A STUDY OF MPEG-2 AND H.264 VIDEO CODING

A Thesis

Submitted to the Faculty

of

Purdue University

by

Michael Igarta

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Electrical and Computer Engineering

December 2004

If we knew what it was we were doing, it would not be called research, would it?

Albert Einstein

ACKNOWLEDGMENTS

I would like to thank my advisory committee for supporting me in this work. I would especially like to thank Professor Edward Delp for serving as my advisor throughout graduate school. After taking his video systems class, I grew a strong interest in the field of digital video. This led to my summer project involving video compression, which was followed by my involvement in a "secret" project involving digital video, which then led to this thesis. Thank you Professor Delp for giving me all these opportunities and a chance to work on something that I truly enjoy. Professor Delp, I can truly say that working in your lab has been a once-in-a-lifetime experience. I will never forget it.

I would also like to thank everyone in the VIPER lab for their support. Yuxin, thank you for offering advice on my project as well as helping with my defense. And I cannot thank Eugene enough for everything, from his programming tips, advice on how to use Word, help with editing my thesis, and simply just being there as a friend.

Last but not least I would like to thank my family for always supporting me. I would not be where I am today without the love from my family. My parents have always stood behind me no matter what I wanted to do. And especially to Abbie and Sea, thank you very much for always being there for me. You both are my number one inspiration for me to succeed.

TABLE OF CONTENTS

Page

LIST OF TABLES vi

LIST OF FIGURES viii

ABSTRACT xiii

1 INTRODUCTION 1

1.1 The Digital Video Compression Problem 1

1.2 Digital Image Capture 3

1.3 Basic Video Color Science 4

1.3.1 The Human Visual System 4

1.3.2 Development of Color Television 5

1.3.3 Color Space Conversions 5

1.3.4 Motion Compensated Video Compression Overview 10

1.4 History of Previous Video Coding Standards 17

1.4.1 H.261 20

1.4.2 MPEG-1 20

1.4.3 MPEG-2 24

1.4.4 MPEG-4 Part 2: Visual 24

1.4.5 H.264 / MPEG-4 Part 10: Advanced Video Coding 27

2 A COMPARISON OF MPEG-2 AND H.264 VIDEO CODING 28

2.1 Transform Video Coding 28

2.1.1 Transform Coding in MPEG-2 28

2.1.2 Undesirable Properties of the DCT 32

2.1.3 Transform Coding in H.264 32

2.2 Motion Compensated Video Coding 38

Page

2.2.1 Motion Compensation in MPEG-2 38
2.2.2 Motion Compensation in H.264 43
2.2.3 In-Loop Deblocking Filter in H.264 50
2.3 Transmission of Compressed Video 51
2.3.1 Transmission of MPEG-2 Compressed Video 51
2.3.2 Transmission of H.264 Compressed Video 55
2.3.3 Error Resilient Coding Tools in H.264 57
2.4 Profiles and Applications 59
2.4.1 MPEG-2 Profiles 59
2.4.2 H.264 Profiles 59
3 EXPERIMENTAL RESULTS 63
3.1 Testing Method 63
3.1.1 Software Used 63
3.1.2 Measuring Quality of Digital Video 65
3.2 Testing Each Feature in H.264 67
3.2.1 H.264 Test Configurations 68
3.3 Experiment 1: Test of the Additional Macroblock Sizing 70
3.4 Experiment 2: Test of the H.264 In-Loop Filter 70
3.5 Experiment 3: Test of Context-Adaptive Binary Arithmetic Coding . 75
3.6 Experiment 4: Test Using Increased Number of Reference Pictures . . 75
3.7 Experiment 5: Test of Combination of Features of H.264 80
3.8 Experiment 6: MPEG-2 versus H.264 Video Compression 80
4 CONCLUSIONS 92
4.1 Suggestions for Future Work 92
APPENDICES 94
Appendix A: Input Sequences 94
Appendix B: Full Experimental Results 104
LIST OF REFERENCES 162

