COMPLEXITY REDUCTION IN HEVC INTRA CODING

AND COMPARISON WITH H.264/AVC

by

VINOOTHNA GAJULA

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 2013

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Dr. K. R. Rao for his constant encouragement, patience, and support at all times. He has been and will be a great source of inspiration for his zeal, resourcefulness and ideas. I am very grateful for the opportunity to work under his guidance.

I would also like to thank Dr. Russell Howard and Dr. Kambiz Alavi for serving on my committee **and** offering insightful comments. **I would like to thank** Dr. Zhan Ma, Senior Standards Researcher, Samsung Research America Dallas whose research has been an inspiration for my work.

I appreciate the inputs from all members of Multimedia Processing Lab, Tushar Saxena and Shreyanka Subbarayappa. I would also like to thank my Intel manager Jaybal Nadar for his immense understanding and support.

Finally, I would like to thank my beloved family, my father G.S.R Krishna Murthy, my mother G.B.S Lalitha and my husband SuryaNarayana Kotikalapudi for their support, love and immense trust in me which made everything possible.

November 27, 2013

ABSTRACT

COMPLEXITY REDUCTION IN HEVC INTRA CODING AND COMPARISON WITH H.264/AVC

Vinoothna Gajula, M.S.

The University of Texas at Arlington, 2013

Supervising Professor: Kamisetty R. Rao

ITU-T (VCEG) and ISO/IEC (MPEG) collaborated and formed the joint collaborative team on video coding (JCT-VC) in April 2010 to develop the next-generation video coding (NGVC) standard.. HEVC standard doubles the coding efficiency and the approximately 50% less bit rate with respect to H.264/AVC, at nearly the same video quality at expense of increased complexity.

In this thesis, a technique is proposed to reduce the complexity of HEVC intra coding to get better encoding time, involving two steps - first by optimizing the PU (prediction unit) size decision process using texture complexity analysis by intensity gradients and second to obtain the reduced prediction modes by applying a combination of rough mode decision (RMD) and most probable modes (MPM) thereby reducing the number of modes based on rate distortion optimization (RDO) followed by residual quad-tree (RQT) which is used to simplify the entire process

The technique developed in this thesis achieved an average gain of 47.25% encoder time when implemented for several test sequences at very less loss in performance with high complexity reduction.

TABLE OF CONTENTS

APPENDIX

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

LIST OF ACRONYMS

3D Three Dimensional

ALF Adaptive loop filter

AVC Advanced Video Coding

B Bi-predictive

CABAC Context Adaptive Binary Arithmetic Coding

CB Coding Block

CIF Common Intermediate Format

CSVT *Circuits and Systems for Video Technology*

CTB Coding Tree Block

CTU Coding Tree Unit

CU Coding Unit

DCT Discrete Cosine Transform

DST Discrete Sine Transform

DPCM Differential pulse code modulation

FAM First angle mode

GOP Group of pictures

GPB Generalized P and B-picture

HDTV High Definition Television

HEVC High Efficiency Video Coding

HM HEVC test model

HVS Human visual system

I Intra frame

IEC International Electro-technical Committee

ISO International Standardization Organization

ITU-T International Telecommunication Union - Telecommunication Standardization Sector

JCT-VC Joint Collaborative Team on Video Coding

JM Joint model

KTA Key technical areas

LCU Largest coding unit

LD Low delay

MLP Multilayer perceptron

MPEG Moving Picture Experts Group

MPM Most Probable Modes

MSE Mean squared error

MV Motion vector

NGVC Next generation video coding

PB Prediction Block

PSNR peak signal to noise ratio

PU Prediction Unit

QCIF Quarter CIF

QP Quantization Parameter

RA Region adaptive

RDO Rate distortion optimization

RDOQ Rate distortion optimized quantization

RMD Rough Mode Decision

RQT Residual Quad tree

SAD Sum of absolute differences

SAG Sum of absolute gradients

SAM Second angle mode

SAO Sample adaptive offset

SATD Sum of absolute transform differences

SEI Supplemental Enhancement Information

SSE Sum of Square error

SSIM Structural similarity index metric

TAM Third angle mode

TB Transform Block

TU Transform Unit

URQ Uniform Reconstruction Quantization

VCEG Video Coding Experts Group

VCL Video Coding Layer

CHAPTER 1

INTRODUCTION

Computing power, network support, and applications for image, audio and video data over the past three decades have seen huge improvements. Attributing to all these changes, video has become a highly consumed form of data in various forms for example, videophony, high definition television (HDTV), videoconferencing, and the digital video disk (DVD). Video data comprising a huge amount of data in uncompressed form needs huge bandwidth. However with growing demand for video/images bandwidth is constrained, making video data compression critical for transmission, storage and utilization of data. The functionality of video compression in storage and transmission is shown in figure 1.1 [1], [2].

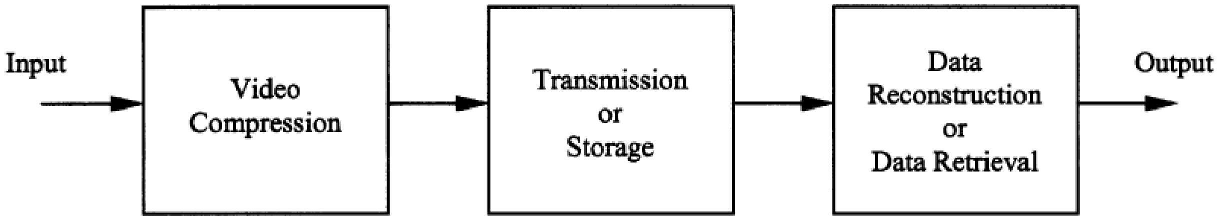


Figure 1.1: Video compression functionality in data transmission and storage [2].

The video data compression is the process of reducing the amount of data required to represent a video sequence. This is also called as video encoding and the complimentary process of retrieving a video sequence from compressed format is known is video decoding. Video coding is a process of video encoding and decoding together [1], [3].

1.1 Video Coding Standards

ITU-T (VCEG) (international telecommunication union – telecommunications (video coding experts group)) developed H.120 standard – the first ever video coding standard in 1984 followed by H.261 standard in 1990. [5]

ISO (international standardization organization) and IEC (international electro-technical committee) started MPEG project (moving picture experts group) with the main aim of developing video coding standards and developed MPEG-1 standard in 1988. [5]

ITU-T (VCEG) and ISO/IEC (MPEG) jointly developed MPEG-2/H.262 standard approved in 1993, MPEG-2 is the ISO name and H.262 is the ITU_T name. In 1995, ITU_T developed H.263 which is a major step towards the current coding standards, especially for progressive video coding. Around the same time MPEG-4 standardization was begun and various new concepts were introduced in this standard which were absent in earlier standards like 3D graphics, interactive graphics etc., The next step in further developing the coding standards was to improve the compression ratio at twice that of earlier standards. With this agenda H.264/MPEG-4 part 10/AVC (advanced video codec) has been developed [5], [6].

ITU-T (VCEG) and ISO/IEC (MPEG) collaborated and formed the Joint Collaborative Team on Video Coding (JCT-VC) in April 2010 to develop the next-generation video coding (NGVC) standard and several meetings were held and the first TMuC (test model under construction) was High Efficiency Video Coding (HEVC) [7]. The evolution of the coding standards is shown in the figure 1.2. [4]

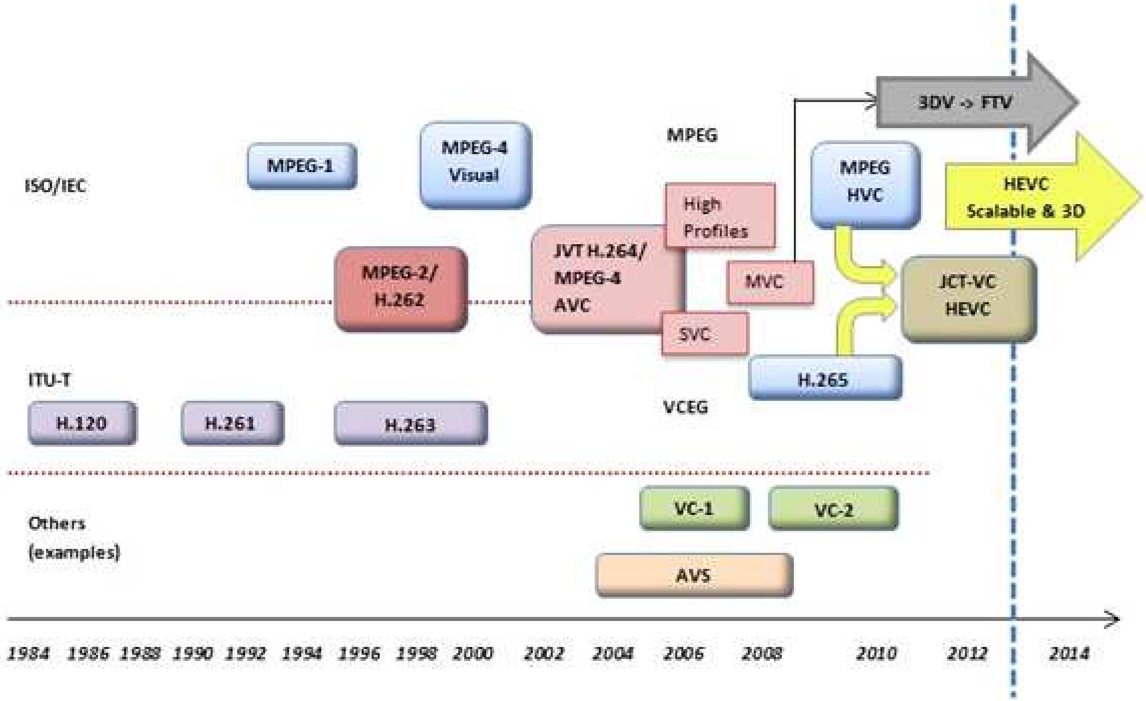


Figure 1.2: Generations of video coding standards [4].

HEVC is the most new coding standard only with some profiles released in January2013. The major achievements of HEVC in comparison with H.264 are [8]:

1. Flexible prediction modes and transform block sizes and better partitioning options.
2. Improved interpolation and deblocking filters, prediction and signaling of modes and motion vectors.
3. Support efficient parallel processing.

HEVC provides better compression at increased processing power or the coding efficiency is improved significantly for high resolution videos. However encoder complexity is much higher than H.264. With the introduction of HEVC into market almost all appliances, applications and software systems will start using HEVC as this gives almost twice the compression of H.264 and also better quality for high resolution video. [12]

1.2 Objective of Thesis

The objective of this thesis is reducing the complexity of HEVC to get better compression time by optimizing both the PU (prediction unit) size decision process using texture complexity analysis by intensity gradient of the lossless image. The best prediction mode is obtained by applying a combination of rough mode decision (RMD) and most probable modes (MPM) thus reducing the number of modes based on RDO followed by residual quad-tree (RQT) process.

1.3 Thesis Outline

The current chapter gives a brief introduction on history of the coding standards.

The chapter 2 gives the overview of HEVC in brief, highlighting on encoder overall working and basic working of the codec and reference software HM [8] is introduced briefly.

In chapter 3, the intra-coding of HEVC is dealt in detail. It introduces the concept of rate distortion optimization (RDO). The best RDO method is selected.

In chapter 4, the main technique for this thesis is introduced and its working is explained in a step by step process followed by chapter 5 with the results, conclusions and the future work.

CHAPTER 2

OVERVIEW OF HIGH EFFICIENCY VIDEO CODING

HEVC standard doubles the coding efficiency and the approximately 50% less bit rate with respect to H.264/AVC, at approximately the same video quality. HEVC encoders are relatively more complex than its predecessors whereas the decoder complexity has only slightly increased which makes HEVC applicable to already existing hardware with very few amendments [10], [12].

The basic design of HEVC remained the same as that of H.264/AVC i.e., the block based hybrid coding approach which efficiently exploits the temporal statistical dependencies and the spatial statistical dependencies [7].

2.1 HEVC Encoder and Decoder Blocks

HEVC employs adaptive and flexible quad-tree coding block partitioning structure which enables efficient use of large and multiple sizes of prediction, coding, and transform block and employs improved intra prediction, adaptive motion parameter prediction, new loop filter and an enhanced context-adaptive binary arithmetic coding (CABAC) as entropy coding method [11]. Figure 2.1 shows the encoder block diagram [7] and figure 2.2 shows the decoder block diagram [9].

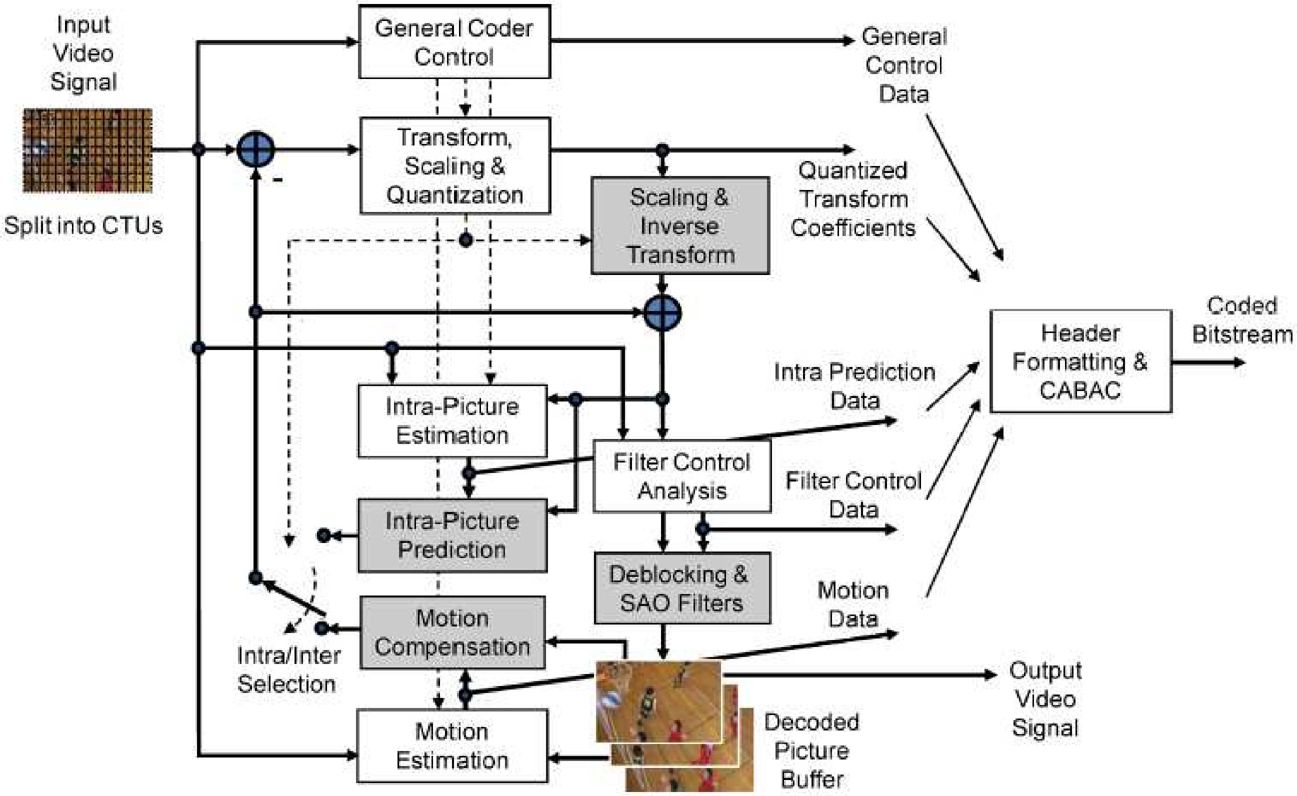


Figure 2.1: HEVC encoder with decoder elements shaded in gray [7].

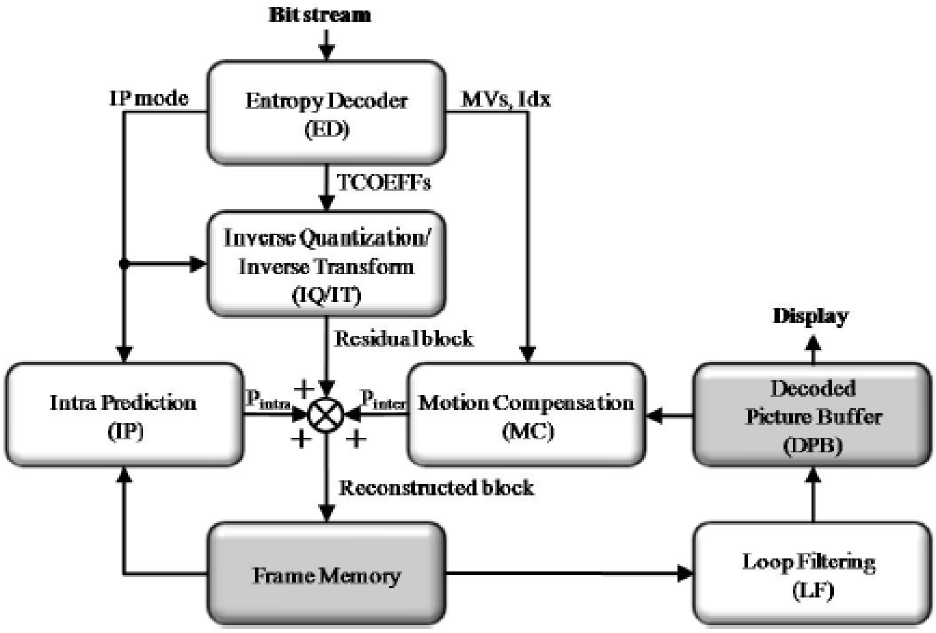


Figure 2.2: HEVC decoder block diagram [9].

2.1.1 Structure of Encoder

The quad-tree block partitioning is based on coding tree unit (CTU) structure which is analogous to macro block in previous standards. A video is a packet or sequence of frames/ pictures and in HEVC each coded video frame/picture is partitioned into tiles and/slices and further into CTUs. CTUs are subdivided into square regions called coding units (CU). CUs are predicted using intra or inter prediction where the first picture /frame at each random access point of a video sequence is coded using only intra prediction so that it has no dependence on other pictures. The remaining frames are mostly coded by inter prediction and then residual is transformed using transform units and encoded using CABAC [10], [11], [12].

2.2 Encoder Building Blocks

The HEVC encoder basic building blocks are explained in detail

2.2.1 Input Video and Sampling

HEVC encoder supports progressive video, but interlaced video can also be encoded by using a special syntax called meta data syntax along with video input [12]. HEVC uses YC_bC_r color space with 4:2:0 color format with 8 bps (bits per color sample). Y is luma component, C_b and C_r are chroma components [11] and this format is shown in figure 2.3 [3]. Figure 2.4 shows various resolutions of video at the user end [27]. The comparison of the sampling methods are considered i.e., common intermediate format (CIF), quarter CIF (QCIF), 2CIF and 4CIF are shown in table 2.1 (a) [30].

