IMPLEMENTATION AND PERFORMANCE ANALYSIS OF 2-D ORDER 16 INTEGER TRANSFORMS IN H.264/AVC AND AVS-VIDEO FOR HIGH DEFENITION VIDEO CODING

by

MADHU PERINGASSERY KRISHNAN

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 2010

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. K. R. Rao, for his guidance, support and encouragement at every stage of development in this research work.

I would like to thank Dr. Manry and Dr. Davis for serving on my committee. I would like to express my gratitude to Dr. P. Topiwala for his inputs towards the completion of this research.

Last but not least, I would like to thank my family and friends who have helped me throughout my life.

November 24, 2010

ABSTRACT

IMPLEMENTATION AND PERFORMANCE ANALYSIS OF 2-D ORDER 16 INTEGER TRANSFORMS IN H.264/AVC AND AVS-VIDEO FOR HIGH DEFENITION VIDEO CODING

Madhu Peringassery Krishnan, M.S

The University of Texas at Arlington, 2010

Supervising Professor: K. R. Rao

H.264/AVC and AVS-video are two video coding standards that have a wide range of applications ranging from high-end professional camera and editing systems to low-end mobile applications. They strive to achieve maximum compression efficiency without compromising the quality of video. To this end many coding tools are defined in them. Transform coding is one among them.

Transform coding represents the signal/image (that is currently in time/spatial domain) in another domain (transform domain), where most of the energy in signal/image is concentrated in a fewer number of coefficients. Thus the insignificant coefficients can be discarded after transform coding to achieve compression. In images/videos the DCT-II (which represents a signal/image as the weighted sum of cosine functions with different frequencies) is primarily used for transform coding.

H.264/AVC and AVS-video utilize integer approximations of the DCT-II (known as integer cosine transform) to reduce computational complexity by performing only fixed-point arithmetic operations and eliminates the mismatches between the forward and inverse transforms. The order (size) of the integer cosine transforms used is small (4 x 4 and 8 x 8). They achieve the best coding efficiency for standard definition and low-resolution videos. But, better coding efficiency can be achieved for high definition videos by using higher order (16 x 16 and 32 x 32) integer cosine transforms. As high definition videos are becoming more and more popular, it is imperative that sooner or later they will be integrated into the standards. For this purpose many higher order (16 x 16 and 32 x 32) integer cosine transforms have been proposed. But, a comparative study on the performance of these higher order integer cosine transforms in H.264/AVC and AVS-video has not been done yet.

The purpose of the research is to analyze some higher order 16 x 16 integer cosine transforms, implement them in H.264/AVC and AVS-video and carry out a comparative study of their performances.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

NOMENCLATURE

PSNR	Peak signal to noise ratio
dB	Decibel
MPEG	Moving picture experts group
JPEG	Joint photographic experts group
DCT	Discrete cosine transform
ICT	Integer cosine transform
KLT	Karhunen-Loeve transform
AVS	Audio video standard
AVC	Advanced video coding
VCEG	Video coding experts group
VBT	Variable block size transform
HP	High profile
EP	Enhanced profile
MCP	Motion compensated prediction
MB	Macroblock
PIT	Prescaled integer transform
QP	Quantization parameter
NNICT	Non-normalized integer cosine transform
VLC	Variable length coding
EAC	Enhanced audio codec

CHAPTER 1

INTRODUCTION

1.1 Discrete cosine transform and video compression

Type II discrete cosine transform (DCT-II) [1] developed by Ahmed, Natarajan, and Rao is the basic building block of a plethora of image and video compression standards. Due to its energy compaction nature DCT-II is regarded as the best sub-optimal orthogonal transform that closely approximates the optimal Karhunen-Loeve transform (KLT) [1].The KLT concentrates its energy in a few number of coefficients by spatial decorrelation. DCT-II is also a unitary transform composed of orthonormal basis vectors.

DCT-II of size 8 is used in many video standards like H.261, JPEG, MPEG-1, MPEG-2, H.263, and MPEG-4 [2 - 4]. A typical two-dimensional DCT-II of size 8 is separable into two one dimensional transforms performed first along one axis of the image or video frame, and then along the other axis of the resultant from the previous procedure, as shown in Figure 1.1.

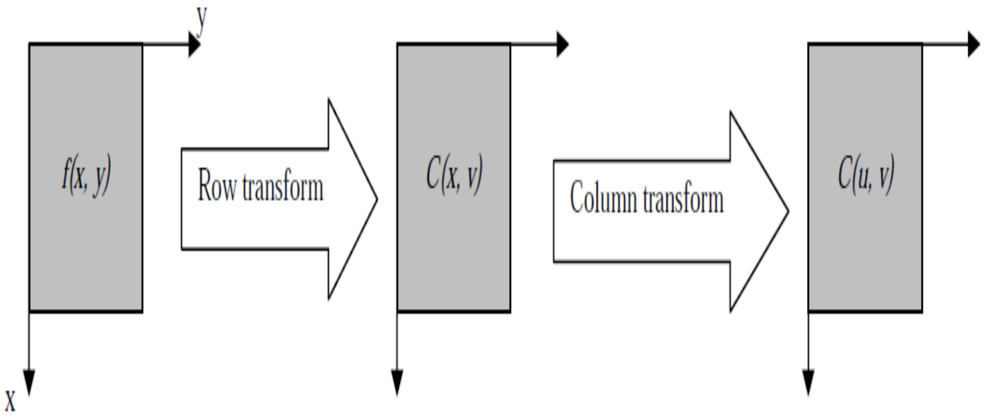


Figure 1.1 Separable property of DCT-II

The resultant cosine values are difficult to approximate in fixed precision integers, thus producing rounding errors in practical applications. Rounding errors can introduce enough error into the computations and alter the orthogonality property of the transform.

The fixed point implementation problem can be solved by either (1) approximating the cosines to integer values or (2) by preserving the relative magnitudes, relationships, and symmetries of the cosines in the cosine matrix to create a reversible integer cosine matrix [5 - 8]. The first approach led to mismatched implementations of the transform in encoders and decoders and thus introducing errant results. The second approach was to specify an integer based transform that could be exactly specified and easily implemented in both the encoder and decoder. Moreover, they were designed to ensure that the forward and inverse transforms are fully reversible, and hence completely orthogonal.

1.2 Integer cosine transforms

Integer cosine transforms (ICT) can be generated from DCT-II by replacing the real numbered elements of the DCT-II matrix with integers keeping the relative magnitudes and orthogonal relationship among the matrix elements [6]. The integer transform coefficients result in a computationally less intense procedure that implements similar energy concentration like DCT-II. It can be implemented using integer arithmetic without mismatch between encoder and decoder. Moreover, fast algorithms have been developed for implementing them. The typical block diagrams of an encoder and decoder is illustrated in Figures 1.2 and 1.3.

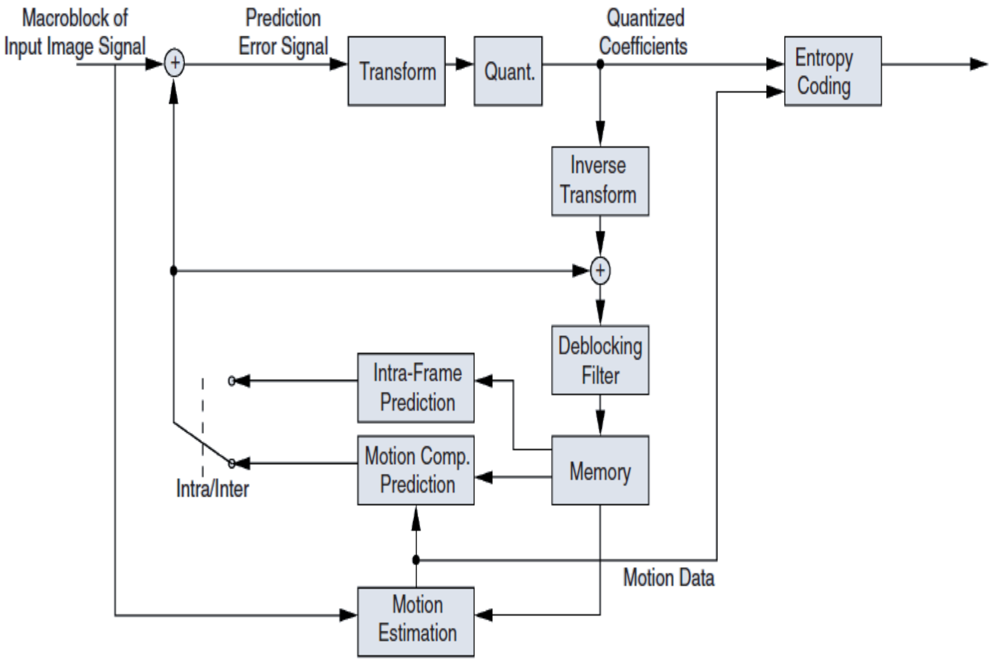


Figure 1.2 Typical block diagram of a video encoder [21].

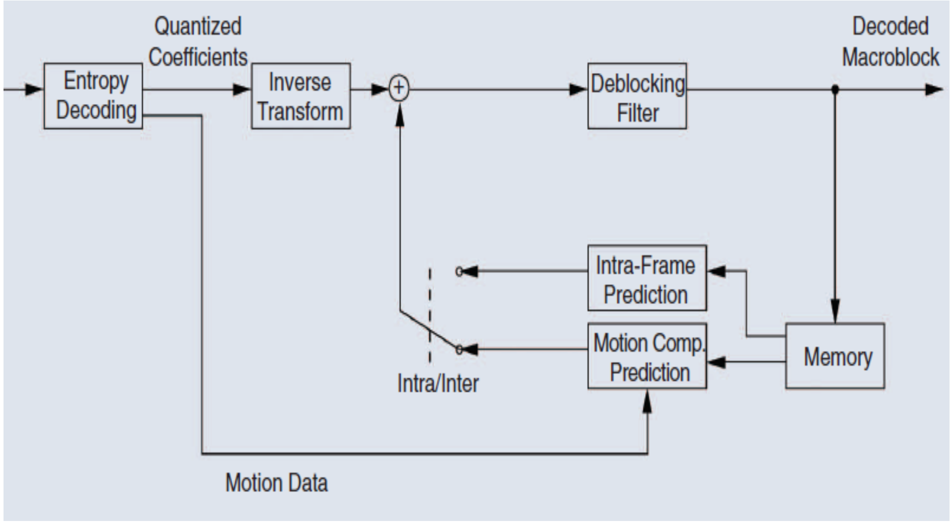


Figure 1.3 Typical block diagram of a video decoder [21].

The orthogonality of ICT depends on the elements of the transform matrix for orders greater than four. Due to this constraint, the magnitudes of elements tend to be quite large for large ICTs (order-16 ICT) [8]. This led to the development of ICTs that are mutually orthogonal by using the principle of dyadic symmetry [5]. Thus the elements in transform matrices can be selected without orthogonality constraint. Usually, the magnitudes of elements are designed to be very small (represented by 4~5 bits) for simple implementation.

In recent years, ICT has been widely used in video coding, such as the order-4 and order-8 ICTs in H.264 [9] and the order-8 ICT in the Audio video coding standard of China (AVS) [10] and WMV9 [22].

1.3 HD video coding and integer cosine transforms

High-definition (HD) videos such as in high-definition television (HDTV) broadcasting, high density storage media, video-on-demand (VOD), and surveillance are of higher resolution than standard-definition (SD) video, and commonly involves resolutions like 720p (1280 x 720) and 1080p (1920 x 1080). MPEG-2 [11] was the predominantly used standard for HD video coding but the significant performance improvements achieved by H.264/AVC [12] has led to its wide spread acceptance. Another efficient standard is the audio and video-coding standard (AVS) [13] developed by China and WMV9 [22] developed by Microsoft.

The H.264/AVC basically employs both intra and inter-prediction tools to achieve accurate prediction and advanced entropy coding schemes to remove statistical redundancy. The standard defines certain sets of capabilities, known as profiles, targeting specific applications. It is based on the traditional hybrid concept of block-based motion-compensated prediction (MCP) and transform coding. The high Profile (HP) [14] designed with HD video coding in mind, utilizes variable block size transforms (VBT) to achieve coding efficiency. The concept of VBT coding involves the adaptation of the transform block size to the block size used for motion compensation as large transforms can provide better energy compaction than a smaller transform. But they introduce ringing artifacts caused by quantization. By choosing the transform size according to the signal properties, the tradeoff between energy compaction and detail can be optimized. A 2-D order-8 ICT is used adaptively as an alternative to the 2-D order-4 ICT in H.264/AVC. These transforms are illustrated in Figure 1.4.

$$H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

(a)

$$T_{8x8} = \begin{bmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 12 & 10 & 6 & 3 & -3 & -6 & -10 & -12 \\ 8 & 4 & -4 & -8 & -8 & -4 & 4 & 8 \\ 10 & -3 & -12 & -6 & 6 & 12 & 3 & -10 \\ 8 & -8 & -8 & 8 & 8 & -8 & -8 & 8 \\ 6 & -12 & 3 & 10 & -10 & -3 & 12 & -6 \\ 4 & -8 & 8 & -4 & -4 & 8 & -8 & 4 \\ 3 & -6 & 10 & -12 & 12 & -10 & 6 & -3 \end{bmatrix}$$

(b)

Figure 1.4 ICTs in H.264/AVC (a) 4 × 4 transform matrix, (b) 8 × 8 transform matrix [21].

AVS developed by China is compatible with MPEG-2 at the system level, and it involves Chinese proprietary IP in the compression algorithms [13]. The coding efficiency of the standard is better than MPEG-2 and is similar to H.264/AVC. Moreover, its implementation in chips is far easier than H.264/AVC. The enhanced profile (EP) [15] of AVS aims at HD video coding although it does not support rich functionality like the H.264/AVC. So, it is quite similar to the main profile of H.264. A 2-D order-8 ICT was initially incorporated in AVS for coding [19], [20]. Figure 1.5 shows the ICT matrix.

$$ICT_{8\times 8} = \begin{bmatrix} 8 & 10 & 10 & 9 & 8 & 6 & 4 & 2 \\ 8 & 9 & 4 & -2 & -8 & -10 & -10 & -6 \\ 8 & 6 & -4 & -10 & -8 & 2 & 10 & 9 \\ 8 & 2 & -10 & -6 & 8 & 9 & -4 & -10 \\ 8 & -2 & -10 & 6 & 8 & -9 & -4 & 10 \\ 8 & -6 & -4 & 10 & -8 & -2 & 10 & -9 \\ 8 & -9 & 4 & 2 & -8 & 10 & -10 & 6 \\ 8 & -10 & 10 & -9 & 8 & -6 & 4 & -2 \end{bmatrix}$$

Figure 1.5 8 × 8 ICT matrix in AVS [19][20].

The Table 1.1 describes the profiles in H.264/AVC and Table 1.2 describes the different profiles in AVS-video.

Table 1.1 Basic profiles in H.264/AVC [14].

Profiles	Applications
Baseline Profile	Mobile applications
Main Profile	SD television broadcasting
Extended Profile	Streaming video
High Profile	HD television broadcasting

Table 1.2 Profiles in AVS-video [19].

Profiles	Applications
Jizhun Profile	HD television broadcasting
Jiben Profile	Mobile applications
Shenzhan Profile	Video surveillance
Jiaqiang Profile	Multimedia entertainment

1.4 Outline

HD video sequences when compared with lower resolution videos have higher prediction error correlation [16]. To exploit this spatial property 2-D order 16 integer transforms [8] [17] can be used. The purpose of the thesis is to implement and analyze various 2-D order-16 integer transforms on both AVS and H.264/AVC [18].

In order to implement and analyze the 2-D order 16 integer transforms it is imperative to have a good grasp of H.264/AVC and AVS China. So the second chapter will deal with the nuances of H.264/AVC and its features. The third chapter considers the AVS China and its functionality. The fourth chapter comprises of the analysis and implementation details of 2-D order 16 integer transforms in the standards. Every detail regarding the implementation will be discussed in this chapter. In the fifth chapter the results will be explained and a conclusion will be deduced from them. Possible avenues for expansion in future are also discussed in this chapter.

CHAPTER 2

H.264/AVC

2.1 Introduction

A good video compression algorithm should reduce the following four types of redundancies [26]:

- Spatial
- Temporal
- Perceptual
- Statistical

The latest video coding standard known as H.264/AVC [28] takes care of all the redundancies mentioned above using its elaborate set of coding tools. The standard was finalized in March 2003 as a result of collaborative work done by ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). Currently it is the most accepted video coding standard as it achieves significant improvement in compression performance compared to prior standards like MPEG-2 [23]. Applications of H.264/AVC include video conferencing, entertainment (broadcasting over cable, satellite, DVDs, video on demand etc.), streaming video, surveillance and tracking applications, and digital cinema.

In order to cater for different applications three basic feature sets called profiles [9] for use in generating conforming bit streams were established:

- Baseline
- Main
- Extended

The baseline profile is intended for low cost mobile applications and video conferencing that are susceptible to data loss. The main profile is designed with an emphasis on compression coding efficiency capability and thus is used for standard-definition digital TV broadcasts. The extended profile tailored for video streaming applications combines the robustness of the Baseline profile with a higher degree of coding efficiency and network robustness.

In September 2004 the final draft amendment work on Fidelity Range Extensions (FRExt) of H.264/MPEG4-AVC was completed [24]. They added four new profiles [9]:

- The high profile (HP) for HD broadcast and storage applications.
- The high 10 (Hi10P) profile that builds on HP to give better decoded picture precision.
- The high 4:2:2 (H422P) profile for interlaced video.
- The high 4:4:4 (H444P) profile supporting up to 4:4:4 chroma sub-sampling.

These profiles added new features like adaptive block-size transform; encoder specified optimized quantization scaling matrices and region specific lossless representation of video.

Four intra profiles were also provided with the above-mentioned profiles for studio editing and professional applications.

H.264 /AVC also specify 16 levels that define upper bounds for the bit stream or lower bounds for the decoder capabilities [21]. Table 2.1 shows the different levels in H.264/AVC. The level ‘1b’ was added by the FRExts amendment. The levels specify a set of constraints indicating a degree of required decoder performance for a profile. A decoder that conforms to a given level can decode all bit streams that are encoded for that level and for all the lower levels. A profile and level together can be used to define points of interoperability for applications with similar functional requirements.

Table 2.1 Levels in H.264/AVC [25].

	Typical picture size	Typical frame rate	Maximum compression bit rate	Maximum number of reference frames
1	QCIF	15	64 kbps	4
1b	QCIF	15	128 kbps	4
1.1	CIF or QCIF	7.5(CIF) / 30 (QCIF)	192 kbps	2 (CIF) / 9 (QCIF)
1.2	CIF	15	384 kbps	6
1.3	CIF	30	768 kbps	6
2	CIF	30	2 Mbps	6
2.1	HHR	30/25	4 Mbps	6
2.2	SD	15	4 Mbps	5
3	SD	30/25	10 Mbps	5
3.1	1280 ×720p	30	14 Mbps	5
3.2	1280 ×720p	60	20 Mbps	4
4	HD formats	60p/30i	20 Mbps	4
4.1	HD formats	60p/30i	50 Mbps	4
4.2	1920 × 1080p	60p	50 Mbps	4
5	2k ×1k	72	135 Mbps	5
5.1	2k × 1k or 4k × 2k	120/30	240 Mbps	5

The Figure 2.1 gives an overall picture of the profiles.

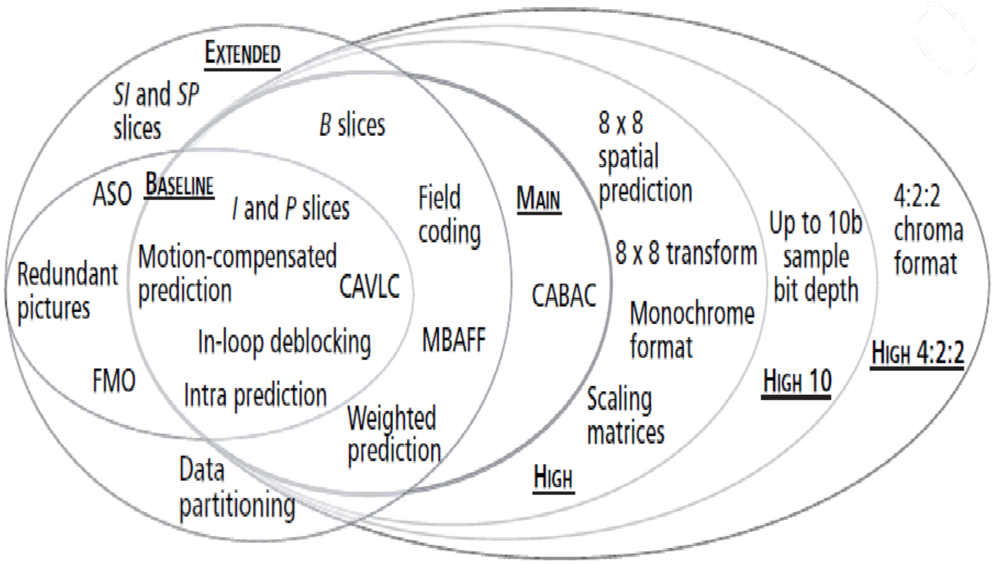


Figure 2.1 Profiles in H.264/AVC [25].

2.2 H.264/AVC encoder

The H.264/AVC encoder can be understood from its typical structural diagram as illustrated in Figure 2.3. The encoding operation starts by splitting the first frame of a sequence or random access point into macroblocks (MB). This frame is usually intracoded with no use of reference frames. The samples in a block are predicted from previously encoded neighboring blocks. The encoding process selects the best neighboring block and determines how the samples from these blocks are combined for inter prediction. The decoder is notified of this selection process.

Frames between the random access points are inter coded to reduce temporal redundancy. Inter coding employs the motion compensation technique (Figure 2.2) of choosing motion data that identifies the reference frame and spatial displacement vectors that are applied to predict the samples of each block [25]. The difference between the original and the predicted samples known as the residual is then transform coded and the resultant is scaled and quantized. The quantized transform coefficients are then entropy coded and transmitted with the coded prediction information so that the decoder can differentiate between the inter and intra coded samples. For the prediction of subsequent blocks in the current picture or subsequent coded pictures a decoder model is present in the encoder itself.

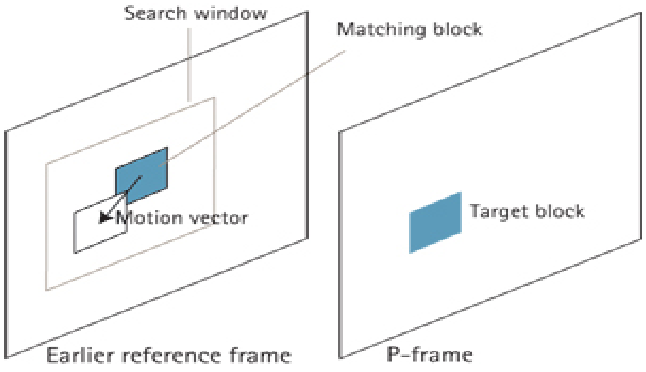


Figure 2.2 Single frame motion compensated prediction

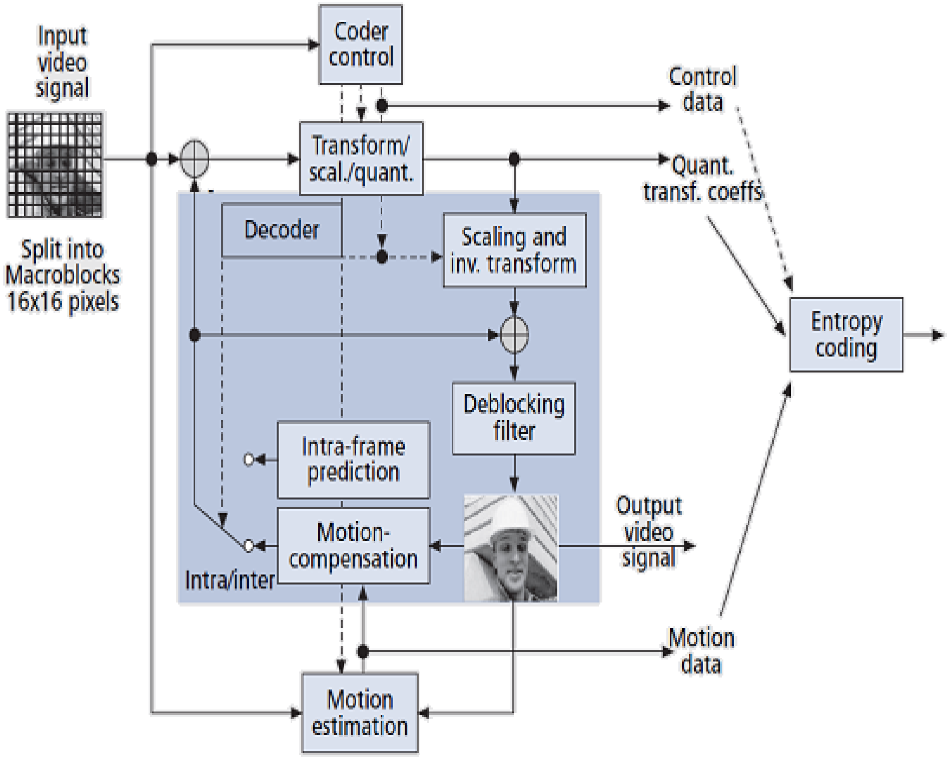


Figure 2.3 H.264/AVC encoder block representation [25].

The decoder (Figure 2.15) performs an inverse of the operations done by the encoder. It inverts the entropy coding process and then using the motion data and the type of prediction performs the prediction process. Inverse scaling and transforming of the residual is also done and the deblocking filter is applied to the result to get the video output.

2.2.1 Intra prediction

A frame or picture is considered to be composed of fixed size MBs that each cover 16 × 16 square samples of the luma component. The MBs in a frame are combined into independent slices (Figure 2.4) each based on type of prediction used. The different slices are:

- I slice (which is intra coded)
- P and B slices (where P stands for predictive and B stands for bi-predictive)
- SP and SI slices (Switching P and Switching I) which are specified for efficient switching between bit streams coded at different bit rates [28].

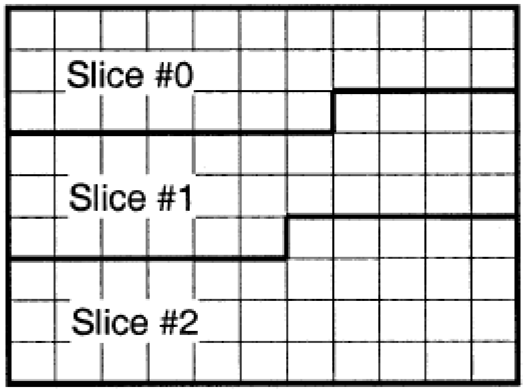


Figure 2.4 Grouping of MBs into slices [29].

Intra prediction allows for exploiting spatial redundancy in a video frame .The samples in an intra predicted MB are predicted by using only information of already coded MBs in the same frame as the neighboring samples are similar. Usually an MB is coded as an intra-MB when temporal prediction is impossible. In H.264/AVC the intra coding types are based on prediction of samples in a given block applied in the spatial domain and not the transform domain as has been done in previous video coding standard [29].

An MB can be coded as one 16 × 16, four 8 × 8, or sixteen 4 × 4 blocks [27]. For the prediction purpose of luma component (Y), nine different prediction modes are supported for 4 × 4 and 8 × 8 block sizes and four prediction modes for the 16 × 16 blocks. In DC mode or mode 2 the current block's samples are predicted using a weighted average (usually mean) of the neighboring samples. The other prediction modes are shown in Figure 2.5.

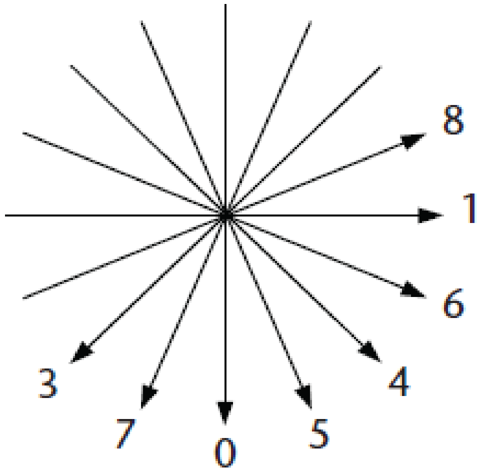


Figure 2.5 Nine prediction directions for the 4 × 4 and 8 × 8 blocks [21].

The 16×16 blocks have four prediction modes:

- Vertical prediction
- Horizontal prediction
- DC-prediction
- Plane-prediction

Plane-prediction involves a linear function between the neighboring samples to the left and to the top in order to predict the current sample. The intra prediction for the chroma samples (Cb, Cr) of a MB is similar to the 16×16 type for the Y samples except that they are performed on 8 × 8 blocks [27].

Figure 2.6 illustrates three out of the nine possible intra prediction modes for the 4 × 4 blocks.

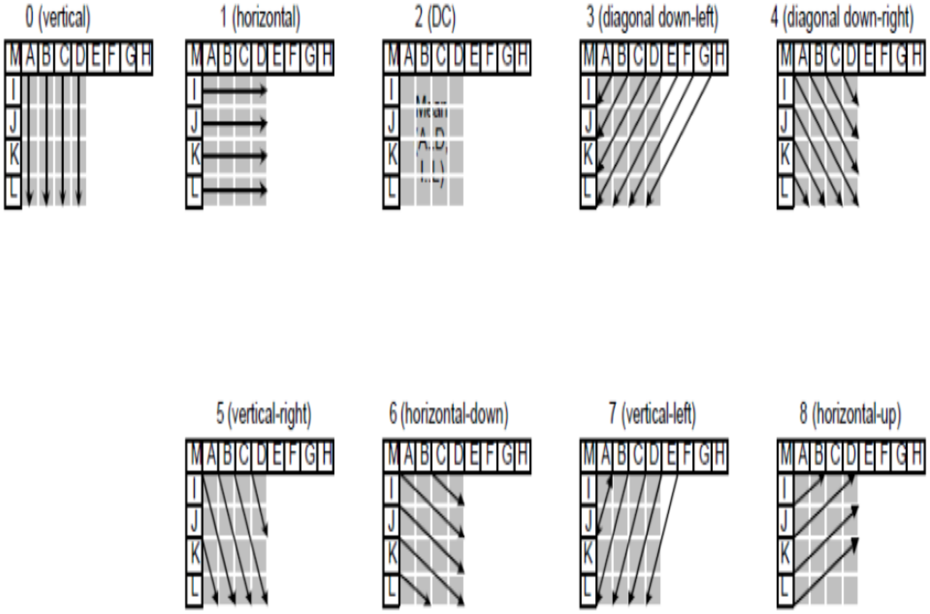


Figure 2.6 Nine intra prediction modes for 4 × 4 and 8 × 8 blocks [31].

2.2.2 Inter prediction

Inter prediction or more commonly known as motion compensated prediction could be done for both P-slices (predictively-coded) and B-slices (bi-predictively coded). A variety of motion-compensated coding types are allowed in P slice-MB. The partitioning of MBs is illustrated in Figure 2.7.

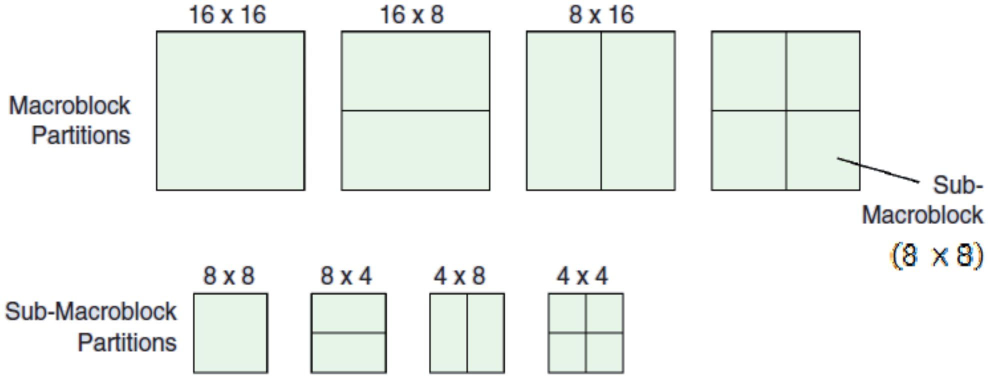


Figure 2.7 Partitioning of MBs for motion compensated prediction [29].

The 16 × 16 P- type MBs can be partitioned into luma block sizes of 16 × 8, 8 × 16 and 8 × 8. The 8 × 8 sub-MB can be further partitioned into 8 × 4, 4 × 8, or 4 × 4 samples [29]. A syntax element is added for each sub-MB to represent the partitioning used.

In multiple frame motion compensated prediction for P slices (Figure 2.8) displacement vectors (motion vector) and picture (composed of the slices in a frame or a field) reference indices are used for the predictively coded block. A multiple picture buffer is used to store the decoded reference pictures. Quarter sample precision is achievable for the individual motion vector. Corresponding samples of the reference picture pointed out by the picture reference index is used if the displacement vector points to an integer sample; else, the prediction sample is obtained using interpolation between integer sample positions. For this a six tap FIR filter [29] with weights (1, -5, 20, 20, -5, 1)/32 is used for predicting the half sample positions and the prediction values at quarter sample positions are generated by averaging samples at integer and half-sample positions (Figure 2.9). Chroma components are predicted through bilinear interpolation. The P slices are coded in skip mode if no quantized prediction error sample or a motion vector or reference index parameter is available.

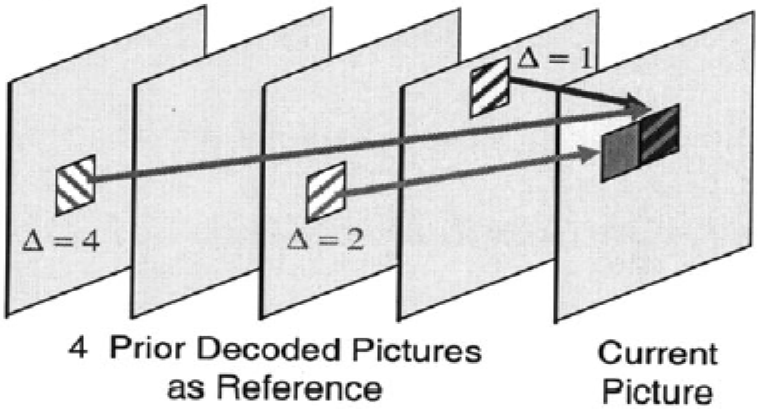


Figure 2.8 Multi-frame motion compensated prediction [30].

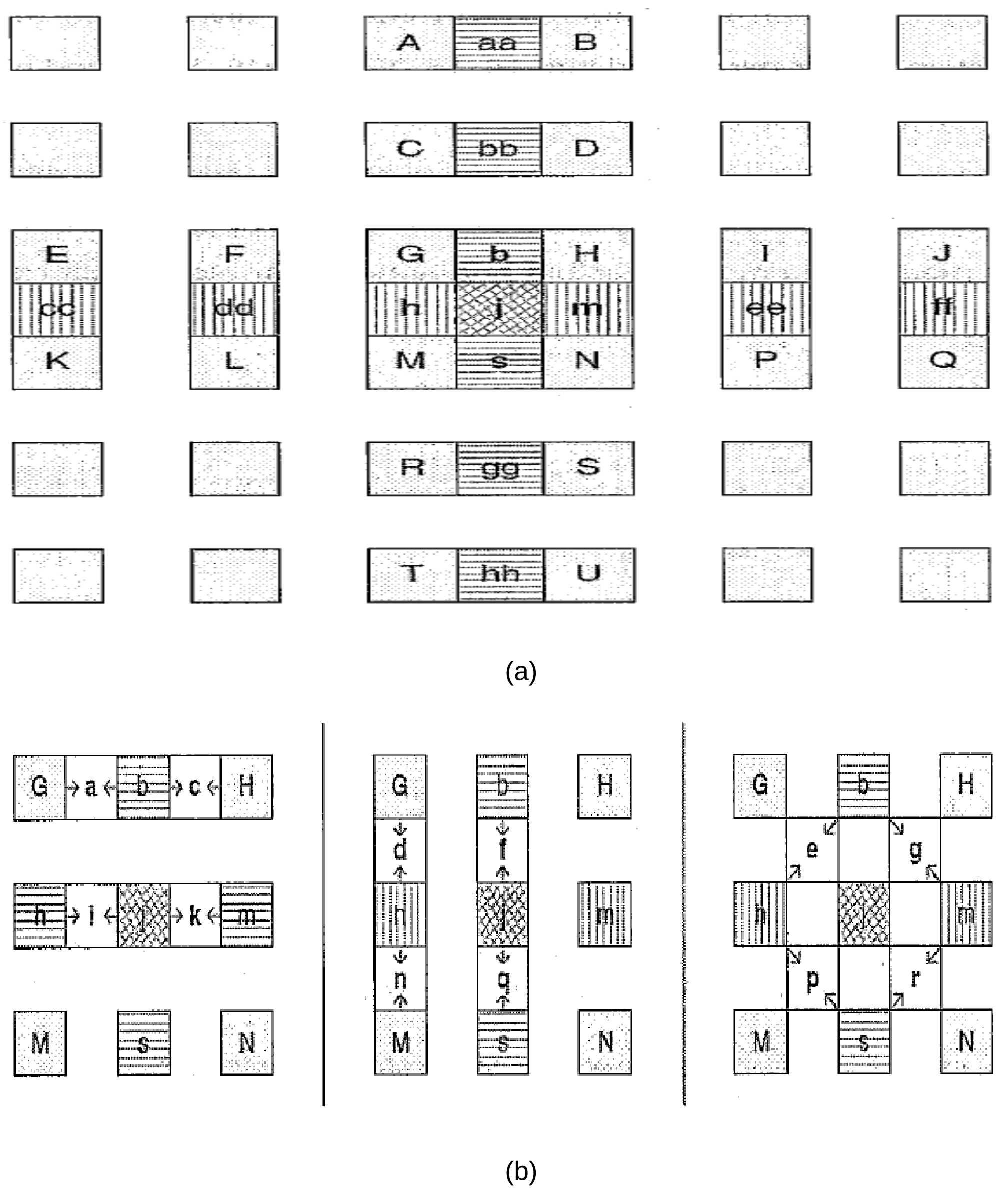


Figure 2.9 (a) Predicting half sample positions (b) Predicting quarter sample positions from the half samples [16].