LIST OF TABLES

Table		Page
1.1	Typical digital video formats with their uncompressed data rates assuming 12-bits/pixel.	2
1.2	Typical transmission/storage capacities.	3
1.3	ITU-R BT.601 Digital Video Parameters.	7
1.4	Variable length codes for motion vectors in MPEG-2. The zero motion vector is the most occurring, hence, it is assigned the shortest code.	18
2.1	Intracoding feature comparsion between MPEG-2 and H.264.	33
2.2	Motion compensation comparsion between MPEG-2 and H.264.	39
2.3	Scope of Video Coding Layer and Network Abstraction Layer	55
2.4	Data partitions in H.264.	58
2.5	MPEG-2 Profiles. This chart shows supported chrominance sub-sampling, maximum resolution, maximum data rates, supported picture types, and picture rate.	60
2.6	H.264 defines three profiles designed to suit a wide variety of applications.	61
3.1	Overview of MPEG-2 configuration.	64
3.2	Overview of H.264 Configurations.	69
3.3	Each feature in H.264 contributes a small but measurable improvement. Larger numbers indicate better performance.	70
3.4	Summary of PSNR improvement for H.264 video. Results were summarized according to resolution. VQEG input sequence results were separated for comparison purposes.	83
A.1	QCIF (176×144) resolution sequences.	95
A.2	CIF (352×240) resolution sequences.	95
A.3	CIF (352×288) resolution sequences.	96
A.4	ITU-R 601 (720×480) resolution interlaced sequences.	96
A.5	High Definition (1280×720) resolution sequences.	96

Table Page

A.6 VQEG PAL (720 × 576) resolution interlaced sequences. 97
A.7 VQEG NTSC (720 × 480) resolution interlaced sequences. 97
B.1 Full results for QCIF (176 × 144) sequences. 148
B.2 Full results for QCIF (176 × 144) sequences. 149
B.3 Full results for CIF (352 × 240) sequences. 150
B.4 Full results for CIF (352 × 240) sequences. 150
B.5 Full results for CIF (352 × 288) sequences. 151
B.6 Full results for CIF (352 × 288) sequences. 152
B.7 Full results for CIF (352 × 288) sequences. 153
B.8 Full results for CIF (352 × 288) sequences. 154
B.9 Full results for (704 × 480) sequences. 154
B.10 Full results for VCEG (720 × 480) sequences. 155
B.11 Full results for VCEG (720 × 480) sequences. 156
B.12 Full results for VCEG (720 × 480) sequences. 157
B.13 Full results for VCEG (720 × 480) sequences. 158
B.14 Full results for VCEG (720 × 480) sequences. 158
B.15 Full results for (720 × 480) sequences. 159
B.16 Full results for VCEG (720 × 576) sequences. 160
B.17 Full results for VCEG (720 × 576) sequences. 161
B.18 Full results for VCEG (720 × 576) sequences. 161

LIST OF FIGURES

Figure Page

1.1 YCrCb sub-sampling formats. The 4:2:0 format is commonly used for digital video. 8

1.2 Interlaced video alternates the display of top and bottom fields. Only a single field is displayed at any given time but a complete picture is perceived by the viewer. 9

1.3 The spatial domain is the set of pixels within a single picture. The temporal domain involves multiple pictures of the video. 10

1.4 Block diagram outlining steps from video encoding to video decoding. . . 11

1.5 A video sequence consists of a hierarchy of different units, with the sample being the smallest addressable unit. 12

1.6 A picture consists of slices. Each slice consists of macroblocks. Macroblocks are 16×16 pixels. 13

1.7 Two consecutive pictures from the 'mobile' sequence. The bottom image shows the difference image between the two images. Very little change is seen between the two consecutive pictures. 15

1.8 Different types of predictive pictures. Arrows point from the reference picture to the prediction picture. 16

1.9 The scope of video coding standard only defines the decoder and not the encoder. 19

1.10 Timeline of video coding standards. MPEG-2/H.262 and MPEG-4: Part 10/H.264 were joint projects of MPEG and ISO. 20

2.1 Quantization matrices for DCT coefficients for intracoded data (top) and intercoded data (bottom). 30

2.2 The normal zig-zag scanning order, shown on the left, is used for frame-coded pictures in MPEG-2. The alternate scanning order of transform coefficient for field-coded pictures is shown on the right. 31

