EVALUATION OF CODING TOOLS FOR SCREEN CONTENT IN HIGH EFFICIENCY VIDEO CODING

by

SHWETHA CHANDRAKANT KODPADI

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2015

Acknowledgements

I would like to thank Dr. K. R. Rao for being a supervisor, mentor and a source of inspiration encouraging me continuously during the course of my thesis.

I would like to thank Dr. J. Bredow and Dr. K. Alavi for serving on my thesis committee.

Last but not the least I would like to thank my family and friends for their support and guidance.

November 24, 2015

Abstract

EVALUATION OF CODING TOOLS FOR SCREEN CONTENT IN HIGH EFFICIENCY VIDEO CODING

Shwetha Chandrakant Kodpadi, MS

The University of Texas at Arlington, 2015

Supervising Professor: K R Rao

High Efficiency Video Coding (HEVC) [1] is the latest Video Coding Standard. It challenges the state-of-the-art H.264/AVC [3] Video Coding standard which is in current use in the industry by being able to reduce the bit rate by 50% and retaining the same video quality. It came into existence in the early 2013 although Joint Collaborative Team on Video Coding (JCT-VC) was formed in January 2001 to carry out developments on HEVC, and ever since then a huge range of developments has been going on. On 13 April 2013 [11], HEVC standard also called H.265 was approved by ITU-T, Joint Collaborative Team on Video Coding (JCTVC), a group of video coding experts from ITU-T Study Group (VCEG) and ISO/IEC JTC 1/SC 29/WG 11 (MPEG).

Coding of screen content video is becoming important because of applications such as wireless displays, graphics, remote desktop, remote gaming, automotive infotainment, cloud computing, distance education etc. Video in these applications often has mixed content consisting of natural video, text and graphics in the same picture. Coding of screen content, very high bit-rate and lossless coding, coding of auxiliary pictures (e.g., alpha transparency planes), and direct coding of RGB source content were included in HEVC Range Extensions (RExt) [20] and focused in HEVC SCC Extension.

As part of this thesis, SCM test model 5 is used as the latest Screen content model. Different coding tools and non-normative algorithms for screen content coding in HEVC

Version1, HEVC-RExt and HEVC-SCC are explained in detail. Coding effieciency of the main Screen content coding tools, Intra block Copy (IBC), Palette mode (PM), Adaptive Colour Transform (ACT) are evaluated using SCM5.2. Further, the coding efficiency of HEVC16.6+SCM5.2 is evaluated against HEVC16.4+RExt and state of the art H.264/AVC.

SCM with IBC gives bitrate savings from 5%-45%, SCM with PM gives 14-67 % and SCM with ACT gives 0.001% to 0.0038 % compared to SCM without IBC, without PM and without ACT, respectively. Also, SCM is evaluated against JM19.0 and HEVC-RExt. It can be seen that SCM gives bitrate saving of about 45% – 83% compared to HEVC+RExt under lossless condition and 23%-87% compared to JM19.0 (AVC) under lossless condition. Under lossy condition, SCM gives 57%-81% BD-bitrate savings compared to HEVC+RExt and 62%-88% BD-bitrate savings compared to JM19.0.

TABLE OF CONTENTS

List of Illustrations

List of Tables

Chapter 1

INTRODUCTION

1.1 Need for Compression

Everything we watch on screen today uses Video Compression. From Mobile phones to Television, wherever there is video there is compression. This is mainly because of one big reason. Raw video files are extremely big in terms of size. On an average, a raw video file (uncompressed video) of size 1920x1080 with frame rate of 25 fps and length of 60 minutes will be of size 559.8 GBytes or about 140 single layer DVDs. The bitrate for this video will be 156 MBytes/s. In communication systems, bandwidth and bitrate are directly proportional to each other. Considering the growing need for HD and UHD videos, compression becomes quintessential. Hence, compression of raw video is essential for both storage and transmission.

What is compression? Compression is a process in which an original data is reduced to fewer number of bits for storage or transmission. Compression can be lossless or lossy. In lossy compression the number of bits required to represent the data is reduced by removing the unimportant data. This kind of compression involves certain information loss. In cases where we cannot afford information loss, we use lossless compression. In lossless compression, number of bits are reduced by eliminating spatial redundancy.

Video Coding means compressing and decompressing the video. The device or software which can do this is called codec. Codec stands for encoder and decoder. Encoder carries out compression and decoder carries out decompression. Decompression of video bitstream is carried out before it is displayed on a screen. Video coding can be done using Image coding techniques but they do not enable efficient coding. Therefore, there is a list of video coding standards that have evolved from time to time. Figure 1 shows the evolution of video coding standards.

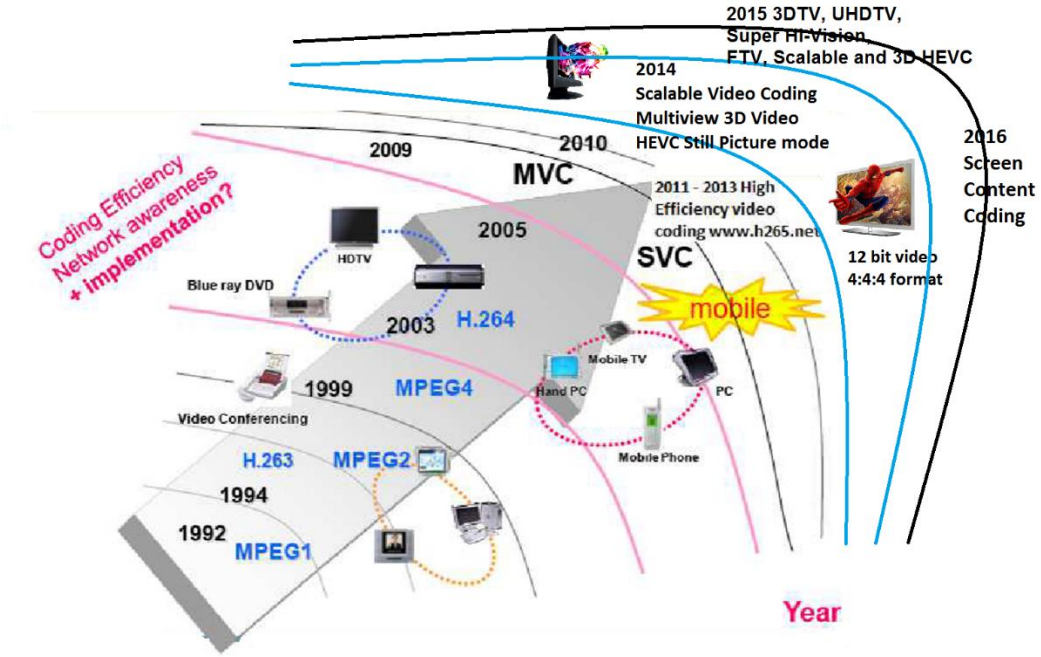


Figure 1.1 Evolution of video coding standards [37]

1.2 Video and its representation

A video sequence is a series of pictures over a period of time. For a video sequence to give a clear impression of motion, there must be as least as 25 pictures per second. Pictures per second is called frames per second (fps), where each picture is a frame. The minimum frame rate depends on the lighting conditions and the content to be displayed. High definition (HD) video today applies picture rates of 50–60 fps (Hz). For Ultra HD formats, picture rates of up to 120 fps are specified.

Each frame or a picture is represented by 2 dimensional array of samples with intensity values [46]. Each sample is called a pixel. If an image is monochrome, it has one single color component, the picture consists of a single sample array. For a color image or video, usually three color components are employed. Hence three intensity arrays, with one array for each component are required. Since the impression of any human perceivable color can be generated by the

mixture of three light sources i.e, red, green and blue, RGB color space is used to represent any color image. For coding purpose, RGB color space is converted to YC_bC_r color space. YC_bC_r is used since it can be represented using lesser number of bits than its RGB equivalent. In YC_bC_r, Y stands for the luma component which represents the brightness of the image and C_bC_r represents the chromaticity of the image. Our visual system is less sensitive to color than it is to structure and texture. So, in most of the applications where we need to represent image in limited number of bits, a compromise on chroma components can be done but not on luma components. Especially in consumer applications, sub-sampling of the chroma components is commonly applied.

The common notation for sub-sampling is YC_bC_r Y:X1:X2 where Y denotes the number of luma samples, The value X1 and X2 describe the sub-sampling format of the chroma components relative to the luma value. In the most common formats, Y = 4 is used. The X1 value specifies the horizontal sub-sampling. The value X2 = 0 indicates that the same X1 sub-sampling factor is applied for the vertical direction. X2 = X1 indicates that no vertical sub-sampling is performed and both chroma components apply the same horizontal sub-sampling factor [46]. A visualization for the three most common formats YC_bC_r 4:2:0, YC_bC_r 4:2:2 and YC_bC_r 4:4:4 is shown in Figure 1.2 and 1.3.

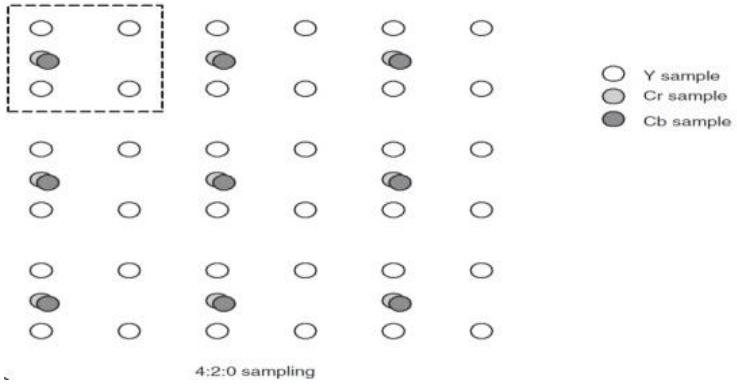


Figure 1.2: 4:2:0 sampling pattern [21]

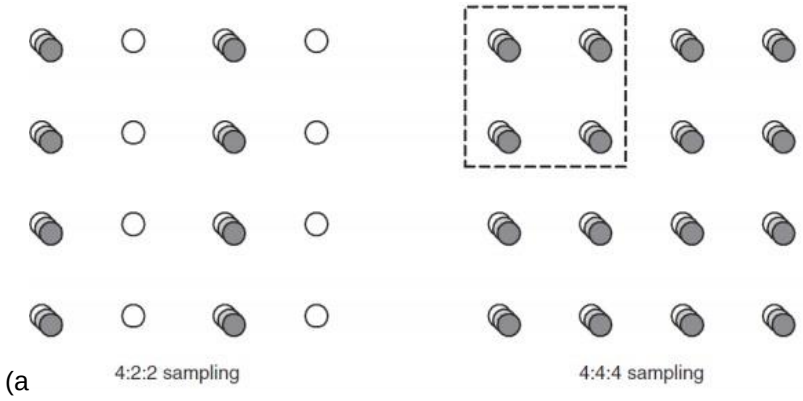


Figure 1.3: 4:2:2 and 4:4:4 sampling patterns [21]

Frames in video formats can be interlaced or progressive. In interlaced video, the even and odd sample lines are collected in the top and bottom field pictures, respectively. The two fields are alternately displayed [5]. A pair of these two fields constitutes a frame. In progressive scan, a frame constitutes a single picture. Interlaced videos were in big use during the CRT screen era. Video sequences can be of different aspect ratios, to make them suitable for the video displays. Aspect ratio of an image is the ratio between its width and height. Most commonly used aspect ratio is 16:9.

1.3 Fundamentals of Video Coding Systems

A video coding system will be organized according to the block diagram presented in Figure 1.4. First the video from source is subjected to pre-processing. During pre-processing, raw video is, trimmed, de-noised, color corrected or converted. This is followed by representing the input video sequence into a coded bitstream by generating a compact representation of the input video suitable for transmission and storage. This is called encoding. Encoded video bitstream is packaged into an appropriate format for transmission. With sending and receiving bitstream, transmission also implements potential methods for loss protection and loss recovery. At the receiver end, received bitstream is transformed into reconstructed video sequence. This is carried out by decoder. Decoding is the process of reconstructing from the encoded bitstream, using the

information supplied by the bitstream itself. Encoded bitstream includes header and payload. Generally, header contains all the decoding instruction. Reconstructed video signal is post-processed and displayed. The primary focus of the codec is on the encoding and decoding process [46].

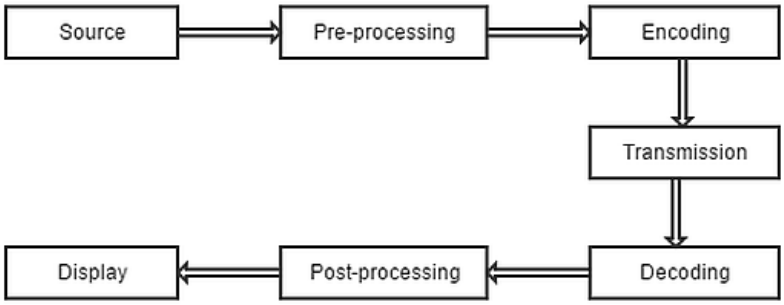


Figure 1.4: Video Coding System

Effectively video compression exploits both temporal and spatial redundancies. A frame which is compressed by exploiting the spatial redundancies is termed as intra frame and the frames which are compressed by exploiting the temporal redundancies are termed as inter frames. The compression of a inter frame requires a reference frame which will be used to exploit the temporal redundancies. In addition to this the inter frame is of two types namely a P - frame and B – frame. The P - frame make use of one already encoded/decoded frame which may appear before or after the current picture in the display order i.e. a past or a future frame as its reference, whereas the B - frame make use of two already encoded/decoded frames one of which is a past and the other being the future frame as its reference frames thus providing higher compression but also higher encoding time as it has to use a future frame for encoding [5] [6].

The classical order of frames called as the Group of Pictures or the GOP is given in Figure 1.5 [5] [6]. We observe that the very first frame is always encoded using intra frame encoding indicating transmission order: Display order: 3 by the letter I, this is because the first frame does not have any frame as a reference. The I frame is followed by a P – frame, as it has

the ability to make use of just one reference frame. After the P - frame the B - frame is encoded which makes use of both the I and the P - frames as its references. This pattern is followed with the subsequent frames where the P - frame takes the position of the I - frame.

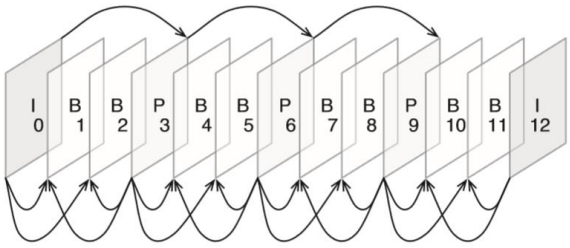


Figure 1.5: GOP Structure [5]

1.4 Scope of the thesis

In chapter 2, introduction and basics of High Efficiency Video Coding (HEVC) are discussed which is followed by discussion on screen content coding and proposed method in chapter 3. Chapter 4 gives the results based on the comparison between proposed method and the current method followed by conclusions and future work in chapter 5.

Chapter 2

OVERVIEW OF HIGH EFFICIENCY VIDEO CODING

2.1 HEVC Encoder and Decoder

Encoding is more challenging than decoding [12]. HEVC offers several choices for the encoders, leaving the user to choose the one which can represent the video most effectively. Also preliminary product implementations have already shown that HEVC encoding is entirely feasible [1]. Although decoding requirements for HEVC are relatively high compared to AVC standard, the increase is moderate.

Figures 2.1 and 2.2 represent block diagrams of standard encoder and decoder of HEVC respectively.

HEVC standard is based on block-based hybrid video coding. It implements block partitioning concept which partitions the picture into blocks. Each of these blocks is predicted by using either intra-picture or inter-picture prediction. Intra-picture exploits the spatial redundancy among blocks inside a picture. Inter-picture makes use of the temporal redundancy among pictures. The prediction error is taken by difference between original picture and the predicted picture in case of both intra and inter-picture prediction. The resulting prediction error is transmitted using transform coding followed by quantization and entropy coding. [12]

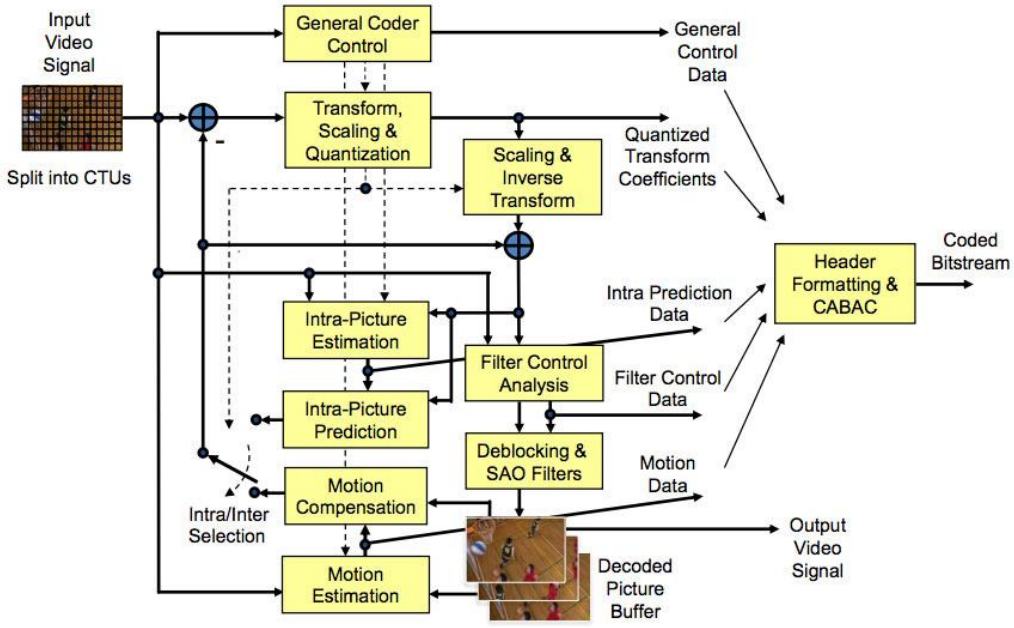


Figure 2.1: Encoder block diagram for HEVC [1]

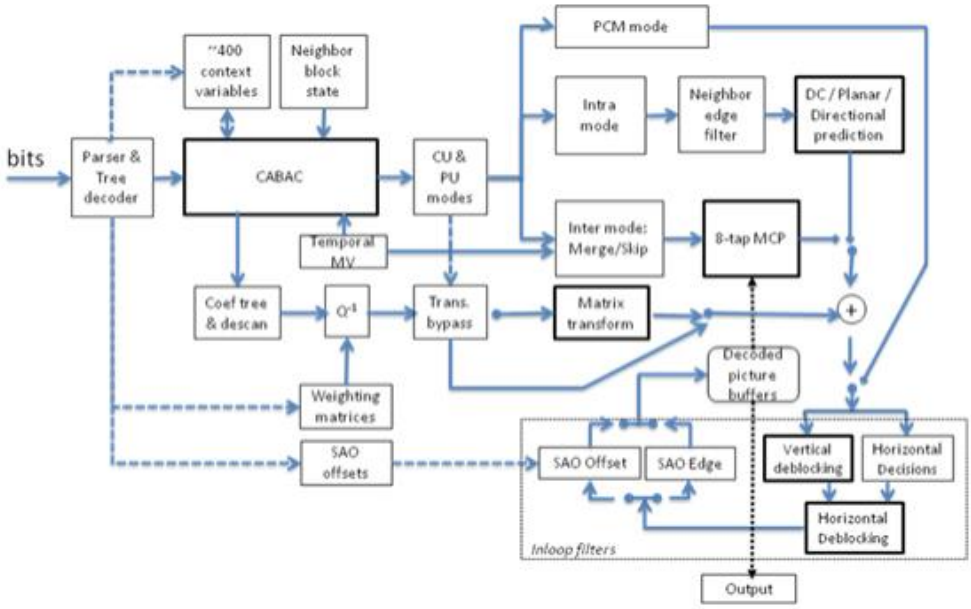


Figure 2.2: Decoder block diagram for HEVC [13]

2.2 HEVC features and coding tools

2.2.1 High-level syntax of HEVC

High-level syntax of HEVC provides a robust, flexible and extensible framework for carrying the coded video and associated information to enable the video content to be used in the most effective possible ways and in many different application environments. An HEVC bitstream consists of a sequence of data units called network abstraction layer (NAL) units. It includes the structure of the bitstream as well as signaling of high-level information that applies to one or more entire slices or pictures of a bitstream [12]. Figure 2.3 shows NAL unit header for HEVC.

