

COMPRESSION ARTIFACT REDUCTION IN HEVC USING ADAPTIVE BILATERAL FILTER

by

ROHITH REDDY ETIKALA

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

April 2016

Copyright © by Rohith Reddy Etikala 2016

All Rights Reserved



ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude and thank Dr. K. R. Rao for being supervisor, mentor and inspiring source during my thesis. I would also like to thank Dr. Howard T. Russell and Dr. Ioannis Schizas for being in my thesis committee.

I would also like to thank Dr. Yuriy A. Reznik and Abhijith Jagannath for their constant help during my internship at InterDigital, San Diego.

I would also like to thank our MPL Group members especially Deepak, Srikanth, and Vasavee for their support in EE5356 and their inputs during my research.

Last but not the least, I would like to thank my parents, my brother and family for supporting me to pursue my masters.

14 Apr 2016

ABSTRACT

COMPRESSION ARTIFACT REMOVAL IN HEVC USING ADAPTIVE BILATERAL FILTER

Rohith Reddy Etikala, M.S.

The University of Texas at Arlington

Supervising Professor: Dr. K. R. Rao

The High Efficiency Video Coding (HEVC) [8] is the latest video standard developed by Joint Collaborative Team on Video Coding (JCT-VC), a group of video coding experts from ITU-T Video Coding Experts Group and ISO/IEC Moving Picture Experts Group (MPEG).

As the demand for HD video (4K and 8K) increased, there is a need for higher coding efficiency than H.264/AVC. Also, there is increased use of parallel processors. So, HEVC [8] has been introduced to support increased video resolution and parallel processing. HEVC obtains about 50% reduction in bit rate when compared to its predecessor H.264/AVC at the same visual quality.

All these modern coding standards use block based transforms and coarse quantization to save video bandwidth over a channel by achieving compression. Though HEVC employs in-loop filters such as deblocking and SAO filters to remove compression artifacts obtained due to block based coding and coarse quantization of transform coefficients, there is still a scope for improvement. As in-loop filters are part of the standard, any modification of in-loop filters would modify the coding standard. So, post-processing techniques have gained popularity as they would not disturb the existing standard and reduce compression artifacts.

The thesis focusses on applying bilateral filter on HEVC decoded frames adaptively to reduce compression artifacts and still maintain very good visual quality and take less time to apply the filter than the in-loop filters. PSNR (Peak Signal to Noise Ratio) metric is used to evaluate subjective quality of the video.

Table of Contents

Contents

ACKNOWLEDGEMENTS.....	3
ABSTRACT.....	4
List of Acronyms:.....	10
Chapter 1 – Introduction.....	12
1.1 Multimedia Systems.....	12
1.2 Challenges with Multimedia Systems	13
Chapter 2— Image and video compression.....	15
2.1 Image compression	15
2.2 Image coding standards	17
2.3 Video compression.....	22
2.4 Video coding standards.....	27
Chapter 3 – High Efficiency Video Coding (HEVC) Standard	35
3.1 HEVC encoder and decoder:	35
3.2 Block Structures in HEVC.....	36
3.3 Parallelism in HEVC	38
3.4 Prediction in HEVC	39
3.5 Transform and Quantization.....	41
3.6 Entropy Coding.....	42
3.7 In-loop Filters	42
Chapter 4 – Artifacts due to Image and Video Compression.....	43
4.1 Different types of artifacts.....	43
4.2 Reduction of visual artifacts.....	46
4.3 Bilateral filter	47
4.4 Reduction of Compression Artifacts in HEVC through in-loop Filters.....	50
Chapter 5 – Adaptive Bilateral Filtering to Remove Compression Artifacts	53
Chapter 6 – Results	55
Chapter 7 – Conclusions and Future work.....	77
APPENDIX A: Test Conditions.....	78
References:	80
Biographical Information	83

List of tables:

Table 1. BasketballDrillText_832x480_50.yuv sequence quality metrics.....	55
Table 2. BasketballDrillText_832x480_50.yuv filter time	55
Table 3. RaceHorses_416x240_30.yuv sequence quality metrics	65
Table 4. RaceHorses_416x240_30.yuv filter time	66
Table 5. KristenAndSara_1280x720_60.yuv sequence quality metrics	70
Table 6. KristenAndSara_1280x720_60.yuv filter time	71

List of Figures:

Figure 1. Elements of multimedia Transmitter [1].....	12
Figure 2. Elements of multimedia Receiver [2].....	13
Figure 3. Texture image containing less redundancy [1].....	15
Figure 4. Basic flow of image compression coding [2].....	16
Figure 5. Elements of image encoding system [1].....	16
Figure 6. Elements of image decoding system [1].....	17
Figure 7. JPEG Encoder [1].....	17
Figure 8. The encoder model for JPEG compression standard [2].....	18
Figure 9. JPEG Decoder [1].....	18
Figure 10. The decoder model for JPEG compression standard [2].....	19
Figure 11. 4:2:0, 4:2:2: and 4:4:4 sampling patterns for Luminance and Chrominance [3].....	20
Figure 12. Zig-zag scanning order in JPEG [3].....	22
Figure 13. Spatial and temporal sampling of a video sequence [3].....	23
Figure 14. Hybrid video encoder [1].....	24
Figure 15. Hybrid video decoder [1].....	24
Figure 16. GOP (Group of pictures) [1].....	25
Figure 17. Backward motion estimation with frame k as the current frame and frame (k-1) as the reference frame [1].....	26
Figure 18. Forward motion estimation with frame k as the current frame and frame (k+1) as the future reference frame [1].....	26
Figure 19. Prediction of a macro block from multiple reference frames [4].....	27
Figure 20. Searching for the best matching block [4].....	27
Figure 21. Evolution of video coding standards [41].....	28
Figure 22. 6-layer hierarchical structure of MPEG-1 [1].....	29
Figure 23. Encoder structure of MPEG-1 [4].....	30
Figure 24. Simplified decoder structure of MPEG-1 [6].....	30
Figure 25. Encoder structure of MPEG-2 [4].....	31
Figure 26. Decoder structure of MPEG-2 [7].....	32
Figure 27. Encoder structure of H.264 [3].....	33
Figure 28. Decoder structure of H.264 [3].....	33
Figure 29. Block Diagram of HEVC Encoder [8].....	35
Figure 30. Block Diagram for HEVC Decoder [14].....	36
Figure 31. CTU partitioning and quad tree structure [11].....	36
Figure 32. 64x64 CTU partitioning using quad-tree structure [9].....	37
Figure 33. Division of a 64x64 luma CTB using quad-tree structure [9].....	37
Figure 34. Comparison of motion compensation block sizes supported in different standards [9].....	38
Figure 35. Picture, Slice, Coding Tree Unit (CTU), Coding Unit (CU) [10].....	38
Figure 36. Slices and tiles in HEVC [8].....	39
Figure 37. Modes and directional orientations for intra prediction in HEVC [8].....	40
Figure 38. Illustration of Motion Estimation Process [12].....	40
Figure 39. Integer and fractional sample positions for luma interpolation [8].....	41
Figure 40. Filter coefficients for luma and chroma fractional sample interpolation [8].....	41
Figure 41. Highly compressed image block [17].....	43

Figure 42. Blocking artifacts due to block based image transform [17]	44
Figure 43. Blocking artifact resulting from predictive coding [9]	44
Figure 44. Example of the ringing effect, where it is most evident around the bright table-edge and the boundary of the arm [18].....	45
Figure 45. Example of the ringing effect in a sequence coded at a relatively high bit-rate. Most prominent along the edge formed by the upper-arm in the scene [16]	45
Figure 46. Common artifacts due to block based coding.	46
Figure 47. Gaussian smoothing [20]	47
Figure 48. Bilateral Filtering [20].....	47
Figure 49. Input image	48
Figure 50. Limited and strong smoothing [20].....	48
Figure 51. Bilateral filter [20]	49
Figure 53. Exploring parameter space for bilateral filter [20]	50
Figure 54. Performance of Deblocking filter in HEVC for Basketball Drive Sequence [9].....	51
Figure 55. Performance of Deblocking filter in HEVC for KristenAndSara Drive Sequence [9]	52
Figure 56. Performance of SAO filter in HEVC on SliceEditing sequence [9]	52
Figure 57. Performance of SAO filter in HEVC on RaceHorses sequence [9].....	52
Figure 58. Parameter controlling fall-off weight in spatial domain.....	53
Figure 59. Block discontinuity Map for 8x8 pixel block [22]	54
Figure 60. Original BasketballDrillText_832x480_50.yuv sequence frame1	56
Figure 61. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter disabled.	57
Figure 62. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.	58
Figure 63. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.	59
Figure 64. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter enabled.....	60
Figure 65. Original BasketballDrillText_832x480_50.yuv sequence frame1	61
Figure 66. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter disabled.	62
Figure 67. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.	63
Figure 68. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.	64
Figure 69. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter enabled.....	65
Figure 70. Original RaceHorses_416x240_30.yuv sequence frame1.....	66
Figure 71. RaceHorses_416x240_30.yuvs equence frame1 with both in-loop filters and bilateral filter disabled.....	67
Figure 72. RaceHorses_416x240_30.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.....	68
Figure 73. RaceHorses_416x240_30.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.	69

Figure 74. RaceHorses_416x240_30.yuv sequence frame1 with both in-loop filters and bilateral filter enabled. 70

Figure 75. Original KristenAndSara_1280x720_60.yuv sequence frame1..... 72

Figure 76. KristenAndSara_1280x720_60.yuv sequence frame1 with both in-loop filters and bilateral filter disabled. 73

Figure 77. KristenAndSara_1280x720_60.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled. 74

Figure 78. KristenAndSara_1280x720_60.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled..... 75

Figure 79. KristenAndSara_1280x720_60.yuv sequence frame1 with both in-loop filters and bilateral filter enabled..... 76

List of Acronyms:

AMVP – Advance Motion Vector Prediction

AVC - Advanced Video Coding

AVS – Audio Video Standard

CABAC – Context Adaptive Binary Arithmetic Coding

CAVLC – Context Adaptive Variable Length Coding

CB – Coding Block

CCITT – Consultative Committee International Telephone and Telegraph

CIF – Common Intermediate Format

CPB – Coded Picture Buffer

CSVT – Circuits and Systems for Video Technology

CTB – Coding Tree Block

CTU – Code Tree Unit

CU – Coding Unit

dB – decibel

DCT – Discrete Cosine Transform

DPB – Decoded Picture Block

DST – Discrete Sine Transform

FPS – Frames Per Second

FS – Full Search

HD – High Definition

HEVC – High Efficiency Video Coding

HHR – Half Horizontal Resolution

IDCT – Inverse Discrete Cosine Transform

IEC – International Electrotechnical Commission

ISO – International Standards Organization

ISDN – Integrated Services Digital Network

ITU-T– International Telecommunication Union

JCT-VC - Joint Collaborative Team on Video Coding

JVT - Joint video team

kbps – kilo bits per second

MB - Macroblock

MC – Motion Compensation

MCP – Motion Compensated Prediction

MPEG – Moving Pictures Experts Group

MV – Motion Vector

NAL – Network Abstraction Layer

PB – Prediction Block

POTS – Plain Old Telephone Systems

PSNR – Peak-Signal-to-Noise-Ration

PU – Prediction Unit

QP – Quantization Parameter

RAM – Random Access Memory

SAO – Sample Adaptive Offset

SD – Standard Definition

SEI - Supplemental Enhancement Information

SI – Switched Intra

SP – Switched Predictive

TB – Transform Block

TU – Transform Unit

URQ – Uniform Reconstruction Quantization

VLC – Variable Length Coding

VLD – Variable Length Decoding

Chapter 1 – Introduction

1.1 Multimedia Systems

Multimedia is an effective tool for communication. To communicate or express an idea, we use as much media as available these days to make it very clear. 20 years ago, we did not have sophisticated multimedia technologies as the computers which process these multimedia signals were costly and less powerful. But today, due to the availability of faster processors, availability of very large scale circuits for real-time processing, effective data compression tools and algorithms to eliminate redundancies and achieve bandwidth reduction, advancement in networking technologies etc., paved the path for sophisticated multimedia technologies.

A typical multimedia communication involves a transmitter and a receiver. The elements involved in multimedia transmitter are shown in Figure 1.

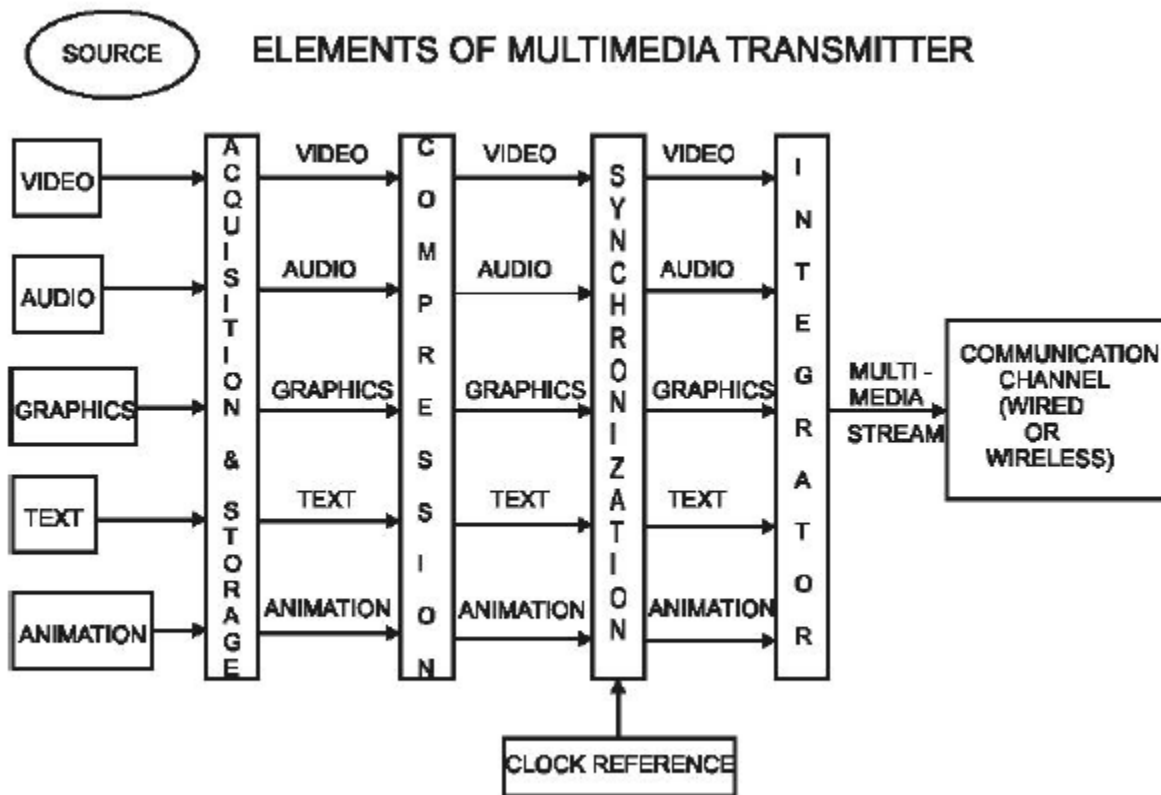


Figure 1. Elements of multimedia Transmitter [1]

As shown in Figure 1, we receive information in the form of video, audio, graphics, text etc. from various devices like camera, keyboard, voice recorder etc. All further processing is done by computer. The data acquisition and storage is followed by compression to remove redundancies present in the data and then synchronization of this media takes place. Following this, we integrate the media and then transmit the integrated multimedia stream through a wired or wireless channel.

The elements of multimedia receiver is shown in Figure 2.

The elements of multimedia receiver are shown in Figure 2. The destination end receives this multimedia stream and then extracts all the individual media and then synchronization happens. Finally, the uncompressed data is fed back to multimedia device.

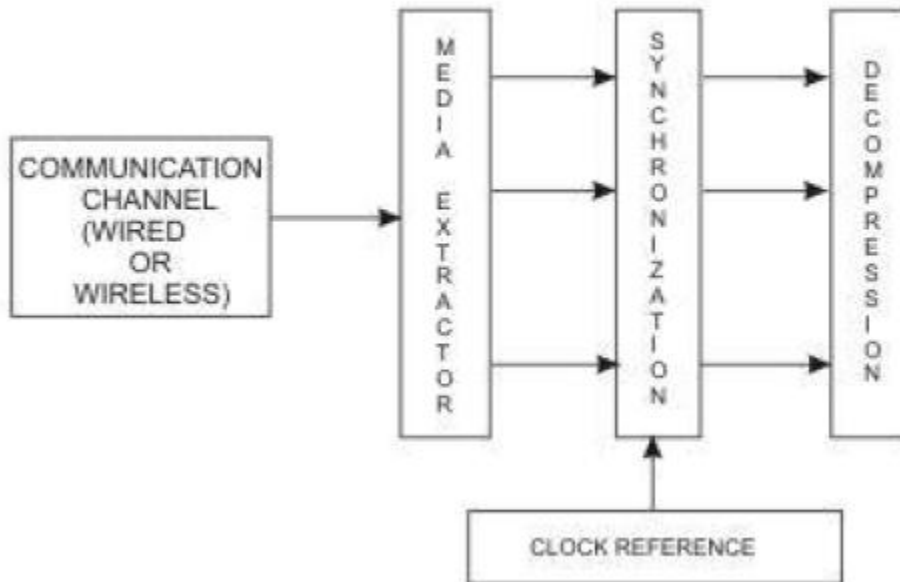


Figure 2. Elements of multimedia Receiver [2]

1.2 Challenges with Multimedia Systems

Bandwidth limitations: Real time video transmission requires huge amounts of data to be transmitted over a channel in a very short period of time. For example, a video sequence with fairly small resolution such as 352x288 pixels, having 24 bits/pixel with frame rate of 30 frames/sec requires: $352 \times 288 \times 24 \times 30$ bits/sec, which is about 72.9 megabits/sec [1]. So, transmitting this video over a multimedia channel requires very high bandwidth. If this video is coupled with audio, text etc. it requires even more bandwidth.

Real-time processing requirements: Even after meeting the bandwidth requirements, significant amount of processing will be involved to exploit redundancies in the data. The advantage with data compression will be lost if we cannot meet the data processing requirements. For example, a 30 frames/second video requires a single frame to be processed in 33 milliseconds. So, our processing must be completed in less than 33 milliseconds to support real time delivery of the media.

Media synchronization: Media available from various sources must be synchronized properly. Lip sync is the main problem in multimedia communication. The audio and video data delivered must remain in synchronization. Lack of synchronization would defeat the purpose of efficient multimedia communication.

The extent to which a frame can be compressed depends on the content present in the frame. Some frames may be less compressed and some may be compressed higher. A multimedia system must also

address variable bit-rate issue with each frame. Further, multimedia systems need proper storage and retrieval with the growing multimedia demand.

Chapter 2— Image and video compression

The major challenge with multimedia communication is transmission of image and video data over limited bandwidth channels. Image and video compression is achieved by exploiting redundancies which reduce the bit-rate for transmitting them across the transmission channel. Image or video techniques can be lossless and lossy. Lossy compression techniques achieve higher compression with a reduction in visual quality.

2.1 Image compression

As the pixels in the image are similar to the neighboring ones, compression is possible by exploiting these redundancies. The extent of redundancies vary from image to image. For example, figure 3 shows image with less redundancy as it has more details.

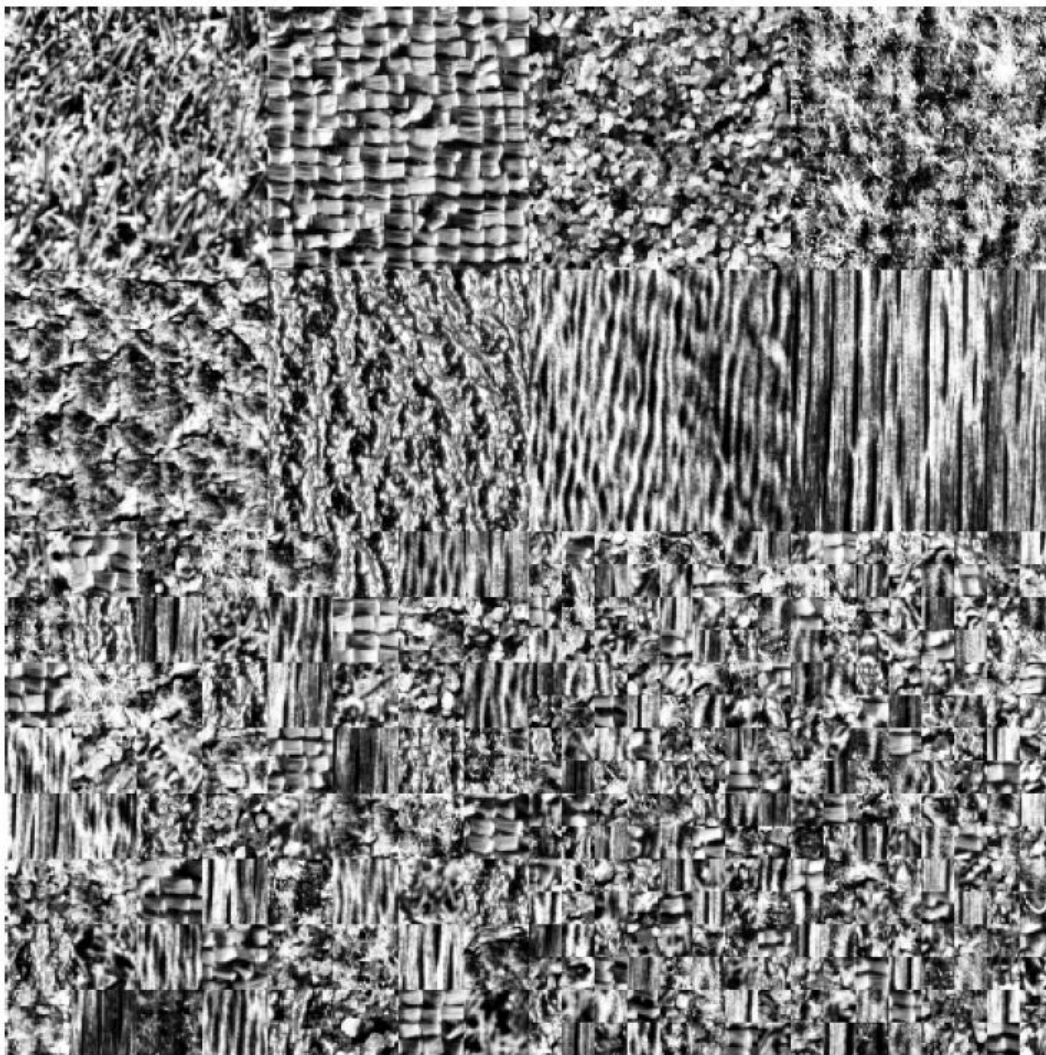


Figure 3. Texture image containing less redundancy [1]

Redundancies in an image are categorized into statistical redundancy (pixels within the image are similar to the neighboring pixels) and psychovisual redundancy (the extent to which image details are preserved

as human eyes are more sensitive to slow and gradual changes in intensities than perceiving finer details and rapid changes in intensities).

The basic flow of image compression is shown in Figure 4.