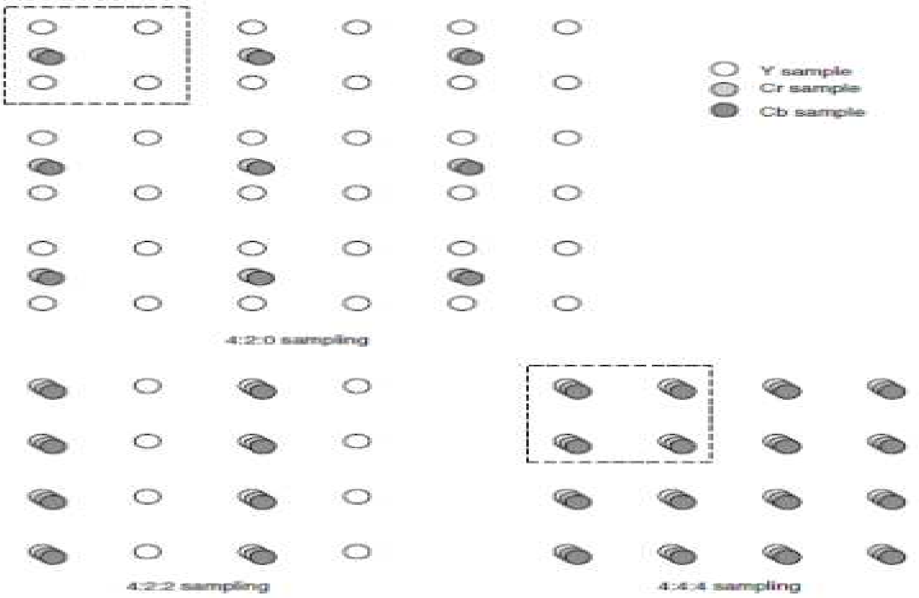


Figure 2.3: 4:2:0, 4:2:2, 4:4:4 Y C_b C_r sampling patterns [3]

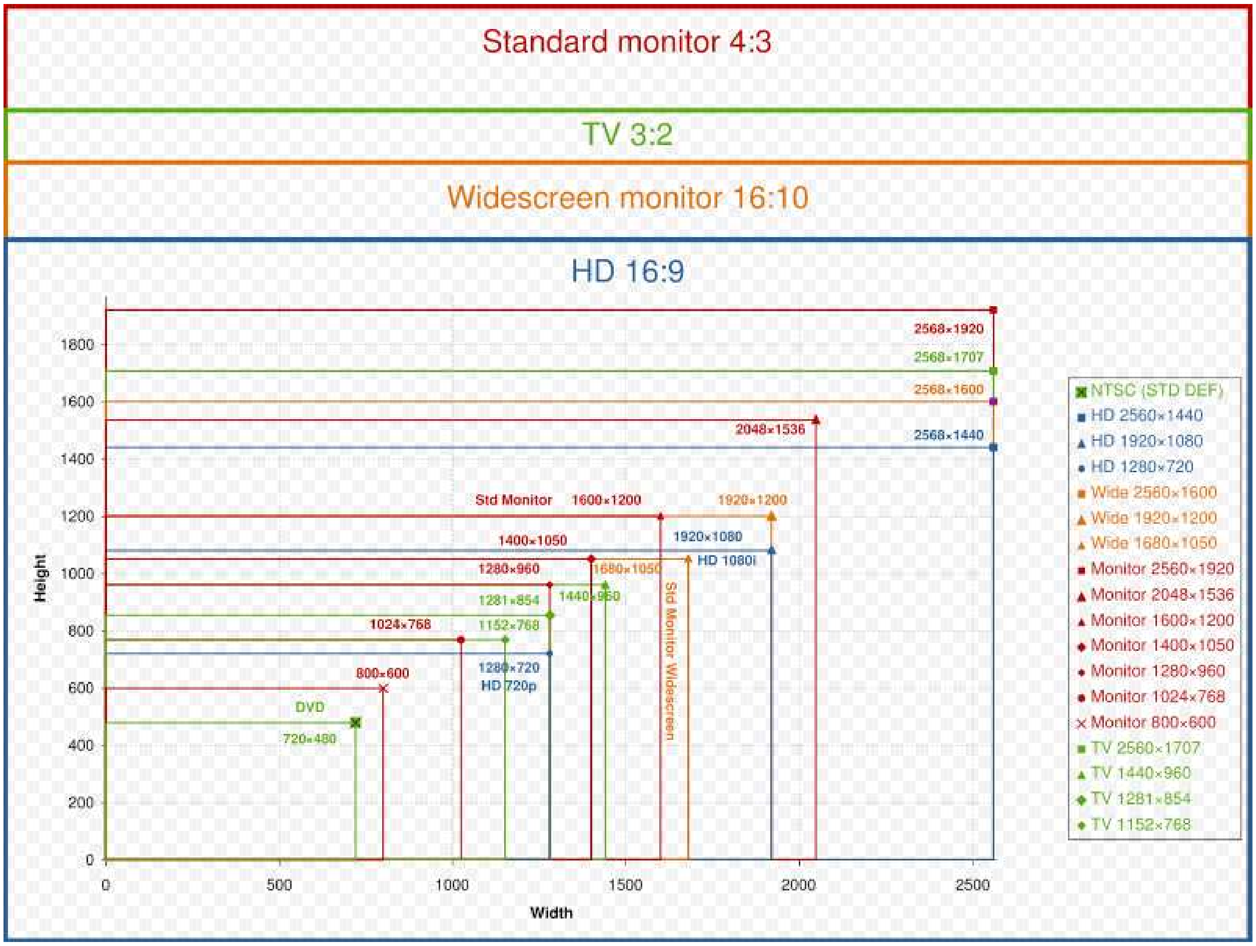


Figure 2.4: Various screen resolutions at users end [27].

Table 2.1(a) Description of video formats [30]

Term	**Pixels (W x H)**	**Notes**
QCIF	176 x 120	Quarter CIF (half the height and half the width as CIF)
CIF	352 x 240	CIF is taken as reference
2CIF	704 x 240	2 times CIF width
4CIF	704 x 480	2 times CIF width and 2 times CIF height

2.2.2 Coding Tree Units (CTUs) , Coding Units (CUs) and Prediction Units(PUs)

To start with the encoding process one needs to know the quad-tree in detail, so starting with CTU as it is the root of quad-tree. CTU is made of a luma coding tree block (CTB), two chroma CTB and corresponding quad-tree syntax, where the luma CTB is a block of size NxN and chroma CTBs are of size (N/2)x(N/2). N is chosen inside the bit stream and can be 16, 32 or 64. The size of CTB is largest supported size of coding block (CB). CTB may contain one or more coding units (CU). CU has an associated partitioning into prediction units (PUs) and transform units (TUs). The coding mode, intra or inter prediction, is selected at CU level. This entire structure is shown in figures 2.5 and 2.6 [7], [11], [12].

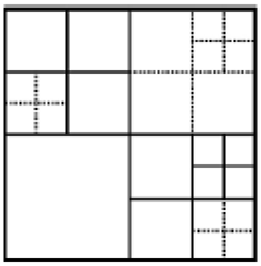


Figure 2.5: CTB and its partitions

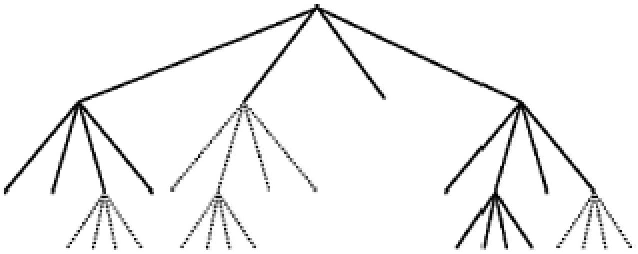


Figure 2.6: Quad tree structure

Solid lines indicate splitting CTBs into CBs and dotted lines indicate TB boundaries [7]

Based on prediction type CBs are split and prediction is done for prediction blocks (PBs) whose size varies from 64x64 to 4x4 samples, the various formats of splitting PBs is shown in figure 2.7. The entire division can also be clearly shown in a picture frame in figure 2.8 [7], [14].

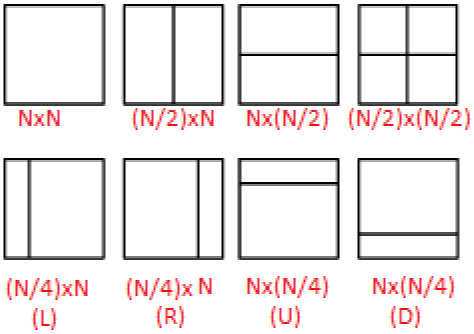


Figure 2.7: PB sizes split from CB where NxN and Nx(N/2) are only supported by inter prediction [7].

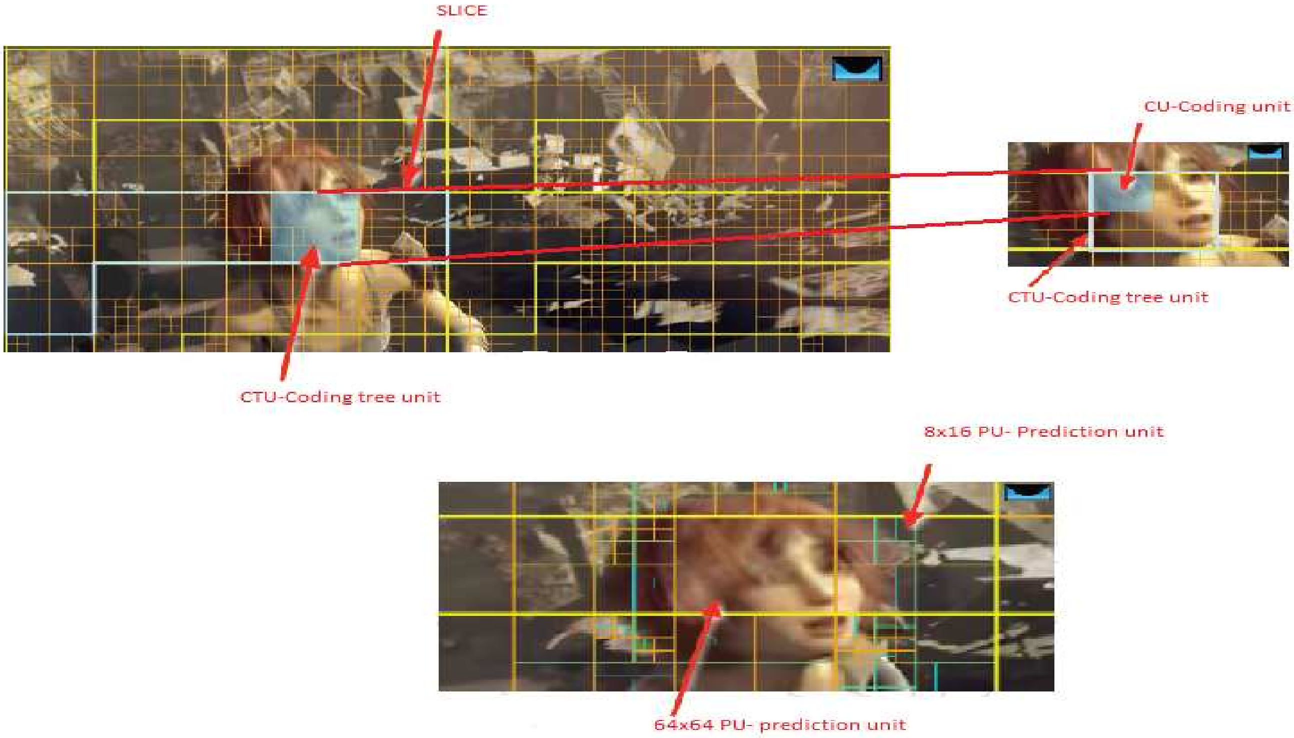


Figure 2.8 Pictorial representations of various block divisions and subdivisions.

2.2.3 Transform Blocks (TBs)

The luma and chroma CB residual are identical to luma and chroma transform block (TB) or further split into multiple luma and chroma TBs respectively. The HEVC transform functions, integer basis functions, are defined for the TB sizes (4×4), (8×8), (16×16), and (32×32) as shown in figure 2.9. When TB size is 4×4, the luma intra prediction transform used is derived from discrete sine transform (DST) [12].

32x32, 16x16, 8x8
and 4x4 transform
sizes

Figure 2.9: TB sizes

2.2.4 Slices and Tiles

Slices and tiles are used in coding of predicted frames. A slice can be defined as a group of CTUs in an independent slice segment and all dependent slice segments. A slice segment can be defined as group of CTUs ordered consequently in a tile scan and contained in a NAL unit. Independent slice segment values i.e., slice segment syntax of slice segment header is not inferred from preceding slice segment whereas a dependent slice segment components are inferred from previous independent slice segments or dependent slice segment boundaries [12].

A tile can be defined as a rectangular region containing a group of CTUs in a CTB raster scan.

The various types of slices are:

I (Intra) slice: All CUs of I slice are coded using only intra prediction.

P (Predictive) slice: Some CUs of a P slice can be coded using inter prediction with at most one MC prediction signal (using reference picture list 0) per PB and some CUs are coded similar to I slice.

*B (Bi-prediction) slice: S*ome CUs of the B slice can be coded using inter prediction with at most two MC prediction signals (using reference picture list 0 and list 1) per PB and some CUs are coded similar to I slice.

One or both of the following conditions are fulfilled for each slice/slice segment and tile:

– All coding tree units in a slice/slice segment belong to the same tile.

– All coding tree units in a tile belong to the same slice/slice segment

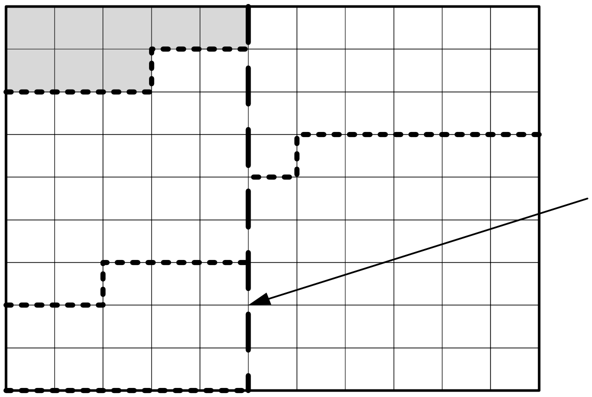


Figure 2.10: Tile and slice division 1 [12]

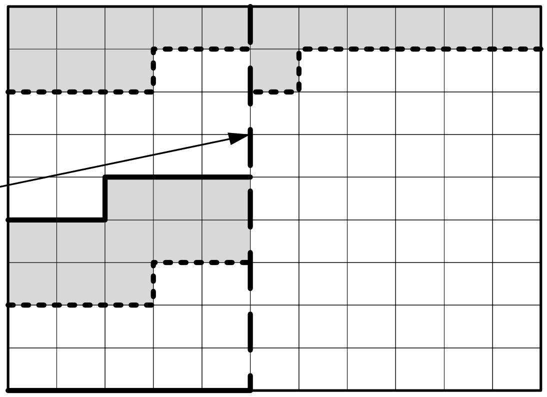


Figure 2.11: Tile and slice division 2 [12].

Figures 2.10 and 2.11 are 2 models of tile and slice division in a 11x9 frame. Figure 2.10 is of a picture that is partitioned into 2 tiles in total, and 1 slice followed by 1 independent slice segment (shaded) and 3 dependent slice segments. Figure 2.11 is partitioned into 2 tiles and 3 slices where left tile has 2 independent slice segments and 2 dependent slice segments and right tile has 1independent slice [12].

2.2.5 Intra Prediction

Intra prediction is dependent on TB size and prediction signal is formed from previously decoded boundary samples. Intra prediction has a total of 35 directional prediction modes, consisting of 33 angular modes, DC and planar modes for luma component of each PU (fig. 2.12). For the chroma component of intra PU, the best chroma prediction modes among five modes including planar, DC, vertical, horizontal and a direct copy of the intra prediction mode for the luma component is selected by the encoder [12], [13].

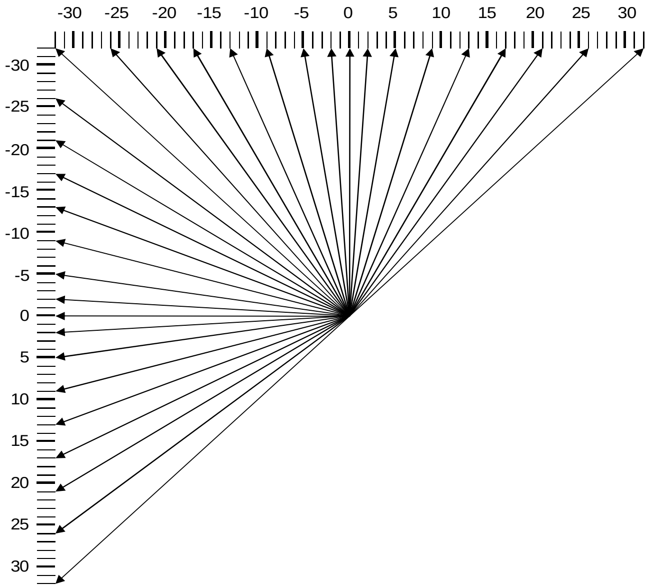


Figure 2.12: 33 Intra prediction directions [12].

In luma component the neighboring samples used for intra prediction sample generations are filtered before the sample generation process. The filtering process is controlled by the given intra prediction mode and TB size. If DC intra prediction mode or the 4x4 transform block is selected, neighboring samples are not filtered. If the distance between the given intra prediction mode and vertical mode/horizontal mode is larger than threshold, a bi-linear filter is enabled [12]. Intra prediction of HEVC is explained in detail in chapter 3.

2.2.6 Inter Prediction

Inter prediction supports more number of PB divisions than intra prediction mode. Inter coded PU has a set of motion parameters comprising- motion vector, reference picture index, reference picture list usage flag (for inter prediction sample generation) [7], [12].

When a CU is coded with skip mode- the CU is represented as one PU that has no significant transform coefficients and motion parameters obtained by merge mode. In inter coded PUs, the encoder can use merge mode or explicit transmission of motion parameters exclusively for each PU. The merge mode can be applied to inter coded PU and skip mode. The merge mode is to find the neighboring inter coded PU such that its motion can be inferred as the one for the current PU, from multiple candidates formed by spatially and/or temporally neighboring PUs, and to transmit the corresponding index indicating the chosen candidate [12].

HEVC has motion vectors with units of one quarter of the distance between luma samples and units of one eighth of the distance between chroma samples as the sampling format for 4:2:0 sampling. For each PB, one or two MVs can be transmitted. An advanced motion vector prediction (AMVP) is used in HEVC [11], [12]. All the filters used are shown in table 2.2. Tables 2.3 and 2.4 show the DCT-If coefficients of 8-tap luma and 4-tap chroma filter respectively [12].

Table 2.2 Filters used in HEVC [7].