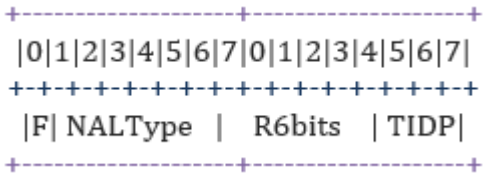


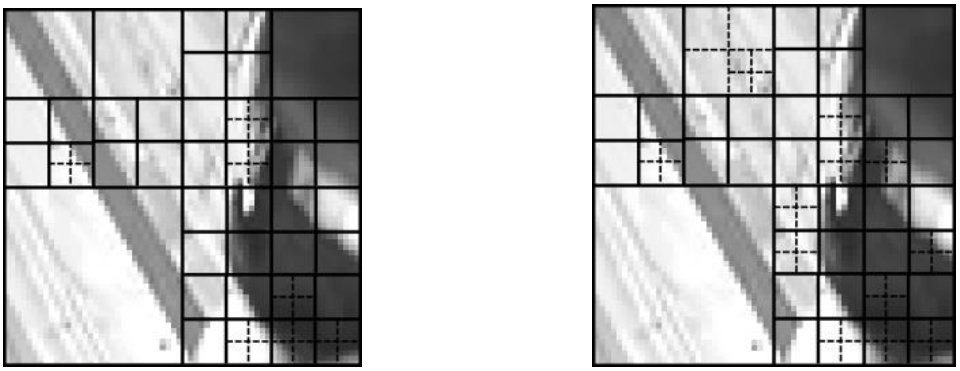
Figure 2.3: HEVC NAL unit header [14]

2.2.2 Block structures

The concept of macroblock in HEVC [15] is represented by the Coding Tree Unit (CTU). CTU size can be 16x16, 32x32 or 64x64, while AVC macroblock size is 16x16. Larger CTU size aims to improve the efficiency of block partitioning on high resolution video sequence. Larger blocks provoke the introduction of quad-tree partitioning (Figure 2.4) of a CTU into smaller coding units (CUs). A coding unit is a bottom-level quad-tree syntax element of CTU splitting. The CU contains a prediction unit (PU) and a transform unit (TU).

The TU is a syntax element responsible for storing transform data. Allowed TU sizes are 32x32, 16x16, 8x8 and 4x4. The PU is a syntax element to store prediction data like the intra-prediction angle or inter-prediction motion vector. The CU can contain up to four prediction units. CU splitting on PUs can be 2Nx2N, 2NxN, Nx2N, NxN, 2NxnU, 2NxnD, nLx2N and nRx2N (Figure 2.5) where 2N is a size of a CU being split. In the intra-prediction mode only 2Nx2N PU splitting

is allowed. An NxN PU split is also possible for a bottom level CU that cannot be further split into sub CUs.



a) b)

Figure 2.4: CTU splitting example with solid lines for CU split: (a) with PU splitting depicted as dotted lines (b) with TU splitting depicted as dotted lines [15]

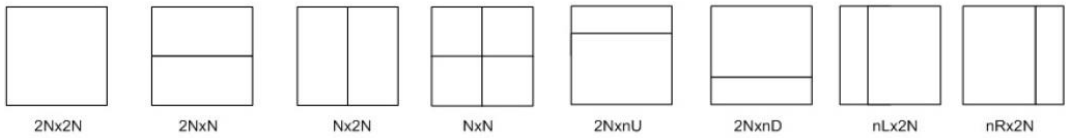


Figure 2.5: PU Splitting (U: Up, D: Down, L: left, R: Right) [15]

2.2.3 Parallelism features

In order to overcome the limitations of the parallelization strategies employed in H.264, HEVC provides VCL-based coding tools that are specifically designed to enable processing on high-level parallel architectures. Two new tools aiming at facilitating high-level parallel processing have been included in the HEVC standard [12]:

- Wavefront Parallel Processing (WPP): A parallel processing approach along the wavefront scheduling principle, which is based on a partitioning of the picture into CTU rows such that the dependencies between CTUs of different partitions, both in terms of predictive coding and entropy coding are preserved to a large extent.

- Tiles: A picture partitioning mechanism similar to slices, which is based on a flexible subdivision of the picture into rectangular regions of CTUs such that coding dependencies between CTUs of different partitions are prohibited. Figure 2.6 demonstrates slices and tiles.

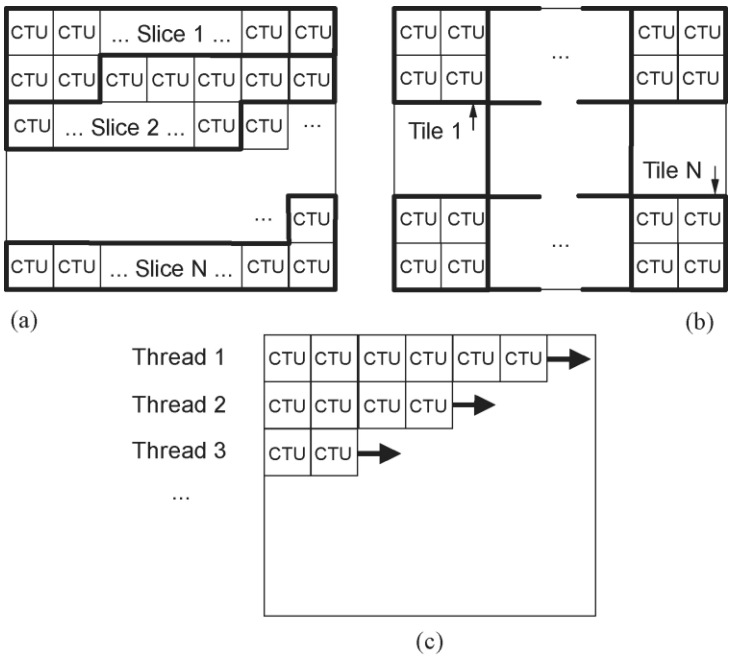


Figure 2.6: Subdivision of a picture into (a) slices and (b) tiles (c) Illustration of wavefront parallel processing [1]

2.2.4 Intra-picture Prediction

There are a total of 35 intra-prediction modes in HEVC: planar (mode 0), DC (mode 1) and 33 angular modes (modes 2-34 in Figure 2.7). DC intra-prediction is the simplest mode in HEVC. All PU pixels are set equal to the mean value of all available neighboring pixels. Planar intra-prediction is the most computationally expensive. It is a two- dimensional linear interpolation. Angular intra-prediction modes 2-34 are linear interpolations of pixel values in the corresponding directions. Vertical intra-prediction (modes 18- 34) is an up-down interpolation of neighboring

pixel values. Also, intra prediction can be done at different block sizes, ranging from 4 X 4 to 64 X 64 (whatever size the PU has) (In Figure 2.8).

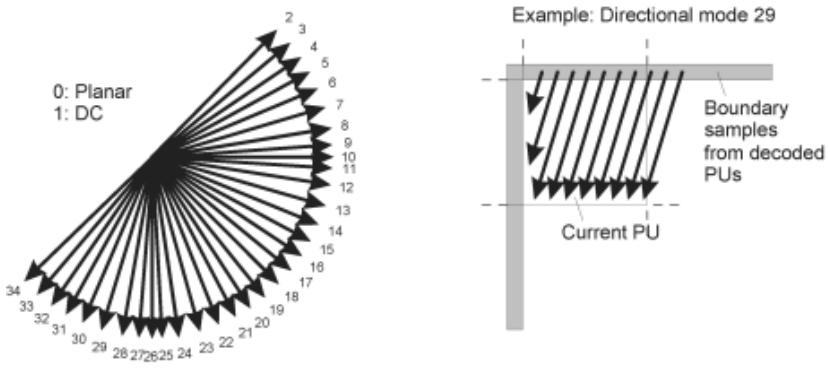


Figure 2.7: Modes and directional orientations for intra picture prediction for HEVC [1]

Luma intraprediction modes supported for different PU sizes.

PU Size	Intraprediction Modes
4 # 4	0–16, 34
8 # 8	0–34
16 # 16	0–34
32 # 32	0–34
64 # 64	0–2, 34

Figure 2.8: Luma intra prediction modes for different PU sizes in HEVC [8]

2.2.5 Inter-picture prediction

Inter-picture prediction [12] makes use of the temporal correlation between pictures in order to derive a motion-compensated prediction (MCP) for a block of image samples. For this block-based MCP, a video picture is divided into rectangular blocks.

For each block, a corresponding block in a previously decoded picture can be found that serves as a predictor. The general concept of MCP based on a translational motion model is illustrated in Figure 2.9. (Δx, Δy) are motion vectors and Δt is a reference index to a reference picture list.

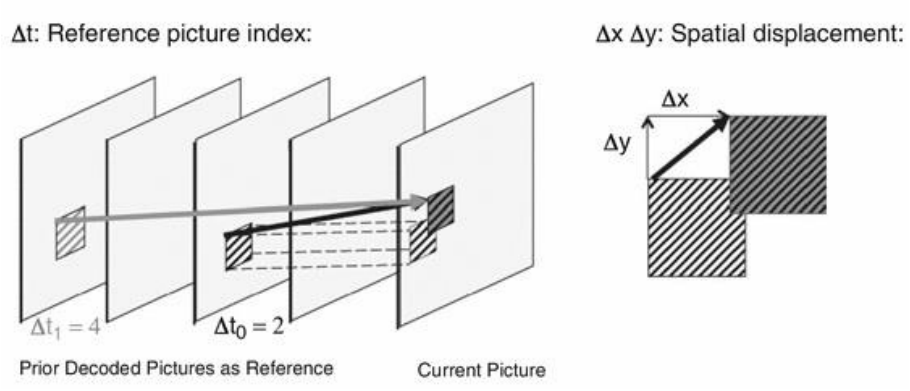


Figure 2.9: Inter-picture prediction concept and parameters using a translational motion model [12]

2.2.6 Transform and Quantization

Any residual data remaining after prediction is transformed using a block transform based on the integer Discrete Cosine Transform (DCT) [4]. Only for 4x4 intra luma, a transform based on Discrete Sine Transform (DST) is used. One or more block transforms of sizes 32x32, 16x16, 8x8 and 4x4 are applied to residual data in each CU. Then the transformed data is quantized. Figure 2.10 shows range of transform sizes.

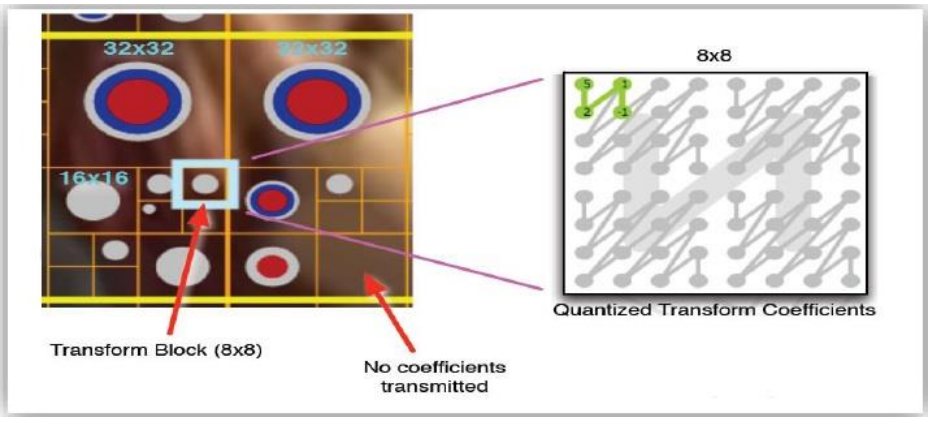


Figure 2.10: CTU showing range of transform (TU) sizes [16]

2.2.7 In-loop deblocking filter

The HEVC standard specifies two in-loop filters, a deblocking filter and a sample adaptive offset (SAO) [12]. The in-loop filters are applied in the encoding and decoding loops, after the inverse quantization and before saving the picture in the decoded picture buffer. The deblocking filter is applied first. It attenuates discontinuities at the prediction and transform block boundaries. The second in-loop filter, SAO, is applied to the output of the deblocking filter and further improves the quality of the decoded picture by attenuating ringing artifacts and changes in sample intensity of some areas of a picture. The most important advantage of the in-loop filters is improved subjective quality of reconstructed pictures. In addition, using the filters in the decoding loop also increases the quality of the reference pictures and hence also the compression efficiency. Figure 2.11 shows block diagram of HEVC decoder with deblocking and SAO filters.

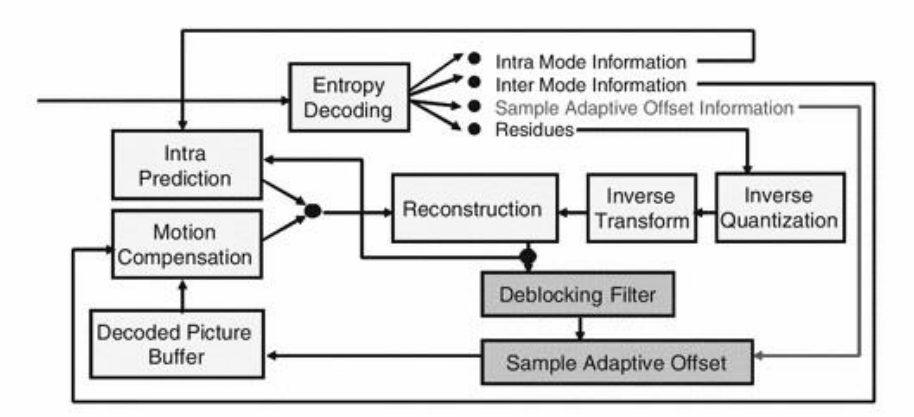


Figure 2.11: Block diagram of HEVC decoder with deblocking and SAO filters [12]

2.2.8 Entropy coding

Context adaptive binary arithmetic coding (CABAC) is used for entropy coding. This is similar to the CABAC scheme in H.264/MPEG-4 AVC [3], but has undergone several changes to improve its throughput speed (especially for parallel-processing architectures) and its compression performance, and to reduce its context memory requirements. Figure 2.12 shows three key operations in CABAC, binarization, context selection, and arithmetic coding.

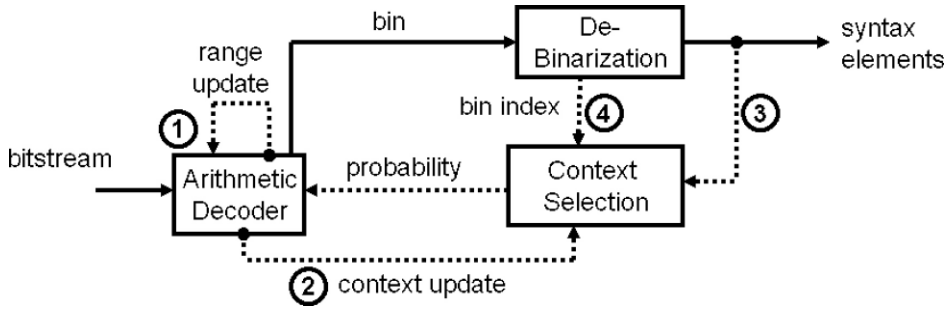


Figure 2.12: Three key operations in CABAC: binarization, context selection, and arithmetic coding. (Feedback loops in the decoder are highlighted with dashed lines) [17]

2.2.8 Profiles, Levels and Tiers

Profiles define the syntax and coding features that can be used for the video content. HEVC released 3 profiles in its first version [12]:

- Main Profile: This profile represents video data with 8 bits per sample and the typical representation with a “luma” brightness signal and two “chroma” channels that have half the luma resolution both horizontally and vertically.
- Main still Picture profile: This profile is a subset of the capabilities of the Main profile. Typically used for images, or for the extraction of the snapshots from video sequences.
- Main 10 profile: This profile supports upto 10 bits per sample. This profile is a superset of the capabilities of the Main profile. Provides increased bit depth for increased brightness dynamic range, extended color-gamut content, or simply higher fidelity color representations to avoid contouring artifacts and reduce rounding errors.

Version 2 of HEVC adds 21 range extensions profiles, two scalable extensions profiles, and one multi-view profile [18] [19]: Monochrome, Monochrome 12, Monochrome 16, Main 12, Main 4:2:2 10, Main 4:2:2 12, Main 4:4:4, Main 4:4:4 10, Main 4:4:4 12, Monochrome 12 Intra, Monochrome 16 Intra, Main 12 Intra, Main 4:2:2 10 Intra, Main 4:2:2 12 Intra, Main 4:4:4

Intra, Main 4:4:4 10 Intra, Main 4:4:4 12 Intra, Main 4:4:4 16 Intra, Main 4:4:4 Still Picture, Main 4:4:4 16 Still Picture, High Throughput 4:4:4 16 Intra

All of the inter frame range extensions profiles have an Intra profile. Scalable Main, Scalable Main 10, Multi-view Main [19] [38].

Different bit rates are required for consumer use and professional use. For this purpose, Tiers was introduced. Several levels in HEVC have both a Main tier and a High tier of capability specified, based on the bit rates they are capable of handling. [18]

Levels define the degree of capability within a given feature set. Levels of capability are defined to establish the picture resolution, frame rate, bit rate, buffering capacity, and other aspects that are matters of degree rather than basic feature sets. The HEVC standard defines fifteen levels. [30][18] Following are the 15 levels: none, 1, 2, 2.1, 3, 3.1, 4, 4.1, 5, 5.1, 5.2, 6, 6.1, 6.2, and 8.5. Table 2.1 shows defined resolution and frame rate for each level. For levels below level 4 only the Main tier is allowed. A decoder that conforms to a given tier/level is required to be capable of decoding all bit streams that are encoded for that tier/level and for all lower tiers/levels.