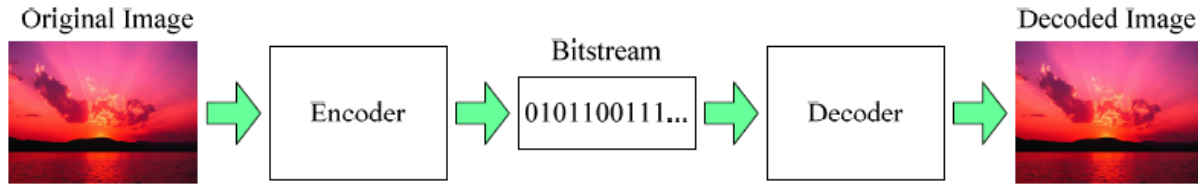


Figure 4. Basic flow of image compression coding [2]

Elements of image compression system are shown in Figure 5.

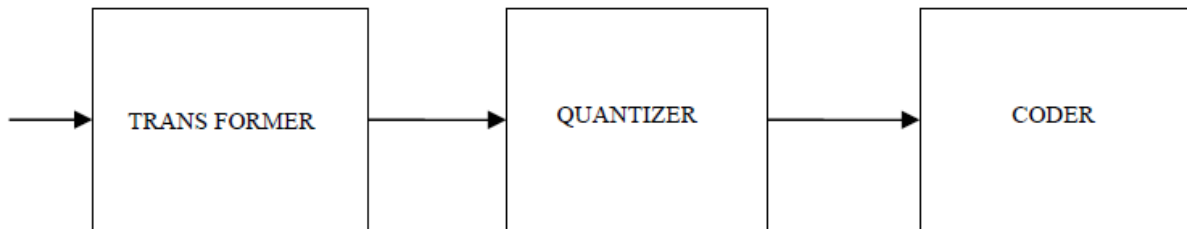


Figure 5. Elements of image encoding system [1]

The transformer transforms the input image in to a form that is more amenable to compression where only a few coefficients contain most of the energy and efficient compression is possible. Examples of image transforms are Discrete Fourier Transforms (DFT) [41], Discrete Cosine Transforms (DCT) [26], Karhunen-Love Transforms (KLT) [41], Discrete Wavelet Transforms (DWT) [41] etc.

The quantizer then generates a limited number of symbols that can be used to represent the transformed data. And, finally coder assigns a code word to each symbol of the quantizer. The code word can be fixed or variable depending on whether a fixed length or variable length coding technique is used.

The elements of image de-compression technique are shown in Figure 6. These perform the exact inverse operations of elements in the image encoding system.

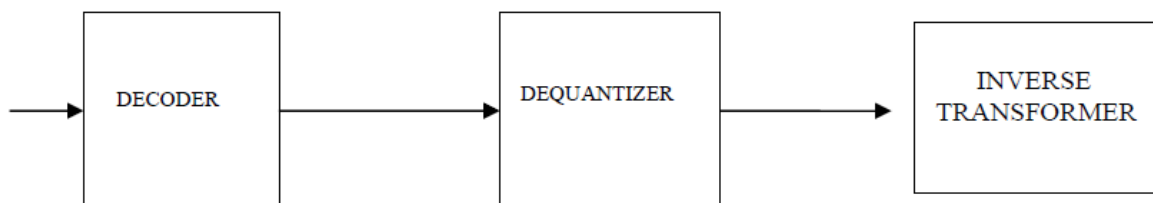


Figure 6. Elements of image decoding system [1]

2.2 Image coding standards

With the rapid developments of imaging technology, image compression and coding tools and techniques, it is necessary to develop coding standards so that there is compatibility and interoperability between the image communication and storage products manufactured by different vendors in the multimedia market. Without the availability of such standards, encoders and decoders cannot communicate with one another and hence the service providers will have to support a variety of formats to meet the needs of the customers and the customers will have to install a number of decoders to handle a large number of data formats. Towards the objective of setting up coding standards to address this issue, the international standardization agencies, such as International Standards Organization (ISO), International Telecommunication Union (ITU), International Electro-technical Commission (IEC) etc. have formed expert groups and solicited proposals from industries, universities and research laboratories. These standards use the coding and compression techniques (both lossless and lossy).

The first standard developed for compressing and coding monochrome and color images of any size and sampling rate was developed by Joint Photographic Experts Group (JPEG) and is known as JPEG. Later JPEG-2000 has been developed for still images.

The block diagram of JPEG encoder is shown in figure 7.

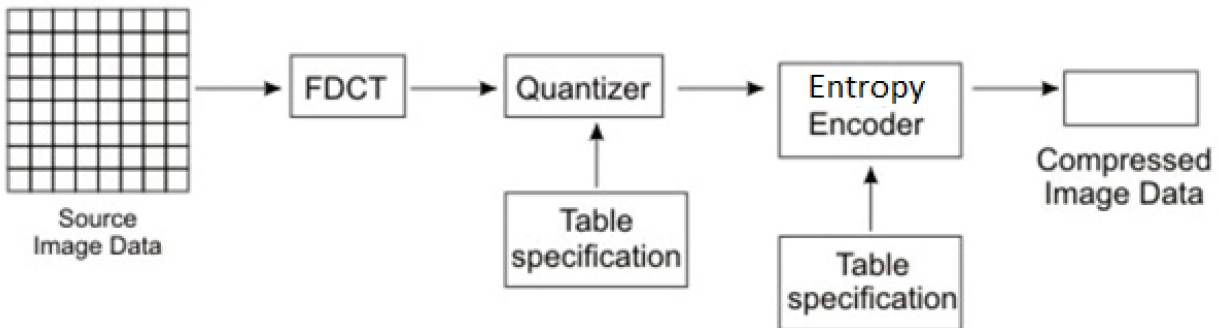


Figure 7. JPEG Encoder [1]

A more detailed JPEG encoder is shown in figure 8, in which RGB image is converted to YUV image and then chrominance may be down sampled as human eyes are more sensitive to luminance component than chrominance.

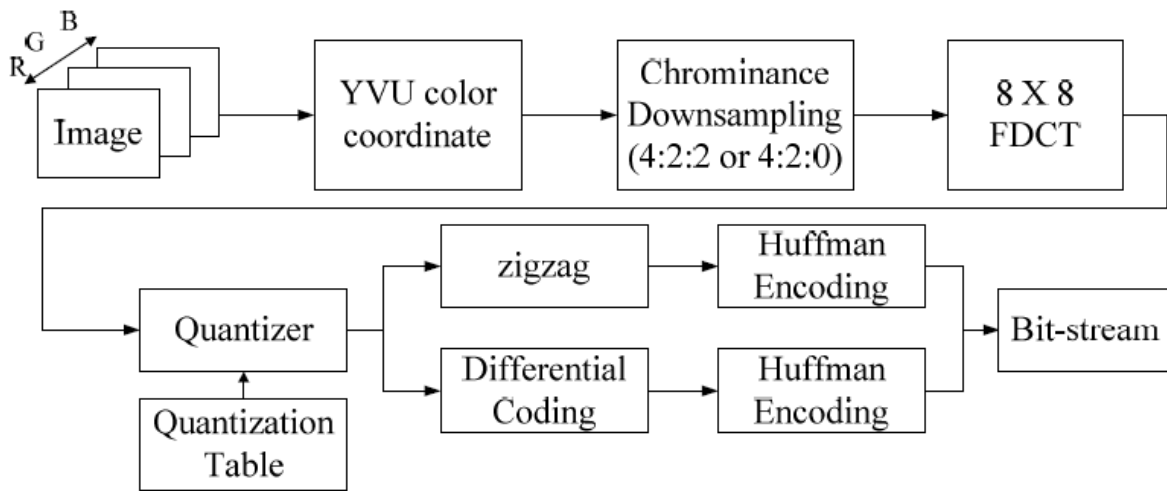


Figure 8. The encoder model for JPEG compression standard [2]

Similarly, at the decoder we do inverse operations as shown in figure 9.

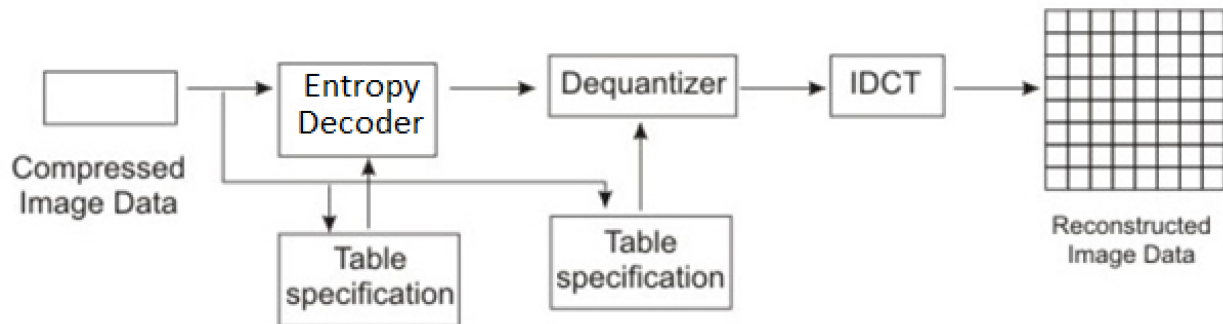


Figure 9. JPEG Decoder [1]

A more detailed JPEG decoder is shown in Figure 10 and it clearly shows conversion of YUV image back to RGB image. The RGB image format is used especially for color display systems. For many applications, it is desirable to describe color in terms of luminance (Y) and chrominance (C_b and C_r) to enable efficient processing and transmission. Down sampling reduces the bit rate and saves bandwidth. The down sampled images are converted back to 4:4:4 and then converted to RGB at the decoder. For 8 bit image, the $Y C_b C_r$ values in the YUV coordinate are related to the RGB values in the RGB coordinate by

$$\begin{aligned}
 Y &= 0.299 \times R + 0.587 \times G + 0.114 \times B + 0 \\
 C_b &= -0.169 \times R - 0.331 \times G + 0.499 \times B + 128 \\
 C_r &= 0.499 \times R - 0.418 \times G - 0.0813 \times B + 128
 \end{aligned}$$

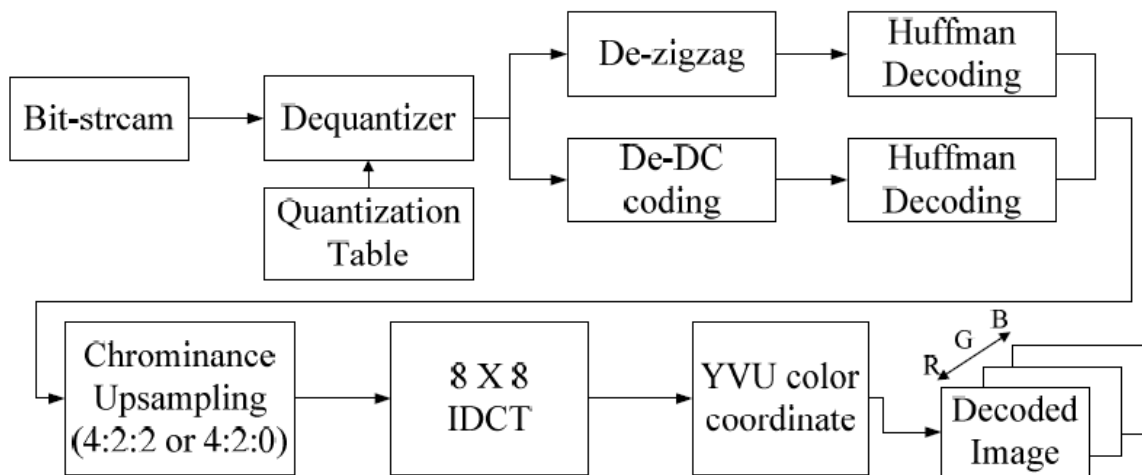


Figure 10. The decoder model for JPEG compression standard [2]

The various sampling patterns (4:2:0, 4:2:2 and 4:4:4) for Luminance and Chrominance supported by JPEG are shown in Figure 11.

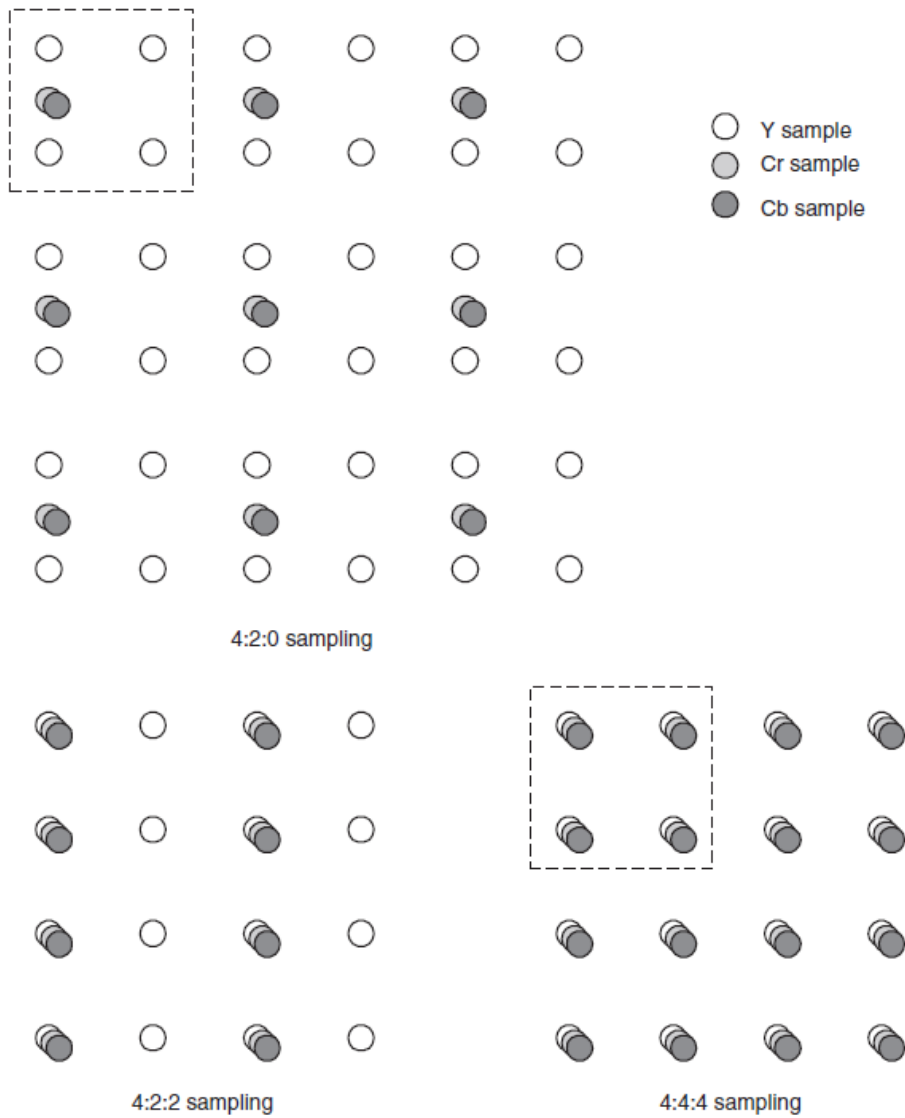


Figure 11. 4:2:0, 4:2:2 and 4:4:4 sampling patterns for Luminance and Chrominance [3]

The image is first divided into 8x8 blocks and then FDCT (Forward Discrete Cosine Transform) [26] is applied to the 8x8 block. The forward and inverse DCT are defined as

Forward DCT

$$F(u, v) = \frac{2}{N} C(u)C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right]$$

for $u = 0, \dots, N-1$ and $v = 0, \dots, N-1$

$$\text{where } N = 8 \text{ and } C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases}$$

Inverse DCT

$$f(x, y) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v) F(u, v) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right]$$

for $x = 0, \dots, N-1$ and $y = 0, \dots, N-1$ where $N = 8$

The $f(x,y)$ is the value of each pixel in the selected 8×8 block, and the $F(u,v)$ is the DCT coefficient after transformation. The transformation of the 8×8 block is also a 8×8 block composed of $F(u,v)$. The DCT is a lossless procedure and the data can be recovered by Inverse DCT. The 8×8 DCT matrix is quantized by dividing each coefficient by its corresponding quantization value from the default quantization table.

$$F(u, v)_{\text{Quantization}} = \text{round}\left(\frac{F(u, v)}{Q(u, v)}\right)$$

$$F(u, v)_{\text{deQ}} = F(u, v)_{\text{Quantization}} \times Q(u, v)$$

The JPEG committee came up with a 8×8 quantization matrix that works well with performance close to optimal condition. The quantization matrix for chrominance and luminance are

$$Q_Y = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

$$Q_c = \begin{pmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{pmatrix}$$

The quantized coefficients (which contain one DC and 63 AC coefficients) are zig-zag scanned as shown in the Figure 12. Then the coefficients are encoded using Huffman encoding.

0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

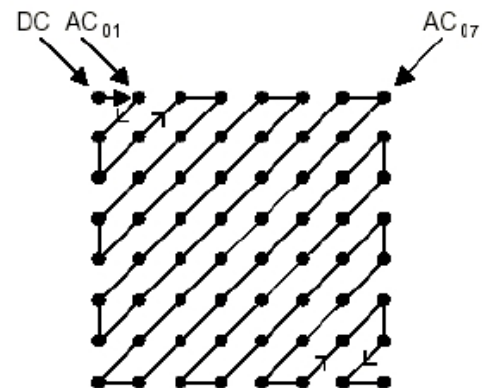


Figure 12. Zig-zag scanning order in JPEG [3]

The similar inverse process is followed at the JPEG decoder as shown in figure 10.

JPEG encoding is very popular, but images compressed using JPEG at a very low bit rate show severe blocking artifacts as we use block based DCT. So, an advanced imaging standard known as JPEG-2000 [38] was developed. JPEG-2000 is based on EBCOT (Embedded Block Coding with Optimized Truncation) wavelet coding technique.

2.3 Video compression

Spatial and temporal sampling of a video is shown in figure 13. Video compression aims at exploiting redundancies in spatial and temporal domains. Temporal redundancy is exploited by predicting the current frame using the information from the past frames.

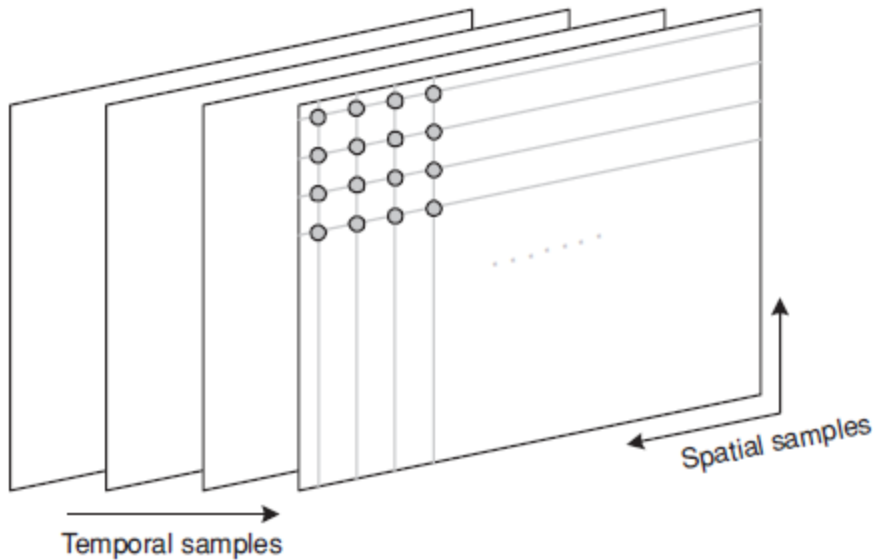


Figure 13. Spatial and temporal sampling of a video sequence [3]

The block diagram of a hybrid video encoder is shown in figure 14. The hybrid video decoder is shown in figure 15. These codecs (encoder and decoder) are popularly called as hybrid codecs as they use predictive and transform domain techniques. Most of the modern video codecs resemble hybrid video codec with a slight modification.

The predictive techniques temporally current frame from previously stored frames and it assumes that video sequences exhibit similarity between consecutive frames. The predicted frame is subtracted from the current frame and the difference image is obtained. The spatial redundancy in the difference image is exploited by applying image transforms such as DCT and Integer-DCT and the coefficients are quantized and entropy coded. The encoded information is sent to the receiver.

The encoder has a built in decoder to reconstruct the difference image. The difference image is added to the predicted image to generate the buffer. The motion estimation block in the encoder determines the displacement between current frame and previously stored frames. The displacements computed are applied on the stored frames to obtain the predicted frame.