Filter	Sample to be Interpolated
8-tap filter	Luma half-sample positions
7-tap filter	Luma quarter-sample positions
4-tap filter	Chroma sample

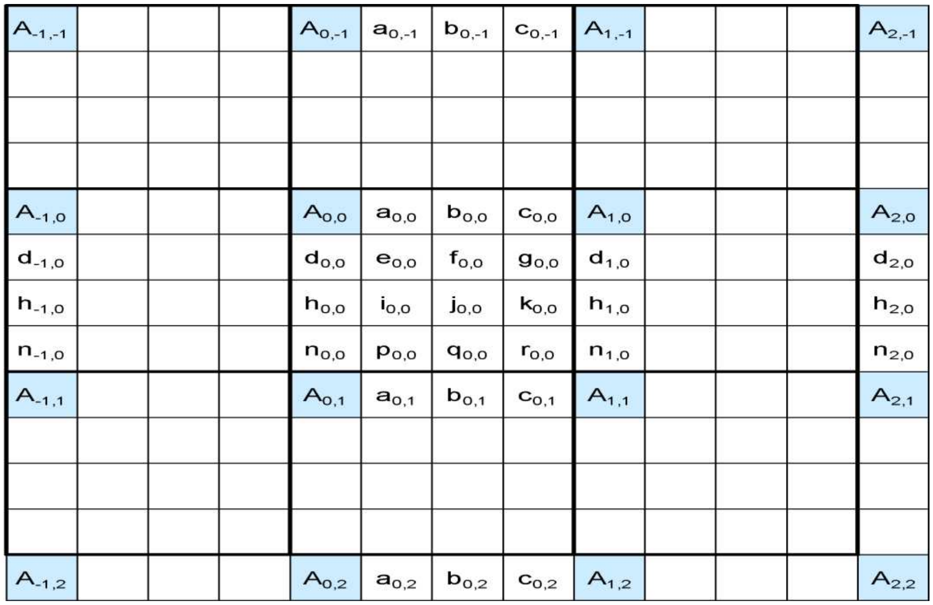
Table 2.3: 8-tap filter coefficients (DCT-IF) [12]

Position	Filter coefficients
¼	{ -1, 4, -10, 58, 17, -5, 1 }
2/4	{ -1, 4, -11, 40, 40, -11, 4, -1 }
¾	{ 1, -5, 17, 58, -10, 4, -1 }

Table 2.4: 4-tap coefficients for 1/8th chroma interpolation (DCT-IF) [12]

Position	Filter coefficients
1/8	{ -2, 58, 10, -2 }
2/8	{ -4, 54, 16, -2 }
3/8	{ -6, 46, 28, -4 }
4/8	{ -4, 36, 36, -4 }
5/8	{ -4, 28, 46, -6 }
6/8	{ -2, 16, 54, -4 }
7/8	{ -2, 10, 58, -2 }

HEVC uses a single separable interpolation process for generating all fractional positions without intermediate rounding operations, which improves precision and simplifies fractional sample interpolation. Figure 2.13 shows the sample positions, $A_{i,j}$ represent the available luma samples at integer sample locations and positions labeled with lower-case letters represent samples at non-integer sample locations [12].



$A_{-1,-1}$				$A_{0,-1}$	$a_{0,-1}$	$b_{0,-1}$	$c_{0,-1}$	$A_{1,-1}$				$A_{2,-1}$
$A_{-1,0}$				$A_{0,0}$	$a_{0,0}$	$b_{0,0}$	$c_{0,0}$	$A_{1,0}$				$A_{2,0}$
$d_{-1,0}$				$d_{0,0}$	$e_{0,0}$	$f_{0,0}$	$g_{0,0}$	$d_{1,0}$				$d_{2,0}$
$h_{-1,0}$				$h_{0,0}$	$i_{0,0}$	$j_{0,0}$	$k_{0,0}$	$h_{1,0}$				$h_{2,0}$
$n_{-1,0}$				$n_{0,0}$	$p_{0,0}$	$q_{0,0}$	$r_{0,0}$	$n_{1,0}$				$n_{2,0}$
$A_{-1,1}$				$A_{0,1}$	$a_{0,1}$	$b_{0,1}$	$c_{0,1}$	$A_{1,1}$				$A_{2,1}$
$A_{-1,2}$				$A_{0,2}$	$a_{0,2}$	$b_{0,2}$	$c_{0,2}$	$A_{1,2}$				$A_{2,2}$

Figure 2.13: Fractional and integer positions for luma interpolation [12].

When an inter-predicted CB is not coded in the skip or merge modes, the MV is differentially coded using MV predictor. The difference between the actual motion vector and its predictor and the index are transmitted to the decoder. [12]

2.2.7 Transform, Quantization and Reconstruction.

Residuals generated by subtracting the prediction value from the input value are spatially transformed and quantized. In the transform process partial butterfly structure is implemented. Approximations to discrete cosine transform (DCT) are used. In the case of 4x4 luma intra predicted residuals a variation of discrete sine transform (DST) is used. 52-level quantization steps and rate-distortion optimized quantization (RDOQ) are used in the quantization. The generated samples and quantized transform coefficients are encoded using context adaptive binary arithmetic coding (CABAC) [7].

Reconstructed samples are obtained by inverse quantization and inverse transform. Reconstructed CTUs are assembled to construct a picture and to encode the next picture this data is stored in the decoded picture buffer. To improve the visual quality and obtain better coding efficiency of the reconstructed samples two in-loop filtering processes, deblocking filtering and sample adaptive offset (SAO) are adopted, Figure 2.14 shows the scope of HEVC encoder and decoder [7], [12], [14].

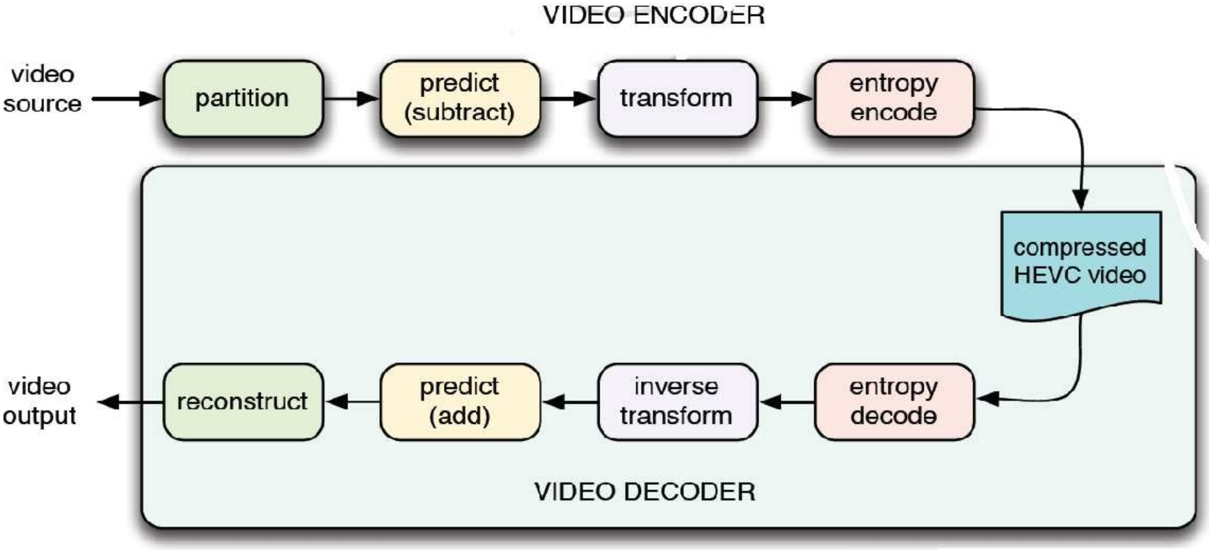


Figure 2.14 Structure of HEVC encoder and decoder [14]

Summary

In this chapter HEVC encoder, intra and inter predictions are briefly introduced. In Chapter 3 intra, RDOQ concept and the technique developed in this thesis are explained in detail.

CHAPTER 3

HEVC INTRA PREDICTION

Intra prediction is used to remove spatial redundancies. In this chapter, HEVC intra coding and RDOQ are discussed in detail. Intra prediction in HEVC is spatial sample prediction followed by transform coding and the use of more flexible block sizes and more intra modes. In total there are 35 prediction modes of which 33 angular modes, a planar and DC modes are supported (fig. 2.5). Figure 3.1 shows the intra prediction modes. To reduce the complexity, three most probable modes (MPM) for each PU are specified based on the neighboring PU, enabling fixed length coding possible for the rest of 32 intra modes [15], [16].

Table 3.1 Intra prediction modes [15]

Intra Prediction	Mode Associated Names
0	Planar
1	DC
2 to 34	Angular (N), N = 2...34

The basic elements in the HEVC intra coding design are shown below [15]:

1) Quad tree-based coding structure;

2) 33 prediction directions for angular predictions;

3) Planar prediction to generate smooth sample surfaces;

4) Adaptive smoothing of the reference samples;

5) Filtering of the prediction block boundary samples;

6) Residual transform and coefficient scanning;

7) Intra mode coding based on contextual information;

3.1 Angular Intra Prediction

In HEVC intra prediction for each CU- PUs is defined, where each PU specifies a region with individual prediction parameters. CU is further split into a quad-tree of TUs, and each TU is residual coded with a transform of the size of the TU. Figure 3.2 shows the PU sizes and corresponding number of intra prediction [15]. Figure 3.1 shows the mapping between the intra prediction direction and the intra prediction mode number [7].

Table 3.2 PU sizes and corresponding number of intra prediction [15].

PU size	Number of intra prediction
64x64	4
32x32	35
16x16	35
8x8	35
4x4	18

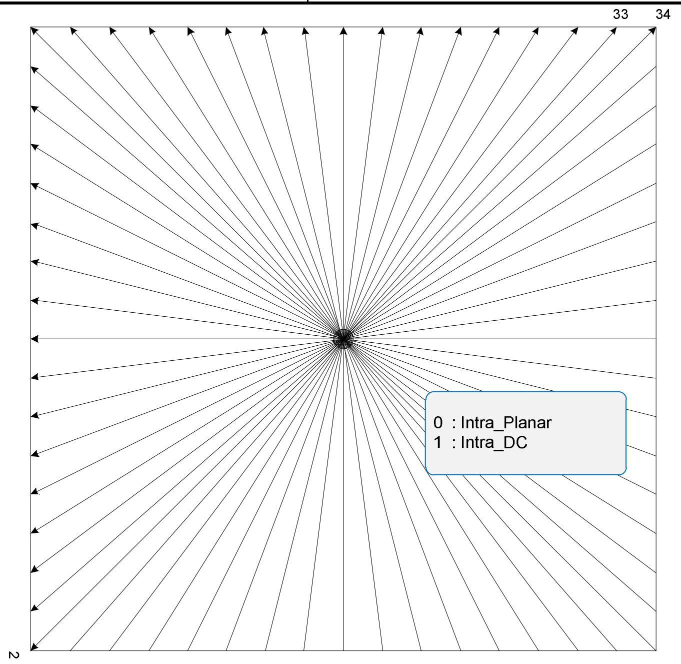


Figure 3.1 Mapping between the intra prediction direction and mode number [7].

In intra−angular mode, each TB is predicted by angle from spatially neighboring samples that are reconstructed. If the TB is M×M in size, a total of 4*M*+1 spatially neighboring samples can be used for the prediction. Samples from left, above, and above right and sometimes lower left (if available) of the TB being predicted can be used for prediction. To avoid sample-by-sample switching between reference row and column buffers, for intra modes as shown in figure 3.2 for the modes 2–17, the samples in the above row are projected as additional samples located in the left column and for the range of 18–34, the samples located at the left column are projected as samples located in the above row and the sample location is calculated with 1/32 sample accuracy. Bilinear interpolation uses two closest reference samples located at integer positions to obtain the value of the projected reference sample [15], [16]

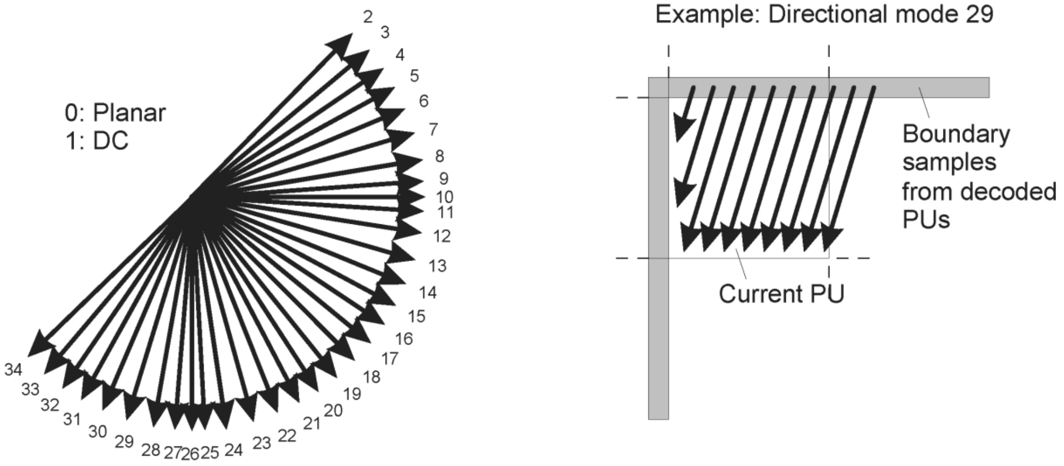


Figure 3.2 Intra prediction angular orientation and example of direction mode 29 [7].

The angles are designed to provide denser coverage for near-horizontal and near-vertical angles and coarser coverage for near-diagonal angles. All the prediction modes use the same basic set of top left reference sample of the image block to be predicted. $R_{x,\ y}$ represents the reference samples and $P_{x,y}$ represents the sample to be predicted, where x, y are the spatial coordinates. $P_{x,\ y}$ is derived from reference $R_{x,\ y}$ which is one pixel above and to the left of the block's top-left corner, The sub pixel location in between $R_{i,\ 0}$ and $R_{i+1,\ 0}$, c_x and c_y represents pixel parameters corresponding to x and y coordinates, w_y represents the weight/weighing parameter of weighed prediction, i stands for reference sample index, '>>' denotes a bit shift operation to the right and & denotes logical AND operation. The parameters c_y and w_y depend only on y coordinate in vertical mode i.e., 18-34. Similarly parameters c_x and w_x depend only on x coordinate for horizontal modes i.e., 2-17 [15].

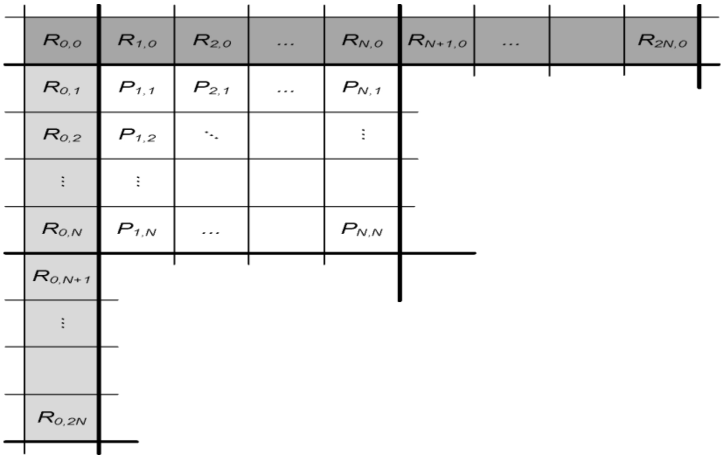


Figure 3.3 Reference samples $R_{x,\ y}$ and dependent samples $P_{x,\ y}$ of NxN samples [15]

Sample prediction equation for vertical modes i.e., angular modes 18-34:

$P_{x,y}= ((32 – w_y) \cdot R_{i,0} + w_y \cdot R_{i+1,0} + 16)$…………………[3.1]

$c_y= (y \cdot d) >>5$ ………………………………………………[3.2]

$w_y= (y \cdot d)\ \&\ 31$………………………………………………[3.4]

$i= x + c_y$ ………………………………………………[3.5]

Sample prediction equation for horizontal modes i.e angular modes 2-17:

$P_{x,y}= ((32 – w_x) \cdot R_{i,0} + w_x \cdot R_{i+1,0} + 16)$ …………………[3.6]

$c_x= (x \cdot d) >> 5$ ………………………………………………[3.7]

$w_x= (x \cdot d)\ \&\ 31$ ………………………………………………[3.8]

$i= y+ c_x$ ………………………………………………[3.9]

For chroma component of intra PU, the best chroma prediction mode among five modes including planar, DC, vertical, horizontal and a direct copy of the intra prediction mode for the luma component is selected by the encoder [12], [13]. When Intra prediction mode number for chroma component is 4 then the intra prediction direction used is same as that of luma prediction. Otherwise the intra prediction mode number for luma component is used for chroma prediction and the intra prediction direction of 34 is used for the intra prediction sample generation for chroma component as shown in table 3.3 [11].

Table 3.3 Mapping between intra prediction direction and intra prediction mode for chroma [11].

Chroma Intra prediction mode	**Intra prediction direction**				
	0	**26**	**10**	**1**	**X (0 <= X < 35)**
0	34	0	0	0	0
1	26	34	26	26	26
2	10	10	34	10	10
3	1	1	1	34	1
4	0	26	10	1	X

3.2 Planar and DC Predictions

Intra−DC prediction uses an average value of reference samples for the prediction. In intra planar mode to prevent discontinuities along the block boundaries, the average values of two linear predictions using four corner reference samples are used. In HEVC all block sizes are supported by planar prediction [7].

3.3 Reference Sample Smoothing and Boundary Sample Smoothing

HEVC applies smoothing filters to get a smooth prediction. The filters are based on detected discontinuity and block size. No filters are applied for 4×4 blocks. For 8×8 blocks in diagonal directions, i.e., inter prediction modes 2, 18, or 34 use the reference sample smoothing.

For 16×16 blocks, the reference samples are filtered except for the near-horizontal and near-vertical directions i.e., for inter prediction modes 9–11 and 25–27. For 32×32 blocks, all directions except the horizontal and vertical i.e. the intra prediction modes 10 and 26 respectively use the smoothing filter and when discontinuity exceeds a threshold, bilinear interpolation is applied from the three neighboring region samples. In HEVC planar mode, smoothing filter is used when the block size is greater than or equal to 8×8, and no smoothing filter is required for the Intra−DC case. [7]

3.4 Rate Distortion Optimization (RDO)

With the increase in number of modes and prediction methods in HEVC the complexity also has increased. RDO method is used in selection of best modes thereby decreasing the complexity of HEVC intra prediction. Mode selection problem is overcome by optimization of the procedure to minimize the distortion (D) for a given bit rate (R) to get the least cost (J).

$$J = D + \lambda \cdot R \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(3.10)}$$

The RDO selection algorithm attempts to derive best combination that minimizes the cost. So there is trade-off between Rate and Distortion which can be controlled by the Lagrange multiplier λ. A smaller λ will minimize D, with the effect of increased R whereas a larger λ will minimize R at the expense of a higher D. Selecting the best λ for a particular cost equation is the task of the algorithm described in this thesis. The intra prediction in HEVC can be divided into two stages. The prediction unit (PU) size decision and mode decision. Correspondingly there are two costs which need to be calculated.

3.4.1 Calculation of Parameters for Cost Functions

The lambda values for these cost computations are defined as below [12], [17]:

$\lambda_{\text{mode}} = \alpha * W_k * 2^{((QP-12)/3.0)}$ …………………………………….. (3.11)

$\lambda_{\text{pred}} = \sqrt{\lambda_{\text{mod } e}}$ ………..…………………………………………. (3.12)

λ_{mode} lambda mode represents the Lagrangian multiplier for mode decision, λ_{pred} is Lagrangian multiplier for size prediction, W_k represents weighting factor dependent on encoding configuration, and QP represents luma quantization parameter. W_k is derived from table 3.4, and is to be multiplied by 0.95 when SATD based motion estimation is used [12].