Table 2.1 Defined resolution and frame rate for each level [18] [30]

Level	Resolution	Frames per second (fps)
1	128×96, 176×144	33.7 , 15.0
2	176×144, 320×240, 352×240, 352×288	100.0, 45.0, 37.5, 30.0
2.1	320×240, 352×240, 352×288, 352×480, 352×576, 640×360	90.0, 75.0, 60.0, 37.5, 33.3, 30.0
3	352×480, 352×576, 640×360, 720×480, 720×576, 960×540	84.3, 75.0, 67.5, 42.1, 37.5, 30.0
3.1	720×480, 720×576, 960×540, 1280×720	84.3, 75.0, 60.0, 33.7
4	1280×720, 1280×1024, 1920×1080, 2048×1080	68.0, 51.0, 32.0, 30.0
4.1	1280×720, 1280×1024, 1920×1080, 2048×1080	136.0, 102.0, 64.0, 60.0

Table 2.2 – continued

5	1920×1080, 2048×1024, 2048×1080, 2048×1536, 2560×1920,3672×1536, 3840×2160, 4096×2160	128.0, 127.5, 120.0, 85.0, 54.4, 46.8, 32.0, 30.0
5.1	1920×1080, 2048×1024, 2048×1080, 2048×1536, 2560×1920, 3672×1536, 3840×2160, 4096×2160	256.0, 255.0, 240.0, 170.0, 108.8, 93.7, 64.0, 60.0
5.2	1920×1080, 2048×1024, 2048×1080, 2048×1536, 2560×1920, 3672×1536, 3840×2160, 4096×2160	300.0, 300.0, 300.0, 300.0, 217.6, 187.5, 128.0, 120.0
6	3840×2160, 4096×2048, 4096×2160, 4096×2304, 7680×4320, 8192×4320	128.0, 127.5, 120.0, 113.3 32.0, 30.0
6.1	3840×2160, 4096×2048, 4096×2160, 4096×2304, 7680×4320, 8192×4320	256.0, 255.0, 240.0, 226.6, 64.0, 60.0
6.2	3840×2160, 4096×2048, 4096×2160, 4096×2304, 7680×4320, 8192×4320	300.0, 300.0, 300.0, 300.0, 128.0, 120.0

In August 2013 [2], some standard extensions for HEVC were developed. They basically fall into three areas: 1) the range extensions, which expand the range of bit depths and color sampling formats supported by the standard, and include an increased emphasis on high-quality coding, lossless coding, and screen-content coding; 2) the scalability extensions, which enable the use of embedded bitstream subsets as reduced-bit-rate representations of the video content; and 3) the 3D video extensions, which enable stereoscopic and multiview representations and consider newer 3D capabilities such as the use of depth maps and view-synthesis techniques.

2.3 Summary

In this chapter, HEVC and its features and tools are discussed. Chapter 3 gives introduction about Screen Content coding and detailed explanation of Screen Content coding tools in HEVC.

Chapter 3

SCREEN CONTENT CODING

Coding of screen content video is becoming important because of applications such as wireless displays, graphics, remote desktop, remote gaming, automotive infotainment, cloud computing, distance education etc. Video in these applications often has mixed content consisting of natural video, text and graphics in the same picture. Coding of screen content, very high bit-rate and lossless coding, coding of auxiliary pictures (e.g., alpha transparency planes), and direct coding of RGB source content were included in HEVC Range Extensions (RExt) standard [20]. Figure 3.1 shows the example video snapshots of the screen content consisting of graphics, text, natural camera shots.

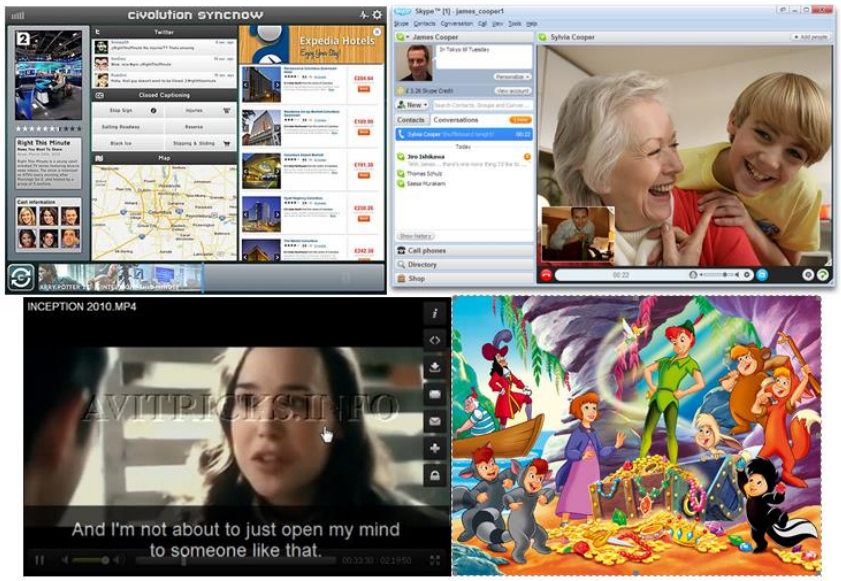


Figure 3.1: video snapshots of the screen content consisting of graphics, text, natural camera shots

Unlike camera-captured content, screen content frequently contains no sensor noise, and such content may have large uniformly flat areas, repeated patterns, highly saturated or a limited number of different colors, and numerically identical blocks or regions among a sequence of

pictures. These characteristics, if properly leveraged, can offer opportunities for significant improvements in compression efficiency over a coding system designed primarily for camera-captured content.

HEVC version 1 concentrated on coding tools which can improve performance on camera captured content. Residual Scalar Quantization (RSQ) and Base Colors and Index Map (BCIM) [4] were proposed early during the HEVC development process. Because screen content often has high contrast and sharp edges, RSQ directly quantized the intra prediction residual, without applying a transform. BCIM took advantage of the observation that the number of unique colours in screen content pictures is usually limited as compared to camera-captured content. RSQ and BCIM could respectively be considered early forms of transform skip, which is part of HEVC version 1, and palette mode.

Additional modes such as transform bypass where both the transform and quantization steps are bypassed for lossless coding, and the use of differential pulse code modulation (DPCM) for sample-based intra prediction were proposed [59]. Figure 3.2 shows transform bypass mode. Because screen content often contains repeated patterns, dictionary and Lempel-Ziv coding tools were shown to be effective at improving coding efficiency, especially on pictures containing text and line graphics [60], [61], [62].

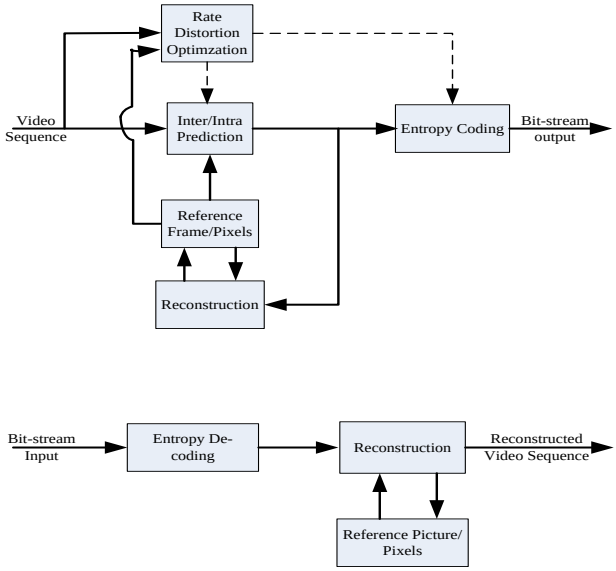


Figure 3.2 – Transform Bypass Mode [59]

Following section describes screen content coding support in HEVC version 1 and HEVC range extensions.

3.1 Screen Content Coding Support in HEVC

HEVC screen content coding extension (HEVC-SCC) is developed based on HEVC version 1 [1] and HEVC range extensions (HEVC-REext) [12] [65]. Thus, it inherits the coding structure and coding tools of HEVC version 1 and HEVCREext. HEVC-SCC also maintains backward compatibility to HEVC version 1 and HEVC-REext. Although a large importance was not given to Screen content during the development of HEVC version 1 and HEVC-Rext, it was considered during the design process. Following section explains few coding tools that are part of HEVC version 1 and HEVC-Rext, which targeted Screen content. Figure 3.3 demonstrates the framework of the Screen Content Coding.

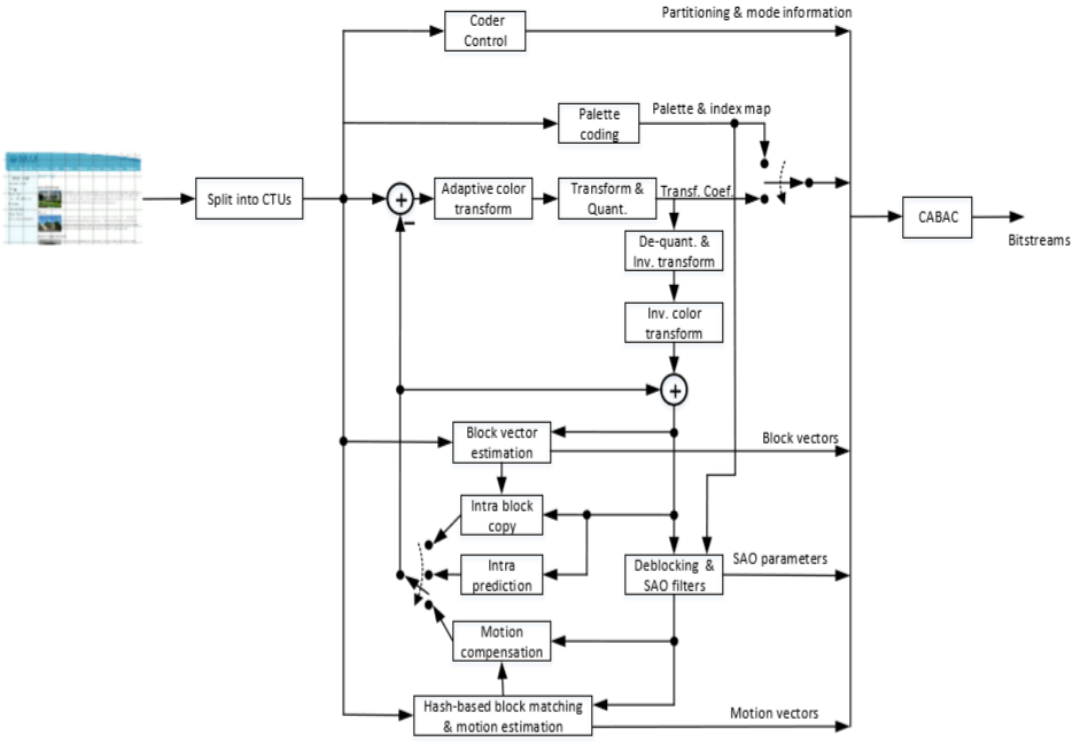


Figure 3.3: Framework of the Screen Content Coding [58]

3.1.1 Transform Skipping

For those blocks in Screen content, skipping the transform and quantizing data in the spatial domain can be a better choice, as was demonstrated for H.264/AVC in [58]. HEVC version 1 can skip the transform for a 4×4 TU, whether it is intra or inter. This transform skip is equivalent to applying an identity transform to the TU. Thus, the quantization process after applying transform skip is the same as that applied after the spatial transform. It turns out that such a simple design can lead to significant coding efficiency improvement for screen content, e.g. the bit-saving brought by the transform skip mode is about 7.5% for typical 4:2:0 screen content . When applied to 4:4:4 screen content, the coding gain for transform skip is much larger, ranging from 5.5% to 34.8%.

The Transform Skip Mode (TSM) defines the transform skip in one or both directions on which a transform would be applied under normal conditions, Figure 3.4. As illustrated in Figure

3.4(a), TSM also covers the traditional approach where the transform is applied on both rows and columns of a block. A summary of TSM modes is given in Table 3.1.

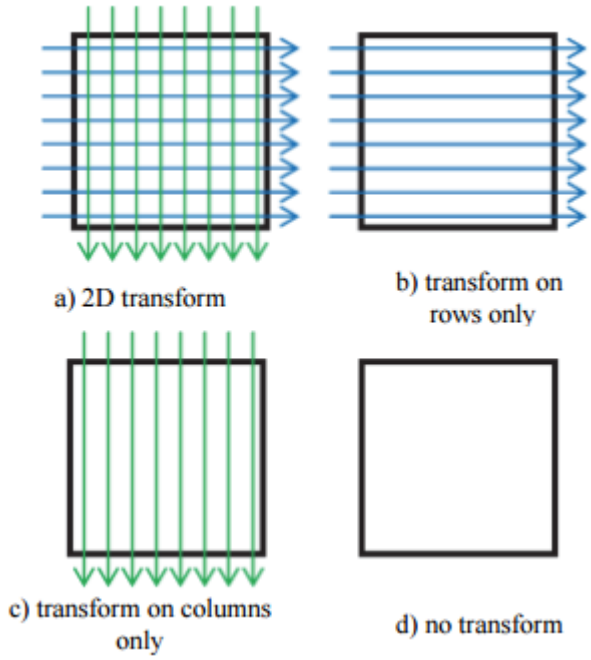


Figure 3.4: Transform choices enabled by TSM [66]

Table 3.1: Transform skip modes [66]

TSM mode	Horizontal direction	Vertical direction
TS0	Transformed	Transformed
TS1	Transformed	Skipped
TS2	Skipped	Transformed
TS3	Skipped	Skipped

HEVC-RExt After HEVC version 1, HEVC-RExt was developed to support non-4:2:0 colour formats, e.g. 4:4:4 and 4:2:2, and high bit-depth video, e.g. up to 16-bit. Because most screen content is captured in the 4:4:4 colour format, which is not supported by HEVC version 1,

more attention was given to coding of screen content in HEVC-Rext [65][58]. Following are the methods that improved coding efficiency for screen content in HEVC-RExt.

3.1.2 Improvements to transform skip mode

HEVC version 1 only supports transform skip for 4×4 TUs. HEVC-RExt extends transform skip to all TUs, regardless of their size [58] [17]. Enabling transform skip for all TUs has two benefits. One is that the coding efficiency for screen content can be further improved. The other is that encoders have the flexibility to exploit the transform skip mode. For example, a specific encoder may support only large transform units so that the encoding complexity can be reduced. If transform skip is allowed only for 4×4 TUs, the performance of such an encoder would be affected adversely since it cannot exploit the benefit brought by transform skip, which can be much more noticeable for screen content.

3.1.3 Residual differential pulse code modulation (RDPCM)

Even after intra prediction, there is still correlation in the residual signal which can be exploited [48]. Residual differential pulse code modulation (RDPCM) predicts the current residual using its immediately neighboring residual. In HEVC-RExt, RDPCM was proposed for intra lossless coding [58] [51]. Then it was extended to lossy coding and inter coding. In Figure 3.5, $r_{i,j}$ $0 \le I \le N$-1, $0 \le j \le N$-1 denotes the residuals at the (i, j) position of a NxN block. Residual samples denoted by $r_{i,j}$ after the RDPCM are given with the differential coding. For example, if the vertical RDPCM is used, the top-most residual samples are first used for predicting the second row of the samples, and the RDPCM process is repeated to the end of the row. The RDPCM directions were aligned to the prediction direction. In other words, the vertical/horizontal RDPCM was implicitly selected with vertical/horizontal angular prediction, respectively.

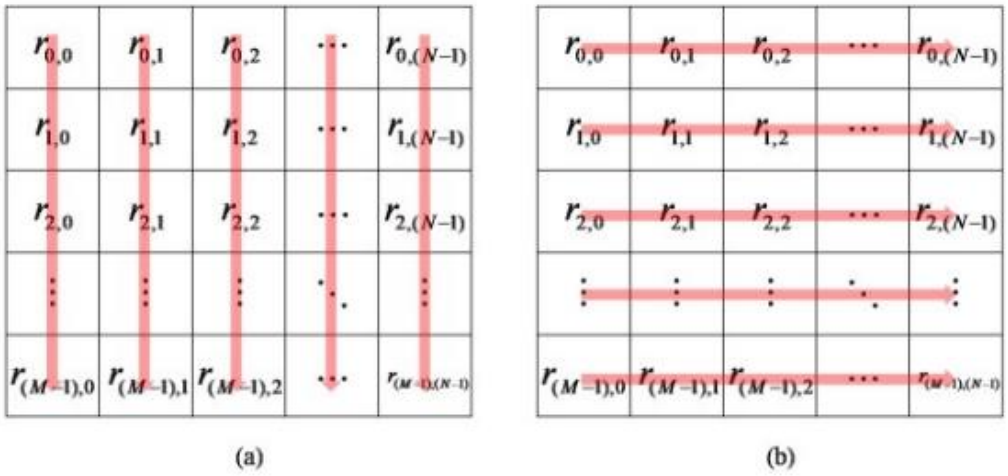
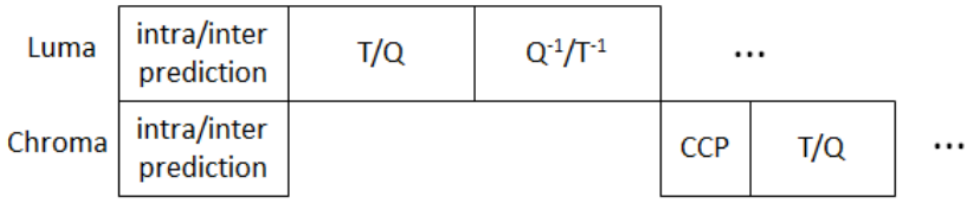


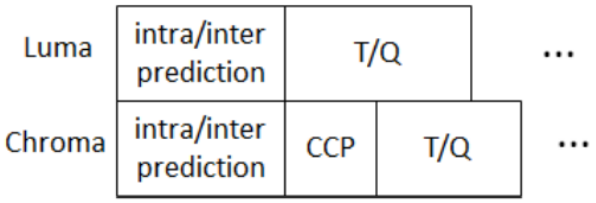
Figure 3.5: Two RDPCM modes when the intra prediction mode is (a) vertical and (b) horizontal directions [48]

3.1.4 Cross-component prediction (CCP)

CCP [58] [67] was proposed to exploit correlation among color components [24]. In CCP, First, luma component is set as the predictor component, and two chroma components are predicted separately from the luma component. Therefore, there are two α values, one for the C_b or B component, and the other one for the C_r or R component. These values are coded into bitstream, so there is no need to calculate these values at the decoder side. The signaling of α occurs at TU level in order to maximally decrease the local correlation. This is roughly illustrated in Figure 3.6. CCP is very effective in coding of screen content.



(a) CCP using the reconstructed luma residual signal.



(b) CCP using the original luma residual signal.

Figure 3.6 CCP using the original luma residual signal [68]

3.1.5 Other improvements

Some other aspects of HEVC-RExt, although not specifically designed for screen content coding, also improve the coding efficiency for screen content. For example, the initialization of Rice parameters based on previous similar blocks was primarily designed for high bit depth coding; but it also showed improvement for coding screen content [58].

Unlike HEVC version 1 and HEVC-RExt, the tools added for the HEVC-SCC extension focus primarily on coding screen content. HEVC-SCC is based on the HEVC framework with new tools added to it [58] [64].

The new coding tools in HEVC-SCC are:

1. Intra Block Copy
2. Palette mode
3. Adaptive Color Transform
4. Adaptive motion vector resolution

In the following section. We will discuss the details of these coding tools.

3.2 Intra Block Copy

HEVC-SCC introduces a new CU mode in addition to the conventional intra and inter modes, referred to as intra block copy (IBC).Intra block copy (or intra motion compensation mentioned in the first place) was studied a decade ago [69]. Recently, it was brought up again and introduced into HEVC-RExt [58] [65] to enable inter-alike motion estimation and compensation technology using fixed block size for better coding efficiency.