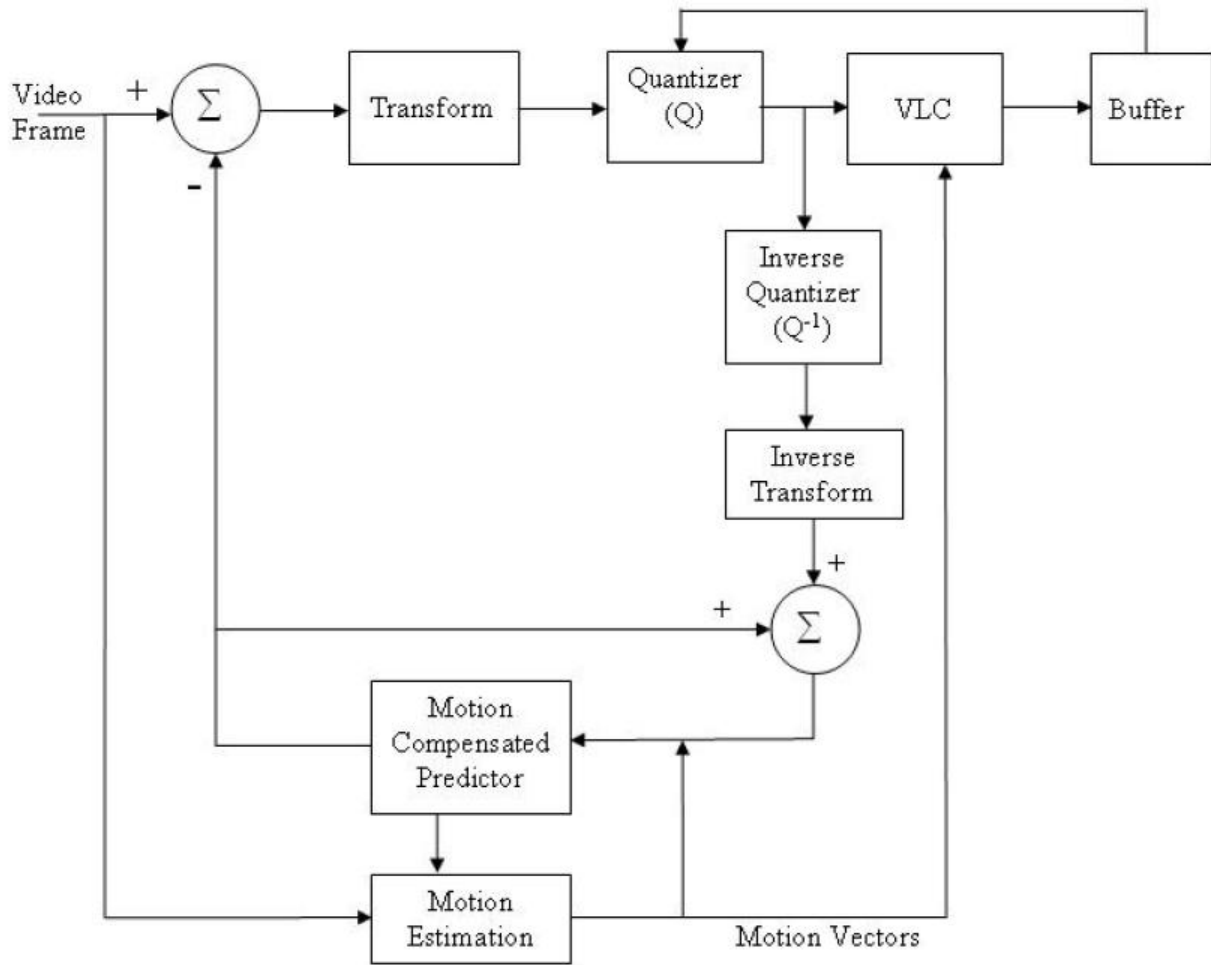


Figure 14. Hybrid video encoder [1]

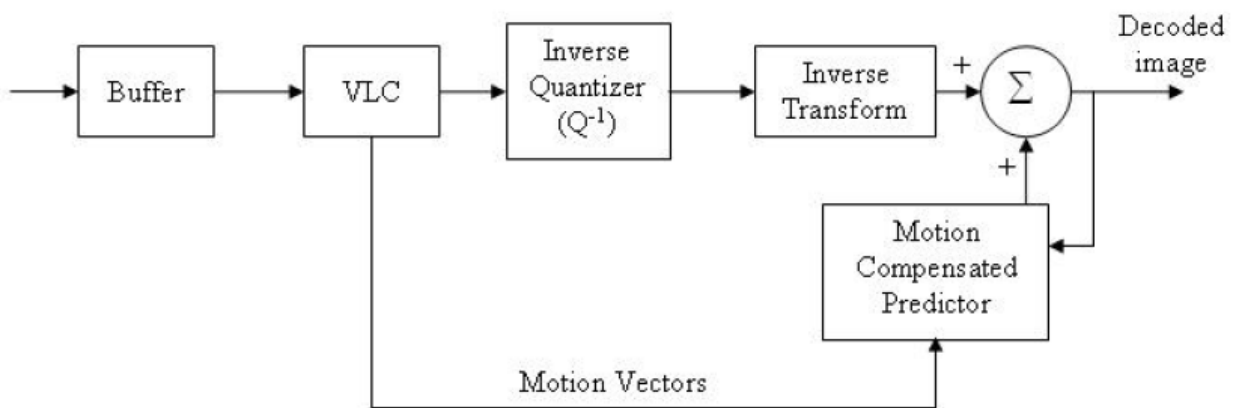


Figure 15. Hybrid video decoder [1]

A video sequence itself is considered to have a group of pictures as shown in figure 16.

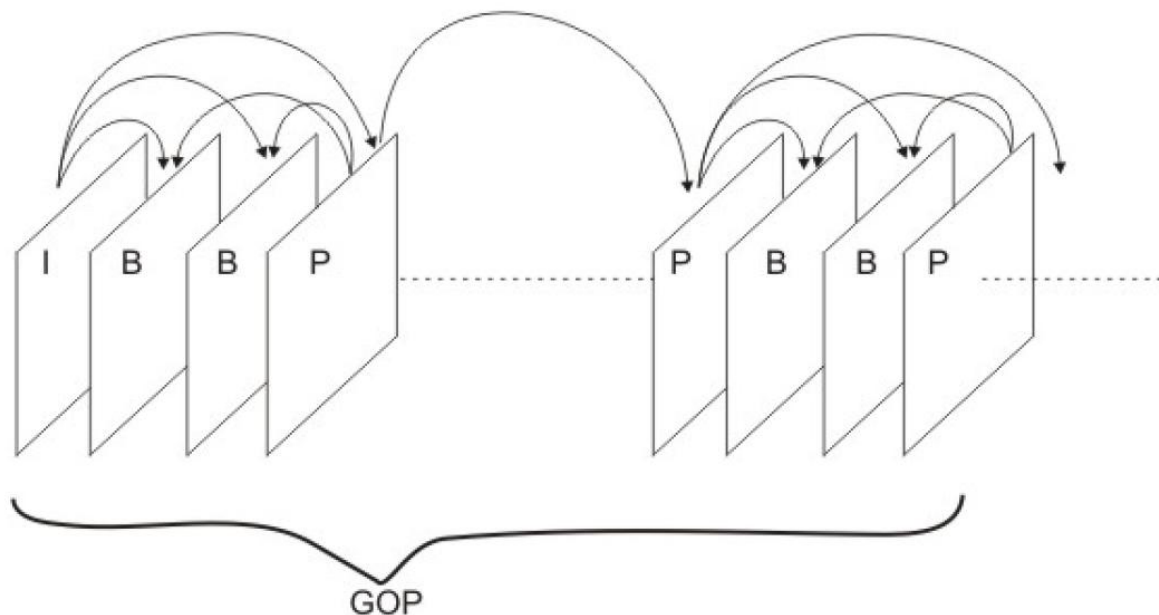


Figure 16. GOP (Group of pictures) [1]

The intra coded pictures (I-Pictures) are coded without any reference to other pictures in the video sequence. These pictures are coded in a way similar to JPEG. The inter frame predicted pictures (P-Pictures) are coded with reference to nearest I-picture or P-picture using motion compensation. The Bi-directionally predicted pictures (B-Pictures) use bi-directional motion estimation to predict the current frame from I-picture or P-picture in both the directions in temporal order.

I-pictures achieve least compression as they do not exploit temporal redundancy while B-pictures achieve highest compression as they exploit temporal redundancy in both the directions in the temporal order and hence most of the frames in GOP are B-pictures.

Motion estimation computes the displacement between the current frame and the past frames and the displacement vector, commonly known as motion vector is used in the motion compensation block to predict the current frame by applying appropriate displacements to the reference frame.

Motion estimation can be backward or forward as shown in figures 17 and 18. Backward motion estimation leads to forward motion prediction as motion vectors are searched in the past frame to predict the current frame and vice versa.

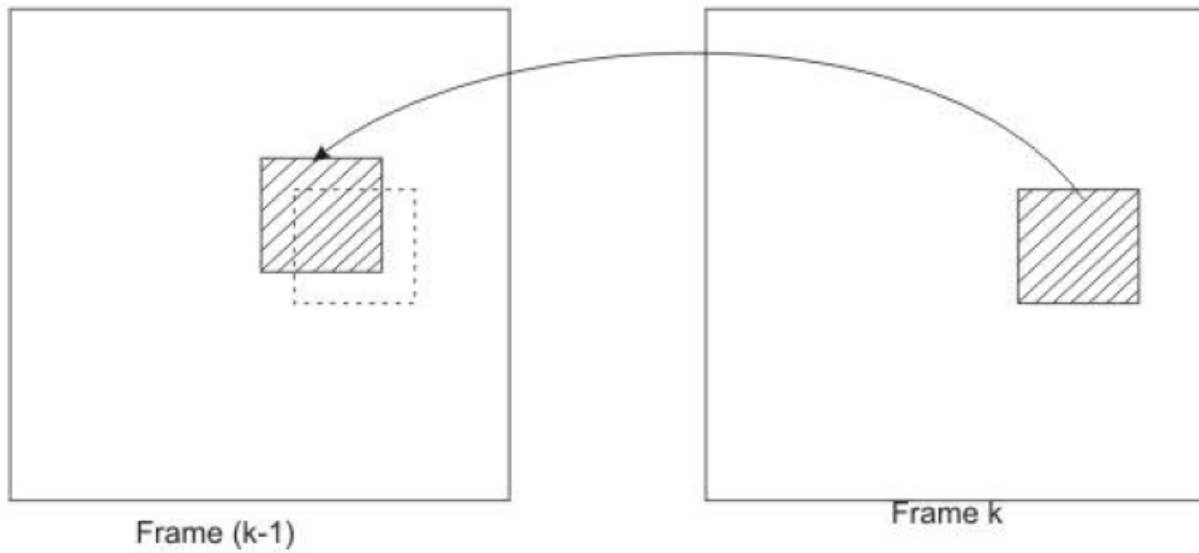


Figure 17. Backward motion estimation with frame k as the current frame and frame $(k-1)$ as the reference frame [1]

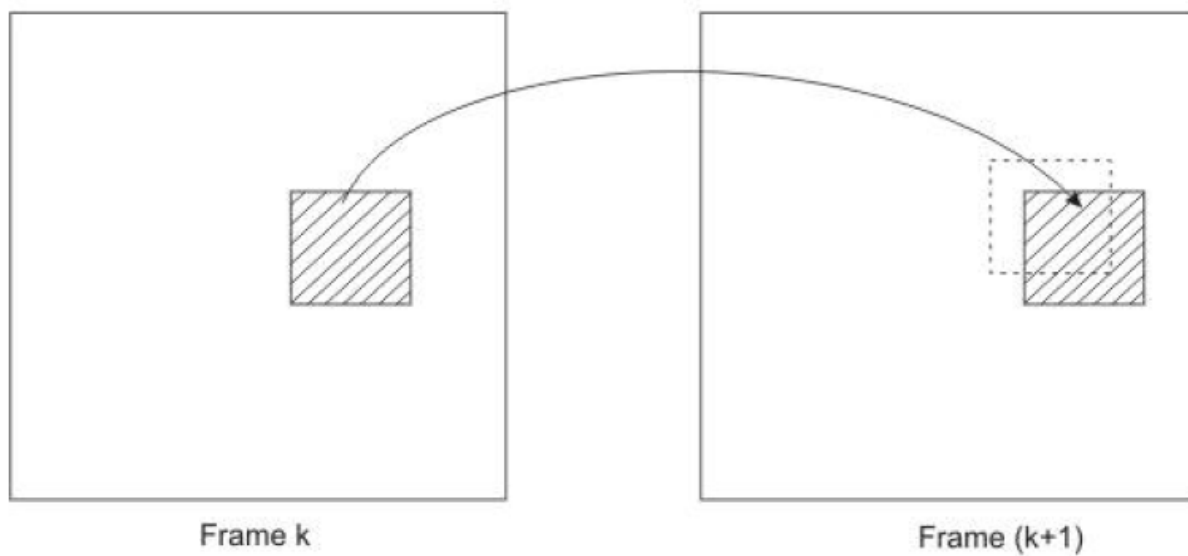


Figure 18. Forward motion estimation with frame k as the current frame and frame $(k+1)$ as the future reference frame [1]

Further, the latest video codes use multiple reference frames to predict a macro block (which are sub divisions of a frame), which is shown in figure 19.

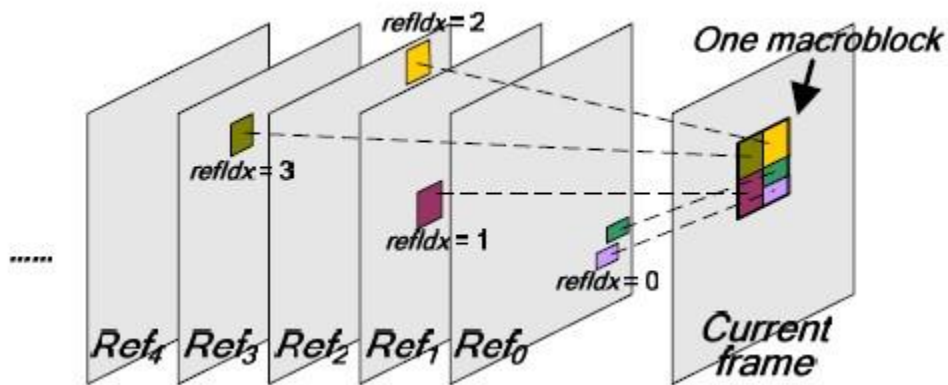


Figure 19. Prediction of a macro block from multiple reference frames [4]

Further, to determine the motion vector, we determine a search region S over which the search happens to predict the current block from reference frame, which is depicted in figure 20.

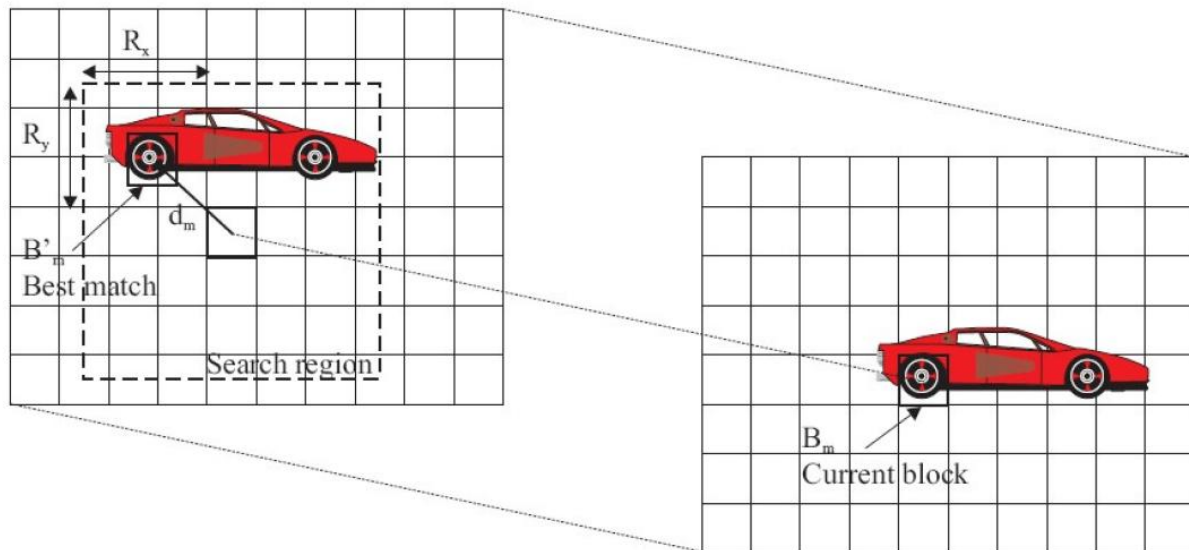


Figure 20. Searching for the best matching block [4]

2.4 Video coding standards

The evolution of video coding standards over the years is shown in figure 21.

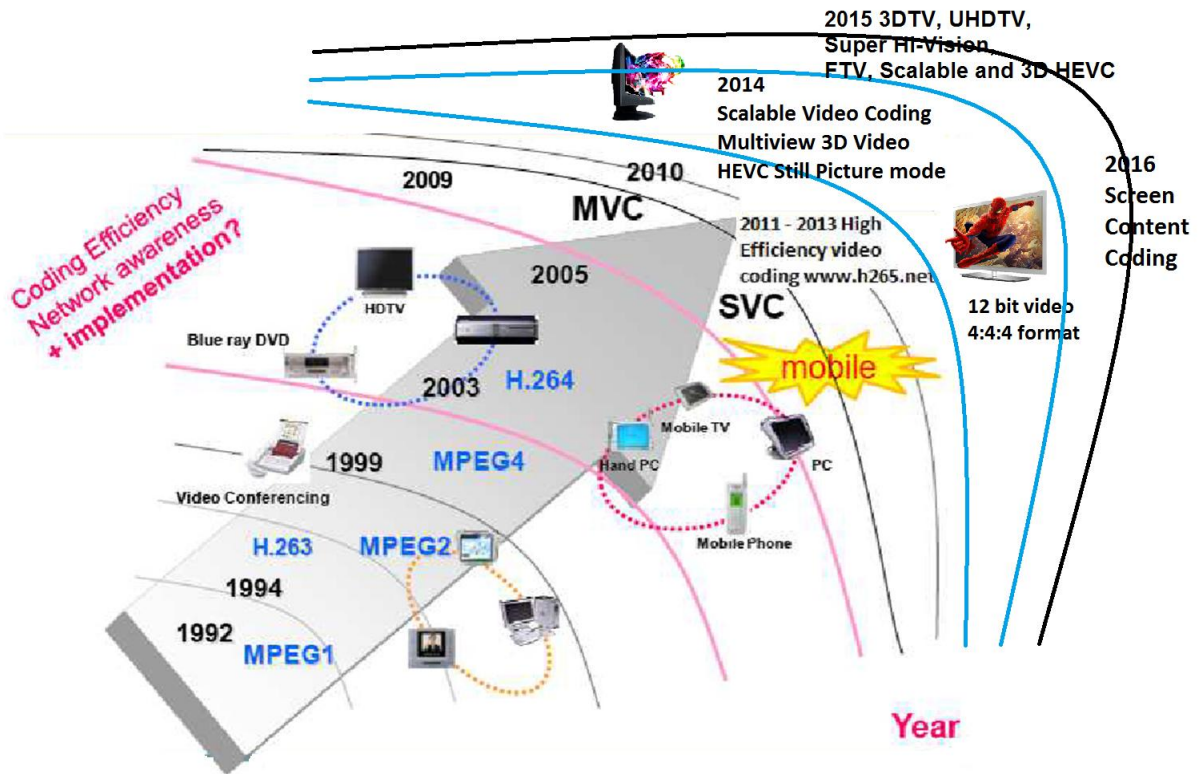


Figure 21. Evolution of video coding standards [41]

Efforts to standardize video data exchange via storage media or via communication networks are actively in progress since early 1980s. A number of international video and audio standardization activities started within the International Telephone Consultative Committee (CCITT), followed by the International Radio Consultative Committee (CCIR), and the International Standards Organization / International Electrotechnical Commission (ISO/IEC). An experts group, known as the Moving Pictures Experts Group (MPEG) was established in 1988 in the framework of the Joint ISO/IEC Technical Committee with an objective to develop standards for coded representation of moving pictures, associated audio, and their combination for storage and retrieval of digital media[1]. The standard was finalized in 1992 and was nicknamed MPEG-1.

MPEG-1 follows a 6 layer hierarchical structure as shown in figure 22.

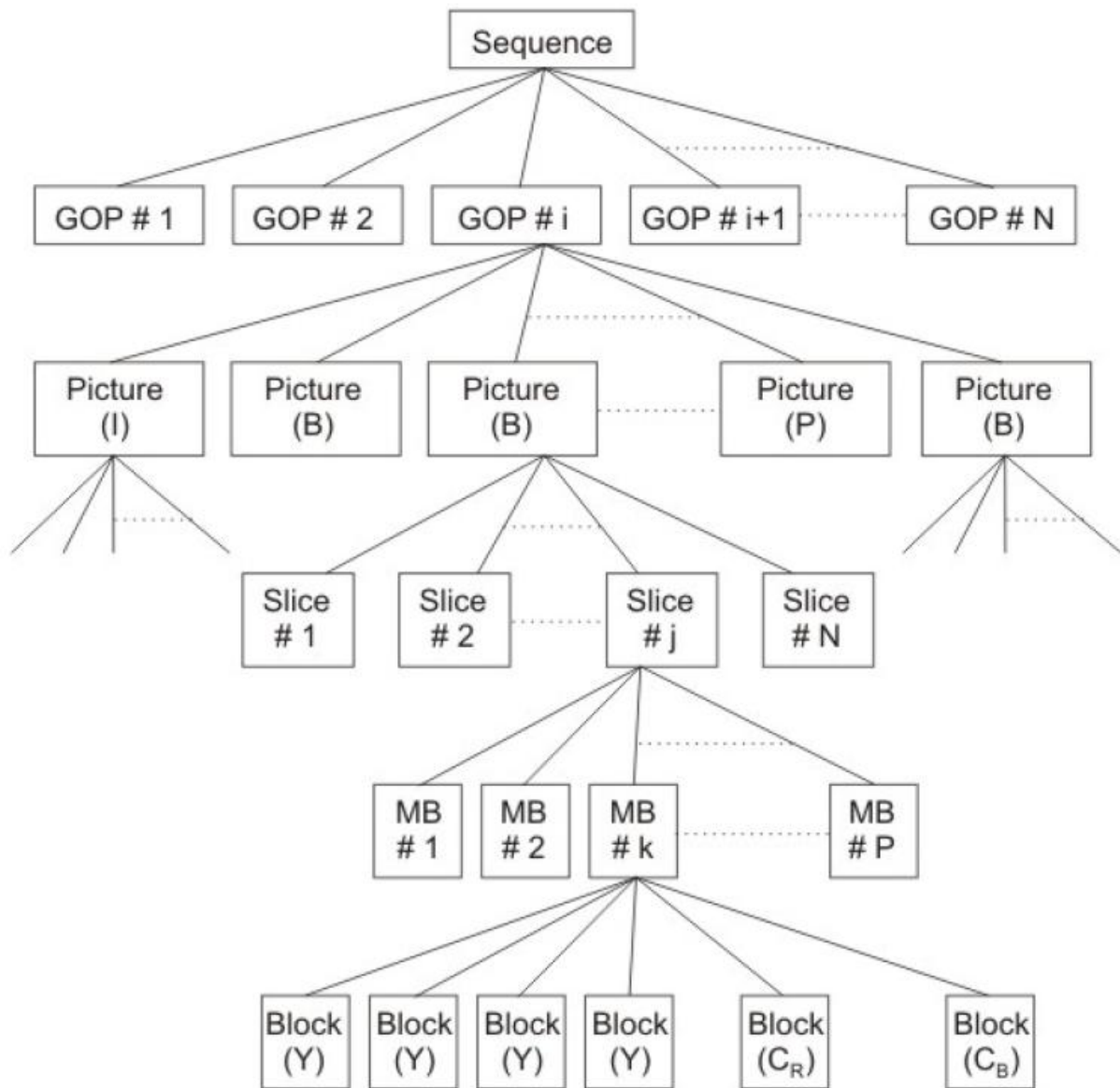


Figure 22. 6-layer hierarchical structure of MPEG-1 [1]

A video sequence is split into several GOPs (group of pictures) and these pictures are further divided into slices which are sequence of macro blocks in raster scan order. These macro blocks are composed of chrominance and luminance blocks. In MPEG-1, the size of macro block is 16x16 pixels which are sub divided in to 8x8 blocks before applying the DCT. The encoder and decoder structures of MPEG-1 are shown in figures 23 and 24 respectively.