Table 3.4 Derivation of W_k [12]

K	QP offset hierarchal level	Slice type	Hierarchal level of referenced picture	W_k
0	0	I	-	0.57
1	0	GPB	1	RA: 0.442 LD: 0.578
2	1, 2	B or GPB	1	RA: 0.3536 * Clip3(2.0, 4.0, (QP-12)/6.0) LD: 0.4624 * Clip3(2.0, 4.0, (QP-12)/6.0)
4	3	B	0	RA: 0.68 * Clip3(2.0, 4.0, (QP-12)/6.0)

Lambda chroma, λ_{chroma}, is used for chroma predictions in RDOQ, SAO and ALF processes. λ_{chroma}, is dependent upon weighting parameter w_{chroma}, which are defined as follows [12]:

$w_{\text{chroma}} = 2^{(\text{QP}-\text{QP}_{\text{chroma}})/3}$ …………………….. (3.13)

$\lambda_{chroma} = \lambda_{\text{mode}} / w_{\text{chroma}}$ ……………………. (3.14)

w_{chroma} is also used to define cost function to be used for mode decision in order to weight chroma part of sum of square error (SSE).

3.4.2 Cost Function for Prediction Parameter Decision

The cost for prediction parameter decision $J_{\text{predsize,}}$ is defined as

$$J_{\text{predsize}} = (D_{SAD} \text{ or } D_{SATD}) + \lambda_{\text{pred}} * B_{\text{pred}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.15)$$

B_{pred} represents bit cost for making decision, which depends on each decision case. D_{SAD} is sum of absolute difference (SAD) and D_{SATD} is Hadamard transformed SAD.

3.4.3 Cost Function for Mode Decision

The cost for mode decision J_{mode} is defined as

$$J_{\text{mode}} = (SSE_{\text{luma}} + w_{\text{chroma}} * SSE_{\text{chroma}}) + \lambda_{\text{mode}} * B_{\text{mode}} \ldots\ldots\ldots (3.16)$$

B_{mode} represents bit cost for mode decision, which depends on each decision case, and SSE_{luma} and $\text{SSE}_{\text{chroma}}$ are sum of square errors, sum of squares of the difference between two blocks with the same block size, for luma and chroma respectively.

Summary:

This chapter provides an in-depth analysis of HEVC intra prediction and RDO in HM encoder. This gives very clear idea on Intra prediction cost calculation and various components and parameters which are required for intra prediction. In chapter 4, the gradient RDO for HM 9.0 is analyzed in detail and the best optimization method is selected to reduce the complexity of HEVC.

CHAPTER 4

FAST ALGORITHM FOR INTRA PREDICTION

HEVC introduced more number of modes, more features and larger prediction blocks. Correspondingly the complexity also increased. The objective of this thesis is to reduce the complexity and improve the HEVC coding time. To understand the working of HEVC intra coding, I frame coding is analyzed and the average time consumed by all intra coding tools is found out as shown in table 4.1 [17].

Table 4.1 Intra frame coding tools and corresponding time consumed [17].

Tool	Time(%)
Intra prediction	30.55
Transform and quantization	19.08
Entropy coding	19.74
Deblocking filtering	0.13
Sample adaptive offset	0.5
Adaptive loop filtering	29.59
Others	0.39

It is understood from table 4.1, that intra prediction contributes to maximum complexity. Hence to reduce HEVC complexity this thesis focuses on the optimization of intra prediction. This thesis proposes a fast intra mode decision method by optimizing both the PU size decision process using texture complexity analysis by intensity gradient, and reducing prediction modes by applying a combination of rough mode decision (RMD) and most probable modes (MPM) thereby reducing the number of modes to RDO followed by residual quad-tree (RQT) checking is used to simplify the entire process [18].

4.1 Fast Algorithm for Intra Coding Luma Prediction

The fast algorithm used in this thesis is explained in an elaborate step by step process:

Step 1: In HEVC, a picture is divided into a sequence of LCUs (large CUs) and further into CUs. The LCU is divided into the smallest (4x4) unit and the intensity gradient calculator is applied to extract the texture complexity and the texture direction of the LCU. Fine CU partition is used for textured region and coarse CU partition is to be used for homogenous region. Then go to step 2.

Step 2: Texture gradient calculations: Intensity gradient of pixel at (x, y) coordinates, g(x, y), is derived from sample of image to be predicted p(x, y).The intensity gradients are calculated for all four possible directions to explore the texture complexity of the original LCU [19].

(a) Horizontal gradient: The intensity gradient formulae for horizontal orientation are as follows [19]:

$$g_h(x, y) = |p(x, y) - p(x + 2, y)| \tag{4.1a}$$

$$g_h(x + 1, y) = |p(x + 1, y) - p(x + 3, y)| \tag{4.1b}$$

$$g_h(x, y + 1) = |p(x, y + 2) - p(x + 2, y + 2)| \tag{4.1c}$$

$$g_h(x + 1, y + 1) = |p(x + 1, y + 2) - p(x + 3, y + 2)| \tag{4.1d}$$

(b) Vertical gradient: The intensity gradient formulae for vertical orientation are as follows [19]:

$$g_v(x; y) = |p(x; y) - p(x + 2; y)| \tag{4.2a}$$

$$g_v(x + 1; y) = |p(x + 1; y) - p(x + 3; y)| \tag{4.2b}$$

$$g_v(x; y + 1) = |p(x; y + 2) - p(x + 2; y + 2)| \tag{4.2c}$$

$$g_v(x + 1; y + 1) = |p(x + 1; y + 2) - p(x + 3; y + 2)| \tag{4.2d}$$

(c) Left-down gradient: The intensity gradient formulae for left-down orientation are as follows

[19]: $g_{ld}(x, y) = |p(x, y) - p(x + 2, y)|$ …………………................ (4.3a)

$g_{ld}(x + 1, y) = |p(x + 1, y) - p(x + 3, y)|$ …………………..... (4.3b)

$g_{ld}(x, y + 1) = |p(x, y + 2) - p(x + 2, y + 2)|$ ……………….... (4.3c)

$g_{ld}(x + 1, y + 1) = |p(x + 1, y + 2) - p(x + 3, y + 2)|$ ……….. (4.3d)

(d) Right-down gradient: The intensity gradient formulae for right-down orientation are as

follows [19]: $g_{rd}(x, y) = |p(x, y) - p(x + 2, y)|$ …………………….………… (4.4a)

$g_{rd}(x + 1, y) = |p(x + 1, y) - p(x + 3, y)|$ ……………………….. (4.4b)

$g_{rd}(x, y + 1) = |p(x, y + 2) - p(x + 2, y + 2)|$ ………….............. (4.4c)

$g_{rd}(x + 1, y + 1) = |p(x + 1, y + 2) - p(x + 3, y + 2)|$ ………….... (4.4d)

The gradient is angular if it is almost equal or equal to zero [18]. The intensity gradient is applied in four directions namely horizontal, vertical, right-down and left-down as shown in figure 4.1a and the pixels positions of a 4x4 block are shown in figure 4.1b:

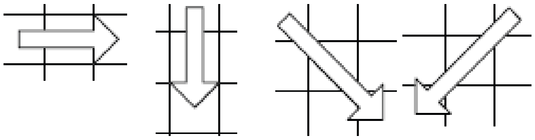


Figure 4.1a : Directions of the intensity gradients.

(x, y)	(x+1, y)	(x+2, y)	(x+3, y)
(x, y+1)	(x+1, y+1)	(x+2, y+1)	(x+3, y+1)
(x, y+2)	(x+1, y+2)	(x+2, y+2)	(x+3, y+2)
(x, y+3)	(x+1, y+3)	(x+2, y+3)	(x+3, y+3)

Figure 4.1b: The positions of the pixels considered in the gradient calculation in a 4x4 LCU.

All four directional gradients are derived by using above equations for each 4x4 LCU. After that sum of absolute gradient for any PU size at all four directions is calculated using the euation 4.5 [19]:

$$SAG = \Sigma \mid g(x, y) \mid \text{.....................}(4.5)$$

To determine the actual texture complexity, normal direction gradient is introduced which determines the actual texture strength. SAG of texture orientation is SAG_{min} and that of the orthogonal orientation is $SAG_{ort,}$ and complexity is calculated as a difference of these values and a threshold (T), where T is QP times PU size (PU_{size}) as shown in equations 4.6 and 4.7 [19]:

$$Complexity = SAG_{ort} - SAG_{min} \text{ .... } (4.6)$$

$$T = QP \times PU_{size} \text{ ....}(4.7)$$

If complexity is less than threshold then PU is homogeneous. If CU has more PUs and they are also homogeneous then they are merged together. A CU is thus divided into number of PUs based on equations (4.1) thru (4.7) (fig 4.2). [19]

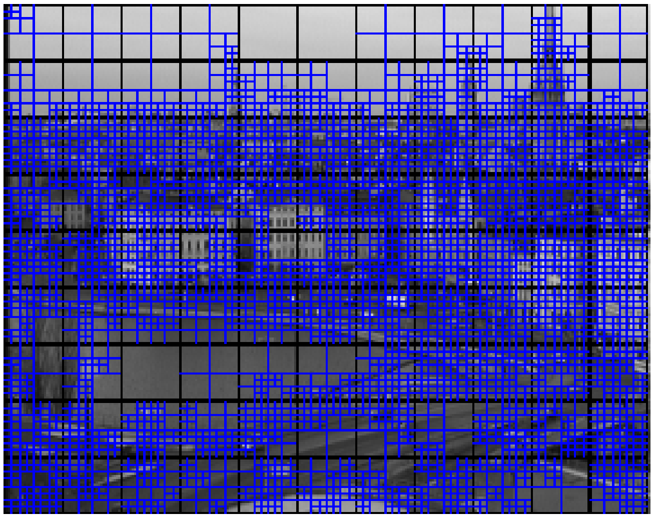


Figure 4.2: CU division into PUs based on homogeneousness of the image [19].

Step 3: To calculate the best mode SATD, from the SATD modes consider three neighboring directions for regular texture PU areas or two non-neighboring directions for non-homogeneous texture PU are used in the RDO process. DC and planar modes are also considered to calculate RDO for smoothing areas. If PU size is 4x4 or 8x8 go to step 4 else go to step.

Step 4: Based on the PU size the number of modes are selected. If the PU size is 4x4 or 8x8 there are 8 modes by default. In this technique the first two modes i.e., the first angle mode (FAM) and the second angle mode (SAM) are compared, If FAM and SAM are directional neighbors (not more 2 modes apart) then candidate list is made of FAM, SAM and third angle mode (TAM). Then go to step 9 else go to step 5.

Step 5: If FAM and SAM are non-neighbors then the candidate list is made of FAM and SAM. Then go to step 7.

Step 6: If the PU size is 16x16 or 32x32, then number of modes by default are only 3 and all the modes are taken into list if no DC or planar mode then go to 7 else go to step 8.

Step 7: Check if DC or planar mode exists in first ‘l’ SATD medium modes (l is 4 in case of 4x4/8x8 and 3 for others), if yes then add them to candidate list but if the PU is 4x4 or 8x8 PU go to step 9 else go to step 8.

Step 8: Add the MPM to the list.

Step9: With this reduced number of PU sizes and prediction modes the total cost is calculated.

To increase the efficiency and reduce the complexity, the process of RQT splitting is simplified in addition to optimized fast intra coding. The RQT depth maximum is set to one in this technique. The block diagram of the fast intra prediction is shown in the figure 4.3.

4.2 The Block Diagram of the Fast Intra Coding Process is Shown in Figure 4.3.

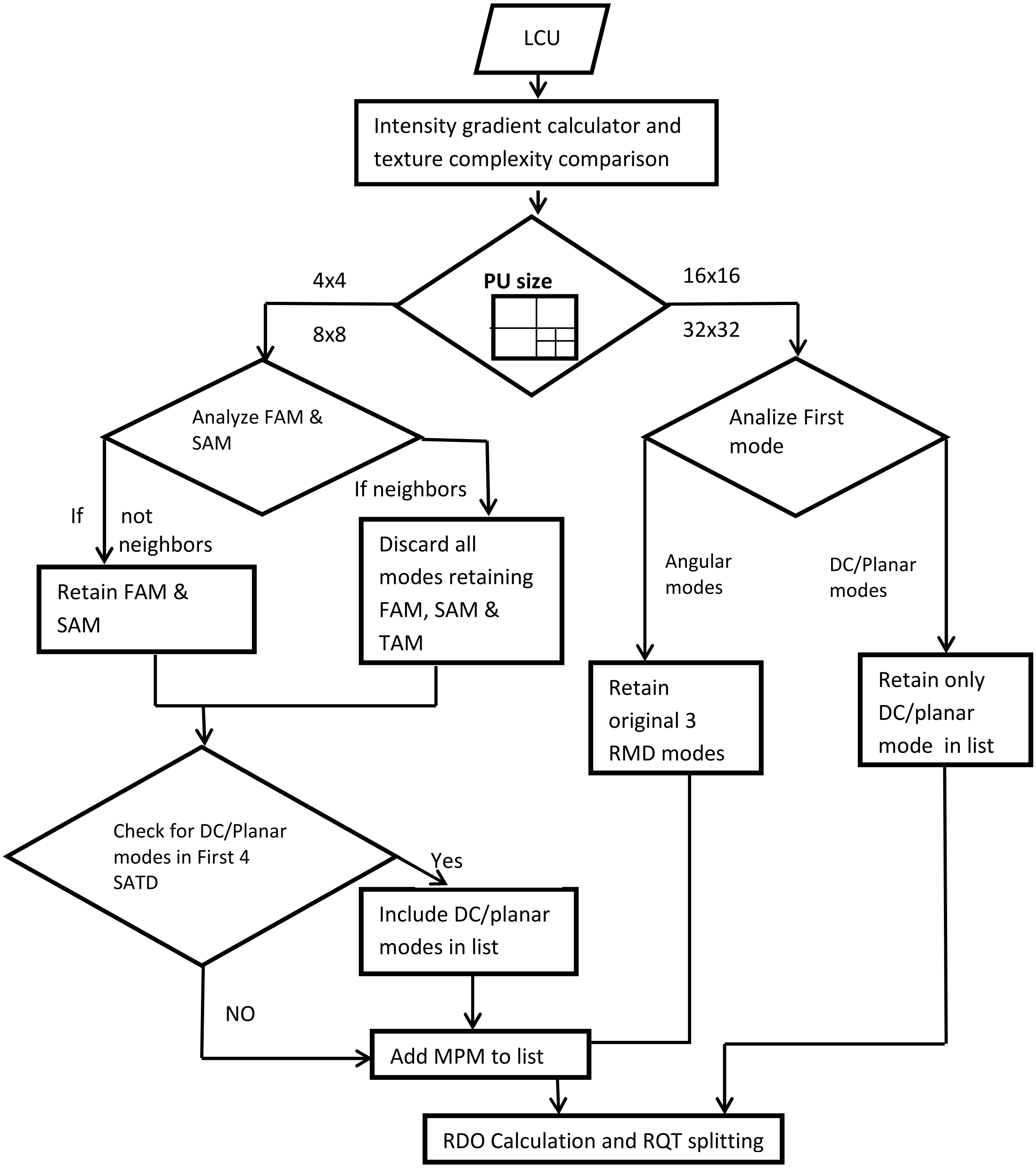

LCU
Intensity gradient calculator and texture complexity comparison
PU size
4x4
8x8
16x16
32x32
Analyze FAM & SAM
If neighbors
If not neighbors
Retain FAM & SAM
Discard all modes retaining FAM, SAM & TAM
Analize First mode
Angular modes
DC/Planar modes
Retain original 3 RMD modes
Retain only DC/planar mode in list
Check for DC/Planar modes in First 4 SATD
Yes
NO
Include DC/planar modes in list
Add MPM to list
RDO Calculation and RQT splitting

Figure 4.3: The fast intra coding technique developed in this thesis

Summary:

The optimization i.e., the complexity reduction process of PU size prediction and prediction mode are explained in detail. The next chapter analyzes the experimental results followed by the conclusions and scope for future work.

CHAPTER 5

IMPLEMENTATION AND RESULTS

In January 2013 the HEVC standard was released with three profiles: Main, Main 10, and Main Still Picture and later in August 2013 draft five additional profiles: Main 12, Main 4:2:2 10, Main 4:2:2 12, Main 4:4:4 10, and Main 4:4:4 12 [21] were also released. In this thesis, the HEVC software used in coding the fast algorithm for intra coding is HM 9.0 [22] and the H.265 software used for comparison is JM 10.0 in visual studio 10.0 [23].

The configuration of the system used is Windows 7.0 operating system, dual core - Intel i5 processor and the display is 32 inch high definition screen.

5.1 Quality Measurement Metrics

In order to actually evaluate and compare video sequences, measurement of quality is critical. Visual quality is mostly subjective and very difficult to measure quantitatively. To get repeatable results some metrics are used which help in determining the quality of the video. Peak signal to noise ratio (PSNR) and bit rate (BD rate) are used [22].

5.1.1. PSNR

PSNR is logarithmic value which depends on the mean squared error (MSE) of a MxN size frame, between the original (x(m, n)) and lossy reconstructed image (y(m, n)), where (m, n) represent coordinates of the image frame considered, relative to the square of the highest-possible value in the image i.e., $(2^b -1)^2$, where b is the number of bits per image sample. $PSNR_{diff}$ [25] is the difference between the fast mode intra coded PSNR ($PSNR_{fast}$) and the original PSNR ($PSNR_{ori}$) [24], [25].

$$PSNR_{dB} = 10 \log_{10} ((2^b - 1)^2 / MSE)\text{………………………. } (5.1)$$

$$MSE = \frac{1}{M * N} \sum_{m=1}^{M} \sum_{n=1}^{N} \left[x(m,n) - y(m,n) \right]^2 \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(5.2)}$$

$$PSNR_{diff} = PSNR_{fast} - PSNR_{ori} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.\text{(5.3)}$$

where x (m,n) and y (m,n) are original and decoded pels. Size of each frame is MxN. The encoder output readily gives the PSNR values. Also DB-PSNR change is calculated based on appendix B.

5.1.2 Bit rate

The quality of a video file is measured by the bit rate in a similar way that resolution is used to measure quality of image file. Bit rate is the number of bits used per unit of playback time representing continuous medium data compression. As bit size used in HEVC is 8 bits, the encoding bit rate of a sequence is the size of the sequence in bytes divided by the playback time of the recording (in seconds), multiplied by eight. Bit rate% is ratio of difference between fast intra mode and original bit rate and original bit rate multiplied by 100 [26].

Bit rate = ((File size in bytes / playback time in secs) x 8) bits/sec…………….(5.4)

Also DB - bit rate gain is calculated based on appendix B.

5.1.3 Percentage Increase or Decrease in Time (Δ T_{Enc})

It is the ratio of difference between fast intra mode encoding and original encoding time and original bit rate multiplied by 100 [26].

$$\Delta\, T_{Enc} = \frac{}{} \qquad \ldots\ldots\ldots\ldots.\textit{(5.5)}$$

5.2 Test Sequences

Four test sequences [28], [29] each belonging to a different class are selected based on the complexity and size. The test sequences are shown in table 5.1.

Table 5.1 Test sequences [28], [29].

Test sequence	Size	Class
Traffic	2560x1600	Class A
Parkscene	1920x1080	Class B
RaceHorses	832x480	Class C
BQSquare	416x240	Class D

5.3 Experimental Results

The encoding time gain, loss in PSNR and bit rate increase of the proposed fast intra coding algorithm HEVC with original HEVC are computed.

