	297.7	171.9	406.0	577.9	245.4	642.0	887.4
	34	52.9	112.4	165.3	106.4	238.8	345.1	165.7	409.4	575.1
BlueSky	22	57.2	301.8	359.0	126.3	607.7	734.0	181.3	835.0	1,016.3
	26	44.5	228.8	273.3	107.1	500.6	607.7	161.6	722.3	883.9
	30	30.5	153.8	184.3	87.2	388.5	475.8	141.9	608.9	750.8
	34	18.0	86.0	104.0	65.1	273.5	338.7	119.0	488.1	607.1
BasketballDrill	22	67.3	353.8	421.1	131.3	657.6	788.9	186.9	938.4	1,125.3
	26	47.6	198.8	246.4	89.7	374.7	464.4	131.3	570.7	701.9
	30	31.7	112.7	144.4	60.8	207.4	268.2	90.4	316.6	406.9
	34	18.9	65.0	83.9	38.8	122.3	161.1	62.5	187.9	250.4
PartyScene	22	135.3	752.2	887.5	233.5	1,125.0	1,358.5	271.6	1,307.1	1,578.6
	26	114.0	625.1	739.2	200.8	939.6	1,140.3	245.8	1,120.3	1,366.2
	30	83.0	498.7	581.7	157.0	756.3	913.2	206.3	919.6	1,125.9
	34	47.8	345.0	392.8	105.5	567.4	672.9	155.2	724.1	879.3
RaceHorses	22	32.9	134.5	167.4	75.3	383.0	458.3	116.2	525.7	642.0
	26	26.9	109.3	136.2	58.9	297.9	356.8	94.9	429.0	523.9
	30	19.0	83.6	102.6	42.0	221.9	263.9	71.6	337.7	409.3
	34	11.2	50.2	61.4	23.1	143.9	167.0	41.3	236.6	277.8

most objects are static or standstill, offers the least embedding capacity. The embedding capacity for this video sequence is expected to be small because more slices in the video are coded by using larger PB.

Results also suggest that when Υ is set to a larger value, higher embedding capacity is achieved. This trend is most apparent for the sequence *RushHour* because when the value of Υ is increased, more split blocks are coded instead of the block sizes suggested by the RD optimizer. Hence, a trade-off between embedding capacity and bit rate can be achieved by tuning Υ. It is observed that AI achieves, in general, a higher embedding capacity in comparison to LDP. The largest difference is observed in the sequence *FourPeople*, followed by *BasketballDrill*, *BlueSky*, *PartyScene*, *RaceHorses* and *RushHour*. Notably, for the sequence *FourPeople*, AI achieves, on average, $\sim 5.9\times$ more embedding capacity for Υ_{10} in comparison to LDP. This is because the sequence *FourPeople* contains 70%

static background scenes. On the other hand, for the sequence *RushHour*, AI achieves $2.2 \sim 4.3\times$ more embedding capacity in comparison to LDP for Υ_{10}.

6.4.4 Embedding Cost

For performance comparison, let's define the embedding cost, $\gamma = (b_e - b_o)/\rho$ as the number of bits spent on the video test sequence for encoding one payload bit, where b_o and b_e are the number of bits spent on coding (bit stream size) the original and processed video sequences, respectively, and ρ is number of payload bits embedded into the video sequence of interest. Based on the results, for Υ_0, on average, 0.71 and 0.35 bit are required to embed 1 bit for the LDP and AI configurations, respectively. In comparison, to embed one payload bit into the video, Aly et al.'s method for MPEG-2 compressed video requires 5.36 bits (Aly, 2011), while Noorkami et al.'s method for H.264 compressed video requires 1.50 bits (Noorkami & Mersereau, 2008). When Υ is increased, more bits are spent to embed each payload bit because split blocks might be coded (instead of a bigger block), which required some signaling to the decoder. In other words, the increase in bit stream size is due to data embedding (i.e., sub-optimal mode is selected hence larger prediction error) as well as additional signaling to the decoder. For example, for Υ_5, the embedding cost falls in the range of $[2.41, 6.25]$ and $[0.97, 2.27]$ for the LDP and AI configurations, respectively. Similarly, for Υ_{10}, the ranges are $[2.78, 7.06]$ and $[1.19, 2.48]$ for the LDP and AI configurations, respectively.

To investigate into bit stream size increment solely due to data embedding (i.e., choosing non-optimal prediction mode), the video test sequences are re-encoded (i.e., without embedding data bit) by using Υ_5 and Υ_{10}, which are then used as the baseline for comparison purposes. For Υ_5, the embedding cost is less than 0.54 and 0.56 for the LDP and AI configurations, respectively. Similarly, for Υ_{10}, the embedding cost is less than 0.48 and 0.66 for the LDP and AI configurations, respectively. These results suggest that the bit stream size increment due to data embedding is relatively small.

6.4.5 Comparison with Conventional Methods

For the purposes of fair comparison and analysis, the experiment environment and data set should be exactly the same, viz., using the same video encoder settings and the same video test sequences. However, there are three major challenges: (a) different video test sequences are used in the literature; (b) most experiments reported in the literature are conducted by using earlier video coding standards, and; (c) incomplete information about the parameter settings used for conducting experiments. These challenges prevent us from conducting and furnishing a thorough, fair and meaningful experiment using the conventional methods. Despite these challenges, a comparative analysis based on the available information is still provided for the completion of discussion. Table 6.6 records the embedding capacity for the proposed and conventional methods (Pang & Wong, 2019), (Mareen et al., 2018), (Pang, Wong, & Liong, 2017), (Van et al., 2015) using the same QP settings. It is observed that the proposed method (which uses multiple syntax elements) achieves higher embedding capacity in comparison to the conventional methods (which uses a single syntax element) considered. The embedding capacity is doubled for LDP when both MVP and intra prediction mode are utilized for data embedding. It is also observed that higher embedding capacity is achieved for AI configuration. This is because more prediction modes are manipulated for data embedding in the proposed method.

Table 6.6: Comparison of embedding capacity (kbps).

	LDP							AI		
Sequence	Υ_0			Υ_5		Υ_10				
	♦	■	**Proposed**	■	**Proposed**	■	**Proposed**	★	●	**Proposed**
RushHour	11.44	12.10	22.04	46.84	81.16	145.18	222.96	48.58	52.22	57.81
FourPeople	13.12	14.71	80.32	39.58	179.96	59.3	297.27	180.89	285.96	346.73
BlueSky	28.63	33.88	69.18	65.01	161.98	75.34	261.31	114.52	116.67	206.29
BasketballDrill	19.88	22.48	71.83	41.04	156.86	54.18	245.52	141.71	171.75	281.63
PartyScene	47.76	56.53	144.87	102.78	332.51	118.92	447.22	238.05	304.11	462.46

♦: (Van et al., 2015)
■: (Pang & Wong, 2019)
★: (Pang et al., 2017)
●: (Mareen et al., 2018)

Table 6.7: Relative functional comparison of conventional and proposed data embedding techniques within various video coding standards

Functional comparison	Embedding venue	Applicable to I-frame	Applicable to P/B frame	Embedding capacity	Quality after embedding	Bit rate overhead
1	Intra prediction mode	✓	✓	H	H	M
2	MVD	X	✓	L	H	L
3	Block partition	✓	✓	H	M	M
4	Coefficient	✓	✓	H	M	M
5	Merge mode	X	✓	M	H	L
6	Block partition + Coefficient	✓	✓	H	M	M
7	Intra prediction mode + MVP	✓	✓	H	H	M

H: High, M: Moderate, and L: Low
1:(Saberi et al., 2020; Mareen et al., 2018; Pang et al., 2017; Hu et al., 2007)
2:(Van et al., 2015; Aly, 2011)
3:(Y. Yang et al., 2019; Shanableh, 2018)
4:(Sun et al., 2021; Konyar et al., 2020; Buhari et al., 2016; Noorkami & Mersereau, 2008)
5:(Pang, Wong, & Ito, 2019)
6:(Tew & Wong, 2014)
7:*Proposed*

Next, a relative functional comparison is conducted by considering the conventional methods proposed for MPEG-2 (Aly, 2011), H.264/AVC (Hu et al., 2007; Noorkami & Mersereau, 2008), H.264/SVC (Buhari et al., 2016; Sun et al., 2021), and HEVC (Tew & Wong, 2014; Van et al., 2015; Mareen et al., 2018; Shanableh, 2018; Konyar et al., 2020; Y. Yang et al., 2019; Saberi et al., 2020).

Table 6.7 compares the aforementioned methods in terms of embedding venue, applicability to intra-/inter-prediction, embedding capacity, video quality and bit rate overhead. Here, a relative comparison is performed because the results and values reported by the respective authors in the existing works are collected by using different video test sequences, video coding standards, parameter settings, etc. The labels of high (H), moderate (M) and low (L) are context dependent and they are relative in nature. Firstly, for embedding capacity, we assign one of the three labels depending on the number available syntax elements in a video. Specifically, more syntax elements provides more venues for data embedding, and vice versa (Shen, Zhang, & Liu, 2014). While classical venues such as coefficient offers high embedding capacity, recently identified venues (e.g., MVD in HEVC) will, in general, offer less embedding capacity. Nonetheless, the proposed technique has the flexibility to embed different amount of payload by adjusting Υ. When the required embedding capacity is low, Υ_0 can be adopted so that the bit rate overhead and distortion can be reduced. On the other hand, more embedding capacity could be achieved by using a larger Υ so that each CTU can be split into smaller blocks. Note that the total number of smaller blocks in a CTU is determined by the RD cost, and it is guided by Υ so that block-splitting will not be performed when the cost is higher than Υ.

Next, the impact of data embedding on the quality of the video is analyzed. It is observed that the quality of the video may be affected depending on when (i.e., at which stage) data is embedded within the video encoding process as well as the embedding capacity. For example, when data embedding takes place before the computation of the prediction error/residual such as using intra prediction mode and MV, the residual will be obtained by using the new predicted value, therefore the quality of the video can be maintained, although the residual is slightly larger. On the other hand, manipulating coding block structure may affect the selection of the optimal prediction block size and prediction mode, which will indirectly cause higher prediction error. Therefore, the quality of the video is affected. Similarly, manipulating the transformed coefficients causes the entire block of pixel values to change. The perceived quality of these two methods will then be affected by the quantization process applied on the prediction residues. Similar findings are observed from the related works, for example, the perceptual quality degradation for the intra prediction mode and merge mode is less than 0.2 dB (Hu et al., 2007; Mareen et al., 2018; Saberi et al., 2020) while MVD (Aly, 2011) is less than 0.06 dB. Since the quality degradation is less than 1 dB, which is negligible, therefore the quality is labelled as ‘H’. In addition, it is observed that the quality degradation for coefficient-based methods (Noorkami & Mersereau, 2008; Buhari et al., 2016) and coding block structure-based methods (Y. Yang et al., 2019; Shanableh, 2018; Tew & Wong, 2014) fall in the range of $[0.2, 4.8]$ dB and $[1.0, 2.3]$ dB, respectively. Since the quality is slightly inferior in comparison to the former, ‘M’ is labeled for these methods.

Subsequently, the impact of data embedding on the bit rate based on these works is analyzed. Generally, when more bits are embedded into a video sequence, the bit rate overhead will be higher. Besides that, more bits are required to code the non-optimal block structure or predicted values, as well as additional syntax elements resulting from data embedding. Therefore, when more changes to the syntax elements or more syntax elements are coded due to data embedding, the bit rate overhead increases accordingly. These findings are consistent with the observation. Specifically, it is observed that the number of MVD is relatively low as compared to other syntax elements, and the bit rate overhead incurred is less than 1% (Aly, 2011). A coding block and its neighboring blocks are usually correlated with each other and they contain the same moving object with similar motion. The MV after mapping the merging block to the payload bit is likely to be maintained. Therefore, the bit rate overhead after manipulating merge mode can be maintained at a relatively low value (e.g., 2.3% as reported in (Pang et al., 2019)). Since the bit rate overhead is maintained at a minimum level, these methods are labeled as ‘L’. On the other hand, the bit rate overhead caused by using intra prediction mode to embed data in (Mareen et al., 2018) is capped at 4%, while the bit rate overhead caused by manipulating coding block structure falls in the range of [5, 5.5] in Yang et al.’s method (Y. Yang et al., 2019). In addition, the bit rate overhead caused by manipulating coefficient falls in the range of [1.4, 5.0] in Noorkami et al.’s method (Noorkami & Mersereau, 2008) and Buhari et al.’s method (Buhari et al., 2016). Since the bit rate overhead for these methods is relatively higher in comparison to the former methods, they are labeled as ‘M’.

In general, all considered methods embed data during the encoding process. However, depending on the eventual mode of encoding of a block (i.e., MB in H.264 or CU in HEVC/SHVC), most existing methods may or may not be able to embed data. For example, the intra picture prediction-based method will not be able to embed data into a block when the inter picture prediction mode turns out to be more cost effective (hence the block is coded by the using inter prediction mode), and vice versa. On the other hand, the proposed method is able to embed data into both intra and inter predicted blocks. In other words, eventually one mode will be chosen (i.e., intra or inter), and in any case, data can be embedded by the proposed method. Similar to the proposed method, Tew et al.'s method (Tew & Wong, 2014) can embed data in each CU since block size is exploited, while Shanableh (Shanableh, 2018) offers the same capability by exploiting the block-splitting flag.