When a CU is coded in IBC mode, the PUs of this CU find similar reconstructed blocks within the same picture. Instead of searching the reference in previously (temporally) reconstructed frame, it searches then reconstructed region in the current frame and carries the block vector and compensation residual to the decoder. This technology does not show impressive performance gains for camera captured content but significant gains for screen content.

IBC was proposed in the context of AVC/H.264 [3] but the coding gain was not consistently high across different test sequences, which at the time were primarily camera-captured sequences and not screen content material. IBC has been a part of HEVC-SCC test model since the beginning of the HEVC-SCC development although it was proposed to be part of HEVC-RExt.

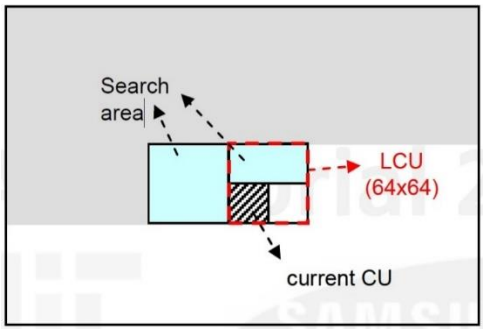


Figure 3.7: Example for Intra block copy [44]

For each Intra BC block, the prediction signal is obtained from its reference block pointed by the corresponding block vector (BV). Previously, the Intra BC mode is signaled at coding unit (CU) level, and it supports various CU partitions, including 2Nx2N, 2NxN, Nx2N, NxN partitions. A block vector was coded to specify the location of the predictor block.(citeibcforhevc) Currently, IBC is performed at the prediction unit (PU) level and is treated as an inter PU. The current picture can also be used as reference picture using the inter mode design [58].

Since both IBC and inter mode share the concept of vectors representing displaced blocks, it is natural to unify the design of IBC and inter mode. Methods to unify these modes have shown that also using the inter mode syntax design for IBC is an adequate choice.

There are few constraints on the way IBC mode is operated. Predictor block should not be from the current CU and they should belong to the same slice and tile. The predictor block should be entirely contained in the search region as shown in the figure. It is so designed to avoid affecting the parallel processing capability provided by wavefronts. Block vector precision is full-pel.

Following are the encoding algorithms that are developed for better coding efficiency of IBC mode. These updated non normative methods are part of SCM5.2

3.2.1 Intra Block Vector search

In order to evaluate the rate-distortion (RD) cost of using the IBC mode, for each CU, block matching (BM) is performed at the encoder to find the optimal block vector. In SCM, first a local area search is performed. This is followed by a search over the entire picture for certain CU sizes [64].

3.2.1.1 Local block vector search for IBC mode

The following modifications are made in SCM test model 5, for Local block vector search for IBC mode [64]. In order to find the optimal block vector from the local region, luma as well as chroma information is utilized.

In the first step, the four best block vectors are selected according to their RD cost, where $\mathrm{RD\_cost} = \mathrm{SAD}_{\mathrm{luma}} + \mathrm{Lambda} \times \mathrm{BV}_{\mathrm{bits}}$.

In second step, both the luma and chroma components are used in the calculation of the SAD for the four best block vectors selected from step 1. The block vector with the minimum RD cost is selected as the locally optimal block vector, $\mathrm{BV}_{\mathrm{opt}}^{\mathrm{local}}$. The RD cost in this step is calculated as

$\mathrm{RD\_cost} = \mathrm{SAD}_{\mathrm{luma}} + \mathrm{SAD}_{\mathrm{chroma}} + \mathrm{Lambda} \times \mathrm{BV}_{\mathrm{bits}}$.

The RD cost corresponding to $\mathrm{BV}_{\mathrm{opt}}^{\mathrm{local}}$ is denoted by $\mathrm{RD\_cost}_{\mathrm{opt}}^{\mathrm{local}}$.

3.2.1.2 Global block vector search for IBC mode

In addition to the local search, global block vector search is performed for 8×8 and 16×16 blocks [64]. The global search area is a portion of the reconstructed current picture before loop filtering, as depicted in 3.8. Additionally, when slices/tiles are used, the search area is further restricted to be within the current slice/tile. For 16×16 blocks, only a one-dimensional search is conducted over the entire picture.

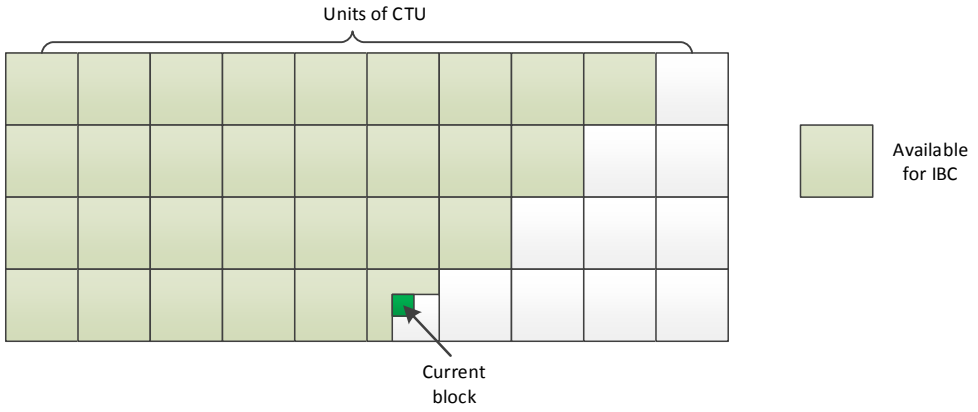


Figure 3.8: IBC prediction area [64]

This means that only the block vectors with one zero component are searched, i.e. the search is horizontal or vertical only. For 8×8 blocks, a hash-based search is used to speed up the full picture search. The bit-length of the hash table entry is 16. Each node in the hash table records the position of each block vector candidate in the picture. With the hash table, only the block vector candidates having the same hash entry value as that of the current block are examined.

The 16-bit hash entries for the current block and the reference block are calculated using the original pixel values. Let Grad denote the gradient of an 8×8 block and let DC0, DC1, DC2 and DC3 denote the DC values of the four 4×4 sub-blocks of the 8×8 block. Then, the 16-bit hash entry H is calculated as

H=MSB(DC0,3)<<13+MSB(DC1,3)<<10+MSB(DC2,3)<<7+MSB(DC3,3)<<4+MSB(Grad,4)

Where, MSB(X, n) represents the n most significant bits of X.

For 8×8 and 16×16 blocks, let the block vector with the minimum RD cost corresponding to the full-picture search be denoted by $\mathrm{BV}_{\mathrm{opt}}^{\mathrm{global}}$ and the corresponding RD cost be denoted by $\mathrm{RD\_cost}_{\mathrm{opt}}^{\mathrm{global}}$. Then, $\mathrm{RD\_cost}_{\mathrm{opt}}^{\mathrm{global}}$ and $\mathrm{RD\_cost}_{\mathrm{opt}}^{\mathrm{local}}$ are compared to choose the block vector with the minimum RD cost.

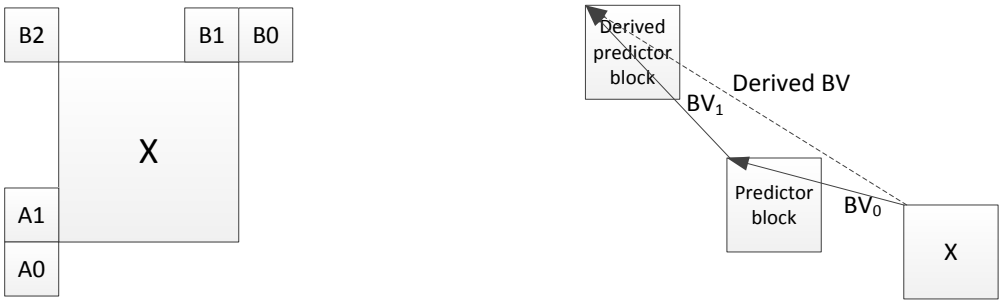
3.2.1.3 Fast block vector search for IBC mode

In addition to the local and global block vector search, some fast search and early termination methods are employed [64]. The fast IBC search is performed after evaluating the RD cost of inter mode, if the residual of inter prediction is not zero.

In the fast search, the SAD-based RD costs of using a set of block vector predictors are calculated. The set includes the five spatial neighboring block vectors as utilized in inter merge mode (as shown in Figure 3.9 (a)) and the last two coded block vectors. In addition, the derived block vectors of the blocks pointed to by each of the aforementioned block vector predictors are

also included (see Figure 3.9 (b)). This fast search is performed before the evaluation of intra prediction mode. It is applied only to 2Nx2N partition of various CU sizes.

If the residue of fast IBC search is not zero, then regular intra prediction mode will be evaluated followed by a full range IBC search.



(a) Spatial BV predictors (b) Derived BV predictors

Figure 3.9: IBC prediction area [64]

3.2.1.4 IBC block vector signalling

In SCM 4, the block vector signalling for the IBC mode is unified with the inter signalling. This is accomplished by adding the current picture to the reference picture list [64]. In SCM 5, the following IBC aspects were changed:

A disabling flag is added for IBC at the picture level. Non-integer IBC chroma displacement vectors are allowed. The current picture may appear both in list 0 and list 1, however weighted prediction is disabled when one of the motion vectors points to the current picture. In SCM 5, I-slices are possible even when IBS is enabled at the picture level. The current picture is placed at the last position in the reference picture list and the list is long enough to contain it. When in-loop filtering is disabled for the current picture and IBC is enabled, the unfiltered current picture is considered a part of DPB.

3.2.2 Inter block search

Compared to the HEVC Range extensions test model 7, SCM modifies the inter block search in two ways [58] [64]. The inter search is modified to adapt to the characteristics commonly found in screen content sequences. Furthermore, the inter block search is extended to the whole picture using hash-based techniques. Inter search in HEVC Range extensions has improved by using Multistage approximate SAD computation, Modified initial search, Modified early skip detection.

3.2.2.1 Hash-based inter search

Hash-based search is applied only to 2N×2N blocks. An eighteen bit hash based on original pixels is used [58] [64]. The first 2 bits are determined by the block size, e.g. 00 for 8x8, 01 for 16x16, 10 for 32x32, and 11 for 64x64. The remaining 16 bits are determined by the original pixels.For one block, two hash values are calculated in a similar way but with different CRC truncated polynomials. The first hash value is used for retrieval and the second hash value is used to exclude some of the hash conflicts. The hash value is calculated as follows:

- For each row, calculate the 16-bit CRC value for all the pixels Hash [i].
- Group the row hash values together (Hash [0] Hash [1]…) and then calculate the 24-bit CRC value H.
- The lower 16 bits of H will be used as the lower 16 bits of hash value of the current block.

Early termination based on hash search is also applied. If all of the following conditions are satisfied, the RD optimization process will be terminated without checking other modes and CU splitting.

- Hash match is found.
- The quality of the reference block is no worse than the expected quality of the current block (the QP of the reference block is no greater than the QP of the current block).

- Current CU depth is 0.

3.3 Palette Coding

3.3.1 Palette mode

Color table/palette method was studied almost two decades ago [31] [58]. For screen content, it is observed that for many blocks, a limited number of different colour values may exist. Thus, palette mode enumerates those colour values and then for each sample, sends an index to indicate to which colour it belongs. Palette mode can be more efficient than the prediction-then-transform representation. The palette mode was adopted into the HEVC SCC test model 2 at the 18th JCT-VC meeting.

3.3.1.1 Overview of palette mode

The palette mode is signalled at the CU level and is typically used when most of the pixels in the CU can be represented by a small set of representative colour values. Palette mode is useful for lossy and lossless coding [58].

Samples in the CU are represented by a small set of representative colour values. This set is referred to as the palette. It is also possible to indicate a sample that is outside the palette by signalling an escape symbol followed by (possibly quantized) component values [64]. This is illustrated in Figure 3.10.

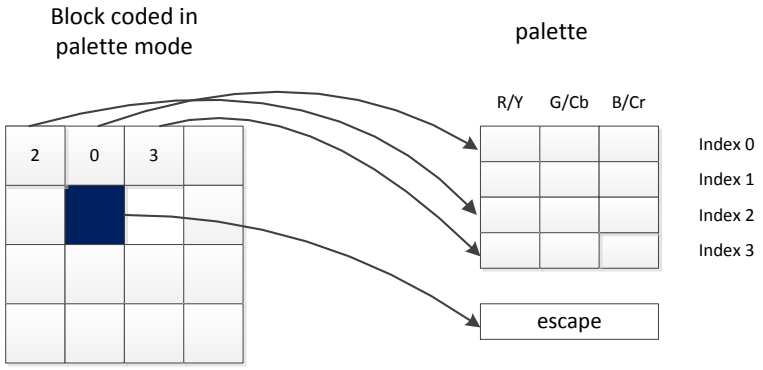


Figure 3.10: Example of a block coded in palette mode [64]

In this example, the palette size is 4. The first 3 samples use palette entries 2, 0, and 3, respectively, for reconstruction. The blue sample represents an escape symbol. If escape symbols are present, the palette is augmented by one and the last index is assigned to the escape symbol. Thus, in Figure 3.10, index 4 is assigned to the escape symbol.

For decoding a palette-coded block, the decoder needs to have the following information: Palette entries and Palette indices. In addition, on the encoder side, it is necessary to derive the appropriate palette to be used with that CU.

3.3.1.2 Palette derivation

For derivation of the palette for lossy coding, k-means clustering algorithm was used. For lossless coding, a different derivation process is used [58] [64]. A histogram of the samples in the CU is calculated. The histogram is sorted in a decreasing order of frequency. Then, starting with the most frequent histogram entry, each entry is added to the palette. Histogram entries that occur only once are converted to escape symbols if they are not a part of the palette predictor.

After palette derivation, each sample in the block is assigned the index of the nearest palette entry. Then, the samples are assigned to 'INDEX' or 'COPY_ABOVE' mode. For each sample for which either 'INDEX' or 'COPY_ABOVE' mode is possible, the run for each mode is determined. Then, the cost of coding the mode, the run and possibly the index value (for 'INDEX' mode) is calculated. The mode for which the cost is lower is selected.

3.3.1.3 Coding of the palette entries

For coding of the palette entries, a palette predictor is maintained [64]. For each entry in the palette predictor, a reuse flag is signalled to indicate whether it is part of the current palette. This is illustrated in Figure 3.11. The reuse flags are sent using run-length coding of zeros. After this, the number of new palette entries are signalled using exponential Golomb code of order 0. Finally, the component values for the new palette entries are signalled.

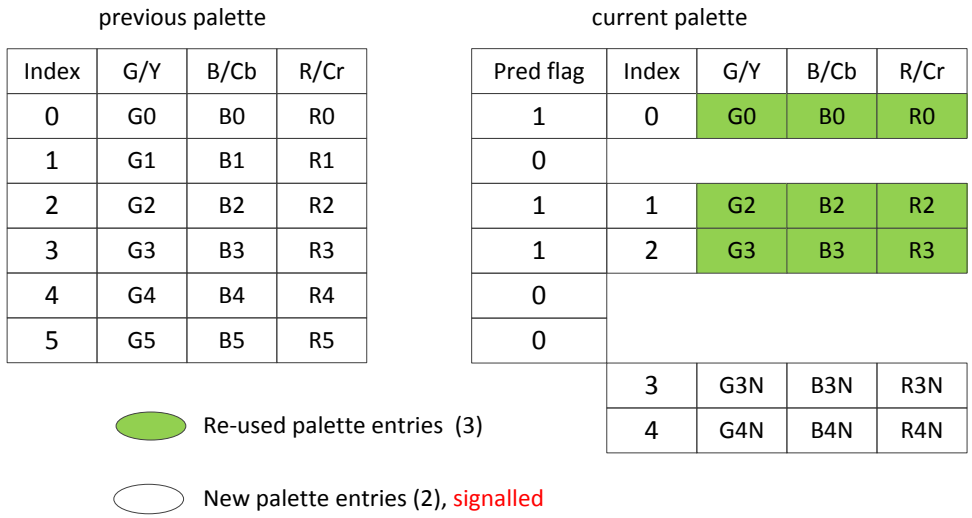


Figure 3.11: Use of palette predictor to signal palette entries [64]

3.3.1.4 Coding of palette indices

The palette indices are coded using horizontal and vertical traverse scans as shown in Figure 3.12. Horizontal scan is assumed for the following example. The palette indices are coded using two main palette sample modes: 'INDEX' and 'COPY_ABOVE'. As explained previously, the escape symbol is also signalled as an 'INDEX' mode and assigned an index equal to the maximum palette size. The mode is signalled using a flag except for the top row or when the previous mode was 'COPY_ABOVE'.

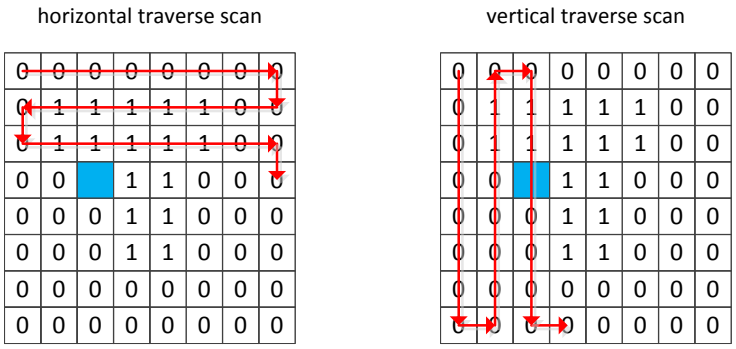


Figure 3.12: Horizontal and vertical traverse scans

In the 'COPY_ABOVE' mode, the palette index of the sample in the row above is copied. In the 'INDEX' mode, the palette index is explicitly signalled. For both 'INDEX' and 'COPY_ABOVE' modes, a run value is signalled which specifies the number of subsequent samples that are also coded using the same mode. When escape symbol is part of the run in 'INDEX' or 'COPY_ABOVE' mode, the escape component values are signalled for each escape symbol [58] [64]. The coding of palette indices is illustrated in Figure 3.13.

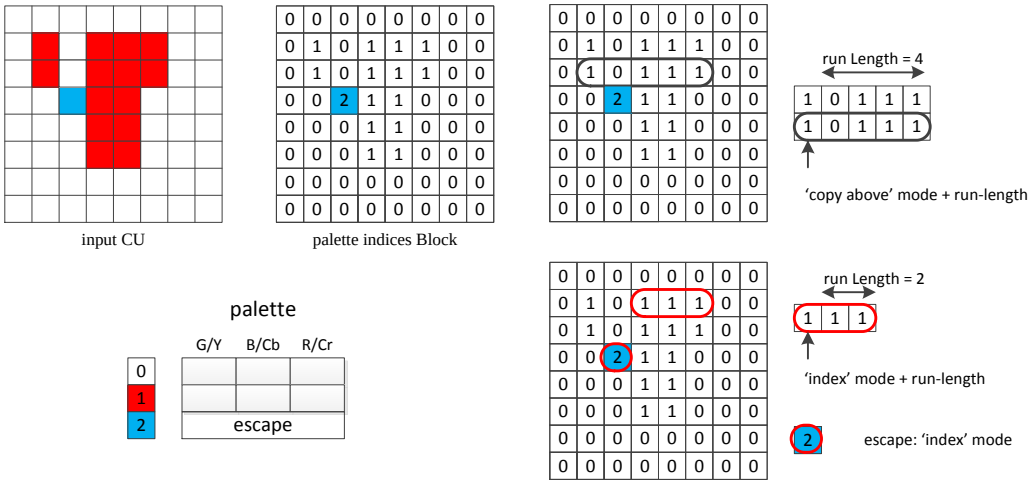


Figure 3.13: Coding of palette indices [64]

3.4 Adaptive Color Transform

Much screen content is captured in the RGB color space. To remove inter color component redundancy, color space conversion is useful For an image block in RGB color space, usually there can be strong correlation among different color components.