Table 5.2 The encoding time gain, loss in PSNR and bit rate increase of the proposed fast intra coding HEVC with original HEVC

Test sequence	Size	Class	Δ T$_{Enc}$ Time gain %	Bit Rate% increase	PSNR (dB) loss
Traffic	2560x1600	Class A	51	-0.78	-0.072
Parkscene	1920x1080	Class B	52	-0.67	-0.098
RaceHorses	832x480	Class C	39	-1.78	-0.085
BQSquare	416x240	Class D	47	-2.26	-0.12
Average Gain			47.25	-1.37	-0.09375

The average encoding time gain is calculated for the proposed HEVC technique for all the test sequences and the same is shown in figure 5.1.

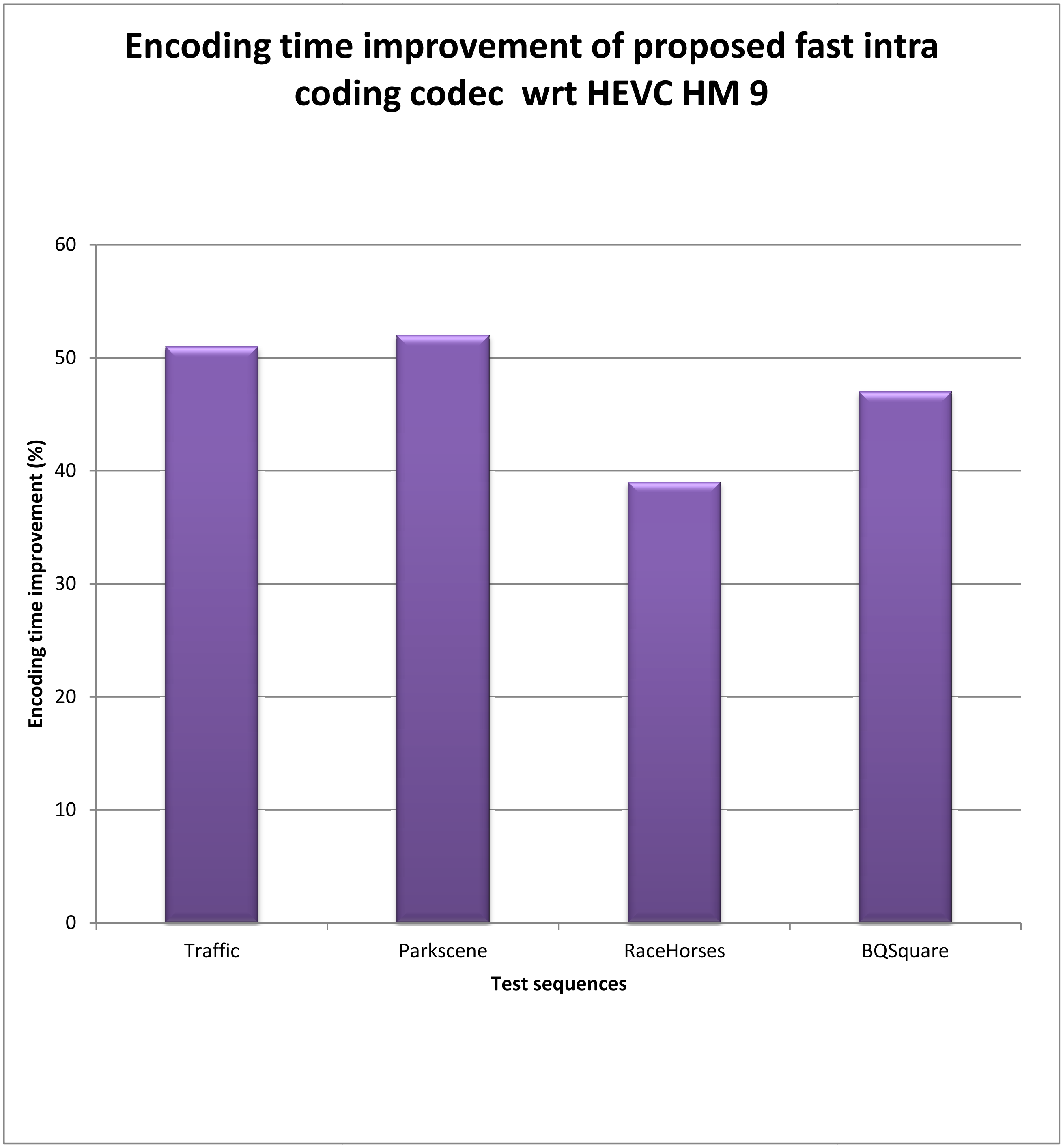


Figure 5.1: Encoding time gain

5.4 Comparison of Proposed HEVC With Respect to Original HEVC and H.264

Comparison of proposed HEVC with respect to original HEVC and H.264 is shown for four test sequences for 30 frames at quantization parameter values 20, 24, 30 and 40. The PSNR and bit rate values for proposed HEVC, original HEVC and H.264 are shown in tables 5.3, 5.4, 5.5 ad 5.6 and corresponding graphs are shown in figures 5.2, 5.3 , 5.4 and 5.5. Typical values for the PSNR for 8 bit data lie between 30 and 50 dB. For 16 bit data typical PSNR values lie between 60 and 80 dB. The higher the PSNR value the better the quality [35].

Table 5.3: PSNR and bit rate values for proposed HEVC, original HEVC and H.264 for Traffic test sequence.

QP	Proposed HEVC- Bit rate (kbps)	Proposed HEVC- PSNR(dB)	Original HEVC - Bit rate (kbps)	Original HEVC - PSNR(dB)	H.264 - Bit rate (kbps)	H.264 - PSNR(dB)
40	36.87	37.73	36.59	37.80	53.79	37.40
30	61.79	38.92	61.29	39.00	91.93	38.50
24	79.80	39.94	79.28	40.00	121.30	39.00
20	136.59	42.03	135.36	42.11	197.63	41.90

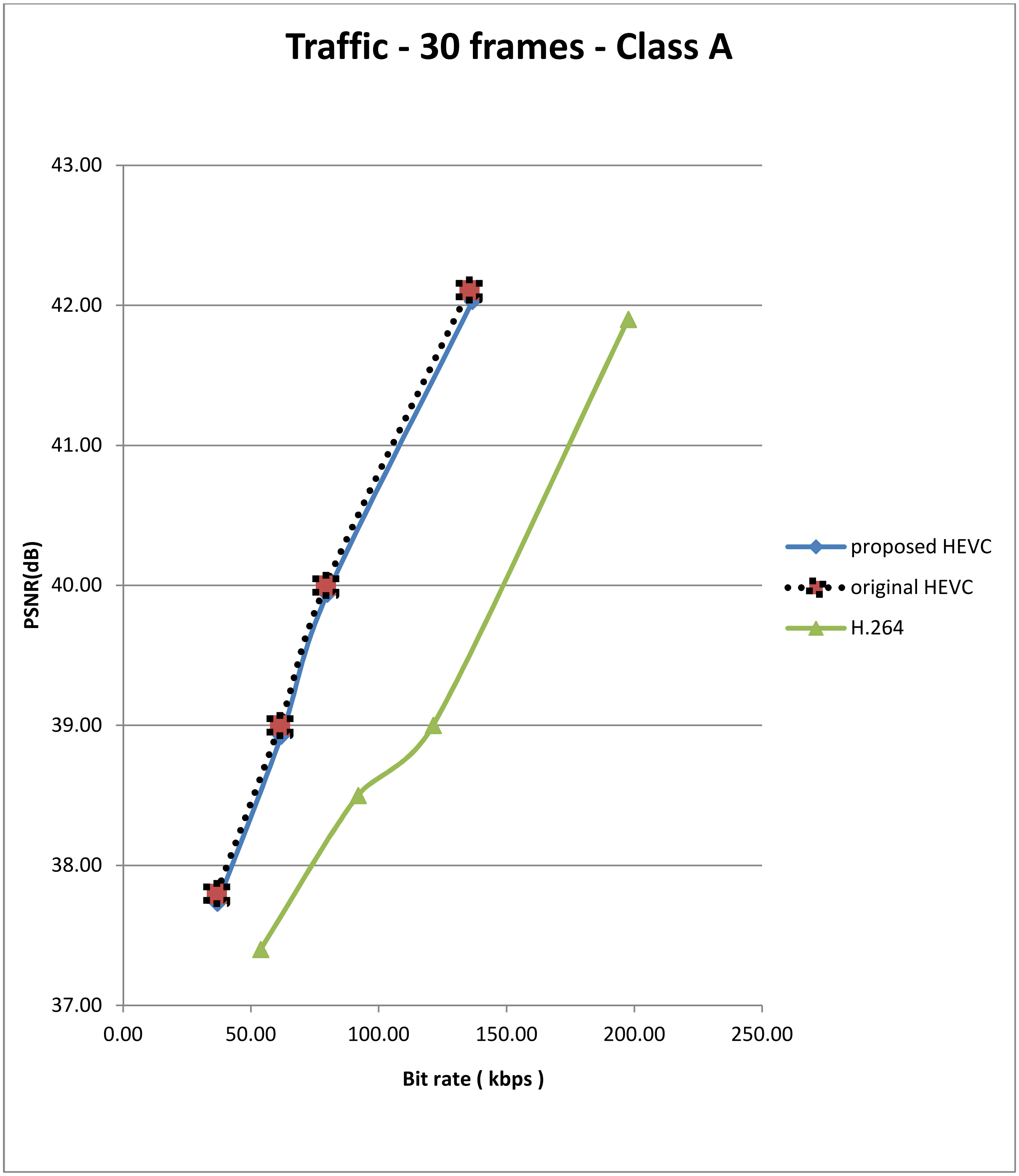


Figure 5.2: Bit rate (kbps) vs. PSNR (dB) of proposed HEVC, original HEVC and H.264 for Traffic test sequence.

Table 5.4: PSNR and bit rate values for proposed HEVC, original HEVC and H.264 for Parkscene test sequence

QP	Proposed HEVC-Bit rate (kbps)	Proposed HEVC-PSNR(dB)	Original HEVC - Bit rate (kbps)	Original HEVC - PSNR(dB)	H.264 - Bit rate (kbps)	H.264 - PSNR(dB)
40	24.47	38.47	24.30	38.57	35.72	38.10
30	54.90	39.90	54.57	40.00	81.85	39.00
24	73.31	41.42	72.87	41.50	111.49	41.10
20	120.53	44.99	119.56	45.10	174.56	44.70

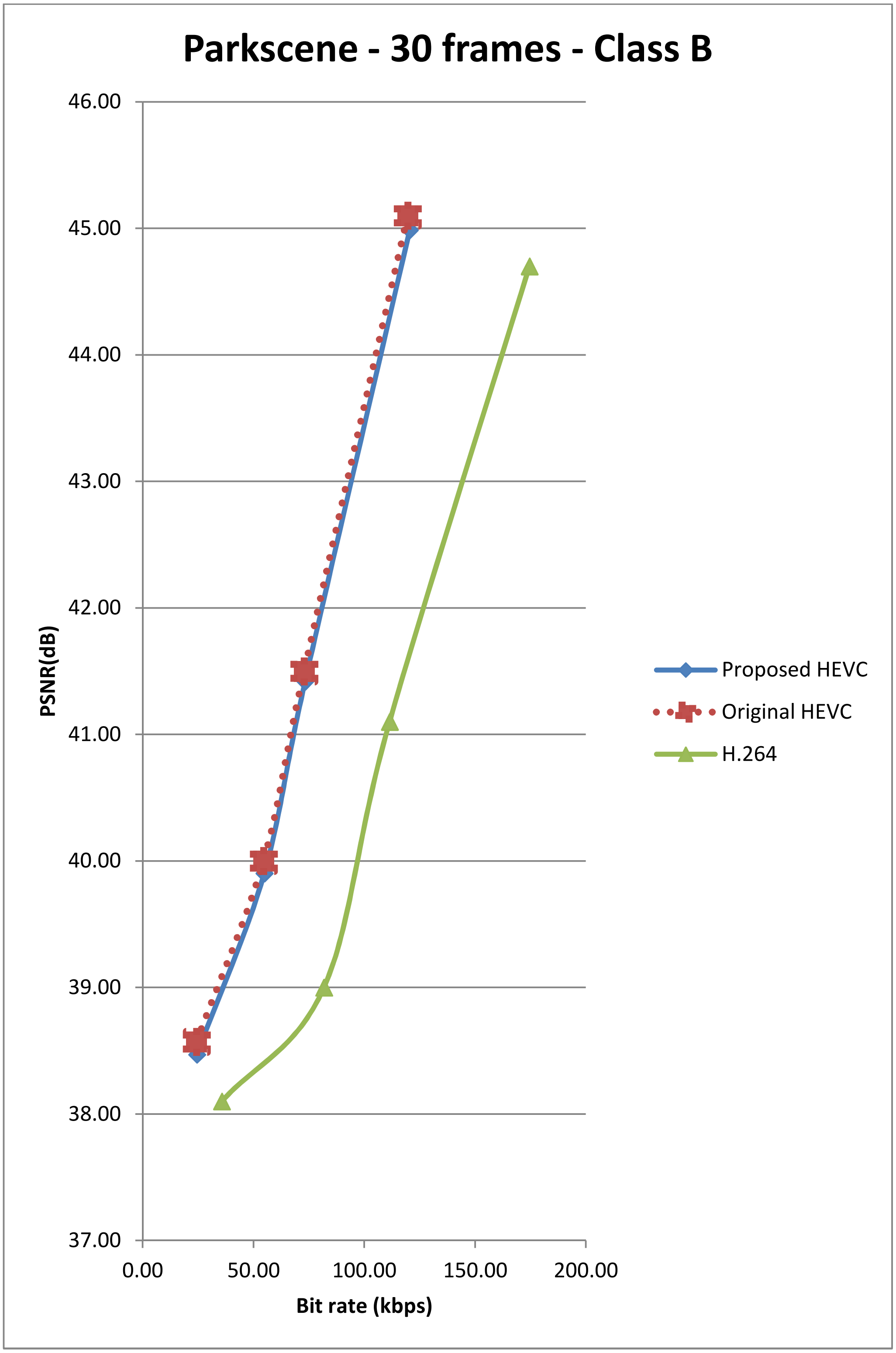


Figure 5.3: Bit rate (kbps) vs. PSNR (dB) of proposed HEVC, original HEVC and H.264 for test sequence Parkscene.

Table 5.5: PSNR and bit rate values for proposed HEVC, original HEVC and H.264 for Racehorses test sequence

QP	Proposed HEVC- Bit rate (kbps)	Proposed HEVC- PSNR(dB)	Original HEVC - Bit rate (kbps)	Original HEVC - PSNR(dB)	H.264 - Bit rate (kbps)	H.264 - PSNR(dB)
40	22.63	36.50	22.18	36.59	32.61	34.00
30	35.11	37.71	34.56	37.80	51.84	34.70
24	63.36	39.82	62.23	39.90	95.21	36.00
20	101.90	43.02	100.11	43.10	146.15	42.90

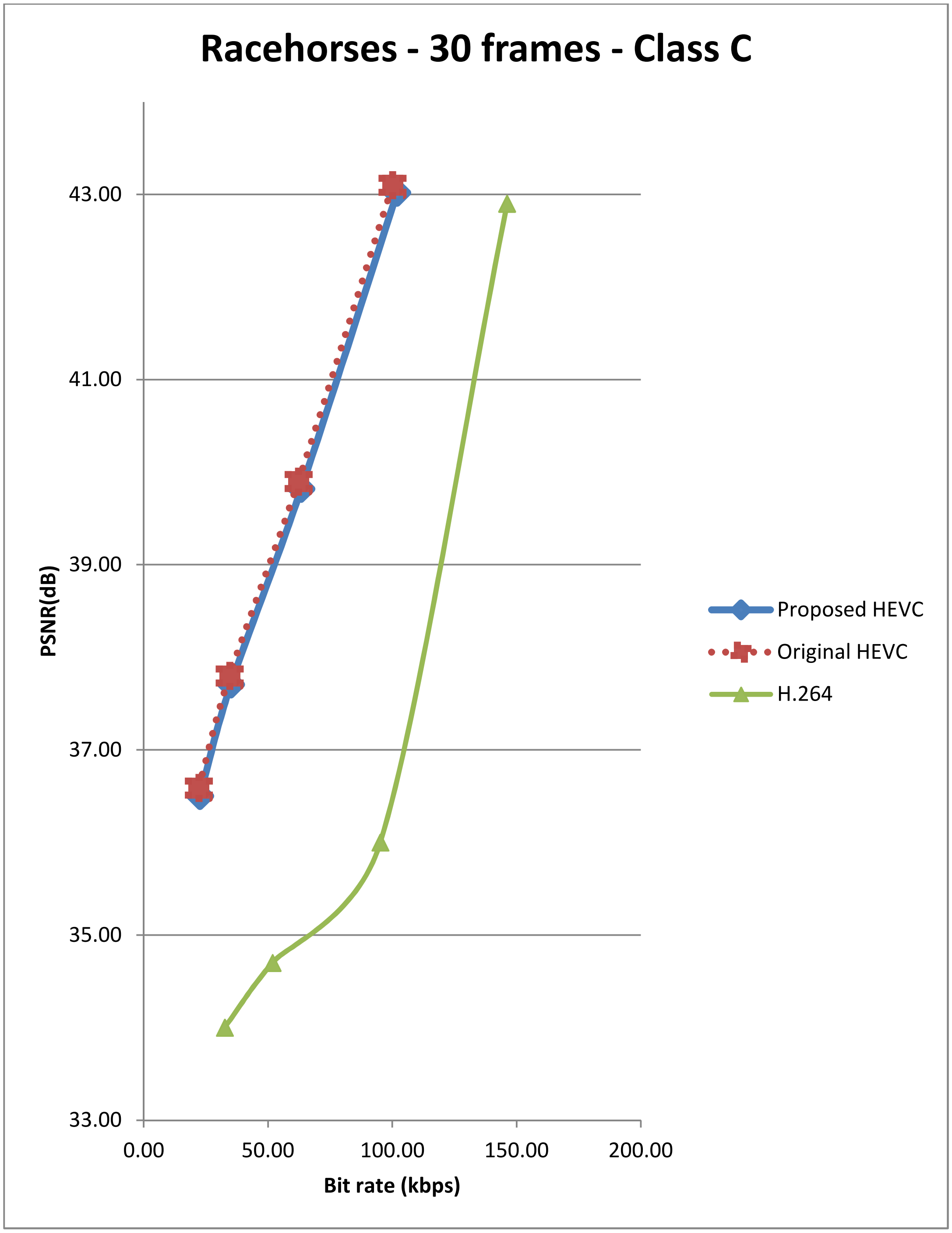


Figure 5.4: Bit rate (kbps) vs. PSNR (dB) of proposed HEVC, original HEVC and H.264 for test sequence Racehorses.

Table 5.6: PSNR and bit rate values for proposed HEVC, original HEVC and H.264 for BQ Square test sequence.