2.3 Different methods of spatial prediction in H.264. More prediction leads to less explicitly-coded data. 37

Figure Page

2.4 MPEG-2 interpolates half-pixel positions for more accurate motion prediction. 41

2.5 H.264 can use multiple reference pictures for prediction. Early standards were limited to a maximum of two reference pictures. 44

2.6 H.264 allows several different macroblock sizes: (a) 4×4, (b) 4×8, (c) 8×4, (d)16×8, (e) 8×16, and (f) 16×16. Each box represents a single pixel. 45

2.7 Interpolation positions of 1/4 pixels in H.264. 47

2.8 Transform scan orders in H.264. 48

2.9 Picture #8 of the CIF foreman sequence, encoded with H.264 QP=24 without and with the in-loop filtering enabled. The resulting PSNR and data rates are 38.97 dB at 806.0 kb/s and 38.93 dB at 701.6 kb/s respectively. 52

2.10 Picture #8 of the CIF foreman sequence, encoded with H.264 QP=32 without and with the in-loop filtering enabled. The resulting PSNR and data rates are 33.97 dB at 271.4 kb/s and 33.97 dB at 220.1 kb/s respectively. 53

2.11 Picture #8 of the CIF foreman sequence, encoded with H.264 QP=40 without and with the in-loop filtering enabled. The resulting PSNR and data rates are 29.46 dB at 99.4 kb/s and 29.54 dB at 78.3 kb/s respectively. 54

2.12 Decoupling of the transmission of setup and control information (the picture/GOP, and Sequence information) from the lower layers. 56

3.1 Data rate savings of additional macroblock sizing in ‘akiyo’. 71

3.2 Data rate savings of additional macroblock sizing in ‘foreman’ 72

3.3 Data rate savings of in-loop filter in ‘akiyo’. 73

3.4 Data rate savings of in-loop filter in ‘foreman’ 74

3.5 Data rate savings of CABAC in ‘akiyo’. 76

3.6 Data rate savings of CABAC in ‘foreman’. 77

3.7 Data rate savings of additional reference pictures in ‘akiyo’. 78

3.8 Data rate savings of additional reference pictures in ‘foreman’. 79

3.9 Data rate savings of full features in ‘akiyo’. 81

3.10 Data rate savings of full features in ‘foreman’. 82

Figure Page

3.11 The CIF 'carphone' sequence compressed to 246 kb/s using MPEG-2 and H.264. Severe blocking is visible in the MPEG-2 sequence. 84

3.12 The CIF 'carphone' sequence compressed to 708 kb/s using MPEG-2 and H.264. The visual difference between the two sequences is less apparent at higher data rates. 85