In multiple frame motion compensated B slice prediction distinct, forward and backward arbitrary motion-compensated prediction values for building the prediction sample can be used. Three different types of inter-picture prediction can be chosen for each MB:

- List 0
- List 1
- Bi-Predictive (superposition of List 0 and List 1)

The B slices can also be coded in direct mode without any additional information [10].

2.2.3 Transform coding

H.264/AVC applies the transform coding to the prediction residual. The version 1 provides two different 4 × 4 transform coding methods for the luma components:

- 4 × 4 integer transform
- 4 × 4 Hadamard transform applied to the DC coefficients of 4 × 4 integer transforms.

The second method is done for intra predicted MBs of type 16 × 16. For the chroma components either the transform coding methods used are:

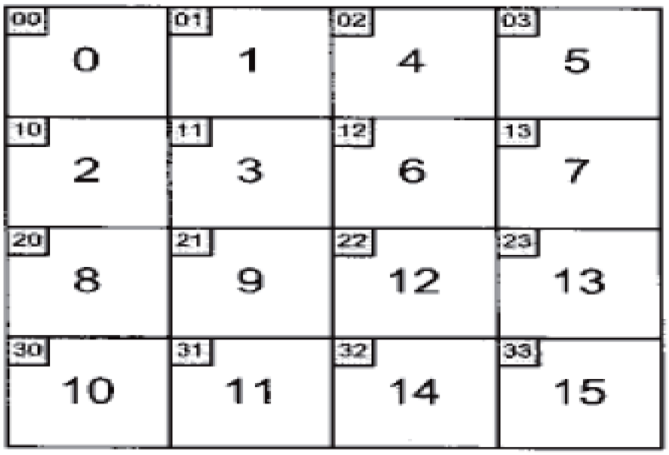
- 4 × 4 integer transform
- 4 × 4 Hadamard or 2 × 2 Haar/Hadamard transform applied to DC coefficients of 4 × 4 integer transforms depending on the chroma format.

The second method is again applied for only intra predicted MBs of type 16 × 16.

Figures 2.10 and 2.11 show the transform matrices. The entire integer transforms and its inverse can be implemented using simple shift and addition operations. The integer precision needed is 8+*b* bits for processing a *b* bit video [13].

$$H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

Figure 2.10 The 4 × 4 Integer transform [5]



00 0	01 1	02 4	03 5
10 2	11 3	12 6	13 7
20 8	21 9	22 12	23 13
30 10	31 11	32 14	33 15

(a)

$$H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \qquad H_3 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

(b)

Figure 2.11 (a) The sixteen 4 × 4 blocks with DC coefficients (b) The Hadamard transforms applied on the DC coefficients.

The main features of small transforms [9] are:

- Improved intra and inter prediction process decorrelates the samples. So the dependence on transform coding has decreased. Thus a small transform is as good as a large one.
- Smaller transforms result in less ringing artifacts
- Computational requirement is less

In HD videos the smoothness and texture is preserved if larger transform sizes are used. With this in consideration the FRExt amendment introduced the separable 8 × 8 integer transforms that approximates the 2D 8 ×8 DCT [13][14]. Figure 2.12 illustrates the transform matrix used.

$$T_{8x8} = \begin{bmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 12 & 10 & 6 & 3 & -3 & -6 & -10 & -12 \\ 8 & 4 & -4 & -8 & -8 & -4 & 4 & 8 \\ 10 & -3 & -12 & -6 & 6 & 12 & 3 & -10 \\ 8 & -8 & -8 & 8 & 8 & -8 & -8 & 8 \\ 6 & -12 & 3 & 10 & -10 & -3 & 12 & -6 \\ 4 & -8 & 8 & -4 & -4 & 8 & -8 & 4 \\ 3 & -6 & 10 & -12 & 12 & -10 & 6 & -3 \end{bmatrix}$$

Figure 2.12 The 8 × 8 integer transform matrix [4]

The forward and the inverse transforms can be implemented using simple butterfly operations. The quantization of transform coefficients is performed using uniform-reconstruction quantizer [4]. The 52 quantizer step size scaling parameters can be selected using the quantization parameter (QP). The quantized transform coefficients are zigzag scanned (Figure 2.13) and entropy coded to get the bitstream.

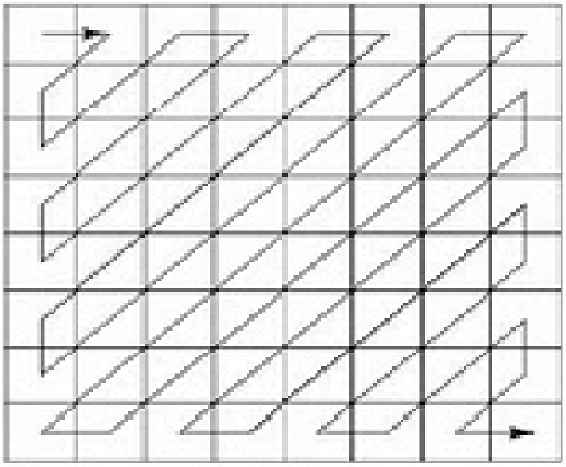


Figure 2.13 Zigzag scan

2.2.4 Entropy coding

H.264/AVC supports two entropy coding techniques:

- Context-adaptively switched sets of variable length codes(CAVLC)
- Context-based adaptive binary arithmetic coding (CABAC)

CAVLC is the baseline entropy coding method of H.264/AVC. Its basic coding tool consists of a single variable length coder (VLC) of structured Exponential Golomb codes. CAVLC utilizes VLC tables that are designed to match the conditional probabilities of the context and thus the entropy coding performance is much improved.

CABAC design is based on three components [11]:

- Binarization (a unique mapping of non-binary syntax elements to a sequence of bits called the bin string)
- Context modeling (the bin string can either be processed in the regular coding or the bypass mode depending on the context)
- Binary arithmetic coding

Compared to CAVLC, CABAC can typically provide reductions in bit rates for the same objective video quality when coding standard definition and HD videos [11].

2.2.5 Deblocking filter

Perceptual quality is of high importance to users in modern video applications. The block based coding scheme employed in H.264/AVC results in visually noticeable discontinuities along the block boundaries. So in order to improve perceptual quality and prediction efficiency H.264/AVC applies a deblocking filter to the block edges, except at picture and block boundaries (Figure 2.14). The transform size determines the block size used for deblocking. The deblocking filter is adaptive to different levels in the picture (slice level, block edge level and sample level) and thus prevents any de-blurring effects. Both horizontal and vertical edges are deblocked by this procedure [9]. The filter is applied to MBs in the raster scan order: first from left to right and then top to bottom. This is done after the MBs have been predictively reconstructed.

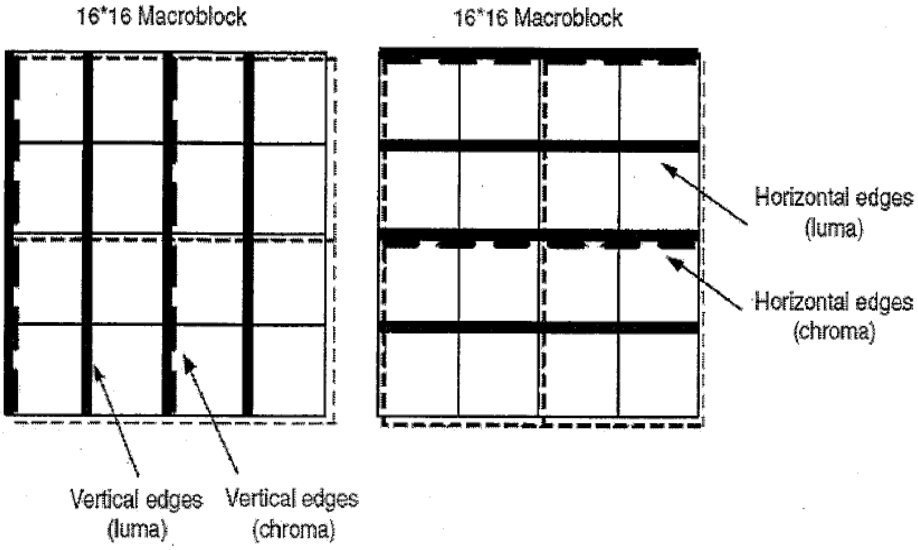


Figure 2.14 Luma and chroma boundaries to be filtered by in loop deblocking filter [16].

2.2.6 Error resilience

The encoded video should support all prevalent and future network protocols and architectures. The standard provides a video coding layer (VCL) and a network abstraction layer (NAL). The VCL deals with efficient representation of the encoded video. The

NAL deals with the interface between the codec and the outside network architectures.

Several error resilience features like parameter sets, data partitioning, flexible ordering of MBs, and flexible ordering of slices are also supported [9].

2.3 H.264/AVC decoder

The H.264/AVC decoder can be illustrated as shown in Figure 2.13.

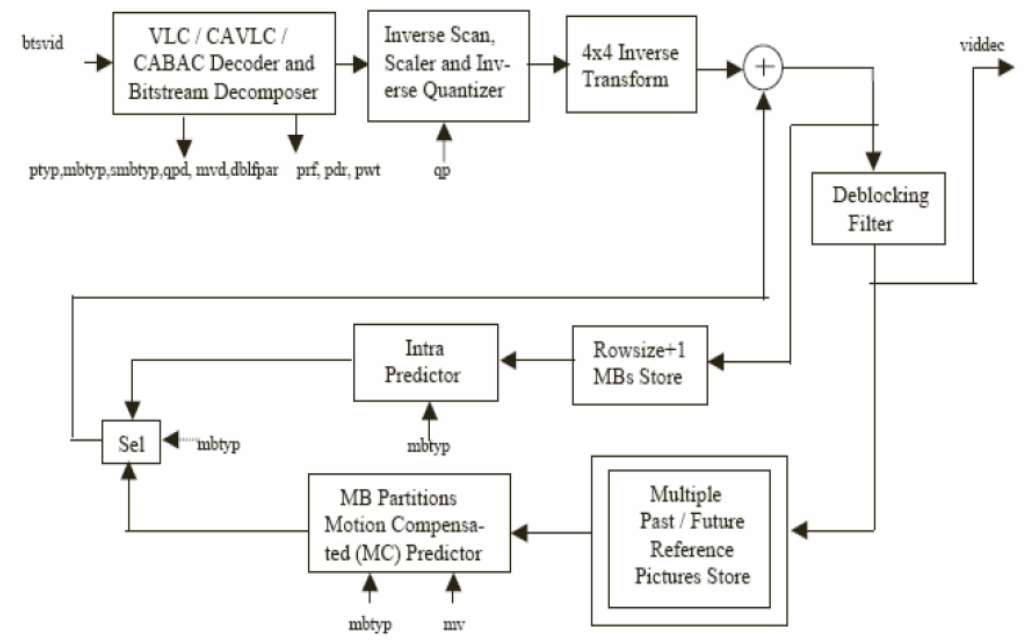

btsvid
VLC / CAVLC / CABAC Decoder and Bitstream Decomposer
Inverse Scan, Scaler and Inverse Quantizer
4x4 Inverse Transform
viddec
ptyp,mbtyp,smbtyp,qpd, mvd,dblfpar
prf, pdr, pwt
qp
Deblocking Filter
Intra Predictor
Rowsize+1 MBs Store
Sel
mbtyp
mbtyp
MB Partitions Motion Compensated (MC) Predictor
Multiple Past / Future Reference Pictures Store
mbtyp
mv

Figure 2.15 H.264/AVC decoder block diagram [12].

The decoder has to do all the inverse operations of the encoder in the reverse order. The first step is to decode the CAVLC/CABAC encoded bitstream. Then the bitstream is then rescaled, de-quantized and converted from the transform domain to get the prediction error values. The prediction errors are added to a predicted block obtained from motion vectors, reference frames and previously decoded frame. Deblocking filter is applied before the picture is displayed or stored.

CHAPTER 3

AVS-VIDEO

3.1 Introduction

Audio-video coding standard (AVS) of China was formulated by the AVS Workgroup (established in June 2002) [32]. AVS standard comprises of four main technical standards [33]:

- System
- Video
- Audio
- Digital copyright management
- Support standards such as consistency verification

AVS-video specifies the techniques used for coding video. It defines four profiles (Table 1.2) [34] targeting different applications like commercial broadcasting and storage media, mobile video having smaller picture resolution, video surveillance and multimedia entertainment respectively.

3.2 AVS-video encoder

The typical AVS-video encoder block diagram is shown in Figure 3.1 [35]. A picture of a video is split into macroblocks (MB) of size 16 × 16. A group of continuous independent MBs can be combined to form a slice. Intra or inter prediction is done for each MB in a slice. The prediction used will be the same for all MBs in a slice. Then the prediction residual, which is the difference between the original sample and the intra or inter predicted sample is transform coded. The resultant transform coefficients are quantized, scanned and entropy coded to form the bitstream. The encoder also contains a model of the decoder (Figure 3.12) that reconstructs the predicted frame/field and stores it in a buffer. They are used to decide the type of prediction for the succeeding frames/fields.

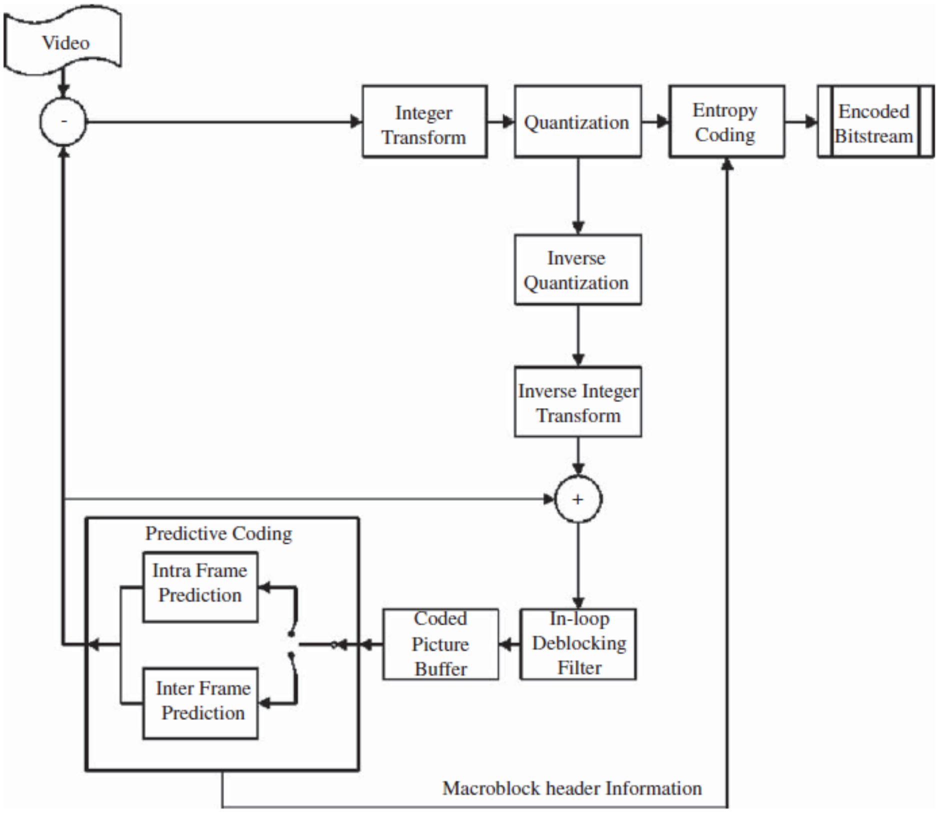


Figure 3.1 AVS-video encoder [35].

3.2.1 Intra prediction

Intra prediction exploits spatial redundancy between samples within a picture. AVS-video supports intra prediction [36] on:

- 8 × 8 blocks
- 4 × 4 blocks
- Adaptive block size (8 × 8 / 4 × 4)

The 8 × 8 block size intra prediction allows five prediction modes (Figure 3.2) for luma components: DC (Mode 2), vertical (Mode 0), horizontal (Mode 1), diagonal down left (Mode 3), diagonal down right (Mode 4). The four 8 × 8 luma blocks in a 16 × 16 block can each be predicted using any of the above five modes. But before using modes 2, 3 and 4 a three tap low pass filter with weights (1, 2, and 1) is applied on the reference samples. This is to prevent visible artifacts due to large size blocks for intra prediction [37]. The four prediction modes for chroma components are: DC, vertical, horizontal and plane.

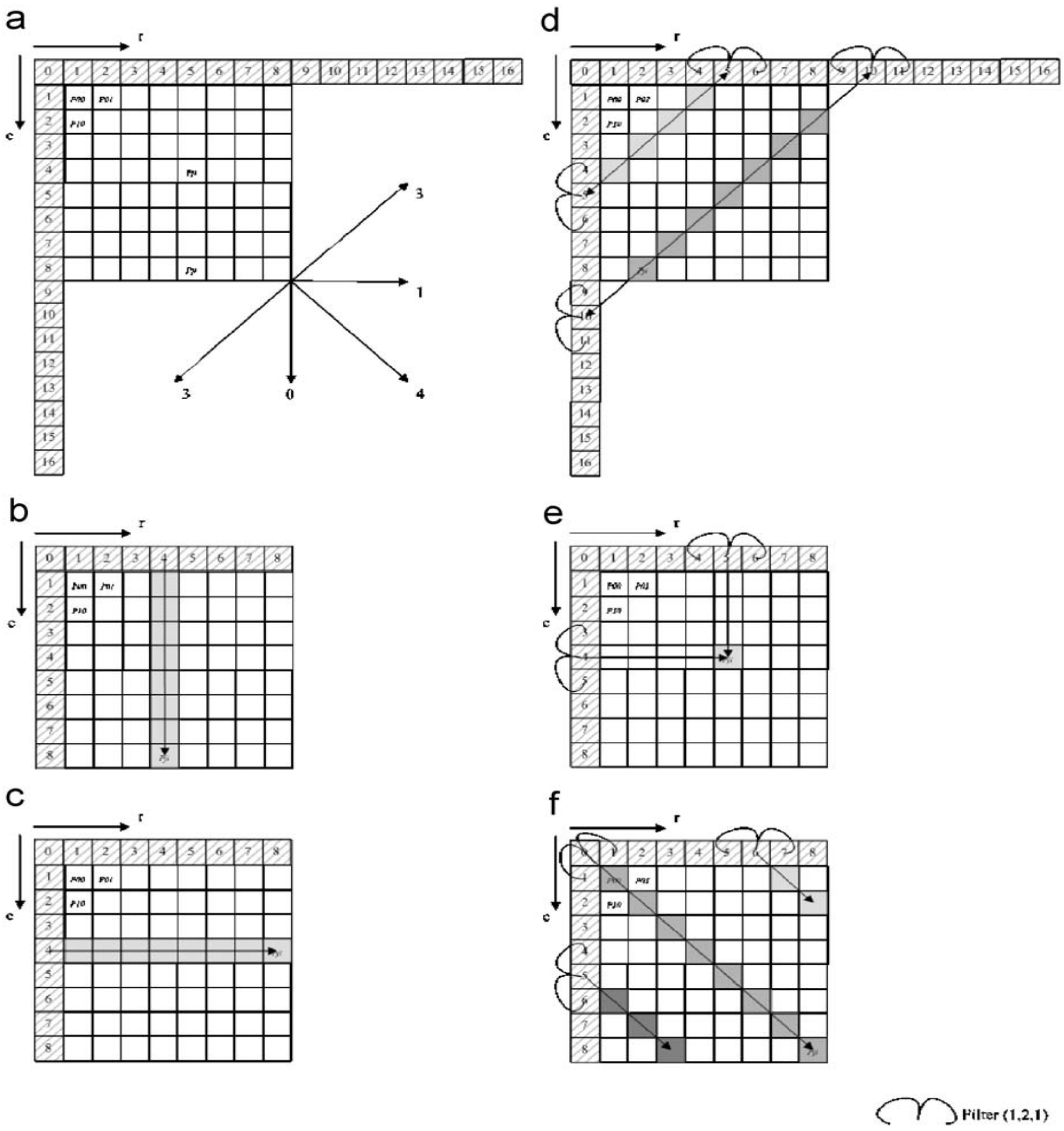
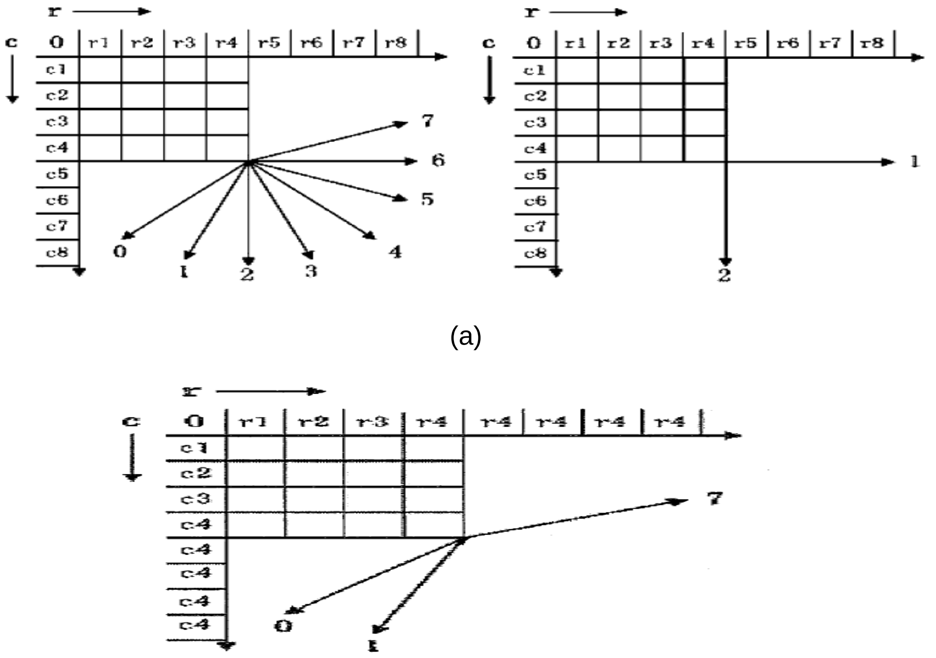


Figure 3.2 Five prediction modes for 8 × 8 luma blocks [35].

Intra prediction using 4 × 4 blocks gives better coding efficiency for low-resolution videos [52]. There are nine modes (Figure 3.3 (a)) that invoke some specific techniques like:

- Direct intra prediction (DIP) in which the all the 4 × 4 luma sub-blocks in MBs marked as Dip mode takes the most probable mode as its intra prediction mode even if it is different for each.
- Padding before prediction (PBP) where the reference samples r5, r6, r7 and r8 (Figure 3.3 (b)) are padded from r4, and c5, c6, c7 and c8 are padded from c4, so as to skip the conditional availability test of up-right (Mode 7) and down-left (Modes 0 and 1) reference samples in both the luma and chroma components [52].
- Simplified chrominance intra prediction (SCI) which limits the availability of intra prediction modes for the chroma components to three: DC, vertical and horizontal modes [38].



(a)

Figure 3.3 (a) Directional modes for 4 × 4 luma (left) and chroma intra prediction (right)
(b) Padding of r5, r6, r7, r8 from r4 and c5, c6, c7, c8 from c4 [52].

In adaptive block size intra prediction the 4 × 4 block size can be used adaptively with the 8 × 8 blocks. An indicator in the MB header notifies its use [39]. If the block sizes used by the current block and the neighboring blocks are different a mapping between the modes used by the two block sizes is needed (Table 3.1).

Table 3.1 Mapping between 4 × 4 and 8 × 8 intra prediction modes [35].

4 × 4 intra prediction modes	8 × 8 intra prediction modes
0-Intra_Luma_Down_Left	3-Intra_8_8_Down_Left
1-Intra_Luma_Vertical_Left	
2-Intra_Luma_Vertical	0-Intra_8_8_Vertical
3-Intra_Luma_Vertical_Right	
4-Intra_Luma_Down_Right	4-Intra_8_8_Down_Right
5-Intra_Luma_Horizontal_Down	
6-Intra_Luma_Horizontal	1-Intra_8_8_Horizontal
7-Intra_Luma_Horizontal_Up	
8-Intra_Luma_DC	2 –Intra_8_8_DC

3.2.2 Inter prediction

Inter frame prediction exploits the temporal redundancy in a video sequence by utilizing previously decoded frames/fields. Inter frame prediction can be done for 16 × 16, 16 × 8, 8 × 16, 8 × 8 block sizes. A maximum of two reference frames or four reference fields are used for inter frame coding. The different techniques used for inter prediction are:

- P-prediction
- Bi-prediction
- Interpolation
- Motion vector prediction

In P-prediction the reference frame/field should be previous to the currently coded frame/field in display order. Only one motion vector and reference index is used in this scheme. The prediction involves five MB modes as listed in Table 3.2 [35].

Table 3.2 Five modes in P-prediction [35].

MB type	MvNum (Mode number)
P_Skip	0
P_16 × 16	1
P_16 × 8	2
P_8 × 16	3
P_8 × 8	4

Bi-prediction uses both forward and backward decoded reference frames/fields for prediction. Two motion vectors and references indices are used here. Bi-prediction can be further subdivided into direct prediction (DP) and symmetric prediction (SP) [40]. The forward and backward motion vectors in DP are obtained from the motion vector of its corresponding block in the backward reference and vary with the temporal block distance between predicted and reference blocks. The forward motion vector in SP is transmitted for each partition of the current MB while the backward motion vector is derived from the forward motion vector by the symmetric rule (Figure 3.4).

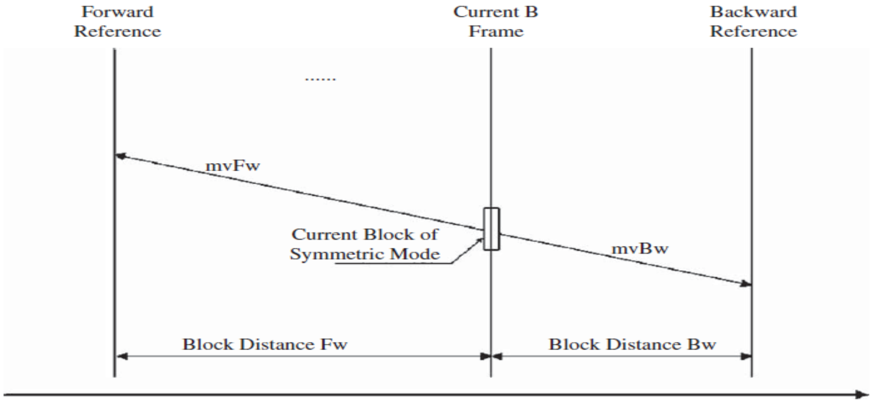


Figure 3.4 Symmetric prediction AVS-video [40].

Interpolation allows quarter sample motion vector accuracy. The sub-sample interpolation is called as two steps four taps (TSFT) interpolation [41] .The first step derives half-sample values and the filter coefficients for the first step are (-1, 5, 5, -1). The second step with filter coefficients (1, 7, 7, 1) derives the quarter-sample values from the half-samples (Figure 3.5). An adaptive interpolation filter is also defined [42] but it involves higher computational complexity.

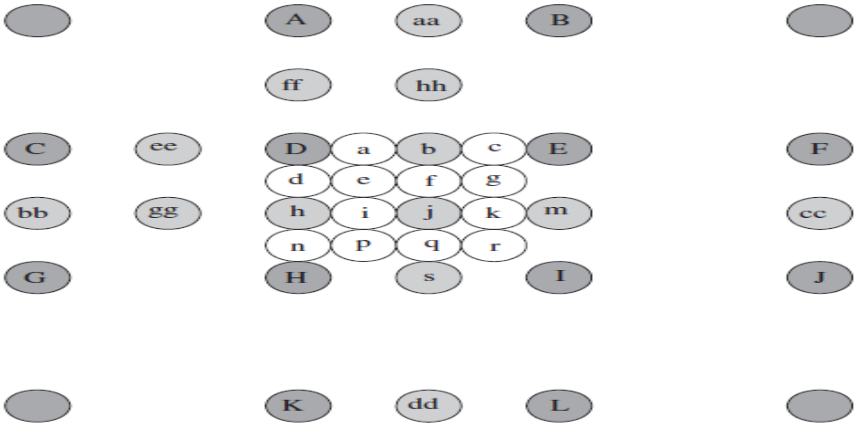


Figure 3.5 Quarter-sample interpolation [52].

In AVS-video the difference between the predicted motion vector and the actual motion vector is transmitted in the bitstream. Motion vector prediction (Figure 3.6) takes the actual motion vectors of the neighboring blocks and scales them to motion vector candidates of the current block. The scaling factor depends on the temporal distance between the current and the reference frames/fields.

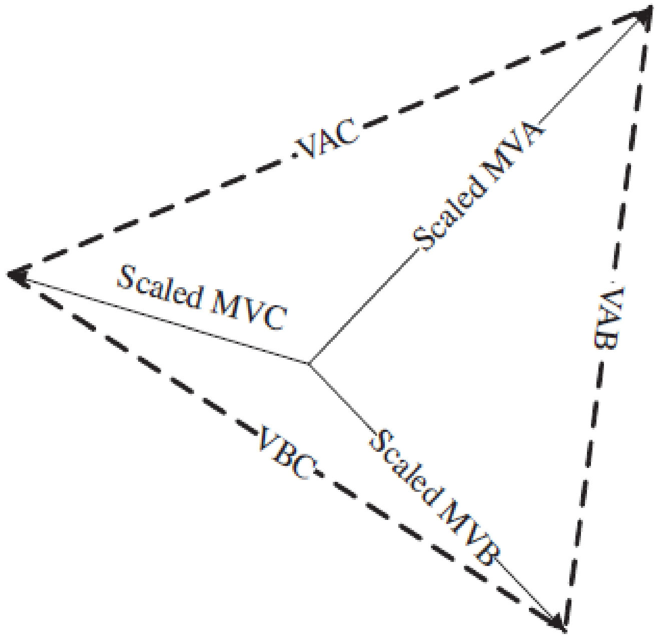


Figure 3.6 Motion vector prediction (Scaled MVC is predicted from motion vectors of neighboring blocks; Scaled MVB and Scaled MVA) [35].

3.2.3 Transform coding

In AVS-video, transform coding is carried out on prediction residuals. Due to considerations of implementation complexity, the forward and the inverse transforms are designed as non-normalized ICTs (NNICT) [43]. The integer transforms can be divided into two types depending on the size of the transform kernel:

- 8 × 8 block size transform
- 16 × 16 block size transform

The 8 × 8 NNICT, an approximation of 8 × 8 DCT, is performed with the pre-scaled integer transform (PIT) technique [43]. The transform kernel is illustrated in Figure 1.5. The 16 × 16 NNICT which has been recently added is an extension of 8 × 8 NNICT as shown in Figure 3.7. It is particularly useful for coding high definition videos [39].

$$E_{16} = \begin{bmatrix}
8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\
10 & 10 & 9 & 9 & 6 & 6 & 2 & 2 & -2 & -2 & -6 & -6 & -9 & -9 & -10 & -10 \\
10 & 10 & 4 & 4 & -4 & -4 & -10 & -10 & -10 & -10 & -4 & -4 & 4 & 4 & 10 & 10 \\
9 & 9 & -2 & -2 & -10 & -10 & -6 & -6 & 6 & 6 & 10 & 10 & 2 & 2 & -9 & -9 \\
8 & 8 & -8 & -8 & -8 & -8 & 8 & 8 & 8 & 8 & -8 & -8 & -8 & -8 & 8 & 8 \\
6 & 6 & -10 & -10 & 2 & 2 & 9 & 9 & -9 & -9 & -2 & -2 & 10 & 10 & -6 & -6 \\
4 & 4 & -10 & -10 & 10 & 10 & -4 & -4 & -4 & -4 & 10 & 10 & -10 & -10 & 4 & 4 \\
2 & 2 & -6 & -6 & 9 & 9 & -10 & -10 & 10 & 10 & -9 & -9 & 6 & 6 & -2 & -2 \\
2 & -2 & -6 & 6 & 9 & -9 & -10 & 10 & 10 & -10 & -9 & 9 & 6 & -6 & -2 & 2 \\
4 & -4 & -10 & 10 & 10 & -10 & -4 & 4 & -4 & 4 & 10 & -10 & -10 & 10 & 4 & -4 \\
6 & -6 & -10 & 10 & 2 & -2 & 9 & -9 & -9 & 9 & -2 & 2 & 10 & -10 & -6 & 6 \\
8 & -8 & -8 & 8 & -8 & 8 & 8 & -8 & 8 & -8 & -8 & 8 & -8 & 8 & 8 & -8 \\
9 & -9 & -2 & 2 & -10 & 10 & -6 & 6 & 6 & -6 & 10 & -10 & 2 & -2 & -9 & 9 \\
10 & -10 & 4 & -4 & -4 & 4 & -10 & 10 & -10 & 10 & -4 & 4 & 4 & -4 & 10 & -10 \\
10 & -10 & 9 & -9 & 6 & -6 & 2 & -2 & -2 & 2 & -6 & 6 & -9 & 9 & -10 & 10 \\
8 & -8 & 8 & -8 & 8 & -8 & 8 & -8 & 8 & -8 & 8 & -8 & 8 & -8 & 8 & -8
\end{bmatrix}$$

Figure 3.7 16 × 16 NNICT [35].

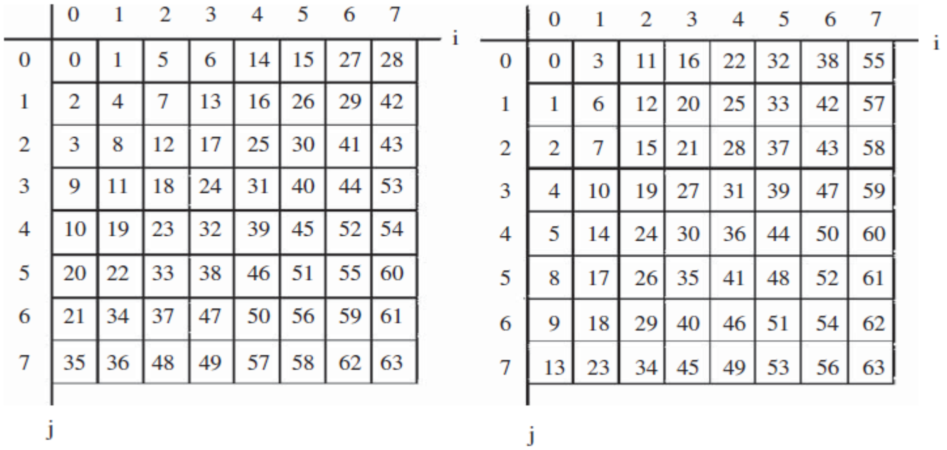
3.2.4 Quantization and scanning

In frequency un-weighted quantization [35] all the coefficients of 8 × 8 transform in AVS-video are quantized with the same quantization parameter (QP).The quantization parameter can take 64 values. For the coefficients of 4 × 4 transform a QP shift parameter is transmitted in the bitstream [39]. The QP for blocks applying 4 × 4 transform will be the resultant of subtracting QP shift from original QP. The adaptive quantization mode [44] provides frequency weighted quantization modes: normal, detail-protecting and non-detail-protecting with the purpose of preserving details or eliminating details by using different weighting quantization matrices for different frequency bands. Normal mode is the same as un-weighted quantization. The other two modes use three pre-defined frequency band distributions, a set of weighting parameters - (128, 167, 154, 141, 141, 128) for detail-protecting mode and (122, 115, 115, 102, 102, 78) for non-detail protecting mode [44] and a model parameter to produce frequency-weighted quantization matrices.

Scene - adaptive weighted quantization [44] can be used to modify quantization parameters separately for 'U' and 'V' components of chroma. Scanning of the transform coded prediction residuals can be done in two ways:

- Fixed order scan
- Adaptive Scan

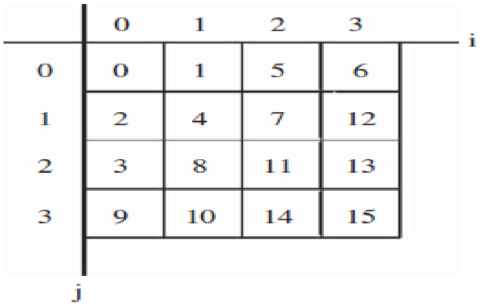
In fixed order scan the order of scanning 8 × 8 blocks differ in MBs coded with frame or field coding (Figure 3.8). The zigzag scan is adopted for 4 × 4 blocks (Figure 3.9).



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

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

Figure 3.8 Scan order for 8 × 8 blocks in frame coding (left) and field coding (right) [35].



j \ i	0	1	2	3
0	0	1	5	6
1	2	4	7	12
2	3	8	11	13
3	9	10	14	15

Figure 3.9 Scan order for 4 × 4 blocks [52].

Adaptive scan technique [45] involves the adaptive use of both the scanning techniques (Figures 3.9 and 3.10) in conjunction with adaptive block transforms to provide for the best scan order for the current frame/field.

3.2.5 Entropy coding

The entropy coding methods employed in AVS-video are: variable length coding (VLC) enhanced arithmetic coding (EAC). VLC provides low complexity less efficient coding by either fixed length or exp-Golomb codes. The syntax elements are coded using VLC while for quantized transform coefficients a more efficient scheme 2-D VLC [46] is adopted.

EAC provides higher coding efficiency at relatively higher computational complexity. The encoding process consists of three steps [47]:

- Binarization where a given non-binary valued syntax element is uniquely mapped to a binary sequence
- Context modeling where the statistics of the already coded syntax elements are used to estimate probability of the current syntax element
- Binary arithmetic coding

3.2.6 In-loop deblocking filter

Deblocking filters remove blocking artifacts present in the video. Several application specific deblocking filters are employed in AVS-video. The default deblocking filter is defined to operate on 8 × 8 blocks. The vertical boundaries are first filtered horizontally followed by the horizontal boundaries filtered vertically (Figure 3.10). A much more simplified deblocking filter is defined for the 4 × 4 blocks [48]. Moreover, a sample/pixel level deblocking filter [49] can also be used.