QP	Proposed HEVC-Bit rate (kbps)	Proposed HEVC-PSNR(dB)	Original HEVC - Bit rate (kbps)	Original HEVC - PSNR(dB)	H.264 - Bit rate (kbps)	H.264 - PSNR(dB)
40	5.29	34.57	5.17	34.67	7.93	34.20
30	7.47	37.05	7.28	37.23	11.13	37.00
24	10.09	40.01	9.88	40.10	13.11	39.20
20	13.19	43.40	12.95	43.50	18.47	42.90

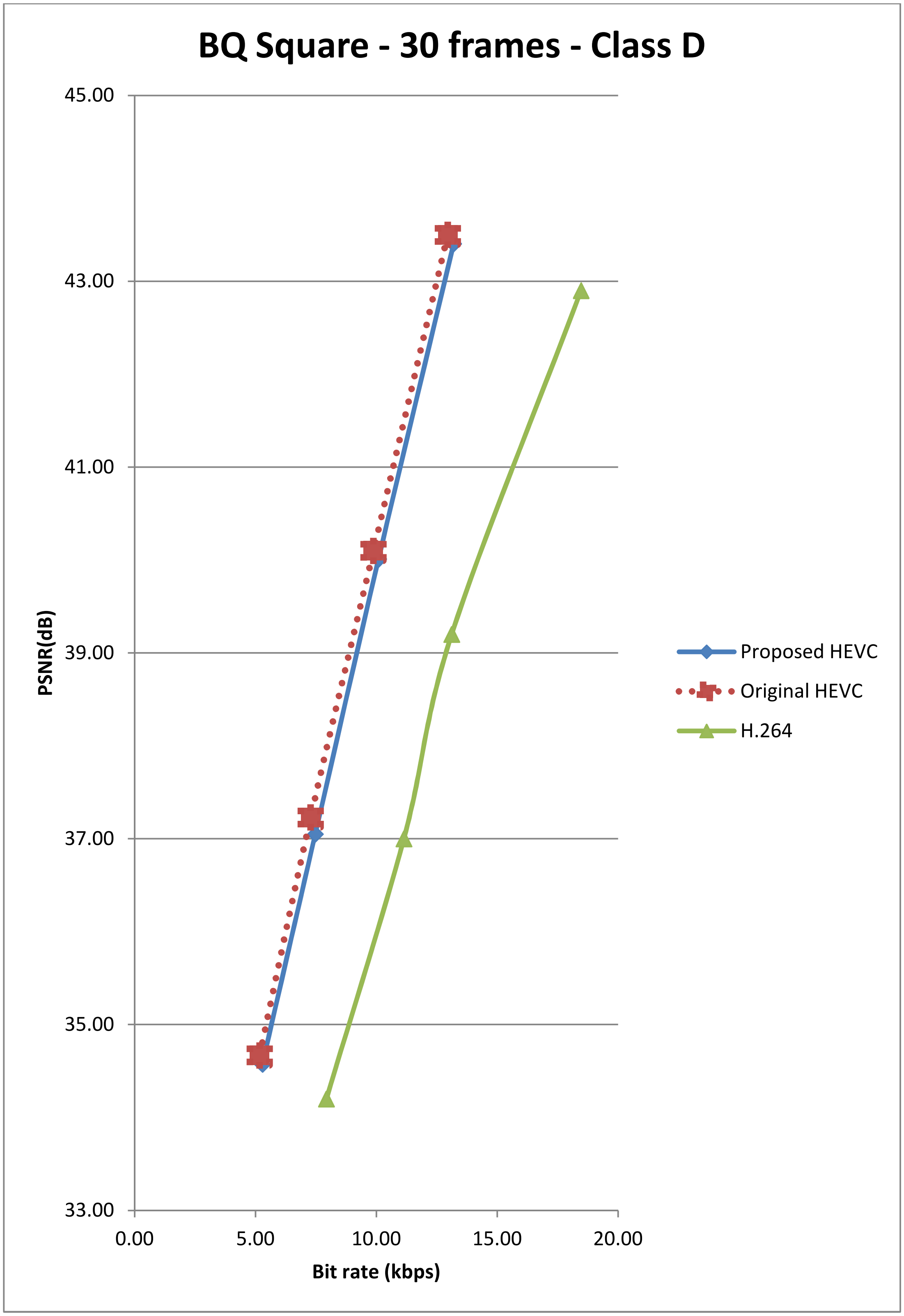


Figure 5.5: Bit rate (kbps) vs. PSNR (dB) for proposed HEVC, HEVC and original H.264 for test sequence BQ Squares.

5.5 BD-bit rate Gain and BD- PSNR Loss.

The BD-bit rate gain for all the test sequences is calculated by the code given in appendix B [34]. The graphs of BD-bit rate gain over QP values 20, 24, 30 and 40 are shown in figures 5.6 thru 5.9 and there is a drop in the PSNR using BD metrics as shown in figures 5.10 thru 5.13. Ideally, BD-PSNR should increase and BD-bit rate should decrease [32].

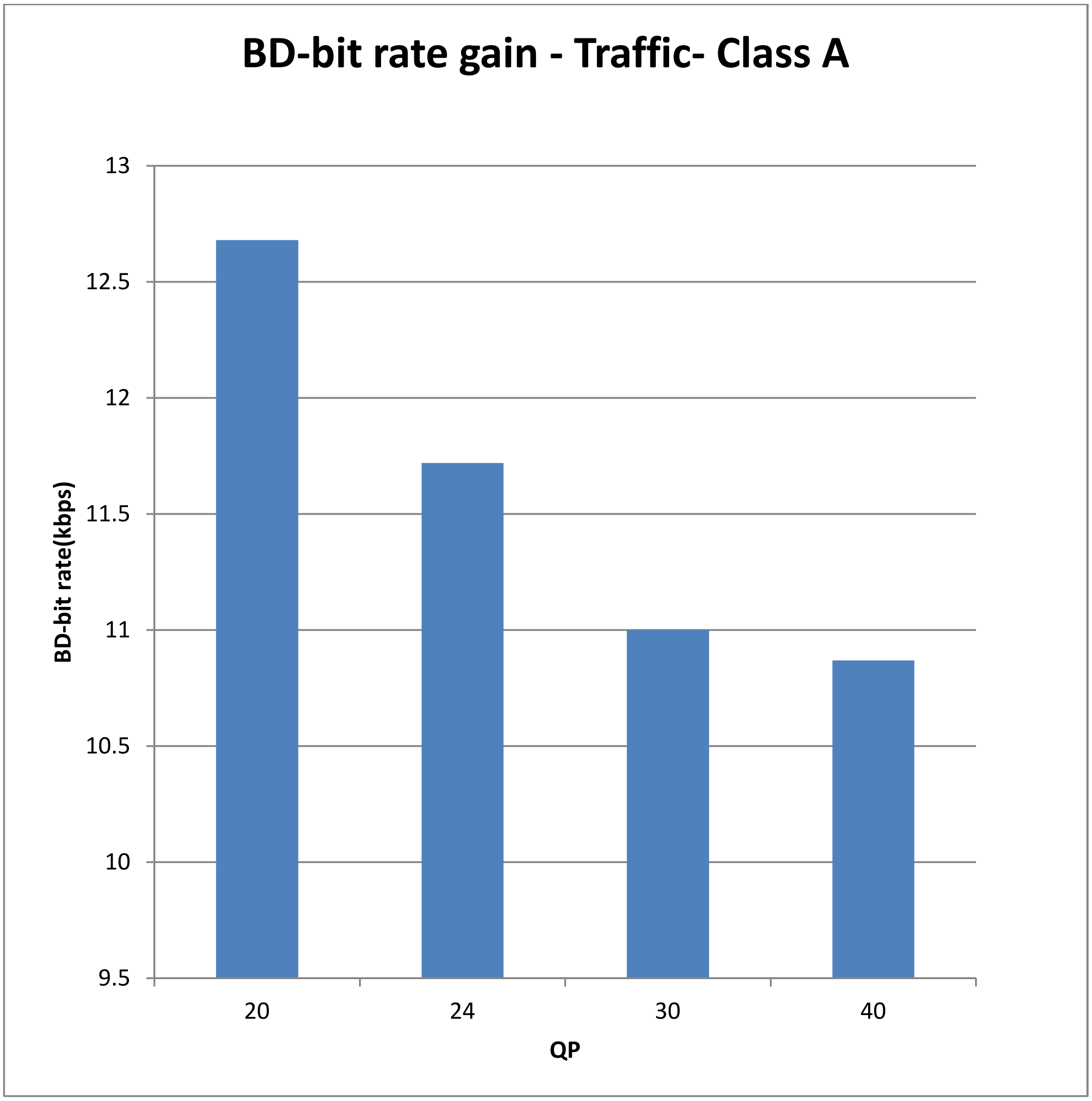


Figure 5.6: QP vs BD-bit rate (kbps) between proposed HEVC and original HEVC for test sequence Traffic.

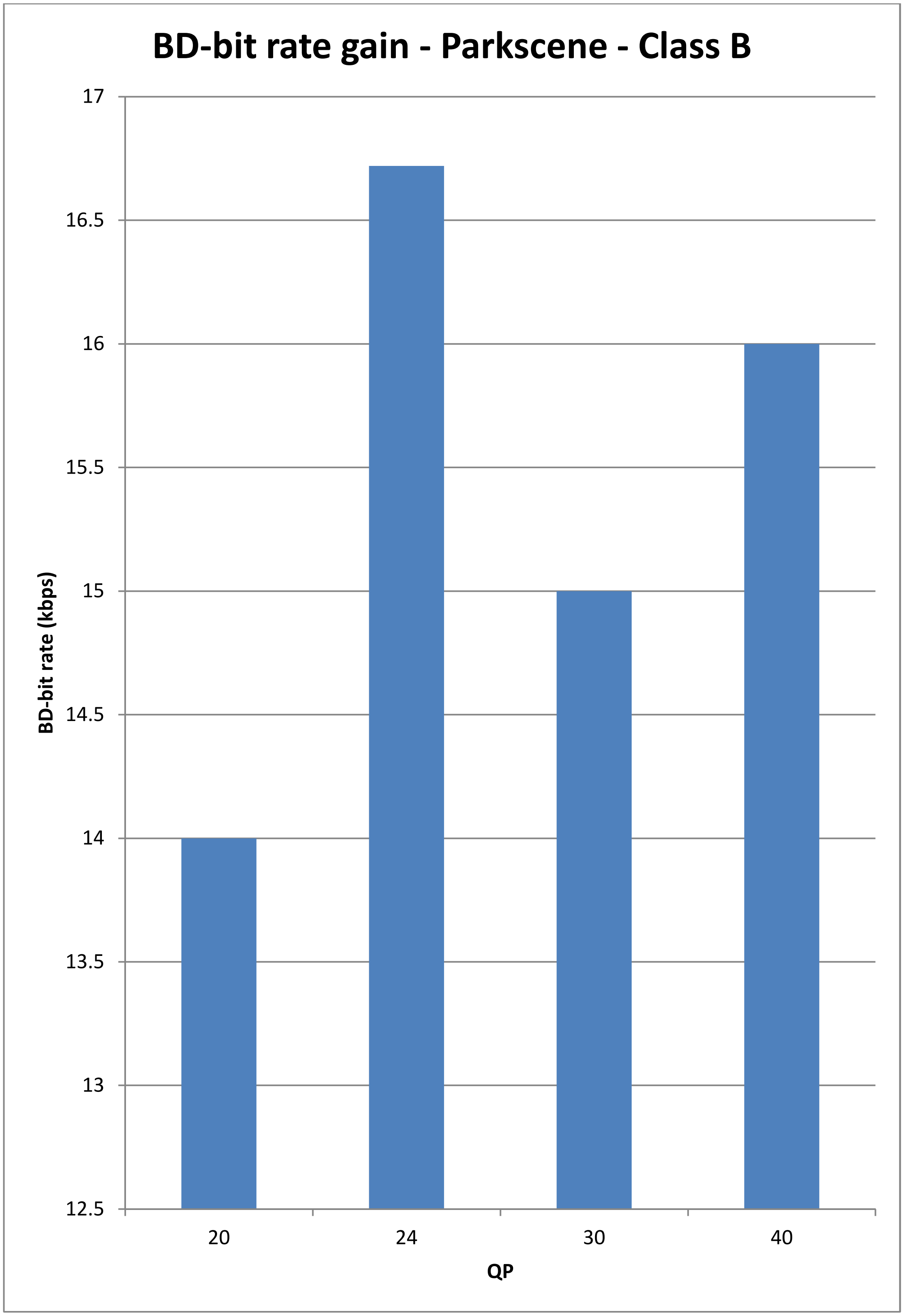


Figure 5.7: QP vs BD-bit rate (kbps) between proposed HEVC and original HEVC for test sequence Parkscene.

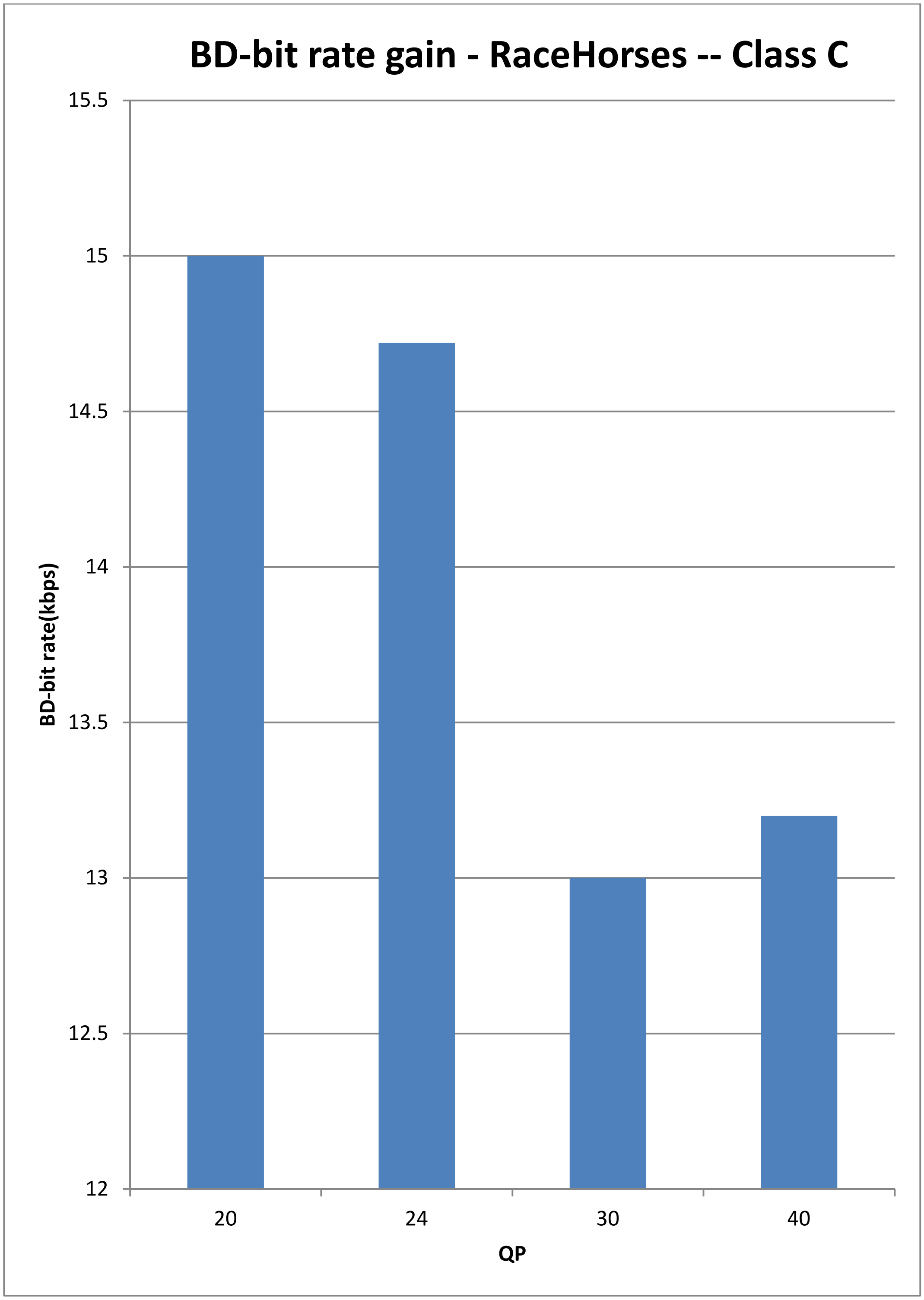


Figure 5.8: QP vs BD-bit rate (kbps) between proposed HEVC and original HEVC for test sequence RaceHorses.

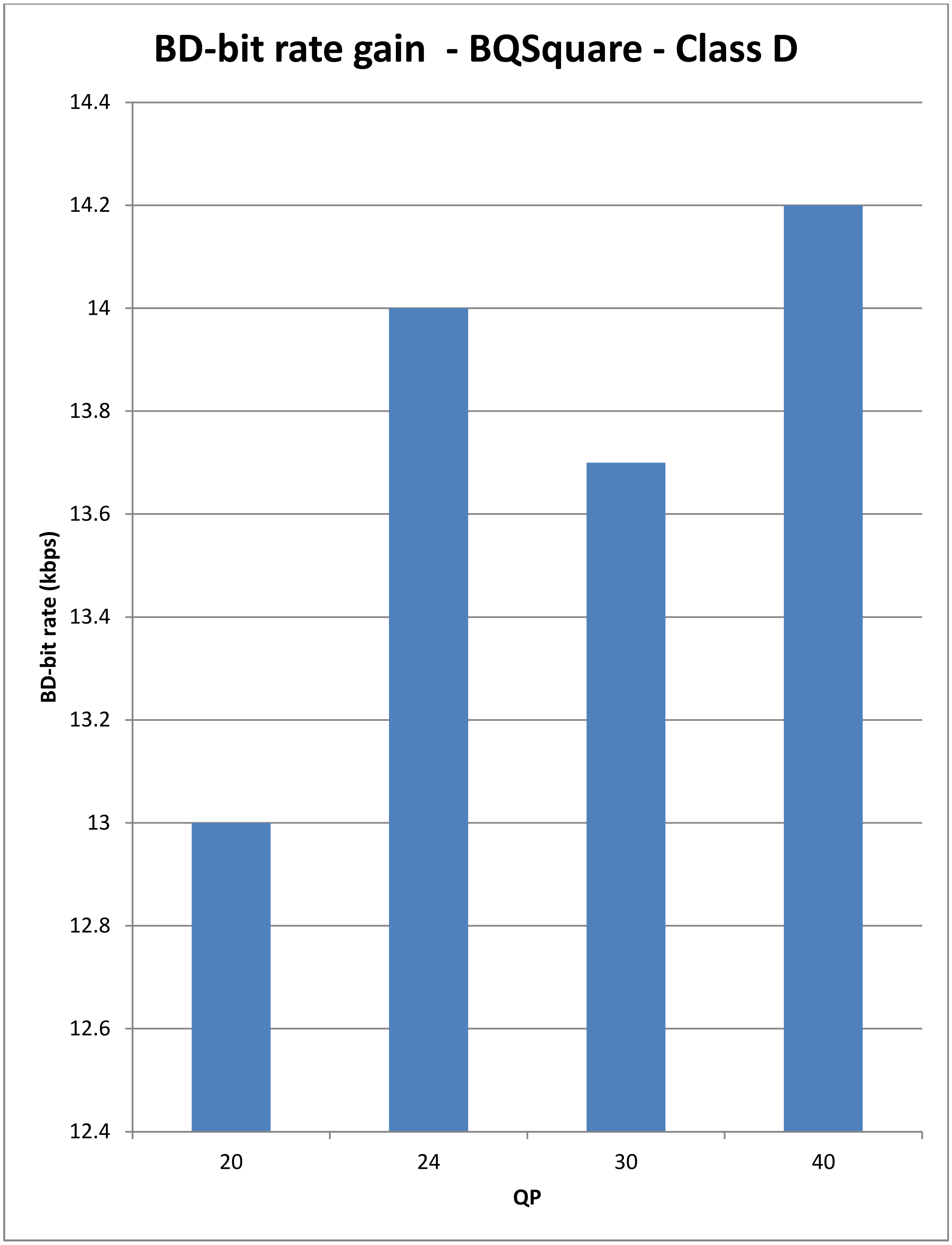


Figure 5.9: QP vs BD-bit rate (kbps) between proposed HEVC and original HEVC for test sequence BQSquare.