Since technology constantly advances to newer standards, we also evaluate the feasibility of applying the conventional data embedding methods to the SHVC standard. The MVD in (Aly, 2011) is not feasible for data embedding in EL when an inter-layer reference picture is utilized in EL, because all MVs are set to *zero* to fulfil the requirement of bit stream conformance in EL(s) (Chen, Boyce, Ye, & Hannuksela, 2015). On the other hand, when the fast encoder mode setting is enabled, only blocks with size $2N \times 2N$ are coded, hence any method that is based on block partitioning (e.g., (Shanableh, 2018)) can only be applied to BL, unless the fast encoder mode is disabled. Likewise, although a coefficient-based method can be easily adopted in SHVC, propagation of error should be handled carefully to avoid poor perceptual quality for inter-layer predicted picture.

All in all, in comparison to AVC-based and HEVC-based methods, the achievable embedding capacity in scalable coded video by the proposed method is encouraging because the syntax elements in all coded layers can be exploited for data embedding purposes. Furthermore, some existing techniques can be combined with our work to achieve better trade-off among embedding capacity, quality, and bit rate. For example, the PB embedding technique can be applied to BL while the MVD, coefficient and merge mode can be applied to all layers to achieve a higher combined embedding capacity. This trade-off will be further investigated as part of our future work.

6.5 Summary

In this work, both intra prediction mode and MVP of SHVC video are jointly utilized to embed data. The data embedding opportunity is hence increased by jointly utilizing both intra prediction mode and MVP. The encoder calculates the prediction errors based on the original and the predicted values after the predictive syntax elements are modified to match the payload bits. Therefore, the reconstructed video is a close approximation of the original video and the perceptual quality of the video is maintained without error drift. Intra prediction mode in EL can embed up to 5 payload bits in a single prediction mode without drift-error by utilizing the correlation of inter layer prediction mode from the reference layer. With an adjustable threshold introduced to the rate-distortion guided block splitting, the embedding capacity is improved by making the PBs assume smaller sizes, which directly results in having more manipulable predictive syntax elements for data embedding purposes. Specifically, the strength of the proposed method is that the embedding capacity can be adjusted according to the varying payload requirements. When the required embedding capacity is low, the room given to code more smaller blocks can be reduced (i.e., reducing Υ) so that the bit rate overhead and distortion can be kept to the minimum. On the other hand, when there is a need to accommodate more data, the video encoding process is made to code more coding blocks by using the smallest block size (i.e., increasing Υ). Experimental results suggest that the achieved embedding capacity is encouraging while the video quality and bit rate are maintained. A trade-off between the increase in embedding capacity and bit rate overhead can be achieved by adjusting the threshold Υ depending on the application of interest. In the best case scenario, the sequence *PartyScene* can embed up to 516.9 kbps of payload with an average 7.0% bit rate overhead for the LDP configuration, while the same video can embed up to $1,578.6$ kbps with an average 2.9% bit rate overhead for the All Intra (AI) configuration. As future work, the plan is to suppress bit rate overhead due to data embedding, and investigate into different ways to combine use of the proposed and conventional data embedding methods.

CHAPTER 7: CUPSEED - A COMBINED USE OF PREDICTIVE SYNTAX ELEMENTS TO EMBED DATA IN SHVC VIDEO

7.1 Introduction

Data can be embedded into difference venues of a video sequence during video encoding. In general, data embedding in video takes place either in the spatial, temporal or transformed domain. Usually a single predictive syntax element is utilized for data embedding. Although a CU is manipulated to embed payload, it cannot guarantee whether a payload bit can be successfully embedded into the CU due to the competition among the prediction modes and different coding block size. The prediction modes and partitioning depth are assessed by RDO, and only the one with the best RDC is adopted for coding. In this chapter, a data embedding technique that utilizes multiple predictive syntax elements (PSEs) is proposed. Arguably, prediction is one of the most important ingredients in video coding, because this technique is able to effectively remove spatial and temporal redundancies. Specifically, predictive syntax elements, which include intra prediction mode (IPM), motion vector predictor (MVP), motion vector difference (MVD), merge mode (MRG) and block structure (BSZ), are manipulated in a combined manner to embed data.

The main contributions of this work are:

1. comprehensive analysis on the impact of using different SHVC prediction elements to embed data;
2. improve payload (i.e., number of bits that can be embedded) by utilizing multiple PSEs without compromising video quality while minimizing bit rate variation, and;
3. propose a data embedding technique - CUPSEED, which automatically selects different prediction elements for data embedding.

The experimental results demonstrate that, in comparison to the conventional single-venue data embedding techniques, combined use of predictive syntax elements can achieve higher payload while preserving the perceptual quality with minimize bit rate variation. In the best case scenario, a total of 556.1 kbps is embedded into the video sequence *PartyScene* with a drop of 0.15 dB in PSNR while experiencing bit rate overhead of 7.4% when all predictive syntax elements are utilized altogether. Subsequently, the impact of data embedding to the coding block structure as well as the prediction mode of SHVC video is analyzed.

Based on extensive literature review, there is no prior work that manages data embedding by using multiple venues in SHVC compressed video. Considering the advantages and potentials of embedding data into scalable coded video, recommendations for managing multiple predictive syntax elements in embedding data into SHVC are put forward.

7.2 Selection of Prediction Mode

In a video frame, a PU can be predicted by using an intra- or inter-picture prediction mode which are details in Chapter 2. For PU coded in inter-picture prediction mode, either the predicted MV or MV of neighboring block/frame (i.e., merge mode) is used for deriving the motion data. The selection of prediction mode is based on the best RDC. Let Ψ be the set of all possible intra and inter prediction modes, and let $\psi_i \in \Psi$ be the coding mode applied on block b_i. The coding mode ψ_i is selected according to

$$\psi_i = \underset{\psi \in \Psi}{\operatorname{argmin}} \mathcal{D}_i(\psi) + \lambda \mathcal{R}_i(\psi), \tag{7.1}$$

where the distortion $\mathcal{D}_i(\psi)$ represents the sum of squared differences between the original block b_i and the reconstructed block b_i' (i.e., the result of coding b_i by using mode ψ). The term $\mathcal{R}_i(\psi)$ represents the number of bits spent on coding the blocks b_i by using the coding mode ψ. It includes the number of bits required for signaling the coding mode and the associated side information (e.g., MVs, reference indices, IPM and coding modes for all partitioned blocks of b_i), as well as the number of bits spent on transmitting the transformed coefficient levels to store the residual signal (Ohm, Sullivan, Schwarz, Tan, & Wiegand, 2012).

7.3 Proposed Data Embedding Technique

The advantage of combined use of all prediction modes is that it allows some form of optimization, where the best mode can be selected for data embedding. In contrast, when restricting data embedding to a single prediction mode (venue), there is no guarantee that a payload bit can be embedded successfully into a particular block. For example, suppose only IPM is manipulated for data embedding. After embedding data into a particular block, a potential outcome is that IPM costs more than MV for coding the block in question. In such a situation, IPM will not be employed and the payload bit fails to be embedded into the block. From another perspective, the combined use of MVP, MVD, IPM and MRG also increases the payload. Specifically, each block holds one bit, unless it is identified to be a skipped block.

7.3.1 Combined Use of Predictive Syntax Elements for Data Embedding

In order to achieve high payload, the intra and inter prediction PSEs in all scalable coded layers are utilized in a combined manner for data embedding purposes. Due to the complexity of the SHVC codec, the quad-tree splitting and pruning process in CTUs has to be managed well to ensure that each data bit is embedded accordingly. The flow of processes is illustrated in Figure 7.1. Let $M = \{m_i\}_1^N$ be the payload data, where $m_i \in \{0, 1\}$ and N is the length of M. Then, data embedding takes place at five venues, namely, BSZ, IPM, MVP, MVD, and MRG.

First, the data bit m_i is mapped to the PB by selecting the best block structure which

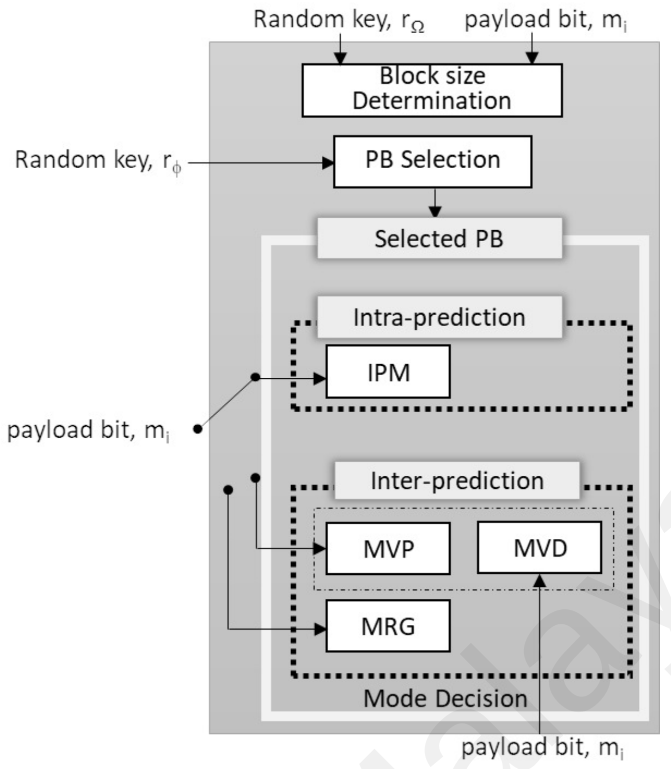


Figure 7.1: Flow of embedding process

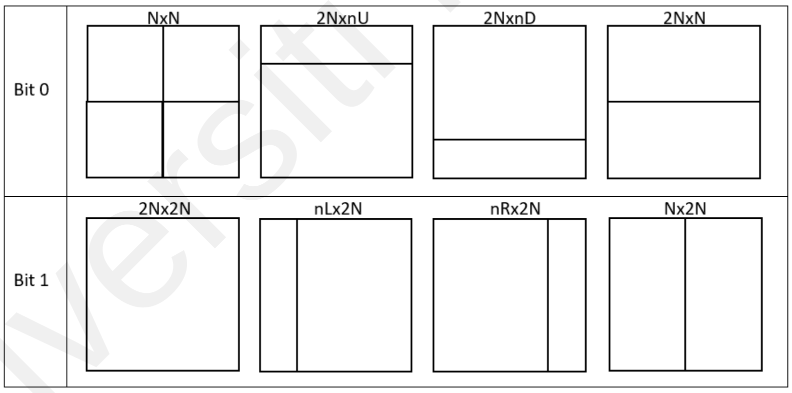


Figure 7.2: Coding block structure and payload bits representation.

represents m_i. Specifically, all coding block structures mapped to m_i are evaluated by using all applicable prediction modes, and the one achieving RDC_{best} is coded. One of the possible mapping rules is shown in Figure 7.2. Data embedding using BSZ is applicable to the I, P and B-slices. For the case of I-slice, all PBs are encoded by using $N \times N$ block for $m_i = 0$, or $2N \times 2N$ block for $m_i = 1$, respectively. On the other hand, for the case of P- and B-slices, there are more coding block structure for consideration. In both cases,

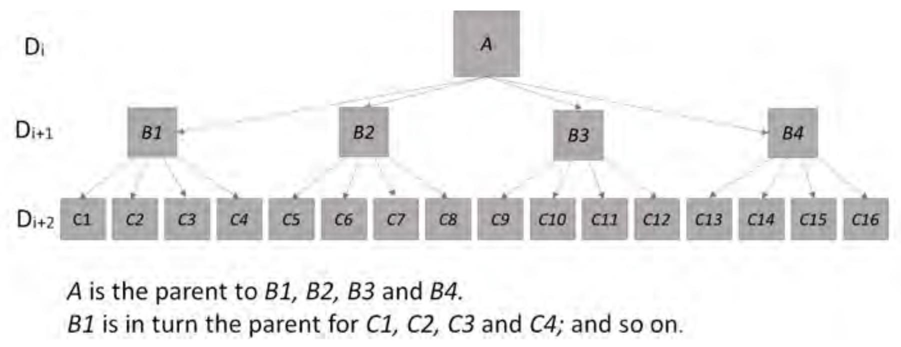
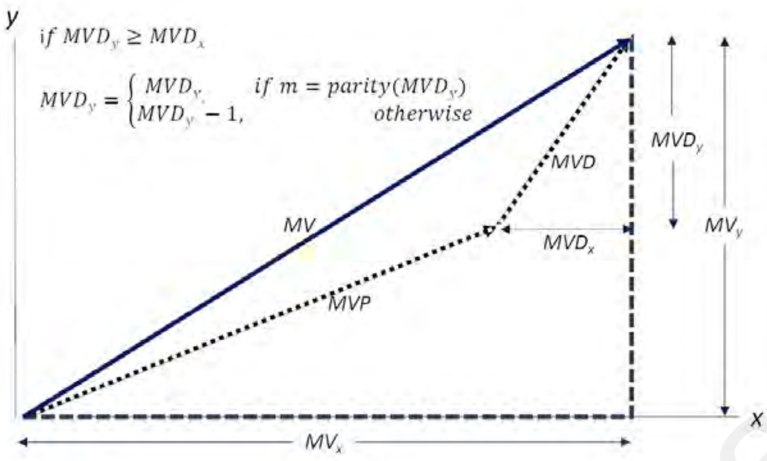

Di
A
Di+1
B1
B2
B3
B4
Di+2
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C12
C13
C14
C15
C16
A is the parent to B1, B2, B3 and B4.
B1 is in turn the parent for C1, C2, C3 and C4; and so on.