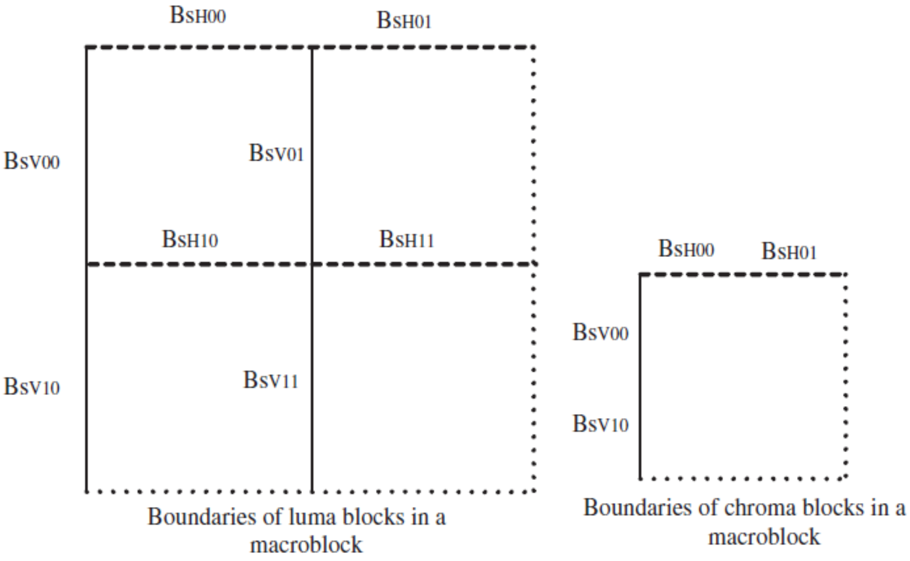

BSH00
BSH01
BSV00
BSV01
BSH10
BSH11
BSV10
BSV11
Boundaries of luma blocks in a macroblock
BSH00
BSH01
BSV00
BSV10
Boundaries of chroma blocks in a macroblock

Figure 3.10 Vertical boundaries (solid line) and horizontal boundaries (broken lines) in an 8 × 8 block of luma (left) and chroma (right) [35].

3.2.7 Error resilience

Error resilience in AVS-video is achieved through: scene signaling where the type of scene transition can be used to construct lost parts of the current frame/picture from previous frame/picture/, specifying core frame/picture [50], using a flexible picture header [51], using flexible slice set [51] and using constrained DC intra prediction mode which constrains the prediction values of DC mode to be fixed to 128 for 8 × 8 luma blocks to prevent error accumulation [51].

3.3 AVS-video decoder

The AVS-video decoder (Figure 3.11) takes in the encoded bitstream with all the syntax elements and reconstructs the original video sequence from the bitstream. The bitstream is VLC or EAC decoded first. As the second step it is inverse quantized depending on the quantization parameter. Then it is inverse transform coded to get the prediction residual. After that depending on whether it is intra frame/field coded or inter frame/field coded, intra prediction or motion compensated inter prediction is done respectively. The reconstructed frame/field is stored in a frame buffer for further predicting successive frames/fields.

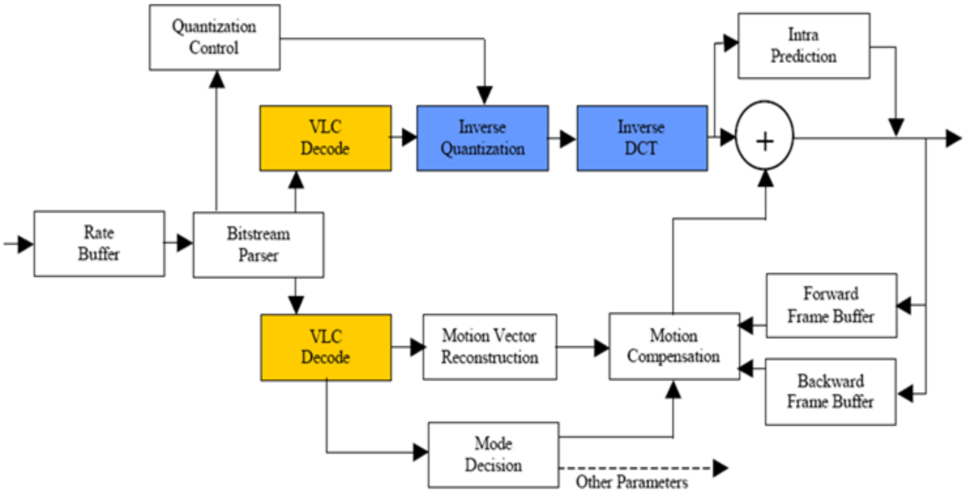


Figure 3.11 AVS-video decoder [50].

The main features of AVS-video standard were explained in this chapter. The next chapter will be dealing with the utility of higher order 2-D 16 × 16 ICTs in HD video coding. Different types of 2-D 16 × 16 ICTs will be introduced and their implementation in AVS-video and H.264/AVC will be discussed.

CHAPTER 4

HIGHER ORDER 2-D ICTS FOR HD VIDEO CODING

4.1 Introduction

Spatial correlation in pictures/frames of a video is the statistical measure of the degree of dependency between samples/pixels that are near to each other [53]. HD videos have higher spatial correlation when compared to lower resolution videos [18]. The correlation coefficient r (4.1) measures the correlation of nearby samples/pixels, and can be considered to be a stochastic process [55].

$$r(n_1, n_2) = E[(x(n_1) - \mu_1) \cdot (x(n_2) - \mu_2)]/\sigma_1\sigma_2 \tag{4.1}$$

The symbols n_1 and n_2 denote nearby samples, x (n_1) and x (n_2) represent the intensity values of the samples n_1 and n_2, μ_1, μ_2 and σ_1, σ_2 represent the respective means and standard deviations and E is the expectation operator. Table 4.1 shows the correlation coefficients computed for prediction residuals of video sequences belonging to different resolutions. Here r (1) is the correlation of adjacent pixels and r (2) the correlation of alternate pixels, μ_{r1} and μ_{r2} are their respective means and σ_{r1} and σ_{r2} are their respective standard deviations. From Table 4.1 we can see that μ_{r1} of HD video sequences are relatively higher than that of lower resolutions like wide video graphics array (WVGA) and wide quarter video graphics array (WQVGA). Moreover, μ_{r2} for lower resolution video sequences are almost zero while for HD sequences it remains high. Thus, HD videos show higher spatial correlation in general. This property of HD videos can be exploited to achieve better coding efficiency by using higher order ICTs [54].

In the succeeding sections the development of ICTs from DCT-II in general and three specific 2-D order 16 ICTs that will be implemented in H.264/AVC and AVS-video particularly for HD video coding will be discussed.

Table 4.1 Correlation of nearby pixels for various video resolutions.

Test sequences	Resolution	r(1)		r(2)	
		μ_{r1}	σ_{r1}	μ_{r2}	σ_{r2}
Kimono	1920 × 1080 (HD)	0.8673	0.1284	0.7311	0.1434
Parkscene		0.7431	0.1820	0.6695	0.1967
Cactus		0.8542	0.1692	0.7483	0.1245
Vidyo1	1280 × 720 (HD)	0.7539	0.2401	0.4073	0.1842
Vidyo2		0.6643	0.1982	0.3060	0.1569
Vidyo3		0.5474	0.1125	0.3221	0.2923
PartyScene	832 × 480 (WVGA)	0.4953	0.1598	0.2019	0.1757
BQMall		0.4517	0.2145	0.1966	0.2450
BasketballDrill		0.5594	0.1183	0.2301	0.1032
BQSquare	416 × 240 (WQVGA)	0.3543	0.2935	0.0964	0.1722
BlowingBubbles		0.2879	0.1515	0.0473	0.1906
BaketballPass		0.2177	0.1784	0.0355	0.2098

4.2 Integer cosine transforms

DCT-II [2] maps a vector x of length N into another vector X of transform coefficients having the same length (4.2).

$$X = H\ \ x \qquad (4.2)$$

Here the matrix H, called the transform matrix, can be defined as in equation (4.3).

$$H(k,n)\ =\ c_k\sqrt{\frac{2}{N}}\ cos[(n+0.5)\frac{k\pi}{N}] \qquad (4.3)$$

where $k,\ n= 0,1,.....,N\text{-}1$

$$c_k = \frac{1}{\sqrt{2}} for\ k = 0$$

$$c_k = 1 \; for \; k \neq 0$$

The elements of the matrix H and H^T are irrational numbers. Thus, the finite bit precision in a computer will not reconstruct the same data if forward and inverse transforms are done in cascade. Moreover, if the forward and the inverse transforms are implemented in different machines with different floating-point representations, the error can be large. Scaling the matrices H and H^T and then rounding it to the nearest integer can negate the errors. But, if the scaling factors are large, the norms of the rows (basis vectors) turn out to be very high and the computational complexity will increase. Thus, an orthogonal (4.4) matrix H with small integer elements is highly desired.

$$H\,H^T = I \tag{4.3}$$

This has led to the development of ICTs by the principle of dyadic symmetry [6]. The ICTs maintain the structure such as relative magnitudes, signs, dyadic symmetries, and orthogonality among the elements of the transform matrix.

4.3 Simple 2-D order 16 ICT

The order 16 ICT is developed from the DCT-II matrix defined in equation (4.5) [8].

$$D(i,j) = \begin{cases} \frac{1}{4}, & i = 1 \\ \frac{\sqrt{2}}{4} cos\left[\frac{(i-1)\,(j-0.5)\pi}{16}\right], & 1 \leq j \leq 16 \end{cases} \tag{4.5}$$

The matrix D can be modified using the principle of dyadic symmetry [6] into matrix T_{16} as shown in Figure 4.1. The dots show that the matrix extends to the right with alternating even and odd symmetry along the solid line drawn inside.

$$T_{16} = \left[\begin{array}{cccccccc|ccc}
x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & \cdots \\
x_1 & x_3 & x_5 & x_7 & x_9 & x_{11} & x_{13} & x_{15} & -x_{15} & -x_{13} & \cdots \\
x_2 & x_6 & x_{10} & x_{14} & -x_{14} & -x_{10} & -x_6 & -x_2 & -x_2 & -x_6 & \cdots \\
x_3 & x_9 & x_{15} & -x_{11} & -x_5 & -x_1 & -x_7 & -x_{13} & x_{13} & x_7 & \cdots \\
x_4 & x_{12} & -x_{12} & -x_4 & -x_4 & -x_{12} & x_{12} & x_4 & x_4 & x_{12} & \cdots \\
x_5 & x_{15} & -x_7 & -x_3 & -x_{13} & x_9 & x_1 & x_{11} & -x_{11} & -x_1 & \cdots \\
x_6 & -x_{14} & -x_2 & -x_{10} & x_{10} & x_2 & x_{14} & -x_6 & -x_6 & x_{14} & \cdots \\
x_7 & -x_{11} & -x_3 & x_{15} & x_1 & x_{13} & -x_5 & -x_9 & x_9 & x_5 & \cdots \\
x_8 & -x_8 & -x_8 & x_8 & x_8 & -x_8 & -x_8 & x_8 & x_8 & -x_8 & \cdots \\
x_9 & -x_5 & -x_{13} & x_1 & -x_{15} & -x_3 & x_{11} & x_7 & -x_7 & -x_{11} & \cdots \\
x_{10} & -x_2 & x_{14} & x_6 & -x_6 & -x_{14} & x_2 & -x_{10} & -x_{10} & x_2 & \cdots \\
x_{11} & -x_1 & x_9 & x_{13} & -x_3 & x_7 & x_{15} & -x_5 & x_5 & -x_{15} & \cdots \\
x_{12} & -x_4 & x_4 & -x_{12} & -x_{12} & x_4 & -x_4 & x_{12} & x_{12} & -x_4 & \cdots \\
x_{13} & -x_7 & x_1 & -x_5 & x_{11} & x_{15} & -x_9 & x_3 & -x_3 & x_9 & \cdots \\
x_{14} & -x_{10} & x_6 & -x_2 & x_2 & -x_6 & x_{10} & -x_{14} & -x_{14} & x_{10} & \cdots \\
x_{15} & -x_{13} & x_{11} & -x_9 & x_7 & -x_5 & x_3 & -x_1 & x_1 & -x_3 & \cdots
\end{array}\right]$$

Figure 4.1 Matrix T_{16} generated from DCT-II matrix D [8].

The 15 elements (x_0, x_1, etc.) are chosen to be integers that maintain the orthogonality of T_{16}. The selection can be considered to be a tradeoff between complexities of implementation, transform coding gain and coding efficiency [56]. The T_{16} proposed by Fong [56] is an extended version of the order 8 ICT used in AVS-video or H.264/AVC, depending on the standard where it has to be integrated. Thus two different T_{16} can be generated for implementation in AVS-video and H.264/AVC. The flow diagrams for the order 8 ICTs are shown in Figures 4.2 and 4.3 [16]. They can be implemented using basic additions and shift operations.

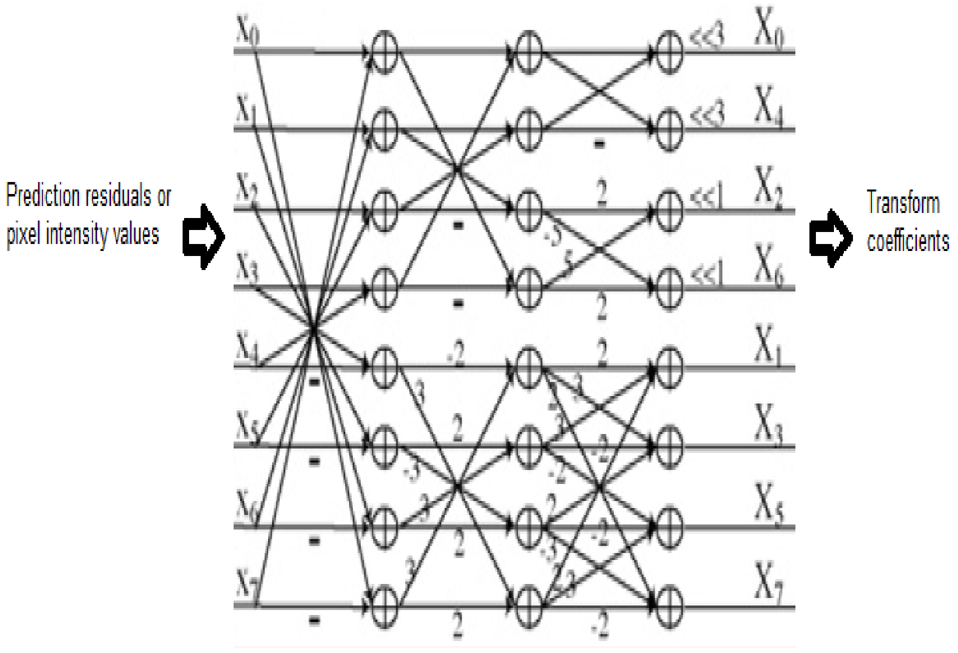


Figure 4.2 Flow diagram for 8 point forward ICT in AVS-video [16].

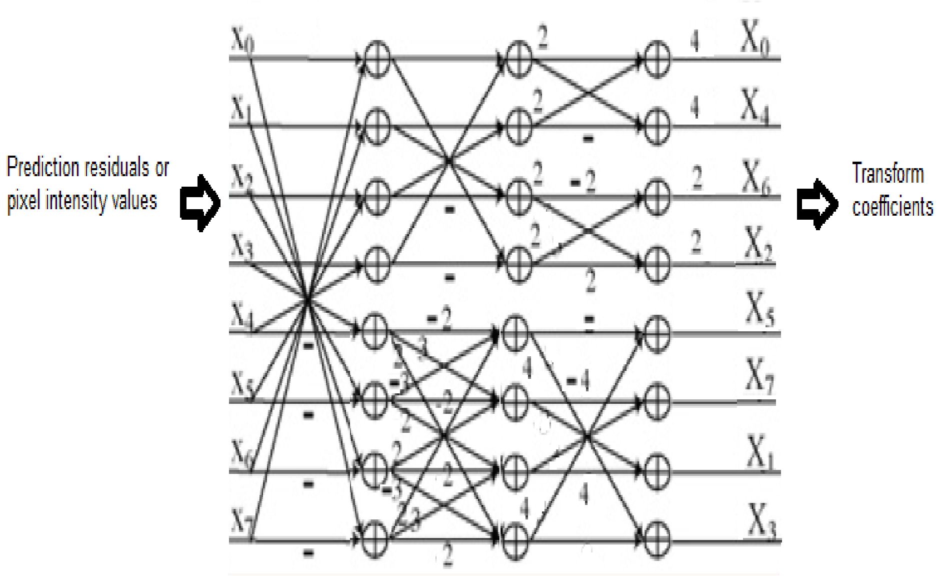


Figure 4.3 Flow diagram for 8 point forward ICT in H.264/AVC [16].

Here the elements x_0, x_1…. x_7 represent prediction residuals or pixel intensity values and X_0, X_1…. X_7 represent transform coefficients. The flow diagram for T_{16} comprises of two such 8 point forward ICT blocks implemented in parallel as shown in Figure 4.4 [56].

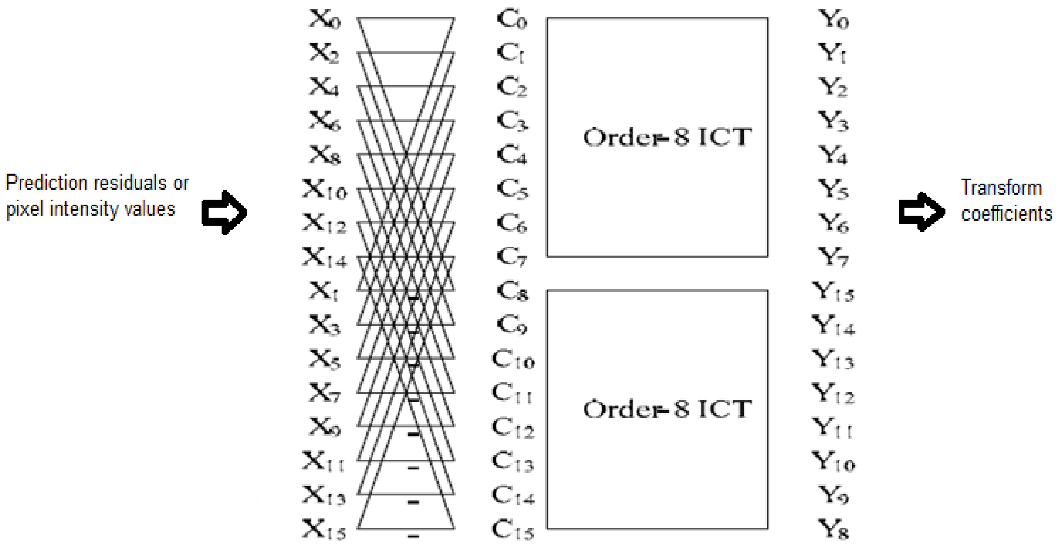


Figure 4.4 Flow diagram for 1-D order 16 forward ICT [56].

In Figure 4.4 elements X_0, X_1…. X_{15} represent prediction residuals or pixel intensity values, C_0, C_1…. C_{15} are the intermediate results from addition operation and Y_0, Y_1…. Y_{15} are the resulting transform coefficients. The separability property allows 2-D 16 × 16 ICT to be implemented using 1-D order 16 ICT (Figure 1.1). The number of additions and shifts required for the implementation are shown in Table 4.2.

Table 4.2 Number of operations [56].

Simple order 16 ICT	Number of shifts	Number of additions	Total number of operations
	24	88	112

4.4 Modified 2-D order 16 ICT

The magnitude of matrix elements in odd part (having odd symmetry) is relatively higher than the even part (having even symmetry) of the general transform T_{16}. Thus a modified order 16 ICT was proposed by Dong et al. [16], where the even part is kept the same and the odd part is redesigned to reduce computational complexity with respect to order 16 DCT-II. Thus, two different order 16 ICTs can be generated for implementing in AVS-video and H.264/AVC respectively. Figure 4.5 shows the even (T_{8e}), odd parts (T_{8o}) of T_{16} and the modified odd part (M_{8o}).

$$T_{8e} = \begin{bmatrix} x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 \\ x_2 & x_6 & x_{10} & x_{14} & -x_{14} & -x_{10} & -x_6 & -x_2 \\ x_4 & x_{12} & -x_{12} & -x_4 & -x_4 & -x_{12} & x_{12} & x_4 \\ x_6 & -x_{14} & -x_2 & -x_{10} & x_{10} & x_2 & x_{14} & -x_6 \\ x_8 & -x_8 & -x_8 & x_8 & x_8 & -x_8 & -x_8 & x_8 \\ x_{10} & -x_2 & x_{14} & x_6 & -x_6 & -x_{14} & x_2 & -x_{10} \\ x_{12} & -x_4 & x_4 & -x_{12} & -x_{12} & x_4 & -x_4 & x_{12} \\ x_{14} & -x_{10} & x_6 & -x_2 & x_2 & -x_6 & x_{10} & -x_{14} \end{bmatrix}$$

(a)

$$T_{8o} = \begin{bmatrix} x_1 & x_3 & x_5 & x_7 & x_9 & x_{11} & x_{13} & x_{15} \\ x_3 & x_9 & x_{15} & -x_{11} & -x_5 & -x_1 & -x_7 & -x_{13} \\ x_5 & x_{15} & -x_7 & -x_3 & -x_{13} & x_9 & x_1 & x_{11} \\ x_7 & -x_{11} & -x_3 & x_{15} & x_1 & x_{13} & -x_5 & -x_9 \\ x_9 & -x_5 & -x_{13} & x_1 & -x_{15} & -x_3 & x_{11} & x_7 \\ x_{11} & -x_1 & x_9 & x_{13} & -x_3 & x_7 & x_{15} & -x_5 \\ x_{13} & -x_7 & x_1 & -x_5 & x_{11} & x_{15} & -x_9 & x_3 \\ x_{15} & -x_{13} & x_{11} & -x_9 & x_7 & -x_5 & x_3 & -x_1 \end{bmatrix} .$$

(b)

$$M_{8o} = \begin{bmatrix} x_1 & x_3 & x_5 & x_7 & x_9 & x_{11} & x_{13} & x_{15} \\ x_9 & x_{11} & x_{13} & x_{15} & -x_1 & -x_3 & -x_5 & -x_7 \\ x_5 & x_7 & -x_1 & -x_3 & -x_{13} & -x_{15} & x_9 & x_{11} \\ x_{15} & x_{13} & -x_{11} & -x_9 & x_7 & x_5 & -x_3 & -x_1 \\ x_{13} & -x_{15} & -x_9 & x_{11} & x_5 & -x_7 & -x_1 & x_3 \\ x_3 & -x_1 & -x_7 & x_5 & -x_{11} & x_9 & x_{15} & -x_{13} \\ x_7 & -x_5 & x_3 & -x_1 & -x_{15} & x_{13} & -x_{11} & x_9 \\ x_{11} & -x_9 & x_{15} & -x_{13} & x_3 & -x_1 & x_7 & -x_5 \end{bmatrix} .$$

(c)

Figure 4.5 (a) Even part of T_{16} (b) Odd part of T_{16} (c) Modified odd part [16].

The selection of elements in M_{8o} is based on three considerations:

- The magnitudes of the elements should be comparable to the even part T_{8e}
- The second basis vector of the resultant order modified ICT (MICT) should be similar to T_{16}
- Fast algorithms can be developed for implementing it.

Since the elements of M_{8o} are selected without orthogonality constraint, they can have a smaller magnitude than the elements of T_{8o} without affecting the inherent orthogonal property of T_{16}. The first three basis vectors of resultant MICT matrix (M_{16}) are similar to T_{16}. So the low frequency coefficients are unchanged and good energy compaction is achieved for HD videos. Thus a tradeoff can be achieved between computational complexity and performance in the high frequency end of the coefficients. This has led to the choice of (x_1, x_3, x_5, x_7, x_9, x_{11}, x_{13}, x_{15}) as (11, 11, 11, 9, 8, 6,4, 1) [16]. The M_{16} can be implemented using basic addition and shift operations (Figure 4.6). The even part, which is T_{8e} itself, can be implemented as in Figures 4.2 and 4.3. The matrix M_{8o} could be split into three matrices M_1, M_2 and M_3 such that:

$$M_{8o} = M_1 \times M_2 \times M_3 \tag{4.6}$$

The matrices M_1, M_2 and M_3 can be represented using additions and shifts as explained in [57].

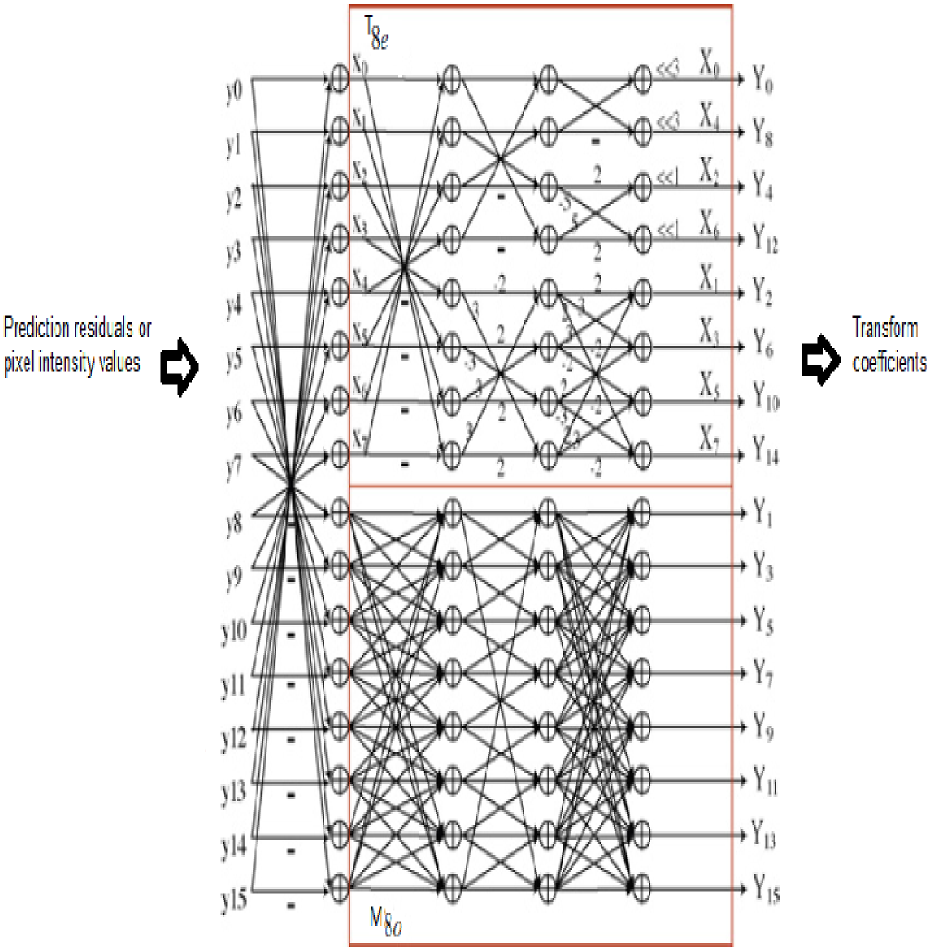


Figure 4.6 Flow diagram for M_{16} [57].

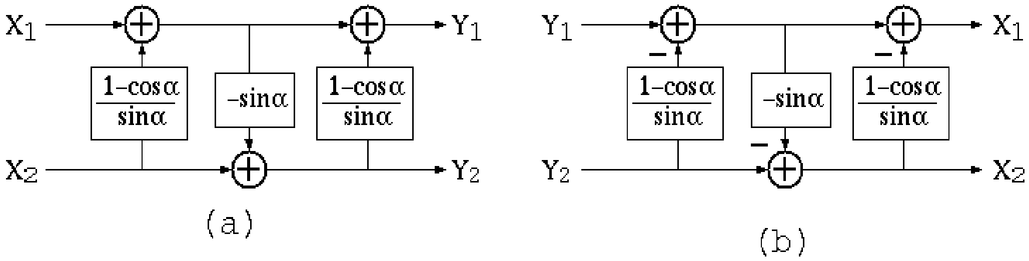
In Figure 4.6 y_0, y_1, etc. are the prediction residuals or pixel intensity values and Y_0, Y_1, etc. are the transform coefficients. The number of additions and shifts required for the implementation are shown in Table 4.3.

Table 4.3 Number of operations [16].

	Number of shifts	Number of additions	Total number of operations
Modified Order 16 ICT	32	150	182

4.5 2-D order 16 binDCT based on Loeffler's factorization

The DCT-II can be factorized into planar rotations and butterflies (Figure 4.9) [58, 59]. The factorization of order 16 DCT-II as proposed by Loeffler et al. [63] (Figure 4.8) is more efficient than that proposed by Chen et al [58], in terms of the number of operations required for implementation. This representation requires 31 shifts and 81 additions [63]. In order to reduce the number of shifts, planar rotations can be represented in terms of three shears, or lifting steps [60, 61] (Figures 4.7). Thus a rotation, which notionally needs 4 shifts and 2 additions, can be implemented by 3 shifts and 3 additions.



X_1
Y_1
$\frac{1-\cos\alpha}{\sin\alpha}$
$-\sin\alpha$
$\frac{1-\cos\alpha}{\sin\alpha}$
X_2
Y_2
(a)
Y_1
X_1
−
$\frac{1-\cos\alpha}{\sin\alpha}$
$-\sin\alpha$
−
$\frac{1-\cos\alpha}{\sin\alpha}$
−
Y_2
X_2
(b)

Figure 4.7 (a) Representation of a plane rotation by 3 lifting steps. (b) Inverse transform [60, 61].

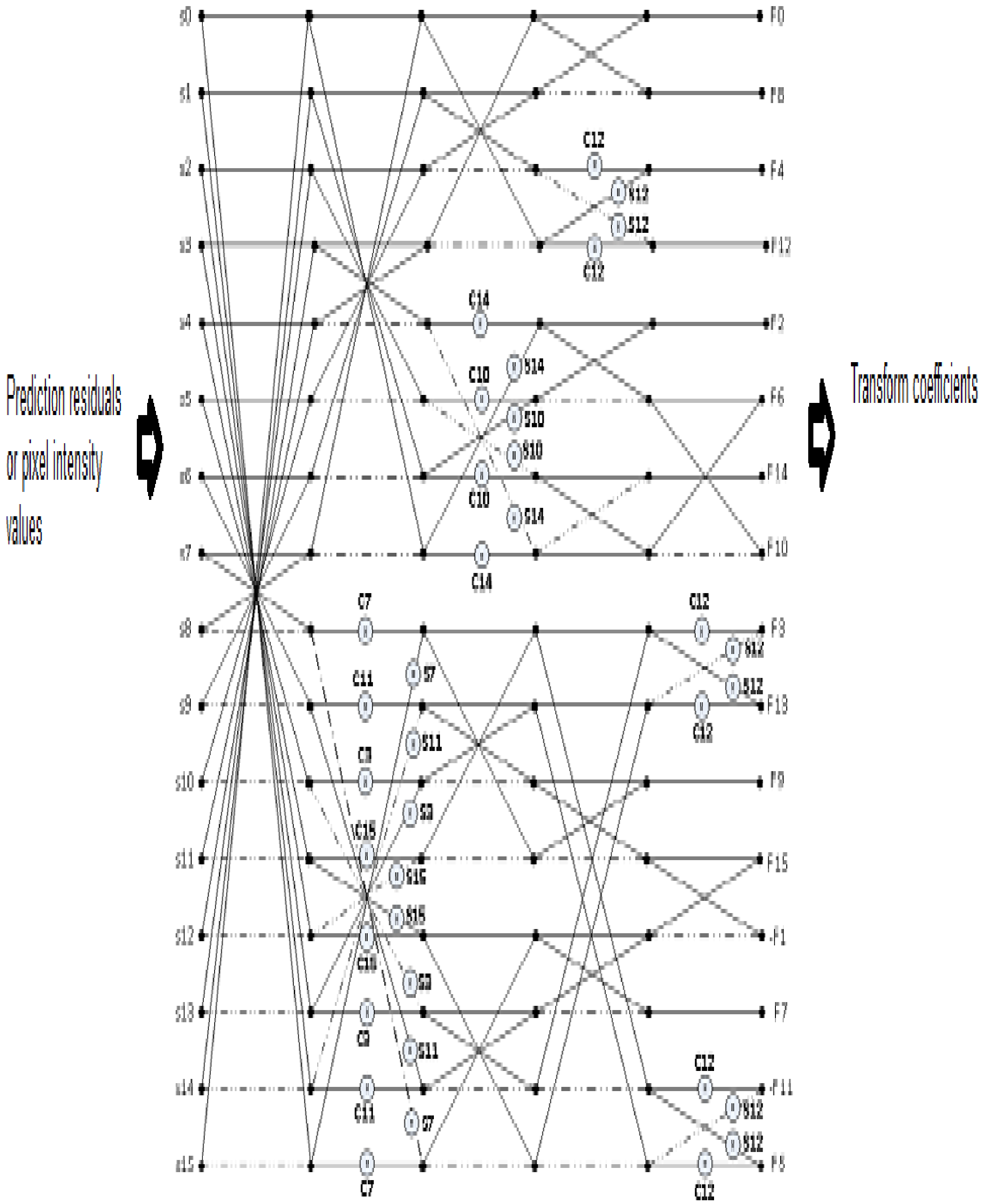


Figure 4.8 Flow graph for order 16 DCT-II proposed by Loeffler et al. [58] (Cn =$\cos\frac{n\pi}{32}$, Sn =$\sin\frac{n\pi}{32}$).

The relations between X_1, X_2 and Y_1, Y_2 (Figures 4.7 (a) and (b)) are derived in equations 4.7 to 4.10. In many cases this can be further reduced (Figure 4.8) to 2 shifts and 2 additions. In Figure 4.9 (a) blocks p and u represent the shifts and K_1 and K_2 represent the scaling factors. The reduction is achieved by moving scaling factors to the quantization stage. The shears still contain parameters that are generally irrational but can be approximated by dyadic-rational coefficients as proposed by Liang and Tran [62]. The tradeoff between coding efficiency and computational complexity is achieved by tuning the approximations.

$$Y_1 = X_1 + 2aX_2 + a^2\, bX_2 + abX_1 \qquad (4.7)$$

$$Y_2 = X_2 + b(aX_2 + X_1) \qquad (4.8)$$

where $a = \frac{1-cos\propto}{sin\propto}$ and $b = sin \propto$

$$X_1 = Y_1 + 2aY_2 + a^2\, bY_2 + abY_1 \qquad (4.9)$$

$$X_2 = Y_2 + b(aY_2 + Y_1) \qquad (4.10)$$

where $a = -\frac{1-cos\propto}{sin\propto}$ and $b = -sin \propto$

The binDCT [62] based on Loeffler et al [63] factorization (binDCT-L) as proposed by Liang and Tran can implement the order 16 DCT-II with an optimum number of shifts and multiples. Although the binDCT is not exactly an integer cosine transform, it uses only fixed point arithmetic operations for computing. The flow diagram for binDCT is shown in Figure 4.10. The dotted blocks represent the scaling factors.

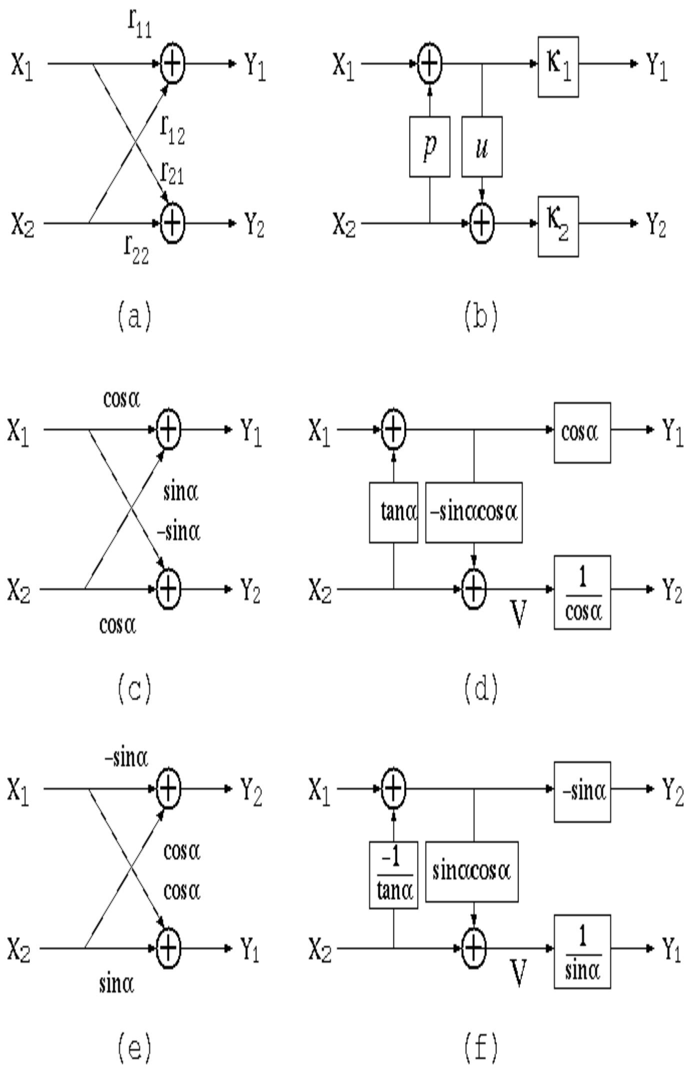


Figure 4.9 (a) General rotation; (b) Scaled lifting structure for (a); (c) Orthogonal plane rotation; (d) Scaled lifting structure for (c); (e) Permuted version of (c); (f) Scaled lifting structure for (e) [62, 64, 65].