For image blocks exhibiting less correlation among color components, coding in RGB space is more effective [58] [64] [63].

The Adaptive Color Transform (ACT) was adopted into the HEVC SCC test model 2 at the 18th JCT-VC meeting. ACT performs in-loop color space conversion in the prediction residual domain using color transform matrices based on the YC_oC_g and YC_oC_g-R color spaces. ACT is operated at level. ACT can be combined with Cross Component Prediction (CCP). When both

are enabled, ACT is performed after CCP at the decoder, as shown in Figure 3.14. In SCM 5, the signaling of ACT is moved from the CU level to the TU level to align it with CCP.

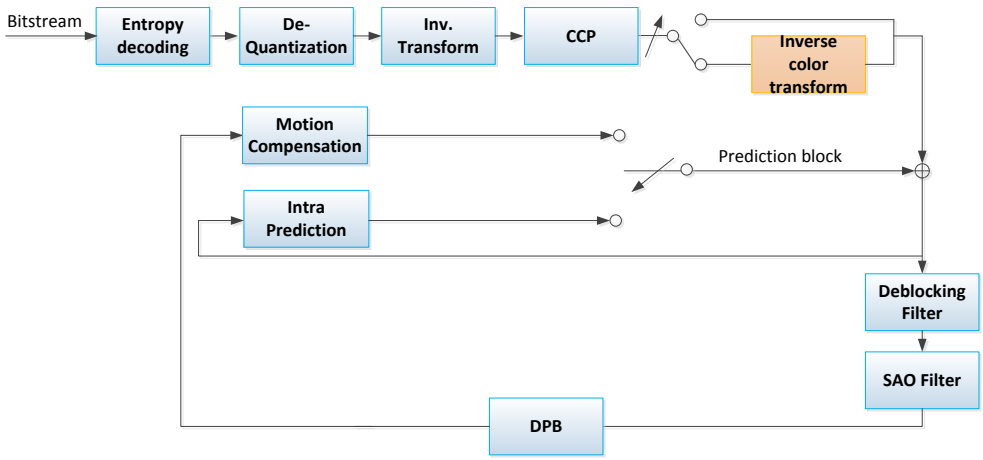


Figure 3.14: SCC decoder flow of in-loop ACT [64] [63]

3.4.1 Color space conversion in ACT

The color space conversion in ACT is based on the YC_oC_g-R transform [64]. Both lossy coding and lossless coding use the same inverse transform, but an additional 1-bit left shift is applied to the C_o and C_g components in the case of lossy coding. Specifically, the following colour space transforms are used for forward and backward conversion for lossy and lossless coding:

Forward transform for lossy coding (non-normative):

$$\begin{bmatrix} Y \\ Co \\ Cg \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 0 & -2 \\ -1 & 2 & -1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} / 4$$

Forward transform for lossless coding (non-normative):

$$\begin{aligned} Co &= R - B \\ t &= B + (Co >> 1) \\ Cg &= (G - t) \\ Y &= t + (Cg >> 1) \end{aligned}$$

Backward transform (normative):

$$
\begin{aligned}
&if\,(lossy)\\
&\{\\
&\quad Co = Co << 1\\
&\quad Cg = Cg << 1\\
&\}\\
t &= Y - (Cg >> 1)\\
G &= Cg + t\\
B &= t - (Co >> 1)\\
R &= Co + b
\end{aligned}
$$

When the input bit-depths of the color components are different, appropriate left shifts are applied to align the sample bit-depths to the maximal bit-depth during ACT, and appropriate right shifts are applied to restore the original sample bit-depths after ACT.

3.4.2 Encoder optimization for ACT

Care is taken on the encoder side when performing ACT, in order to avoid doubling the encoder complexity by searching over all the possible modes twice - in both the original colour space and the converted colour space [64]. Many methods are implemented for this purpose. One such method is as follows.

The chroma lambda adjustment method is used to reduce encoder complexity. Specifically, the chroma lambda used to calculate RD cost is increased compared to that for the luma component. The chroma lambda value is modified based on the input QP, using the following equation:

$$\lambda_{chroma} = \frac{\lambda_{luma}}{W_{QP}}$$

where $W_{QP} = 2^{delta(QP)/3}$ and Table 3.2 specifies the mapping between QP and delta(QP).

Table 3.2: Specification of delta(QP) used in chroma lambda adjustment for ACT [64]

QP	[0, 14]	[15, 29]	[30, 36]	[37, 38]	[39, 40]	[41, 42]	[43, 52]
delta(QP)	0	-1	-2	-3	-4	-5	-6

3.5 Adaptive motion vector resolution

Since screen content has a granularity of one or more samples, it is not necessary to use fractional motion compensation. In HEVC-SCC, a slice-level control is enabled to switch the motion vectors between full-pel and fractional resolutions. Savings in bit-rate can be achieved by not signalling the fractional portion of the motion vectors. In HEVC-SCC, adaptive motion vector resolution (AMVR) [44] defines a slice-level flag to indicate that the current slice uses integer (full-pel) motion vectors for luma samples [58] [64].

Adaptive MV resolution allows the MVs of an entire picture to be signalled in either quarter-pel precision (same as HEVC version 1) or integer-pel precision. Hash based motion statistics are kept and checked in order to properly decide the appropriate MV resolution for the current picture without relying on multi-pass encoding.

To decide the MV precision of one picture, the encoder performs the following check with the help of hashes. For every non-overlapped 8×8 block in a picture, the encoder checks whether it can find a matching block by hash in the first reference picture in list 0. The blocks are classified into the following categories:

- C: number of blocks matching with collocated block.
- S: number of blocks not matching with collocated block but belong to smooth region. For smooth region, it means every column has a single pixel value or every row has a single pixel value.
- M: number of blocks not belonging to C or S but can find a matching block by hash value.

T is the total number of blocks in one picture.

CSMRate = (C+S+M)/T,

MRate = M/T.

AverageCSMRate is the average CSMRate of current picture and the previous 31 pictures. AverageMRate is the average MRage of the current picture and the previous 31 pictures.

The MV resolution is determined as:

- If CSMRate < 0.8, use quarter-pel MV.
- Otherwise, if C == T, use integer-pel MV.
- Otherwise, if AverageCSMRate < 0.95, use quarter-pel MV.
- Otherwise, if M > (T−C−S)/3, use integer-pel MV.
- Otherwise, if CSMRate > 0.99 and MRate > 0.01, use integer-pel MV.
- Otherwise, if AverageCSMRate + AverageMRate > 1.01, use integer-pel MV.
- Otherwise, use quarter-pel MV.

3.6 Summary

In this chapter, Screen content tools are explained. In Chapter 4, implementation and results are discussed. Results with respect to bitrate savings and encoding time are discussed.

Chapter 4

Results

There are few performance comparison methods, among them SSIM, BD-PSNR [14] and BD-Bitrate [14] are the mostly used.

Bjøntegaard-Delta [45] Bit-Rate Measurements As rate-distortion (R-D) performance assessment, Bjøntegaard-Delta bit-rate (BD-BR) measurement method is used for calculating average bit-rate differences between R-D curves for the same objective quality (e.g., for the same PSNRYUV values), where negative BD-BR values indicate actual bit-rate savings. As part of this thesis BD-BR performance metric will be used to determine average bit-rate savings for lossy coding. For lossless coding, average bit-saving percentages are listed.

Simulations were conducted to evaluate the new coding tools in HEVC-SCC and to compare the coding efficiency of HEVC-SCC with HEVC-RExt and H.264/AVC. Test models SCM5.2, HM16.6 and JM19.0 are used for HEVC-SCC, HEVC-RExt and H.264/AVC, respectively.

The common test conditions are used to generate the results. Use of different kind of test sequences have been made like text and graphics with motion and mixed content. All these sequences are of 4:4:4 YUV formats.

For lossless coding, QP value is set to 0, and for lossy coding, four QPs (22, 27, 32, 37) are applied. All the simulations are carried out in All Intra (AI) mode.

Configuration:

- SCM5.2 (HEVC+SCC) : encoder_intra_main_scc (Lossless- *CostMode:lossless, along with several other parameters*)
- HM16.6 (HEVC+RExt): encoder_intra_main_rext
- JM19.0 (H.264/AVC): HM-like (encoder_JM_Intra_HE)

The following set of tables and plots show the comparison of SCM5.2 with and without SCC coding tools. The tables give details of Bitrates and Encoding times for each sequence and plots give the bitrate savings in percentages.

Table 4.1 Comparison of Anchor(HM16.6+SCM5.2) with versus without IBC

Comparison of Anchor(HM16.6+SCM5.2) with versus without IBC					
		With IBC		**Without IBC**	
Index	**Sequence**	**Bitrate (Kbps)**	**Encoding time(sec)**	**Bitrate (Kbps)**	**Encoding time(sec)**
1	twist_tunnel	21141.168	297.813	22262.208	130.872
2	web_browsing	25369.008	159.838	30190.320	118.527
3	video_conferencing	21812.904	211.915	30855.000	143.667
4	ppt_doc_xls	21123.584	374.528	29417.440	282.681
5	pcb_layout	8913.472	463.419	11210.288	252.147

Table 4.2 Comparison of Anchor(HM16.6+SCM5.2) with versus without PM

Comparison of Anchor(HM16.6+SCM5.2) with versus without PM					
		With PM		**Without PM**	
Index	**Sequence**	**Bitrate (Kbps)**	**Encoding time(sec)**	**Bitrate (Kbps)**	**Encoding time(sec)**
1	twist_tunnel	21141.168	297.813	27259.680	275.196
2	web_browsing	25369.008	159.838	29067.504	131.189
3	video_conferencing	21812.904	211.915	26024.208	177.219
4	ppt_doc_xls	21123.584	374.528	35381.264	356.648
5	pcb_layout	8913.472	463.419	20737.184	446.797

Table 4.3 Comparison of Anchor(HM16.6+SCM5.2) with versus without ACT

Comparison of Anchor(HM16.6+SCM5.2) with versus without ACT					
		With PM		**Without ACT**	
Index	**Sequence**	**Bitrate (Kbps)**	**Encoding time(sec)**	**Bitrate (Kbps)**	**Encoding time(sec)**
1	twist_tunnel	21141.168	297.813	21141.696	268.170
2	web_browsing	25369.008	159.838	25366.872	121.756
3	video_conferencing	21812.904	211.915	21819.288	157.313
4	ppt_doc_xls	21123.584	374.528	21131.680	299.582
5	pcb_layout	8913.472	463.419	8914.032	392.633

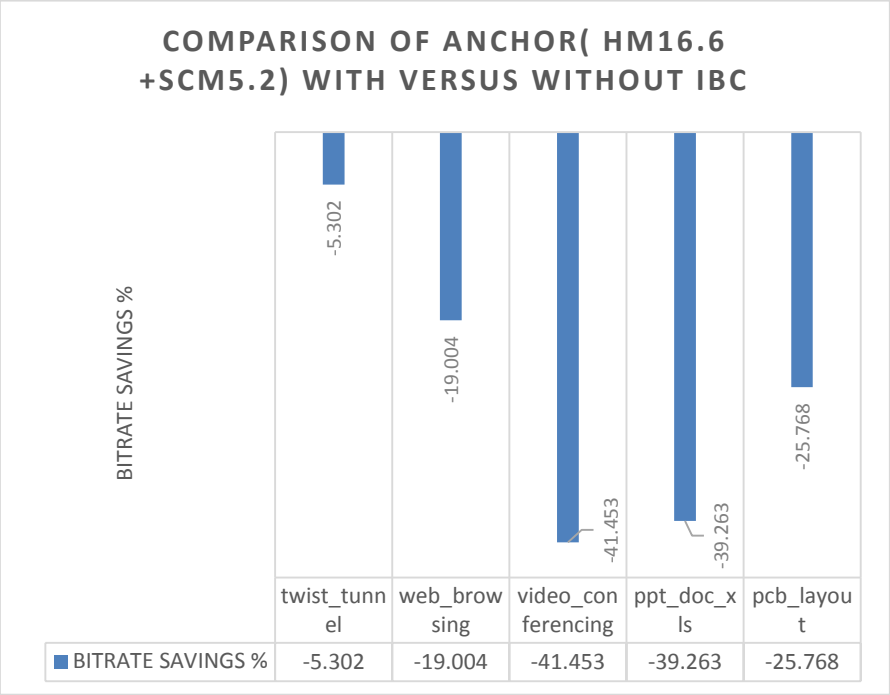


Figure 4.1 Comparison of Anchor (HM16.6+SCM5.2) with versus without IBC

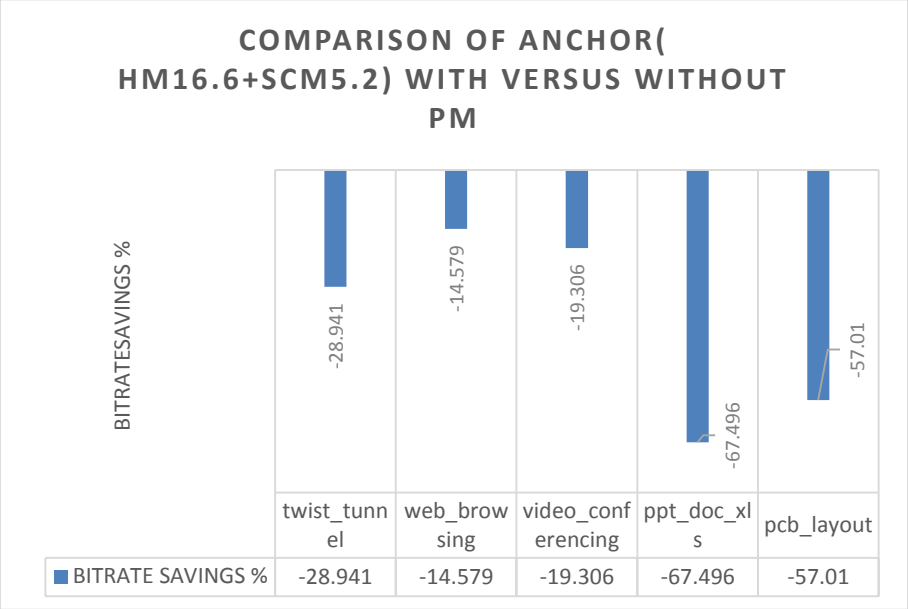


Figure 4.2 Comparison of Anchor (HM16.6+SCM5.2) with versus without PM

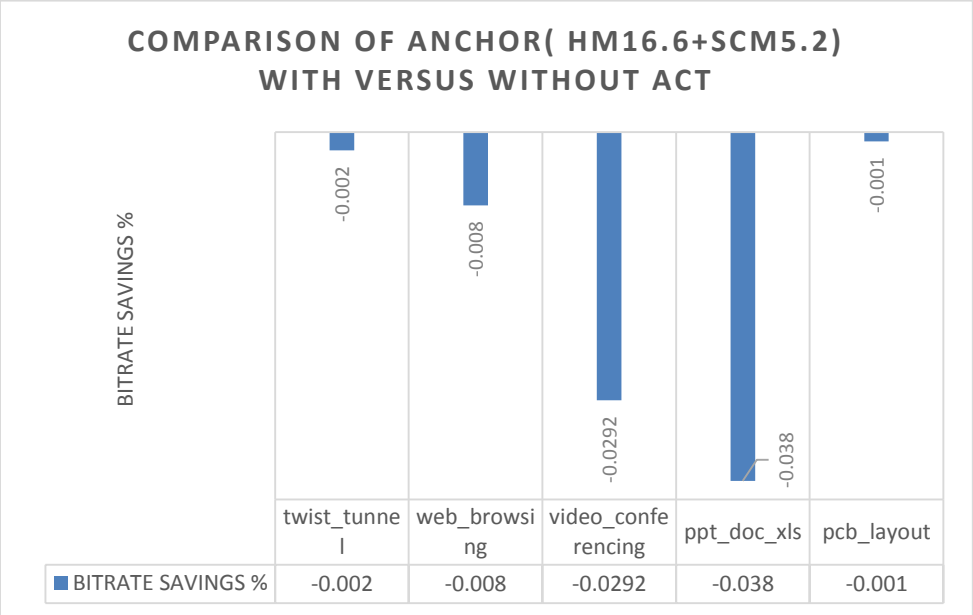


Figure 4.3 Comparison of Anchor (HM16.6+SCM5.2) with versus without ACT

Following tables and graphs show the coding efficiency comparison between HEVC-SCC, HEVC-RExt and H.264/AVC for lossless coding.