Figure 7.3: Illustration of parent block and its children block.

the coding block structure are divided into two groups, where one group represents '0' and the other represents '1'. In particular, the first group includes blocks of type $N \times N$, $2N \times nU$, $2N \times nD$ and $2N \times N$, which is utilized to represent payload bit '0'. On the other hand, the other group includes blocks of type $2N \times 2N$, $nL \times 2N$, $nR \times 2N$ and $N \times 2N$, which is utilized to represent payload bit '1'. The block associated with the data (i.e., m_i) having the best RDC will be selected for coding. Here, random keys κ_Ω can be utilized for selecting or skipping candidate blocks for embedding.

The prediction modes of IPM, MVP and MRG are manipulated to embed payload bit m_i during the encoding process. Here, another key, κ_ϕ can be utilized to select/skip blocks for data embedding. For each partitioning block, all prediction modes after associated with payload bit are evaluated. RDO will then determine and select the one with the optimal cost for coding. Basically, a video frame is partitioned into a number of CTUs (i.e., 64×64). For each CTU (i.e., at D_0), the coding block is further split into smaller blocks. To ease the presentation, let the term *parent block* refer to the block before splitting as illustrated in Figure 7.3. For each block, the encoder predicts the block by using all possible prediction modes, and for each prediction mode, all possible coding block structures (i.e., partitioning) are considered. Specifically, RDO determines the best coding structure for each prediction mode. Then, comparison of RDC for different prediction



y
if $MVD_y \geq MVD_x$
$MVD_y = \begin{cases} MVD_y & if\ m = parity(MVD_y) \\ MVD_y - 1, & otherwise \end{cases}$
MVD
MV
MVP
MVD_x
MVD_y
MV_y
MV_x
x

Figure 7.4: Example of payload bit representation using MVD

modes is performed, and the one achieving RDC_{best} is retained for subsequent comparison (i.e., line 3 to 6, and 8 to 11 in Algorithm 3). Next, each of these blocks (i.e., at D_1) is further split into smaller blocks (i.e., D_2). The same prediction and splitting processes (i.e., line 13 to 23 in Algorithm 3) are performed for each block, and the processes is repeated until the smallest block size is reached. When the best prediction mode in the smallest blocks (e.g., those labelled with C at the bottom of Figure 7.3) is decided, the RDC for the sibling split blocks are accumulated and compared with the RDC for the parent block. As an example, the sum of RDC of $C1$, $C2$, $C3$ and $C4$ is compared with RDC of $B1$. The prediction-splitting combination that yields RDC_{best} is selected to code the 64×64 block. The comparison process starts from the bottom (i.e., smallest block size), then moves up one level at a time, and eventually to the root until the entire block-partitioning process in the quad-tree is completed (i.e., at D_0). At each partitioning depth level d, the prediction modes compete among each other, and the one with RDC_{best} is identified. Hence, the combined used of prediction modes of IPM, MVP and MRG can increase the opportunities for data embedding.

During motion compensation prediction, both MVP and MVD are considered to increase

Algorithm 1: Embed data bit into MRG

```
input  : $RDC_{best}$, payload data $M$, $BestMode$
output : $RDC_{best}$, $BestMode$
get payload bit $m_i$ from $M$
Construct a list of merge candidate, $C$ based on payload bit $m_i$
$RDC_{MRG} \leftarrow MAX_{DOUBLE}$
for $c_i$ in C do
    Perform motion compensation and residual prediction
    Calculate $RDC_i$ for $c_i$
    if $RDC_i < RDC_{MRG}$ then
        $RDC_{MRG} \leftarrow RDC_i$
        $mergeCand \leftarrow c_i$
    end
end
if $RDC_{MRG} < RDC_{best}$ then
    $RDC_{best} \leftarrow RDC_{MRG}$
    $BestMode \leftarrow MergeMode$
end
```

Algorithm 2: Embed data bits into MVP and MVD

```
input  : $RDC_{best}$, payload data $M$, $BestMode$
output : $RDC_{best}$, $BestMode$
get payload bit $m_i$ from $M$
Derive MVP candidates from spatial and temporal candidate list
if $MVP\ exist$ then
    select MVP with MVP_index that matches $m_i$
    get payload bit $m_i$ from $M$
end
$MVD \leftarrow MV - MVP$
Check and get the longest magnitude of $MVD_i$, $i$ is horizontal or vertical
  component of MVD
if $m_i \neq parity(MVD_i)$ then
    if $MVD_i \geq 0$ then
        $MVD_i \leftarrow MVD_i - 1$
    else
        $MVD_i \leftarrow MVD_i + 1$
    end
end
Perform motion compensation and residual prediction
Check $RDC_{MV}$ for inter prediction mode
if $RDC_{MV} < RDC_{best}$ then
    $RDC_{best} \leftarrow RDC_{MV}$
    $BestMode \leftarrow InterMode$
end
Algorithm 1
```

Algorithm 3: Combined Used of Predictive Syntax Elements for data embedding

```
CUPSEED($CU, d_i, M$)
get $m_i$ from $M$
if $m_i = 1$ then
    for block structure $\in 2N \times 2N$ , $N \times 2N$, $nL \times 2N$, $nR \times 2N$ do
        Algorithm 2 /* Embed data bits into inter mode */
    end
    Embed data bit into IPM using $2N \times 2N$
else
    for block structure $\in N \times N$ , $2N \times N$, $2N \times nU$, $2N \times nD$ do
        Algorithm 2
    end
    Embed data bit into IPM using $N \times N$
end
if $d_i < d_{max}$ then
    split $CU$ into $cu_j$, $j \in 1..4$
    for $j \leftarrow 1$ to 4 do
        CUPSEED($cu_j, d_{i+1}, M$)
    end
    calculate accumulated $RDC_{d_{i+1}}$
    if $RDC_{d_{i+1}} < RDC_{best}$ then
        $RDC_{best} \leftarrow RDC_{d_{i+1}}$
        $CU \leftarrow cu_j, j \in 1..4$
    end
end
return;
```

payload. In particular, attempts are made to map m_i to indices of MVP for a particular PB. The MVP candidates are depicted in Figure 7.5. For each PB, m_i is mapped to either the best or first runner-up MVP index. Then, MVD is calculated, and the magnitude for the horizontal and vertical components are compared. Note that smaller MVD requires less bits for coding, and vice versa. Hence, if the parity bit of the MVD component with the larger magnitude differs from m_i, the magnitude of the MVD is reduced by one as depicted in Figure 7.4. The prediction information for the best PB (denoted by RDC_{MV}) is then selected for subsequent comparison.

The block merging candidate selection decision is manipulated to embed data based on some pre-defined mapping rules. Again, merge candidates are divided into two groups, where one is associated with bit ‘0’ and the other associated with bit ‘1’. An example of

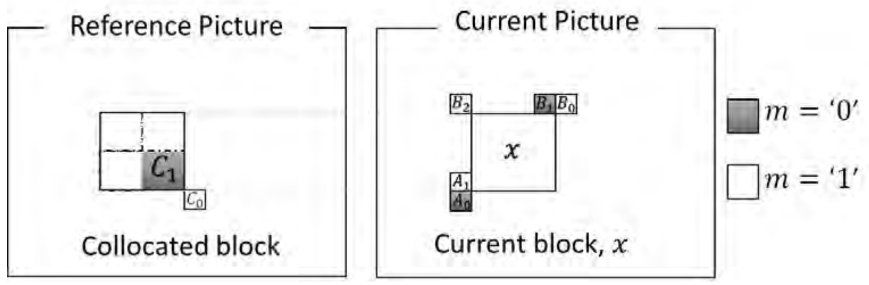


Figure 7.5: Merge candidates mapping.

mapping rules applied to merge candidates is shown in Figure 7.5. To determine the merge candidate for the current block (CurrBlk), m_i is mapped to the merge candidates associated to the payload bit as illustrated in Algorithm 1. The prediction information of merge candidate block with the best cost (denoted by RDC_{MRG}) is selected. The RDC_{MRG} is then compared with RDC_{best}. The one with the lower cost is selected for subsequent comparison.

For IPM, only the prediction directional modes are considered for data embedding. This is to ensure that the bit rate overhead is kept to the minimum. During intra picture prediction, the parity bit of the prediction mode is manipulated for data embedding. When the best predicted direction in BL has an index that differs from the data bit m_i, it is modified by increasing the prediction direction mode by unity. In contrast, when the parity bit differs from m_i in EL, the prediction direction mode is reduced by unity. For the boundary cases, the opposite prediction direction is applied. In such way, any changes or noise introduced to the prediction mode in BL can be compensated in EL when changes are made towards the opposite prediction direction. Specifically, the angular difference between the best and selected modes is small, which is $\sim 5.6^\circ$ (J. Wang et al., 2015). When a video is coded in more than two layers, the error introduced in the reference layer (i.e., BL) can be compensated in the EL to reduce the bit rate variation (Pang et al., 2017). The prediction mode that yields the best cost (denoted by RDC_{IPM}) is then compared with RDC_{best}. The one with the lower cost is selected for coding. Note that only the smallest coding blocks are utilized for data embedding in IPM, MVP and MVD in order to minimize bit rate variation and to preserve video quality. In addition, RDO plays an important role in determining the prediction mode for coding. For example, suppose the original best prediction mode for a PB is IPM. When embedding m_i into that PB, RDC for all prediction modes associated with m_i are evaluated. Suppose the merge mode has RDC_{best}. In such a situation, merge mode will be coded instead of IPM.

7.3.2 Data Extraction

During decoding, the embedded data can be extracted from each predictive syntax element in the PBs identified by using κ_{Ω} and κ_{ϕ} (i.e., to determine which one is skipped or selected), which are only known by the authorized parties. First, the embedded data is retrieved by checking the *parity* of the intra prediction directional mode in the identified embedding block. Recall that when the parity bit of the MVD component with the larger magnitude differs from the payload bit, the magnitude of the MVD is reduced by unity. Hence, for extracting the embedded data from MVP and MVD, the *parity* of the MVP (i.e., indices) and MVD components with the same or larger MV magnitude for the identified PBs are extracted. As an illustration, Figure 7.4 shows the MV with MVP and MVD components. The magnitude of MVD_y is greater than MVD_x, hence payload bit is extracted from the MVD_y (instead of MVD_x). Next, the payload bits are extracted based on the predefined mapping rule for the merge mode candidates (see Figure 7.5). Similarly, the embedded data can be extracted from the identified PBs by checking the block size based on the agreed-upon mapping rules, such as the one illustrated in Figure 7.2.

7.4 Experimental Results

The SHVC reference software SHM-12.0 (Joint Collaborative Team on Video Coding, 2017) is modified to implement the proposed data embedding technique. Experiments are conducted by using two-layer spatial scalability for LDP settings with group of pictures (GOP) structure of 4, i.e., IPPPIPPP$\cdots$. The scalable configuration with fixed QP is considered, where QP = 22 and 20 are set for the BL and EL, respectively. The remaining parameters are set to the SHM default configuration. A pseudo-random number generator is employed to generate a sequence R of 0's and 1's, which is then embedded by manipulating the predictive syntax elements determined by the proposed technique. Six standard video test sequences for SHVC from (Universität-Hannover, 2013; Xiph.org, 2013), namely *BasketballDrill, BlueSky, FourPeople, PartyScene, RaceHorses* and *RushHour* are considered to evaluate the performance of the proposed data embedding techniques. The video test sequences considered in the experiments are sufficiently diverse, i.e., having complex textures and smooth regions of varying sizes and motions at various speeds. R is embedded into the designated venues of the first 200 frames (which is the maximum length for one of the sequence - *BlueSky*, rounded to hundreds) in each video test sequence. It is verified that the processed video is still SHVC compliant, and the embedded data can be extracted correctly from the respective PSEs. By using SHVC compressed video as the baseline, the processed videos are evaluated in terms of variation in bit rate, video quality and payload. A total of 9 sets of experiment are conducted using different combination of PSEs for data embedding. These combinations include: IPM, MVP, MVD, MVP+MVD (referred to as MVA), Merge Mode (MRG), Block Structure (BSZ), MVP+MVD+MRG (referred to as MVG), IPM+MVP+MVD+MRG (referred to as IVG), and IPM+MVP+MVD+MRG+BSZ (referred to as ALL).

Table 7.1: Comparison of bit rate overhead (%) for different data embedding techniques.

Test Sequence	**IPM**	**MVP**	**MVD**	**MVA**	**MRG**	**MVG**	**IVG**	**BSZ**	**ALL**
RushHour @30 Hz	0.6	0.4	0.9	1.0	1.4	2.1	2.6	7.5	16.2
FourPeople @60 Hz	1.9	0.1	0.3	0.4	1.0	2.2	3.5	7.1	13.8
BlueSky @24 Hz	0.6	0.2	0.9	1.3	0.8	3.4	3.4	4.3	8.8
BasketballDrill @50 Hz	2.4	0.3	0.8	1.0	1.1	3.1	4.7	6.0	12.5
PartyScene @50 Hz	1.0	0.3	0.8	1.1	0.7	2.8	3.5	3.6	7.4
RaceHorses @30 Hz	0.5	0.6	0.9	1.0	0.9	2.4	2.6	3.4	6.5

7.5 Discussion and Analysis

To examine the experimental results, bit rate overhead, quality distortion and payload capacity are measured and compared as in the following subsections.