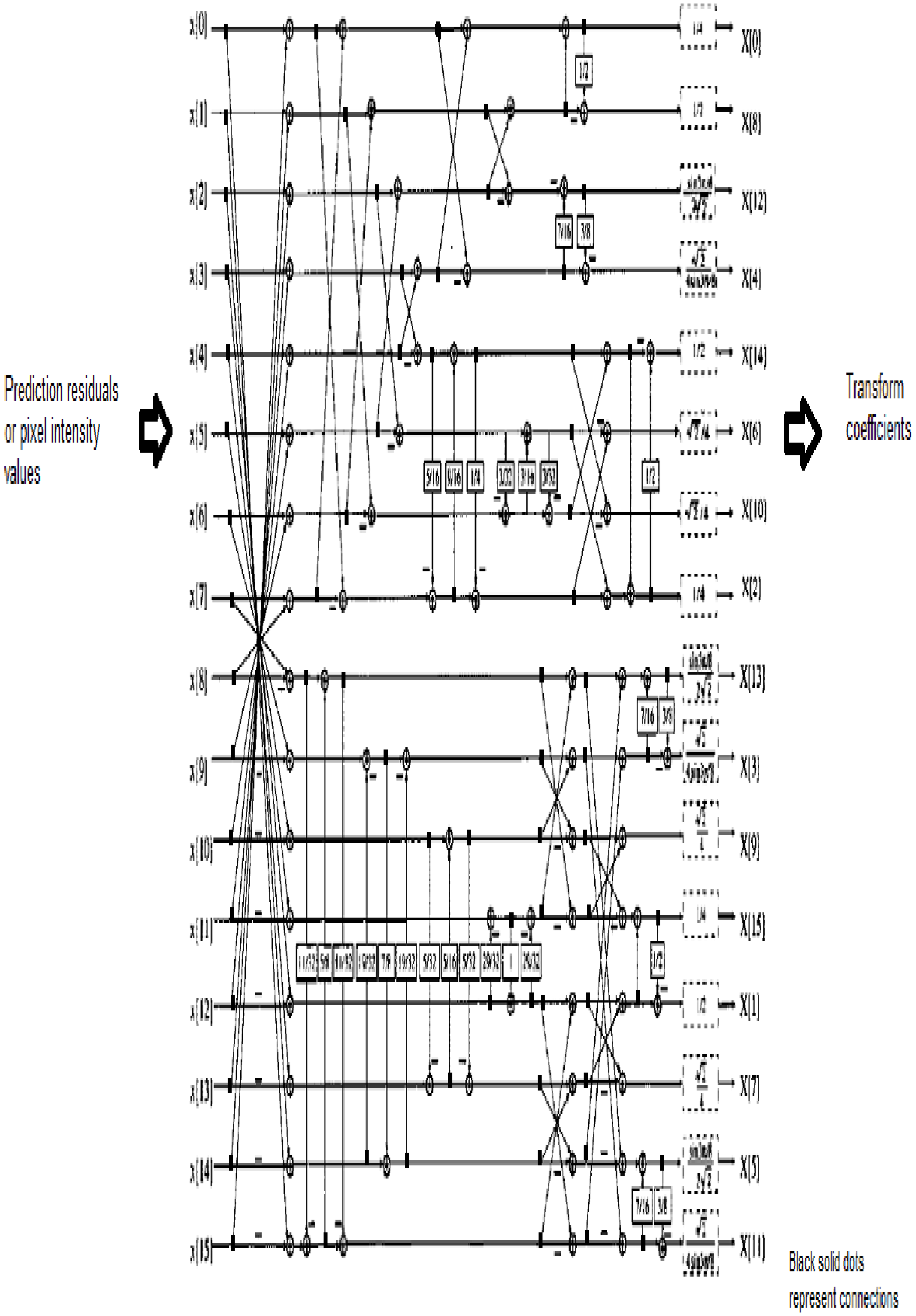


Figure 4.10 Order 16 binDCT-L [62].

The number of shifts and multiplies for order 16 binDCT is given in Table 4.4.

Table 4.4 Number of operations [62].

Order 16 binDCT-L	Number of shifts	Number of additions	Total number of operations
	51	106	157

In the next chapter implementation details of the above order 16 ICTs in H.264/AVC and AVS-video and comparisons on the basis of transform coding gain and objective quality (BD-PSNR and BD-Bitrate [66] observed on video sequences of various resolutions will be discussed.

CHAPTER 5

PERFORMANCE ANALYSIS AND CONCLUSIONS

5.1 Introduction

Transform coding gain [67] G_{TC} (5.1) is a good measure of the energy compaction efficiency of a transform.

$$G_{TC} = \frac{\frac{1}{N}\sum_{i=1}^{N} \sigma^2{}_{y_{ii}}}{\left(\prod_{i=1}^{N} \sigma^2{}_{y_{ii}}\right)^{1/N}} \tag{5.1}$$

where $\sigma^2{}_{y_{ii}}$ is the covariance of the elements of the source. The numerator represents the arithmetic mean of variances of transform coefficients. The denominator represents the corresponding geometric mean. To compare the transform coding gain of three ICTs explained in the previous chapter a 1-D, zero mean, unit variance first order Markov process (Figure 5.1) is considered as the source. The adjacent element correlation of the source can be denoted as ρ. From Table 4.1 we can see that HD video sequences have ρ varying from 0.5 to 0.9. Figure 5.1 and Table 5.1 compares the variation of transform coding gain (in dB) with ρ for all the three ICTs with respect to order 16 DCT-II. Here simple order 16 ICT (SICT) 1 and SICT 2 represent the ICT (explained in section 4.3 of Chapter 4) implemented in H.264/AVC and AVS-video respectively. The modified order 16 ICT (MICT) 1 and MICT 2 represent the ICT (explained in section 4.3 of Chapter 4) implemented in H.264/AVC and AVS-video respectively. From the graph we can come to the conclusion that order 16 binDCT-L gives the best transform coding gain for higher values of ρ (9.4499 dB for ρ = 0.95) when compared to order 16 DCT-II (9.4554 dB for ρ = 0.95. The modified order 16 ICT 1 and MICT 2 give 8.8925 dB and 8.8201 dB respectively for ρ = 0.95. The corresponding values for SICT 1 and SICT 2 are 8.7693 dB and 8.7416 dB respectively.

$$T = \begin{bmatrix} 1 & \rho & \cdots & \rho^{14} & \rho^{15} \\ \rho & 1 & \cdots & \rho^{13} & \rho^{14} \\ \vdots & & \ddots & & \vdots \\ \rho^{14} & \rho^{13} & \cdots & 1 & \rho \\ \rho^{15} & \rho^{14} & \cdots & \rho^{15} & 1 \end{bmatrix}$$

Figure 5.1 Matrix representation of first order Markov source of size 16 with correlation ρ.

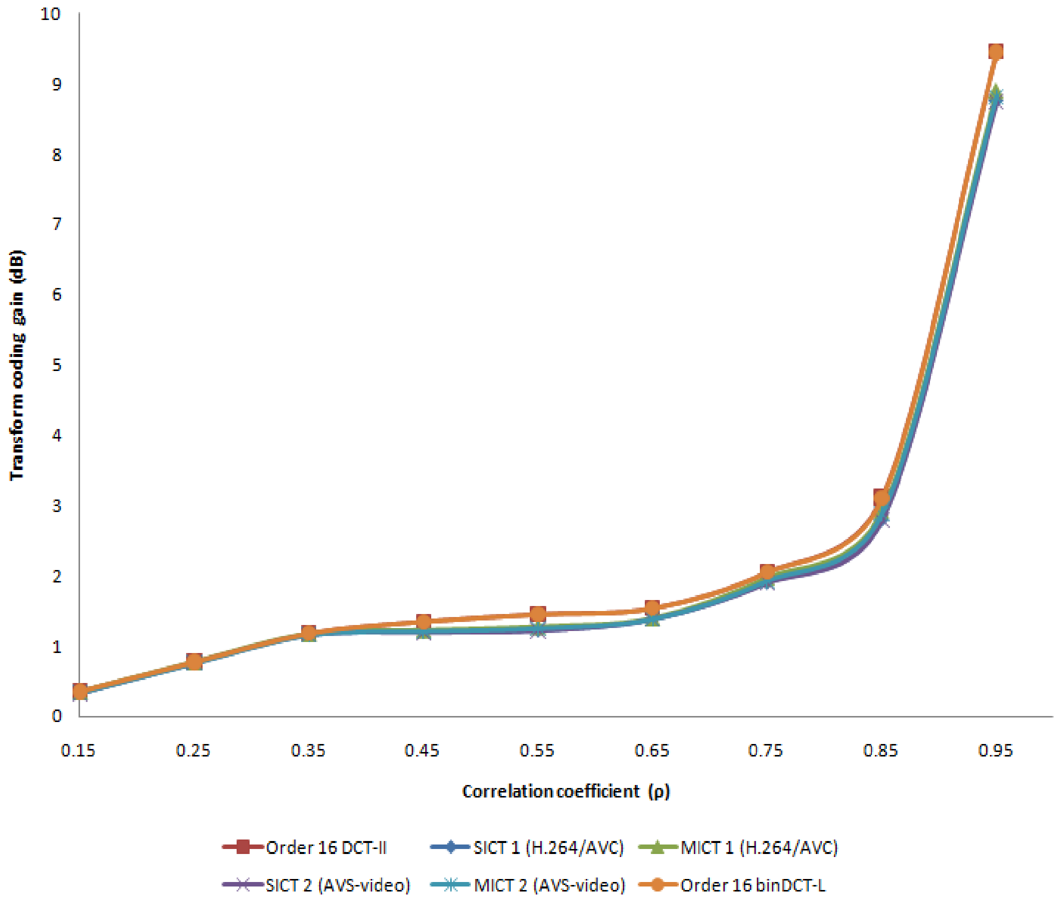
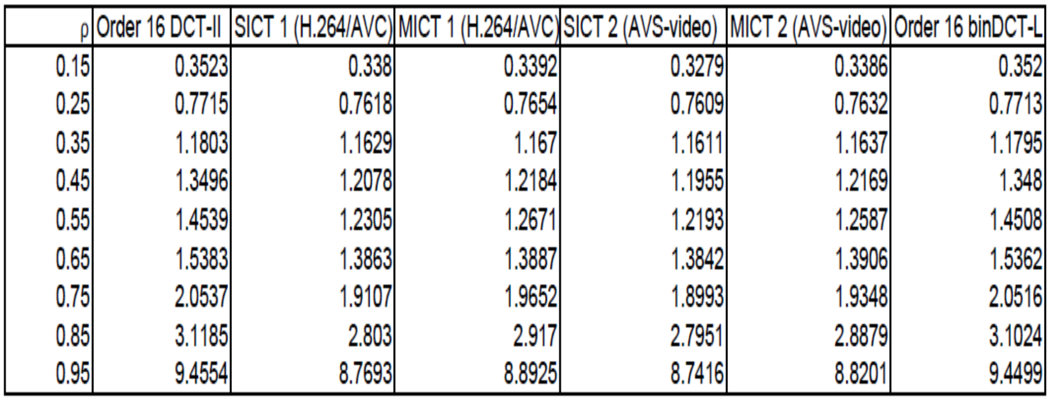


Figure 5.2 Comparison of transform coding gains of various order 16 ICTs with respect to order 16 DCT-II

Table 5.1 Variation of transform coding gain with ρ



ρ	Order 16 DCT-II	SICT 1 (H.264/AVC)	MICT 1 (H.264/AVC)	SICT 2 (AVS-video)	MICT 2 (AVS-video)	Order 16 binDCT-L
0.15	0.3523	0.338	0.3392	0.3279	0.3386	0.352
0.25	0.7715	0.7618	0.7654	0.7609	0.7632	0.7713
0.35	1.1803	1.1629	1.167	1.1611	1.1637	1.1795
0.45	1.3496	1.2078	1.2184	1.1955	1.2169	1.348
0.55	1.4539	1.2305	1.2671	1.2193	1.2587	1.4508
0.65	1.5383	1.3863	1.3887	1.3842	1.3906	1.5362
0.75	2.0537	1.9107	1.9652	1.8993	1.9348	2.0516
0.85	3.1185	2.803	2.917	2.7951	2.8879	3.1024
0.95	9.4554	8.7693	8.8925	8.7416	8.8201	9.4499

The performance of the ICTs can be evaluated on the basis of BD-bitrate and BD-PSNR [66] achieved when the standards (H.264/AVC and AVS-video) are integrated with the ICTs. The values are calculated with respect to the default performance achieved by the standards. Here BD-PSNR gives the absolute PSNR gain at the same bitrate and BD-bitrate gives the percentage bitrate savings at the same PSNR.

5.2 Implementation in H.264/AVC and performance analysis

The three 2-D order 16 ICTs are implemented into JM 17.2 reference software [68] for H.264/AVC. The scaling and quantization matrices used for simple and modified 2-D order 16 ICTs are the same as that in the reference software as they are extended versions of the order 8 ICT, while that used for 2-D order 16 binDCT-L is specified in the Appendix. The selection of transform size is based on MB level R-D cost [71]. The H.264 high profile is used for encoding the video sequences. The video sequences can be downloaded from [69]. Table 5.2 specifies some of the configuration parameters used for encoding. The encoding is performed with a system having Intel i7 Quad 4, 2.60 GHz processor supporting 6GB RAM. The operating system used is Windows 7. Video sequences belonging to HD (1920 x 1080, 1280 x 720), WVGA (832 x 480) and WQVGA (416 x 240) are used for evaluating the performance. The results (bitrates verses PSNRs, percentage bit rate savings (BD-bitrate) verses BD-PSNR for each sequence) are tabulated. The graphs for one sequence from each of the different resolutions are also shown.

Table 5.2 Configuration Parameters.

Group of pictures (GOP) size	8
GOP structure	IBBBBBBP
Intra frame period	0.5 s
R-D optimization	on
QP	22, 27, 32, 37
Reference frames	2
Fast motion estimation	on
Search range	± 32
Deblocking filter	on
Entropy coding	CABAC

5.2.1 Performance of simple 2-D order 16 ICT (SICT)

Results for HD (1920 x 1080) sequences:

Table 5.3 Comparison of bitrates and PSNRs for three 1920 x 1080 sequences (H.264/AVC with SICT).

		H.264/AVC with Simple 2-D ICT				Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Kimono	22	11129.712	37.631	43.06	45.14	11134.11	37.60	43.00	45.16
	27	9436.688	35.326	40.6766	42.6057	9440.6723	35.319	40.6692	42.4108
	32	5621.776	33.4748	39.0813	41.1953	5632.26	33.456	39.0561	41.1871
	37	3502.688	32.6496	38.3154	40.7401	3507.433	32.6123	38.2045	40.5126
ParkScene	22	4360.639	37.44	42.99	43.34	4362.4896	37.26	42.88	43.33
	27	2820.96	35.3146	40.7148	42.6534	2835.451	35.3071	40.7042	42.6567
	32	1513.9264	33.482	39.1326	41.2028	1518.362	33.479	39.1252	41.2028
	37	928.6464	32.6537	38.5015	40.7291	933.2678	32.512	38.4948	40.7265
Cactus	22	7377.94	37.7416	41.9747	42.0858	7379.276	37.6983	41.9654	42.0832
	27	5818.04	35.6108	38.0594	40.0161	5825.11	35.6091	38.0569	40.0139
	32	3820.96	34.2778	37.3159	38.8342	3821.732	34.2755	37.3137	38.8297
	37	2669.344	32.8594	36.5494	37.6315	2674.938	32.8548	36.5451	37.6284

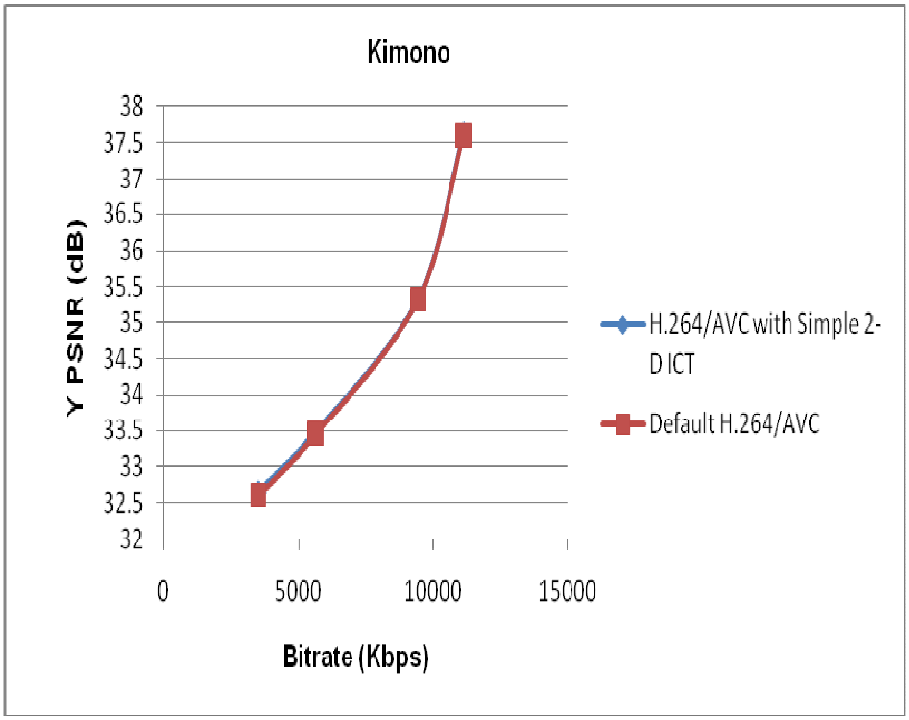


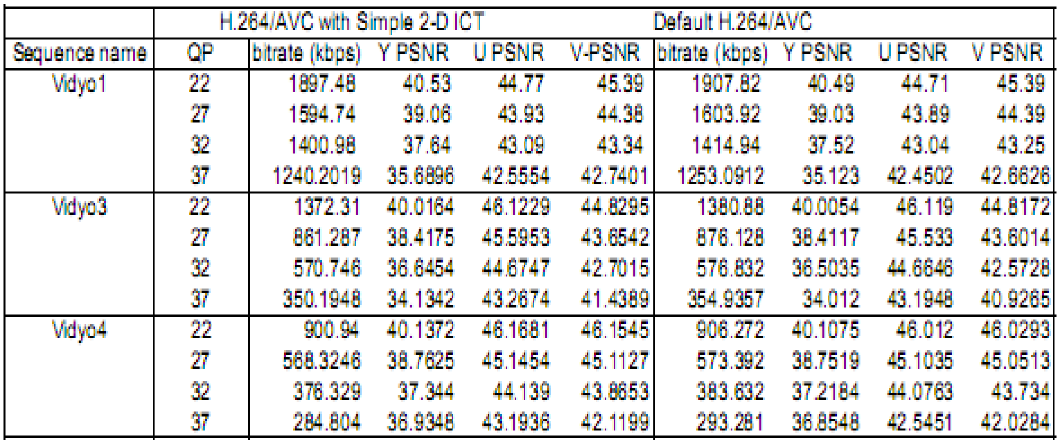
Figure 5.3 Y PSNR variations with bitrate for Kimono sequence (H.264/AVC with SICT).

Table 5.4 BD-bitrate and BD-PSNR (H.264/AVC with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Kimono	-4.32	0.27
ParkScene	-5.91	0.34
Cactus	-3.59	0.22

Results for HD (1280 x 720) sequences:

Table 5.5 Comparison of bitrates and PSNRs for three 1280 x 720 sequences (H.264/AVC with SICT).



	H.264/AVC with Simple 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Vidyo1	22	1897.48	40.53	44.77	45.39	1907.82	40.49	44.71	45.39
	27	1594.74	39.06	43.93	44.38	1603.92	39.03	43.89	44.39
	32	1400.98	37.64	43.09	43.34	1414.94	37.52	43.04	43.25
	37	1240.2019	35.6896	42.5554	42.7401	1253.0912	35.123	42.4502	42.6626
Vidyo3	22	1372.31	40.0164	46.1229	44.8295	1380.88	40.0054	46.119	44.8172
	27	861.287	38.4175	45.5953	43.6542	876.128	38.4117	45.533	43.6014
	32	570.746	36.6454	44.6747	42.7015	576.832	36.5035	44.6646	42.5728
	37	350.1948	34.1342	43.2674	41.4389	354.9357	34.012	43.1948	40.9265
Vidyo4	22	900.94	40.1372	46.1681	46.1545	906.272	40.1075	46.012	46.0293
	27	568.3246	38.7625	45.1454	45.1127	573.392	38.7519	45.1035	45.0513
	32	376.329	37.344	44.139	43.8653	383.632	37.2184	44.0763	43.734
	37	284.804	36.9348	43.1936	42.1199	293.281	36.8548	42.5451	42.0284

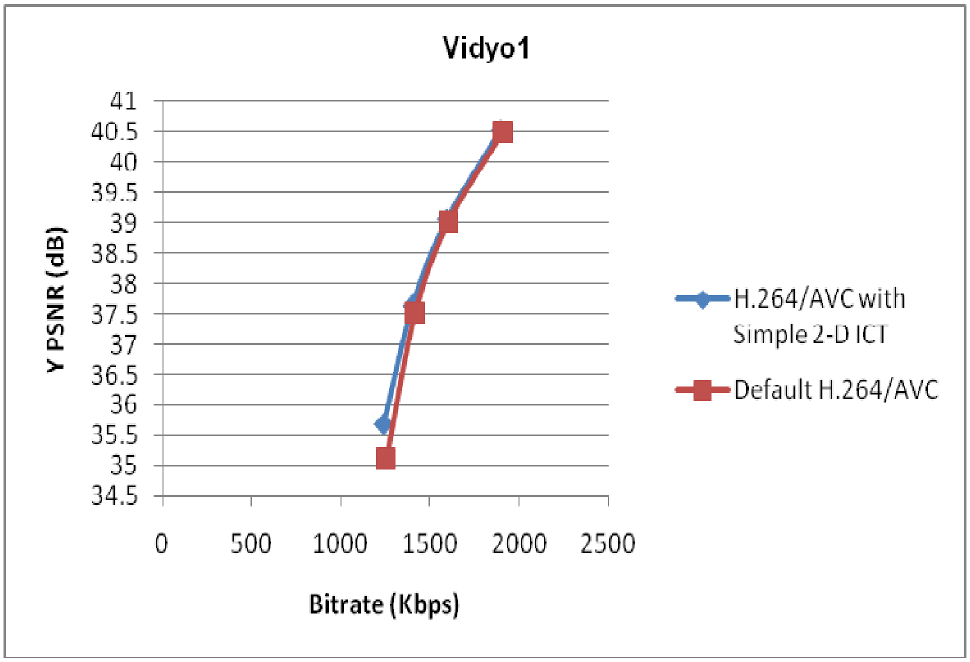


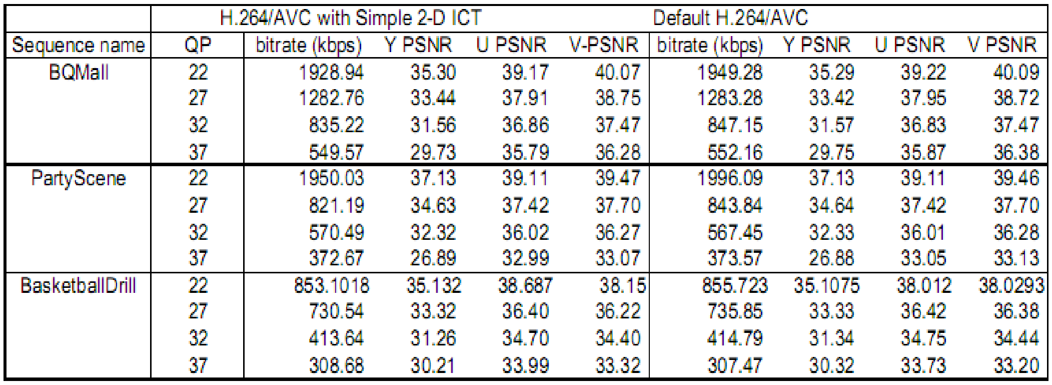
Figure 5.4 Y PSNR variations with bitrate for Vidyo1 sequence (H.264/AVC with SICT).

Table 5.6 BD-bitrate and BD-PSNR (H.264/AVC with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Vidyo1	-2.57	0.19
Vidyo2	-3.81	0.22
Vidyo3	-1.91	0.12

Results for WVGA (832 x 480) sequences:

Table 5.7 Comparison of bitrates and PSNRs for three 832 x 480 sequences (H.264/AVC with SICT).



	H.264/AVC with Simple 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQMall	22	1928.94	35.30	39.17	40.07	1949.28	35.29	39.22	40.09
	27	1282.76	33.44	37.91	38.75	1283.28	33.42	37.95	38.72
	32	835.22	31.56	36.86	37.47	847.15	31.57	36.83	37.47
	37	549.57	29.73	35.79	36.28	552.16	29.75	35.87	36.38
PartyScene	22	1950.03	37.13	39.11	39.47	1996.09	37.13	39.11	39.46
	27	821.19	34.63	37.42	37.70	843.84	34.64	37.42	37.70
	32	570.49	32.32	36.02	36.27	567.45	32.33	36.01	36.28
	37	372.67	26.89	32.99	33.07	373.57	26.88	33.05	33.13
BasketballDrill	22	853.1018	35.132	38.687	38.15	855.723	35.1075	38.012	38.0293
	27	730.54	33.32	36.40	36.22	735.85	33.33	36.42	36.38
	32	413.64	31.26	34.70	34.40	414.79	31.34	34.75	34.44
	37	308.68	30.21	33.99	33.32	307.47	30.32	33.73	33.20

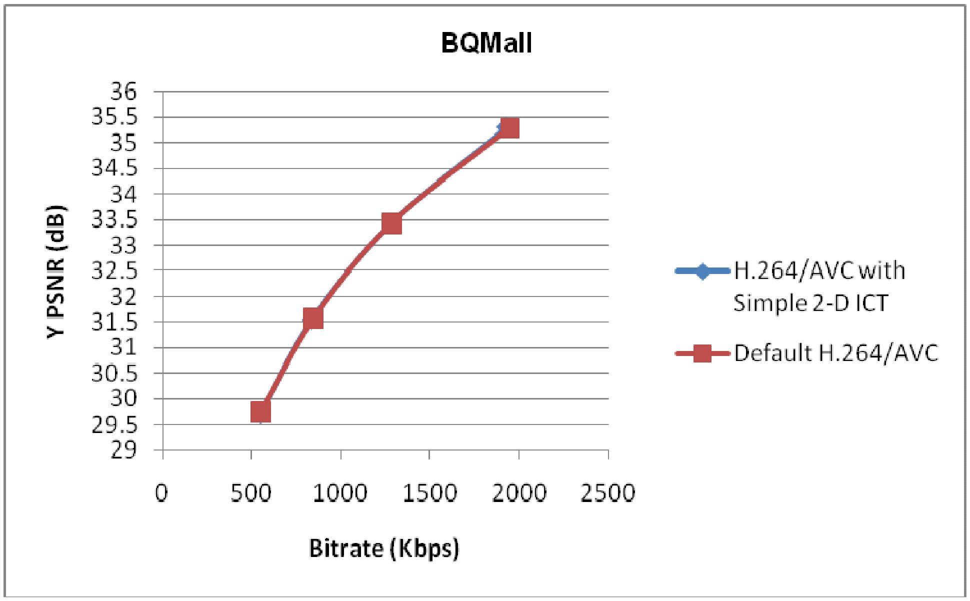


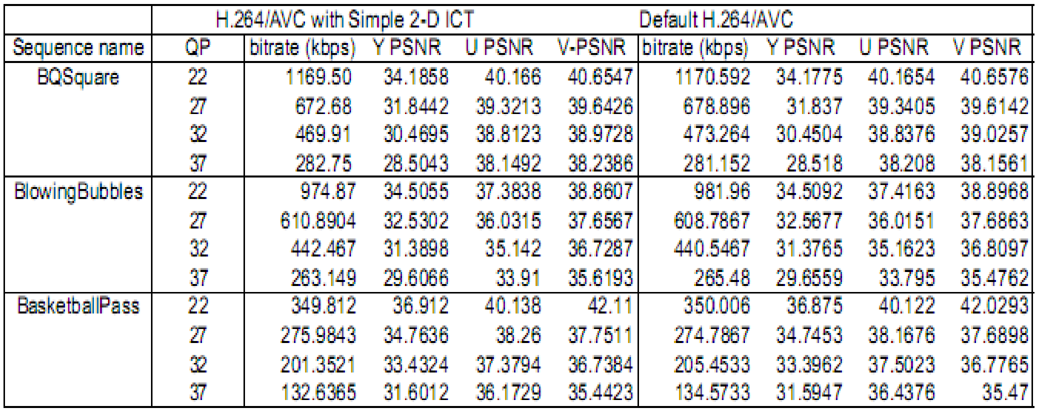
Figure 5.5 Y PSNR variations with bitrate for BQMall sequence (H.264/AVC with SICT).

Table 5.8 BD-bitrate and BD-PSNR (H.264/AVC with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQMall	-1.35	0.08
PartyScene	1.05	-0.04
BasketballDrill	1.02	-0.01

Results for WQVGA (416 x 240) sequences:

Table 5.9 Comparison of bitrates and PSNRs for three 416 x 240 sequences (H.264/AVC with SICT).



	H.264/AVC with Simple 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQSquare	22	1169.50	34.1858	40.166	40.6547	1170.592	34.1775	40.1654	40.6576
	27	672.68	31.8442	39.3213	39.6426	678.896	31.837	39.3405	39.6142
	32	469.91	30.4695	38.8123	38.9728	473.264	30.4504	38.8376	39.0257
	37	282.75	28.5043	38.1492	38.2386	281.152	28.518	38.208	38.1561
BlowingBubbles	22	974.87	34.5055	37.3838	38.8607	981.96	34.5092	37.4163	38.8968
	27	610.8904	32.5302	36.0315	37.6567	608.7867	32.5677	36.0151	37.6863
	32	442.467	31.3898	35.142	36.7287	440.5467	31.3765	35.1623	36.8097
	37	263.149	29.6066	33.91	35.6193	265.48	29.6559	33.795	35.4762
BasketballPass	22	349.812	36.912	40.138	42.11	350.006	36.875	40.122	42.0293
	27	275.9843	34.7636	38.26	37.7511	274.7867	34.7453	38.1676	37.6898
	32	201.3521	33.4324	37.3794	36.7384	205.4533	33.3962	37.5023	36.7765
	37	132.6365	31.6012	36.1729	35.4423	134.5733	31.5947	36.4376	35.47

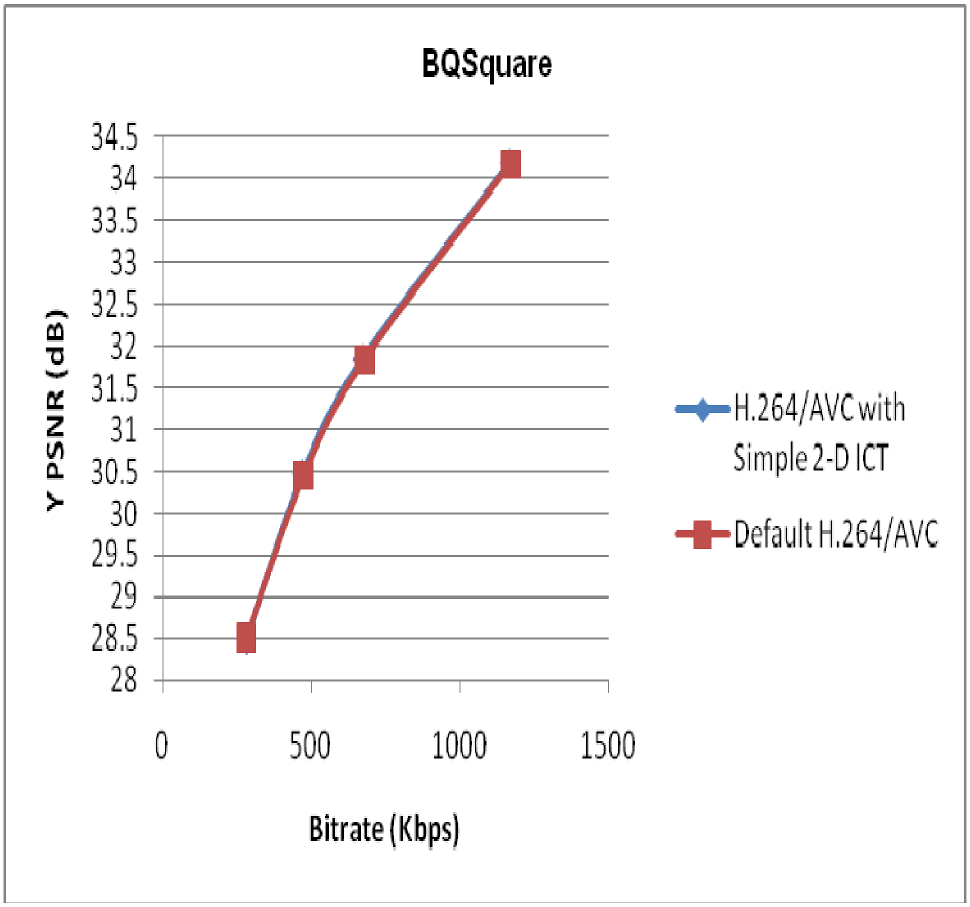


Figure 5.6 Y PSNR variations with bitrate for BQSquare sequence (H.264/AVC with SICT).

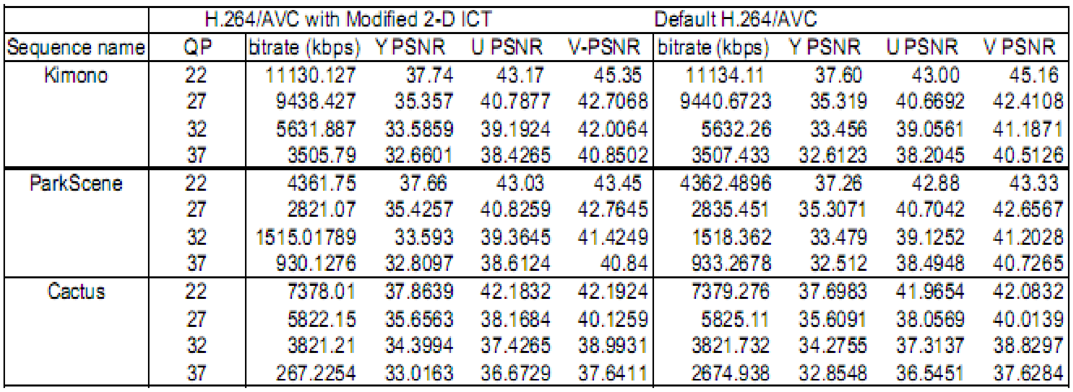
Table 5.10 BD-bitrate and BD-PSNR (H.264/AVC with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQSquare	1.16	-0.04
BlowingBubbles	1.34	-0.11
BasketballPass	-1.1	0.17

5.2.2 Performance of Modified 2-D order 16 ICT (MICT)

Results for HD (1920 x 1080) sequences:

Table 5.11 Comparison of bitrates and PSNRs for three 1920 x 1080 sequences (H.264/AVC with MICT).



	H.264/AVC with Modified 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Kimono	22	11130.127	37.74	43.17	45.35	11134.11	37.60	43.00	45.16
	27	9438.427	35.357	40.7877	42.7068	9440.6723	35.319	40.6692	42.4108
	32	5631.887	33.5859	39.1924	42.0064	5632.26	33.456	39.0561	41.1871
	37	3505.79	32.6601	38.4265	40.8502	3507.433	32.6123	38.2045	40.5126
ParkScene	22	4361.75	37.66	43.03	43.45	4362.4896	37.26	42.88	43.33
	27	2821.07	35.4257	40.8259	42.7645	2835.451	35.3071	40.7042	42.6567
	32	1515.01789	33.593	39.3645	41.4249	1518.362	33.479	39.1252	41.2028
	37	930.1276	32.8097	38.6124	40.84	933.2678	32.512	38.4948	40.7265
Cactus	22	7378.01	37.8639	42.1832	42.1924	7379.276	37.6983	41.9654	42.0832
	27	5822.15	35.6563	38.1684	40.1259	5825.11	35.6091	38.0569	40.0139
	32	3821.21	34.3994	37.4265	38.9931	3821.732	34.2755	37.3137	38.8297
	37	267.2254	33.0163	36.6729	37.6411	2674.938	32.8548	36.5451	37.6284

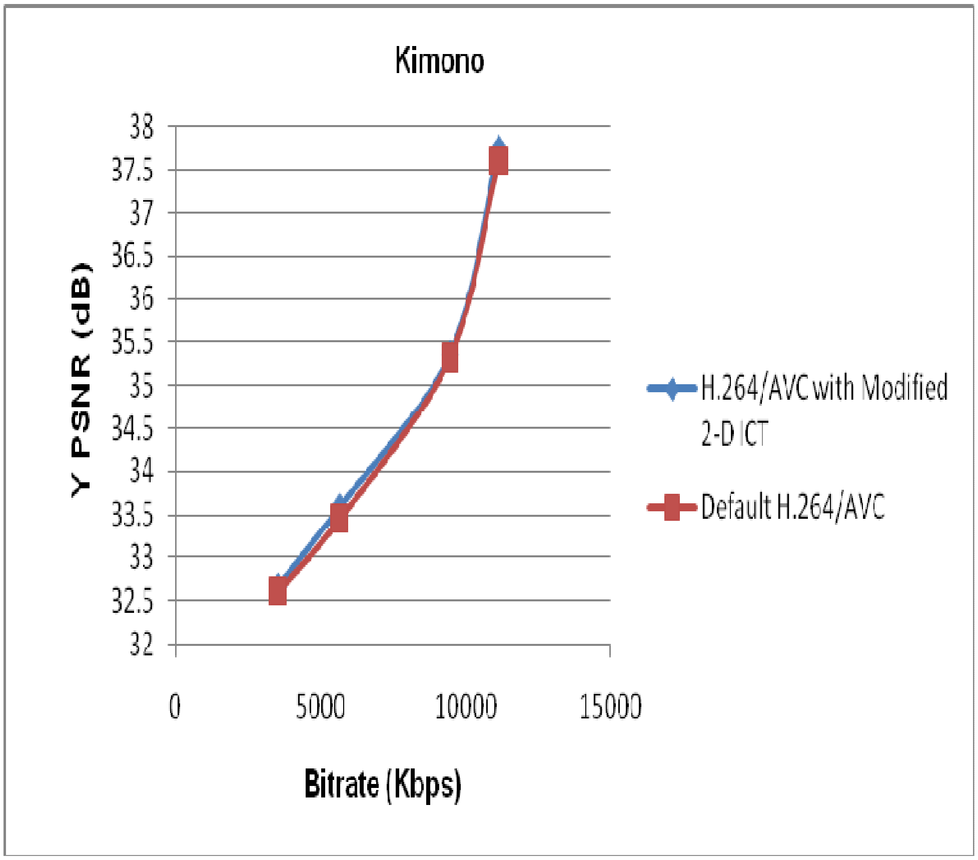


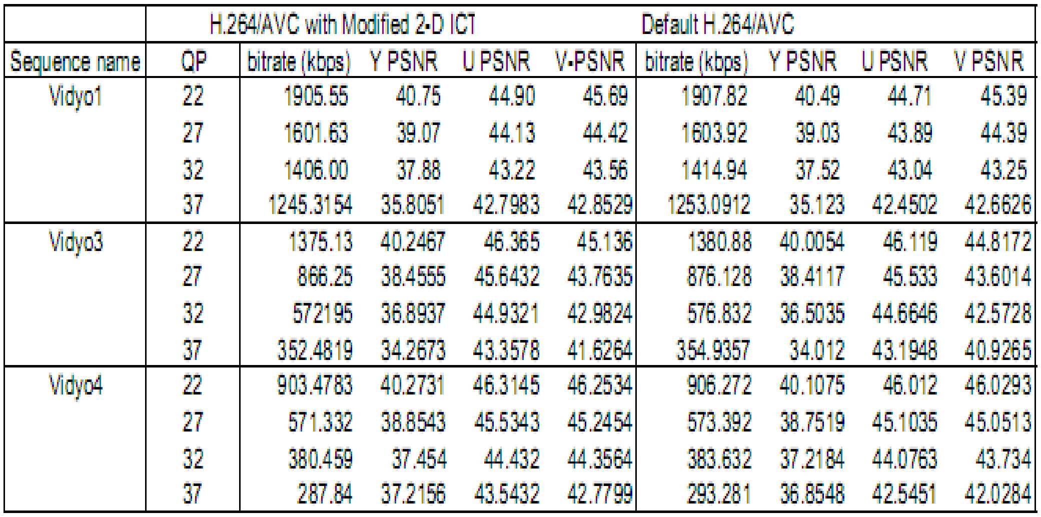
Figure 5.7 Y PSNR variations with bitrate for Kimono sequence (H.264/AVC with MICT).