3.13 Overall data rate savings for QCIF (176×144) 30 fps sequences. 86

3.14 Overall data rate savings for CIF (352×240) 30 fps sequences. 87

3.15 Overall data rate savings for (352×288) 30 fps sequences. 88

3.16 Overall data rate savings for (720×480) 30 fps sequences. 89

3.17 Overall data rate savings for (720×576) 30 fps sequences. 90

3.18 Overall data rate savings for (1280×720) 25 fps sequences. 91

A.1 QCIF (176×144) sequences. 98

A.2 CIF (352×240) sequences. 99

A.3 CIF (352×288) sequences. 100

A.4 ITU-R 601 (720×576) sequences. 101

A.5 ITU-R 601 (720×480) NTSC sequences. 102

A.6 High Definition (1280×720) sequences. 103

B.1 R-D Plot for akiyo sequence QCIF 30 fps. 105

B.2 R-D Plot for carphone sequence QCIF 30 fps. 106

B.3 R-D Plot for claire sequence QCIF 30 fps. 107

B.4 R-D Plot for coastguard sequence QCIF 30 fps. 108

B.5 R-D Plot for container sequence QCIF 30 fps. 109

B.6 R-D Plot for glasgow sequence QCIF 30 fps. 110

B.7 R-D Plot for mthr-dgthr sequence QCIF 30 fps. 111

B.8 R-D Plot for news sequence QCIF 30 fps. 112

B.9 R-D Plot for salesman sequence QCIF 30 fps. 113

B.10 R-D Plot for flowergarden sequence CIF 30 fps. 114

B.11 R-D Plot for foot sequence CIF 30 fps. 115

B.12 R-D Plot for sequence news2 CIF 30 fps. 116

Figure Page

B.13 R-D Plot for sequence salescut CIF 30 fps. 117
B.14 R-D Plot for sequence tennis CIF 30 fps. 118
B.15 R-D Plot for sequence tv18_new CIF 30 fps. 119
B.16 R-D Plot for sequence akiyo CIF 30 fps. 120
B.17 R-D Plot for sequence bus150 CIF 30 fps. 121
B.18 R-D Plot for sequence carphone CIF 30 fps. 122
B.19 R-D Plot for sequence foreman CIF 30 fps. 123
B.20 R-D Plot for sequence missA CIF 30 fps. 124
B.21 R-D Plot for sequence mobile CIF 30 fps. 125
B.22 R-D Plot for sequence news CIF 30 fps. 126
B.23 R-D Plot for src2 sequence 720×576 interlaced 30 fps. 127
B.24 R-D Plot for src3 sequence 720×576 interlaced 30 fps. 128
B.25 R-D Plot for src4 sequence 720×576 interlaced 30 fps. 129
B.26 R-D Plot for src5 sequence 720×576 interlaced 30 fps. 130
B.27 R-D Plot for src6 sequence 720×576 interlaced 30 fps. 131
B.28 R-D Plot for src7 sequence 720×576 interlaced 30 fps. 132
B.29 R-D Plot for src8 sequence 720×576 interlaced 30 fps. 133
B.30 R-D Plot for src9 sequence 720×576 interlaced 30 fps. 134
B.31 R-D Plot for src13 sequence 720×480 interlaced 30 fps. 135
B.32 R-D Plot for src14 sequence 720×480 interlaced 30 fps. 136
B.33 R-D Plot for src15 sequence 720×480 interlaced 30 fps. 137
B.34 R-D Plot for src16 sequence 720×480 interlaced 30 fps. 138
B.35 R-D Plot for src17 sequence 720×480 interlaced 30 fps. 139
B.36 R-D Plot for src18 sequence 720×480 interlaced 30 fps. 140
B.37 R-D Plot for src19 sequence 720×480 interlaced 30 fps. 141
B.38 R-D Plot for src20 sequence 720×480 interlaced 30 fps. 142
B.39 R-D Plot for src21 sequence 720×480 interlaced 30 fps. 143
B.40 R-D Plot for src22 sequence 720×480 interlaced 30 fps. 144

Figure Page

B.41 R-D Plot for shields sequence 1280 × 720 25 fps. 145
B.42 R-D Plot for parkrun sequence 1280 × 720 25 fps. 146
B.43 R-D Plot for mobcal sequence 1280 × 720 25 fps. 147

ABSTRACT

Igarta, Michael. M.S.E.C.E., Purdue University, December, 2004. A Study of MPEG-2 and H.264 Video Coding. Major Professor: Edward J. Delp.

Digital video is present in many applications, including video conferencing, video game console entertainment, DVDs, and broadcast television. Since digital video requires a prohibitive amount of storage in uncompressed form, lossy digital video compression is commonly employed as a compromise between data rate and quality. One of the most prevalent video compression standards is MPEG-2, found in DVD video. Recent advances in digital video coding tools have led to the introduction of the recent H.264 video coding standard, which promises increased visual quality and reduced bandwidth. In this thesis, we analyze and compare MPEG-2 and H.264 video compression. Although H.264 is similar to MPEG-2 in that both are hybrid coders that use motion compensation, H.264 also includes advanced features such as improved entropy encoding, in-loop filtering of reference frames, flexible macroblock sizing, and multiple reference frame capability. Many experiments were performed to illustrate the coding gains of each feature in H.264 as well as to compare H.264 to MPEG-2 video. Quality was measured using two different objective video metrics: peak signal-to-noise ratio and Structural Similarity Index. A variety of natural video test sequences were used with varying resolutions and data rates. TM5 and JM reference software were used to create MPEG-2 and H.264 compressed bitstreams. Results for all experiments show significant coding gain with H.264 versus MPEG-2 when compressing natural video sequences, especially at low data rates.