7.5.1 Bit Rate Variation

The comparison of bit rate overhead among different combinations of PSEs is summarized in Table 7.1. Results indicate that, when multiple PSEs (e.g., IVG) are jointly utilized for data embedding, the results yield +3.4% of bit rate variation, which is relatively small in comparison to the achieved payload. One of the reasons is that, among the competing prediction modes, RDO selects the one with the best cost for coding. In addition, the embedding only takes place at the block having the smallest size, i.e., 4×4 for intra-coded block, and 4×8 or 8×4 for inter-coded block. In comparison, Buhari et al.'s method (Buhari et al., 2016) achieves an average bit rate overhead of $\sim$ 2.2%, which is slightly lower that this work, but it has a lower payload although it is acknowledged that their method was evaluated for H.264/SVC video. In general, manipulating the PB of different size causes higher bit rate overhead, while the BSZ technique results in an average bit rate overhead of 5.3%. One of the reasons is that manipulating larger block requires more bits for coding the prediction residual. When BSZ is jointly utilized with IVG, the average bit rate overhead increases by 10.9%.

Table 7.2: Video quality degradation in term of PSNR (dB) for different data embedding techniques.

Test Sequence	**IPM**	**MVP**	**MVD**	**MVA**	**MRG**	**MVG**	**IVG**	**BSZ**	**ALL**
RushHour									
BL (960×540)	0.01	0.01	0.01	0.03	0.01	0.05	0.05	0.12	0.00
EL (1280×720)	0.00	0.01	0.00	0.02	0.01	0.03	0.04	0.72	0.10
FourPeople									
BL (640×360)	0.05	0.00	0.01	0.02	0.02	0.05	0.11	0.13	0.23
EL (1280×720)	0.01	0.00	0.01	0.02	0.02	0.05	0.06	0.07	0.13
BlueSky									
BL (640×360)	0.01	0.01	0.03	0.05	0.01	0.07	0.09	0.10	0.19
EL (1280×720)	0.02	0.01	0.02	0.04	0.01	0.06	0.07	0.07	0.14
BasketballDrill									
BL (416×240)	0.02	0.00	0.01	0.01	0.01	0.04	0.06	0.07	0.15
EL (832×480)	0.01	0.01	0.01	0.02	0.01	0.04	0.05	0.06	0.12
PartyScene									
BL (416×240)	0.03	0.01	0.02	0.03	0.00	0.05	0.09	0.07	0.15
EL (832×480)	0.03	0.00	0.01	0.02	0.01	0.03	0.07	0.05	0.11
RaceHorses									
BL (416×240)	0.02	0.07	0.08	0.09	0.06	0.10	0.07	0.14	0.15
EL (832×480)	0.01	0.04	0.04	0.04	0.04	0.05	0.02	0.08	0.06

Table 7.3: Video quality degradation in term of SSIM (10^{-4}) for different data embedding techniques.

Test Sequence	**IPM**	**MVP**	**MVD**	**MVA**	**MRG**	**MVG**	**IVG**	**BSZ**	**ALL**
RushHour									
BL	3.0	0.0	0.0	3.0	0.0	4.0	5.0	6.0	7.0
EL	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	2.0
FourPeople									
BL	2.0	0.0	1.0	0.0	1.0	1.0	3.0	5.0	11.0
EL	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	2.0
BlueSky									
BL	2.0	1.0	1.0	2.0	0.0	2.0	3.0	2.0	6.0
EL	1.0	0.0	0.0	0.0	0.0	1.0	2.0	2.0	5.0
BasketballDrill									
BL	1.0	0.0	0.0	0.0	1.0	3.0	3.0	5.0	13.0
EL	1.0	0.0	3.0	0.0	0.0	0.0	1.0	1.0	2.0
PartyScene									
BL	3.0	0.0	0.0	1.0	0.0	2.0	6.0	5.0	12.0
EL	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0
RaceHorses									
BL	1.0	3.0	3.0	2.0	3.0	2.0	6.0	5.0	12.0
EL	1.0	0.0	5.0	2.0	2.0	0.0	2.0	0.0	2.0

Table 7.4: Payload (kbps) for different data embedding techniques.

Sequence	**IPM**	**MVP**	**MVD**	**MVA**	**MRG**	**MVG**	**IVG**	**BSZ**	**ALL**
RushHour									
BL	7.8	8.7	8.3	13.0	18.1	38.0	45.7	58.4	126.7
EL	5.4	3.4	3.1	6.8	21.0	46.8	52.0	70.5	162.5
Total	13.1	12.1	11.4	19.8	39.1	84.8	97.7	128.9	289.2
FourPeople									
BL	39.2	4.4	4.0	7.0	19.9	35.5	75.1	32.5	173.5
EL	63.2	10.3	9.2	14.6	64.0	101.3	174.0	112.8	458.0
Total	102.5	14.7	13.1	21.6	83.8	136.7	249.2	145.4	631.5
BlueSky									
BL	9.4	7.8	6.6	10.6	11.5	34.9	43.1	34.5	76.6
EL	33.7	26.0	22.0	35.6	43.5	116.3	146.0	115.3	257.7
Total	43.1	33.9	28.6	46.2	54.9	151.3	189.1	149.8	334.3
BasketballDrill									
BL	18.9	6.3	5.6	10.5	11.8	29.3	48.4	44.1	96.1
EL	63.1	16.2	14.3	25.4	35.2	84.5	149.4	178.5	292.7
Total	82.0	22.5	19.9	36.0	47.0	113.8	197.8	222.6	388.8
PartyScene									
BL	30.8	11.0	9.5	16.9	17.1	48.6	78.8	47.0	114.5
EL	112.4	45.5	38.2	64.3	66.8	195.8	299.4	64.9	441.6
Total	143.2	56.5	47.8	81.2	83.9	244.4	378.2	112.0	556.1
RaceHorses									
BL	13.2	7.6	7.2	12.6	7.5	25.2	30.6	37.2	65.9
EL	30.8	17.9	17.2	28.4	28.1	66.1	77.1	103.2	163.5
Total	44.0	25.4	24.4	41.0	35.6	91.3	107.7	140.4	229.4

7.5.2 Video Quality

The comparison of quality degradation in term of PSNR (dB) between the original (compressed) and processed (compressed + payload) videos are recorded in Table 7.2. , while the results in terms of SSIM (Z. Wang et al., 2004) are recorded in Table 7.3. It is observed that, in most cases, the manipulation of PSEs leads to insignificant quality degradation. Specifically, the average degradation in PSNR ranges from 0.02 dB to 0.11 dB for IVG. In the worst case scenario, PSNR drops by 0.72 dB for the sequence *RushHour* for BSZ. On the other hand, in the best case scenario, when using MVP, the drop in PSNR is < 0.07 dB, where the average drop in PSNR is ~ 0.01 dB. Overall result suggests that the manipulation of PSEs leads to degradation in video quality. In comparison to IVG, the average PSNR degradation observed in Buhari et al.'s method (Buhari et al., 2016) falls in the ranges of $[0.04, 0.36]$ dB.

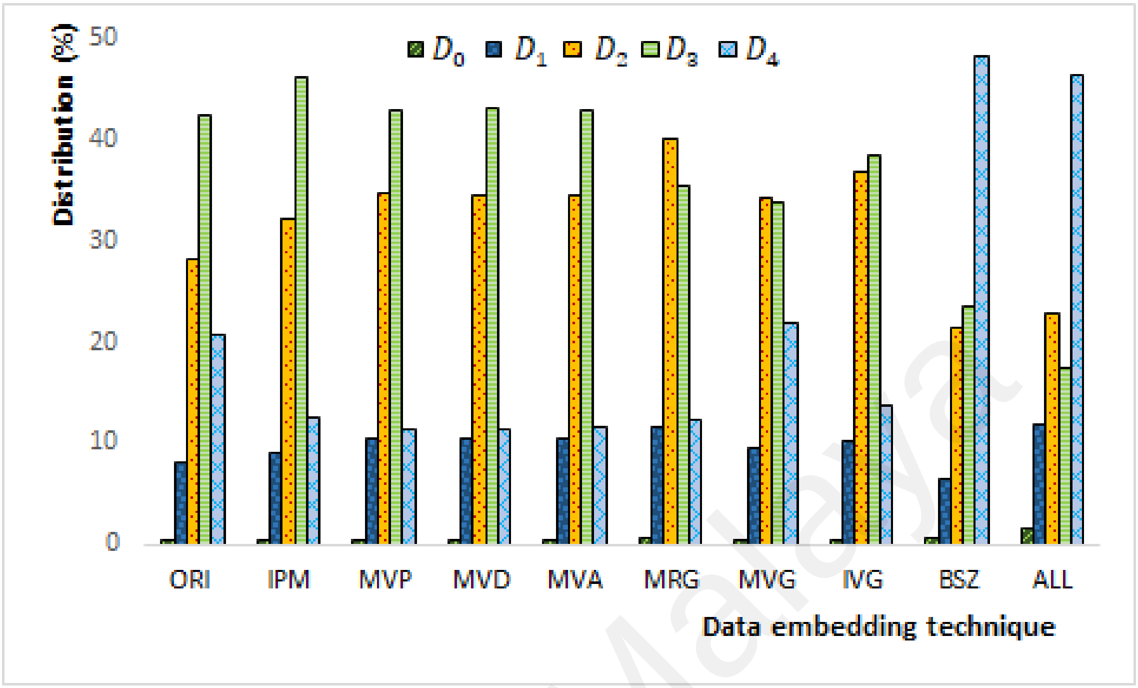
7.5.3 Payload

Table 7.4 records the payload for different combinations of PSEs. As expected, the payload consistently increases when more PSEs are utilized for data embedding purposes. Encouraging results are achieved by ALL, which is evidenced in Table 7.4. The combined use of PSEs has the advantage of being able to choose the PSE having RDC_{best} to represent the payload bits. However, to achieve optimal PSE, the RDO needs to check the RDC for every possible prediction mode, which increases the complexity of the encoder.

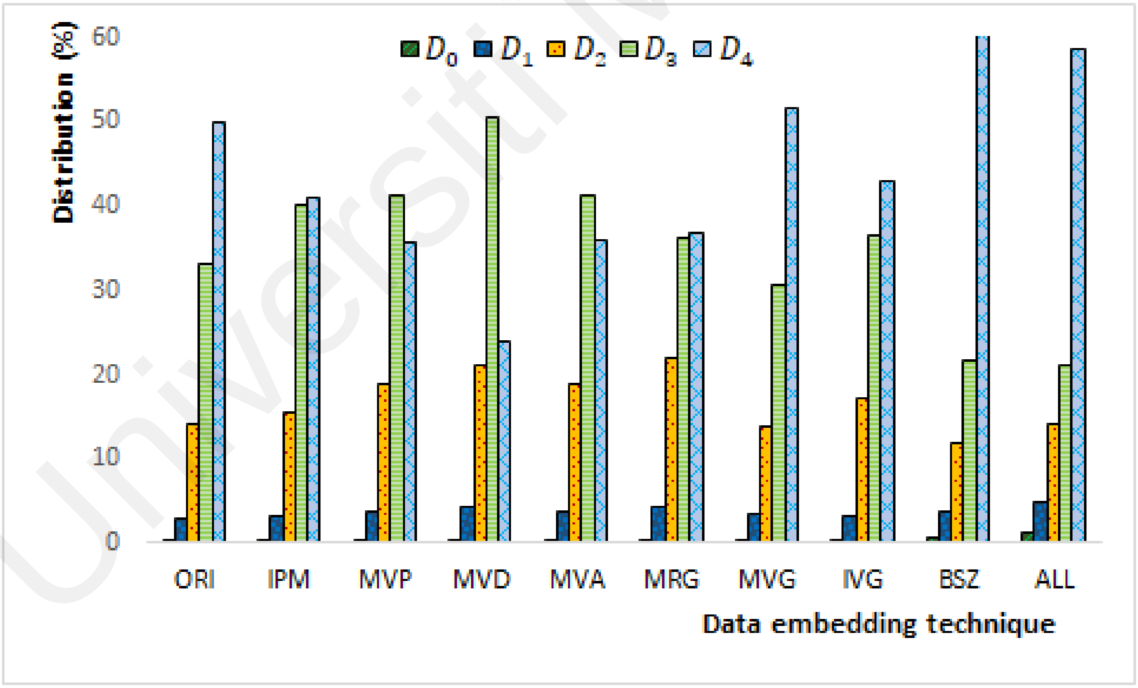
While suppressing bit rate overhead and quality degradation, the attained payload in ALL is significantly more ($\sim 3\times$) in comparison to existing solutions which only utilize one single PSE. For example, in the sequence *PartyScene*, the achieved payload when using ALL is 3.9× more than that of IPM.

7.5.4 Impact to Coding Structure and Prediction Mode

The impact of CUPSEED for data embedding on the coding block structure and prediction mode are analyzed. To facilitate the discussion, a short description of each video test sequence is provided in Subsection 3.4 to highlight its characteristics.