Table 5.12 BD-bitrate and BD-PSNR (H.264/AVC with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Kimono	-6.5	0.34
ParkScene	-8.81	0.38
Cactus	-3.02	0.25

Results for HD (1280 x 720) sequences:

Table 5.13 Comparison of bitrates and PSNRs for three 1280 x 720 sequences (H.264/AVC with MICT).



	H.264/AVC with Modified 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Vidyo1	22	1905.55	40.75	44.90	45.69	1907.82	40.49	44.71	45.39
	27	1601.63	39.07	44.13	44.42	1603.92	39.03	43.89	44.39
	32	1406.00	37.88	43.22	43.56	1414.94	37.52	43.04	43.25
	37	1245.3154	35.8051	42.7983	42.8529	1253.0912	35.123	42.4502	42.6626
Vidyo3	22	1375.13	40.2467	46.365	45.136	1380.88	40.0054	46.119	44.8172
	27	866.25	38.4555	45.6432	43.7635	876.128	38.4117	45.533	43.6014
	32	572195	36.8937	44.9321	42.9824	576.832	36.5035	44.6646	42.5728
	37	352.4819	34.2673	43.3578	41.6264	354.9357	34.012	43.1948	40.9265
Vidyo4	22	903.4783	40.2731	46.3145	46.2534	906.272	40.1075	46.012	46.0293
	27	571.332	38.8543	45.5343	45.2454	573.392	38.7519	45.1035	45.0513
	32	380.459	37.454	44.432	44.3564	383.632	37.2184	44.0763	43.734
	37	287.84	37.2156	43.5432	42.7799	293.281	36.8548	42.5451	42.0284

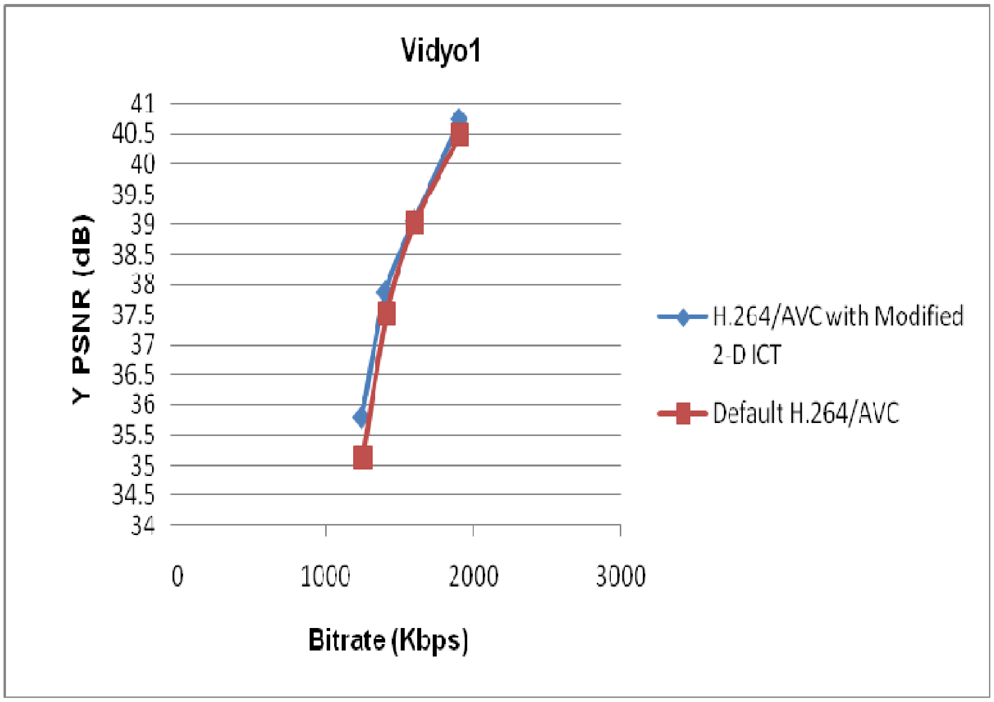


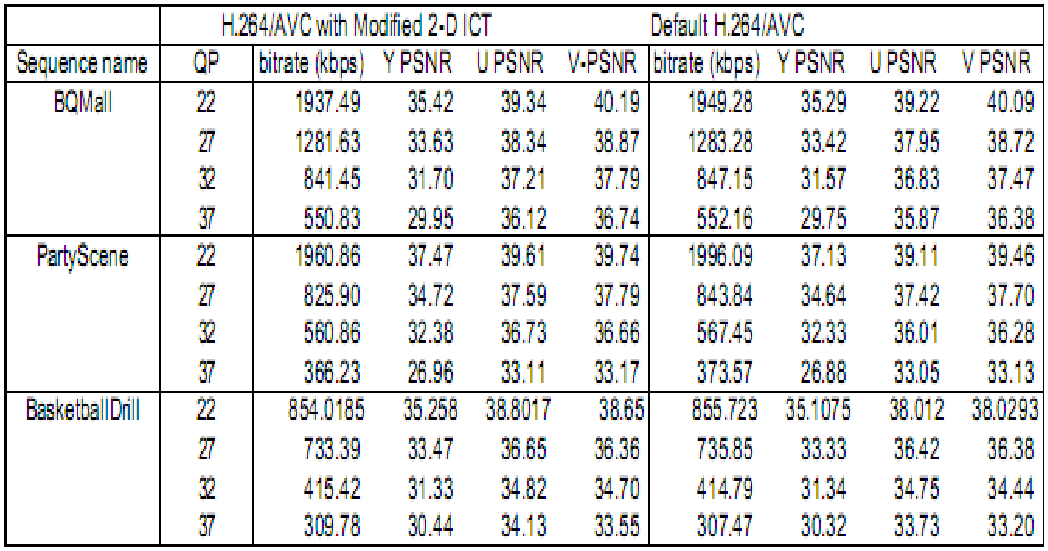
Figure 5.8 Y PSNR variations with bitrate for Vidyo1 sequence (H.264/AVC with MICT).

Table 5.14 BD-bitrate and BD-PSNR (H.264/AVC with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Vidyo1	-5.30	0.31
Vidyo2	-3.55	0.26
Vidyo3	-2.08	0.20

Results for WVGA (832 x 480) sequences:

Table 5.15 Comparison of bitrates and PSNRs for three 832 x 480 sequences (H.264/AVC with MICT).



	H.264/AVC with Modified 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQMall	22	1937.49	35.42	39.34	40.19	1949.28	35.29	39.22	40.09
	27	1281.63	33.63	38.34	38.87	1283.28	33.42	37.95	38.72
	32	841.45	31.70	37.21	37.79	847.15	31.57	36.83	37.47
	37	550.83	29.95	36.12	36.74	552.16	29.75	35.87	36.38
PartyScene	22	1960.86	37.47	39.61	39.74	1996.09	37.13	39.11	39.46
	27	825.90	34.72	37.59	37.79	843.84	34.64	37.42	37.70
	32	560.86	32.38	36.73	36.66	567.45	32.33	36.01	36.28
	37	366.23	26.96	33.11	33.17	373.57	26.88	33.05	33.13
BasketballDrill	22	854.0185	35.258	38.8017	38.65	855.723	35.1075	38.012	38.0293
	27	733.39	33.47	36.65	36.36	735.85	33.33	36.42	36.38
	32	415.42	31.33	34.82	34.70	414.79	31.34	34.75	34.44
	37	309.78	30.44	34.13	33.55	307.47	30.32	33.73	33.20

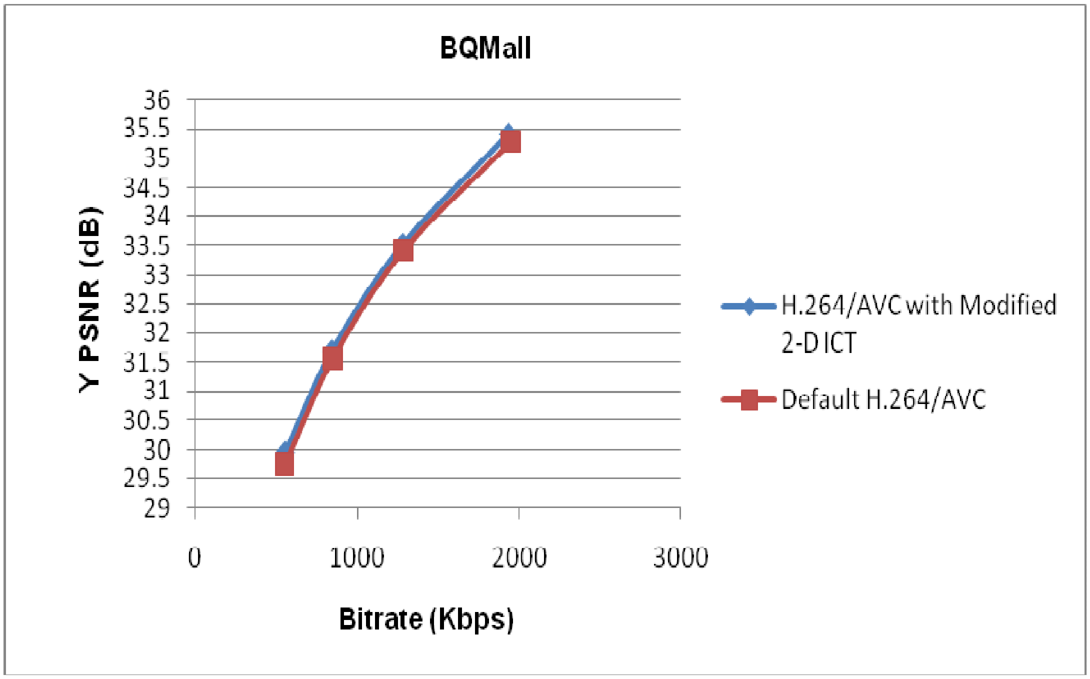


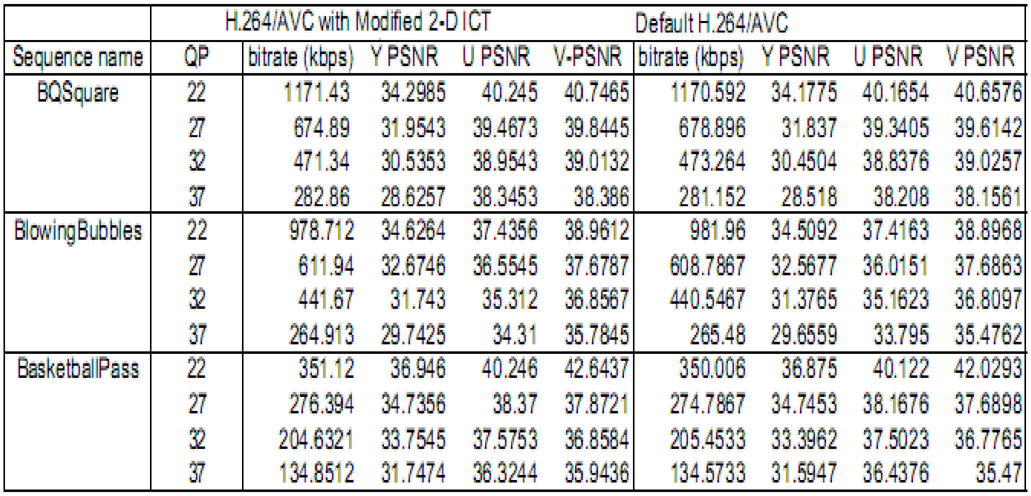
Figure 5.9 Y PSNR variations with bitrate for BQMall sequence (H.264/AVC with MICT).

Table 5.16 BD-bitrate and BD-PSNR (H.264/AVC with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQMall	-2.83	0.18
PartyScene	-1.39	0.11
BasketballDrill	-0.61	0.07

Results for WQVGA (416 x 240) sequences:

Table 5.17 Comparison of bitrates and PSNRs for three 416 x 240 sequences (H.264/AVC with MICT).



	H.264/AVC with Modified 2-D ICT					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQSquare	22	1171.43	34.2985	40.245	40.7465	1170.592	34.1775	40.1654	40.6576
	27	674.89	31.9543	39.4673	39.8445	678.896	31.837	39.3405	39.6142
	32	471.34	30.5353	38.9543	39.0132	473.264	30.4504	38.8376	39.0257
	37	282.86	28.6257	38.3453	38.386	281.152	28.518	38.208	38.1561
BlowingBubbles	22	978.712	34.6264	37.4356	38.9612	981.96	34.5092	37.4163	38.8968
	27	611.94	32.6746	36.5545	37.6787	608.7867	32.5677	36.0151	37.6863
	32	441.67	31.743	35.312	36.8567	440.5467	31.3765	35.1623	36.8097
	37	264.913	29.7425	34.31	35.7845	265.48	29.6559	33.795	35.4762
BasketballPass	22	351.12	36.946	40.246	42.6437	350.006	36.875	40.122	42.0293
	27	276.394	34.7356	38.37	37.8721	274.7867	34.7453	38.1676	37.6898
	32	204.6321	33.7545	37.5753	36.8584	205.4533	33.3962	37.5023	36.7765
	37	134.8512	31.7474	36.3244	35.9436	134.5733	31.5947	36.4376	35.47

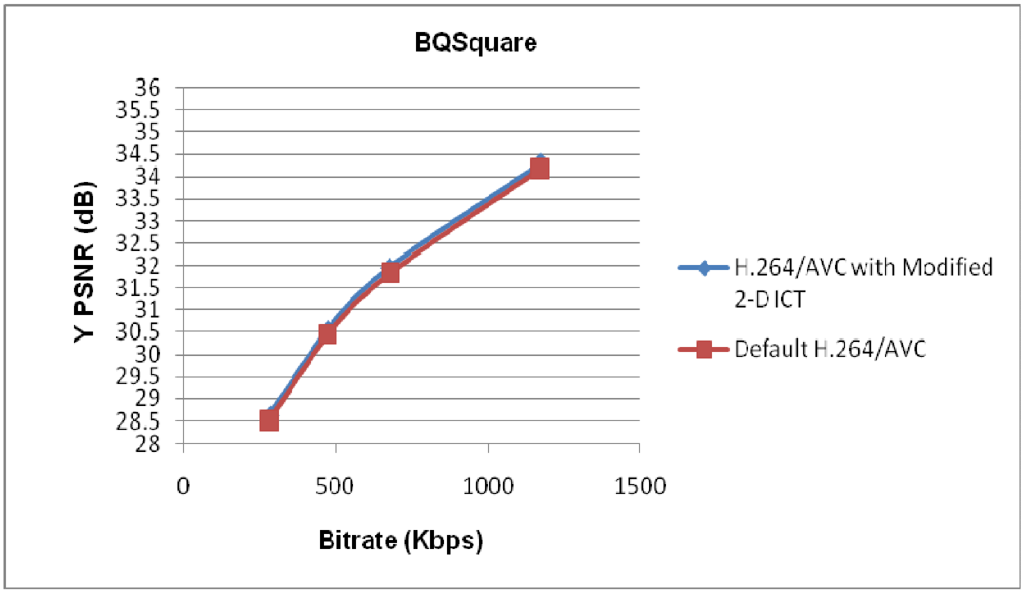


Figure 5.10 Y PSNR variations with bitrate for BQSquare sequences (H.264/AVC with MICT).

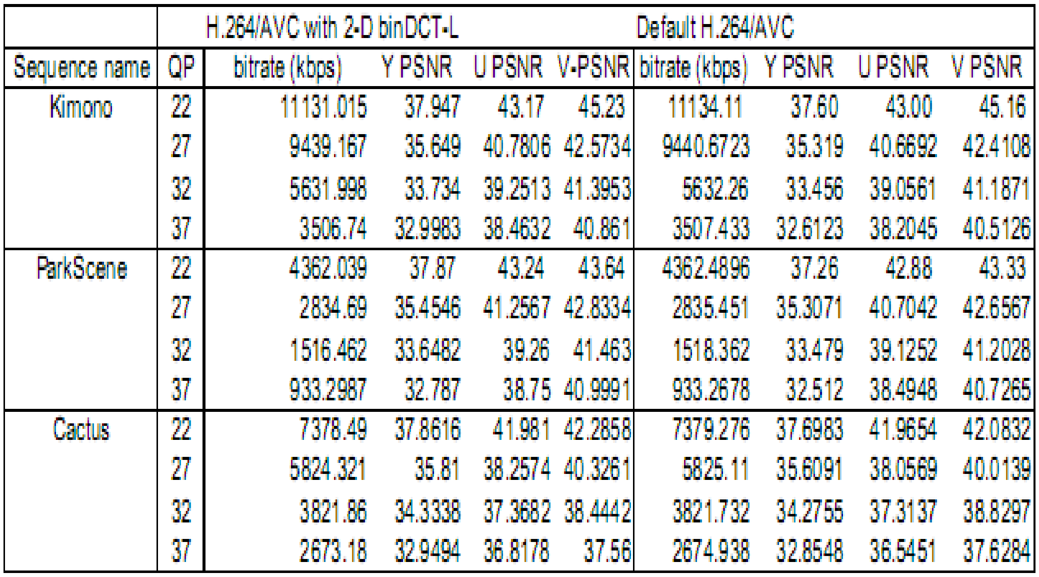
Table 5.18 BD-bitrate and BD-PSNR (H.264/AVC with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQSquare	-1.31	0.22
BlowingBubbles	-2.34	0.27
BasketballPass	-1.03	0.16

5.2.3. Performance of 2-D order 16 binDCT-L

Results for HD (1920 x 1080) sequences:

Table 5.19 Comparison of bitrates and PSNRs for three 1920 x 1080 sequences (H.264/AVC with binDCT-L).



		H.264/AVC with 2-D binDCT-L				Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Kimono	22	11131.015	37.947	43.17	45.23	11134.11	37.60	43.00	45.16
	27	9439.167	35.649	40.7806	42.5734	9440.6723	35.319	40.6692	42.4108
	32	5631.998	33.734	39.2513	41.3953	5632.26	33.456	39.0561	41.1871
	37	3506.74	32.9983	38.4632	40.861	3507.433	32.6123	38.2045	40.5126
ParkScene	22	4362.039	37.87	43.24	43.64	4362.4896	37.26	42.88	43.33
	27	2834.69	35.4546	41.2567	42.8334	2835.451	35.3071	40.7042	42.6567
	32	1516.462	33.6482	39.26	41.463	1518.362	33.479	39.1252	41.2028
	37	933.2987	32.787	38.75	40.9991	933.2678	32.512	38.4948	40.7265
Cactus	22	7378.49	37.8616	41.981	42.2858	7379.276	37.6983	41.9654	42.0832
	27	5824.321	35.81	38.2574	40.3261	5825.11	35.6091	38.0569	40.0139
	32	3821.86	34.3338	37.3682	38.4442	3821.732	34.2755	37.3137	38.8297
	37	2673.18	32.9494	36.8178	37.56	2674.938	32.8548	36.5451	37.6284

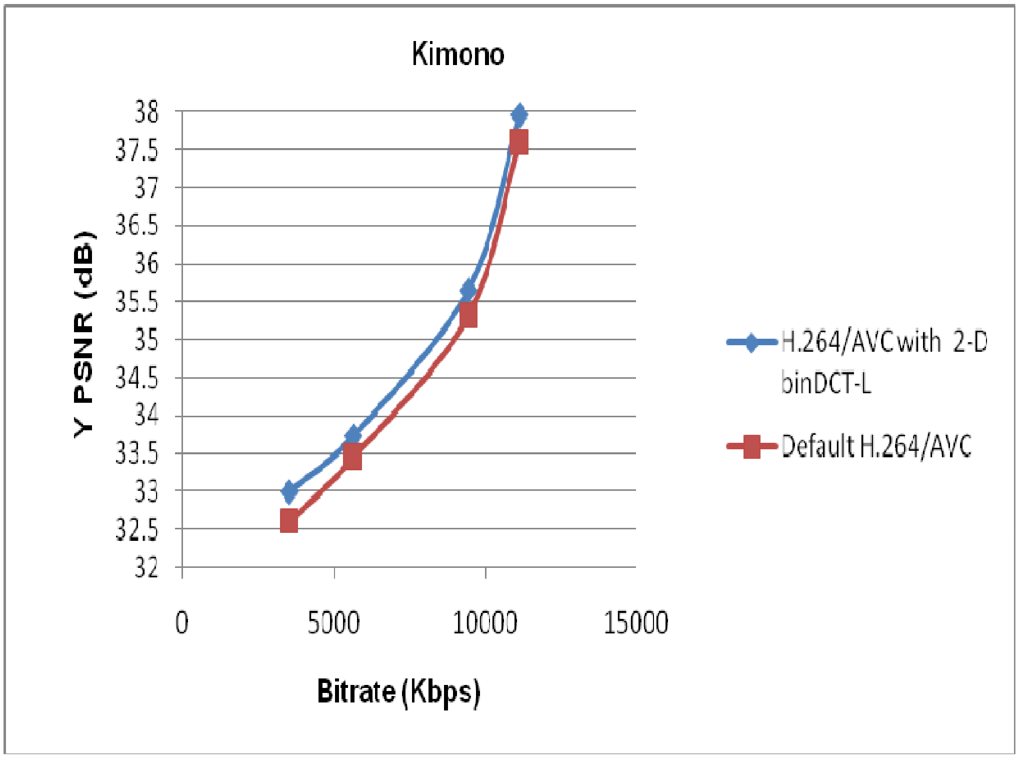


Figure 5.11 Y PSNR variations with bitrate for Kimono sequence (H.264/AVC with binDCT-L).

Table 5.20 BD-bitrate and BD-PSNR (H.264/AVC with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Kimono	-6.45	0.29
ParkScene	-4.18	0.23
Cactus	-3.17	0.16

Results for HD (1280 x 720) sequences:

Table 5.21 Comparison of bitrates and PSNRs for three 1280 x 720 sequences (H.264/AVC with binDCT-L).

	H.264/AVC with 2-D binDCT-L					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Vidyo1	22	1906.82	40.88	44.91	45.62	1907.82	40.49	44.71	45.39
	27	1601.41	39.47	44.52	44.52	1603.92	39.03	43.89	44.39
	32	1410.81	37.91	43.55	43.55	1414.94	37.52	43.04	43.25
	37	1251.1967	35.7096	42.8301	42.821	1253.0912	35.123	42.4502	42.6626
Vidyo3	22	1379.674	40.4642	46.429	45.1295	1380.88	40.0054	46.119	44.8172
	27	873.76	38.6775	45.631	43.6689	876.128	38.4117	45.533	43.6014
	32	574.832	36.8532	44.7476	42.8415	576.832	36.5035	44.6646	42.5728
	37	352.4834	34.421	43.6743	41.5289	354.9357	34.012	43.1948	40.9265
Vidyo4	22	903.478	40.3266	46.4681	46.3451	906.272	40.1075	46.012	46.0293
	27	571.246	39.1125	45.345	45.2223	573.392	38.7519	45.1035	45.0513
	32	382.546	37.6834	44.5391	43.9001	383.632	37.2184	44.0763	43.734
	37	290.041	37.1678	43.4932	42.1344	293.281	36.8548	42.5451	42.0284

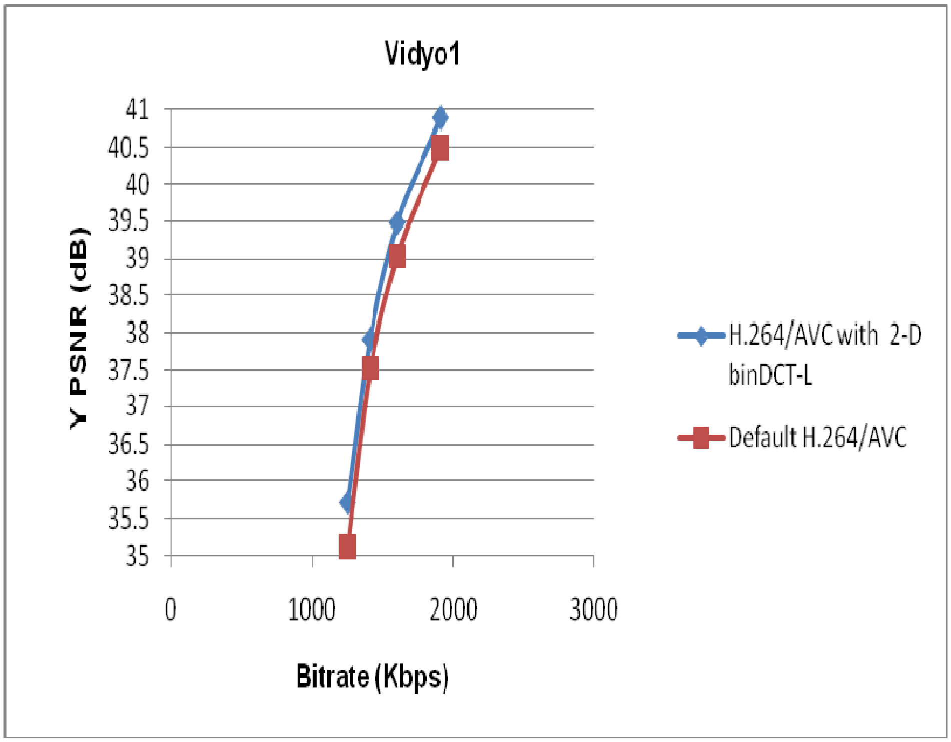


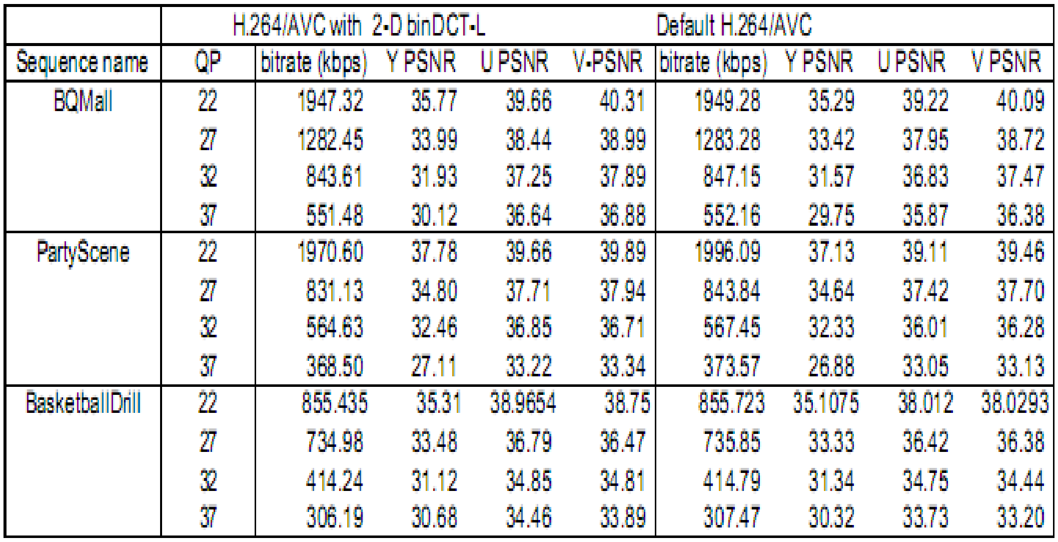
Figure 5.12 Y PSNR variations with bitrate for Vidyo1 sequence (H.264/AVC with binDCT-L).

Table 5.22 BD-bitrate and BD-PSNR (H.264/AVC with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Vidyo1	-4.73	0.36
Vidyo2	-4.35	0.31
Vidyo3	-1.59	0.19

Results for WVGA (832 x 480) sequences:

Table 5.23 Comparison of bitrates and PSNRs for three 832 x 480 sequences (H.264/AVC with binDCT-L).



	H.264/AVC with 2-D binDCT-L					Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQMall	22	1947.32	35.77	39.66	40.31	1949.28	35.29	39.22	40.09
	27	1282.45	33.99	38.44	38.99	1283.28	33.42	37.95	38.72
	32	843.61	31.93	37.25	37.89	847.15	31.57	36.83	37.47
	37	551.48	30.12	36.64	36.88	552.16	29.75	35.87	36.38
PartyScene	22	1970.60	37.78	39.66	39.89	1996.09	37.13	39.11	39.46
	27	831.13	34.80	37.71	37.94	843.84	34.64	37.42	37.70
	32	564.63	32.46	36.85	36.71	567.45	32.33	36.01	36.28
	37	368.50	27.11	33.22	33.34	373.57	26.88	33.05	33.13
BasketballDrill	22	855.435	35.31	38.9654	38.75	855.723	35.1075	38.012	38.0293
	27	734.98	33.48	36.79	36.47	735.85	33.33	36.42	36.38
	32	414.24	31.12	34.85	34.81	414.79	31.34	34.75	34.44
	37	306.19	30.68	34.46	33.89	307.47	30.32	33.73	33.20

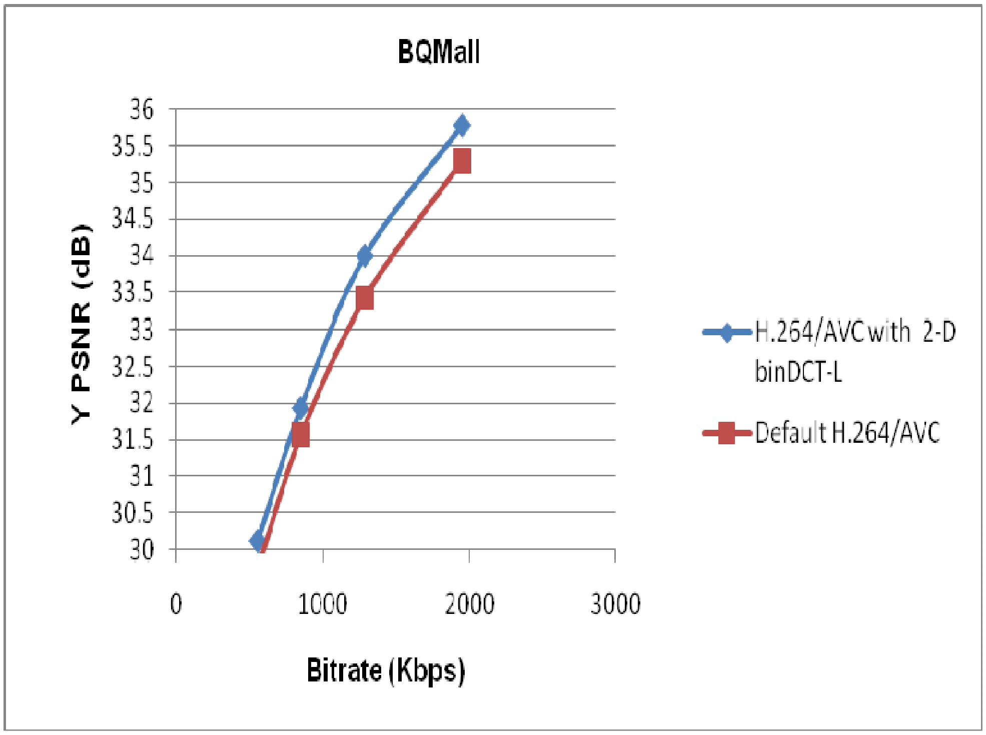


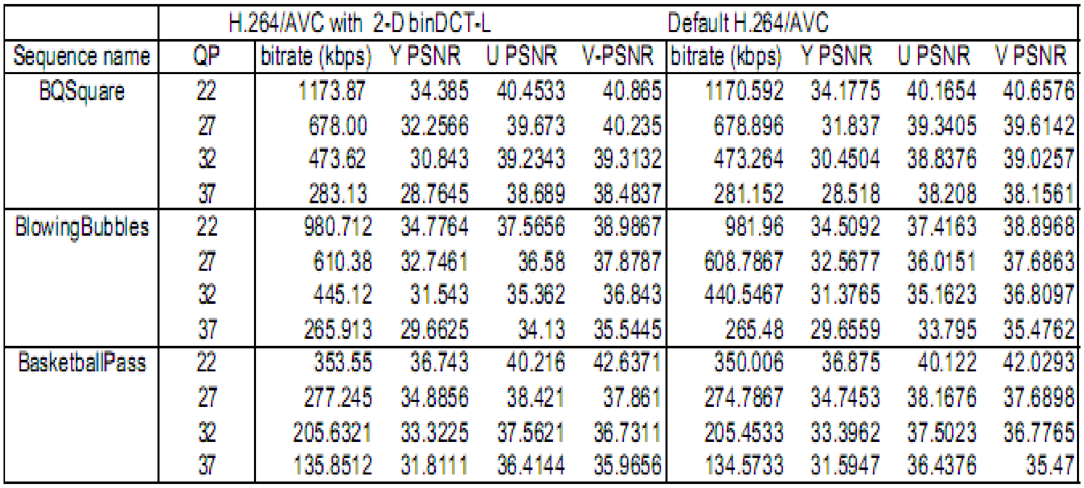
Figure 5.13 Y PSNR variations with bitrate for BQMall sequence (H.264/AVC with binDCT-L).

Table 5.24 BD-bitrate and BD-PSNR (H.264/AVC with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQMall	-6.65	0.47
PartyScene	-5.92	0.32
BasketballDrill	-3.75	0.19

Results for WQVGA (416 x 240) sequences:

Table 5.25 Comparison of bitrates and PSNRs for three 416 x 240 sequences (H.264/AVC with binDCT-L).



		H.264/AVC with 2-D binDCT-L				Default H.264/AVC			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQSquare	22	1173.87	34.385	40.4533	40.865	1170.592	34.1775	40.1654	40.6576
	27	678.00	32.2566	39.673	40.235	678.896	31.837	39.3405	39.6142
	32	473.62	30.843	39.2343	39.3132	473.264	30.4504	38.8376	39.0257
	37	283.13	28.7645	38.689	38.4837	281.152	28.518	38.208	38.1561
BlowingBubbles	22	980.712	34.7764	37.5656	38.9867	981.96	34.5092	37.4163	38.8968
	27	610.38	32.7461	36.58	37.8787	608.7867	32.5677	36.0151	37.6863
	32	445.12	31.543	35.362	36.843	440.5467	31.3765	35.1623	36.8097
	37	265.913	29.6625	34.13	35.5445	265.48	29.6559	33.795	35.4762
BasketballPass	22	353.55	36.743	40.216	42.6371	350.006	36.875	40.122	42.0293
	27	277.245	34.8856	38.421	37.861	274.7867	34.7453	38.1676	37.6898
	32	205.6321	33.3225	37.5621	36.7311	205.4533	33.3962	37.5023	36.7765
	37	135.8512	31.8111	36.4144	35.9656	134.5733	31.5947	36.4376	35.47

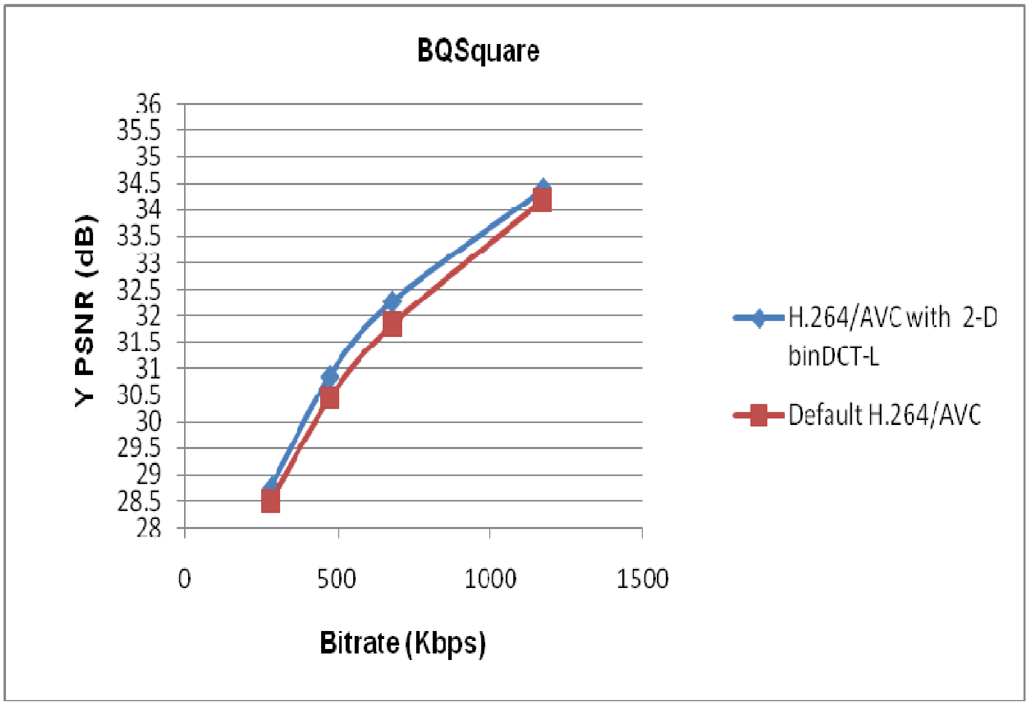


Figure 5.14 Y PSNR variations with bitrate for BQSquare sequence (H.264/AVC with binDCT-L).