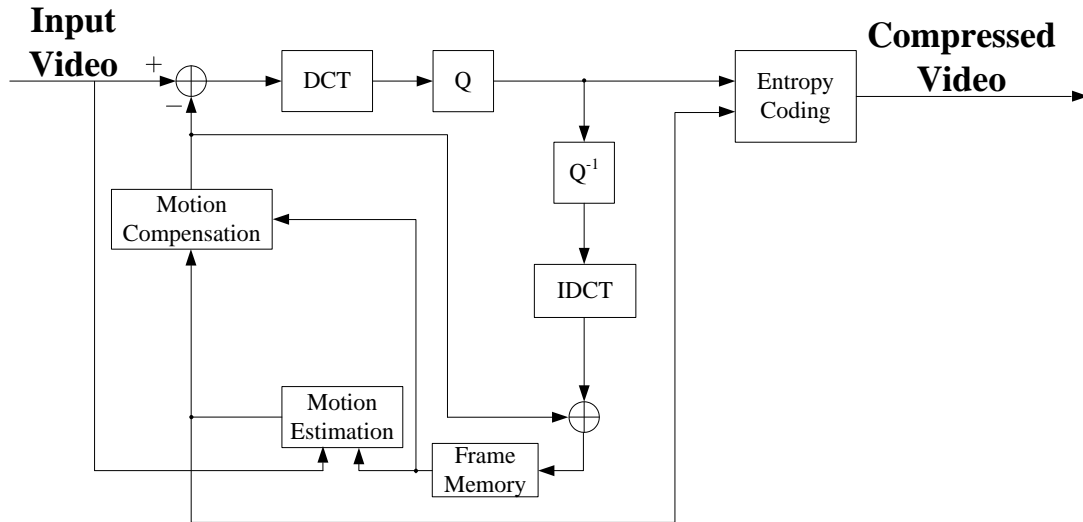


Figure 23. Encoder structure of MPEG-1 [4]

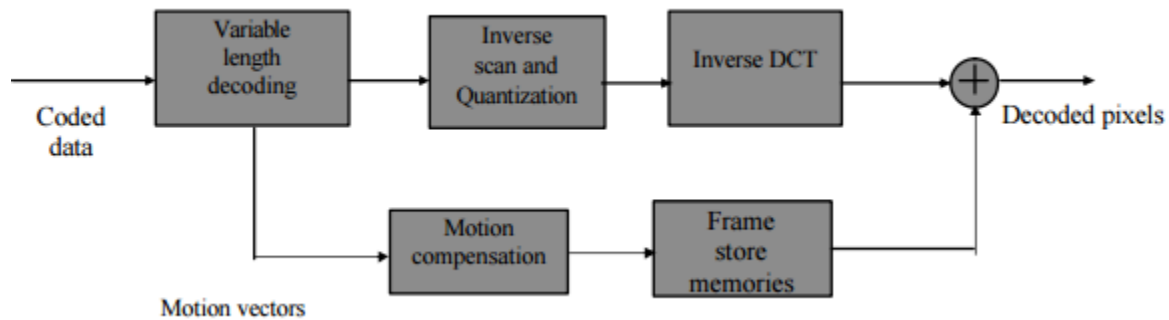


Figure 24. Simplified decoder structure of MPEG-1 [6]

MPEG-1 was the first ISO/IEC standard that was primarily targeted for a bit-rate up to 1.5 megabits/sec for digital storage of video on CDs. The standard was adopted in 1992. As newer application areas like video broadcasting and HDTV emerged, there was a need to develop new standards [1].

MPEG continued its standardization efforts and the next standard, MPEG-2 [42] was given the charter to provide video quality not lower than NTSC/PAL. Video coding for broadcast and storing video on digital video disks (DVD) with an order of 2-15 megabits/sec allocated to audio and video coding were addressed by MPEG-2. MPEG-2 addresses the emerging applications like digital cable television distribution, high definitions televisions (HDTV), satellite digital video broadcasts, networked multimedia through ATM etc. The encoder and decoder structure of MPEG-2 are shown in figures 25 and 26 respectively.

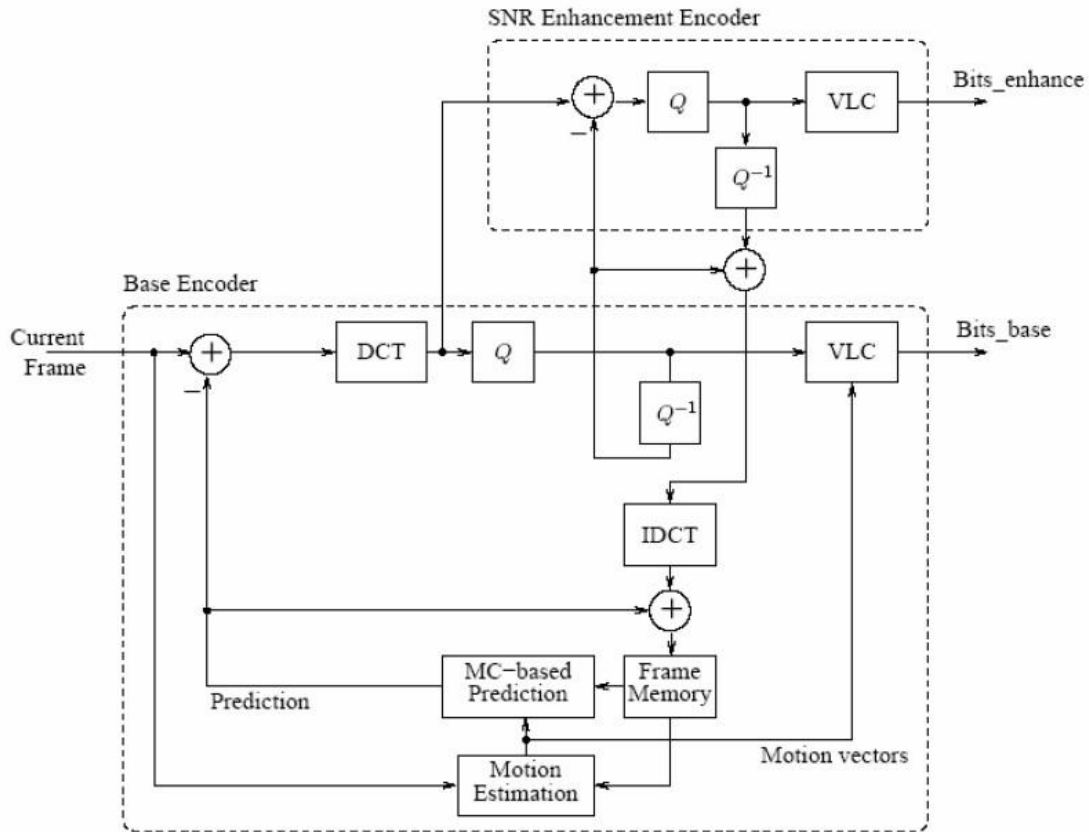


Figure 25. Encoder structure of MPEG-2 [4]

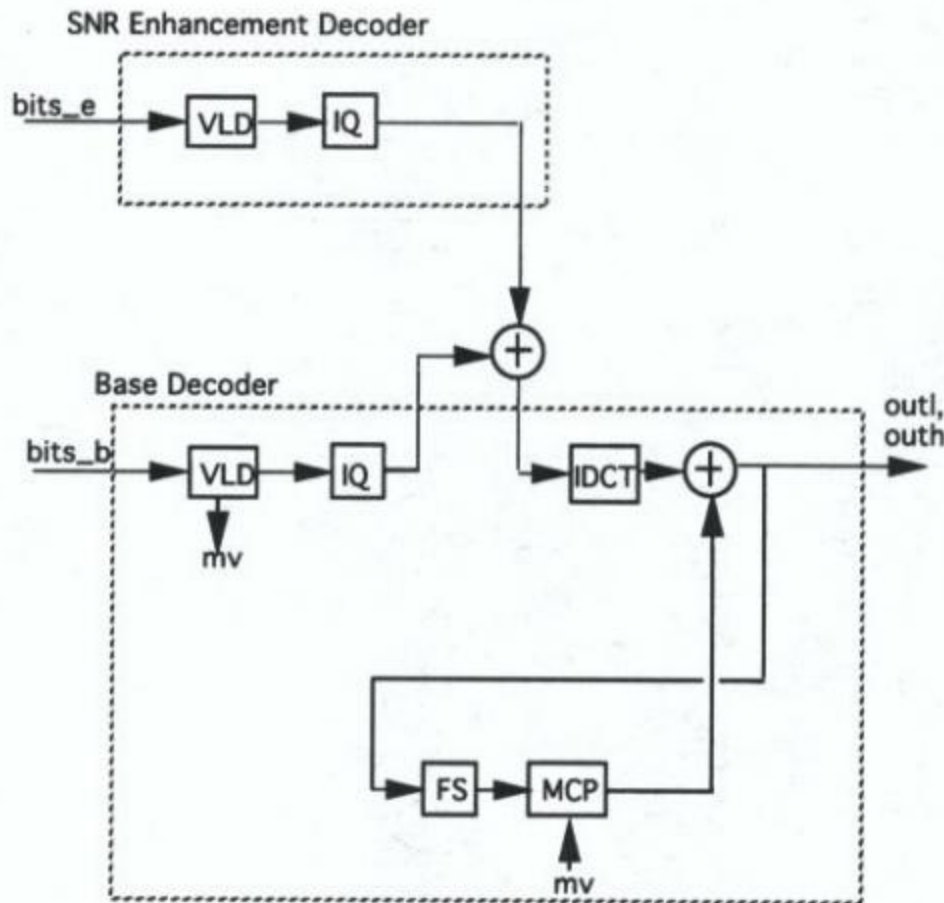


Figure 26. Decoder structure of MPEG-2 [7]

Unlike its predecessors, MPEG-4 [16] coding did not remain confined to the domain of rectangular-sized pictures but adopted an object based coding concept in which arbitrarily shaped and dynamically changing individual audio-visual objects in a video sequence can be individually encoded, manipulated and transmitted through an independent bit-stream. It was standardized to address a wide range of bit-rates- from very low bit rate coding (5-64 Kilobits/sec) to 2 megabits/sec for TV/film applications. In recent times, MPEG-4 part 10 visual has found widespread applications in internet streaming, wireless video, digital video cameras as well as in mobile phones and mobile palm computers [1].

Apart from the MPEG, the International Telecommunication Union- Telecommunications Standardization Sector (ITU-T) also evolved the standards for multimedia communications at restricted bit-rate over the wireline and wireless channels. The ITU-T standardization on multimedia first started with H.261 [34], which was developed for ISDN video conferencing. The next standard H.263 [33] supported Plain Old Telephone Systems (POTS) conferencing at very low bit-rates (64 Kbits/sec and lower).

The need for further improvement in coding efficiency by at least two times for the same fidelity was soon realized. In 1998, the Video Coding Experts Group (VCEG) of the ITU invited proposals for a new video coding project, named H.26L which would have two times better coding efficiency over a broad range of applications. In December 2001, the VCEG and the Moving Pictures Experts Group (MPEG)

formed a Joint Video Team (JVT). Their combined efforts resulted in the new coding standard H.264. This also forms the Part-10 (Advanced Video Coding) of MPEG-4 and is therefore referred to as H.264 / AVC standard.

Some of the major highlighting features of this H.264 coding standard are improved motion estimation up to quarter-pixel accuracy; use of 4 x 4 integer transforms in place of 8 x 8 DCT; improved context based arithmetic entropy coding; advanced prediction modes for intra and inter-coded frames etc. The H.264 [16] standard was designed for enhanced compression performance with network friendly features to address a broad range of applications that include conversational (e.g. Video telephony and Video conferencing) and non-conversational (e.g. storage, broadcast and streaming) applications. The encoder and decoder structures of H.264 are shown in figures 27 and 28 respectively.

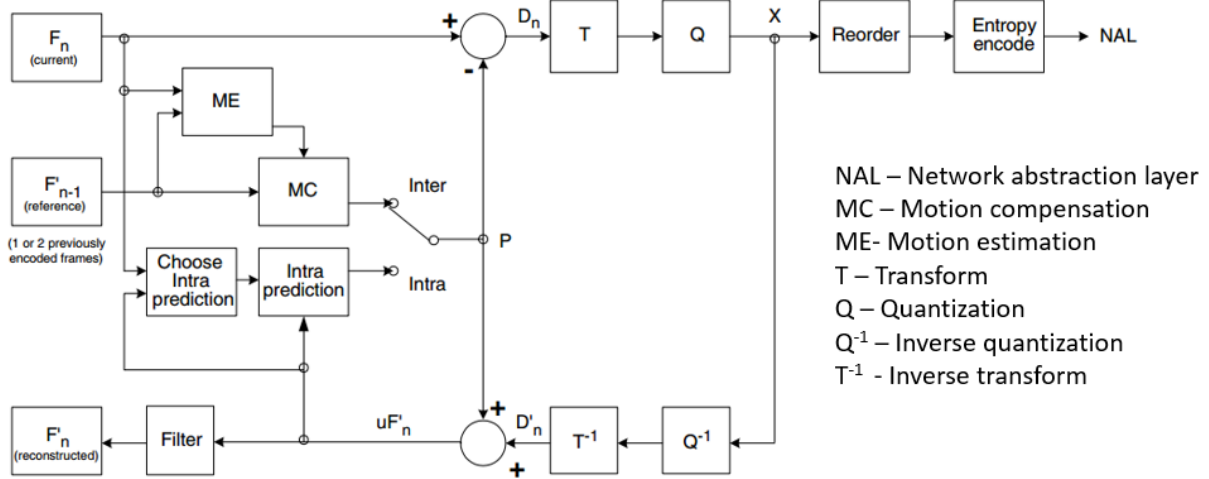


Figure 27. Encoder structure of H.264 [3]

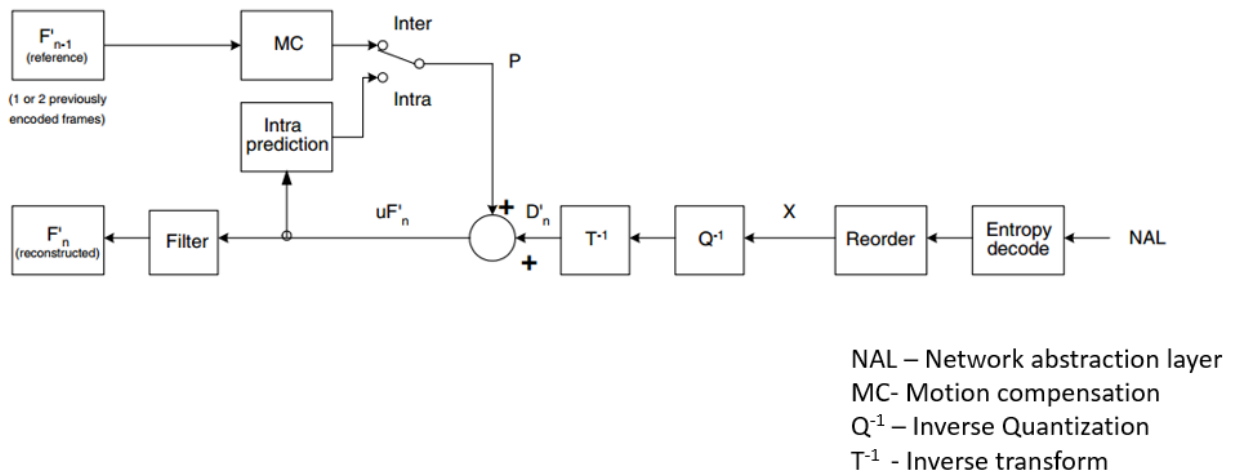


Figure 28. Decoder structure of H.264 [3]

High Efficiency Video Coding Standard (HEVC) was developed after H.264 to provide support for 4K and 8K videos and to support parallel processing architectures. This is discussed in detail in chapter 3.

Chapter 3 – High Efficiency Video Coding (HEVC) Standard

The High Efficiency Video Coding (HEVC) [8] is the latest video standard developed by Joint Collaborative Team on Video Coding (JCT-VC), a group of video coding experts from ITU-T Video Coding Experts Group and SO/IEC Moving Picture Experts Group (MPEG).

As the demand for HD video (4K and 8K) increased, there is a need for stronger coding efficiency than H.264/AVC. Also, there is increased use of parallel processors. So, HEVC [8] has been introduced to support increased video resolution and parallel processing. HEVC obtains about 50% reduction in bit rate when compared to its predecessor H.264/AVC at the same visual quality.

3.1 HEVC encoder and decoder:

The block diagrams for HEVC encoder and decoder are shown in figures 29 and 30 respectively. The input video is split into multiple Coding Tree Units (CTUs) which are predicted using intra/inter prediction. The residual image is transformed, scaled, quantized and then entropy coded.

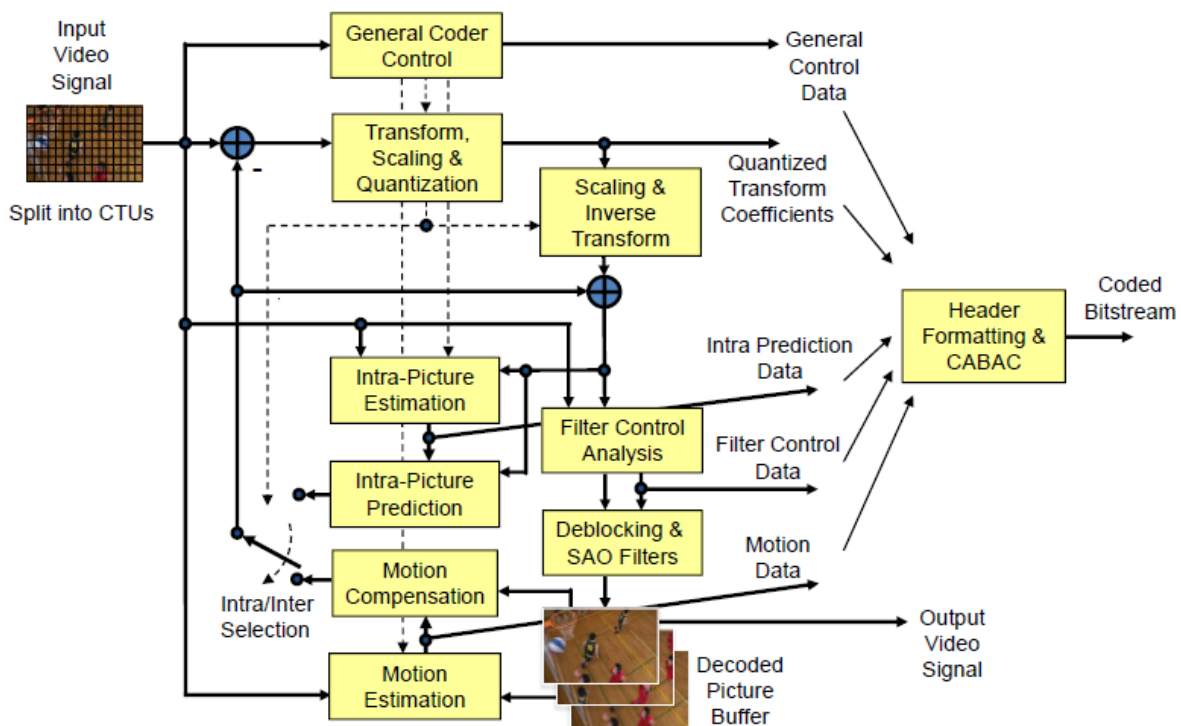


Figure 29. Block Diagram of HEVC Encoder [8].

Similarly, the decoder performs entropy decoding followed by rescaling, inverse transform and then prediction unit is added to reconstruct the video. This process is clearly depicted in Figure 30.

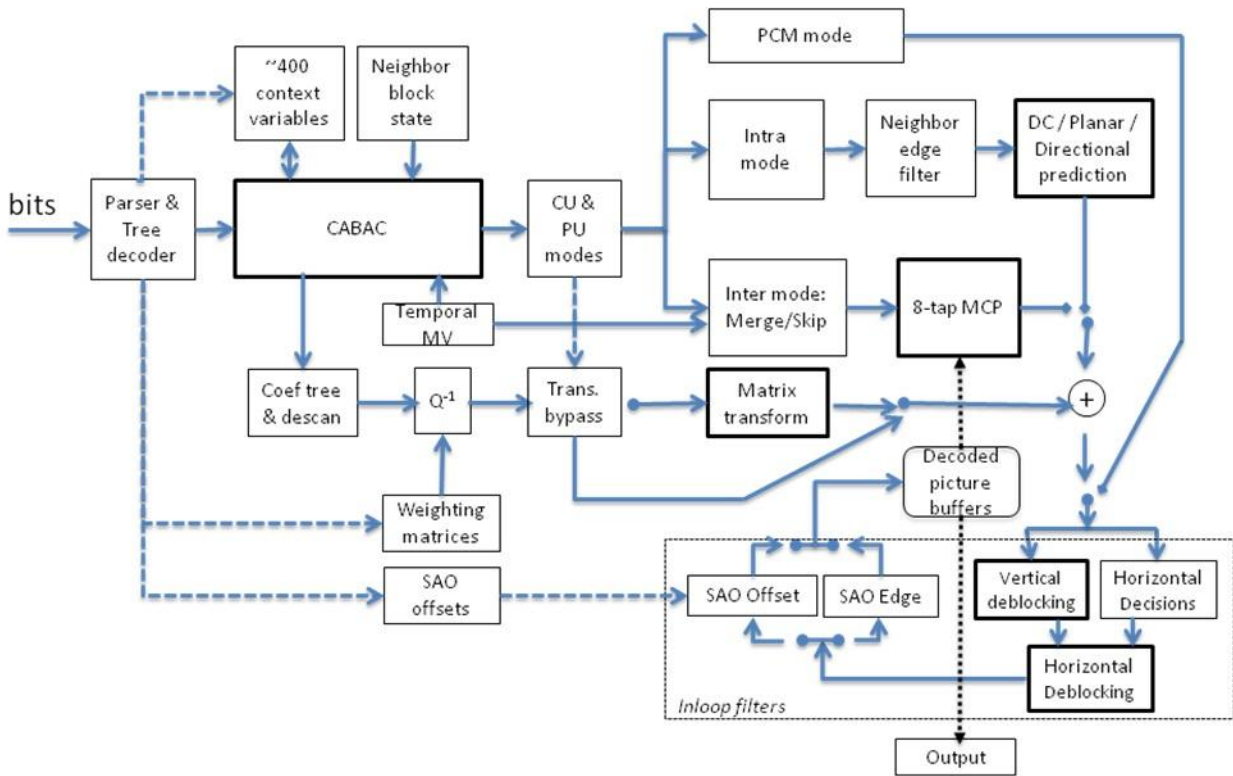
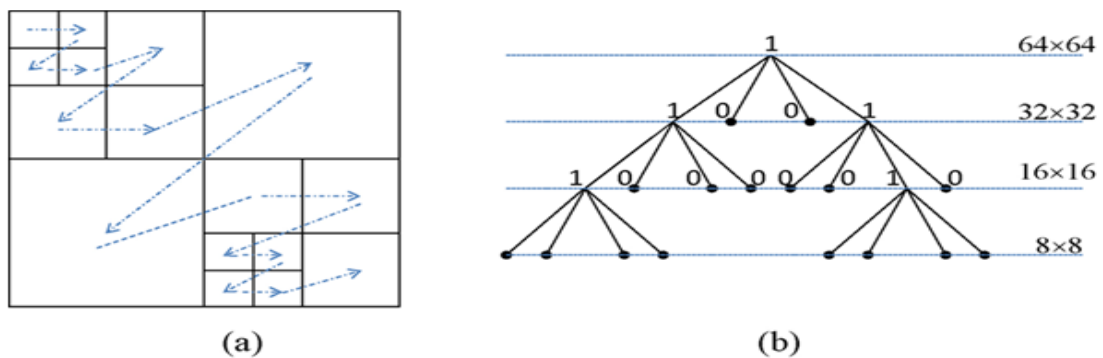


Figure 30. Block Diagram for HEVC Decoder [14].