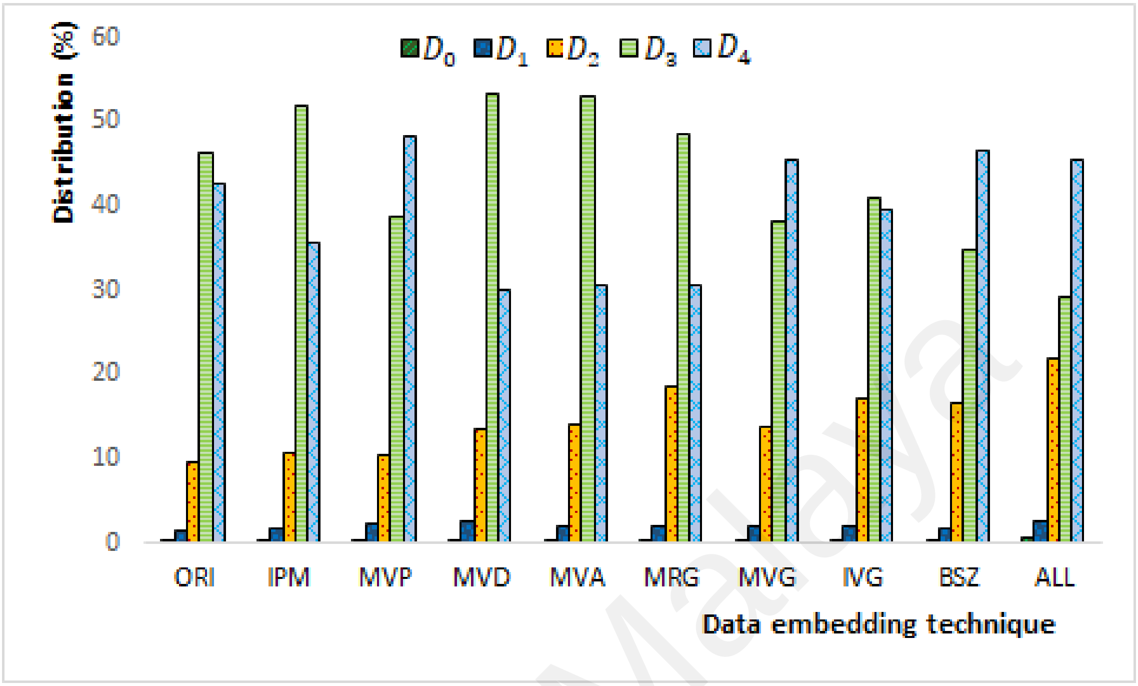


(a) RushHour

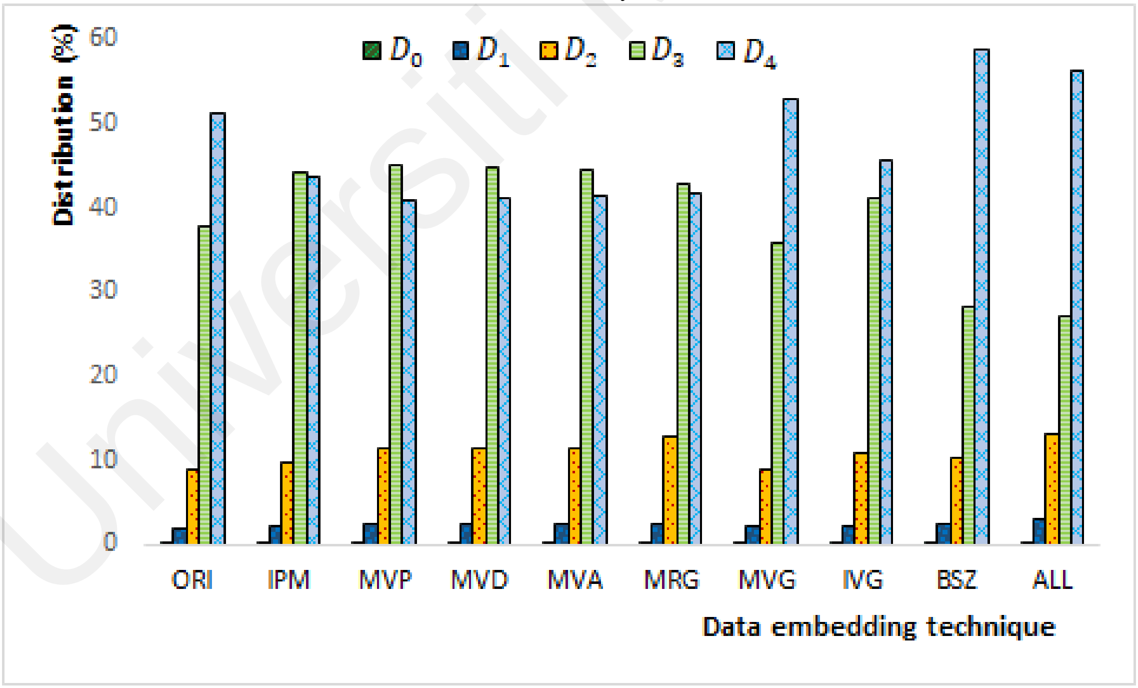


(b) FourPeople

Figure 7.6: Distribution of coding blocks at different partitioning depth level for different data embedding techniques.

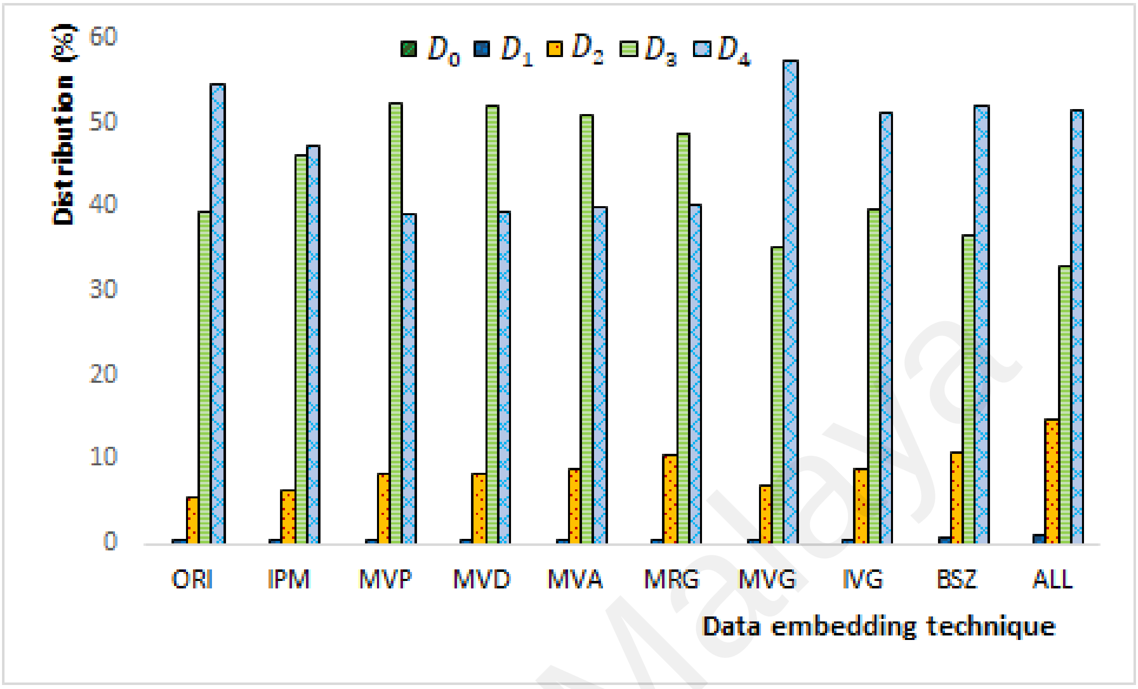


(c) BlueSky

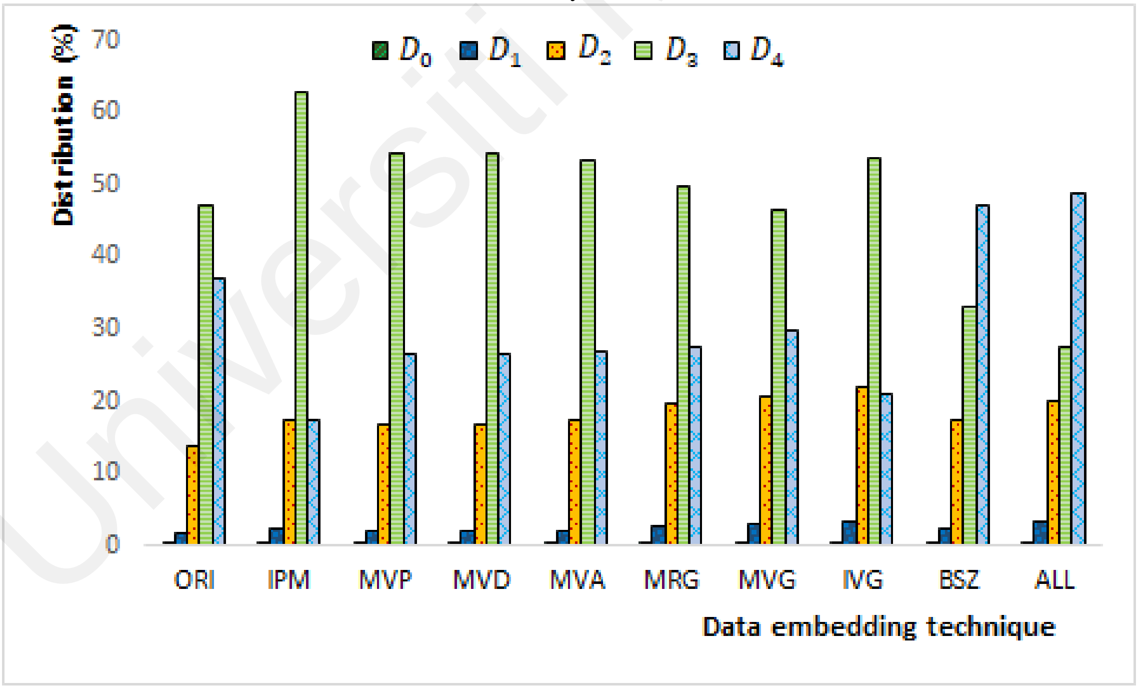


(d) BasketballDrill

Figure 7.6: (Continued).

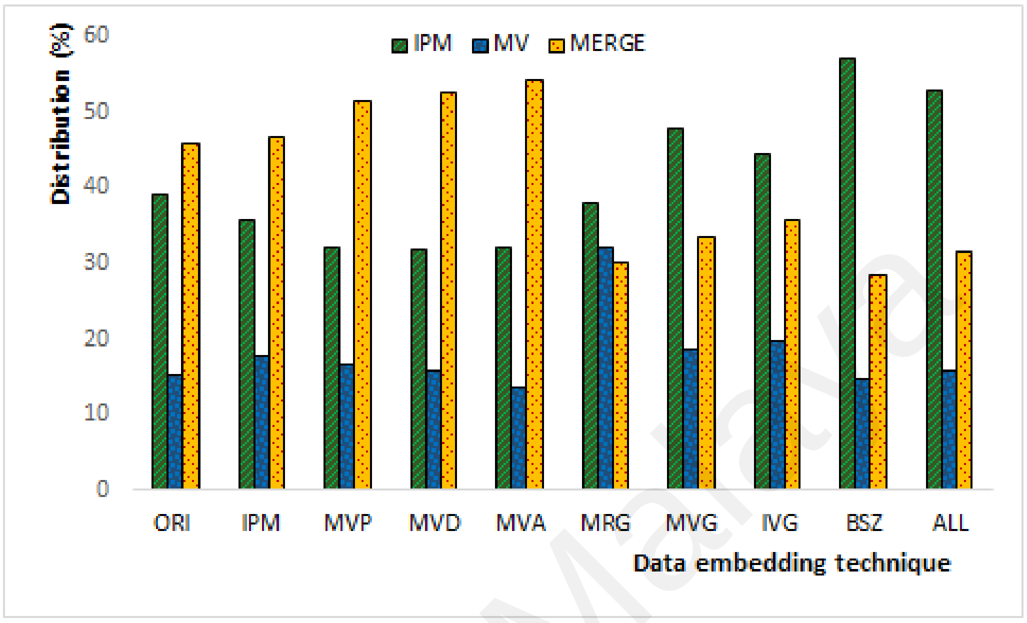


(e) PartyScene

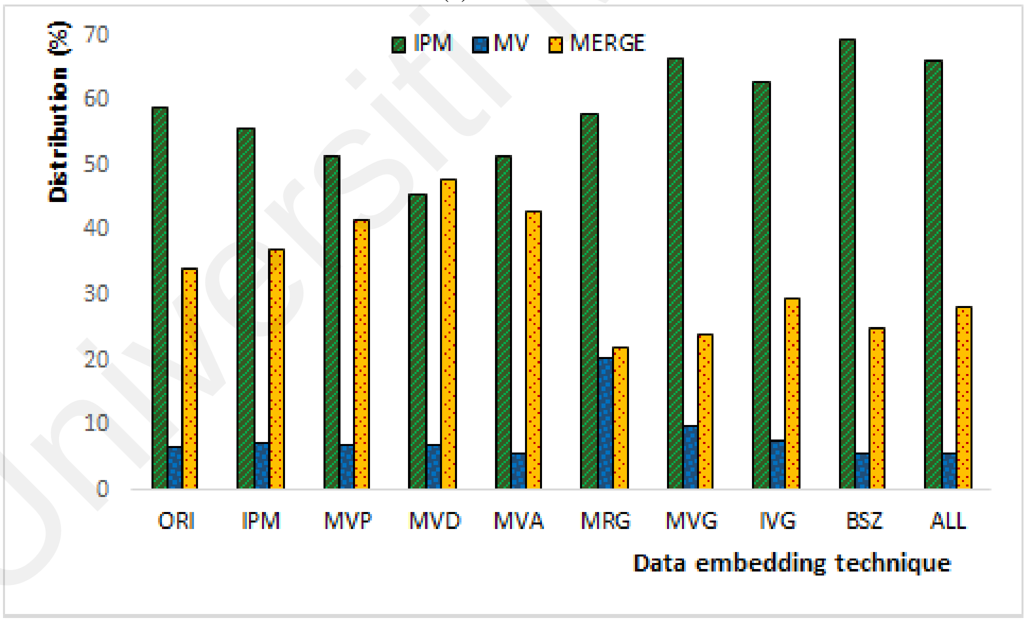


(f) RaceHorses

Figure 7.6: (Continued).