Table 5.26 BD-bitrate and BD-PSNR (H.264/AVC with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQSquare	-1.13	0.19
BlowingBubbles	-0.65	0.06
BasketballPass	1.52	-0.15

5.3 Implementation in AVS-video and performance analysis

The three 2-D order 16 ICTs are implemented into RM 52e reference software [70] for AVS-video. The combined scaling and quantization matrices used for simple and modified 2-D order 16 ICTs are the same as that in RM 52e as they are extended versions of the order 8 ICT, while that used for 2-D order 16 binDCT-L is specified in the Appendix. . The selection of transform size is based on MB level R-D cost [66]. The AVS enhanced profile (Jiaqiang) is used for encoding the video sequences. Table 5.3 specifies some of the configuration parameters used for encoding. The encoding is performed with a system having Intel i7 Quad 4, 2.60 GHz processor supporting 6GB RAM. The operating system used is Windows 7. Video sequences belonging to HD (1920 x 1080, 1280 x 720), WVGA (832 x 480) and WQVGA (416 x 240) are used for evaluating the performance. The results (bitrates verses PSNRs, percentage bit rate savings (BD-bitrate) verses BD-PSNR for each sequence) are tabulated. The graphs for one sequence from each of the different resolutions are also shown.

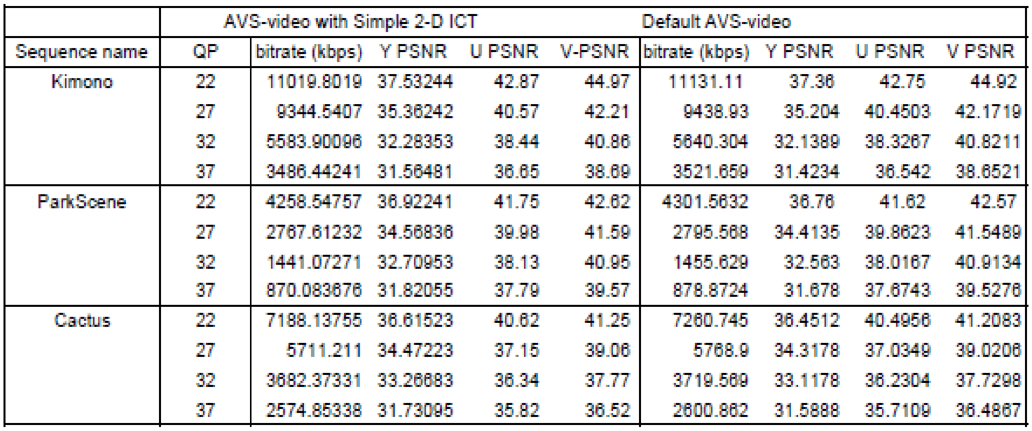
Table 5.27 Configuration Parameters.

Group of pictures (GOP) size	8
GOP structure	IBBBBBBP
Intra frame period	0.5 s
R-D optimization	on
QP	22, 27, 32, 37
Reference frames	2
Fast motion estimation	on
Search range	± 32
Deblocking filter	on
Entropy coding	CABAC

5.3.1. Performance of Simple 2-D order 16 ICT and results

Results for HD (1920 x 1080) sequences:

Table 5.28 Comparison of bitrates and PSNRs for three 1920 x 1080 sequences (AVS-video with SICT).



	AVS-video with Simple 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Kimono	22	11019.8019	37.53244	42.87	44.97	11131.11	37.36	42.75	44.92
	27	9344.5407	35.36242	40.57	42.21	9438.93	35.204	40.4503	42.1719
	32	5583.90096	32.28353	38.44	40.86	5640.304	32.1389	38.3267	40.8211
	37	3486.44241	31.56481	36.65	38.69	3521.659	31.4234	36.542	38.6521
ParkScene	22	4258.54757	36.92241	41.75	42.62	4301.5632	36.76	41.62	42.57
	27	2767.61232	34.56836	39.98	41.59	2795.568	34.4135	39.8623	41.5489
	32	1441.07271	32.70953	38.13	40.95	1455.629	32.563	38.0167	40.9134
	37	870.083676	31.82055	37.79	39.57	878.8724	31.678	37.6743	39.5276
Cactus	22	7188.13755	36.61523	40.62	41.25	7260.745	36.4512	40.4956	41.2083
	27	5711.211	34.47223	37.15	39.06	5768.9	34.3178	37.0349	39.0206
	32	3682.37331	33.26683	36.34	37.77	3719.569	33.1178	36.2304	37.7298
	37	2574.85338	31.73095	35.82	36.52	2600.862	31.5888	35.7109	36.4867

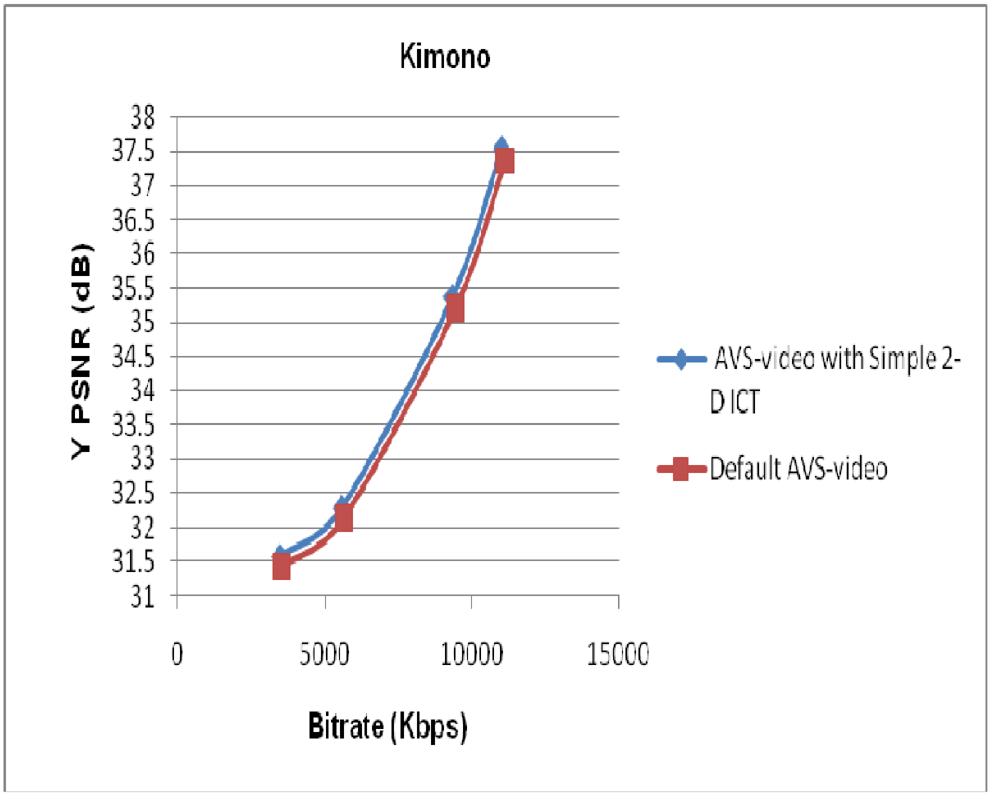


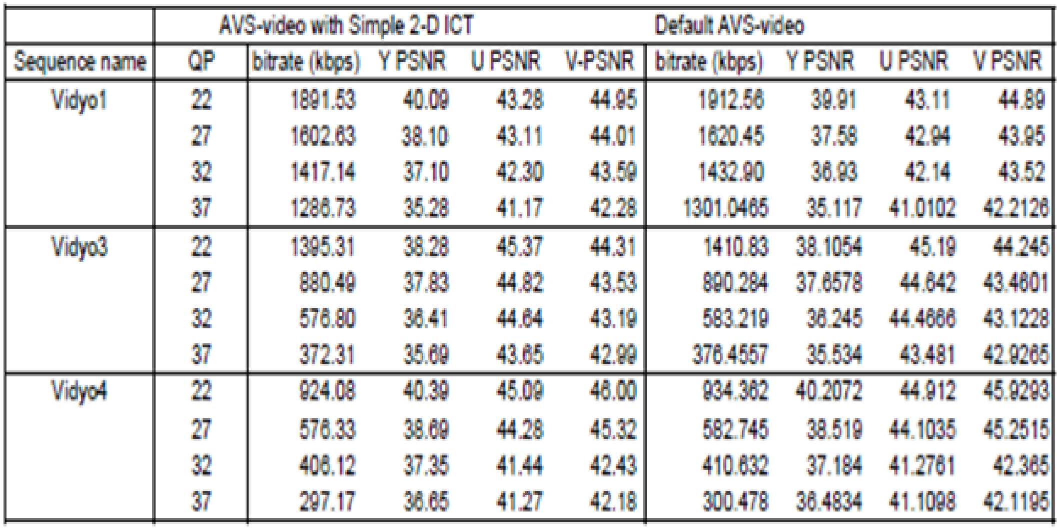
Figure 5.15 Y PSNR variations with bitrate for Kimono sequence (AVS-video with SICT).

Table 5.29 BD-bitrate and BD-PSNR (AVS-video with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Kimono	-8.45	0.37
ParkScene	-7.31	0.33
Cactus	-5.51	0.26

Results for HD (1280 x 720) sequences:

Table 5.30 Comparison of bitrates and PSNRs for three 1280 x 720 sequences (AVS-video with SICT).



	AVS-video with Simple 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Vidyo1	22	1891.53	40.09	43.28	44.95	1912.56	39.91	43.11	44.89
	27	1602.63	38.10	43.11	44.01	1620.45	37.58	42.94	43.95
	32	1417.14	37.10	42.30	43.59	1432.90	36.93	42.14	43.52
	37	1286.73	35.28	41.17	42.28	1301.0465	35.117	41.0102	42.2126
Vidyo3	22	1395.31	38.28	45.37	44.31	1410.83	38.1054	45.19	44.245
	27	880.49	37.83	44.82	43.53	890.284	37.6578	44.642	43.4601
	32	576.80	36.41	44.64	43.19	583.219	36.245	44.4666	43.1228
	37	372.31	35.69	43.65	42.99	376.4557	35.534	43.481	42.9265
Vidyo4	22	924.08	40.39	45.09	46.00	934.362	40.2072	44.912	45.9293
	27	576.33	38.69	44.28	45.32	582.745	38.519	44.1035	45.2515
	32	406.12	37.35	41.44	42.43	410.632	37.184	41.2761	42.365
	37	297.17	36.65	41.27	42.18	300.478	36.4834	41.1098	42.1195

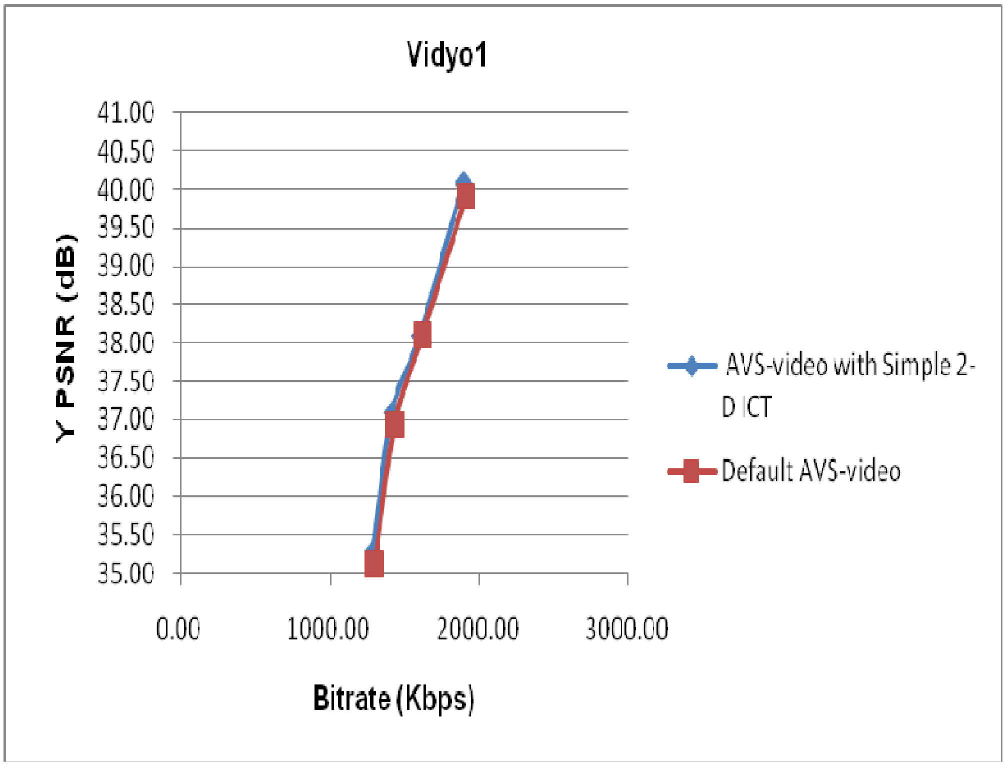


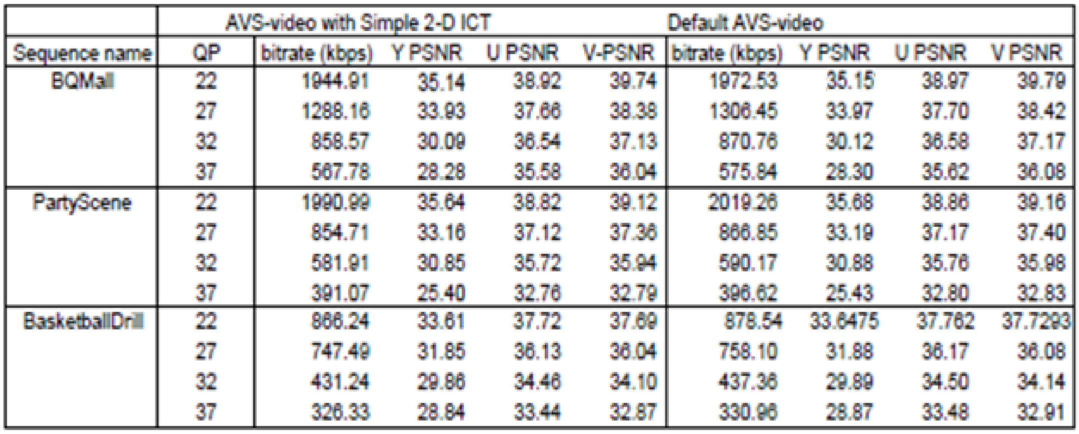
Figure 5.16 Y PSNR variations with bitrate for Vidyo1 sequence (AVS-video with SICT).

Table 5.31 BD-bitrate and BD-PSNR (AVS-video with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Vidyo1	-5.18	0.29
Vidyo2	-6.57	0.34
Vidyo3	-3.23	0.21

Results for WVGA (832 x 480) sequences:

Table 5.32 Comparison of bitrates and PSNRs for three 832 x 480 sequences (AVS-video with SICT).



	AVS-video with Simple 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQMall	22	1944.91	35.14	38.92	39.74	1972.53	35.15	38.97	39.79
	27	1288.16	33.93	37.66	38.38	1306.45	33.97	37.70	38.42
	32	858.57	30.09	36.54	37.13	870.76	30.12	36.58	37.17
	37	567.78	28.28	35.58	36.04	575.84	28.30	35.62	36.08
PartyScene	22	1990.99	35.64	38.82	39.12	2019.26	35.68	38.86	39.16
	27	854.71	33.16	37.12	37.36	866.85	33.19	37.17	37.40
	32	581.91	30.85	35.72	35.94	590.17	30.88	35.76	35.98
	37	391.07	25.40	32.76	32.79	396.62	25.43	32.80	32.83
BasketballDrill	22	866.24	33.61	37.72	37.69	878.54	33.6475	37.762	37.7293
	27	747.49	31.85	36.13	36.04	758.10	31.88	36.17	36.08
	32	431.24	29.86	34.46	34.10	437.36	29.89	34.50	34.14
	37	326.33	28.84	33.44	32.87	330.96	28.87	33.48	32.91

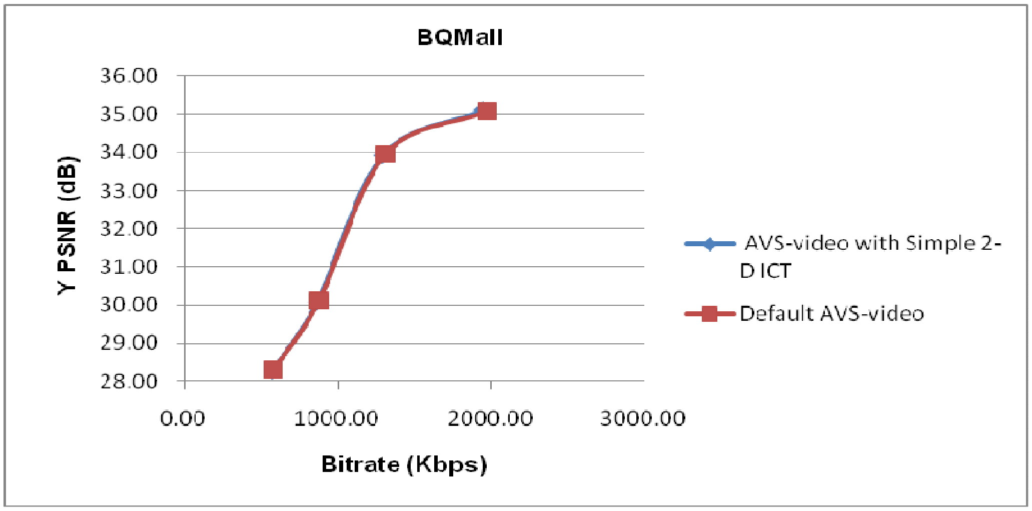


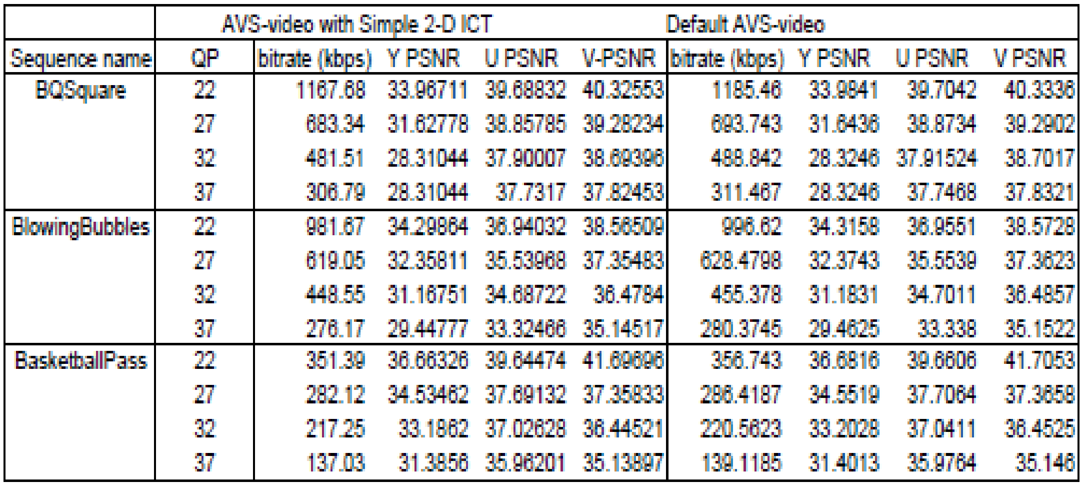
Figure 5.17 Y PSNR variations with bitrate for BQMall sequence (AVS-video with SICT).

Table 5.33 BD-bitrate and BD-PSNR (AVS-video with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQMall	1.14	-0.02
PartyScene	1.26	-0.03
BasketballDrill	1.11	-0.01

Results for WQVGA (416 x 240) sequences:

Table 5.34 Comparison of bitrates and PSNRs for three 416 x 240 sequences (AVS-video with SICT).



	AVS-video with Simple 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQSquare	22	1167.68	33.96711	39.68832	40.32553	1185.46	33.9841	39.7042	40.3336
	27	683.34	31.62778	38.85785	39.28234	693.743	31.6436	38.8734	39.2902
	32	481.51	28.31044	37.90007	38.69396	488.842	28.3246	37.91524	38.7017
	37	306.79	28.31044	37.7317	37.82453	311.467	28.3246	37.7468	37.8321
BlowingBubbles	22	981.67	34.29864	36.94032	38.56509	996.62	34.3158	36.9551	38.5728
	27	619.05	32.35811	35.53968	37.35483	628.4798	32.3743	35.5539	37.3623
	32	448.55	31.16751	34.68722	36.4784	455.378	31.1831	34.7011	36.4857
	37	276.17	29.44777	33.32466	35.14517	280.3745	29.4625	33.338	35.1522
BasketballPass	22	351.39	36.66326	39.64474	41.69696	356.743	36.6816	39.6606	41.7053
	27	282.12	34.53462	37.69132	37.35833	286.4187	34.5519	37.7064	37.3658
	32	217.25	33.1862	37.02628	36.44521	220.5623	33.2028	37.0411	36.4525
	37	137.03	31.3856	35.96201	35.13897	139.1185	31.4013	35.9764	35.146

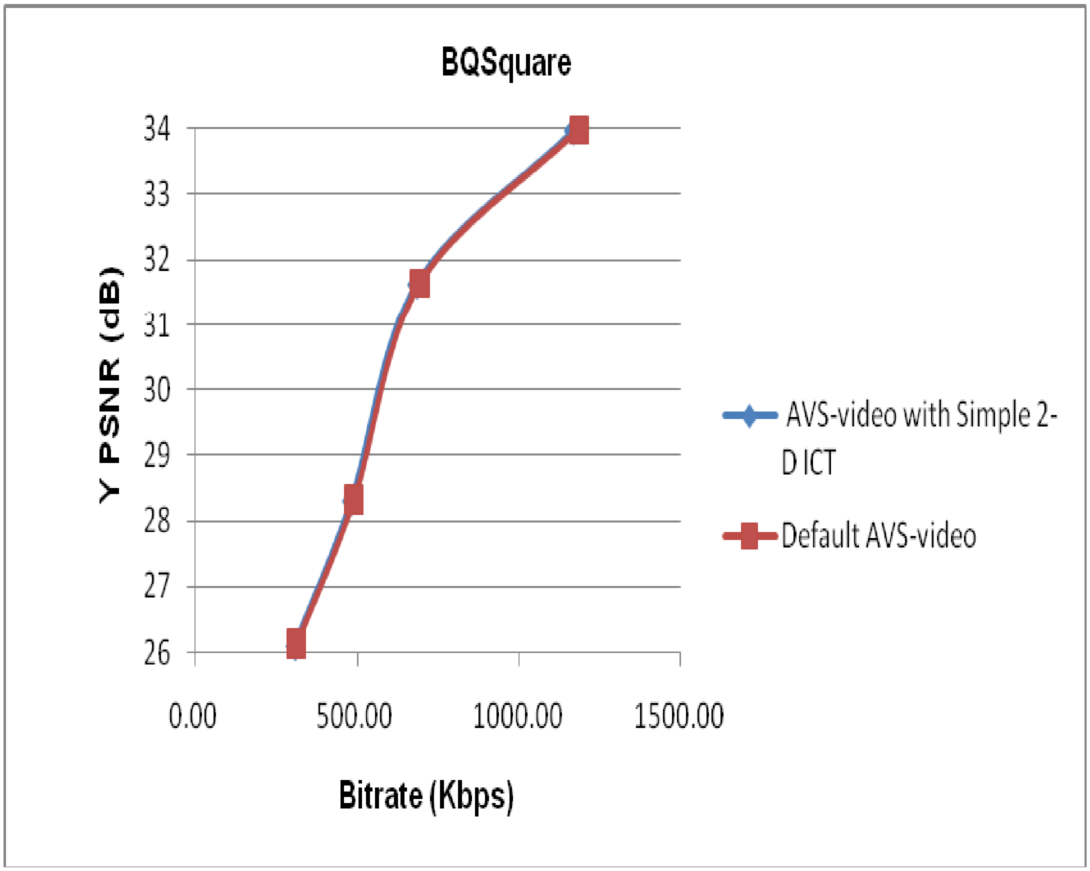


Figure 5.18 Y PSNR variations with bitrate for BQSquare sequence (AVS-video with SICT).

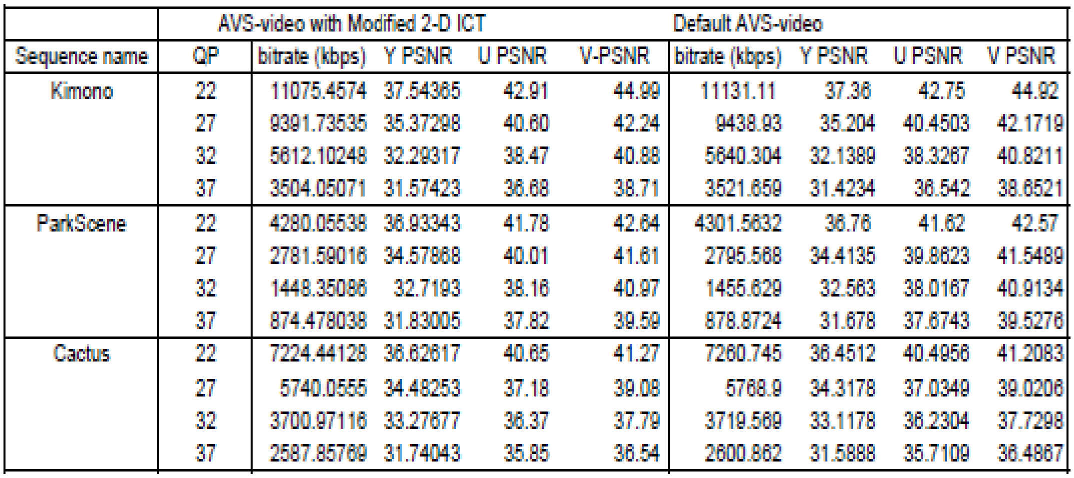
Table 5.35 BD-bitrate and BD-PSNR (AVS-video with SICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQSquare	1.01	-0.01
BlowingBubbles	2.18	-0.08
BasketballPass	1.96	-0.06

5.3.2. Performance of Modified 2-D order 16 ICT

Results for HD (1920 x 1080) sequences:

Table 5.36 Comparison of bitrates and PSNRs for three 1920 x 1080 sequences (AVS-video with MICT).



	AVS-video with Modified 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Kimono	22	11075.4574	37.54365	42.91	44.99	11131.11	37.36	42.75	44.92
	27	9391.73535	35.37298	40.60	42.24	9438.93	35.204	40.4503	42.1719
	32	5612.10248	32.29317	38.47	40.88	5640.304	32.1389	38.3267	40.8211
	37	3504.05071	31.57423	36.68	38.71	3521.659	31.4234	36.542	38.6521
ParkScene	22	4280.05538	36.93343	41.78	42.64	4301.5632	36.76	41.62	42.57
	27	2781.59016	34.57868	40.01	41.61	2795.568	34.4135	39.8623	41.5489
	32	1448.35086	32.7193	38.16	40.97	1455.629	32.563	38.0167	40.9134
	37	874.478038	31.83005	37.82	39.59	878.8724	31.678	37.6743	39.5276
Cactus	22	7224.44128	36.62617	40.65	41.27	7260.745	36.4512	40.4956	41.2083
	27	5740.0555	34.48253	37.18	39.08	5768.9	34.3178	37.0349	39.0206
	32	3700.97116	33.27677	36.37	37.79	3719.569	33.1178	36.2304	37.7298
	37	2587.85769	31.74043	35.85	36.54	2600.862	31.5888	35.7109	36.4867

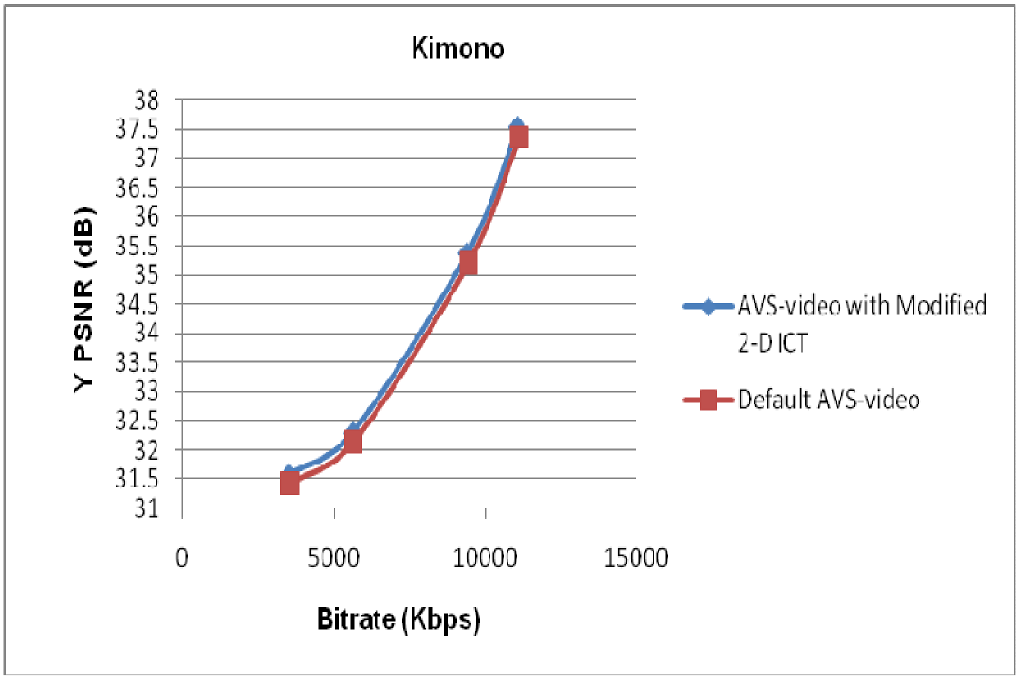


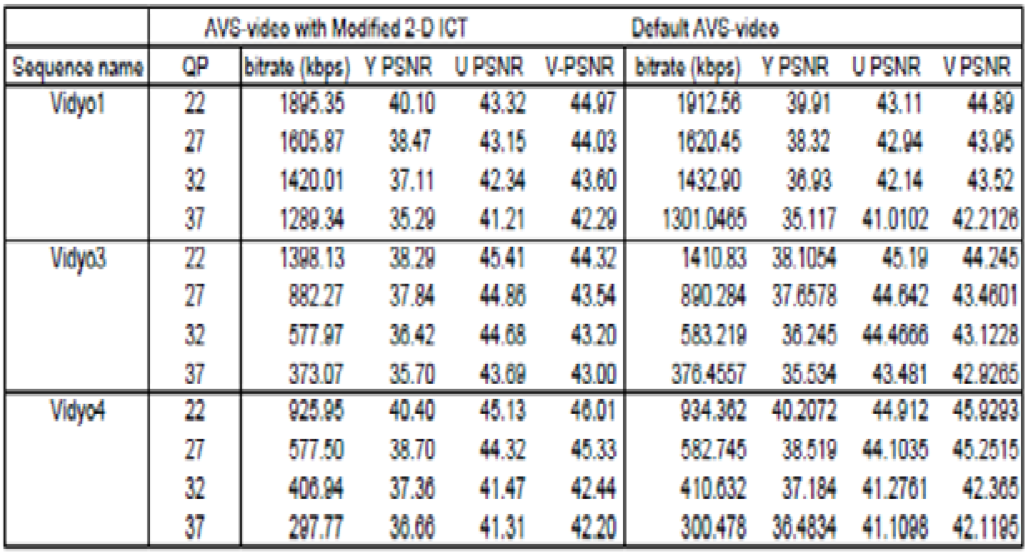
Figure 5.19 Y PSNR variations with bitrate for Kimono sequence (AVS-video with MICT).

Table 5.37 BD-bitrate and BD-PSNR (AVS-video with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Kimono	-5.79	0.37
ParkScene	-4.61	0.29
Cactus	-3.12	0.24

Results for HD (1280 x 720) sequences:

Table 5.38 Comparison of bitrates and PSNRs for three 1280 x 720 sequences (AVS-video with MICT).



	AVS-video with Modified 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Vidyo1	22	1895.35	40.10	43.32	44.97	1912.56	39.91	43.11	44.89
	27	1605.87	38.47	43.15	44.03	1620.45	38.32	42.94	43.95
	32	1420.01	37.11	42.34	43.60	1432.90	36.93	42.14	43.52
	37	1289.34	35.29	41.21	42.29	1301.0465	35.117	41.0102	42.2126
Vidyo3	22	1398.13	38.29	45.41	44.32	1410.83	38.1054	45.19	44.245
	27	882.27	37.84	44.86	43.54	890.284	37.6578	44.642	43.4601
	32	577.97	36.42	44.68	43.20	583.219	36.245	44.4666	43.1228
	37	373.07	35.70	43.69	43.00	376.4557	35.534	43.481	42.9265
Vidyo4	22	925.95	40.40	45.13	46.01	934.362	40.2072	44.912	45.9293
	27	577.50	38.70	44.32	45.33	582.745	38.519	44.1035	45.2515
	32	406.94	37.36	41.47	42.44	410.632	37.184	41.2761	42.365
	37	297.77	36.66	41.31	42.20	300.478	36.4834	41.1098	42.1195

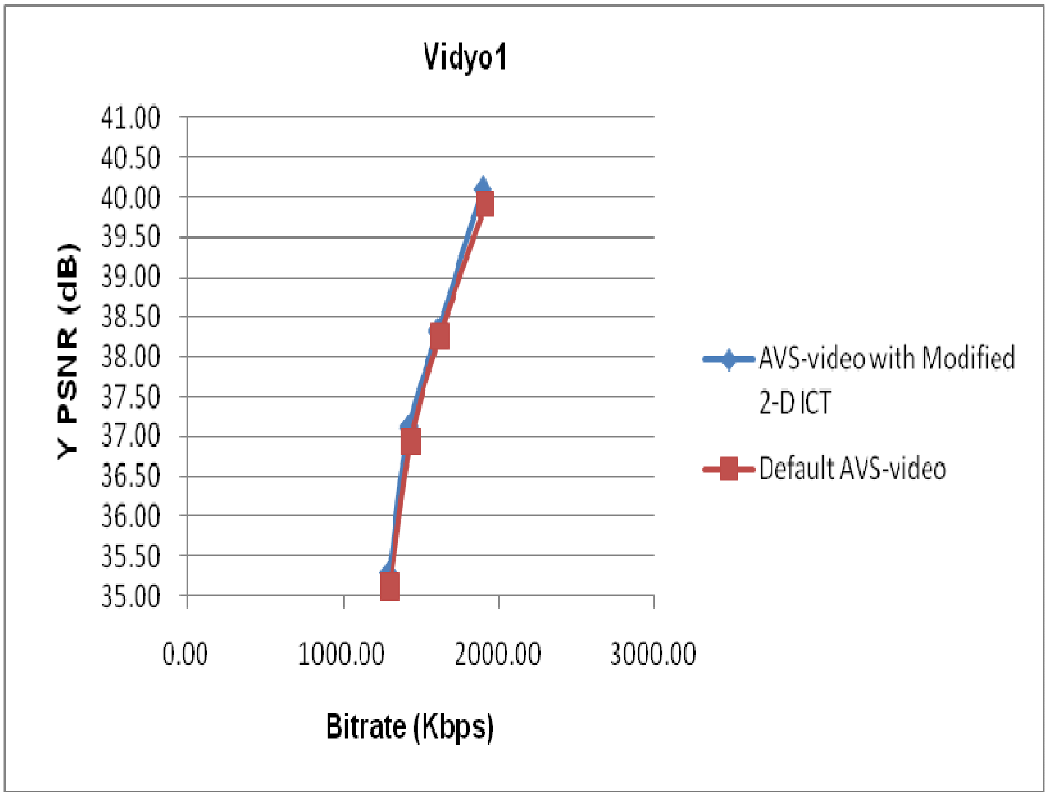


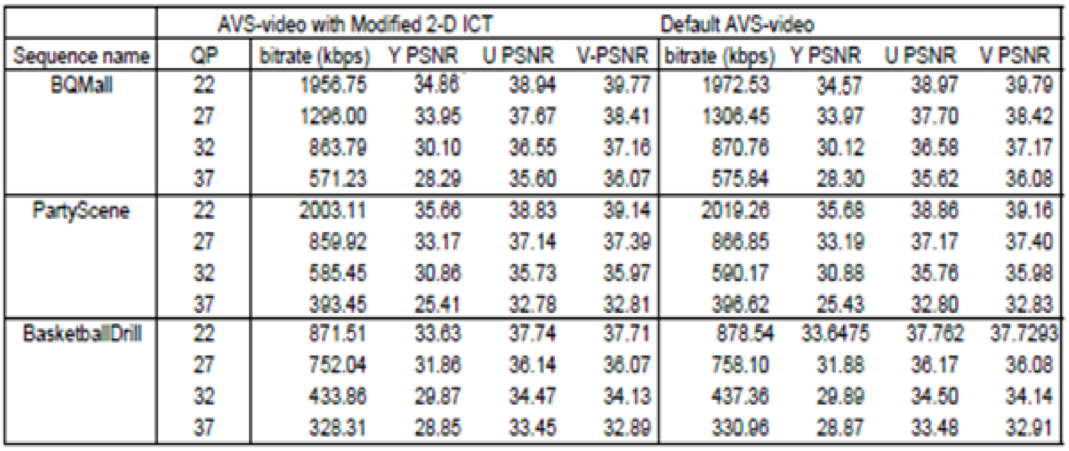
Figure 5.20 Y PSNR variations with bitrate for Vidyo1 sequence (AVS-video with MICT).

Table 5.39 BD-bitrate and BD-PSNR (AVS-video with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Vidyo1	-2.57	0.34
Vidyo2	-2.68	0.26
Vidyo3	-3.12	0.33

Results for WVGA (832 x 480) sequences:

Table 5.40 Comparison of bitrates and PSNRs for three 832 x 480 sequences (AVS-video with MICT).