Table 4.4 Comparison of Anchor(HM16.6+SCM5.2) with HM16.6+RExt

Comparison of Anchor(HM16.6+SCM5.2) with HM16.6+RExt (Lossless)					
		HM16.6+SCM5.2		**HM16.6+RExt**	
Index	**Sequence**	**Bitrate (Kbps)**	**Encoding time(sec)**	**Bitrate (Kbps)**	**Encoding time(sec)**
1	twist_tunnel	21141.168	297.813	38597.688	80.817
2	web_browsing	25369.008	159.838	56437.344	71.813
3	video_conferencing	21812.904	211.915	66867.896	85.798
4	ppt_doc_xls	21123.584	374.528	76018.768	178.431
5	pcb_layout	8913.472	463.419	53289.296	176.725

Table 4.5 Comparison of Anchor(HM16.6+SCM5.2) with JM19.0

Comparison of Anchor(HM16.6+SCM5.2) with JM19.0 (Lossless)					
		HM16.6+SCM5.2		**JM19.0**	
Index	**Sequence**	**Bitrate (Kbps)**	**Encoding time(sec)**	**Bitrate (Kbps)**	**Encoding time(sec)**
1	twist_tunnel	21141.168	297.813	27560.54	506.297
2	web_browsing	25369.008	159.838	41784.98	309.806
3	video_conferencing	21812.904	211.915	55478.09	515.306
4	ppt_doc_xls	21123.584	374.528	73574.45	897.944
5	pcb_layout	8913.472	463.419	68800.21	923.338

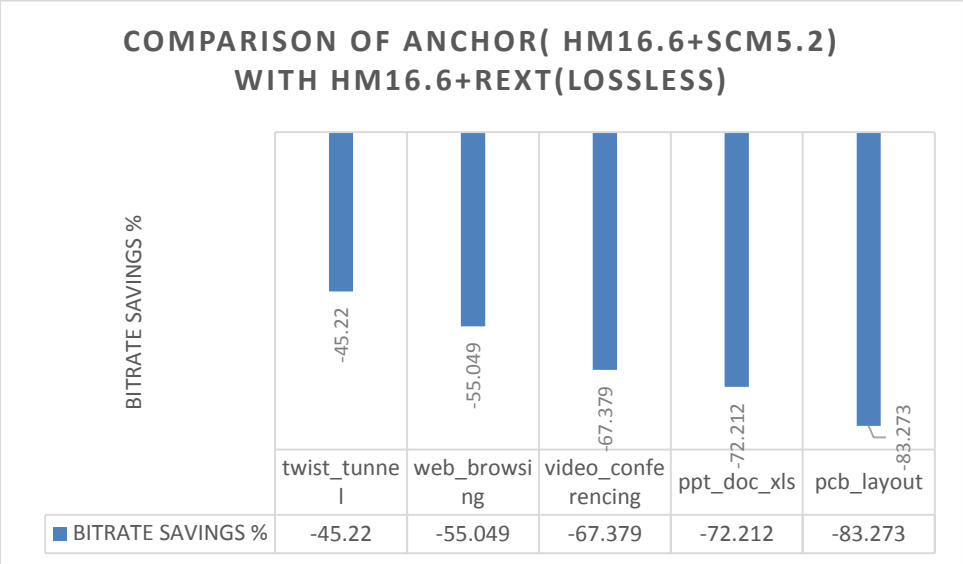


Figure 4.4 Comparison of Anchor (HM16.6+SCM5.2) with HM16.6+RExt

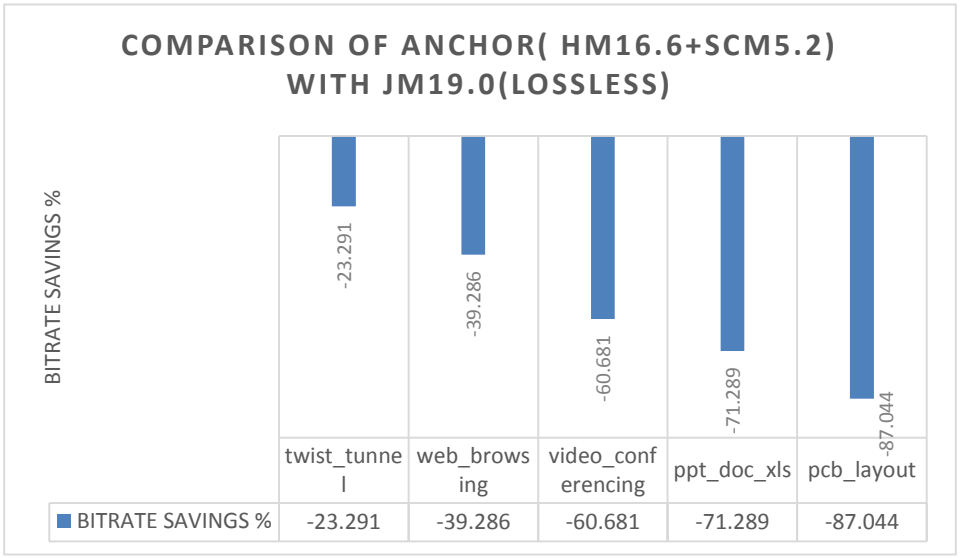


Figure 4.5 Comparison of Anchor (HM16.6+SCM5.2) with JM19.0

Following tables and graphs show the coding efficiency comparison between HEVC-SCC, HEVC-RExt and H.264/AVC for lossy coding. PSNR and bitrate values are recorded for different quantization parameters (22, 27, 32, and 37) for each test sequence.

Table 4.6 Comparison between SCM5.2 and JM19.0 for Twist_tunnel (Lossy)

Twist_tunnel							
		HM16.6+SCM5.2			JM19.0		
Index	**QP**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**
1	22	8567.736	56.9436	511.004	18917.81	47.211	316.856
2	27	7545.720	53.3078	505.226	15452.86	44.738	306.458
3	32	6712.224	48.8198	476.678	11968.80	41.513	287.123
4	37	5786.496	43.4694	484.450	8624.57	37.987	260.545

Table 4.7 Comparison between SCM5.2 and JM19.0 for Web_Browsing (Lossy)

Web_Browsing							
		HM16.6+SCM5.2			JM19.0		
Index	**QP**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**
1	22	7622.712	54.3126	455.631	22198.22	46.623	294.643
2	27	5801.016	49.9936	375.052	15485.30	42.786	262.465
3	32	4245.936	45.2138	337.278	9637.70	38.781	254.510
4	37	2980.344	39.9377	309.421	5029.34	34.780	229.126

Table 4.8 Comparison between SCM5.2 and JM19.0 for Video_Conferencing (Lossy)

Video_Conferencing_Doc_Sharing							
		HM16.6+SCM5.2			JM19.0		
Index	**QP**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**
1	22	8796.936	53.4509	588.268	35412.82	44.004	442.983
2	27	7004.016	49.1352	484.527	26628.00	40.355	380.440
3	32	5655.696	44.2382	443.503	19105.10	36.312	340.334
4	37	4461.624	38.6441	469.755	12438.43	32.444	291.709

Table 4.9 Comparison between SCM5.2 and JM19.0 for Ppt_Doc_Xls (Lossy)

Ppt_Doc_Xls							
		HM16.6+SCM5.2			JM19.0		
Index	**QP**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**
1	22	13557.584	56.2033	977.174	46315.17	42.669	855.594
2	27	11831.456	51.2814	916.877	32254.93	39.438	805.984
3	32	10214.784	46.3903	837.537	22427.65	36.561	690.695
4	37	8262.464	39.7284	795.522	13601.17	33.141	596.128

Table 4.10 Comparison between SCM5.2 and JM19.0 for Pcb_Layout (Lossy)

Pcb_Layout							
		HM16.6+SCM5.2			JM19.0		
Index	QP	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)
1	22	8251.984	63.2813	1179.702	59059.81	42.482	875.574
2	27	8016.336	58.2741	1513.004	43996.13	38.947	848.348
3	32	7689.264	52.3922	976.161	31694.61	35.914	726.792
4	37	7296.688	46.5995	999.931	20959.17	32.281	640.624

Table 4.11 Comparison between SCM5.2 and HM16.6+RExt for Twist_tunnel (Lossy)

Twist_tunnel							
		HM16.6+SCM5.2			HM16.6+RExt		
Index	QP	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)
1	22	8567.736	56.9436	511.004	18516.456	53.3375	149.975
2	27	7545.720	53.3078	505.226	16009.584	49.1012	147.024
3	32	6712.224	48.8198	476.678	13599.528	44.3843	139.143
4	37	5786.496	43.4694	484.450	10785.288	39.0826	135.999

Table 4.12 Comparison between SCM5.2 and HM16.6+RExt for Web_Browsing (Lossy)

Web_Browsing							
		HM16.6+SCM5.2			HM16.6+RExt		
Index	QP	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)
1	22	7622.712	54.3126	455.631	27339.120	50.4953	157.064
2	27	5801.016	49.9936	375.052	21129.768	46.0047	140.270
3	32	4245.936	45.2138	337.278	15280.080	41.6069	132.152
4	37	2980.344	39.9377	309.421	8757.168	36.3275	122.594

Table 4.13 Comparison between SCM5.2 and HM16.6+RExt for Video_Conferen (Lossy)

Video_Conferencing_Doc_Sharing							
		HM16.6+SCM5.2			HM16.6+RExt		
Index	QP	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)	Bitrate (Kbps)	PSNR (dB)	Encoding Time(secs)
1	22	8796.936	53.4509	588.268	38544.480	48.6653	187.387
2	27	7004.016	49.1352	484.527	31461.336	44.2769	174.516
3	32	5655.696	44.2382	443.503	24552.336	39.6574	164.190
4	37	4461.624	38.6441	469.755	17325.960	34.8581	152.045

Table 4.14 Comparison between SCM5.2 and HM16.6+RExt for Ppt_Doc_Xls (Lossy)

Ppt_Doc_Xls							
		HM16.6+SCM5.2			HM16.6+RExt		
Index	**QP**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**
1	22	13557.584	56.2033	977.174	45268.384	50.2732	384.820
2	27	11831.456	51.2814	916.877	37094.944	45.5902	358.798
3	32	10214.784	46.3903	837.537	28891.744	41.0092	328.764
4	37	8262.464	39.7284	795.522	19109.664	35.5834	302.745

Table 4.15 Comparison between SCM5.2 and HM16.6+RExt for Pcb_Layout (Lossy)

Pcb_Layout							
		HM16.6+SCM5.2			HM16.6+RExt		
Index	**QP**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**	**Bitrate (Kbps)**	**PSNR (dB)**	**Encoding Time(secs)**
1	22	8251.984	63.2813	1179.702	39692.960	52.1693	378.089
2	27	8016.336	58.2741	1513.004	34331.504	47.1667	365.530
3	32	7689.264	52.3922	976.161	29721.792	41.7744	347.144
4	37	7296.688	46.5995	999.931	23497.472	36.3990	322.186

Following is the list of R-D plots for each test sequence:

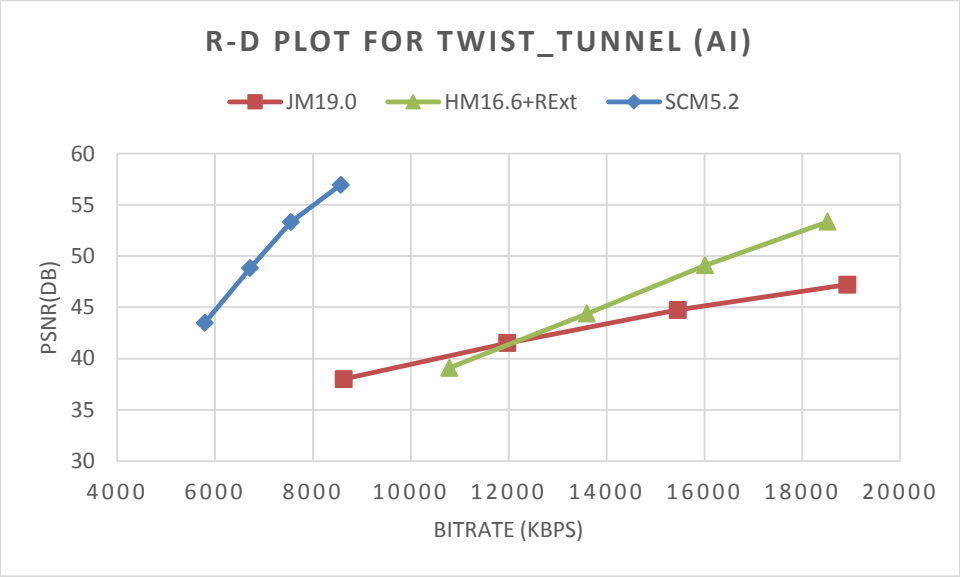


Figure 4.6 R-D Plot for Twist_tunnel (AI)

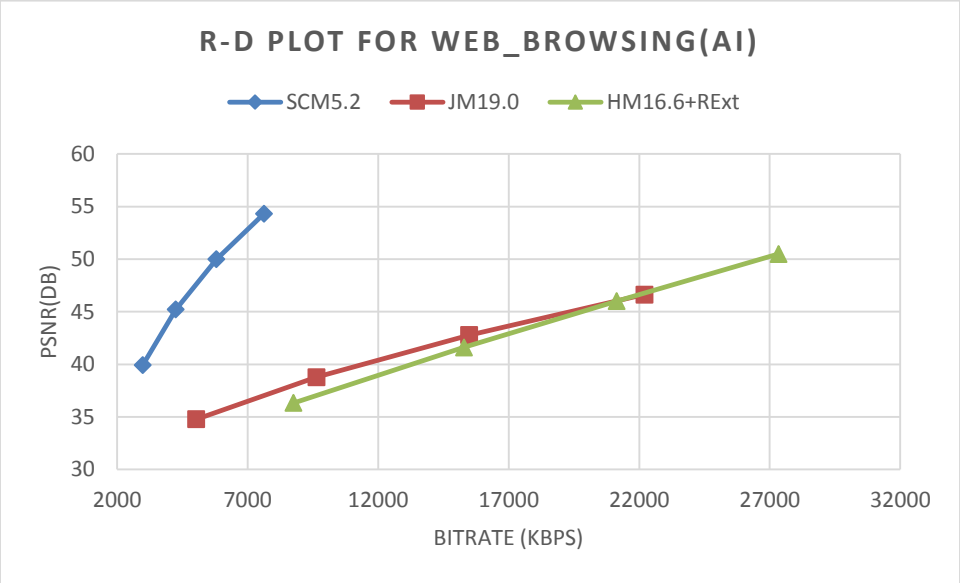


Figure 4.7 R-D Plot for Web_Browsing(AI)

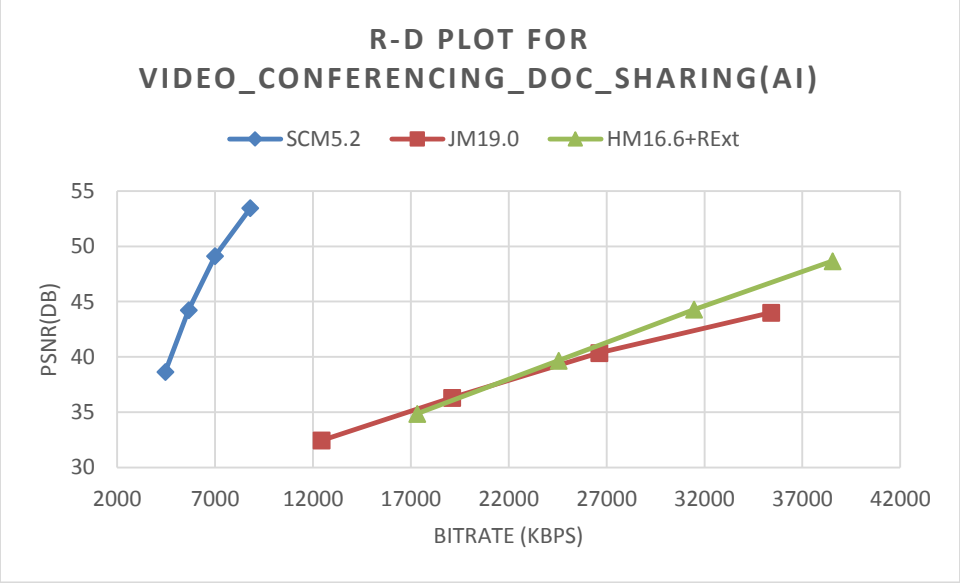


Figure 4.8 R-D Plot for Video_Conferencing_Doc_Sharing (AI)

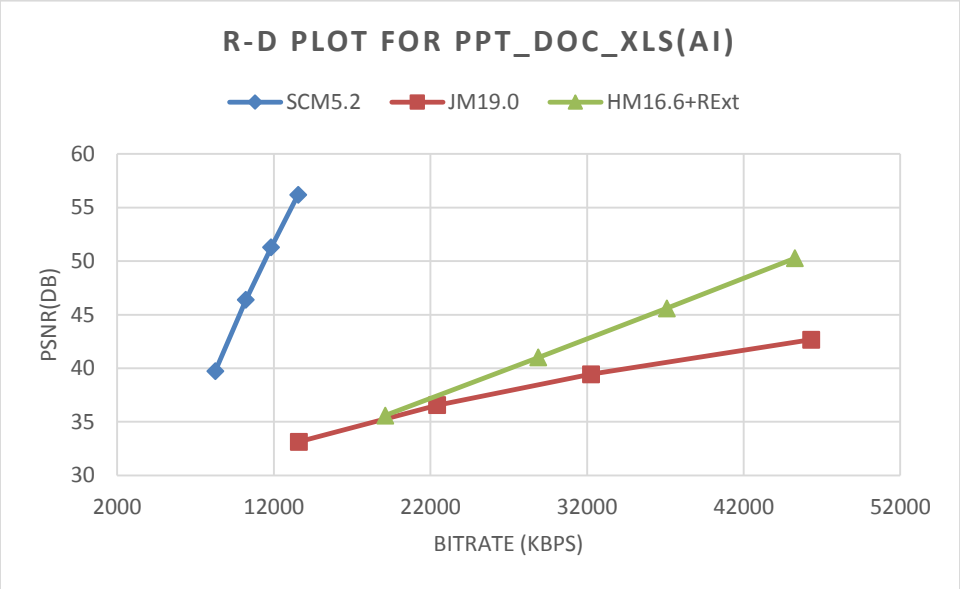


Figure 4.9 R-D Plot for Ppt_Doc_Xls (AI)

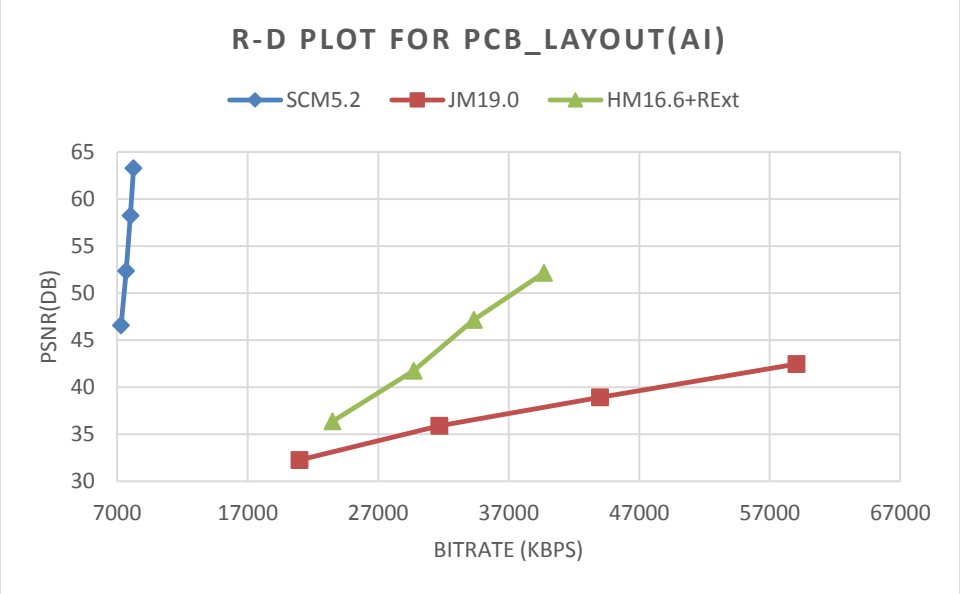


Figure 4.10 R-D Plot for Pcb_Layout (AI)

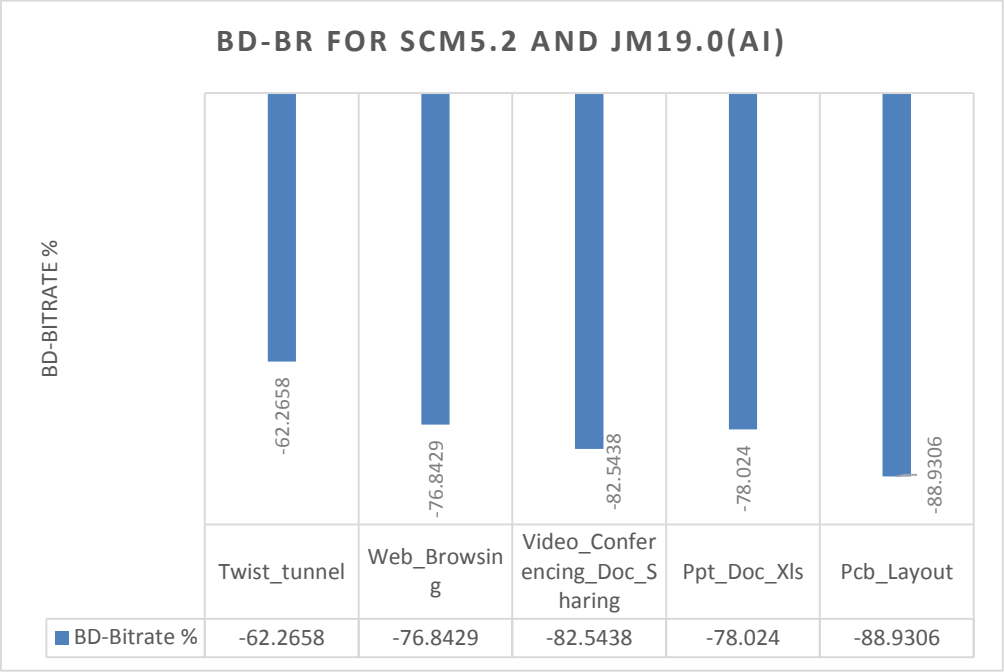


Figure 4.11 BD-BR for SCM5.2 and JM19.0 (AI)