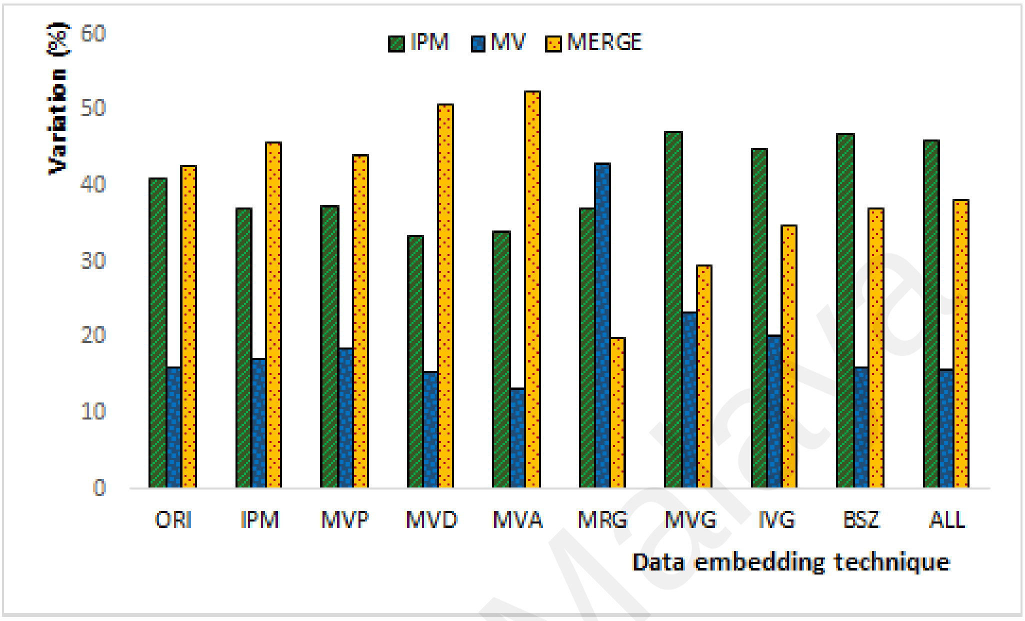


(a) RushHour

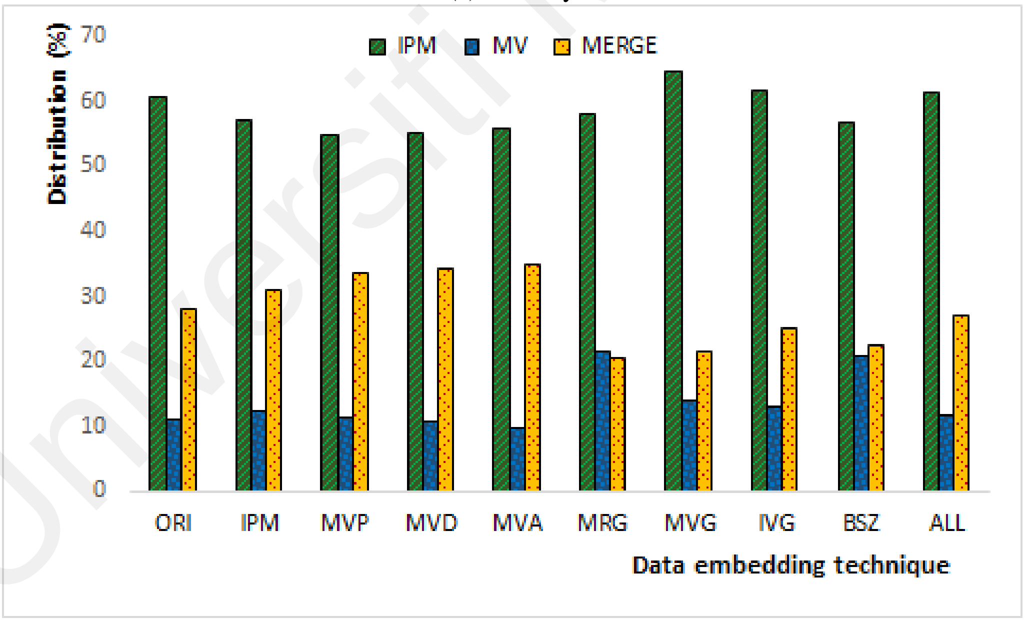


(b) FourPeople

Figure 7.7: Distribution of prediction modes for different data embedding techniques.

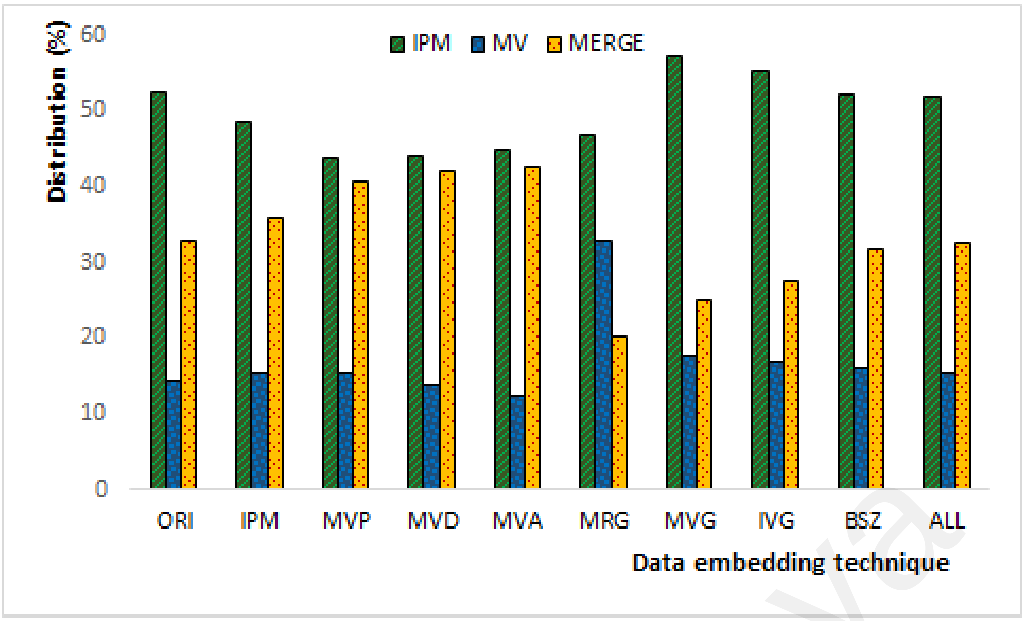


(c) BlueSky

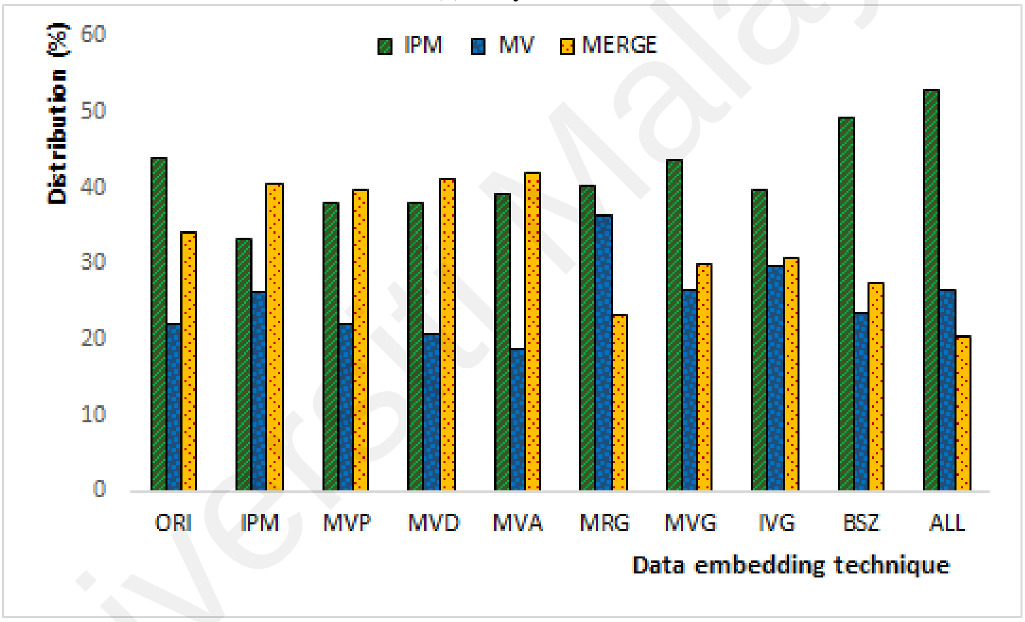


(d) BasketballDrill

Figure 7.7: (Continued).



(e) PartyScene



(f) RaceHorses

Figure 7.7: (Continued).

Figure 7.6 shows the number of blocks coded at different partitioning depth level when different PSEs are utilized for data embedding. Here, ORI refers to the statistics of the original video, i.e., compressed video without embedded data. Let D_4 represent the partitioning depth for 4×4 block. It is observed that the sequence *RushHour, BlueSky* and *RaceHorses* are coded with more larger blocks, while the other sequences have more

4×4 blocks coded. Usually, after attempts are made to embed payload bit into a particular block at partitioning depth D_{i+1}, the total $RDC_{D_{i+1}}$ for the block is increased. This leads to the situation where the total $RDC_{D_{i+1}}$ being larger than that of RDC_{D_i} (i.e., parent block). For example, a $2N \times 2N$ PU (parent) at partitioning depth level D_i is split into four $N \times N$ PUs (children) (i.e., at partitioning depth D_{i+1}) for intra and/or inter picture prediction. After the $N \times N$ PUs are manipulated to embed data, the total RDC for these split blocks are increased. If the total $RDC_{D_{i+1}}$ for the split blocks are greater than the parent block, then the $2N \times 2N$ PU is selected for coding instead of the four-split PUs. In most data embedding techniques (except BSZ and its combination), the distribution of blocks is shifted to the larger blocks (or smaller blocks if available) after data embedding. In general, it is observed that the number of 4×4 blocks for IPM increase when BSZ is used for the sequences which originally contain more smooth regions such as the sequences *RushHour*, *FourPeople* and *RaceHorses*. One of the reasons is that $P(m_i = 0) \sim 0.5$, which causes the mode-split distribution to change after embedding data by using the mapping rules between m_i and PSE.

Figure 7.7 shows the distribution of prediction modes for the original and processed videos. It is observed that there are more merge modes being coded for the sequences *RushHour* and *BlueSky* in the original video. This is because the sequence *RushHour* contains non-static background and the sequence *BlueSky* contains moving background. The sequences *FourPeople*, *BasketballDrill* and *PartyScene* contain either more static objects/background or fine textures while the sequence *RaceHorses* contains relatively more larger smooth regions. Hence, these videos are originally coded with more IPM. When embedding data by using IPM, the distribution of prediction modes shifted to more MV and merge modes, and in some cases blocks with larger size. When MVs are utilized for data embedding (i.e., MVP and MVD), the resulting video tends to have more merge modes. There are more MVs coded when MRG is utilized for data embedding. The trend is slightly different for BSZ, where more 4×4 blocks are coded. It can be concluded that when a particular prediction mode is utilized for data embedding, it results in higher RDC, and hence either other prediction mode or other block size with better RDC is selected for coding.

Recalled that two levels of decision are made by RDO: block size and prediction mode. These results demonstrate that manipulating PSEs affects the distribution of coded block size and prediction mode. After data embedding, video sequence with more static background tends to be coded by using IPM with different block size or merge mode. This trend can be observed in the sequence *FourPeople*. When MV is manipulated, more blocks are coded in merge mode, which can be observed in the sequence *RushHour* and *BlueSky*. However, when merge mode is manipulated, more IPMs is coded, which can be observed in the sequence *BasketballDrill* and *PartyScene*.

7.5.5 Discussion and Analysis

IPM, MVP and MVD are exploited to embed data by manipulating the parity bit of the PSEs, while MRG and BSZ associate the respective PSEs with payload bit. Among these five venues, BSZ achieves the highest payload, followed by IPM and MRG, depending on the video content. Specifically, video with more motion (e.g., *RushHour*) favors MRG. On the other hand, video sequence with more static objects or background (e.g., *FourPeople*) favors IPM. Subsequently, MVP and MVD also offer some venues to embed data.

In general, the availability of prediction mode determines the size of the payload. Specifically, BSZ can be applied to the I, P and B-slices, hence more blocks are available for manipulation. On the other hand, IPM is only applicable to I-slices, which restricts the payload. While MVP, MVD and MRG are applicable to the P and B-slices, PSEs involves MV, which is relatively low in existence.