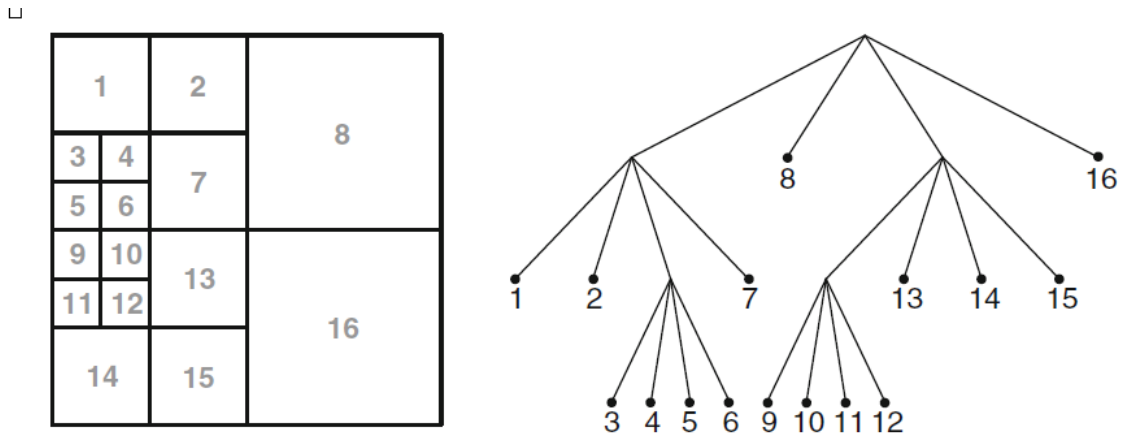
3.2 Block Structures in HEVC

HEVC follows quad-tree block partitioning for improved prediction and transform coding. Each picture is divided into multiple Code Tree Units (CTUs) and they are further sub-divided into multiple Coding Units (CUs) as shown in the figures 31 and 32 and these are inter/intra predicted.



Example of CTU, partitioning and processing order when size of CTU is equal 64×64 and minimum CU size is equal to 8×8 (a) CTU partitioning (b) Corresponding coding tree structure.

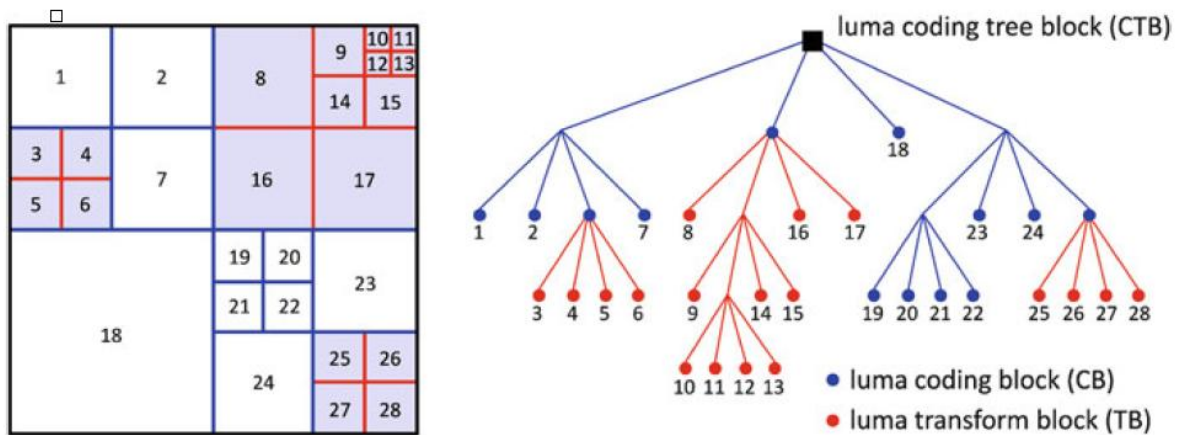
Figure 31. CTU partitioning and quad tree structure [11].



Example for the partitioning of a 64×64 coding tree unit (CTU) into coding units (CUs) of 8×8 to 32×32 luma samples. The partitioning can be described by a quadtree, also referred to as coding tree, which is shown on the *right*. The numbers indicate the coding order of the CUs

Figure 32. 64x64 CTU partitioning using quad-tree structure [9]

An example describing division of luma CTB into luma Code Blocks (CBs) and luma Transform Blocks (TBs) is shown in figure 33.



Example for the partitioning of a 64×64 luma coding tree block (*black*) into coding blocks (*blue*) and transform blocks (*red*). In the illustration on the *right*, the *blue lines* show the corresponding coding tree with the coding tree block (*black square*) at its root and the coding blocks (*blue circles*) at its leaf nodes; the *red lines* show the non-degenerated residual quadtrees with the transform blocks (*red circles*) as leaf nodes. Note that the transform blocks chosen identical to the corresponding coding blocks are not explicitly marked in this figure. The numbers indicate the coding order of the transform blocks

Figure 33. Division of a 64x64 luma CTB using quad-tree structure [9]

In H.264, the picture is divided in to 16×16 size macro blocks and it is further sub divided. But, HEVC supports block sizes up to 64×64 as shown in the figure 34.

Video coding standard	Supported block sizes for motion-compensated prediction
H.262 MPEG-2 Video	16×16
H.263	$16 \times 16, 8 \times 8$
MPEG-4 Visual	$16 \times 16, 8 \times 8$
H.264 MPEG-4 AVC	$16 \times 16, 16 \times 8, 8 \times 16, 8 \times 8, 8 \times 4, 4 \times 8, 4 \times 4$
HEVC	$64 \times 64, 64 \times 48, 64 \times 32, 64 \times 16, 48 \times 64, 32 \times 64, 16 \times 64, 32 \times 32, 32 \times 24, 32 \times 16, 32 \times 8, 24 \times 32, 16 \times 32, 8 \times 32, 16 \times 16, 16 \times 12, 16 \times 8, 16 \times 4, 12 \times 16, 8 \times 16, 4 \times 16, 8 \times 8, 8 \times 4, 4 \times 8$

Figure 34. Comparison of motion compensation block sizes supported in different standards [9]

3.3 Parallelism in HEVC

In HEVC, a picture is sub-divided into slices or tiles to support parallel processing. Slices and tiles consist of a sequence of CTUs as shown in the figures 35 and 36 respectively.

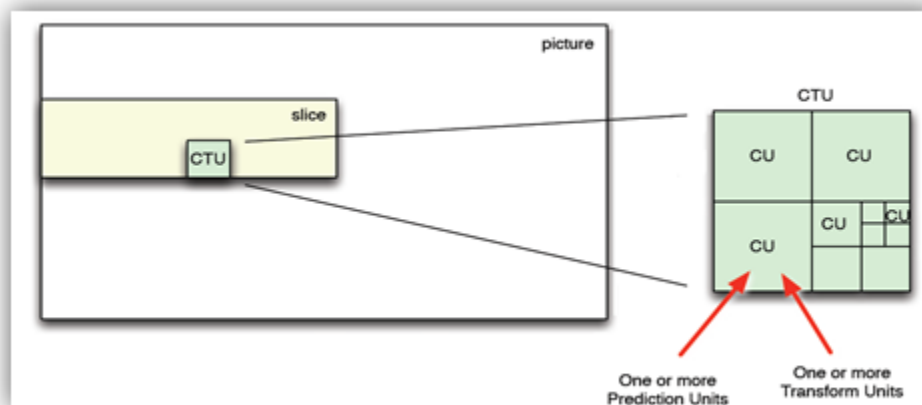
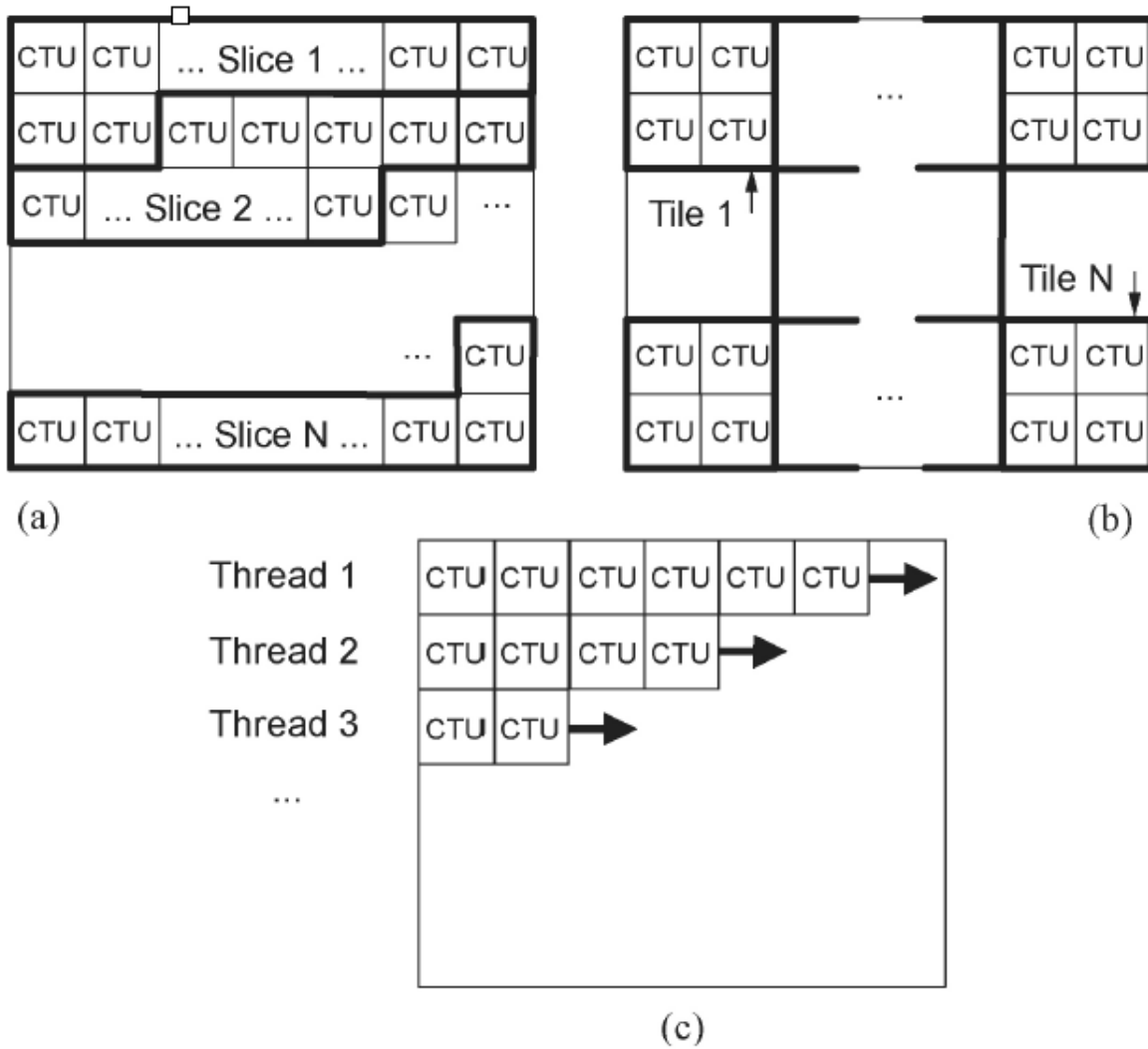


Figure 35. Picture, Slice, Coding Tree Unit (CTU), Coding Unit (CU) [10].



Subdivision of a picture into (a) slices and (b) tiles. (c) Illustration of wavefront parallel processing.

Figure 36. Slices and tiles in HEVC [8]

3.4 Prediction in HEVC

In HEVC, frames are intra or inter predicted.

Intra-prediction: Each prediction unit is predicted from the same picture using 35 modes (33 angular, planar and DC) as shown in figure 37.

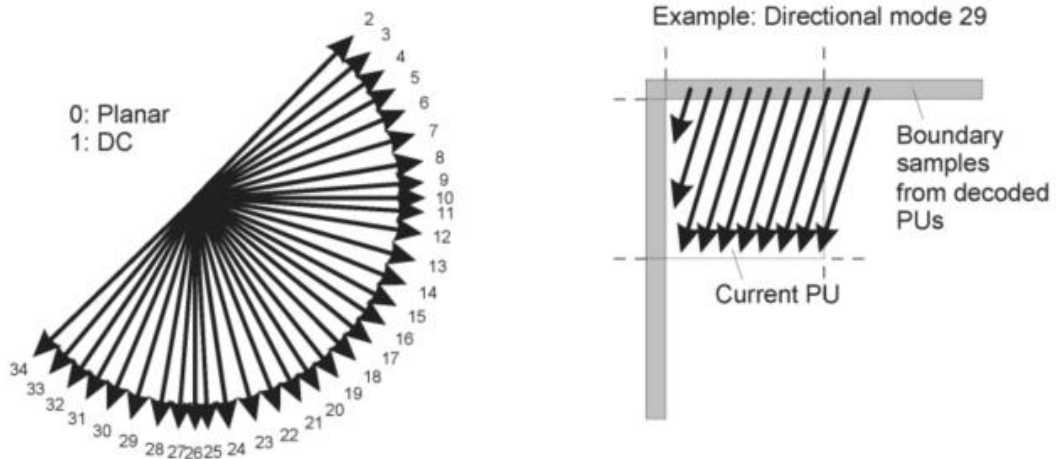


Figure 37. Modes and directional orientations for intra prediction in HEVC [8]

Inter-prediction: Each prediction unit is predicted from neighboring picture data using motion compensated prediction [12] [13] as shown in figure 38.

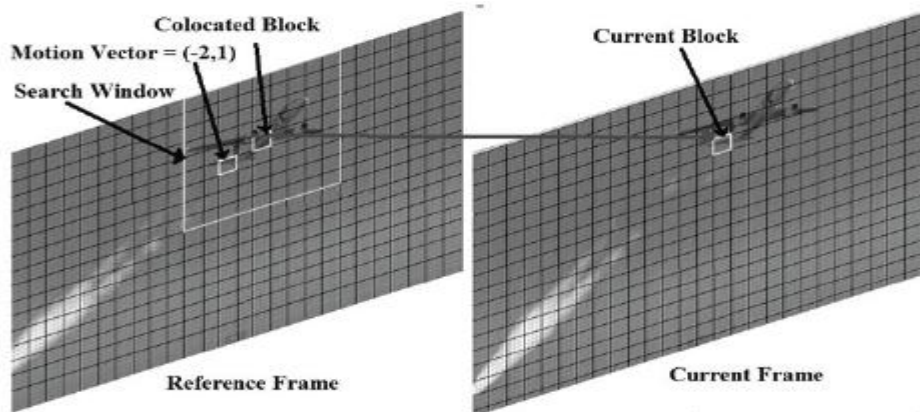


Figure 38. Illustration of Motion Estimation Process [12]

Further, HEVC uses 7-tap or 8-tap filters for fractional sample interpolation (up to quarter-sample precision) whereas H.264 uses 6-tap filter for half-sample precision and linear interpolation for quarter-sample precision. Figure 39 shows integer and fractional sample positions for luma interpolation.

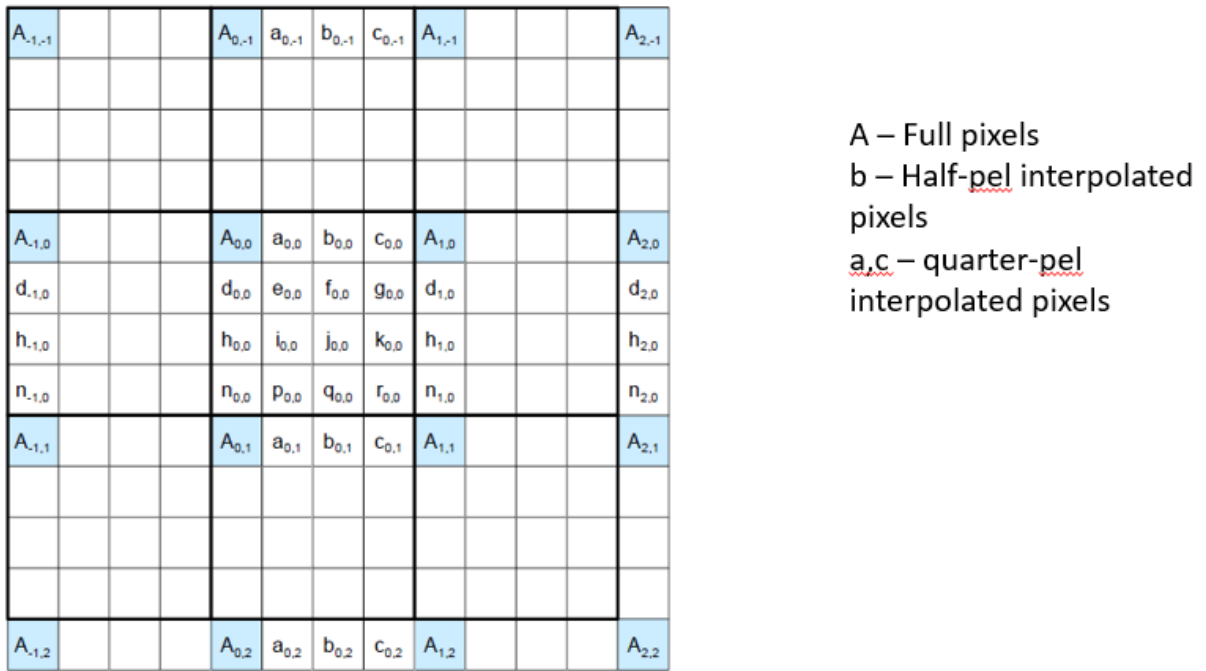


Figure 39. Integer and fractional sample positions for luma interpolation [8]

Further, filter coefficients for luma and chroma fractional sample interpolation are shown in figure 40.

Filter coefficients for luma fractional sample interpolation

index	-3	-2	-1	0	1	2	3	4
hfilter[i]	-1	4	-11	40	40	-11	4	1
qfilter[i]	-1	4	-10	58	17	-5	1	

Filter coefficients for chroma fractional sample interpolation

index	-1	0	1	2
filter1[i]	-2	58	10	-2
filter2[i]	-4	54	16	-2
filter3[i]	-6	46	28	-4
filter4[i]	-4	36	36	-4

Figure 40. Filter coefficients for luma and chroma fractional sample interpolation [8]

3.5 Transform and Quantization

Residual CU is transformed by using block transforms such as integer DCT of sizes 32x32, 16x16, 8x8 and 4x4 and then the transformed data is quantized [10] in HEVC.

3.6 Entropy Coding

Context Adaptive Binary Arithmetic Coding (CABAC) is used to encode quantized transform coefficients and motion vector data [10] in HEVC.

3.7 In-loop Filters

HEVC employs in-loop filters such as deblocking [15] and Sample Adaptive Offset (SAO) [9] as shown in figure 29, while H.264 employs only deblocking filter. Deblocking filter removes block discontinuities due to transform or prediction at block boundaries. SAO filter reduces ringing artifacts.

Chapter 4 – Artifacts due to Image and Video Compression

Block based transforms are widely used in image and video compression standards (JPEG, MPEG, H.261, H.263, H.264 and HEVC) as they have energy compaction and low computational complexity. When the quantization is done in a very coarse way in these standards while encoding an image or a video, to reduce the bit-rate, visual artifacts occur. An example of highly compressed image block is shown in the figure 41.

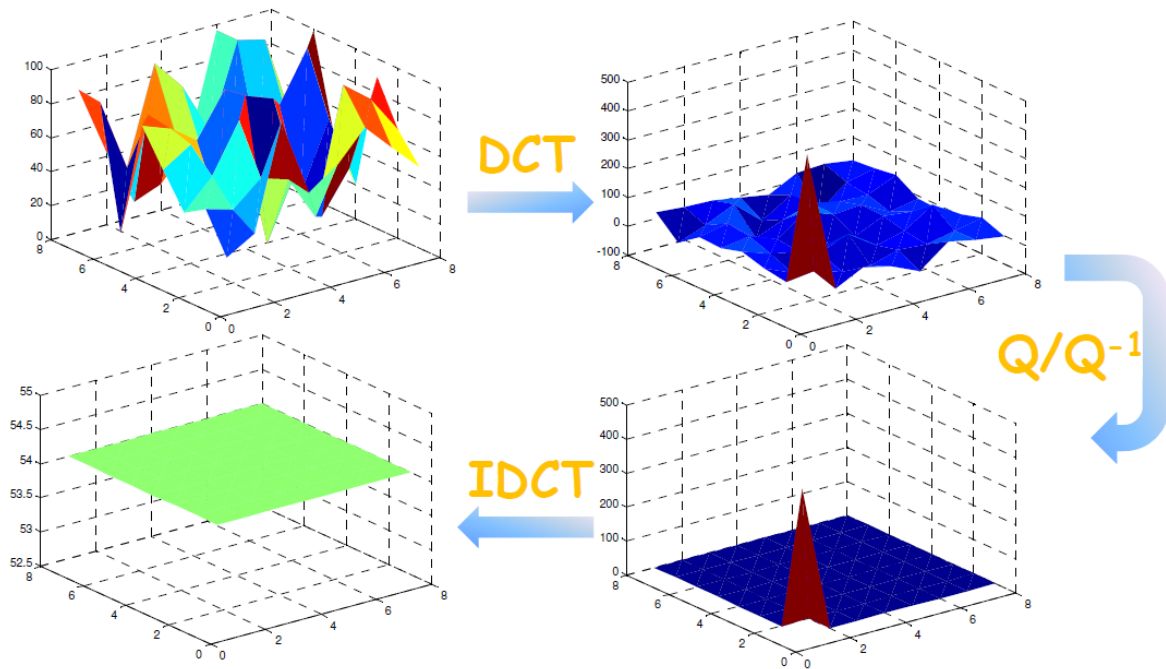


Figure 41. Highly compressed image block [17]

4.1 Different types of artifacts

The most common artifact is blocking artifact, which occurs due to block discontinuities at prediction or transform block boundary which is clearly illustrated in figures 42 and 43.



(a) The original image (b) The highly compressed image

Figure 42. Blocking artifacts due to block based image transform [17]

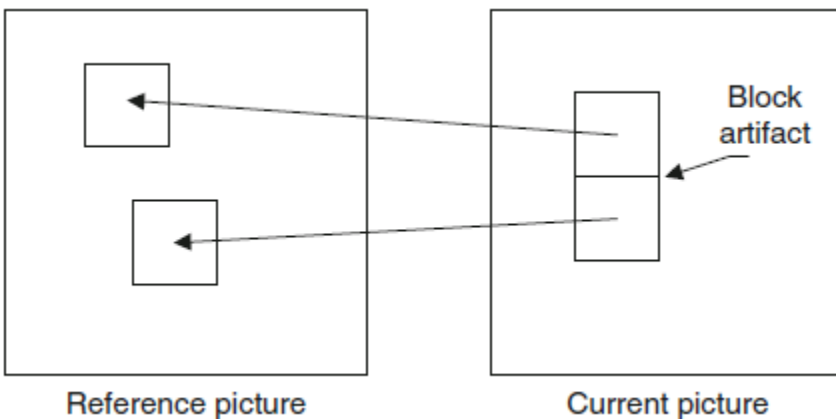


Figure 43. Blocking artifact resulting from predictive coding [9]

Blurring of image occurs due to the loss of spatial detail in moderate to high spatial activity regions of pictures. For intra-frame coded macroblocks, blurring occurs due to suppression of higher order AC coefficients, leaving only the lower order coefficients to represent the contents of a block. For predictive-coded macroblocks, blurring is mainly due to the use of a predicted macroblock with a lack of spatial detail [18].