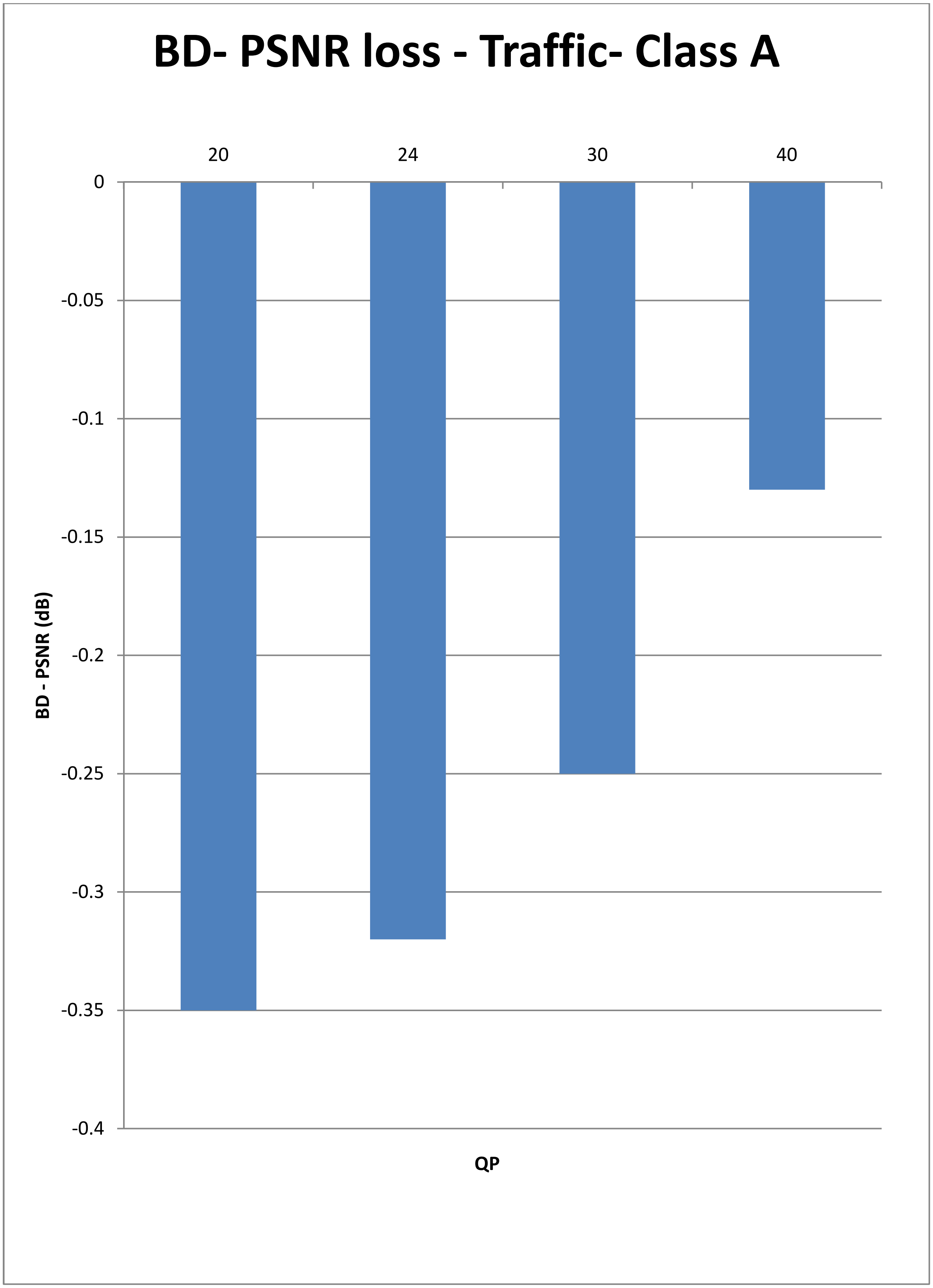


Figure 5.10: BD-PSNR vs. quantization parameter for test sequence Traffic.

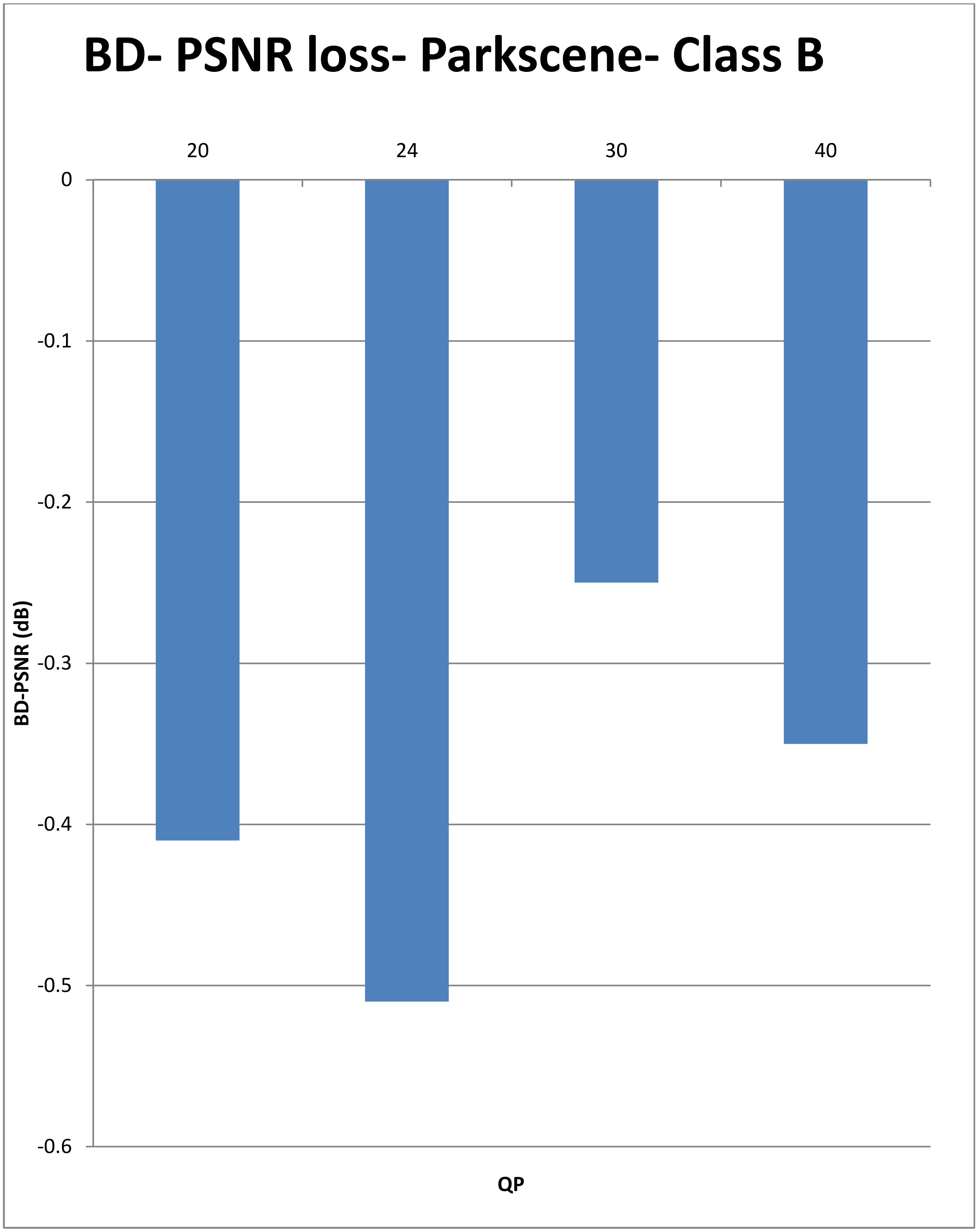


Figure 5.11: BD-PSNR vs. quantization parameter for test sequence Parkscene.

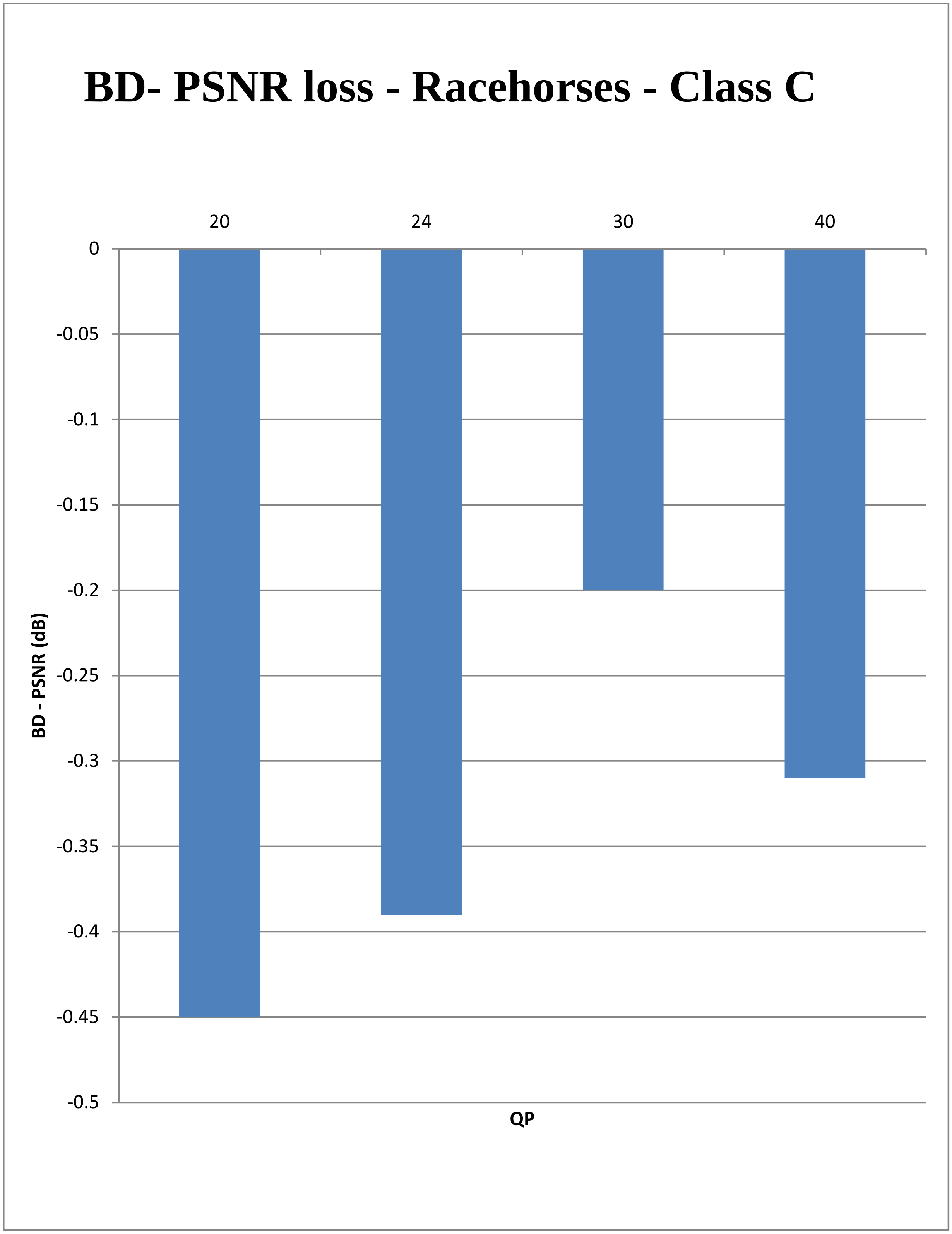


Figure 5.12: BD-PSNR vs. quantization parameter for test sequence Racehorses.

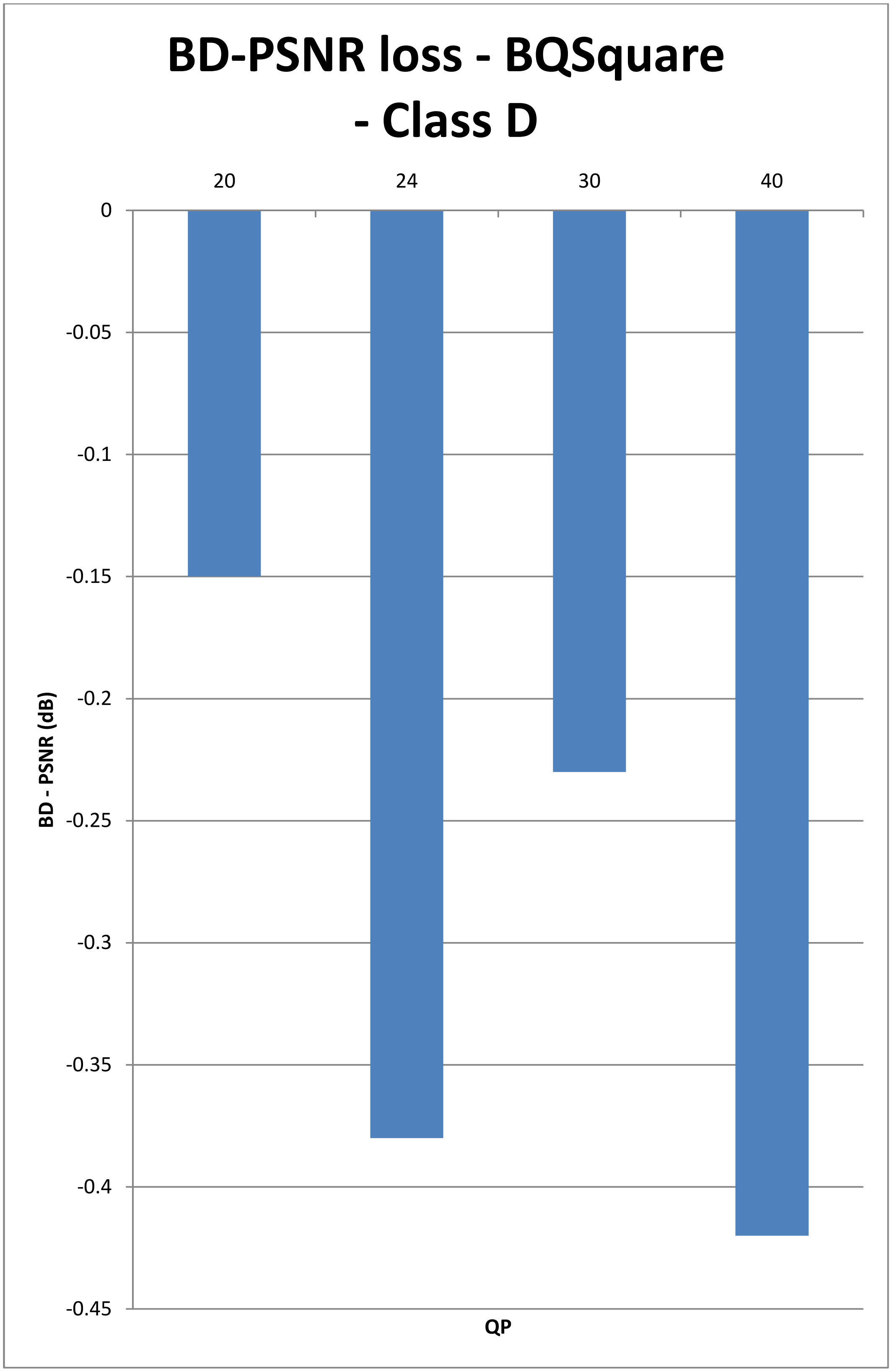


Figure 5.13: BD-PSNR vs. quantization parameter for test sequence BQ Square.

Summary

The average gain in encoding time, PSNR vs bit rate graphs, BD-bit rate gain vs QP and BD – PSNR loss vs QP are shown in this chapter and the conclusions are discussed in the following chapter.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusion

The proposed technique reduced the encoding time alone at the cost of slight increase in bit rate and negligible PSNR loss. There is an average gain of 47.25% encoder time at very less loss in performance with high complexity reduction. The average increase in bit rate is only 1.37 % and average loss in PSNR is only 0.93 dB. In comparison with H.264 [1] standard the average encoding time is larger by only 10%. Hence this research obtains half the bit rate at almost the same complexity as that of H.264 at almost the same quality.

6.2 Applications

The proposed technique reduces the encoding time significantly. The encoding time at the transmitter end is very critical in practical system and operations like scaling (mobile applications), video uploading and updating storage systems which require movement of large video files etc.. This will in turn reduce over-all time consumed in processing the videos saving many man hours.

In transmission and general consumer applications like TVs, internet video, and educational websites with video lectures etc., the critical aspect is the speed of download or upload. This greatly reduces the encoding time which reduces the time required to compress the raw videos.

6.3 Scope for Future Work

The complexity reduction in the proposed technique optimizes intra coding of HEVC, resulting in complexity reduction - inter coding of HEVC can also be optimized. The presented technique can be combined with other fast intra/inter prediction techniques to achieve better bit rate and to improve PSNR.

Wang et al [36] present a technique which studies multiple sign bit hiding scheme in HEVC. This technique designs quantization of transform coefficients coding by data hiding approach achieving good RDO resulting in overall coding gain in HEVC. This method if combined with the proposed thesis can give very good optimization.

Vasudevan [31] has adopted a fast intra coding technique in combination with a fast RQT optimization method. This method can be applied with the proposed technique to further reduce the encoder complexity.

Future research can be conducted to improve both the intra and inter coding to obtain much higher reduction of encoding time, better bit rate and PSNR. The aim should be to reduce the overall complexity of HEVC encoder suitable for hand held devices as well as transmission with limited computing resources.

APPENDIX A

TEST SEQUENCES

The test sequences are selected to cover various features, different quality types and statistics of video like high motion activity and high contrast, comparably static background, dynamic, motion-filled video etc. [28], [29].