1. INTRODUCTION

Digital video [1] can be found in familiar applications, such as the Digital Versatile Disc (DVD) [2], digital television, Internet video streaming, and digital high definition television [3]. Digital video shares all the features of other digital formats, including lossless transmission, lossless storage, and ease of editing.

To ease storage / transmission requirements, compression is commonly performed on the video. Modern lossy compression techniques allow tremendous storage savings with little visible degradation. The MPEG-2 video compression standard [4, 5] has allowed the success of DVD-video and digital high definition television. New advancements in digital video compression technology have led to the recently finalized H.264 video compression standard [6], which is poised to follow the success of the highly accomplished MPEG-2 standard.

The goal of this thesis is to compare the two standards and highlight the differences between the two standards. Although both follow the same general framework, there are several fundamental key advancements in the H.264 standard including a new integer transform, advanced arithmetic entropy coding, and the inclusion of an in-loop filter. The first part of the thesis will provide an introduction to some basics of video compression. The next part will go into more detail of the technical differences between the MPEG-2 and H.264 video coding standards. Finally, we will discuss some experiments performed to illustrate the compression gains offered by H.264 over MPEG-2.

1.1 The Digital Video Compression Problem

Compared with analog video, digital video has many advantages:

Table 1.1
Typical digital video formats with their uncompressed data rates assuming 12-bits/pixel.

Format	Spatial Resolution	Frames/Second	Uncompressed Data Rate (bits/s)
QCIF	176×144	15	4 561 920
CIF	352×240	30	30 412 800
ITU-R 601	720×480	30	124 416 000
HDTV	1280×720	30	331 776 000
HDTV	1920×1080	30	746 496 000

- A digital format, such as DVD-video, allows lossless copies to be created without the inherent generation loss associated with analog formats, such as VHS tape. For example, a duplicate copy of a VHS tape will not be an exact reproduction of the original. On the other hand, a duplicate copy of a DVD will result in a exact reproduction of the original.

- Digital video formats lend themselves more easily for transmission, especially over error-prone, or lossy, channels. Traditional error prevention and recovery techniques, such as error-correcting codes [7] can be employed.

- Digital video editing is easier to perform than analog video editing.

One of the major drawbacks of a digital representation of video is storage and transmission requirements. Uncompressed digital video can require large bandwidths, especially as spatial and temporal resolution increase. Table 1.1 shows examples of the required data rate of several video formats. Capacities of typical storage and transmission formats are listed in Table 1.2.

Table 1.2
Typical transmission/storage capacities.

Format	Maximum Capacity
10/100 Ethernet	10/100 Mb/s
DSL	768kb/s-1.5Mb/s downstream
	128kb/s upstream (typical)
V.90 modem	56 kb/s
Single Layer DVD-5	4.7 Gbytes
Dual Layer DVD-9	8.5 Gbytes

1.2 Digital Image Capture

When a camera captures light from the physical world and transforms physical data into digital data, it must perform two main steps in the digital to analog conversion of the signal: sampling and quantization. Light in the physical world can assume an infinite, or uncountable, number of values. Sampling reduces the continuous signal into a countable number of discrete samples, or pixels. Quantization maps each of these samples into a finite set of pixel values.

Let $x_{ct}(t)$ represent an arbitrary, real-valued function with the continuous time parameter t. The function $x_{ct}(t)$ can be sampled to create $x_{dt}[n]$, which is a function of the discrete time parameter n [8]:

$$x_{dt}[n] = \sum_{s=\infty}^{\infty} x_{ct}(n - Ts). \tag{1.1}$$

The real-valued function $x_{dt}[n]$ can be quantized to be a integer-valued function $\hat{x}[n]$ with finite possible values by

$$\hat{x}[n] = \alpha \times round\left(\frac{x_{dt}[n]}{\beta}\right), \tag{1.2}$$

where β determines the quantization levels and α is an scaling factor.