Ringling effect is mainly due to the poor energy compaction of DCT for horizontal and vertical edges for coarse quantization. As edges contain mostly high frequency components, coarse quantization leads to the suppression of these high frequency components, thereby causes the edges to exhibit ringing phenomenon (shown in figures 44 and 45) during reconstruction of the image.



Figure 44. Example of the ringing effect, where it is most evident around the bright table-edge and the boundary of the arm [18]



Figure 45. Example of the ringing effect in a sequence coded at a relatively high bit-rate. Most prominent along the edge formed by the upper-arm in the scene [16]

The most common artifacts are blocking, ringing and blurring (Figure 46). Apart from these other types of artifacts include mosaic pattern effect, color bleeding, staircase effect, basis image effect, false contouring, false edges, mosquito effect, chrominance mismatch, etc.

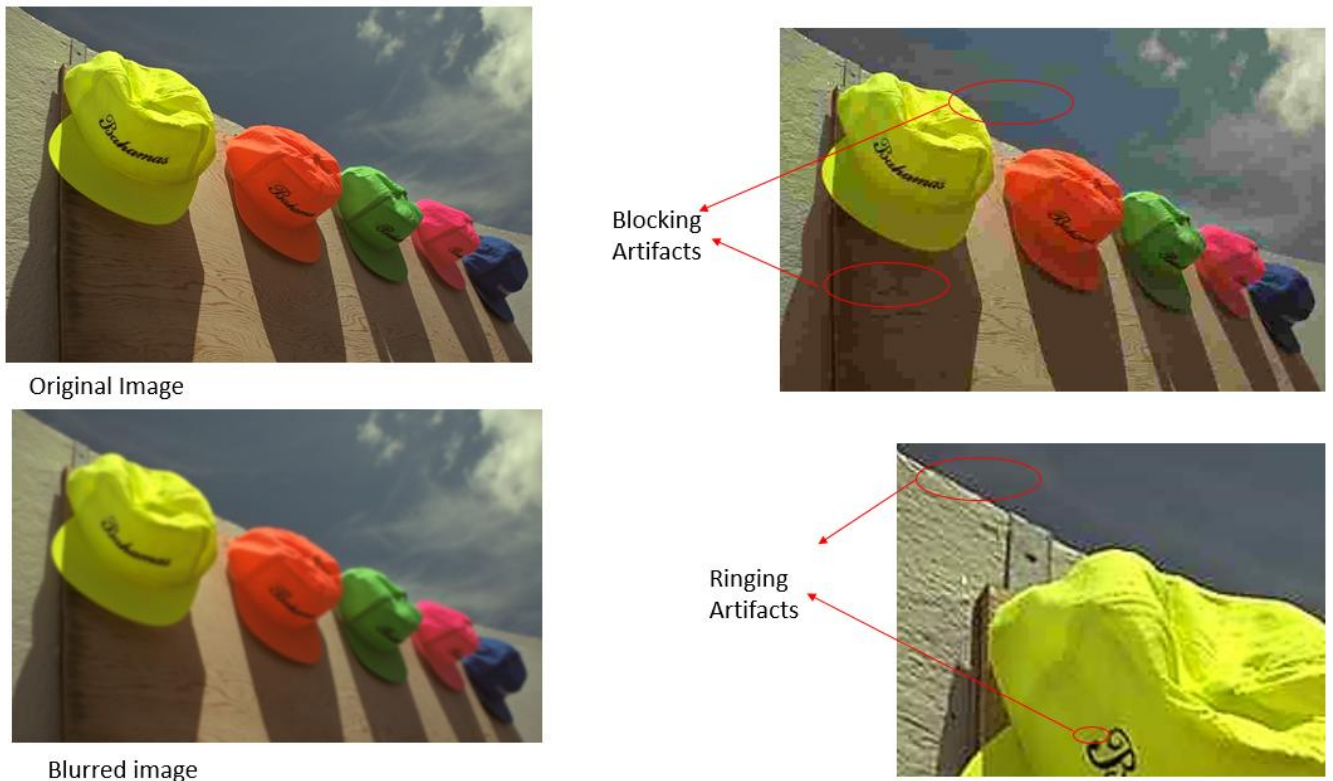


Figure 46. Common artifacts due to block based coding.

4.2 Reduction of visual artifacts

Visual artifacts can be removed with the help of in-loop filters, pre-processing, post-processing and overlapped block methods.

In-loop filtering methods insert filters into the encoding and decoding loops of the video codec. H.264 employs in-loop deblocking filter and HEVC employs deblocking as well as SAO in-loop filters.

The overlapped block methods include lapped transform, which is included as a part of encoding process and hence modifies the codec.

The pre-processing method pre-processes the image so that compression artifacts are reduced under low bit-rate. Post-processing methods apply low pass filters and algorithms after the image or video has been decoded to improve the visual quality.

Pre-processing and post-processing algorithms became more popular as they do not modify the codec. This thesis mainly focuses on designing a post-processing filter for HEVC standard to remove visual artifacts.

Post-processing methods are mainly categorized into two types: enhancement based and restoration based techniques. Enhancement based techniques try to improve perceptual quality without an explicit

optimization process while restoration techniques recover original image based on some optimization criteria. This thesis mainly focusses on an enhancement technique using bilateral filter.

4.3 Bilateral filter

A bilateral filter [19] does a weighted spatial averaging and these weights depend on both the spatial and intensity distances. So, by this way, edges can be preserved while smoothing the image. The main disadvantage with Gaussian smoothing is that it does not preserve edges as shown in figure 47.

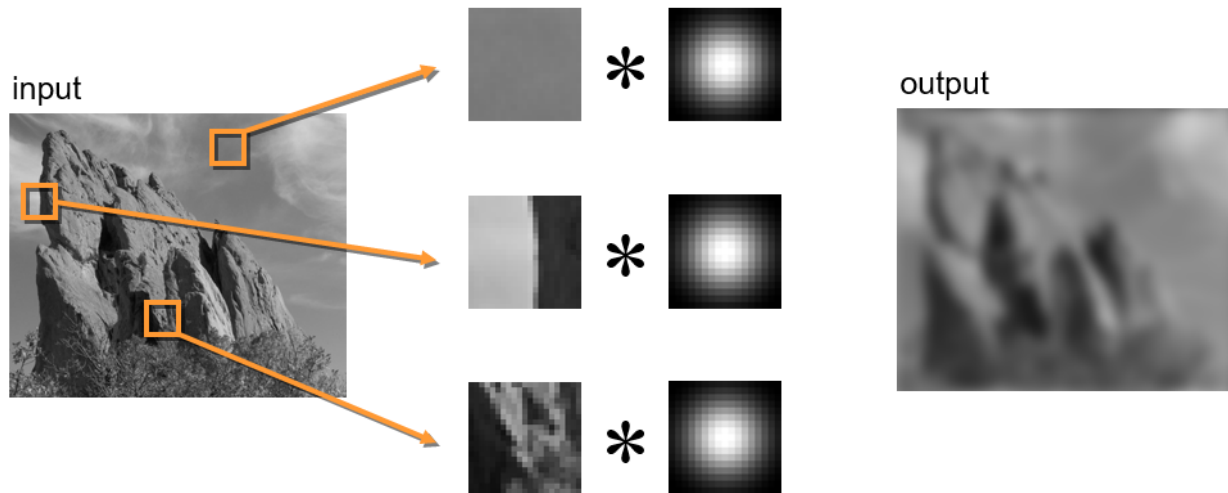


Figure 47. Gaussian smoothing [20]

We notice in figure 47 that the kernel is fixed and edges are not preserved in the output image. But, if we apply bilateral filter on the same image, we obtain the output as shown in the figure 48.

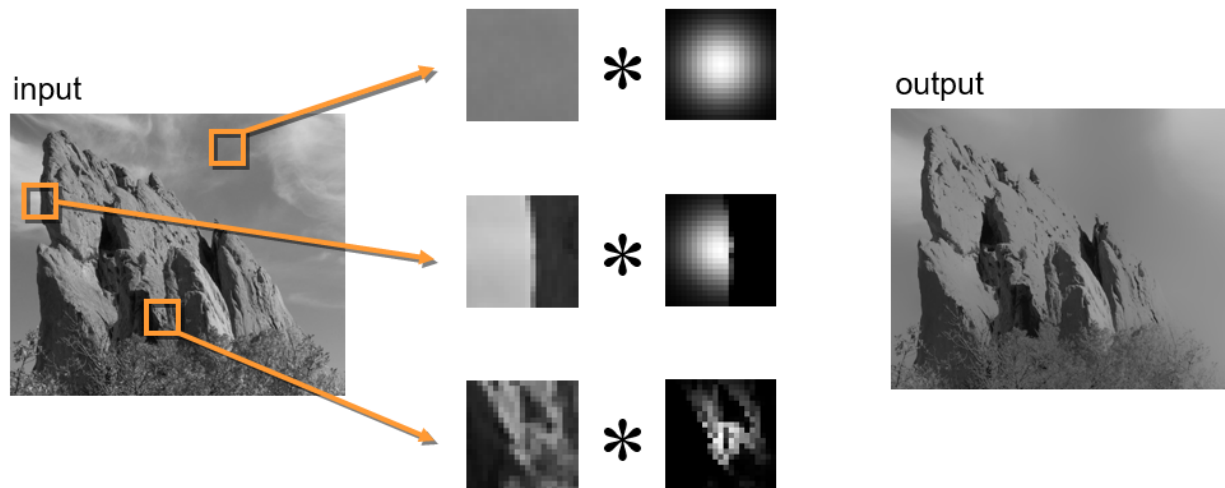


Figure 48. Bilateral Filtering [20]

We notice in figure 48 that, the kernel size depends on the image content and edges are preserved. A Gaussian filter smoothens edges as it does averaging of pixels across the edges while bilateral filter does not do averaging of pixels across its edges.

Gaussian blurred image is obtained by applying the 2D Gaussian function at each pixel in the image.

The 2D-Gaussian Function is,

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- σ is the standard deviation.

As the size of the kernel or σ increases, the image is strongly smoothed and it is shown in figure 50.



Figure 49. Input image

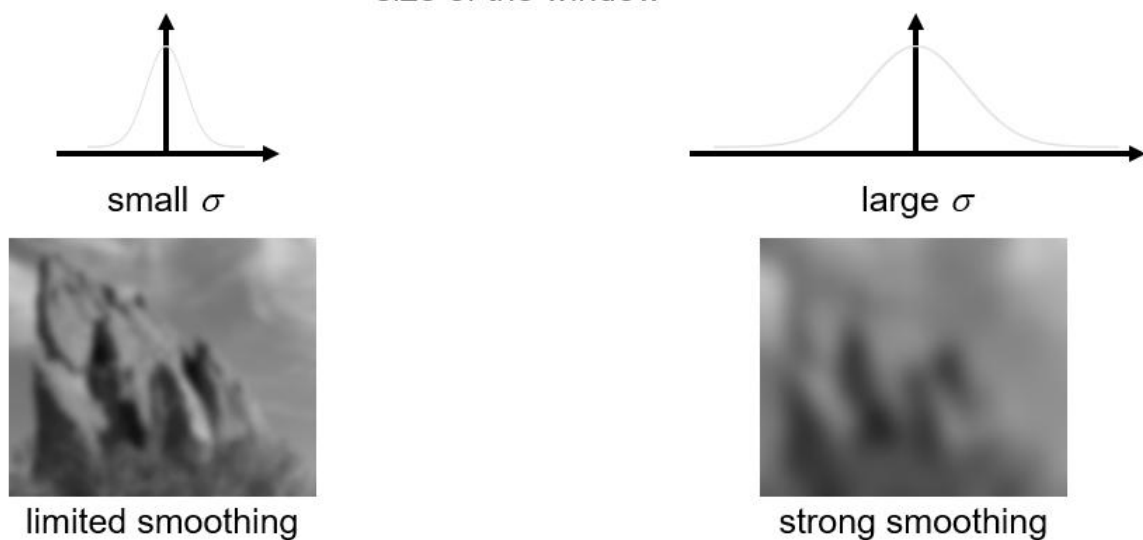


Figure 50. Limited and strong smoothing [20]

Hence for the Gaussian filter, only spatial distance matters and no special case is considered for edges. Whereas bilateral filter has an additional edge term as shown in figure 51.

At a pixel location $\mathbf{x} = (x_1, x_2)$, the output of a bilateral filter,

$$\tilde{I}(\mathbf{x}) = \frac{1}{C} \sum_{\mathbf{y} \in \mathcal{N}(\mathbf{x})} \exp\left[-\frac{\|\mathbf{y}-\mathbf{x}\|^2}{2\sigma_d^2}\right] \exp\left[-\frac{|I(\mathbf{y})-I(\mathbf{x})|^2}{2\sigma_r^2}\right] I(\mathbf{y}),$$

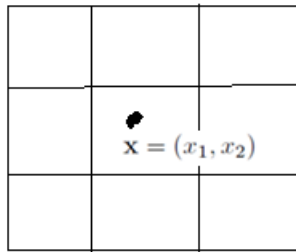
$$\text{Where, } C = \sum_{\mathbf{y} \in \mathcal{N}(\mathbf{x})} \exp\left[-\frac{\|\mathbf{y}-\mathbf{x}\|^2}{2\sigma_d^2}\right] \exp\left[-\frac{|I(\mathbf{y})-I(\mathbf{x})|^2}{2\sigma_r^2}\right]$$

σ_d and σ_r are parameters controlling the fall-off of weights in spatial and intensity domains.

$\mathcal{N}(\mathbf{x})$ is a spatial neighborhood of pixel $I(\mathbf{x})$

$\|\mathbf{x}\|$ is absolute value of \mathbf{x} , and

$\|\mathbf{X}\|$ is the norm of $\mathbf{X}(x_1, x_2)$ and it is the distance from origin (0,0) and is given as $\text{Sqrt}(x_1^2+x_2^2)$



$\mathcal{N}(\mathbf{x})$ is 3x3 neighborhood around \mathbf{x}

Figure 51. Bilateral filter [20]

The application of bilateral filter for an input image for varying space and range parameters is shown in figure 53. (x_1, x_2)

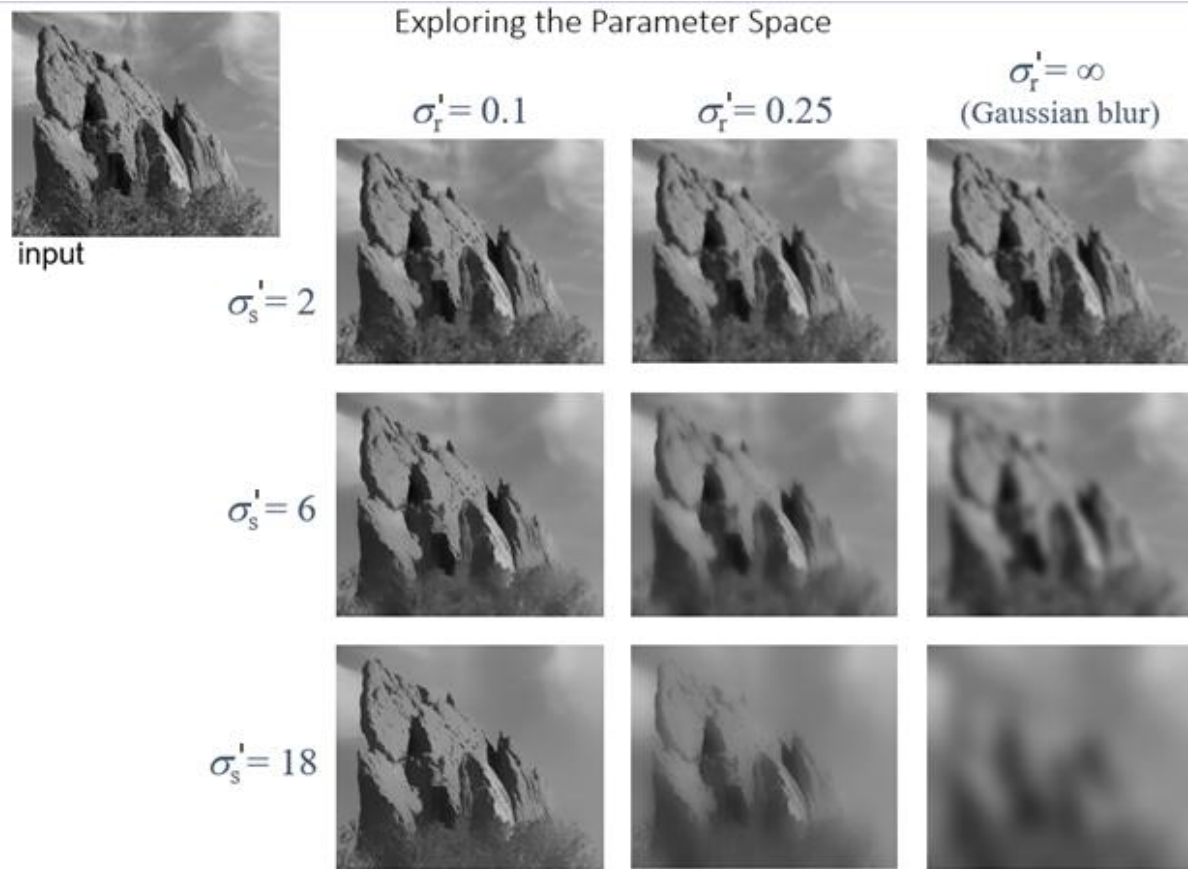


Figure 52. Exploring parameter space for bilateral filter [20]

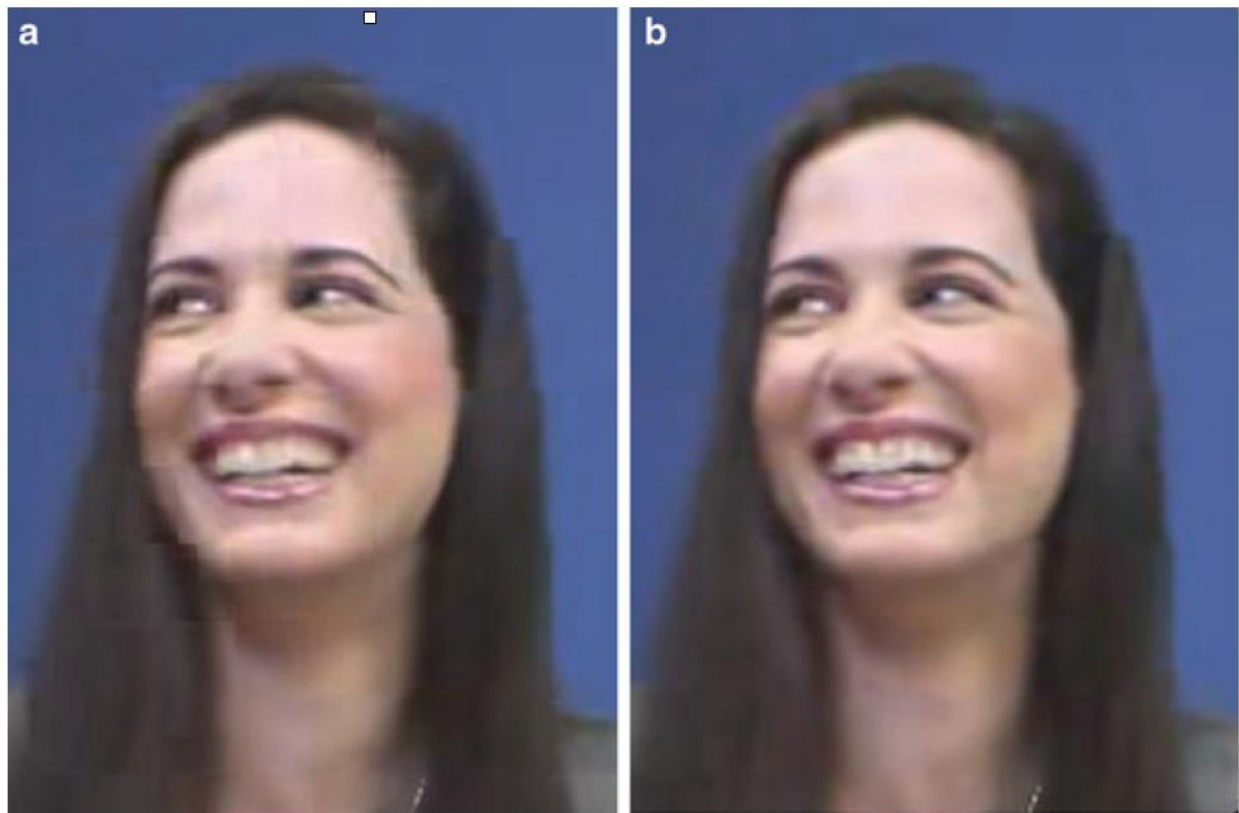
4.4 Reduction of Compression Artifacts in HEVC through in-loop Filters

HEVC has in-loop filters (deblocking filter and SAO filter) which reduce compression artifacts due to blocking and ringing. The deblocking filter is applied first and then SAO filter is applied later for the decoded video sequences. Figures 54 and 55 show the function of deblocking in HEVC and we can clearly see that blocking artifacts are greatly reduced.



Sequence *BasketballDrive*, Random Access, QP32: (a) deblocking turned off, (b) deblocking turned on

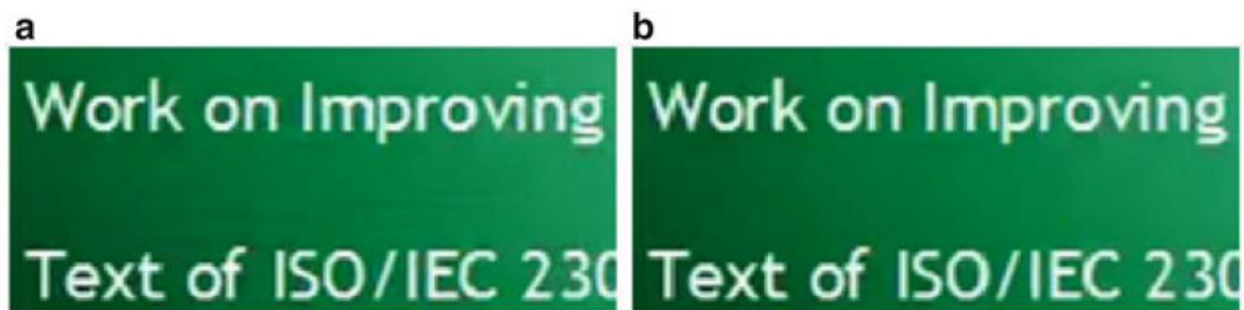
Figure 53. Performance of Deblocking filter in HEVC for Basketball Drive Sequence [9]



Sequence *KristenAndSara*, Low Delay, QP37: (a) deblocking turned off, (b) deblocking turned on

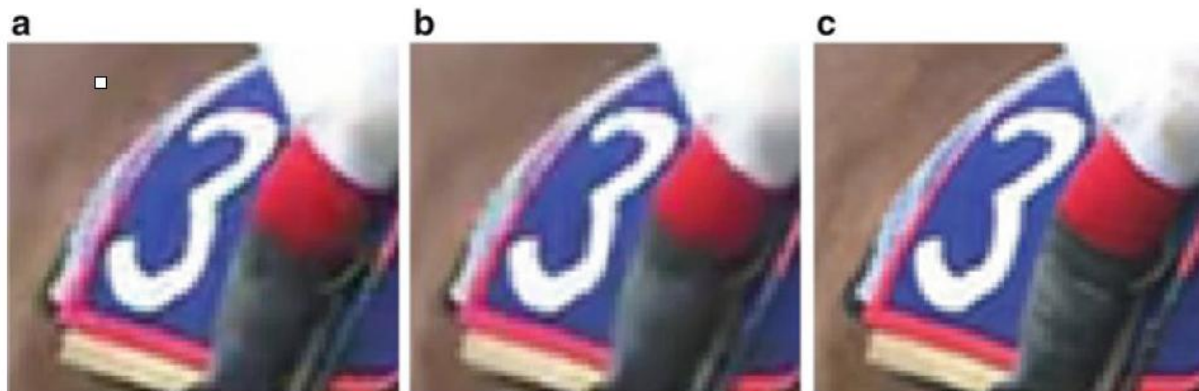
Figure 54. Performance of Deblocking filter in HEVC for KristenAndSara Drive Sequence [9]

Similarly, performance of HEVC SAO filter is shown in figures 56 and 57.