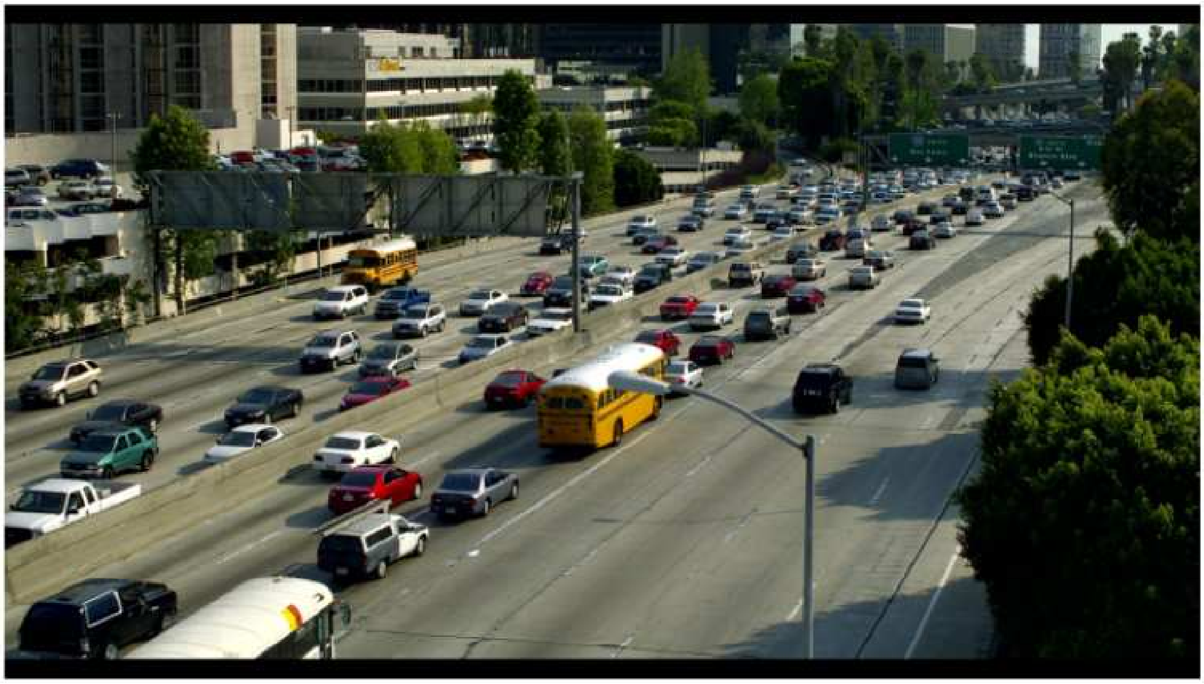


Figure A.1 Traffic 2560x1600

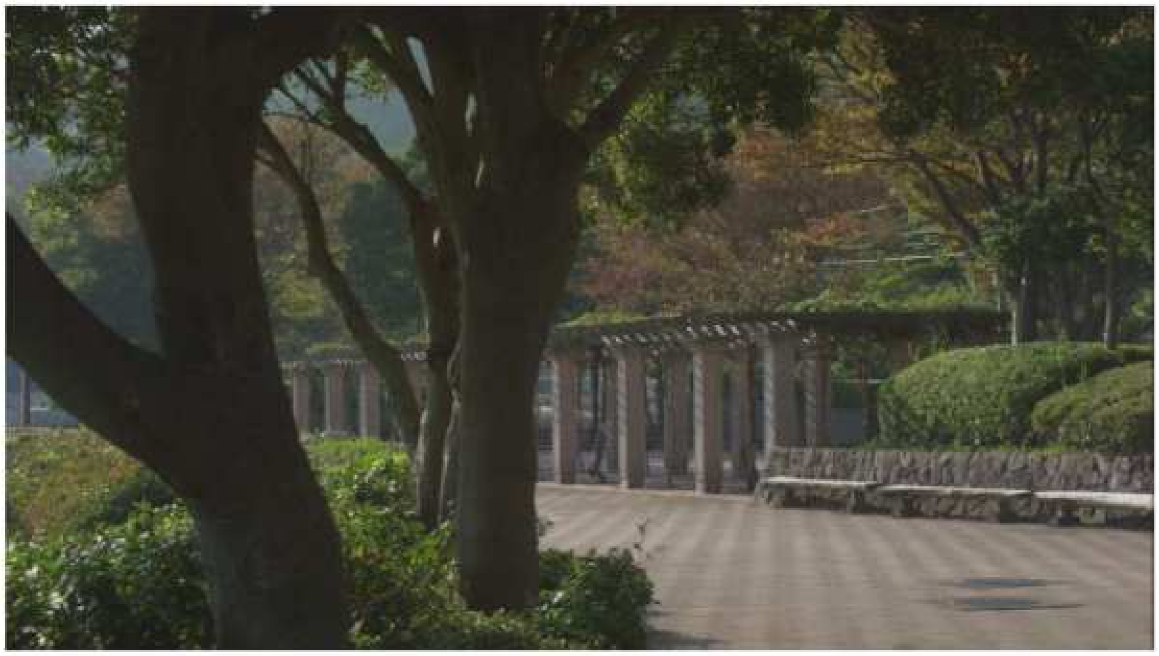


Figure A.2 Park scene 1920x1080

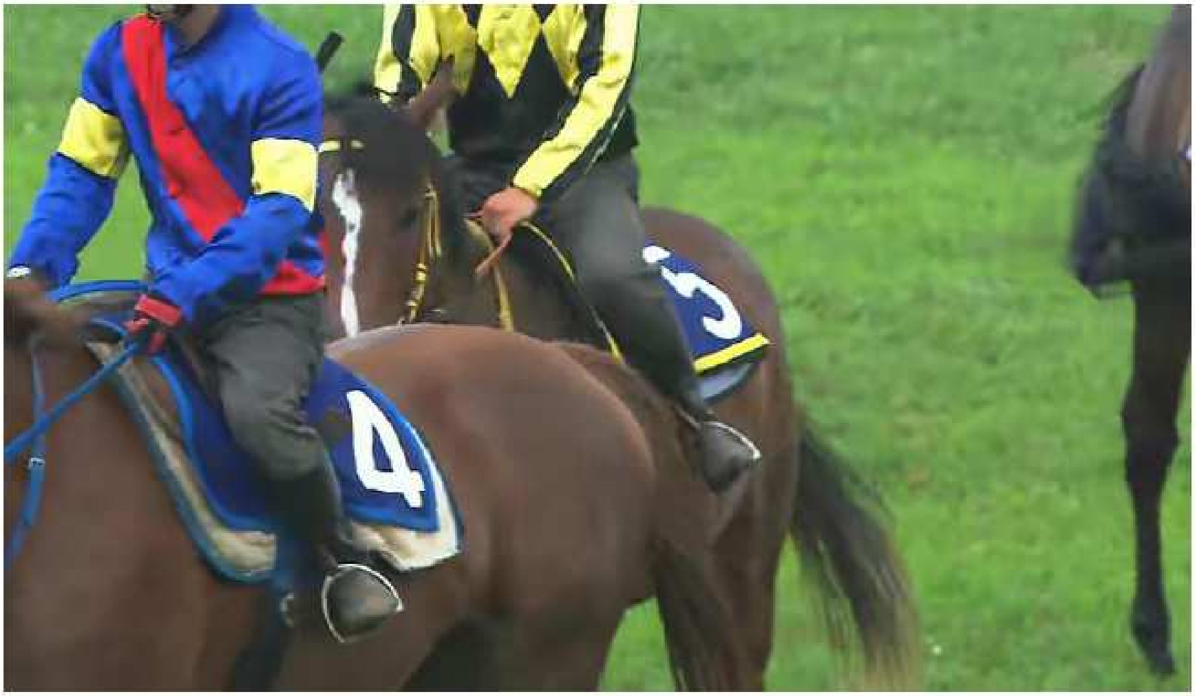


Figure A.3 RaceHorses 832x480

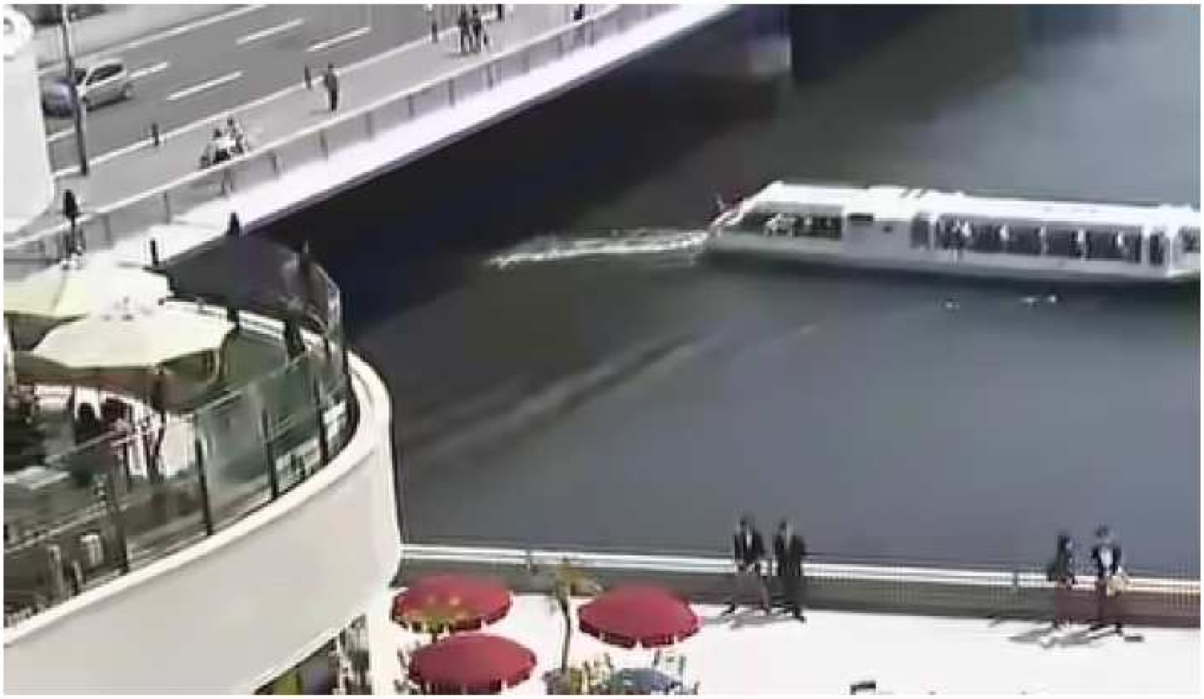


Figure A.4 BQ Square 416x240

APPENDINX B

BD- PSNR AND BD bit rate

BD-PSNR (Bjontegaard – PSNR) and BD-bit rate (Bjontegaard – bit rate) metrics are used to compute the average gain in PSNR and the average per cent saving in bit rate between two rate-distortion graphs respectively and is an ITU-T approved metric [33]. This method was developed by Bjontegaard and is used to gauge compression algorithms from a visual aspect in media industry and referenced by many multimedia engineers. The MATLAB code is available online [34].

```
function avg_diff = bjontegaard(R1,PSNR1,R2,PSNR2,mode)%
%   R1,PSNR1 - RD points for curve 1
%   R2,PSNR2 - RD points for curve 2
%   mode -
%       'dsnr' - average PSNR difference
%       'rate' - percentage of bit rate saving between data set 1 and
%                data set 2
%
%   avg_diff - the calculated Bjontegaard metric ('dsnr' or 'rate')
%
%   (c) 2010 Giuseppe Valenzise
%
%   References:
%
%   [1] G. Bjontegaard, Calculation of average PSNR differences between
%       RD-curves (VCEG-M33)
%   [2] S. Pateux, J. Jung, An excel add-in for computing Bjontegaard metric and
%       its evolution

% convert rates in logarithmic units
lR1 = log(R1);
lR2 = log(R2);

switch lower(mode)
    case 'dsnr'
        % PSNR method
        p1 = polyfit(lR1,PSNR1,3);
        p2 = polyfit(lR2,PSNR2,3);

        % integration interval
        min_int = min([lR1; lR2]);
        max_int = max([lR1; lR2]);
        % find integral
        p_int1 = polyint(p1);
        p_int2 = polyint(p2);

        int1 = polyval(p_int1, max_int) - polyval(p_int1, min_int);
        int2 = polyval(p_int2, max_int) - polyval(p_int2, min_int);

        % find avg diff
        avg_diff = (int2-int1)/(max_int-min_int);

    case 'rate'
        % rate method
        p1 = polyfit(PSNR1,lR1,3);
        p2 = polyfit(PSNR2,lR2,3);

        % integration interval
        min_int = min([PSNR1; PSNR2]);
        max_int = max([PSNR1; PSNR2]);

        % find integral
        p_int1 = polyint(p1);
        p_int2 = polyint(p2);

        int1 = polyval(p_int1, max_int) - polyval(p_int1, min_int);
        int2 = polyval(p_int2, max_int) - polyval(p_int2, min_int);

        % find avg diff
        avg_exp_diff = (int2-int1)/(max_int-min_int);
        avg_diff = (exp(avg_exp_diff)-1)*100;
end
```

REFERENCES

[1] K. Sayood, “Introduction to Data Compression", 4th ed. Morgan Kaufmann, 2012.

[2] Y.Q. Shi , and H. Sun, “Image and video compression for multimedia engineering: fundamentals, algorithms, and standards”, Boca Raton, Fla: CRC Press, 1999.

[3] I. Richardson, “The H.264 Advanced Video Compression Standard", Wiley, 2010.

[4] N. Ling, “High efficiency video coding and its 3D extension: A research perspective," 7th IEEE, Trans. on ICIEA, pp. 2150-2155, July 2012.

[5] K.R. Rao, D.N. Kim and J.J. Hwang, “Video coding standards: AVS China, H.264/MPEG4-Part 10, HEVC, VP6, DIRAC and VC-1”, Springer 2014.

[6] J. Chen, U. Koc, and K. Liu, “Design of Digital Video Coding Systems A Complete Compressed Domain Approach", Marcel Dekker, 2002.

[7] G.J. Sullivan; J. Ohm; W.-J. Han and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard”, IEEE Trans. on CSVT, vol. 22, Issue: 12, pp. 1649-1668, Dec 2012.

[8] F. Bossen, D. Flynn and K. Suhring, “HEVC reference software manual”, July 2011. [Online]. Available:
http://phenix.int-evry.fr/jct/doc_end_user/documents/6_Torino/wg11/JCTVC-F634-v2.zip

[9] B. Bross et al, “High efficiency video coding (HEVC) text specification draft 10”, 12th Meeting Geneva January 2013. [Online]. Available:
http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=7243

[10]F. Bossen, B. Bross, K. Sühring, and D. Flynn, "HEVC Complexity and Implementation Analysis," IEEE Trans. on CSVT, vol.22, no.12, pp.1685-1696, Dec. 2012.

[11] B. Bross et al, “HM9: High Efficiency Video Coding (HEVC) Test Model 9 Encoder Description”, JCTVC-K1002v2, October 2012. [Online]. Available:

http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=6807

[12] B. Bross et al, "High Efficiency Video Coding (HEVC) text specification draft 9 (SoDIS)", JCTVC-K1003v13, October 2012. [Online]. Available:

http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=6803

[13] J.-R. Ohm et al, "Comparison of the coding efficiency of video coding standards – including high efficiency video coding (HEVC)", IEEE Trans. on CSVT, vol. 22, pp.1669-1684, Dec. 2012.

[14] I. Richardson., "HEVC: An introduction to High Efficiency Video Coding", [Online]. Available: https://www.vcodex.com/images/uploaded/452853087451536.pdf

[15] J. Lainema et al, "Intra coding of the HEVC standard", IEEE Trans. on CSVT, vol. 22, pp. 1792-1801, Dec. 2012.

[16] Y. Zhang, C. Zhao and J. Xu, "An adaptive fast intra mode decision in HEVC", IEEE ICIP 2012, pp. 221-224, Orlando, FL, Sept.-Oct., 2012.

[17] Y. Zhang, Z. Li and B. Li, "Gradient-based fast decision for intra prediction in HEVC", IEEE VCIP 2012, pp. 1-6, Nov. 2012.

[18] A-C Tsai, A. Pal, J. Ching Wang, and J-F Wang, "Intensity gradient technique for efficient intra-prediction in h.264/AVC," IEEE Trans. on CSVT, vol. 18, no. 5, pp. 694–698, May. 2008.

[19] W. Y. Ma and B. S. Manjunath, "A texture thesaurus for browsing large aerial photographs," J. Amer. Soc. Inf. Sci.,vol. 49, no. 7, pp. 633–648, May. 1998.

[20] L.-L. Wang and W.-C. Siu."Novel adaptive algorithm for intra prediction with compromised modes skipping and signaling processes in HEVC", IEEE Trans. on CSVT, vol. 23, pp. 1686-1694, Nov. 2013.

[21] D. Flynn, J. Sole, T. Suzuki (2013-08-07). "High Efficiency Video Coding (HEVC) Range Extensions text specification: Draft 4". JCT-VC.Retrieved 2013-08-07.

[22] "HM reference software svn repository." [Online]. Available:

http://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/

[23] JM11 reference software JM11KTA2.3 [Online]. Available:

http://www.h265.net/2009/04/kta-software-jm11kta23.html

[24] G. Correa et al, “Performance and computational complexity assessment of high efficiency video encoders”, IEEE Trans. on CSVT, vol. 22, pp.1899-1909, Dec. 2012.

[25] G. Bjontegaard, “Calculation of average PSNR differences between RD-Curves”, ITU-T SG16, Doc. VCEG-M33, 13th VCEG meeting, Austin, TX, April 2001. [Online]. Available:

http://wfpt3.itu.int/av-arch/video-site/0104_Aus/VCEG-M33.doc

[26] Open source article, “Bit rate”. [Online]. Available: http://en.wikipedia.org/wiki/Bit_rate

[27] Open source article on “video resolution”. [Online]. Available: http://www.newfangled.com/designing_for_changing_web_content_and_context

[28] Repository for freely-redistributable test sequences.[Online] Available: www.media.xiph.org.

[29] “Test video sequences." [Online]. Available:

ftp://hvc:US88Hula@ftp.tnt.uni-hannover.de/testsequences

[30] Open source website “Practical guide to CCTV video resolutions. [Online]. Available: http://optiviewusa.com/cctv-video-resolutions.aspx

[31] S. Vasudevan,“Implementation of fast residual quadtree coding and fast intra prediction in high efficiency video coding”, EE Dept., University of Texas at Arlington, UMI dissertation publishing, Dec. 2013.

[32] Special issue on video coding: HEVC and beyond, IEEE journal of selected topics in signal processing, Dec. 2013.

[33] G. Bjontegaard, “Calculation of average PSNR differences between RD-curves”, Q6/SG16, Video Coding Experts Group (VCEG) ITU-T Standardization Sector, 2-4 April. 2001

[34] BD metrics open source code. [Online]. Available:

http://www.mathworks.com/matlabcentral/fileexchange/27798bjontegaardmetric/content/bjontegaard.m

[35] Open source article on “PSNR”. [Online]. Available:

http://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio#cite_note-8

[36] J. Wang et al, “Multiple sign bits hiding for high efficiency video coding”, Visual Communications and Image Processing, VCIP 2012, San Diego, CA, 27-30, Nov. 2012.

BIOGRAPHICAL INFORMATION

Vinoothna Gajula was born in Chilakaluripet of Guntur, Andhra Pradesh, India. She completed her intermediate (12th grade) in Gowtham junior college in 2005, and completed her B.Tech in Electrical and Electronic Engineering from Bhoj Reddy engineering college for women in 2009. She worked in Areva T&D, circuit breaker factory, as a project management engineer from July 2009 to July 2011.

She joined University of Texas at Arlington to pursue her M.S degree in Electrical Engineering in fall 2011. She joined the Multimedia Processing Lab in the same semester. She is presently working as a media validation intern at Intel, Folsom, California. She intends to find a job in multimedia field where she can utilize her knowledge and experience after her graduation in December 2013.