	AVS-video with Modified 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQMall	22	1956.75	34.86	38.94	39.77	1972.53	34.57	38.97	39.79
	27	1296.00	33.95	37.67	38.41	1306.45	33.97	37.70	38.42
	32	863.79	30.10	36.55	37.16	870.76	30.12	36.58	37.17
	37	571.23	28.29	35.60	36.07	575.84	28.30	35.62	36.08
PartyScene	22	2003.11	35.66	38.83	39.14	2019.26	35.68	38.86	39.16
	27	859.92	33.17	37.14	37.39	866.85	33.19	37.17	37.40
	32	585.45	30.86	35.73	35.97	590.17	30.88	35.76	35.98
	37	393.45	25.41	32.78	32.81	396.62	25.43	32.80	32.83
BasketballDrill	22	871.51	33.63	37.74	37.71	878.54	33.6475	37.762	37.7293
	27	752.04	31.86	36.14	36.07	758.10	31.88	36.17	36.08
	32	433.86	29.87	34.47	34.13	437.36	29.89	34.50	34.14
	37	328.31	28.85	33.45	32.89	330.96	28.87	33.48	32.91

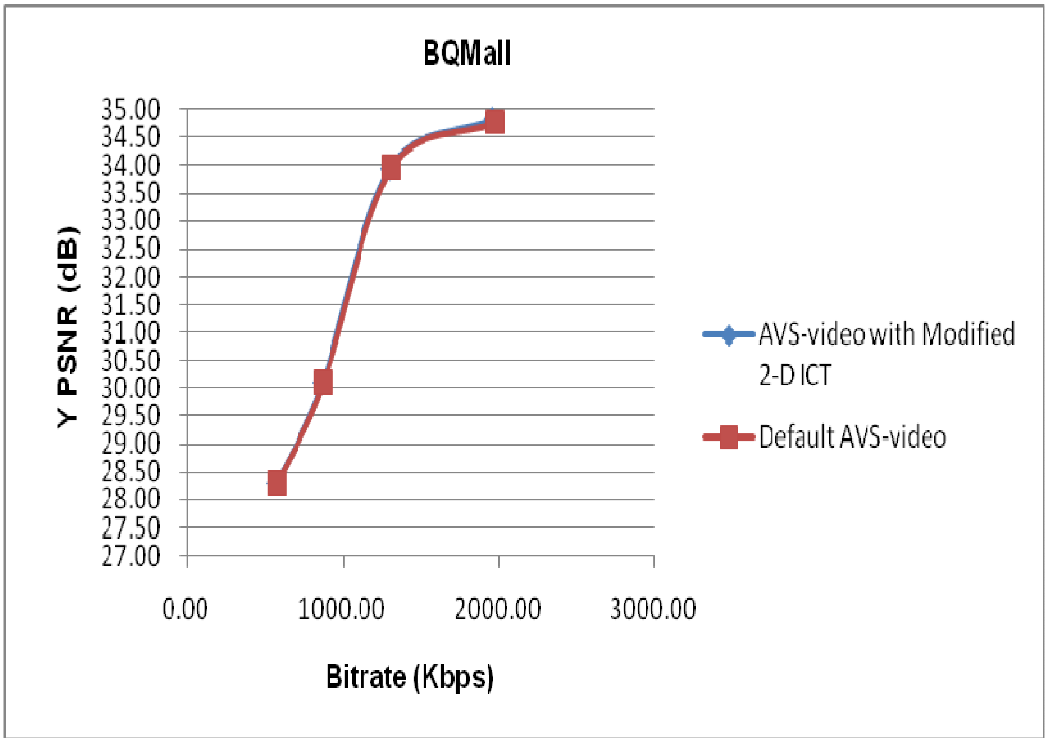


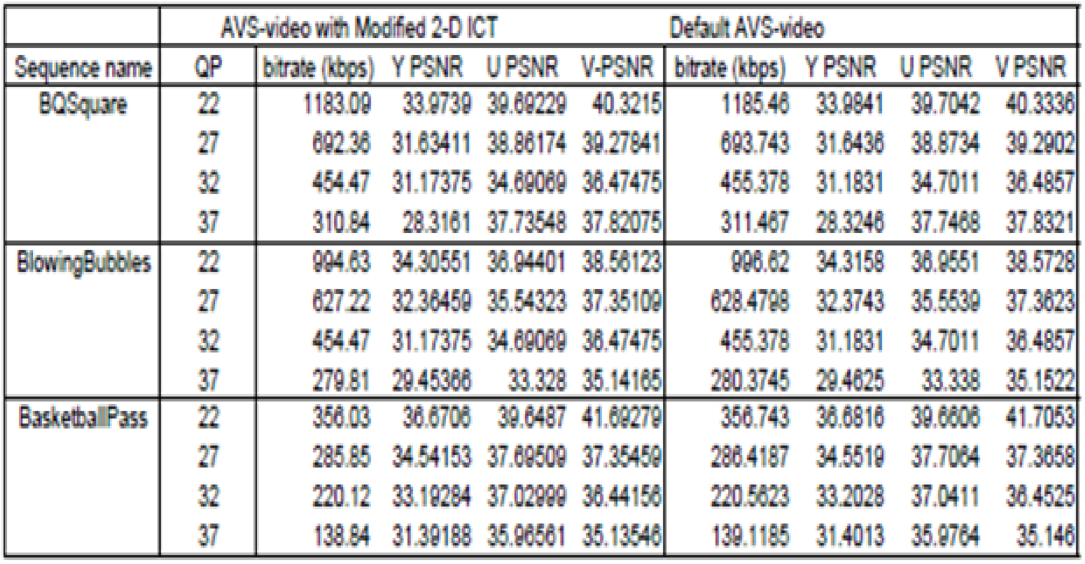
Figure 5.21 Y PSNR variations with bitrate for BQMall sequence (AVS-video with MICT).

Table 5.41 BD-bitrate and BD-PSNR (AVS-video with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQMall	0.35	-0.01
PartyScene	0.27	-0.02
BasketballDrill	0.34	-0.01

Results for WQVGA (416 x 240) sequences:

Table 5.42 Comparison of bitrates and PSNRs for three 416 x 240sequences (AVS-video with MICT).



	AVS-video with Modified 2-D ICT					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQSquare	22	1183.09	33.9739	39.69229	40.3215	1185.46	33.9841	39.7042	40.3336
	27	692.36	31.63411	38.86174	39.27841	693.743	31.6436	38.8734	39.2902
	32	454.47	31.17375	34.69069	36.47475	455.378	31.1831	34.7011	36.4857
	37	310.84	28.3161	37.73548	37.82075	311.467	28.3246	37.7468	37.8321
BlowingBubbles	22	994.63	34.30551	36.94401	38.56123	996.62	34.3158	36.9551	38.5728
	27	627.22	32.36459	35.54323	37.35109	628.4798	32.3743	35.5539	37.3623
	32	454.47	31.17375	34.69069	36.47475	455.378	31.1831	34.7011	36.4857
	37	279.81	29.45366	33.328	35.14165	280.3745	29.4625	33.338	35.1522
BasketballPass	22	356.03	36.6706	39.6487	41.69279	356.743	36.6816	39.6606	41.7053
	27	285.85	34.54153	37.69509	37.35459	286.4187	34.5519	37.7064	37.3658
	32	220.12	33.19284	37.02999	36.44156	220.5623	33.2028	37.0411	36.4525
	37	138.84	31.39188	35.96561	35.13546	139.1185	31.4013	35.9764	35.146

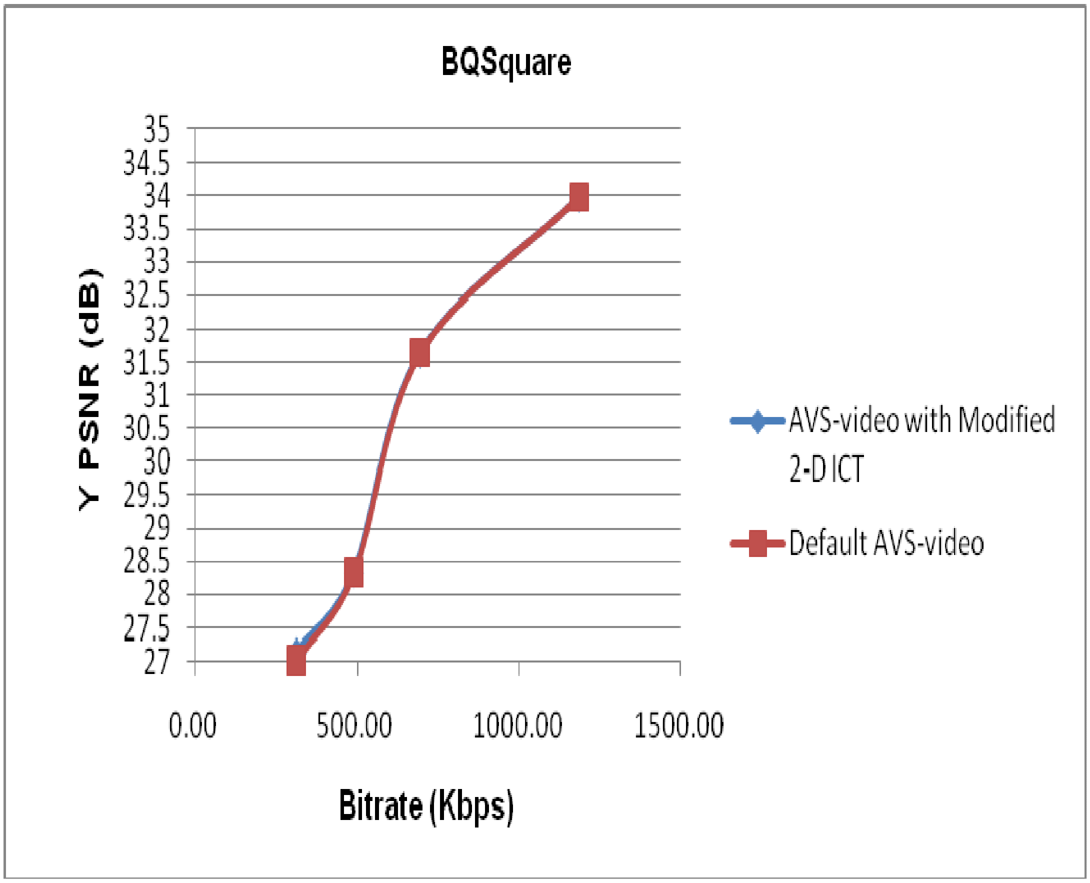


Figure 5.22 Y PSNR variations with bitrate for BQSquare sequence (AVS-video with MICT).

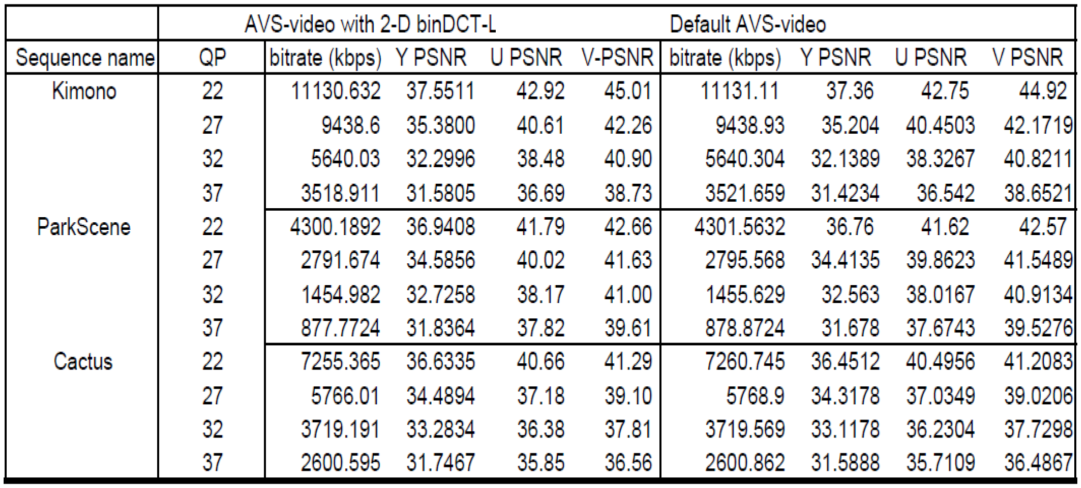
Table 5.43 BD-bitrate and BD-PSNR (AVS-video with MICT).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQSquare	0.15	-0.01
BlowingBubbles	0.17	-0.01
BasketballPass	0.36	-0.02

5.3.3. Performance of 2-D order 16 binDCT-L

Results for HD (1920 x 1080) sequences:

Table 5.44 Comparison of bitrates and PSNRs for three 1920 x 1080 sequences (AVS-video with binDCT-L).



		AVS-video with 2-D binDCT-L				Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Kimono	22	11130.632	37.5511	42.92	45.01	11131.11	37.36	42.75	44.92
	27	9438.6	35.3800	40.61	42.26	9438.93	35.204	40.4503	42.1719
	32	5640.03	32.2996	38.48	40.90	5640.304	32.1389	38.3267	40.8211
	37	3518.911	31.5805	36.69	38.73	3521.659	31.4234	36.542	38.6521
ParkScene	22	4300.1892	36.9408	41.79	42.66	4301.5632	36.76	41.62	42.57
	27	2791.674	34.5856	40.02	41.63	2795.568	34.4135	39.8623	41.5489
	32	1454.982	32.7258	38.17	41.00	1455.629	32.563	38.0167	40.9134
	37	877.7724	31.8364	37.82	39.61	878.8724	31.678	37.6743	39.5276
Cactus	22	7255.365	36.6335	40.66	41.29	7260.745	36.4512	40.4956	41.2083
	27	5766.01	34.4894	37.18	39.10	5768.9	34.3178	37.0349	39.0206
	32	3719.191	33.2834	36.38	37.81	3719.569	33.1178	36.2304	37.7298
	37	2600.595	31.7467	35.85	36.56	2600.862	31.5888	35.7109	36.4867

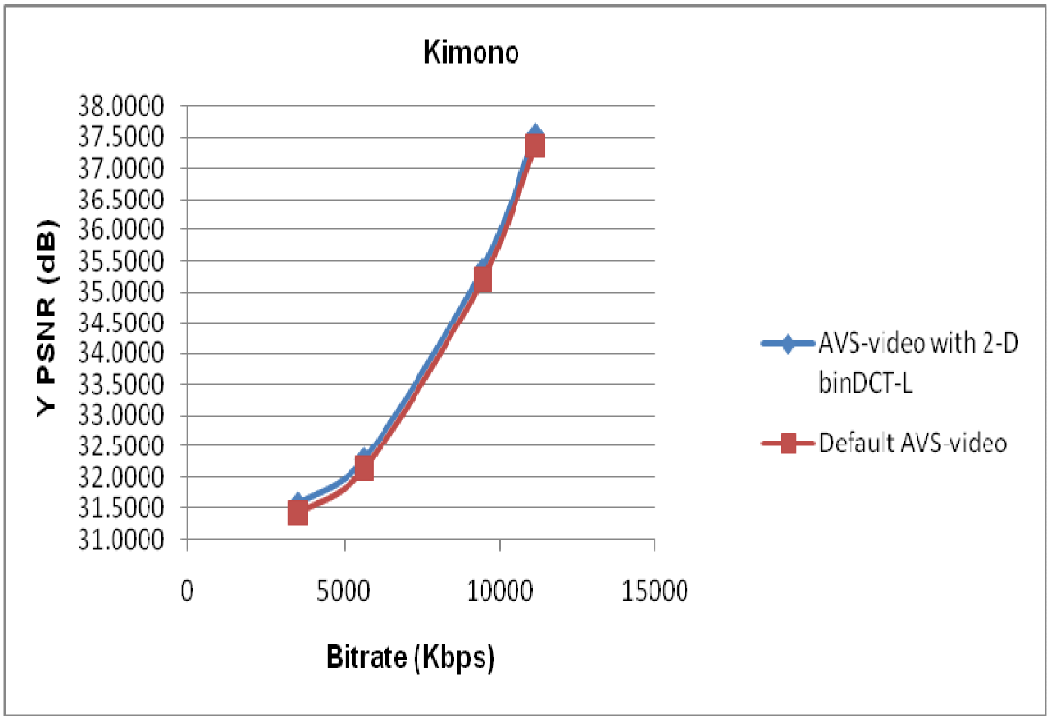


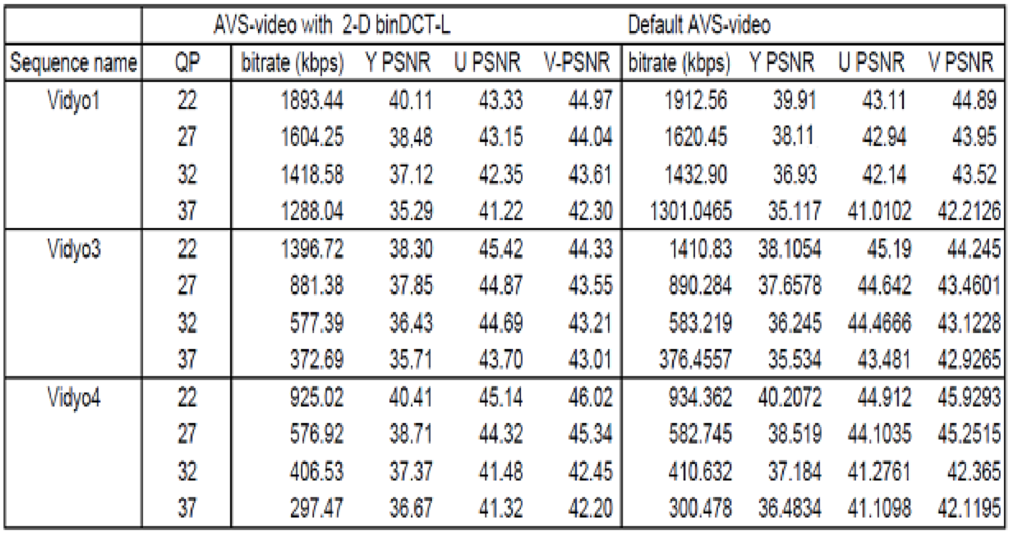
Figure 5.23 Y PSNR variations with bitrate for Kimono sequence (AVS-video with binDCT-L).

Table 5.45 BD-bitrate and BD-PSNR (AVS-video with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Kimono	-8.24	0.34
ParkScene	-7.83	0.28
Cactus	-8.11	0.32

Results for HD (1280 x 720) sequences:

Table 5.46 Comparison of bitrates and PSNRs for three 1280 x 720 sequences (AVS-video with binDCT-L).



	AVS-video with 2-D binDCT-L					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
Vidyo1	22	1893.44	40.11	43.33	44.97	1912.56	39.91	43.11	44.89
	27	1604.25	38.48	43.15	44.04	1620.45	38.11	42.94	43.95
	32	1418.58	37.12	42.35	43.61	1432.90	36.93	42.14	43.52
	37	1288.04	35.29	41.22	42.30	1301.0465	35.117	41.0102	42.2126
Vidyo3	22	1396.72	38.30	45.42	44.33	1410.83	38.1054	45.19	44.245
	27	881.38	37.85	44.87	43.55	890.284	37.6578	44.642	43.4601
	32	577.39	36.43	44.69	43.21	583.219	36.245	44.4666	43.1228
	37	372.69	35.71	43.70	43.01	376.4557	35.534	43.481	42.9265
Vidyo4	22	925.02	40.41	45.14	46.02	934.362	40.2072	44.912	45.9293
	27	576.92	38.71	44.32	45.34	582.745	38.519	44.1035	45.2515
	32	406.53	37.37	41.48	42.45	410.632	37.184	41.2761	42.365
	37	297.47	36.67	41.32	42.20	300.478	36.4834	41.1098	42.1195

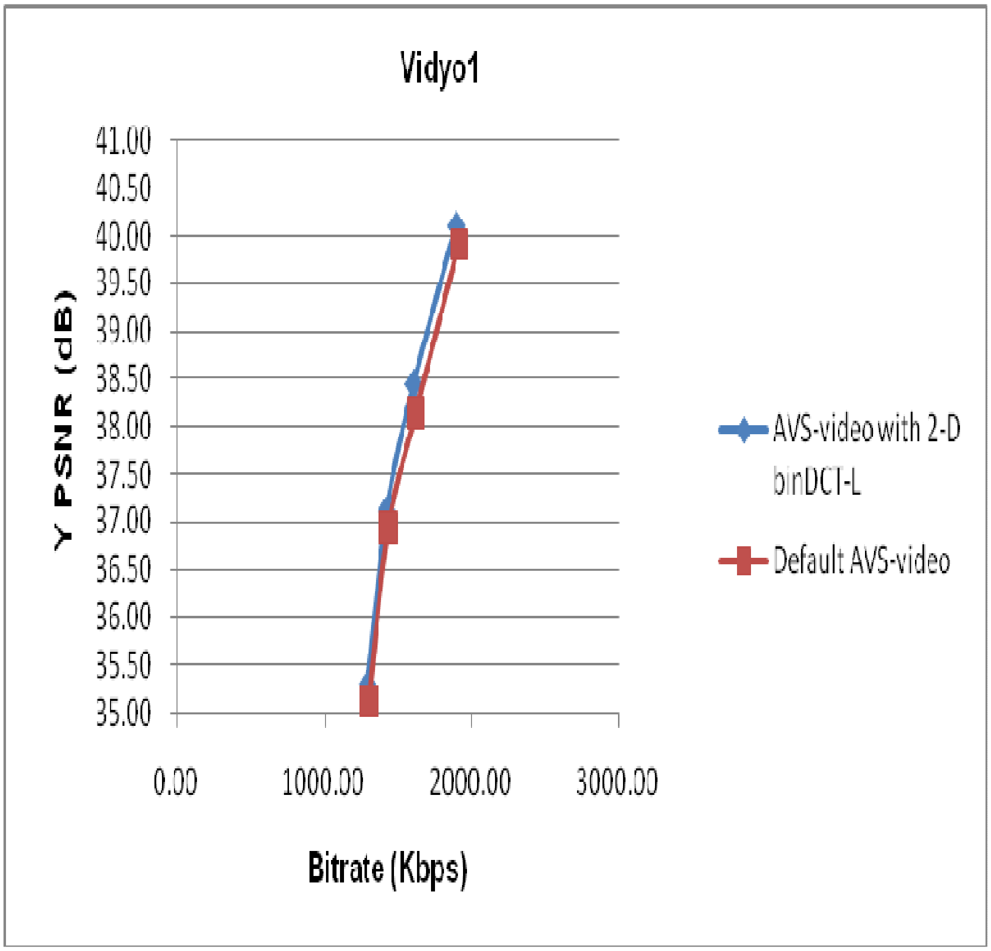


Figure 5.24 Y PSNR variations with bitrate for Vidyo1 sequence (AVS-video with binDCT-L).

Table 5.47 BD-bitrate and BD-PSNR (AVS-video with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
Vidyo1	-7.45	0.41
Vidyo2	-5.28	0.26
Vidyo3	-2.14	0.13

Results for WVGA (832 x 480) sequences:

Table 5.48 Comparison of bitrates and PSNRs for three 832 x 480 sequences (AVS-video with binDCT-L).

	AVS-video with 2-D binDCT-L					Default AVS-video			
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR	V PSNR
BQMall	22	1954.78	34.83	38.96	39.76	1972.53	34.80	38.97	39.79
	27	1294.69	34.14	37.69	38.40	1306.45	33.97	37.70	38.42
	32	862.92	30.27	36.57	37.15	870.76	30.12	36.58	37.17
	37	570.66	28.45	35.62	36.06	575.84	28.30	35.62	36.08
PartyScene	22	2001.09	35.86	38.85	39.14	2019.26	35.68	38.86	39.16
	27	859.05	33.36	37.16	37.38	866.85	33.19	37.17	37.40
	32	584.86	31.04	35.76	35.96	590.17	30.88	35.76	35.98
	37	393.05	25.55	32.80	32.81	396.62	25.43	32.80	32.83
BasketballDrill	22	870.63	33.82	37.76	37.71	878.54	33.6475	37.762	37.7293
	27	751.28	32.04	36.17	36.06	758.10	31.88	36.17	36.08
	32	433.42	30.04	34.50	34.12	437.36	29.89	34.50	34.14
	37	327.98	29.02	33.48	32.89	330.96	28.87	33.48	32.91

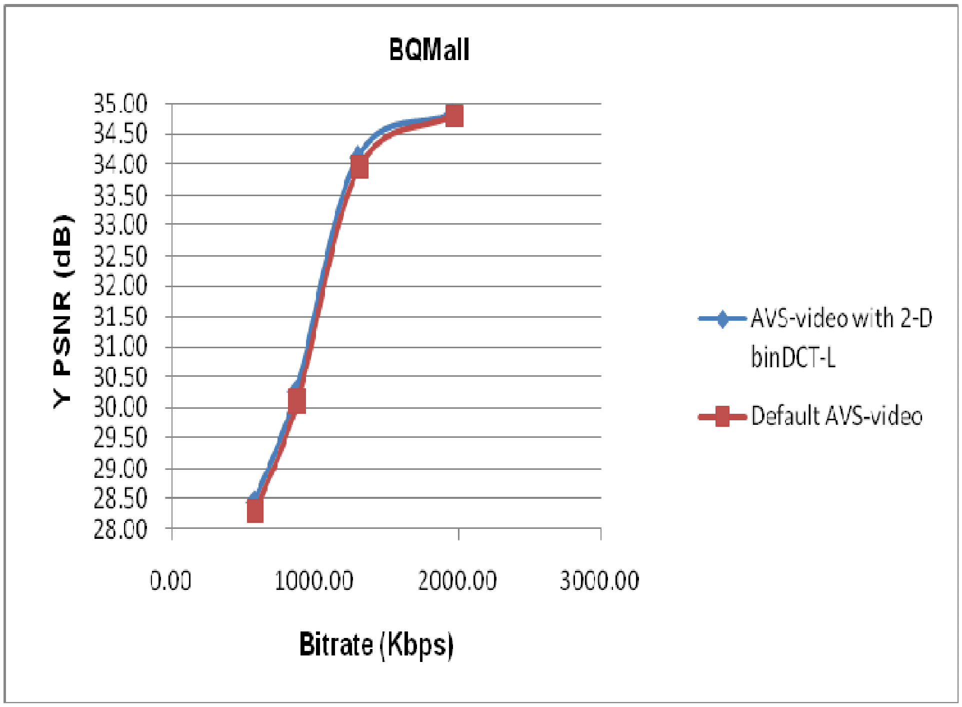


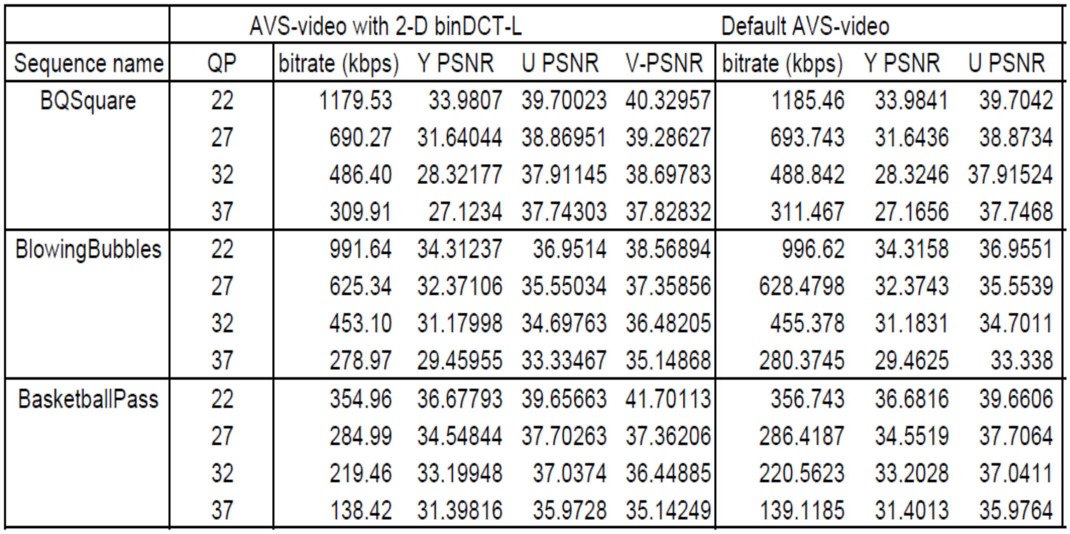
Figure 5.25 Y PSNR variations with bitrate for BQMall sequence (AVS-video with binDCT-L).

Table 5.49 BD-bitrate and BD-PSNR (AVS-video with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQMall	-0.16	0.02
PartyScene	0.35	-0.01
BasketballDrill	0.81	-0.03

Results for WQVGA (416 x 240) sequences:

Table 5.50 Comparison of bitrates and PSNRs for three 416 x 240 sequences (AVS-video with binDCT-L).



		AVS-video with 2-D binDCT-L				Default AVS-video		
Sequence name	QP	bitrate (kbps)	Y PSNR	U PSNR	V-PSNR	bitrate (kbps)	Y PSNR	U PSNR
BQSquare	22	1179.53	33.9807	39.70023	40.32957	1185.46	33.9841	39.7042
	27	690.27	31.64044	38.86951	39.28627	693.743	31.6436	38.8734
	32	486.40	28.32177	37.91145	38.69783	488.842	28.3246	37.91524
	37	309.91	27.1234	37.74303	37.82832	311.467	27.1656	37.7468
BlowingBubbles	22	991.64	34.31237	36.9514	38.56894	996.62	34.3158	36.9551
	27	625.34	32.37106	35.55034	37.35856	628.4798	32.3743	35.5539
	32	453.10	31.17998	34.69763	36.48205	455.378	31.1831	34.7011
	37	278.97	29.45955	33.33467	35.14868	280.3745	29.4625	33.338
BasketballPass	22	354.96	36.67793	39.65663	41.70113	356.743	36.6816	39.6606
	27	284.99	34.54844	37.70263	37.36206	286.4187	34.5519	37.7064
	32	219.46	33.19948	37.0374	36.44885	220.5623	33.2028	37.0411
	37	138.42	31.39816	35.9728	35.14249	139.1185	31.4013	35.9764

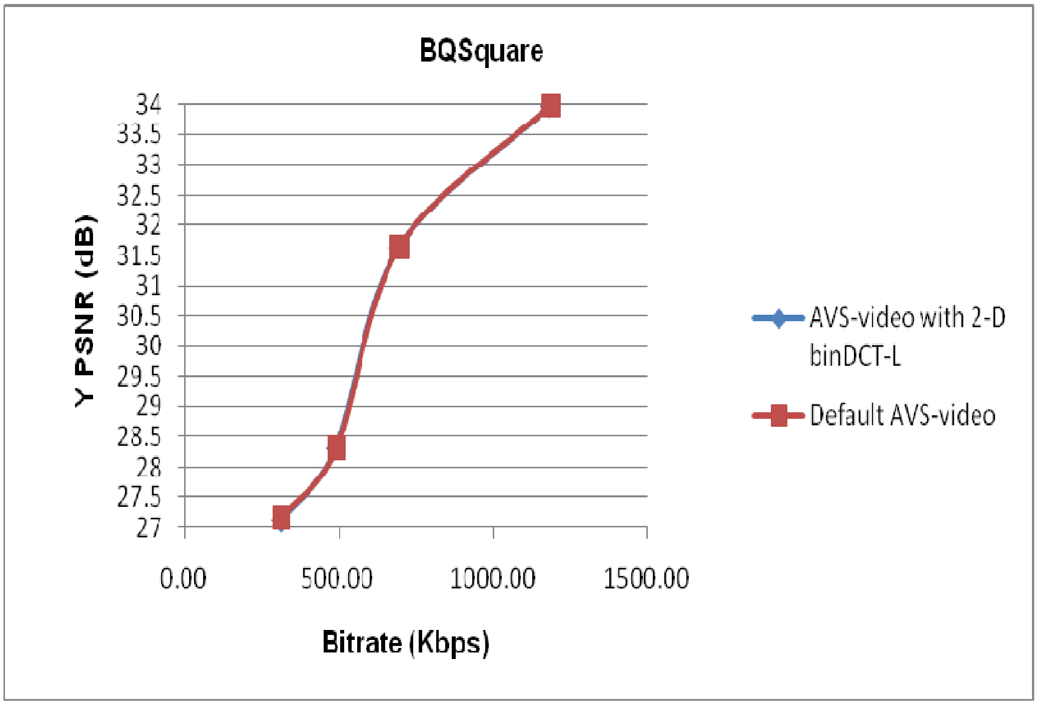


Figure 5.26 Y PSNR variations with bitrate for BQSquare sequence (AVS-video with binDCT-L).

Table 5.51 BD-bitrate and BD-PSNR (AVS-video with binDCT-L).

Sequence name	BD-bitrate (%)	BD-PSNR (dB)
BQSquare	0.16	-0.01
BlowingBubbles	0.22	-0.01
BasketballPass	0.18	-0.02

5.4 Conclusions and future work

The BD-PSNR and BD-bitrates for SICT, MICT and binDCT-L implemented in H.264/AVC show that considerable bitrate savings or PSNR gain can be achieved for HD videos (1920 x 1280, 1280 x 720) when compared to the default ICTs (2-D 8 x 8 and 2-D 4 x 4 ICT) present in the standard. A negative BD-bitrate or positive BD-PSNR denotes better coding efficiency. Moreover, the coding efficiencies of SICT, MICT and binDCT-L for lower resolution videos (WQVGA, WVGA) are similar to the default ICTs. The binDCT-L gives the best coding efficiency among the three implemented 2-D order 16 ICTs. This is due to the fact that binDCT-L is the best approximation of the 2-D order 16 DCT-II among the lot and it can be implemented using relatively few shifts and additions. From the figures in section 5.2 and 5.3 it can be deduced that the BD-PSNR and BD-bitrates for SICT, MICT and binDCT-L implemented in AVS-video follow on similar lines as that of H.264/AVC. The only difference is that for the same bitrates AVS-video shows a relatively smaller PSNR gain and the performance of SICT, MICT and binDCT-L for lower resolution videos though comparable to the default ICT (2-D 8 x 8), is inferior to that of H.264/AVC. The binDCT gives the best performance here too.

The research work dealt with 2-D order 16 ICTs only. Higher order 2-D ICTs (32 x 32, 64 x 64) can be developed and integrated into the standards similarly. These higher order ICTs can give better coding efficiency for extra high definition videos (4096 x 2560) but they will increase the computational complexity. Integration of 2-D order 32 ICT and 2-D order 64 ICT into high efficiency video coding (HEVC) [72] (the successor to H.264/AVC), is being considered by joint collaborative team on video coding (JCT-VC). The test model under consideration (TmuC 0.9) is the latest version of the test model for HEVC.