Example of test sequence *SliceEditing* in LP condition, POC = 100, QP = 32: (a) SAO is disabled, (b) SAO is enabled

Figure 55. Performance of SAO filter in HEVC on *SliceEditing* sequence [9]



Subjective quality comparison of *RaceHorses* test sequence, POC = 20, QP = 32, LP condition: (a) SAO is disabled, (b) SAO is enabled, (c) original (uncoded) sequence

Figure 56. Performance of SAO filter in HEVC on *RaceHorses* sequence [9]

As, we can see from the figures 56 and 57, ringing artifacts are slightly reduced.

Even though HEVC greatly reduces the compression artifacts, there is still scope for improvement such as ringing artifacts and this is the motivation toward my thesis.

Chapter 5 – Adaptive Bilateral Filtering to Remove Compression Artifacts

Though HEVC employs in-loop filters such as deblocking and SAO filters to remove compression artifacts due to block based coding and coarse quantization of transform coefficients, there is still a scope for improvement. As in-loop filters are part of the standard, any modification of in-loop filters would modify the coding standard. So, post-processing techniques have gained popularity as they would not disturb the existing standard and reduce compression artifacts.

The current thesis focusses on applying bilateral filter on HEVC decoded frames adaptively to remove compression artifacts and still maintain very good visual quality.

The bilateral filter discussed in chapter 4.3 is applied at each pixel of the frame. The bilateral filter preserves the edges while smoothing the image. To apply bilateral filter in an adaptive way, we divide the frame into 4x4 blocks and calculate sigma for spatial (σ_s^I) and intensity domains (σ_r^I). First, each 4x4 block is categorized into a strong edge, weak edge, texture or smooth block. This is done by determining the standard deviation (STD) in a 4x4 block around each pixel [21] and then spatial sigma (σ_s^I) is determined with a set of predetermined thresholds as shown in figure 58 [21].

$$\sigma_s^I = \begin{cases} StrongEdge, \sigma_s^I = 0.8 & \text{if MaxSTD} \in [35, \infty) \\ WeakEdge, \sigma_s^I = 1.8 & \text{if MaxSTD} \in [25, 35) \\ Texture, \sigma_s^I = 2.8 & \text{if MaxSTD} \in [15, 25) \\ Smooth, \sigma_s^I = 3.8 & \text{if MaxSTD} \in [0, 15) \end{cases}$$

Figure 57. Parameter controlling fall-off weight in spatial domain.

To compute the sigma in the intensity domain, the HEVC decoded frame is filtered with $[-1, 0, 1]$ and $[-1, 0, 1]^T$ to detect vertical and horizontal block discontinuities respectively and then their absolute values are taken at the block boundaries. Figure 59 shows an example of 8x8 block where block discontinuities are shown in colors apart from grey and white.

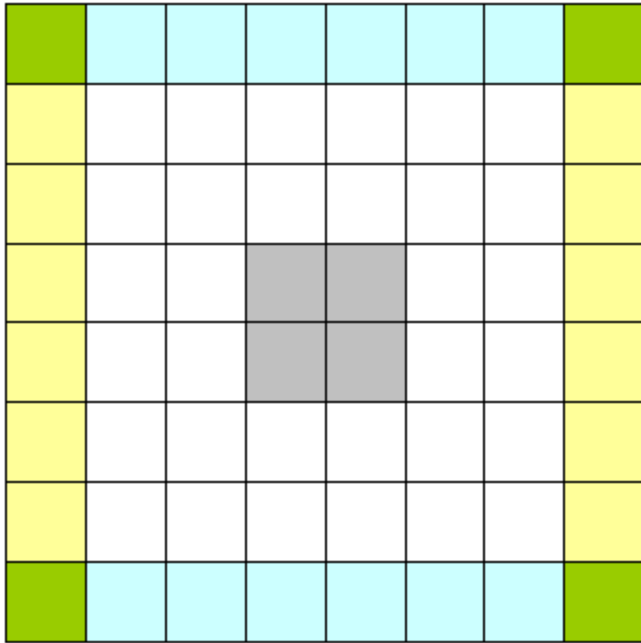


Figure 58. Block discontinuity Map for 8x8 pixel block [22]

Applying block discontinuities on the edges does not remove compression artifacts completely and hence it is applied on all the pixels by taking edge discontinuities along the block boundary and interpolating remainder of pixels using bi-linear interpolation. The center 4 gray pixels are marked zero before interpolation. After interpolating all the pixels in 8x8 block we get 8x8 discontinuity map. These values are used as sigma in the intensity domain. Hence using the sigma values in the spatial and intensity domain for each pixel, bilateral filter is applied and the results are shown in chapter 6.

Chapter 6 – Results

Different test sequences with various resolutions are used with varying Quantization Parameter (QP). This thesis mainly focusses on PSNR (Peak Signal to Noise Ratio) and time to apply the in-loop (SAO and deblocking) and post processing bilateral filters.

$$\text{For 4:2:0 video, YUV-PSNR} = \frac{6 \cdot \text{Y-PSNR} + \text{U-PSNR} + \text{V-PSNR}}{8}$$

Quantization Parameter (QP)	YUV-PSNR (dB), when deblocking and SAO are enabled in HEVC and NO post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are disabled in HEVC and NO post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are disabled in HEVC and post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are enabled in HEVC and post processing bilateral filter is used.
35	32.23	31.95	32.14	32.19
45	26.88	26.62	26.76	26.78

Table 1. BasketballDrillText_832x480_50.yuv sequence quality metrics

Quantization Parameter (QP)	Time in seconds for in-loop filters in HEVC	Time in seconds for bilateral post processing filter for HEVC	Time in seconds for both In-loop filter and bilateral post processing filter for HEVC
35	88	64	152
45	108	63	153

Table 2. BasketballDrillText_832x480_50.yuv filter time


Further visual results are shown in figures 60, 61, 62, 63 and 64 for QP = 35 (Only a part of the frame 1 is shown to better visualize compression artifacts)

 PYUV - BasketballDrillText_832x480_50.yuv

File Control View Settings Help




Figure 59. Original BasketballDrillText_832x480_50.yuv sequence frame1

 PYUV - bask_filtersOff_35.yuv

File Control View Settings Help



Figure 60. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter disabled.

 PYUV - bask_filtersOn_35.yuv

File Control View Settings Help



Figure 61. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.

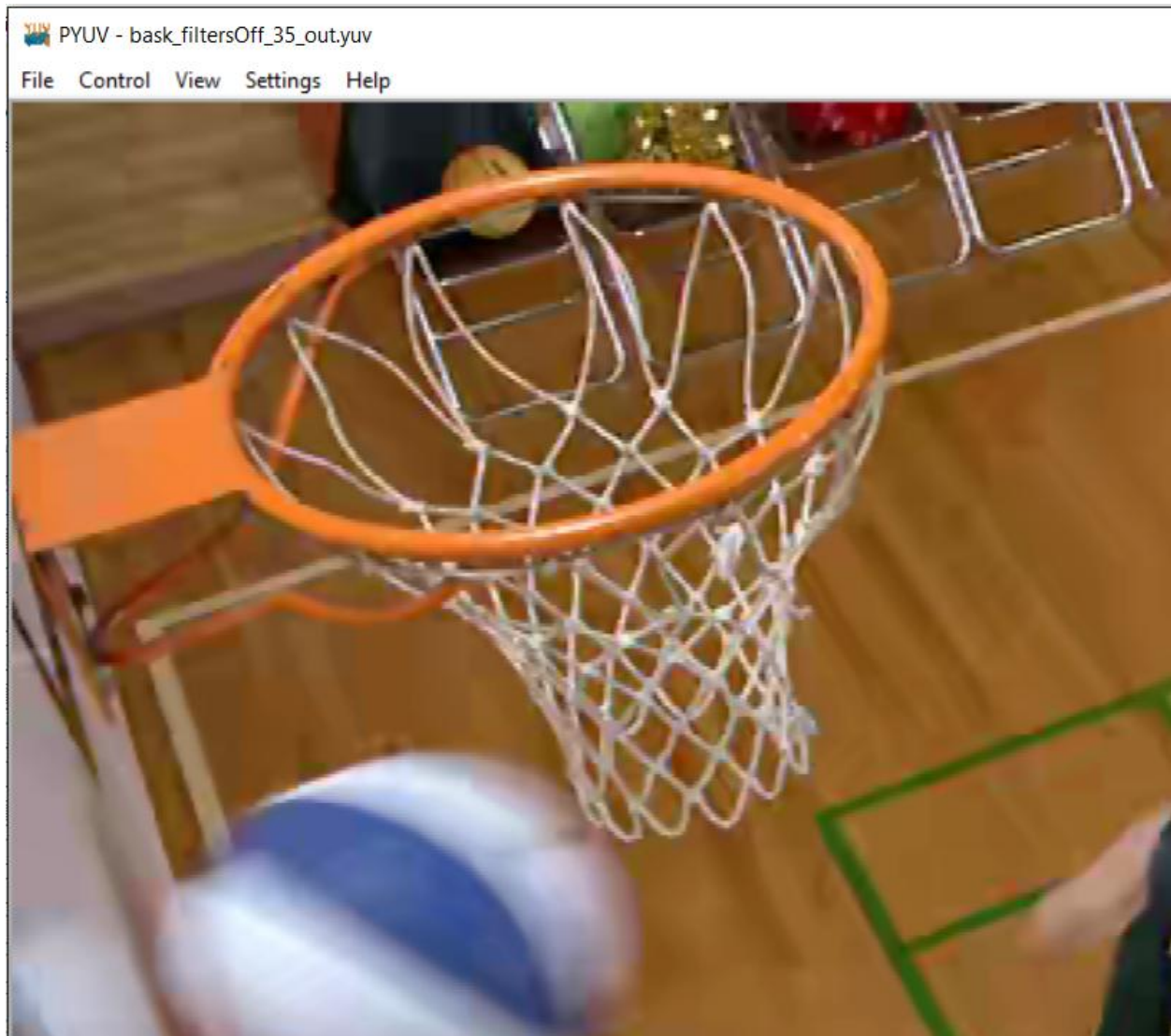


Figure 62. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.

 PYUV - bask_filtersOff_35_on_out.yuv

File Control View Settings Help



Figure 63. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter enabled.

Further visual results are shown in figures 65, 66, 67, 68 and 69 for QP = 45 (Only a part of the frame 1 is shown to better visualize compression artifacts)

 PYUV - BasketballDrillText_832x480_50.yuv

File Control View Settings Help



Figure 64. Original BasketballDrillText_832x480_50.yuv sequence frame1

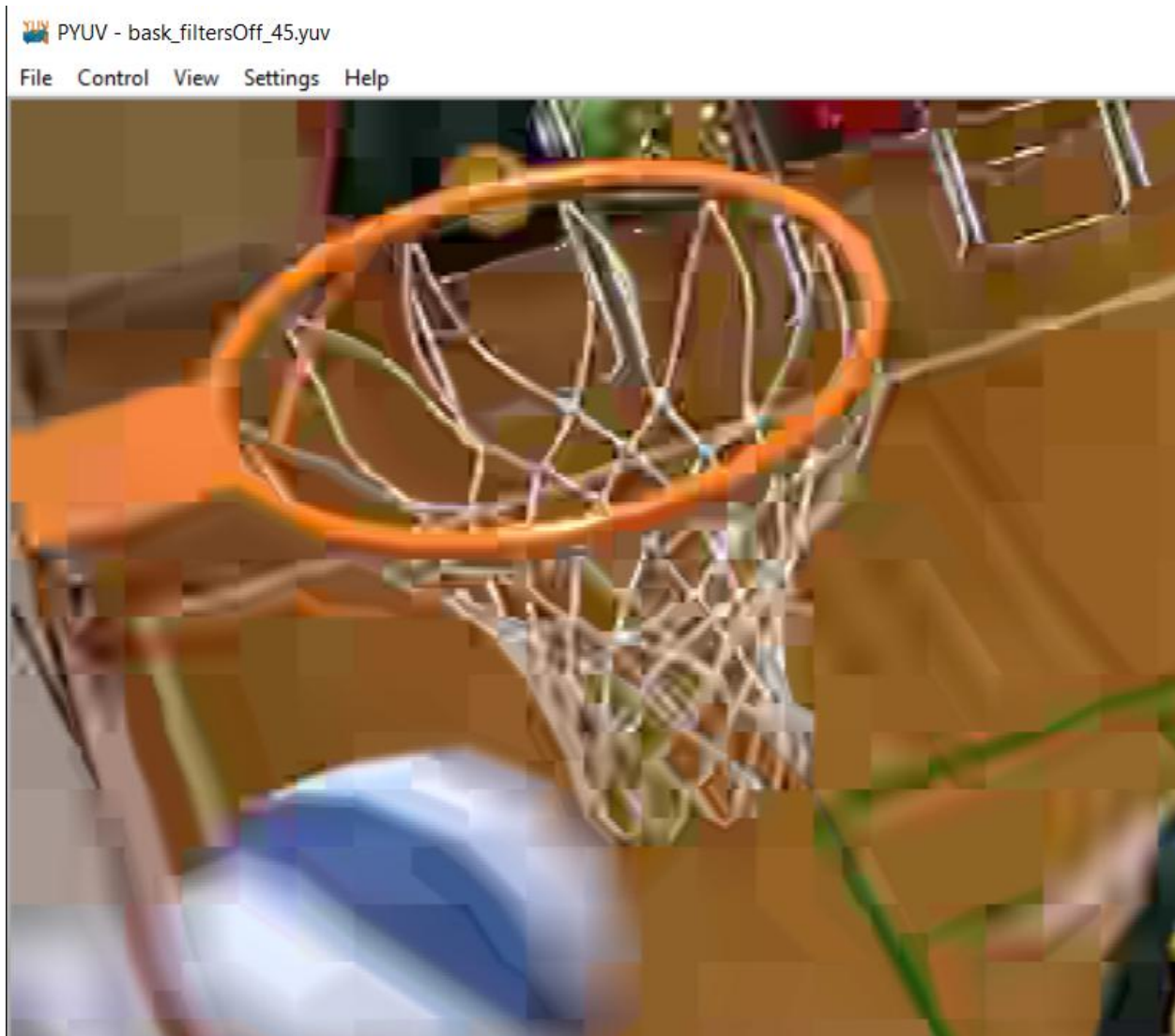


Figure 65. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter disabled.

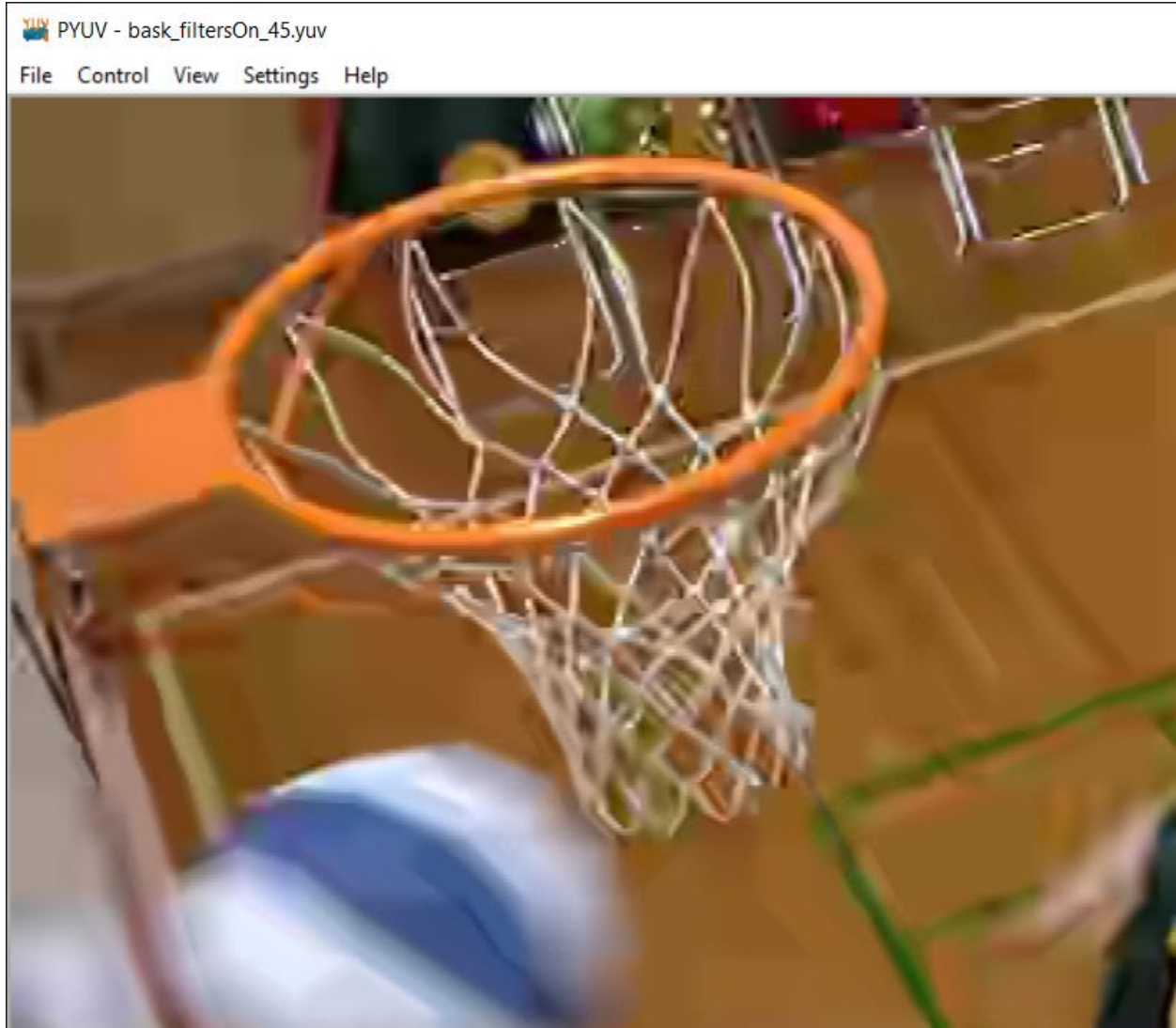


Figure 66. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.

 PYUV - bask_filtersOff_45_out.yuv

File Control View Settings Help



Figure 67. BasketballDrillText_832x480_50.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.

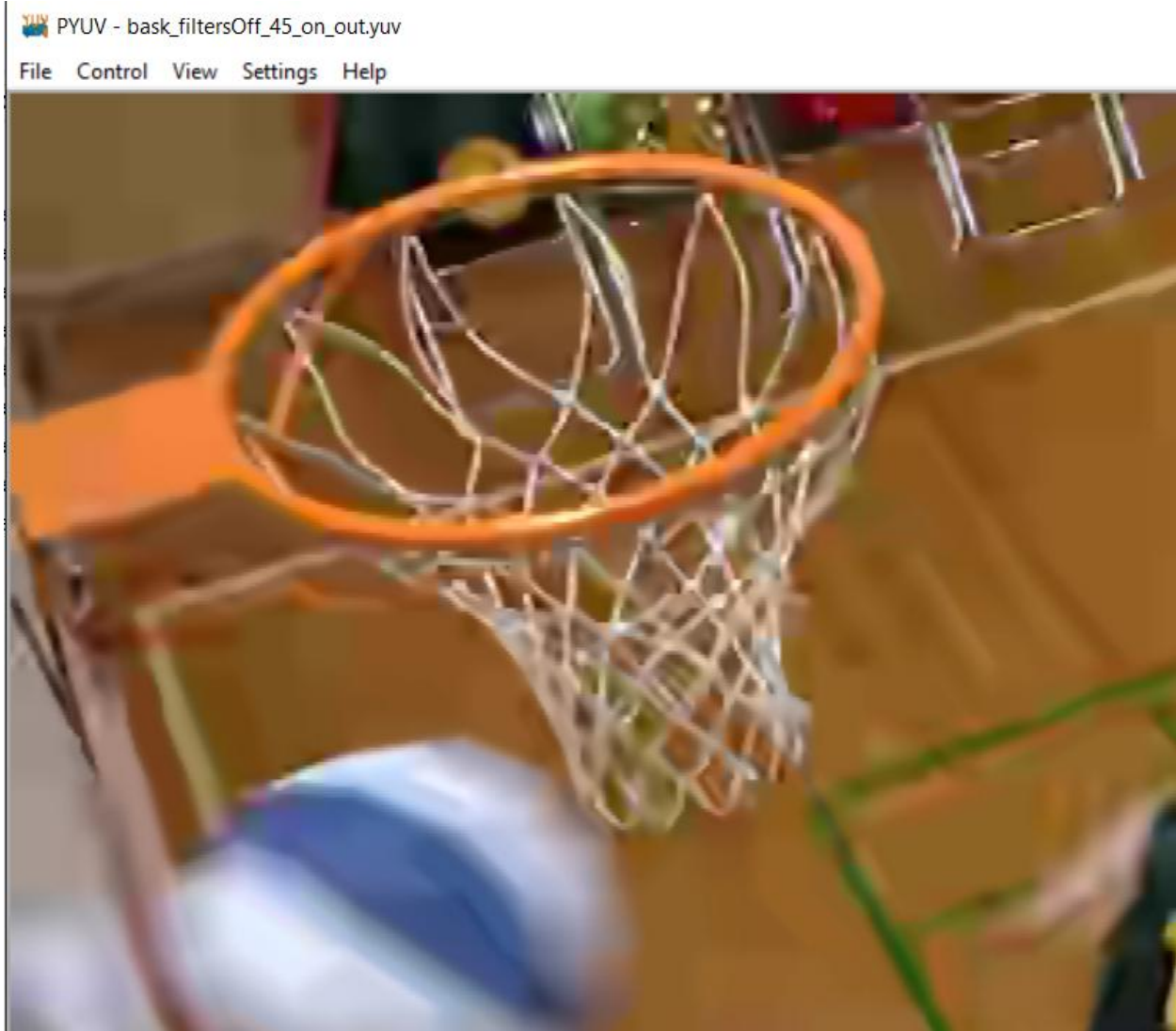


Figure 68. BasketballDrillText_832x480_50.yuv sequence frame1 with both in-loop filters and bilateral filter enabled.