It is noteworthy that embedding data into all venues (PSEs) causes minimal impact on video quality. This is because after data embedding attempts are made to IPM, MV (i.e., MVP and MVD) and MRG, only the one with RDC_{best} is coded as illustrated in Algorithm 3. The result is more encouraging as compared to using single venue embedding because the selection of PSEs to associate with the data bit is the best among all PSEs. Moreover, after each venue is utilized for data embedding, the prediction error/residual is recomputed for reconstruction purposes. The advantage of utilizing these PSEs is that propagation of error can be kept to the minimum in comparison to the manipulation of the transformed-quantized coefficients. While transformed-quantized coefficients can provide significantly higher payload, taking such approach may lead to noticeable quality degradation and error propagation when more coefficients are exploited for embedding.

The bit rate overhead for using MVP, MRG and MVD are relatively low in comparison to the payload. However, the bit rate overhead for using BSZ is higher due to PBs of varying block sizes being manipulated. In general, altering bigger block introduces higher prediction error and more bits are required to code the residual. These techniques maintain coding efficiency with some increase in bit stream size. Hence, a straight forward approach for improving bit rate is to restrict data embedding to smaller blocks.

CUPSEED increases the complexity of the video encoding process, but it is an encouraging solution for achieving high payload and high video quality while suppressing bit rate overhead simultaneously. Results also suggest that when all PSEs are utilized, they can collectively host $\sim 3\times$ more payload in comparison to the conventional single-venue data embedding technique.

7.5.6 Recommendations

With the aforementioned results, the following recommendations are put forward when utilizing PSEs for data embedding:

(a) In general, manipulating blocks of smaller block sizes has less impact to the bit rate variation and video quality. Hence, it is suggested that data embedding in IPM is feasible for most cases, especially for smaller GOP sizes where there are more I-slices. In addition, IPM can be applied to all coded layers since embedding in the BL and EL using opposite directions can suppress the bit rate variation and preserve the video quality in ELs. The achievable payload for video sequence with more static objects or background (e.g., *FourPeople*) is more encouraging. Besides, the video with more fine regions or small objects (e.g., *PartyScene*) can achieve higher payload. All in all, IPM is found to be the most promising individual PSE for data embedding, where an average of 71.3 kbps can be embedded into the 4×4 blocks with bit rate increment of 1.2% and quality degradation of 0.02 dB.

(b) The payload for video sequences with more motion (e.g., *RushHour* and *BlueSky*) is more encouraging when using MVP and MVD. A combined used of MVP and MVD increases the payload, while $< 2\%$ bit rate overhead and slight quality degradation are observed. Therefore, MVP and MVD are suggested for the sequence with more motion. When ILR picture is used as a reference picture for predicting EL (i.e., MV in EL set to zero), these predictive syntax elements can only be applied to BL. Here, IPM, MVD and MVP only utilize the smallest blocks for data embedding. However, IPM, MVD and MVP can be extended to large block size to cater for more payload. In contrast, for applications that require less payload, these PSEs provide the flexibility of selecting blocks at different partitioning depth level and prediction mode to embed the payload.

(c) Merge mode was introduced in HEVC standard to improve the compression efficiency irregardless of the motion speed. The experimental results show that there are more merge modes coded for the sequences with more motion, which implies that higher capacity can be achieved by using the MRG technique. Therefore, MRG is also suggested for the sequence with more motion. Here, MRG is applied to all coded layers and the experimental results demonstrate that it can achieve encouraging payload. For instance, the sequence *BlueSky* and *RushHour* offer higher payload in comparison to IPM, with bit rate variation of $< 3\%$ and video degradation ≤ 0.06 dB. Regardless of the video content, MRG outperforms MV-based techniques as suggested by the results recorded in Table 7.4. In addition, the MVG and IVG techniques offers higher payload as observed in Table 7.4. With IVG, the payload is $2\times$ more of that of single PSE, while the bit rate overhead is $< 5\%$. Hence, it is recommended to use MVG or IVG for applications requiring higher payload. In contrast, when the demand of payload is not high, this combination provides the flexibility to select blocks at different partitioning depth level for data embedding.

(d) The quad-tree block partitioning structure provides greater flexibility in manipulating the block size in a video. BSZ can be applied to smaller blocks only if bit rate overhead is a concern. To further suppress bit rate variation, BSZ can be applied to a single layer. Here, blocks for all partitioning depth level are manipulated for data embedding purpose. Figure 7.6 and 7.7 suggest that more IPMs are coded when BSZ is applied. Therefore a combination of BSZ and IPM can also be applied to increase the payload.

7.6 Summary

A combined use of PSEs, including IPM, MVP, MVD, MRG and BSZ, is put forward to embed data in SHVC coded video. Experiments results suggest that the proposed data embedding technique, called CUPSEED, outperforms single-venue PSE in terms of payload. Results suggest that when all PSEs are utilized, they can collectively host

3× more payload in comparison to the conventional data embedding technique. In the best case scenario, 556.1 kbps payload is embedded into the sequence *PartyScene* with a drop of 0.15 dB in PSNR and an overhead of 7.4% in bit rate. The analysis on the data embedding impact to the prediction modes as well as coding structure of SHVC video are conducted. It is observed that when embedding data into the smallest block, either more larger blocks or other prediction mode, whichever has the better rate-distortion-cost, is coded. Specifically, the texture and motion data decides the proportion of prediction modes and block size. Besides that, the impact analysis of data embedding to the coding mode and coding architecture is performed to serve as a guideline for future data embedding references.

For future work, the proposed technique will be extended for actual application in video management, annotation, hyper-linking and etc. The combined use of predictive syntax elements will also be extended together with other syntax elements such as transformed coefficient and quantization parameter for data embedding purposes.

CHAPTER 8: CONCLUSIONS

8.1 Chapter Overview

This chapter presents a summary of this thesis by recapturing the purpose of this research in Section 8.2, followed by the research outcomes together with the achievements and contributions in Section 8.3. Next, the pros and cons of the proposed methods are discussed in Section 8.4. Lastly, suggestions for future work is put forward in Section 8.5.

8.2 Summary

The purpose of this research is to identify and evaluate the venues in SHVC that can be utilized to embed data, and subsequently to increase payload while preserving the video quality with minimum bit rate overhead. Here, the research questions are revisited:

RQ1. What are the potential venues that can be utilized to embed data in SHVC coded video?

RQ2. How to increase embedding capacity according to the payload requirements of the applications?

RQ3. How to maintain minimum distortion and bit rate overhead with a better rate-distortion performance for data embedding purpose?

First, the syntax elements introduced in HEVC and its extension are identified and utilized to realise data embedding. The inter-picture prediction syntax elements, namely, MVP indices and selected merge mode candidate block are associated with payload bits in all scalable coded layers as detailed in Chapter 4 and Chapter 5. The proposed techniques outperform existing MV-based data embedding techniques. Next, a threshold is introduced to guide the RDO and the data embedding processes to improve the embedding capacity by allowing the PBs to be split into smaller block sizes, which directly results in having more manipulable syntax elements to embed data as suggested in Chapter 4. Payload can be increased by adjusting the threshold at slight increase of bit rate overhead which is close to the threshold value. Subsequently, the work is extended to intra-picture prediction syntax elements of all coded layers to increase the data embedding opportunities without introducing drift-error as detailed in Chapter 6. Then, a data embedding framework based on the combined use of prediction modes and block selection is designed and developed. The proposed framework manages to achieve encouraging payload with high output video quality and reasonable bit rate overhead. As part of the process, an impact analysis of data embedding for difference venues is conducted to get better understanding of the behaviour of the encoder after data embedding as well as changes made to the coding structure and the prediction modes due to the embedded payload as discussed in Chapter 7.

8.3 Achievement and Contribution

This study has achieved its objectives:

1. The *MVP-based* data embedding technique and *merge mode-based* data embedding technique during inter-picture prediction in *multi-coded layers* of SHVC coded video are proposed in Chapters 4 and 5. The proposed methods achieve higher payload as compared to the state-of-the-art data embedding methods using inter-picture prediction syntax elements.
2. The *threshold-guided* block partitioning method increases the number of the smallest PB that can be utilized to embed data, which in turns *increase payload capacity*. More payload can be achieved by using both the intra prediction mode and MVP without introducing drift-error. As reported in Chapter 6, encouraging payload is achieved with slight increase in bit rate overhead when the threshold is adjusted to a higher value.
3. The *CUPSEED* framework is proposed as *a guide in selecting and managing*

different number of different PSEs for data embedding. The proposed method has improved the payload without compromising video quality while minimizing bit rate variation as demonstrated in Chapter 7.

8.4 Pros and Cons

This research has the following highlights as its advantages:

1. The threshold controlled block partitioning technique provides rooms for improving the capacity within the prescribed bit rate overhead while CUPSEED provides the flexibility in using different combinations of venue to increase embedding payload capacity in all scalable coded layers while preserving the output video quality, and;
2. The proposed methods can be applied to other HEVC extension since the core architecture is based on HEVC.

However, there are some shortcomings as mentioned below:

1. SHVC encoding process has higher complexity as compared to HEVC and the previous video coding standards. Hence, prior to data embedding, one has to gain a complete/thorough understanding of the SHVC architecture in order to implement the proposed data embedding method in video encoder/decoder, and;
2. A small number of video sequences are considered. More video test sequences should be considered to better reflect the performance of the proposed methods and to make the arguments more convincing. However, it took a long period of time to encode these HD video sequences just for one particular set of parameter setting. To compensate for the shortcomings, the video test sequences considered in the experiments are sufficiently diverse, i.e., having complex scenes (i.e., spatial activity) and motion of various speed.

8.5 Future Work

In future, there are opportunities to embed data thanks to the ever-increasing number of videos as well as the introduction of new standards. Among the applications mentioned in Section 1.1, applications such as tagging of video content with robust metadata could become an important application area for video management. Besides that, Internet of things (IoT) based technologies bring a completely new perspective. For example, IP cameras are interlinked to allow video stream to be exchanged across platforms for multiple reasons. Since this setup generates an immense amount of video from connected devices, it is essential to ensure that the video is completely secured and no one can manipulate it at any access point. The proposed data embedding framework in this research can be extended to ensure the integrity of the video and its application in IoT. In addition, to further enhance this research, the proposed CUPSEED could be extended for joint utilization with techniques proposed for transformation, quantization and entropy coding. Besides that, this research could also be extended for joint utilization of SHVC video and other digital media such as audio and subtitle (Wong et al., 2020).

REFERENCES

Abdulla, A. A. (2015). *Exploiting similarities between secret and cover images for improved embedding efficiency and security in digital steganography* (Doctoral dissertation, University of Buckingham, Buckingham, United Kingdom). Retrieved from `http://bear.buckingham.ac.uk/149/`

Abdulla, A. A., Sellahewa, H., Jassim, S. A. (2014, May). Steganography based on pixel intensity value decomposition. In *Mobile Multimedia/Image Processing, Security, and Applications 2014* (pp. 19–27). Baltimore, Maryland, United States. doi: 10.1117/12.2050518

Abdulla, A. A., Sellahewa, H., Jassim, S. A. (2019). Improving Embedding Efficiency for Digital Steganography by Exploiting Similarities between Secret and Cover Images. *Multimedia Tools Appl.*, *78*(13), 17799–17823. doi: 10.1007/s11042-019-7166-7

Aly, H. A. (2011). Data hiding in motion vectors of compressed video based on their associated prediction error. *IEEE Transactions on Information Forensics and Security*, *6*(1), 14-18. doi: 10.1109/TIFS.2010.2090520

Amiri, M., Amiri, A., Meghdadi, M. (2019). HVS-based scalable video watermarking. *Multimedia Systems*, *25*, 1-19. doi: 10.1007/s00530-019-00604-0

Bjøntegaard, G. (2001, Apr). *VCEG-M33: Calculation of average PSNR differences between RD curves.* ITU-T SG16 Q.6 Document: VCEG-M33. Austin, Texas, USA.