APPENDIX A

THE 16 x 16 MATRICES FOR QUANTIZATION AND DEQUANTIZATION

The 16 x 16 combined quantization and scaling matrix used for 2-D order 16 binDCT based on Loeffler et al [63] factorization is shown below.

```
{
   57,113, 57, 87, 87, 74, 80, 81, 114, 80, 80, 87, 75, 74, 114, 57

  113, 226, 113, 173, 173, 148, 161, 161, 227, 160,160, 174, 149, 148,
      227,  114

   57,      113,   57,   87,   87,   74,   80,   81,  114,   80,   80,
      87,    74,   74,  114,   57

   87,      173,   87,  133,  133,  114,  123,  123,  174,  123,  123,
      133,  114,  114,  174,   87

   87,      173,   87,  133,  132,  113,  123,  123,  174,  122,  123,
      133,  114,  114,  174,   87

   74,      148,   74,  114,  113,   97,  105,  106,  149,  105,  105,
      114,   97,   97,  149,   75

   80,      161,   80,  123,  123,  105,  114,  114,  161,  113,  114,
      123,  105,  105,  161,   81

   81,      161,   81,  123,  123,  106,  114,  115,  161,  114,  114,
      124,  106,  106,  162,   81

  114,      227,  114,  174,  174,  149,  161,  161,  227,  160,  161,
      174,  149,  149,  228,  114

   80,      160,   80,  123,  122,  105,  113,  114,  160,  113,  113,
      123,  105,  105,  160,   80

   80,      160,   80,  123,  123,  105,  114,  114,  161,  113,  114,
      123,  105,  105,  161,   81

   87,      174,   87,  133,  133,  114,  123,  124,  174,  123,  123,
      133,  114,  114,  174,   87

   75,      149,   74,  114,  114,   97,  105,  106,  149,  105,  105,
      114,   98,   97,  149,   75

   74,      148,   74,  114,  114,   97,  105,  106,  149,  105,  105,
      114,   97,   97,  149,   75

  114,      227,  114,  174,  174,  149,  161,  162,  228,  160,  161,
      174,  149,  149,  228,  114

   57,      114,   57,   87,   87,   75,   81,   81,  114,   80,   81,
      87,    75,   75,  114,   57 }
```

The corresponding16 x 16 combined dequantization and inverse scaling matrix used for 2-D order 16 binDCT based on Loeffler et al [63] factorization is shown below

```
{
  288,      144,  288,  188,  189,  220,  204,  203,  144,  204,  204,
      188,  220,  220,  144,  287

  144,       72,  144,   94,   95,  110,  102,  102,   72,  102,  102,
      94,   110,  110,   72,  144

  288,      144,  288,  188,  189,  220,  204,  203,  144,  205,  204,
      188,  220,  220,  144,  288

  188,       94,  188,  123,  123,  144,  133,  133,   94,  134,  133,
      123,  144,  144,   94,  188

  189,       95,  189,  123,  124,  144,  133,  133,   94,  134,  133,
      123,  144,  144,   94,  188

  220,      110,  220,  144,  144,  169,  156,  155,  110,  156,  156,
      144,  168,  168,  110,  220

  204,      102,  204,  133,  133,  156,  144,  143,  102,  144,  144,
      133,  155,  156,  102,  203

  203,      102,  203,  133,  133,  155,  143,  143,  101,  144,  143,
      133,  155,  155,  101,  202

  144,       72,  144,   94,   94,  110,  102,  101,   72,  102,  102,
      94,   110,  110,   72,  144

  204,      102,  205,  134,  134,  156,  144,  144,  102,  145,  145,
      134,  156,  156,  102,  204

  204,      102,  204,  133,  133,  156,  144,  143,  102,  145,  144,
      133,  155,  156,  102,  203

  188,       94,  188,  123,  123,  144,  133,  133,   94,  134,  133,
      123,  144,  144,   94,  188

  220,      110,  220,  144,  144,  168,  155,  155,  110,  156,  155,
      144,  168,  168,  110,  219

  220,      110,  220,  144,  144,  168,  156,  155,  110,  156,  156,
      144,  168,  168,  110,  220

  144,       72,  144,   94,   94,  110,  102,  101,   72,  102,  102,
      94,   110,  110,   72,  144

  287,      144,  288,  188,  188,  220,  203,  202,  144,  204,  203,
      188,  219,  220,  144,  287 }
```

APPENDIX B

SELECTED FRAMES FROM VIDEO SEQUENCES

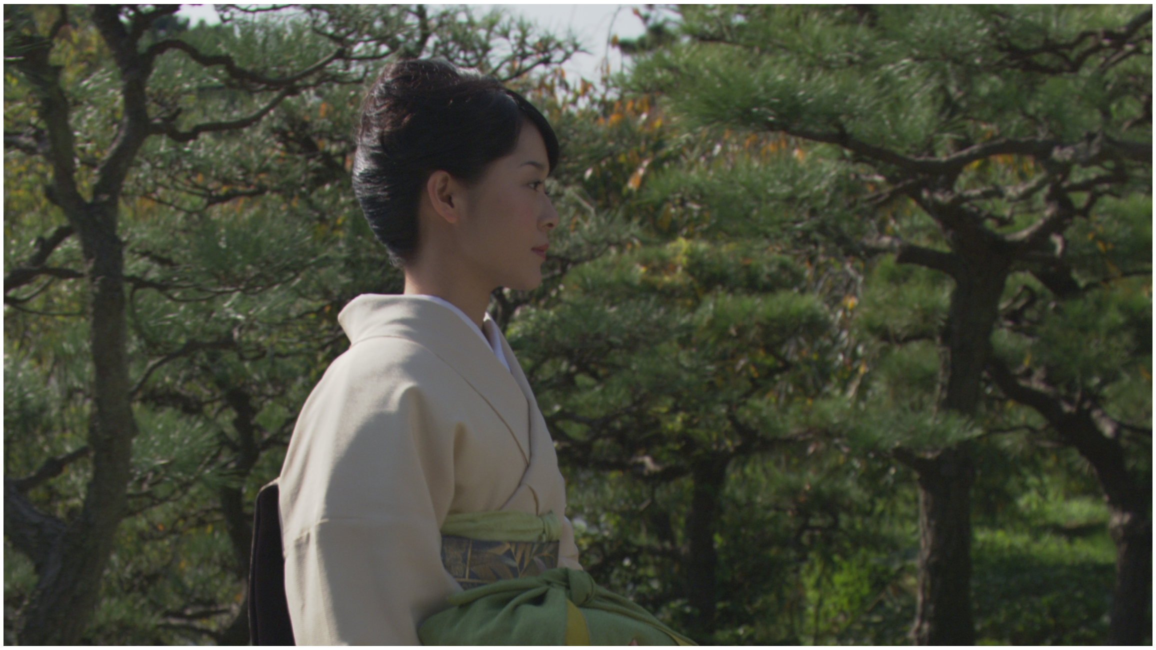


Figure B.1 First frame from Kimono (1920 x 1080)

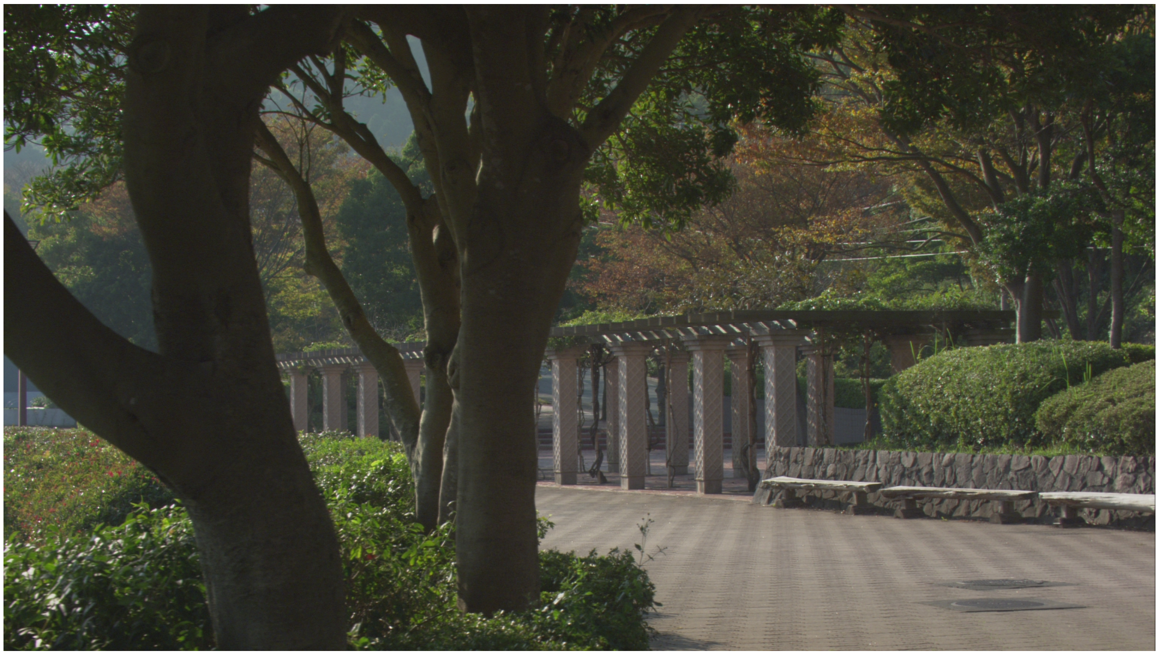


Figure B.2 First frame from Parkscene (1920 x 1080)

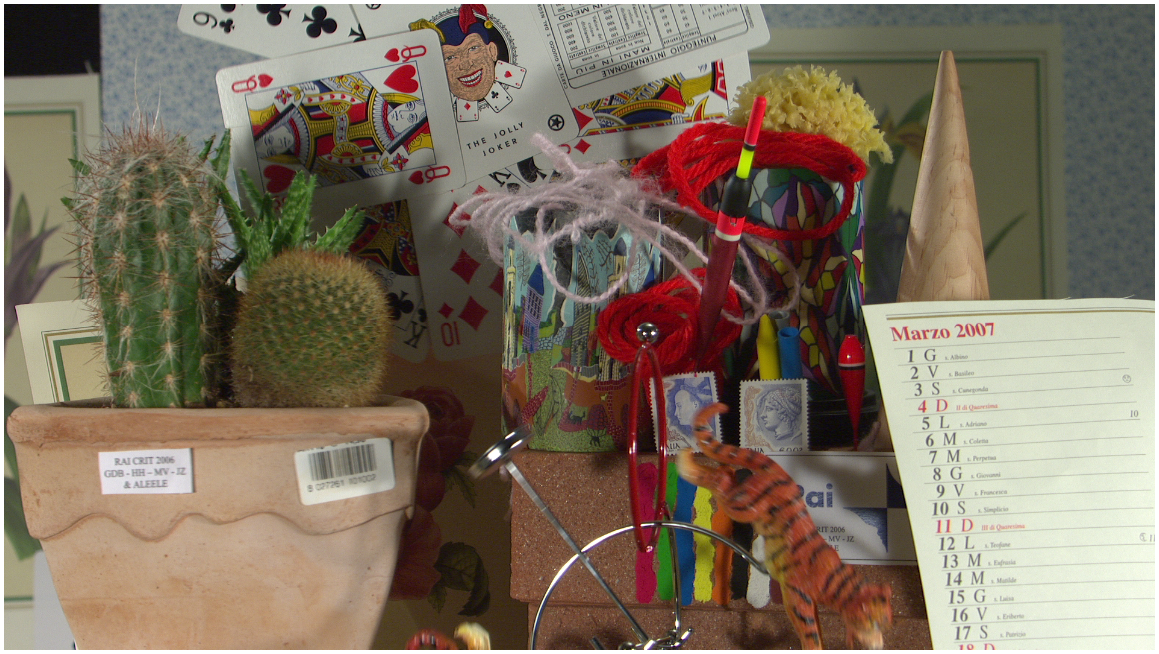


Figure B.3 First frame from Cactus (1920 x 1080)

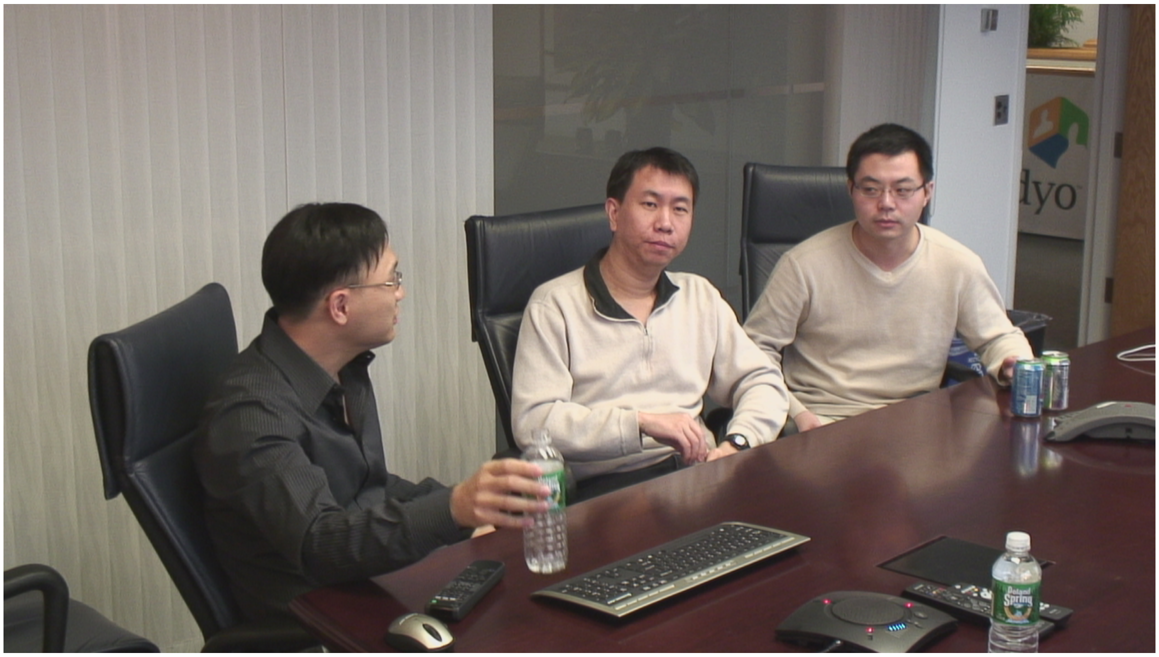


Figure B.4 First frame from Vidyo1 (1280 x 720)

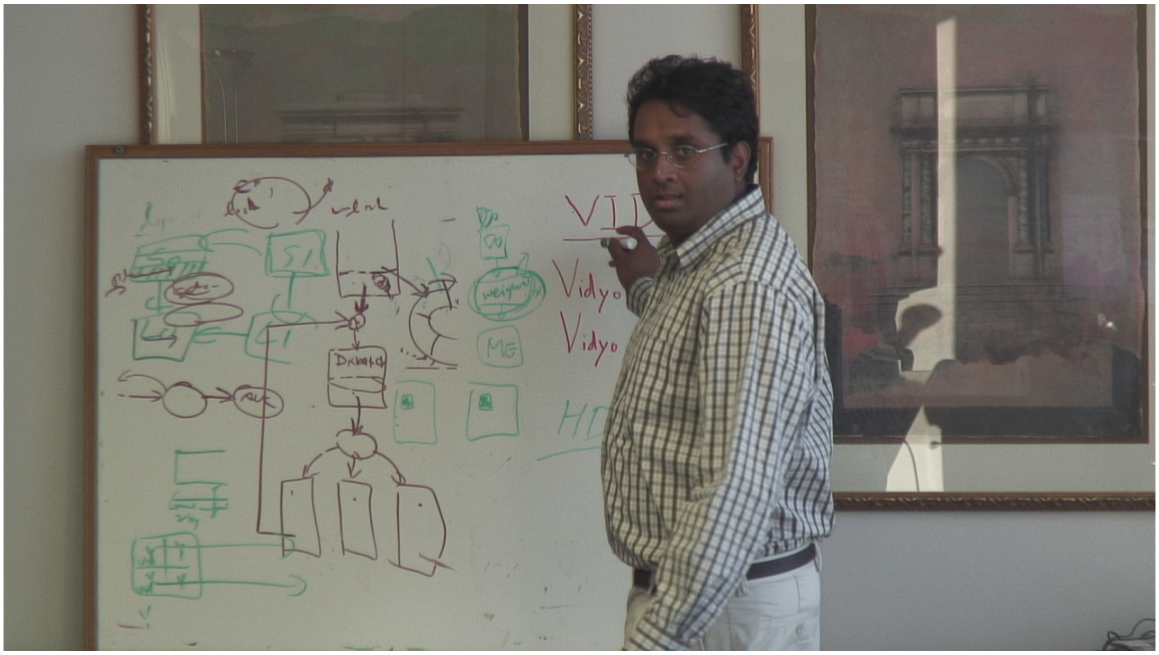


Figure B.5 First frame from Vidyo3 (1280 x 720)

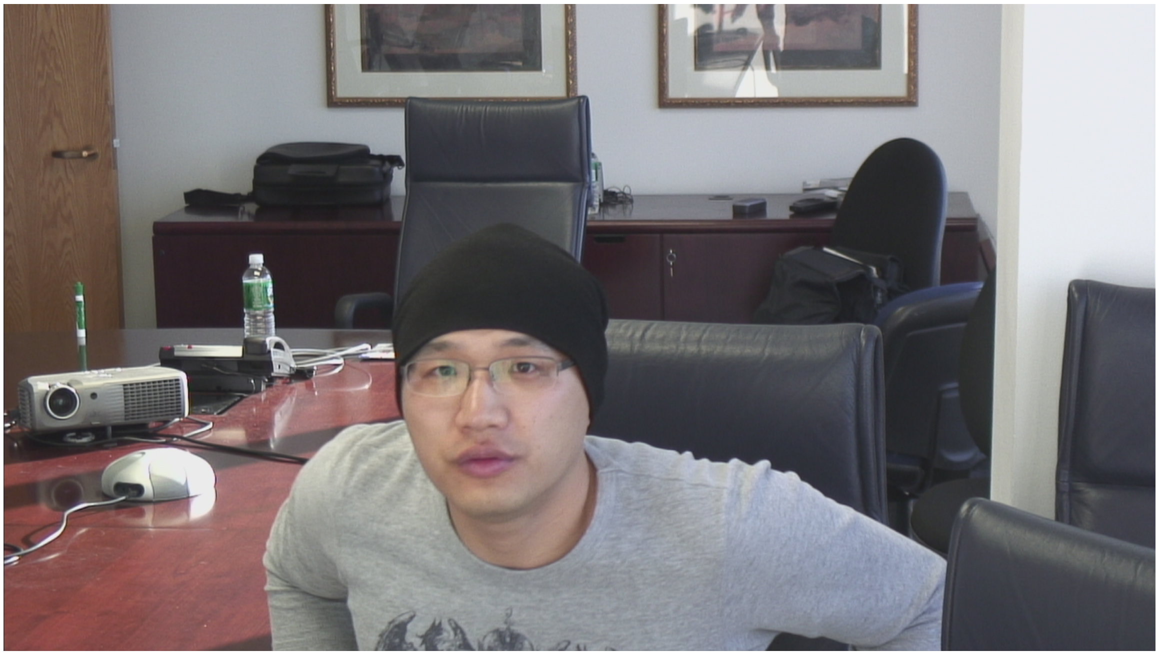


Figure B.6 First frame from Vidyo4 (1280 x 720)

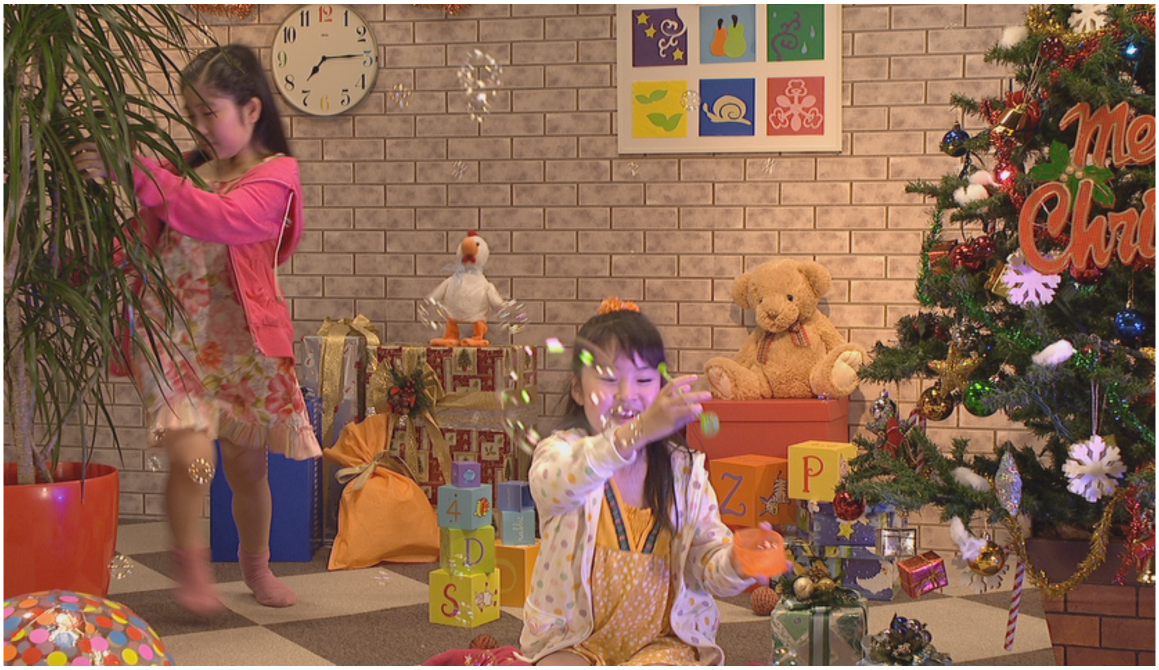


Figure B.7 First frame from PartyScene (832 x 480)

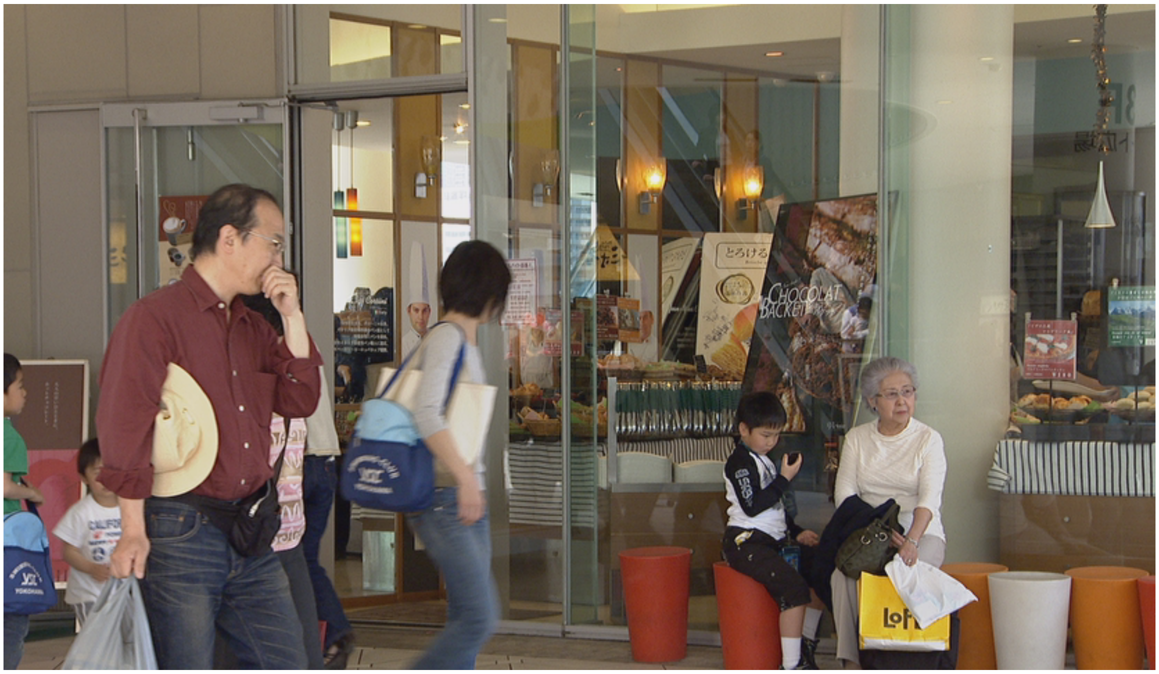


Figure B.8 First frame from BQMall (832 x 480)

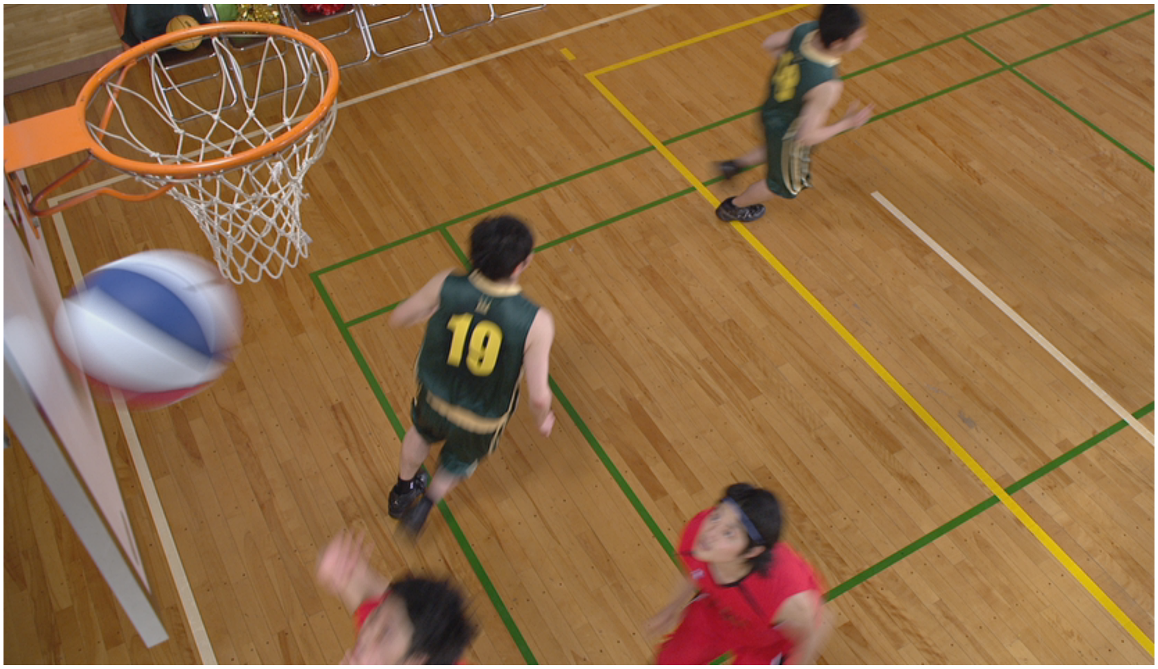


Figure B.9 First frame from BasketballDrill (832 x 480)

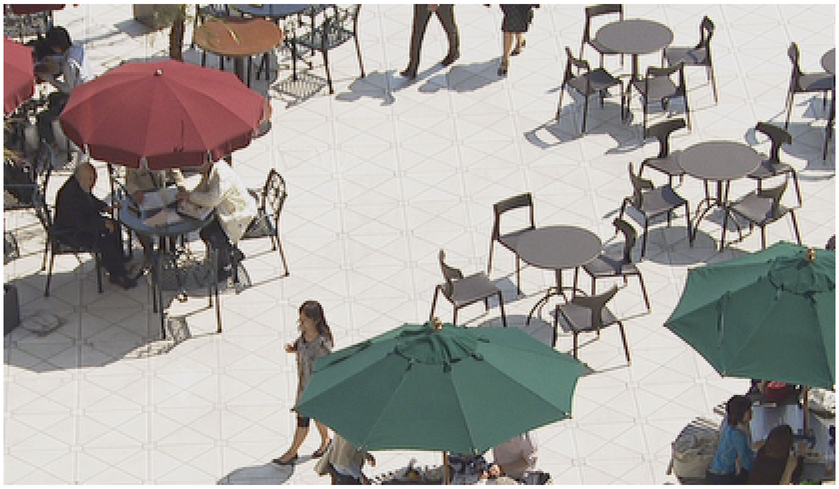


Figure B.10 First frame from BQSquare (416 x 240)

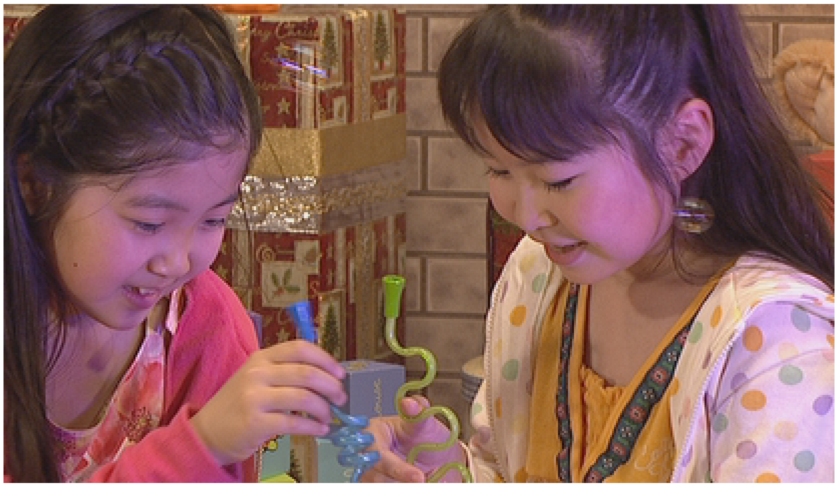


Figure B.11 First frame from BlowingBubbles (416 x 240)

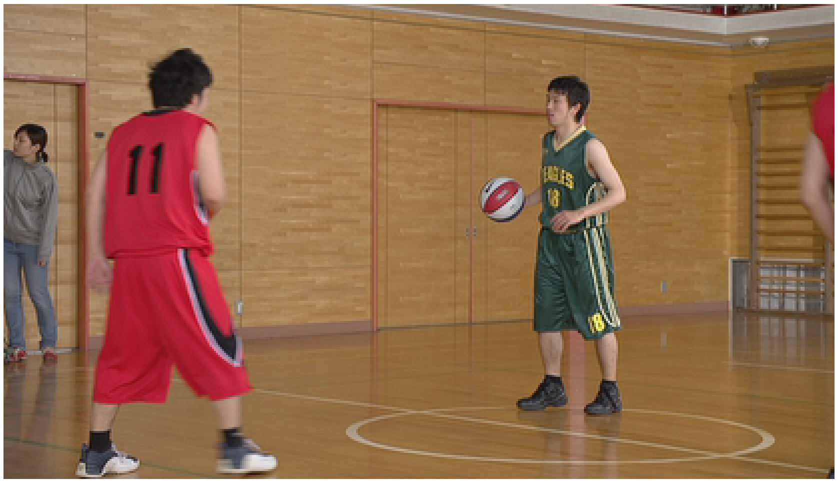


Figure B.12 First frame from BasketballPass (416 x 240)

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete cosine transform,” IEEE Trans. Comput., vol. C-23, pp. 90-93, Jan. 1974.

[2] K. R. Rao and P. Yip, “Discrete cosine transform: Algorithms, advantages, applications,” Boca Raton FL: Academic Press, 1990.

[3] G.J. Sullivan, “Standardization of IDCT approximation behavior for video compression: the history and the new MPEG-C parts 1 and 2 standards,” Proc. SPIE 6696, pp. 669611 1-22, 2007.

[4] ISO/IEC 14496-2, “Information technology – Advanced coding of audio-visual objects - Part 2: Visual”, 2001.

[5] W. K. Cham and R.J. Clarke, “Application of the principle of dyadic symmetry to the generation of orthogonal transform”, IEE Proc. F: Communications, Radar and Signal Processing, Vol. 133, No. 3, pp. 264-270, June 1986.

[6] W. K. Cham, “Development of integer cosine transforms by the principle of dyadic symmetry”, IEE Proc. I: Communications, Speech and Vision, Vol. 136, No. 4, pp. 276-282, Aug. 1989.

[7] H. S. Malvar et al, “Low-complexity transform and quantization in H.264/AVC”, IEEE Trans. Circuits and Systems for Video Technology, Vol. 13, No. 7, pp. 598-603, July 2003.

[8] W. K. Cham and Y. T. Chan, “An Order-16 integer cosine transform”, IEEE Trans. on Signal Processing, Vol. 39, No. 5, pp. 1205-1208, May 1991.

[9] G. Sullivan, P. Topiwala and A. Luthra, “The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions”, SPIE conference on Applications of Digital Image Processing XXVII, vol. 5558, pp. 53-74, Aug. 2004.

[10] AVS Video Group, “Information technology - Advanced coding of audio and video - Part 2: Video”, AVS Doc. AVS-N1317, Sept. 2006.

[11] Generic coding of moving pictures and associated audio information Part 2: Video, ITU-T Rec. H.262 and ISO/IEC 13818-2 MPEG-2, 1998.

[12] Draft ITU-T recommendation and final draft international standard of joint video specification, document JVT-G050, ITU-T Rec. H.264 and ISO/IEC 14496-10 AVC, Mar. 2003.

[13] Information Technology-Advanced coding of audio and video-Part 2: Video, GB/T 20090.2-2006 AVS, May 2006.

[14] Advanced video coding for generic audiovisual services, ITU-T Rec. H.264 and ISO/IEC 14496-10 AVC, Mar. 2005.

[15] AVS Video Group. (Sep. 2006). Information technology-Advanced coding of audio and video: Part 2. Video, AVS-N1318.

[16] J. Dong et al., "2D order-16 integer transforms for HD video coding", IEEE Trans. Circuits and Systems for Video Technology, Vol.19, No.10, pp.1462-1474, Oct. 2009.

[17] A 16-bit Architecture for H.264, treating DCT transform and quantization, document ITU-T SG16/Q6 VCEG and VCEG-M16, Apr. 2001.

[18] S. Naito and A. Koike, “Efficient coding scheme for super high definition video based on extending H.264 high profile,” in Proc. SPIE Vis. Commun. Image Process., vol. 6077, pp. 607727-1-607727-8, Jan. 2006.

[19] M. Xunan et al, “AVS adaptive block-size transform,” AVS-M2372, June 2008 Xiamen.

[20 AVS reference software: ftp://159.226.42.57/public/avs_doc/avs_software; CyaX06Pab.

[21] J. Ostermann et al, “Video coding with H.264/AVC: Tools, performance, and complexity”, IEEE Circuits and Systems Magazine, vol. 4, Issue 1, pp. 7-28, First Quarter 2004.

[22] S. Srinivasan et al, “Windows media video 9: overview and applications,” Signal Process.: Image Commun., vol. 19, pp. 851-875, Sep. 2004.

[23] ITU-T and ISO/IEC JTC 1, “Generic coding of moving pictures and associated audio information – Part 2: Video,” ITU-T Recommendation H.262 – ISO/IEC 13818-2 (MPEG-2), Nov. 1994.

[24] Joint video team of ITU-T and ISO/IEC: “Draft text of H.264/AVC fidelity range extensions amendment”, Doc. JVT-L047, Sept. 2004.

[25] D. Marpe and T. Wiegand, “H.264/MPEG4-AVC fidelity range extensions: Tools, profiles, performance, and application Areas”, Proc. IEEE ICIP, vol. 1, pp. I - 596, 11-14 Sept. 2005.

[26] H. Kalva, “The H.264 video coding standard,” IEEE Multimedia, vol. 13, no. 4, pp. 86–90, Oct. 2006.

[27] ISO/IEC 14496-10:2003, “Coding of audiovisual objects-Part 10: Advanced video coding,” 2003, also ITU-T Recommendation H.264 “Advanced video coding for generic audiovisual services.”

[28] A. Luthra, G. J. Sullivan, and T. Wiegand, Eds., Special issue on the “H.264/AVC Video Coding Standard,” IEEE Tran. CSVT, vol. 13, no. 7, pp. 148-153, July 2003.

[29] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” IEEE Trans. CSVT, vol. 13, no. 7, pp. 688-703, Jul. 2003.

[30] A. M. Tourapis et al, “Direct mode coding for bipredicitve slices in the H.264 standard”, IEEE Trans. CSVT, vol. 15, no. 1, pp. 119-126, Jan. 2005.

[31] D. Marpe, H. Schwarz, and T. Wiegand, “Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard,” IEEE Trans. CSVT, vol. 13, pp. 620–636, July 2003

[32] L. Fan, S. Ma and F. Wu, “Overview of AVS video standard,” IEEE ICME, vol. 1, pp. 423-426, June 2004.

[33] AVS Video Experts Group, “Information Technology - Advanced coding of audio and video-Part 2: Video (AVS1-P2 JQP FCD 1.0),” Audio video coding standard group of China (AVS), Doc. AVS-N1538, Sep. 2008.

[34] GB/T 20090.2, “Information Technology - Advanced coding of audio and video - Part 2: Video," 2006.

[35] L. Yu et al., “Overview of AVS-Video: Tools, performance and complexity,” SPIE VCIP, vol. 5960, pp. 596021-1~ 596021-12, July 2005.

[36] Z. Nan et al, "Spatial prediction based intra-coding," IEEE ICME, vol. 1, pp. 97-100, June 2004.

[37] “Study of ABT for HDTV coding”, document JVT-E073r2.doc, Oct. 2002, [Online]. Available: http://ftp3.itu.ch/av-arch/jvt-site

[38] K. Zhang and L. Yu, “An area-efficient VLSI architecture for AVS intra frame encoder,” SPIE VCIP 2007, vol. 6508, pp. 650822-1~ 650822-10, San Jose, USA, Jan. 2007.

[39] H. Qi et al., “A study on the motion vector prediction schemes for AVS,” SPIE VCIP 2005, vol. 5960, pp. 596066-1~596066-8, Beijing, China, July 2005.

[40] X. Ji et al, "New bi-prediction techniques for B pictures coding," IEEE ICME, vol. 1, pp. 101-104, June 2004.

[41] R. Wang et al, "Sub-pixel motion compensation interpolation filter in AVS,” IEEE ICME, vol. 1, pp. 93-96, June 2004.

[42] Y. Lu, “Adaptive search range based fast quarter-pixel motion estimation for AVS,” MIPPR 2007: Remote sensing and GIS data processing and applications, and innovative multispectral technology and applications, Proc. of SPIE, vol. 6790, pp. 67904D-1-67904D-8, Wuhan, China, Nov. 2007.

[43] C. Zhang et al, "The technique of pre-scaled integer transform," IEEE ISCS, vol. 1, pp. 316-319, May 2005.

[44] L. Wang et. al, “Hardware implementation of transform and quantization for AVS encoder,” IEEE ICALIP 2008, pp. 843-847, Shanghai, China, 2008.

[45] Q. Sun, X. Wu and L. Yu, "An adaptive coefficient scanning scheme for inter-prediction coding," Picture Coding Symposium, Nov. 2007.

[46] Q. Wang et al, "Context-based 2D-VLC for video coding," IEEE ICME, vol. 1, pp. 89-92, June 2004.

[47] L. Zhang et al, "Context-based arithmetic coding reexamined for DCT Video Compression," IEEE ISCAS, pp. 3147-3150, May 2007.

[48] Q. Yang et. al, "An efficient hardware implementation for intra prediction of AVS encoder," IEEE ICALIP 2008, pp. 200205, Shanghai, China, 2008.

[49] Z. Yang et. al., "DSP implementation of deblocking filter for AVS," IEEE ICIP, vol. 6, pp. 205–208, San Antonio, TX, Sept. / Oct. 2007.

[50] W. Gao et al., "AVS – The Chinese next-generation video coding standard," National Association of Broadcasters, Las Vegas, 2004.

[51] X. Jin, S. Li and K. Ngan, "AVS video standard implementation for SoC design," IEEE ICNNSP pp. 660–665, Zhenjiang, China, June 2008.

[52] L. Yu, S. Chen and J. Wang, "Overview of AVS-video coding standards," SP: IC, vol. 24, pp. 247-262, April 2009.

[53] W. K. Pratt, Digital Image Processing. New York: Wiley, 2001.

[54] S. Ma and C. C. Kuo, "High-definition video coding with supermacroblocks," in Proc. SPIE VCIP, vol. 6508, pp. 650816-1-650816-12, Jan. 2007.

[55] A. Papoulis and S. U. Pillai, Probability, random Variables, and stochastic processes. Boston, MA: McGraw-Hill, 2002..

[56] W. Cham and C. Fong "Simple order-16 integer transform for video coding" IEEE ICIP 2010, Hong Kong, Sept.2010

[57] J. Dong et al. "A universal approach to developing fast algorithm for simplified order-16 ICT," IEEE ISCAS, pp. 281-284, June 2007.

[58] W. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," IEEE Trans. Commun., vol. COM-25, pp. 1004-1009, Sept. 1977.

[59] W. Chen and C. H. Smith, “Adaptive coding of monochrome and color images,” IEEE Trans. Commun., vol. COM-25, pp. 1285-1292, Nov. 1997.

[60] I. Daubechies and W. A. Pearlman, “Factoring wavelet transforms into lifting steps,” J. Fourier Anal. Appl., vol. 4, pp. 247-269, 1998.

[61] J. Liang, T. Tran and P. Topiwala, “A 16-bit architecture for H.26L, treating DCT transforms and quantization,” ITU VCEG doc M16, Austin, TX, April, 2001.

[62] J. Liang and T. D. Tran, "Fast multiplierless approximations of the DCT with the lifting scheme," IEEE Trans. on Signal Processing, vol. 49, pp. 3032-3044, Dec. 2001.

[63] C. Loeffler, A. Lightenberg, and G. Moschytz, “Practical fast 1-D DCT algorithms with 11 multiplications,” Proc. IEEE ICASSP, vol. 2, pp. 988-991, Feb. 1989.

[64] F. Bruekers and A. Enden, “New networks for perfect inversion and perfect reconstruction,” IEEE J. Select. Areas Commun., vol. 10, pp. 130-137, Jan. 1992.

[65] W. Sweldens, “The lifting scheme: A custom-design construction of biorthogonal wavelets,” Appl. Comput. Harmon. Anal., vol. 3, no. 2, pp. 186-200, April 1996.

[66] G. Bjontegaard, Calculation of Average PSNR differences between RD curves, VCEG-M33, April 2001.

[67] N. S. Jayant and P. Noll, Digital coding of waveforms: principles and applications to speech and video. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[68] Link for H.264/AVC reference software: http://iphome.hhi.de/suehring/tml/download/

[69] Link for video sequences: ftp.tnt.uni-hannover.de

[70] Link for AVS reference software (RM 52e): ftp://159.226.42.57/public/avs_doc/avs_software

[71] T. Wiegand and B. Girod, “Lagrange multiplier selection in hybrid video coder control,” in Proc. IEEE Int. Conf. Image Process, vol. 3, pp. 542–545, Oct. 2001.

[72] G. J. Sullivan, J-R Ohms, Recent developments in the standardization of high efficiency video coding (HEVC), Proc. SPIE, vol. 7790, pp. 77980V-1-77980V-7, Aug. 2010.

BIOGRAPHICAL INFORMATION

Madhu Peringassery Krishnan completed his Bachelors of Technology in Applied Electronics and Instrumentation from the University of Kerala, India. After that he worked for two years as a software engineer in Verizon, India. Then he joined the University of Texas at Arlington as a graduate student of Electrical Engineering department. During the course of his graduate studies he worked under the guidance of Dr. K. R. Rao. He is currently working as an intern in FastVDO Inc.