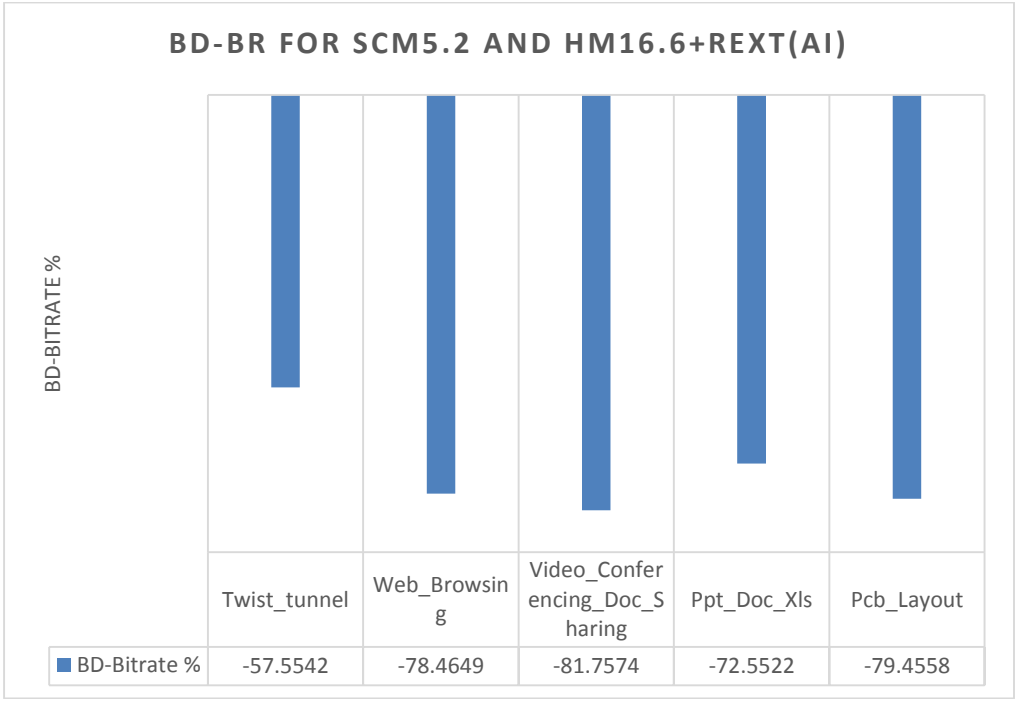
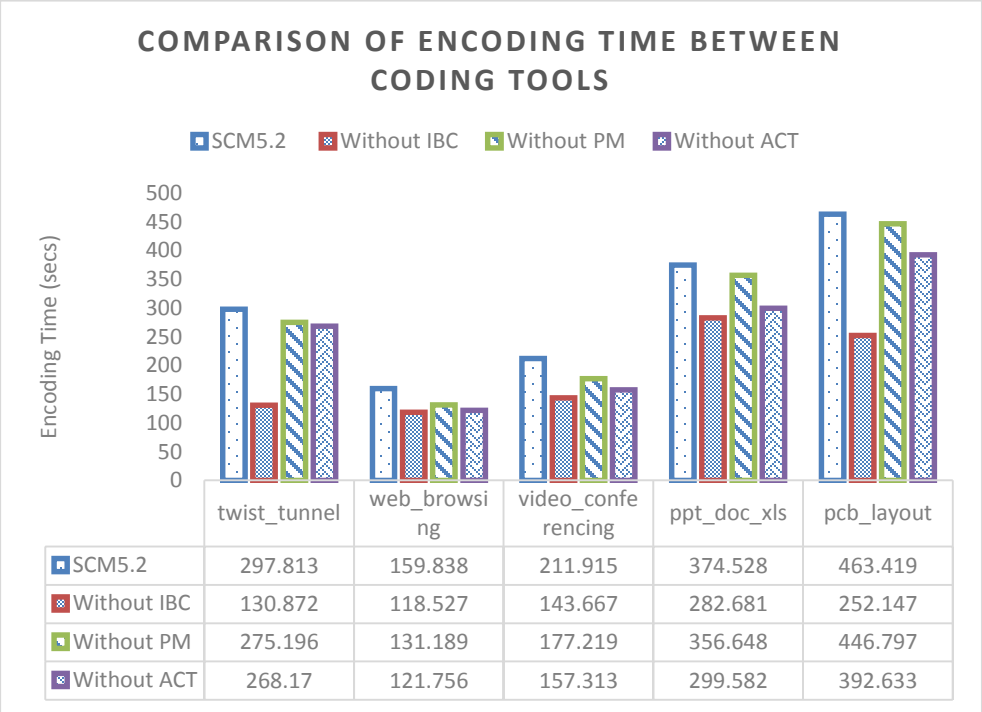
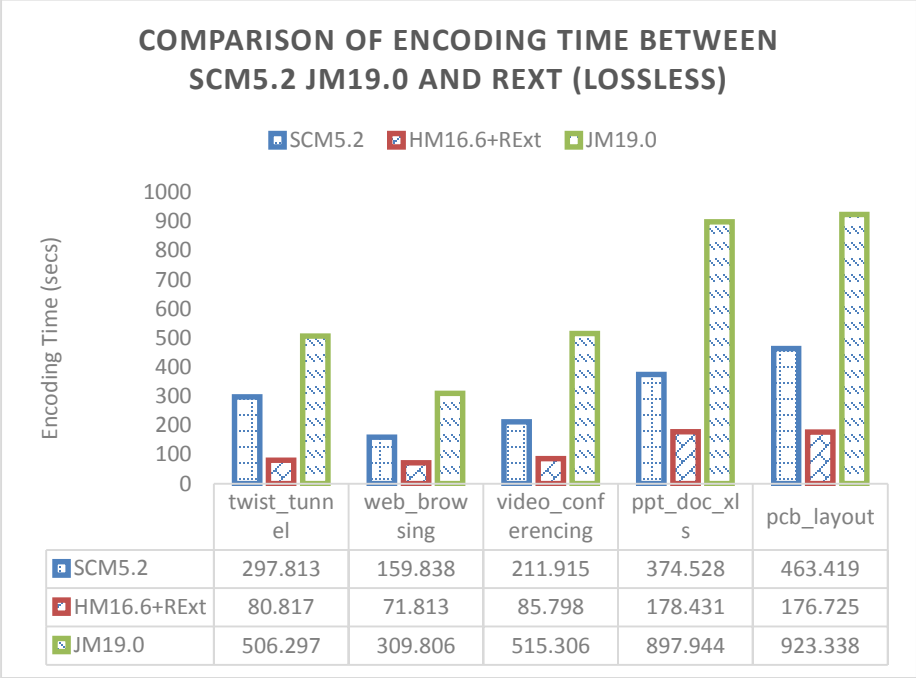


Figure 4.12 BD-BR for SCM5.2 and HM16.6+RExt (AI)



	twist_tunnel	web_browsing	video_conferencing	ppt_doc_xls	pcb_layout
SCM5.2	297.813	159.838	211.915	374.528	463.419
Without IBC	130.872	118.527	143.667	282.681	252.147
Without PM	275.196	131.189	177.219	356.648	446.797
Without ACT	268.17	121.756	157.313	299.582	392.633

Figure 4.13 Comparison of Encoding time between coding tools



	twist_tunnel	web_browsing	video_conferencing	ppt_doc_xls	pcb_layout
SCM5.2	297.813	159.838	211.915	374.528	463.419
HM16.6+RExt	80.817	71.813	85.798	178.431	176.725
JM19.0	506.297	309.806	515.306	897.944	923.338

Figure 4.14 Comparison of Encoding time between SCM5.2 JM19.0 and RExT (lossless)

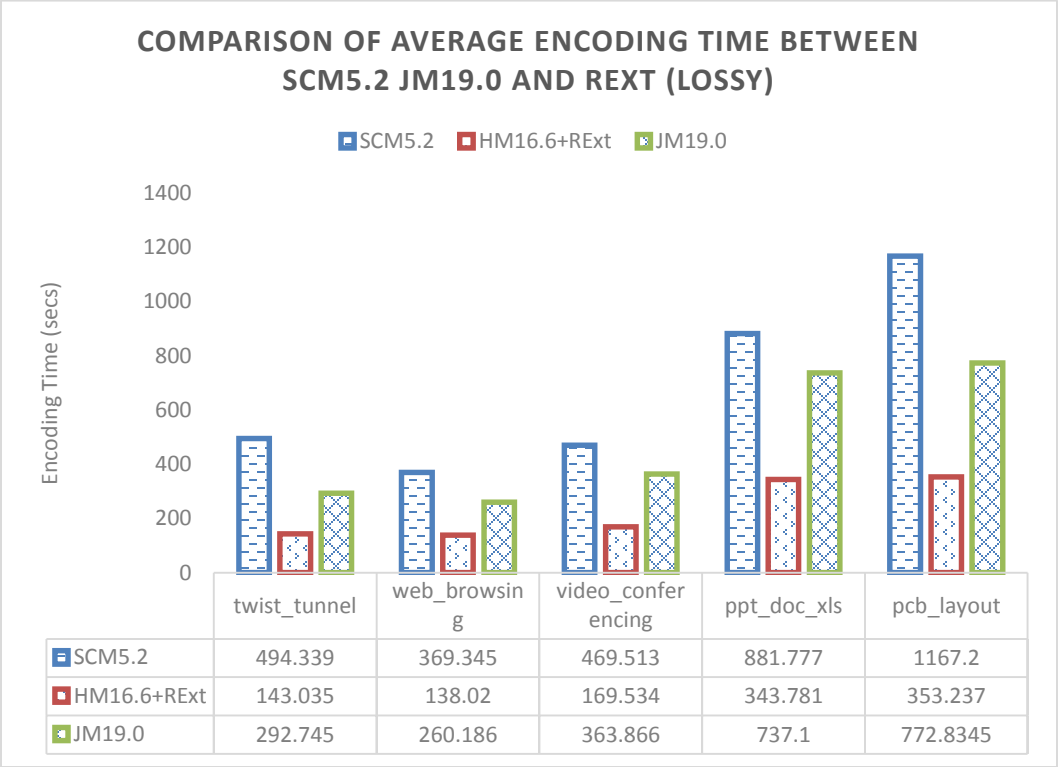


Figure 4.15 Comparison of average Encoding time between SCM5.2 JM19.0 and RExT (lossy)

4.1 Summary

Simulations for SCM5.2 with and without SCC coding tools are run and bitrate savings are calculated and analysed. Also, simulations for HM16.6 (with RExt) and JM19.0 are run and coding efficiency is compared with SCM5.2. In chapter 5, conclusions and future works are discussed.

Chapter 5

Conclusions and Future Work

It can be seen from the Chapter 4 results that SCM with IBC gives bitrate savings from 5%-45%, SCM with PM gives 14%-67% and SCM with ACT gives 0.001% to 0.0038 % compared to SCM without IBC, without PM and without ACT, respectively. Also, SCM is evaluated against JM19.0 and HEVC-RExt. It can be seen that SCM gives bitrate saving of about 45%-83% compared to HEVC+RExt under lossless condition and 23%-87% compared to JM19.0 (AVC) under lossless condition. Under lossy condition, SCM gives 57%-81% BD-bitrate savings compared to HEVC+RExt and 62%-88% BD-bitrate savings compared to JM19.0.

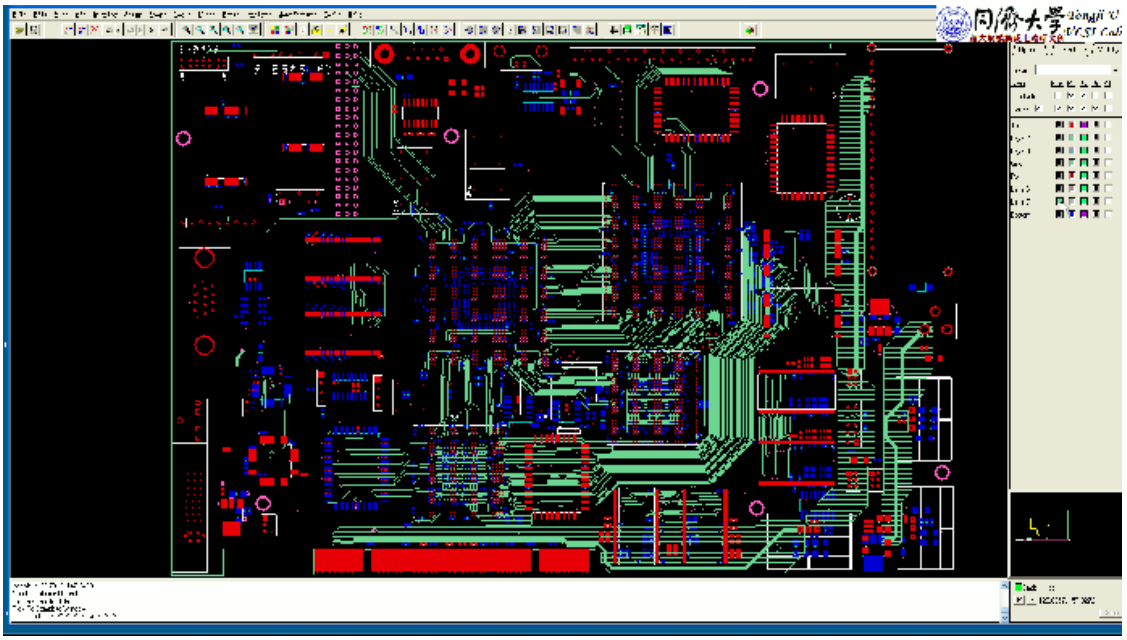
Although SCM reduces bitrate by a great extent, it is observed that it increases the encoding to a significant extent.

This can be good topic of interest for future work. The complexity can be reduced by parallelizing certain parts of the codec which consume more time. This can be done using tools like OpenMP and CUDA.

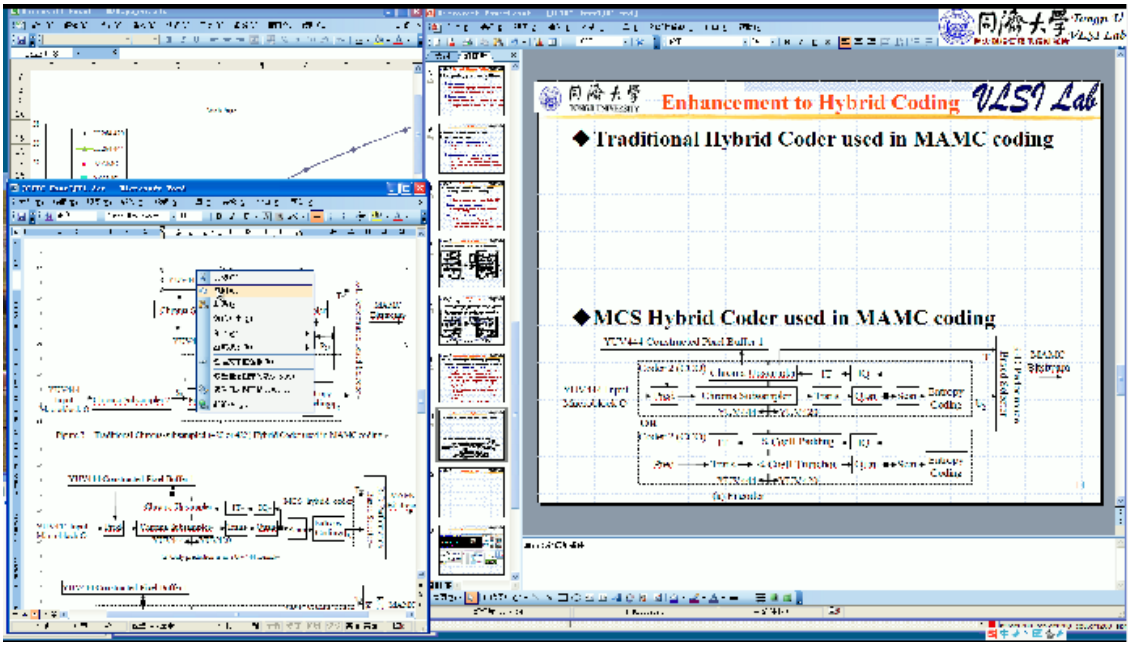
Appendix A

Test Sequences [70]

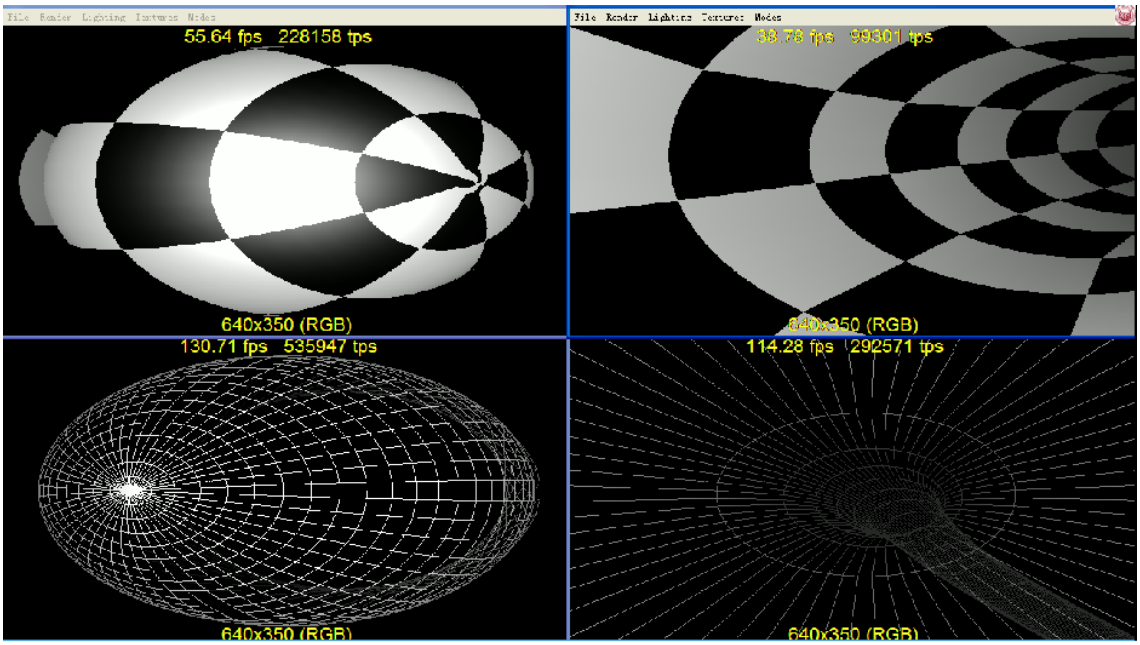
A1. PCB Layout (Resolution: 1920x1080)