Quantization Parameter (QP)	YUV-PSNR (dB), when deblocking and SAO are enabled in HEVC and NO post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are disabled in HEVC and NO post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are disabled in HEVC and post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are enabled in HEVC and post processing bilateral filter is used.
35	30.06	29.85	29.88	29.90
45	25.46	25.26	25.37	25.39

Table 3. RaceHorses_416x240_30.yuv sequence quality metrics

Quantization Parameter (QP)	Time in seconds for in-loop filters in HEVC	Time in seconds for bilateral post processing filter for HEVC	Time in seconds for both In-loop filter and bilateral post processing filter for HEVC
35	14	10	24
45	17	10	27

Table 4. RaceHorses_416x240_30.yuv filter time


Further visual results are shown in figures 70, 71, 72, 73 and 74 for QP = 35 (Only a part of the frame 1 is shown to better visualize compression artifacts)

 PYUV - RaceHorses_416x240_30.yuv

File Control View Settings Help



Figure 69. Original RaceHorses_416x240_30.yuv sequence frame1

 PYUV - RaceHorses_filtersOff_35.yuv

File Control View Settings Help



Figure 70. RaceHorses_416x240_30.yuvs equence frame1 with both in-loop filters and bilateral filter disabled.



Figure 71. RaceHorses_416x240_30.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.



Figure 72. RaceHorses_416x240_30.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.



Figure 73. RaceHorses_416x240_30.yuv sequence frame1 with both in-loop filters and bilateral filter enabled.

Similar visual results are obtained for QP = 45.

Quantization Parameter (QP)	YUV-PSNR (dB), when deblocking and SAO are enabled in HEVC and NO post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are disabled in HEVC and NO post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are disabled in HEVC and post processing bilateral filter is used.	YUV-PSNR (dB), when deblocking and SAO are enabled in HEVC and post processing bilateral filter is used.
35	37.86	37.63	37.72	37.77
45	32.19	31.96	32.08	32.11

Table 5. KristenAndSara_1280x720_60.yuv sequence quality metrics

Quantization	Time in seconds for in-	Time in seconds for	Time in seconds for
--------------	-------------------------	---------------------	---------------------

Parameter (QP)	loop filters in HEVC	bilateral post processing filter for HEVC	both In-loop filter and bilateral post processing filter for HEVC
35	213	164	377
45	249	163	412

Table 6. KristenAndSara_1280x720_60.yuv filter time

Further visual results are shown in figures 75, 76, 77, 78 and 79 for QP = 35 (Only a part of the frame 1 is shown to better visualize compression artifacts)



Figure 74. Original KristenAndSara_1280x720_60.yuv sequence frame1

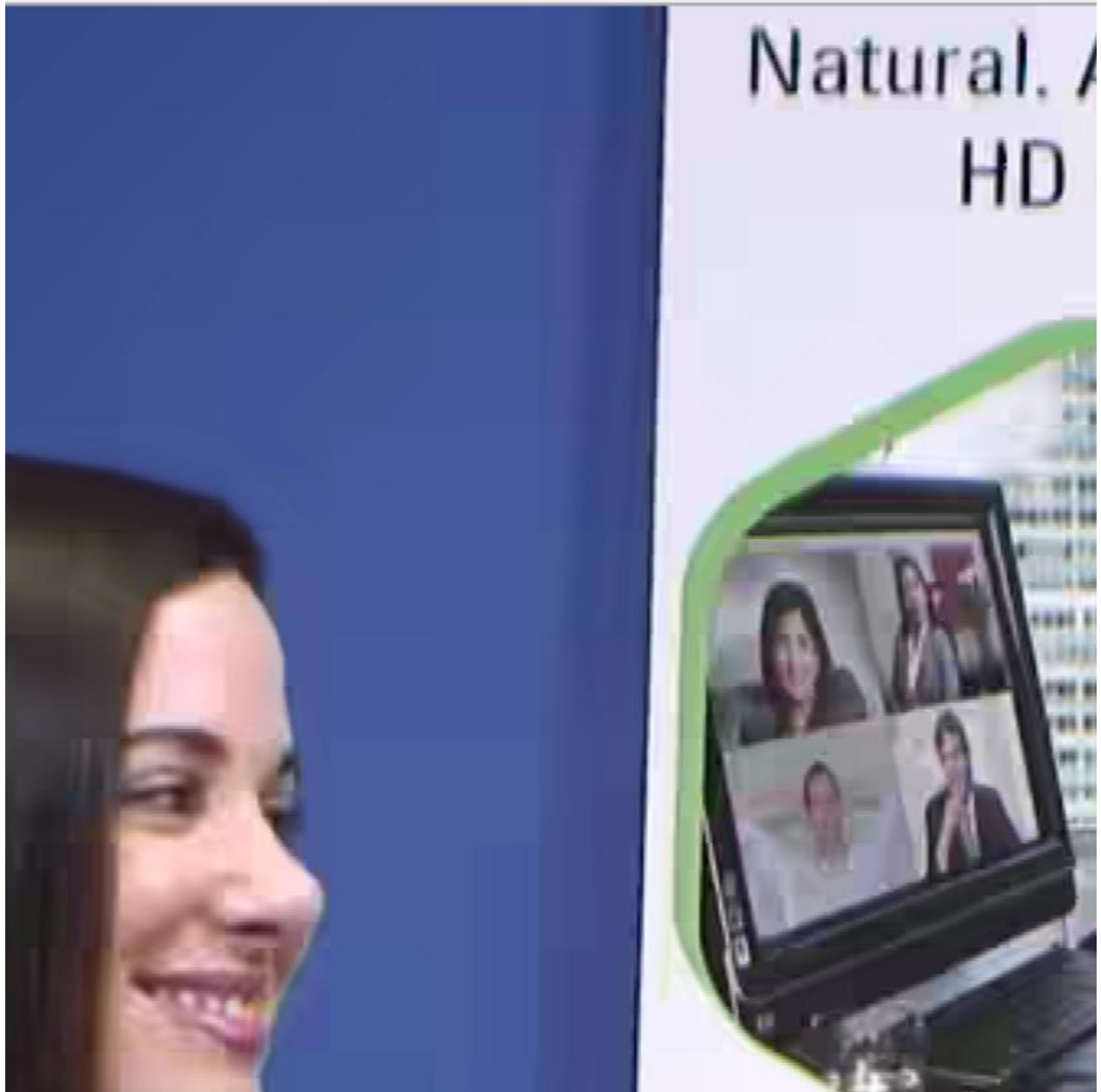


Figure 75. KristenAndSara_1280x720_60.yuv sequence frame1 with both in-loop filters and bilateral filter disabled.



Figure 76. KristenAndSara_1280x720_60.yuv sequence frame1 with in-loop filters enabled and bilateral filter disabled.

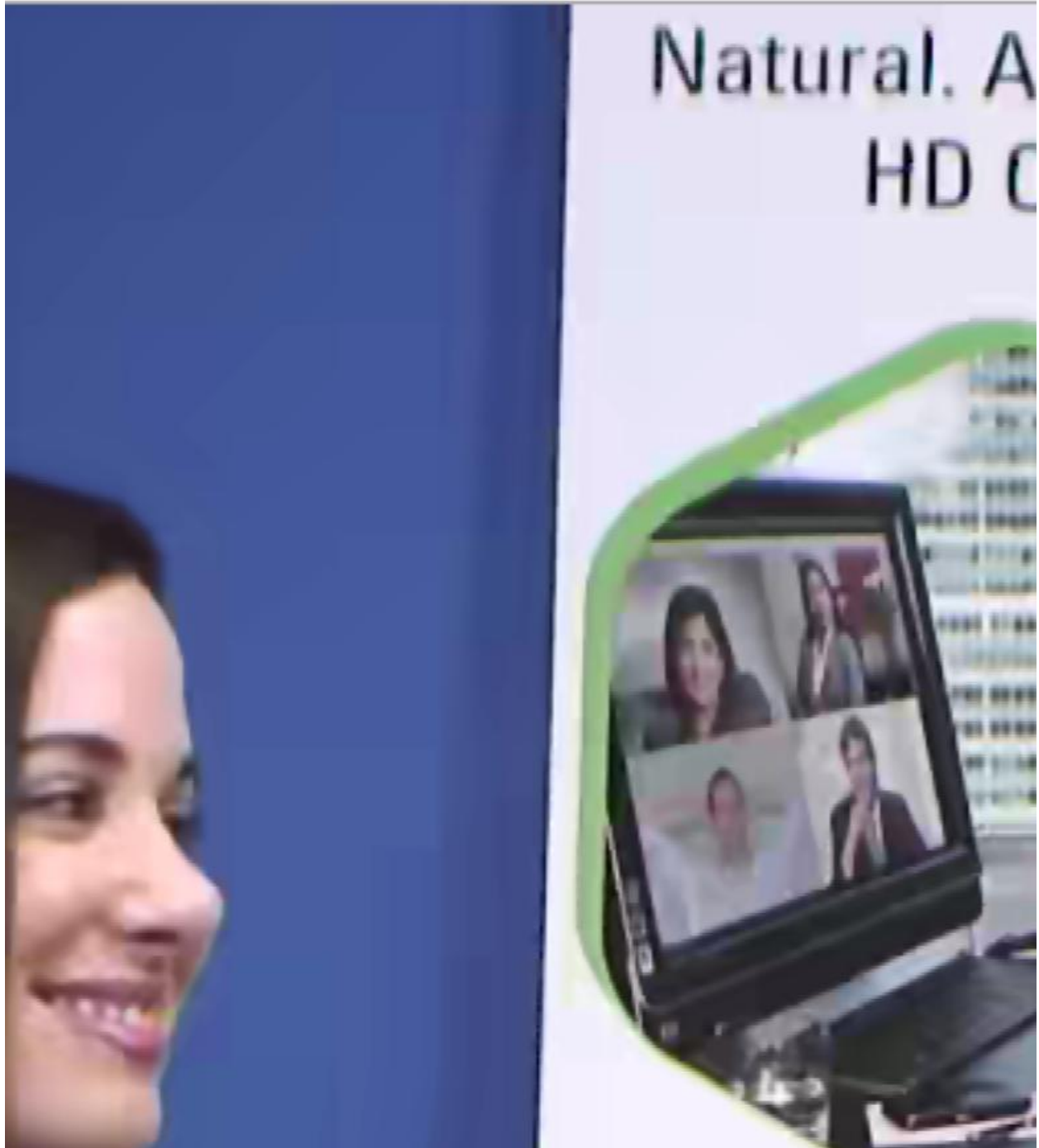


Figure 77. KristenAndSara_1280x720_60.yuv sequence frame1 with in-loop filters disabled and bilateral filter enabled.

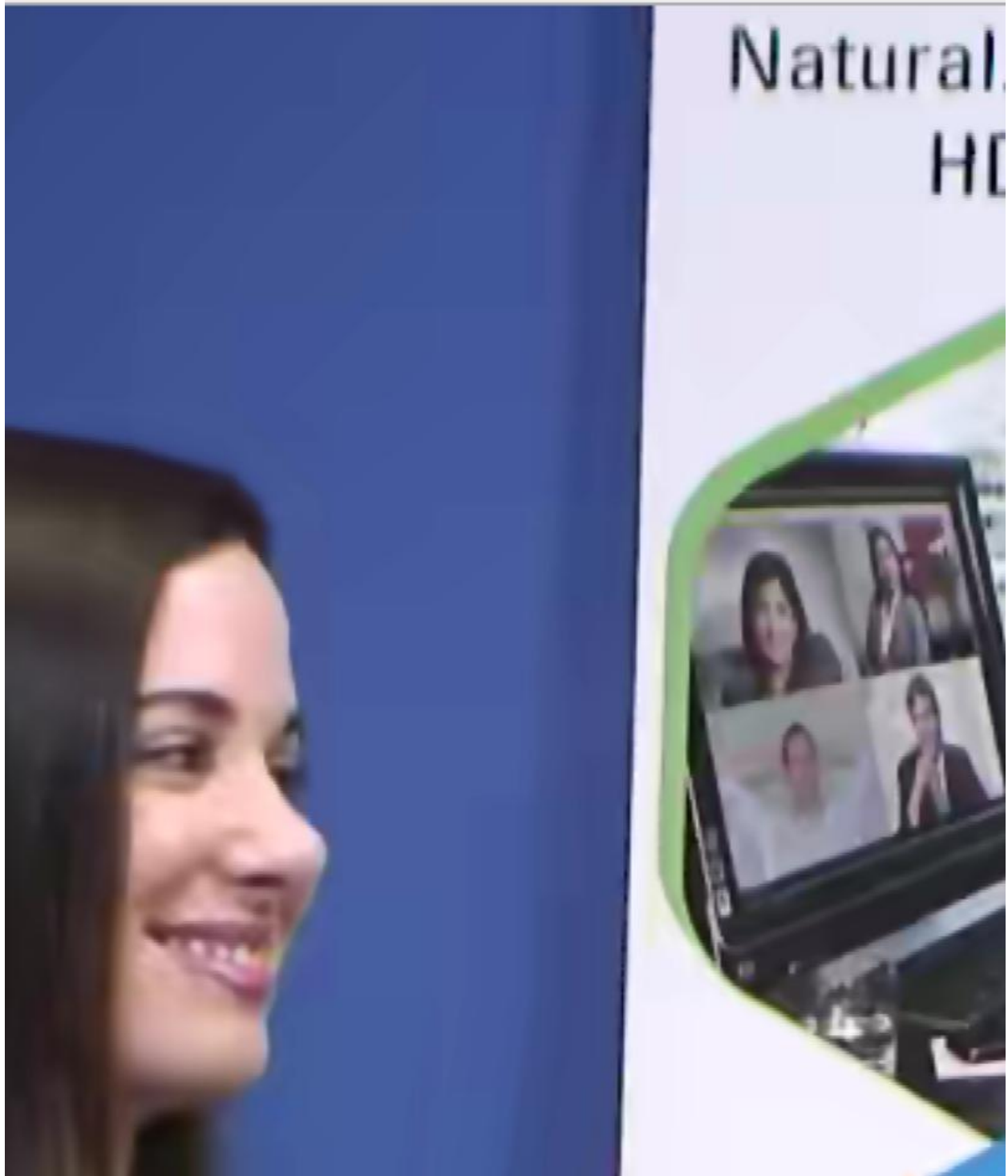


Figure 78. KristenAndSara_1280x720_60.yuv sequence frame1 with both in-loop filters and bilateral filter enabled.

Similar visual results can be obtained for QP=45

Chapter 7 – Conclusions and Future work

The objective of this thesis is to remove compression artifacts in HEVC decoded sequences which are caused mainly by block based transforms and coarse quantization. As we have seen in chapter 6, as the quantization parameter increases, the blocking artifacts, noise and ringing artifacts become more severe in decoded HEVC sequences.

HEVC has in-loop filters (deblocking and SAO filters) which mainly focus on removal of compression artifacts, which are due to coarse quantization and block based transforms. The adaptive bilateral filter which is applied on HEVC decoded sequences removes severe ringing in the sequences along with blocking artifacts and this is clearly shown in chapter 6.

The adaptive bilateral filter implemented for HEVC reduces compression artifacts with only a slight reduction in PSNR as shown in tables 6.1, 6.3 and 6.5 when compared to the conventional in-loop filters. We also notice in tables 6.2, 6.4 and 6.6 that the time taken by the bilateral filter is less than the time taken by conventional in-loop filters by about 30%. Further, when both in-loop filters and bilateral filter is enabled, compression artifacts are reduced which are clearly depicted in chapter 6.

Implementing bilateral filter in parallel using modern Graphics Processing Units (GPUs) can speed up the process further and there by reduces the time it takes to apply the filter and that can be taken a future work.

APPENDIX A: [Test Conditions](#)

HM 16.9 [23], the reference code for HEVC encoder and decoder has been used for this thesis.

All the simulations and work was done on the following System configuration

- Operating System : Windows 10 Home Edition (64bit)
- RAM: 16GB
- Processor: Intel® Core™ i7-4720HQ CPU @ 2.60GHz (x64-based processor)

References:

- [1] IIT Kharagpur Multimedia Processing Notes - http://nptel.ac.in/courses/Webcourse-contents/IIT%20Kharagpur/Multimedia%20Processing/New_index1.html
- [2] W.-Y. Wei, "An Introduction to Image Compression", National Taiwan University, Taipei, Taiwan, ROC
- [3] I.E.G. Richardson, "The H.264 advanced video compression standard", 2nd Edition, Hoboken, NJ, Wiley, 2010.
- [4] W.-Y. Wei, "Digital Video Compression Fundamentals and Standards", National Taiwan University, Taipei, Taiwan, ROC
- [5] N. Ling, "High efficiency video coding and its 3D extension: A research perspective," Keynote Speech, ICIEA, Singapore, July 2012.
- [6] Digital Image Processing Notes From NJIT - <https://web.njit.edu/~shi/courses/ECE789/ch16.pdf>
- [7] I.E.G. Richardson, "Video Codec Design: Developing Image and Video Compression Systems", Wiley, 2002.
- [8] G. J. Sullivan et al, "Overview of the High Efficiency Video Coding (HEVC) Standard", IEEE Trans. on Circuits and Systems for Video Technology, Vol. 22, No. 12, pp. 1649-1668, Dec. 2012.
- [9] V. Sze, M. Budagavi and G.J. Sullivan (Editors), "High Efficiency Video Coding (HEVC): Algorithms and Architectures", Springer, 2014.
- [10] V. Sze and M. Budagavi, "Design and Implementation of Next Generation Video Coding Systems (H.265/HEVC Tutorial)", IEEE International Symposium on Circuits and Systems (ISCAS), Melbourne, Australia, June 2014, available on <http://www.rle.mit.edu/eems/publications/tutorials/>
- [11] Detailed Overview of HEVC/H.265 by Shevach Riabtsev - <https://app.box.com/s/rxxxzr1a1lnh7709yvih>
- [12] M. J. Jakubowski and G. Pastuszak, "Block-based motion estimation algorithms – a survey," Opto-Electronic Review, Vol. 21, pp 86-102, March 2013.
- [13] L.N.A. Alves and A. Navarro, "Fast Motion Estimation Algorithm for HEVC", Proc. IEEE International Conf. on Consumer Electronics -ICCE Berlin, Germany, vol.11, pp. 11 - 14, Sep. 2012.
- [14] C. Fogg, "Suggested figures for the HEVC specification", ITU-T / ISO-IEC Document: JCTVC J0292r1, July 2012.
- [15] A. Norkin et al, "HEVC Deblocking Filter", IEEE Transactions on CSVT, vol. 22, no. 12, pp. 1746-1754, Dec. 2012.
- [16] 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.

- [17] W.-Y. Wei, "Deblocking Algorithms in Video and Image Compression Coding", National Taiwan University, Taipei, Taiwan, ROC
- [18] H.R. Wu and K.R. Rao, "Digital video image quality and perceptual coding", Boca Raton, FL: CRC Press, 2006.
- [19] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images", International Conference on Computer Vision (ICCV), pp. 839-846, Jan. 1998.
- [20] Introduction to bilateral filter - http://people.csail.mit.edu/sparis/bf_course/
- [21] E. Nadernejad, et al, "Adaptive Deblocking and Deringing of H.264/AVC Video Sequences", ICASSP, May, 2013.
- [22] M. Zhang and B. K. Gunturk, "Compression Artifact Reduction with Adaptive Bilateral Filtering", SPIE Proceedings, Vol. 7257, Jan. 2009
- [23] HM 16.9 (HEVC Software) Download Link - https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.9/
- [24] S. Jayaraman, S. Esakkirajan and T. Veerakumar, "Digital image processing", McGraw Hill Education (India) Private Ltd., New Delhi, 2009.
- [25] Website to access JCTVC Documents: <http://www.itu.int/en/ITU-T/studygroups/2013-2016/16/Pages/video/jctvc.aspx>
- [26] N. Ahmed, R. Natarajan and K.R. Rao, "Discrete Cosine Transform", IEEE Trans. on Computers, Vol. C-23, pp.90-93, Jan. 1974.
- [27] G.J. Sullivan et al, "Standardized Extensions of HEVC", IEEE Journal of Selected topics in Signal Processing, Vol.7, no.6, pp.1001-1016, Dec. 2013.
- [28] HEVC tutorial by I.E.G. Richardson: <http://www.vcodex.com/h265.html>
- [29] Video Sequence Download Link- <http://media.xiph.org/video/derf/>
- [30] 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.
- [31] Multimedia Processing course website - http://www.uta.edu/faculty/kr Rao/dip/Courses/EE5359/index_tem.html
- [32] Joint Collaborative Team On Video Coding Information website- <http://www.itu.int/en/ITU-T/studygroups/2013-2016/16/Pages/video/jctvc.aspx>
- [33] H.263: Video Coding for Low Bit Rate Communication, "<http://www.itu.int/rec/T-REC-H.263/en>"
- [34] H.261: Video Codec for Audiovisual Services at px64 kbit/s," <http://www.itu.int/rec/T-REC-H.261-199303-1/en>"
- [35] M. Wien, "High efficiency video coding: Tools and specification", Springer, 2015.

[36] M. J. Jakubowski and G. Pastuszak, "Block-based motion estimation algorithms – a survey," *Opto-Electronic Review*, Vol. 21, pp 86-102, March 2013.

[37] D. Grois et al, "Performance Comparison of H.265/ MPEG-HEVC, VP9, and H.264/ MPEG-AVC Encoders", available on:

http://iphome.hhi.de/marpe/download/Performance_HEVC_VP9_X264_PCS_2013_preprint.pdf

[38] ISO/IEC 15444-1:2000(E), "Information technology-JPEG 2000 image coding system-Part 1: Core coding system", 2000.

[39] D. Grois, B. Bross and D. Marpe, "HEVC/H.265 Video Coding Standard (Version 2) including the Range Extensions, Scalable Extensions, and Multiview Extensions," (Tutorial) Sunday 27 Sept 2015, 9:00 am to 12:30 pm), IEEE ICIP, Quebec City, Canada, 27 – 30 Sept. 2015.

http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=7243

[40] Access the MPL website: <http://www.uta.edu/faculty/krrao/dip/Courses/EE5359/>

[41] Discrete Transforms and their applications:

<http://www.uta.edu/faculty/krrao/dip/Courses/EE5355/ee5355.htm>

[42] ITU-T Rec. H.262 and ISO/IEC 13818-2 (2000) Generic coding of moving pictures and associated audio information: Video. (MPEG-2 Video), 2nd edition.

[43] Access to HM 16.7 Software Manual:

http://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.0/doc/software-manual.pdf

Biographical Information

Rohith Reddy Etikala was born in Hyderabad, Andhra Pradesh, India in 1988. After completing his bachelors in Electronics and Communication Engineering (ECE) from Jawaharlal Nehru Technological University (JNTU) in 2010, he has joined Computer Sciences Corporation (CSC) as a Software Engineer in September 2010 and worked for 3.3 years.

He joined University of Texas at Arlington to pursue his M.S in Electrical Engineering in spring 2014. This was around the time he joined the Multimedia processing Lab to do research with Dr. K. R. Rao. He worked as Video Engineering Intern at InterDigital, San Diego, CA in fall 2015.