Boyce, J. M., Ye, Y., Chen, J., Ramasubramonian, A. K. (2016). Overview of SHVC: Scalable extensions of the High Efficiency Video Coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *26*(1), 20-34. doi: 10.1109/TCSVT.2015.2461951

Buhari, A. M., Ling, H.-C., Baskaran, V. M., Wong, K. (2016). Fast watermarking scheme for real-time spatial scalable video coding. *Image Communication*, *47*(C), 86-95. doi: 10.1016/j.image.2016.06.003

Chang, P.-C., Chung, K.-L., Chen, J.-J., Lin, C.-H., Lin, T.-J. (2014). A DCT/DST-based error propagation-free data hiding algorithm for HEVC intra-coded frames. *Journal of Visual Communication and Image Representation*, *25*(2), 239-253. doi: 10.1016/j.jvcir.2013.10.007

Chen, J., Boyce, J., Ye, Y., Hannuksela, M. (2015, Oct). *JCTVC-U1007: SHVC Test Model 10 (SHM 10) Introduction and Encoder Description.* (accessed Feb 12, 2021)

Chernyak, R. I. (2014). Analysis of the intra predictions in H.265/HEVC. *Applied Mathematical Sciences*, *8*(148), 7389-7408. doi: 10.12988/ams.2014.49750

Cullen, C. (2019, Sept 10). *Sandvine releases 2019 Global Internet Phenomena Report.* [Press Release]. Retrieved from `https://www.sandvine.com/press-releases/sandvine-releases-2019-global-internet-phenomena-report`

Dutta, T., Gupta, H. P. (2016). A robust watermarking framework for High Efficiency Video Coding (HEVC) – Encoded video with blind extraction process. *Journal of Visual Communication and Image Representation*, *38*(C), 29-44. doi: 10.1016/j.jvcir.2015.12.007

Fang, D., Chang, L. (2006, May). Data hiding for digital video with phase of motion vector. In *2006 IEEE International Symposium on Circuits and Systems* (pp. –). Kos, Greece. doi: 10.1109/ISCAS.2006.1692862

Fridrich, J. (2009). *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge, USA: Cambridge University Press. doi: 10.1017/CBO9781139192903

Guan, Z., Wu, H. (2020, July). A Reversible Contrast Enhancement Scheme For Color Images. In *2020 IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1–6). London, UK. doi: 10.1109/ICME46284.2020.9102950

Gui, F., Xue, H. (2017, Dec). A Reversible Data Hiding Scheme for HEVC. In *2017 10th International Symposium on Computational Intelligence and Design (ISCID)* (pp. 34–37). Hangzhou, China. doi: 10.1109/ISCID.2017.151

Hao, B., Zhao, L., Zhong, W. (2011, May). A novel steganography algorithm based on motion vector and matrix encoding. In *2011 IEEE 3rd International Conference on Communication Software and Networks* (pp. 406–409). Xi'an, China. doi: 10.1109/ICCSN.2011.6013622

Hao, Y., Ngo, C., Huet, B. (2020). Neighbourhood Structure Preserving Cross-Modal Embedding for Video Hyperlinking. *IEEE Transactions on Multimedia*, *22*(1), 188-200. doi: 10.1109/TMM.2019.2923121

Helle, P., Oudin, S., Bross, B., Marpe, D., Bici, M. O., Ugur, K., . . . Wiegand, T. (2012). Block Merging for Quadtree-Based Partitioning in HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, *22*(12), 1720-1731. doi: 10.1109/TCSVT.2012.2223051

Hu, Y., Zhang, C., Su, Y. (2007, July). Information Hiding Based on Intra Prediction Modes for H.264/AVC. In *2007 IEEE International Conference on Multimedia and Expo* (pp. 1231–1234). Beijing, China. doi: 10.1109/ICME.2007.4284879

ITU-T. (2008, Apr). *Recommendation ITU-T P.910, Subjective video quality assessment methods for multimedia applications.* (accessed Feb 12, 2021)

Jiang, L., Ma, X., Chen, S., Bailey, J., Jiang, Y.-G. (2019, Oct). Black-Box Adversarial Attacks on Video Recognition Models. In *Proceedings of the 27th acm international conference on multimedia* (pp. 864–872). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3343031.3351088

Joint Collaborative Team on Video Coding. (2017). *SHVC reference software, SHM 12.* Retrieved from `https://hevc.hhi.fraunhofer.de/svn/svn_SHVCSoftware/` (accessed 1-Aug-2017)

Konyar, M. Z., Akbulut, O., Öztürk, S. (2020). Matrix encoding-based high-capacity and high-fidelity reversible data hiding in HEVC. *Signal Image and Video Processing*, *14*, 897-905.

Liu, Y., Zhao, H., Liu, S., Feng, C., Liu, S. (2018). A Robust and Improved Visual Quality Data Hiding Method for HEVC. *IEEE Access*, *6*, 53984-53997. doi: 10.1109/ACCESS.2018.2869148

Long, M., Peng, F., Li, H. (2018). Separable reversible data hiding and encryption for HEVC video. *Journal of Real-Time Image Processing*, *14*(1), 171-182. doi: 10.1007/s11554-017-0727-y

Mareen, H., De Praeter, J., Van Wallendael, G., Lambert, P. (2018). A novel video watermarking approach based on implicit distortions. *IEEE Transactions on Consumer Electronics*, *64*(3), 250-258. doi: 10.1109/TCE.2018.2852258

McGowan, J. F. (2004). *AVI Overview.* Retrieved from `http://www.jmcgowan.com/avi.html` (accessed June 22, 2019)

Muchmore, M. (2021). *The best video editing software for 2021.* Retrieved from `https://sea.pcmag.com/video-editing/14625/the-best-video-editing-software-for-2020` (accessed May 5, 2021)

Nguyen, C., Tay, D. B. H., Deng, G. (2006, Dec). A Fast Watermarking System for H.264/AVC Video. In *APCCAS 2006 - 2006 IEEE Asia Pacific Conference on Circuits and Systems* (pp. 81–84). Singapore. doi: 10.1109/APCCAS.2006.342301

Noorkami, M., Mersereau, R. M. (2008). Digital Video Watermarking in P-Frames With Controlled Video Bit-Rate Increase. *IEEE Transactions on Information Forensics and Security*, *3*(3), 441-455. doi: 10.1109/TIFS.2008.923825

Ohm, J., Sullivan, G. J., Schwarz, H., Tan, T. K., Wiegand, T. (2012). Comparison of the coding efficiency of video coding standards—including High Efficiency Video Coding (HEVC). *IEEE Transactions on Circuits and Systems for Video Technology*, *22*(12), 1669-1684. doi: 10.1109/TCSVT.2012.2221192

Pang, L., Wong, K. (2019, Sept). A Data Embedding Technique for Spatial Scalable Coded Video Using Motion Vector Predictor. In *2019 IEEE International Conference on Image Processing (ICIP)* (pp. 4050–4054). Taipei, Taiwan. doi: 10.1109/ICIP.2019.8803422

Pang, L., Wong, K., Ito, R. (2019, Dec). Merge Mode-based Data Embedding in SHVC Compressed Video. In *2019 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)* (pp. 1–2). Taipei, Taiwan. doi: 10.1109/ISPACS48206.2019.8986316

Pang, L., Wong, K., Liong, S. T. (2017, Dec). Data embedding in scalable coded video. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)* (pp. 1190–1194). Kuala Lumpur, Malaysia. doi: 10.1109/APSIPA.2017.8282210

Saberi, Y., Ramezanpour, M., Khorsand, R. (2020). An efficient data hiding method using the intra prediction modes in HEVC. *Multimedia Tools and Applications*, *79*(43), 33279-33302. doi: 10.1007/s11042-020-09729-1

Schwarz, H., Marpe, D., Wiegand, T. (2007). Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *17*(9), 1103-1120. doi: 10.1109/TCSVT.2007.905532

Seregin, V., He, Y. (2014, Jan). *Common SHM Test Conditions and Software Reference Configurations.* In: proc JCT-VC 16th Meeting: San José, US, 9-17 January 2014.

Shanableh, T. (2012). Matrix encoding for data hiding using multilayer video coding and transcoding solutions. *Signal Processing: Image Communication*, *27*(9), 1025-1034. doi: http://dx.doi.org/10.1016/j.image.2012.06.003

Shanableh, T. (2018). Altering split decisions of coding units for message embedding in HEVC. *Multimedia Tools and Applications*, *77*(7), 8939-8953. doi: 10.1007/s11042-017-4787-6

Shen, L., Zhang, Z., Liu, Z. (2014). Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations. *IEEE Transactions on Circuits and Systems for Video Technology*, *24*(10), 1709-1722.

Sheng, Q., Wang, R., Pei, A., Wang, B. (2016). An Information Hiding Algorithm for HEVC Based on Differences of Intra Prediction Modes. In X. Sun, A. Liu, H.-C. Chao, E. Bertino (Eds.), *Cloud Computing and Security* (pp. 63–74). Springer, Cham. doi: 10.1007/978-3-319-48671-0_6

Singh, R., Nigam, S., Singh, A. K., Elhoseny, M. (2020). On Wavelet Domain Video Watermarking Techniques. In *Intelligent Wavelet Based Techniques for Advanced Multimedia Applications* (p. 65-76). Springer, Cham. doi: 10.1007/978-3-030-31873-4_5

Sullivan, G. J., Ohm, J. R., Han, W. J., Wiegand, T. (2012). Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *22*(12), 1649-1668. doi: 10.1109/TCSVT.2012.2221191

Sun, Y., Wang, J., Huang, H., Chen, Q. (2021). Research on scalable video watermarking algorithm based on H.264 compressed domain. *Optik*, *227*, 165911. doi: 10.1016/j.ijleo.2020.165911

Swati, S., Hayat, K., Shahid, Z. (2014). A Watermarking Scheme for High Efficiency Video Coding (HEVC). *PLoS ONE*, *9*(8), 1-8. doi: 10.1371/journal.pone.0105613

Sze, V., Budagavi, M., Sullivan, G. J. (2014). *High Efficieny Video Coding (HEVC): Algorithms and Architectures*. Springer, Cham. doi: 10.1007/978-3-319-06895-4

Tew, Y., Wong, K. (2014, Oct). Information hiding in HEVC standard using adaptive coding block size decision. In *2014 IEEE International Conference on Image Processing (ICIP)* (pp. 5502–5506). doi: 10.1109/ICIP.2014.7026113

Tew, Y., Wong, K. (2014). An overview of information hiding in H.264/AVC compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, *24*(2), 305-319. doi: 10.1109/TCSVT.2013.2276710

Tew, Y., Wong, K., Phan, R. C.-W., Ngan, K. N. (2018). Separable authentication in encrypted HEVC video. *Multimedia Tools and Applications*, *77*(18), 24165-24184. doi: 10.1007/s11042-018-5611-7

Universität-Hannover. (2013). *Test sequence.* `ftp://ftp.tnt.uni-hannover.de/testsequences/`.

Usman, M., He, X., Lam, K., Xu, M., Bokhari, S. M. M., Chen, J., Jan, M. A. (2019). Error Concealment for Cloud–Based and Scalable Video Coding of HD Videos. *IEEE Transactions on Cloud Computing*, *7*(4), 975-987. doi: 10.1109/TCC.2017.2734650

Van, L. P., Praeter, J. D., Wallendael, G. V., Cock, J. D., de Walle, R. V. (2015, Sept). Out-of-the-loop information hiding for HEVC video. In *2015 IEEE International Conference on Image Processing (ICIP)* (pp. 3610–3614). Quebec City, QC, Canada. doi: 10.1109/ICIP.2015.7351477

Vybornova, Y. (2020, Sept). A new watermarking method for video authentication with tamper localization. In L. J. Chmielewski, R. Kozera, A. Orłowski (Eds.), *Computer Vision and Graphics* (pp. 201–213). Springer, Cham. doi: 10.1007/978-3-030-59006-2_18

Wang, J., Wang, R., Xu, D., Li, W. (2015). An Information Hiding Algorithm for HEVC Based on Angle Differences of Intra Prediction Mode. *Journal of Software*, *10*(2), 213-221. doi: 10.17706/jsw.10.2.213-221

Wang, P., Zheng, Z., Ying, J. (2008, July). A novel video watermark technique in motion vectors. In *2008 International Conference on Audio, Language and Image Processing* (pp. 1555–1559). Shanghai, China. doi: 10.1109/ICALIP.2008.4590271

Wang, Z., Bovik, A. C., Sheikh, H. R., Simoncelli, E. P. (2004). Image quality assessment:

from error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600-612. doi: 10.1109/TIP.2003.819861

Westwater, R., Furht, B. (1997). The MPEG Video Compression Standard. In *Real-Time Video Compression: Techniques and Algorithms* (pp. 15–21). Boston, MA: Springer US. doi: 10.1007/978-0-585-32313-8_2

Wiegand, T., Sullivan, G. J., Bjontegaard, G., Luthra, A. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*(7), 560-576. doi: 10.1109/TCSVT.2003.815165

Winkler, S. (2012). Analysis of public image and video databases for quality assessment. *IEEE Journal of Selected Topics in Signal Processing*, *6*(6), 616-625. doi: 10.1109/JSTSP.2012.2215007

Wong, K., Chan, C., MaungMaung, A. (2020). Lightweight authentication for MP4 format container using subtitle track. *IEICE Transactions on Information and Systems*, *E103.D*(1), 2-10. doi: 10.1587/transinf.2019MUI0001

Xiph.org. (2013). *Derf's collection.* `https://media.xiph.org/video/derf`.

Xu, C., Ping, X., Zhang, T. (2006, Aug). Steganography in compressed video stream. In *First International Conference on Innovative Computing, Information and Control - Volume I (ICICIC'06)* (pp. 269–272). Beijing, China. doi: 10.1109/ICICIC.2006.15830

Xu, D., Wang, R., Shi, Y. Q. (2014). An improved reversible data hiding-based approach for intra-frame error concealment in H.264/AVC. *Journal of Visual Communication and Image Representation*, *25*(2), 410–422. doi: 10.1016/j.jvcir.2013.12.008

Xu, J., Wang, R., Huang, M., Wang, J., Xu, D. (2015). An information hiding algorithm for HEVC based on intra-prediction modes and Hamming+1. *Journal of Computational Information Systems*, *11*(15), 5587-5598. doi: 10.12733/jcis15121

Yang, G., Li, J., He, Y., Kang, Z. (2011). An information hiding algorithm based on intra-prediction modes and matrix coding for H.264/AVC video stream. *AEU - International Journal of Electronics and Communications*, *65*(4), 331-337. doi: 10.1016/j.aeue.2010.03.011

Yang, J., Li, S. (2018). An efficient information hiding method based on motion vector space encoding for HEVC. *Multimedia Tools and Applications*, *77*(10), 11979-12001. doi: 10.1007/s11042-017-4844-1

Yang, Y., Li, Z., Xie, W., Zhang, Z. (2019). High capacity and multilevel information hiding algorithm based on pu partition modes for HEVC videos. *Multimedia Tools and Applications*, *78*(7), 8423-8446.

Yulin Wang. (2004, Jun). An efficient data hiding approach for video retrieval on demand. In *ITRE 2004. 2nd International Conference Information Technology: Research and Education* (pp. 55–58). London, UK. doi: 10.1109/ITRE.2004.1393645