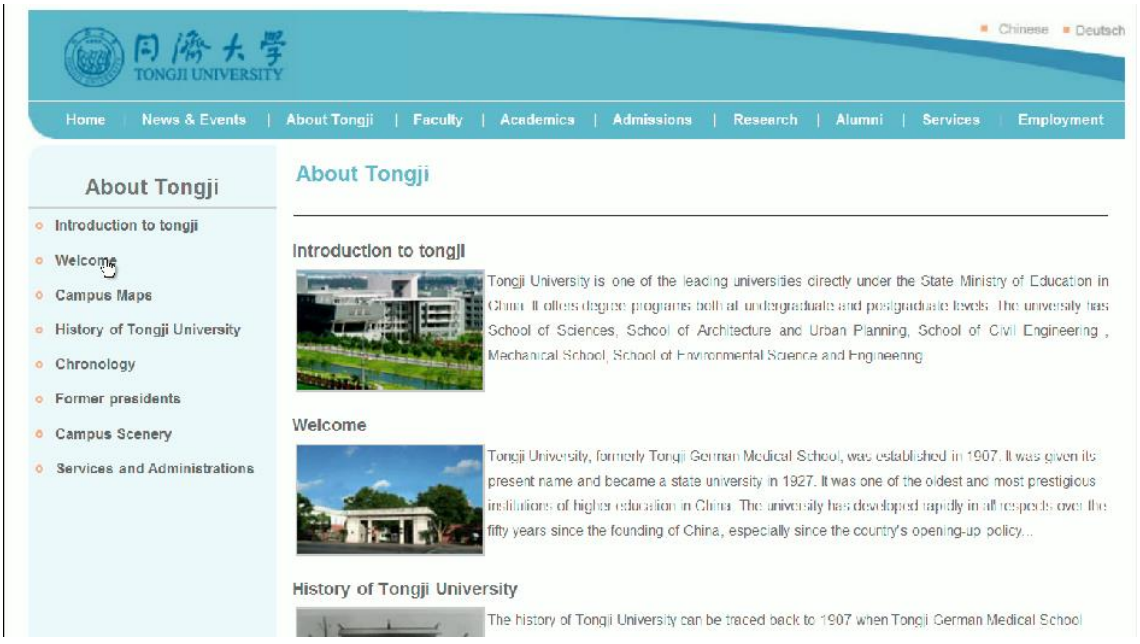
A2. PPT Document (Resolution: 1920x1080)



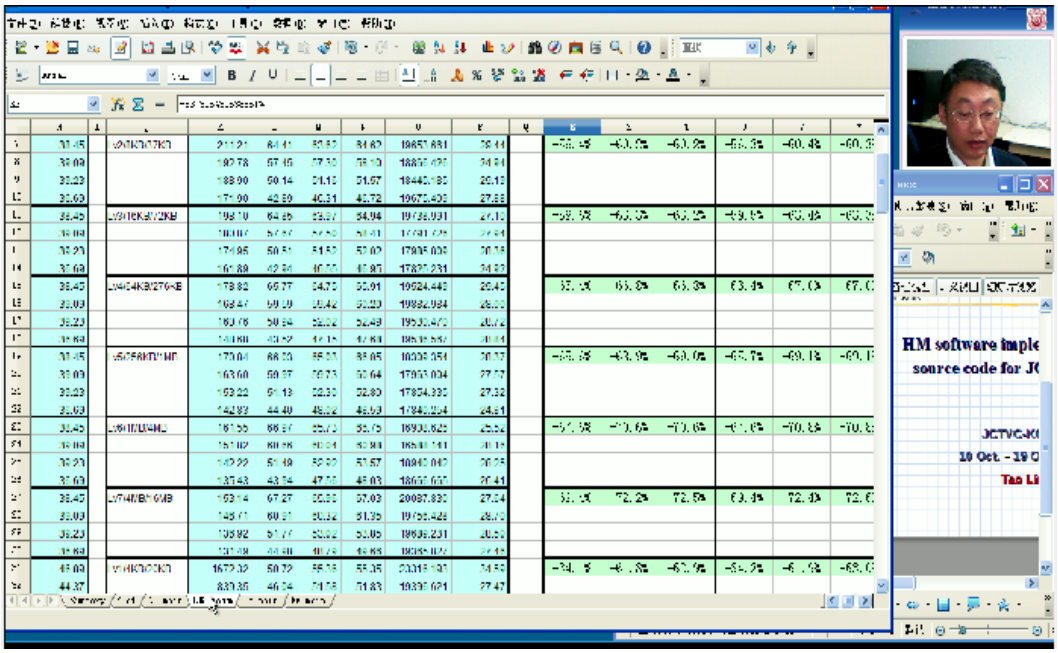
A3. CG Twist Tunnel - Animation (Resolution: 1280x720)



A4. Web Browsing (Resolution: 1280x720)



A5. Video Conferencing (Resolution: 1280x720)



Appendix B

Acronyms

ACT – Adaptive Color Transform

AMVR – Adaptive Motion Vector Resolution

AVC – Advanced Video Coding

BCIM – Base Color Index Map

BD BR – Bjontegaard Delta Bitrate

BM – Block Matching

BV – Block Vector

CRT – Cathode Ray Tube

CU- Coding unit

CTU- Coding tree unit

CCP – Cross Component Prediction

CABAC - Context adaptive binary arithmetic coding

DBF- Deblocking Filter

DPCM – Differential Pulse Code Modulation

DFT – Discrete Fourier Transform

DCT – Discrete Cosine Transform

DST – Discrete Sine Transform

DPB - Decoded Picture Buffer

DC – Direct Current

DictSCC - Dictionary coding for screen content

FPS- Frames Per Second

HD- High definition

HEVC-High Efficiency Video Coding

ITU-T - International Telecommunication Union (Telecommunication Standardization Sector)

IEC - International Electrotechnical Commission

ISO – International Standards Organization

IntraBC – Intra block copy

JPEG - Joint photographic experts group

JCT-VC- Joint collaborative team on video coding

MCP – Motion Compensated Prediction

MV – Motion Vector

MPEG-Moving picture experts group

NAL - Network Abstraction Layer

PU – Prediction Unit

QP- Quantization Parameter

RD – Rate Distortion

RExt – Range Extension

RGB – Red green Blue

RDPCM – Residual Differential Pulse Code Modulation

RSQ – Residual Scalar Quantization

SAD – Sum of Absolute Differences

SAO - Sample Adaptive Offset

SCC - Screen Content Coding

TSM – Transform skip Mode

TU-Transform units

UHDTV - Ultra-high-definition Television

VCEG – Video Coding Experts Group

VCL - Variable Code Length

WPP - Wavefront Parallel Processing

1D - 1 Dimensional

2D – 2 Dimensional

References

[1] G.J. Sullivan et al, “Overview of the high efficiency video coding (HEVC) standard”, IEEE Trans. circuits and systems for video technology, vol. 22, no.12, pp. 1649 – 1668, Dec 2012.

[2] G.J. Sullivan et al, "Standardized Extensions of High Efficiency Video Coding (HEVC)", IEEE Journal of Selected Topics in Signal Processing, vol.7, no.6, pp.1001-1016, Dec. 2013.

[3] T. Wiegand et al, “Overview of the H.264/AVC Video Coding Standard”, IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, No. 7, pp. 560-576, July. 2003.

[4] N. Ahmed , T. Natarajan and K.R. Rao, “Discrete Cosine Transform”, IEEE Trans. on Computers, Vol. C-23, pp. 90-93, Jan. 1974.

[5] I.E. Richardson, “The H.264 advanced video compression standard”, 2nd Edition, Hoboken, NJ, Wiley, 2010.

[6] I.E. Richardson, “Video Codec Design: Developing Image and Video Compression Systems”, Wiley, 2002.

[7] K.R. Rao, D.N. Kim and J.J. Hwang, “Video Coding Standards: AVS China, H.264/MPEG-4 Part 10, HEVC, VP6, DIRAC and VC-1”, Springer, 2014.

[8] Special issue on emerging research and standards in next generation video coding, IEEE Trans. on Circuits and Systems for Video Technology, vol.22, pp.1646-1909, Dec.2012.

[9] Introduction to the issue on video coding: HEVC and beyond, IEEE Journal of selected topics in signal processing, vol.7, pp.931-1151, Dec.2013.

[10] Special issue on Screen Content Video Coding and Applications, IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS), Final manuscripts due on 22nd July 2016.

[11] Article on HEVC - http://en.wikipedia.org/wiki/High_Efficiency_Video_Coding

[12] V.Sze, M.Budagavi and G.J. Sullivan, “High Efficiency Video Coding (HEVC), Algorithms and Architectures”, Springer, 2014

[13] HM Software Manual for version 15: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/branches/HM-15-dev/doc/software-manual.pdf

[14] R. Sjoberg et al, "Overview of HEVC High-Level Syntax and Reference Picture Management", IEEE Transactions on Circuits and Systems for Video Technology, vol.22, no.12, pp.1858-1870, Dec. 2012.

[15] M.P. Sharabayko et al, "Intra Compression Efficiency in VP9 and HEVC" Applied Mathematical Sciences, Vol. 7, no. 137, pp.6803 – 6824, Hikari Ltd, 2013

[16] HEVC tutorial by I.E. Richardson: http://www.vcodex.com/h265.html

[17]V. Sze and M. Budagavi, "High Throughput CABAC Entropy Coding in HEVC," IEEE Trans. on Circuits and Systems for Video Technology, vol.22, no.12, pp.1778-1791, Dec. 2012

[18] J. Boyce et al, "Draft high efficiency video coding (HEVC) version 2, combined format range extensions (RExt), scalability (SHVC), and multi-view (MV-HEVC) extensions", JCT-VC, Retrieved 2014-07-11.

[19] J. Chen et al, "HEVC Scalable Extensions (SHVC) Draft Text 7 (separated text)", JCT-VC, Retrieved 2014-07-13

[20] D.-K. Kwon and M.Budagavi, "Fast intra block copy (IntraBC) search for HEVC screen content coding", 2014 IEEE International Symposium on Circuits and Systems (ISCAS)",pp.9-12, June 2014

[21] Link to download JCT-VC documents: http://phenix.int-evry.fr/jct/

[22] Z. Ma, "Advanced Screen Content Coding Using Color Table and Index Map", IEEE Trans. on Image Processing, vol.23, no.10, pp.4399-4412, Oct. 2014

[23] D.K. Kwon and M. Budagavi, “RCE3: Results of Test 3.3 on Intra Motion Compensation, document”, JCTVC-N0205, Jul. 2013.

[24] J. Chen et al, “Description of Screen Content Coding Technology Proposal by Qualcomm, document”, JCTVC-Q0031, Apr. 2014.

[25] T.-S. Chang et al, RCE3: “Results of Subtest B.1 on Nx2N/2NxN Intra Block Copy, document”, JCTVC-P0176, Jan. 2014

[26] L. Guo et al, Non-RCE3: “Modified Palette Mode for Screen Content Coding, document”, JCTVC-N0249, Jul. 2013.

[27] B. Li et al, “Description of Screen Content Coding Technology Proposal” by Microsoft , JCTVC-Q0035, Apr. 2014.

[28] S.-L. Yu and C. Chrysafis, “New Intra Prediction Using Intra-Macroblock Motion Compensation, document”, JVT-C151, May 2002.

[29] S. Wang and T. Lin, “4:4:4 Screen Content Coding Using Macroblock-Adaptive Mixed Chroma-Sampling-Rate, document”, JCTVC-H0073, Feb. 2012.

[30] Wikipedia article on High Efficiency Video Coding tiers and levels: http://en.wikipedia.org/wiki/High_Efficiency_Video_Coding_tiers_and_levels#endnote_MaxDpbSizeC

[31] A. Zaccarin and B. Liu, “A novel approach for coding color quantized images,” IEEE Trans. on Image Processing, vol. 2, no. 4, pp. 442–453, Oct. 1993.

[32] W. Zhu et al, "2-D Dictionary Based Video Coding for Screen Contents," IEEE Data Compression Conference (DCC), 2014, pp.43-52, March 2014

[33] M. Mrak and J.-Z. Xu, "Improving screen content coding in HEVC by transform skipping", 2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO), pp.1209-1213, Aug. 2012

[34] D. Miao et al, "Layered screen video coding leveraging hardware video codec," 2013 IEEE International Conference on Multimedia and Expo (ICME), pp.1-6, July 2013

[35] G. Braeckman, "Lossy-to-lossless screen content coding using an HEVC base-layer", 2013 18th International Conference on Digital Signal Processing (DSP), pp.1-6, July 2013

[36] Access to HM 16.1 Reference Software: http://hevc.hhi.fraunhofer.de/

[37] Multimedia processing course website: http://www.uta.edu/faculty/krrao/dip/

[38] Link to download software manual for HEVC: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/

[39] Access to HM 15.0 Reference Software: http://hevc.hhi.fraunhofer.de/

[40] Visual studio: http://www.dreamspark.com

[41] Tortoise SVN: http://tortoisesvn.net/downloads.html

[42] Video Sequence Download Link: http://media.xiph.org/video/derf/

[43] Joint Collaborative Team on Video Coding Information we: http://www.itu.int/en/ITU-T/studygroups/2013-2016/16/Pages/video/jctvc.aspx

[44] Design and Implementation of Next Generation Video Coding Systems (H.265/HEVC Tutorial)-IEEE ISCAS Tutorial 2014: http://www.rle.mit.edu/eems/wp-content/uploads/2014/06/H.265-HEVC-Tutorial-2014-ISCAS.pdf

[45] G. Bjøntegaard, “Calculation of average PSNR differences between RD-curves”, ITU-TQ.6/SG16 VCEG 13th Meeting, Document VCEG-M33, Austin, USA, Apr. 2001.

[46] W. Mathias, “High Efficiency Video Coding Coding Tools and Specification”, Springer, 2015

[47] A.C. Bovik, “Handbook of image and video processing”, Elsevier Academic Press, 2005.

[48] J.Kang et al, “Explicit Residual DPCM for Screen Content Coding”, 2014, IEEE-ISCE, pp. 1-2, June 2014

[49] H. Yu et al, “Requirements for an extension of HEVC for coding of screen content,” ISO/IEC JTC 1/SC 29/WG 11 Requirements subgroup, San Jose, California, USA, document MPEG2014/N14174, Jan. 2014.

[50] C. Lan et al, “Screen content coding,” 2nd JCT-VC meeting, Geneva, Switzerland, document JCTVC-B084, Jul. 2010.

[51] R. Joshi, J. Sole, and M. Karczewicz, “AHG8: Residual DPCM for visually lossless coding,” 13th JCT-VC meeting, Incheon, Korea, document JCTVC-M0351, Apr. 2013.

[52] H. Yu et al, “Common test conditions for screen content coding,” 20th JCT-VC meeting, Geneva, Switzerland, document JCTVC-T1015, Feb. 2015.

[53] R. Joshi, et al, “Screen Content Coding Test Model 3 Encoder Description (SCM 3),” 19th JCT-VC meeting, Strasbourg, France, document JCTVC-S1014, Oct. 2014.

[54] B. Li and J. Xu, “Non-SCCE1: Unification of intra BC and inter modes,” 18th JCT-VC meeting, Sapporo, Japan, document JCTVC-R0100, Jul. 2014.

[55] B. Li et al, “Adaptive motion vector resolution for screen content,” 19th JCT-VC meeting, Strasbourg, France, document JCTVC-S0085, Oct. 2014.

[56] C. Pang et al, “CE2 Test1: Intra block copy and inter signalling unification,” 20th JCT-VC meeting, Geneva, Switzerland, document JCTVC-T0094, Feb. 2015.

[57] L. Guo, J. Sole, and M. Karczewicz, “Palette Mode for Screen Content Coding,” 13th JCT-VC meeting, Incheon, Korea, document JCTVC-M0323, Apr. 2013.

[58] J. Xu, R. Joshi and R. A. Cohen, “Overview of the Emerging HEVC Screen Content Coding Extension”, 2015 IEEE Trans. on Circuits and Systems for Video Technology, Sept.2015 (Early Access).

[59] W. Gao, et al, “Near lossless coding for screen content,” 6th JCT-VC meeting, Torino, Italy, document JCTVCF564, Jul. 2011.

[60] C. Lan, et al, “Intra and inter coding tools for screen contents,” 5th JCT-VC meeting, Geneva, Switzerland, document JCTVC-E145, Mar. 2011.

[61] S. Wang and T. Lin, “4:4:4 screen content coding using macro block adaptive mixed Chroma-sampling-rate,” 8th JCT-VC meeting, San Jose, California, USA, document JCTVC-H0073, Feb. 2012.

[62] T. Lin et al, “4:4:4 screen content coding using dual-coder mixed Chroma-sampling-rate (DMC) techniques,” 9th JCT-VC meeting, Geneva, Switzerland, document JCTVC-I0272, Apr. 2012.

[63] P. Lai, S. Liu, and S. Lei, “AHG6: On adaptive color transform (ACT) in SCM2.0,” 19th JCT-VC meeting, Strasbourg, France, document JCTVCS0100, Nov. 2014.

[64] R. Joshi et al, “Screen content coding test model 5 (SCM 5),” 21st JCT-VC meeting, Warsaw, PL, document JCTVC-U1014, Nov. 2014.

[65] C. Rosewarne, et al, “High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Improved Encoder Description Update 3”, 21st JCT-VC meeting, Warsaw, PL, document JCTVC-U1002, June. 2014.

[66] M. Mrak and J. Xu, “Improving screen content coding in HEVC by transform skipping", 2012, Proceedings of the 20th European Signal Processing Conference (EUSIPCO), pp. 1209-1213 ,Aug.2012

[67] T. Nguyen, A. Khairat, and D. Marpe, “Non-RCE1/NonRCE2/AHG5/AHG8: Adaptive Inter-Plane Prediction for RGB Content,” 13th JCT-VC meeting, Incheon, Korea, document JCTVCM0230, Apr. 2013.

[68] W. Kim, et al, “Cross-Component Prediction in HEVC”, 2015, IEEE Transactions on Circuits and Systems for Video Technology, Nov.2015 (Early Access).

[69] S. L. Yu and C. Chrysafis, “New intra prediction using intra-macroblock motion compensation,” 3rd JVT meeting, Fairfax, Virginia, USA, document JVT-C151r1, May 2002.

[70] Link to Screen content Test Sequences:
http://pan.baidu.com/share/link?shareid=3128894651&uk=889443731

Biographical Information

Shwetha Chandrakant Kodpadi was born in Karnataka, India in 1989. She received her Bachelor's degree in Electronics and Communication from Visvesvaraya Technological University, Karnataka in 2011. From 2011 to 2013, she worked in Cognizant Technology Solutions in Bangalore, India. She joined The University of Texas at Arlington to pursue her Master's degree in Spring 2014. She has worked in Multimedia Processing Lab under Dr. K. R. Rao from Fall 2014 to Fall 2015. After graduation, she plans to pursue her career in the fields of Multimedia Processing and Communications to make the best use of her skills and knowledge.