Since some of the signal is discarded in the sampling and quantization process, the perceived fidelity of the resulting digital image depends on both steps. Increased fidelity can be achieved by higher sampling rates (more samples) or a larger set of allowed pixel values.

Video is created by the rapid display of successive images, or pictures. If the pictures are displayed at a sufficient rate, they create the illusion of motion. The time between the display of subsequent pictures must be sufficiently small to avoid visible flashing or flicker. Broadcast television adhering to the National Television System Committee (NTSC) standard refreshes at a rate of 30 pictures per second.

1.3 Basic Video Color Science

1.3.1 The Human Visual System

The development of color video systems has been tailored to suit the characteristics of the human visual system, or HVS [9]. As light enters the human eye, it excites special cells known as photoreceptors. Photoreceptors consist of two main types: rods and cones. Generally speaking, rods are responsible for sensing light and cones are responsible for sensing color [10].

The sensation of color is produced by different distributions of light of varying wavelengths. The HVS is most sensitive to green light, less sensitive to red light, and least sensitive to blue light.

A camera captures color light and separates it into its red, green, and blue primary components, or RGB components. Output light intensity of a display device, e.g. a television, is non-linearly related to its input, generalized by

$$B = cv^{\gamma} + b, \tag{1.3}$$

where B is the intensity, c is a gain factor, gamma is the non-linear factor, and b is an arbitrary offset [5]. To preserve linearity of the original RGB components and

avoid correction systems in receivers, the color components are raised to the power $1/\gamma$ before transmission.

1.3.2 Development of Color Television

During the development of the National Television Systems Committee, or NTSC, standard for analog broadcast color television, one of the major requirements was backwards compatibility with older monochrome television [11]. A solution was devised that would transmit additional color, or chrominance information in addition to the already present monochrome, or luminance information. A signal with the luminance and chrominance signals multiplexed together is referred to as a composite signal. In a composite system, an older monochrome receiver could accept the composite signal and process only the luminance signal while a color-capable receiver would be able to process both the luminance and chrominance information.

1.3.3 Color Space Conversions

To avoid cross-talk between the two signals for maximum fidelity, it is advantageous to separate the luminance and chrominance signals during transmission [12]. Furthermore, the chrominance signals can be represented by two separate color components. The RGB color components can be converted into a luminance/chrominance representation, or color space, using a linear transform defined by

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} \mathbf{A_{11}} & \mathbf{A_{12}} & \mathbf{A_{13}} \\ \mathbf{A_{21}} & \mathbf{A_{22}} & \mathbf{A_{13}} \\ \mathbf{A_{31}} & \mathbf{A_{32}} & \mathbf{A_{13}} \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}, \tag{1.4}$$

where $R'G'B'$ represent gamma corrected RGB components and YUV represents the luminance/chrominance components.

Digital Component Video

For digital video, the ITU-R BT.601 standard [13] defines a digital encoding format denoted $YCrCb$, which is derived from the analog YUV color space. Each sample of the $YCrCb$ components are generated from the $R'G'B'$ components using the linear transform

$$\begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ 0.439 & -0.368 & -0.071 \\ -0.148 & -0.291 & 0.439 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}, \tag{1.5}$$

where each component is represented with 8-bit ($[0 \ldots 255]$) precision with the following dynamic ranges:

$$\begin{aligned} R', G', B' &\in 0 \ldots 255 \\ Y &\in 16 \ldots 235 \\ Cr, Cb &\in 16 \ldots 240. \end{aligned}$$

Due to rounding, the conversion from RGB to $YCrCb$ back to RGB is not lossless. A detailed investigation of error introduced in the conversion is given in [14].

Other parameters for NTSC and PAL are defined in [13] are shown in Table 1.3. PAL, or Phase Alternate Line, is an analog television standard used throughout Europe.

Chrominance Sub-Sampling

Spatial resolution is defined by the dimensions of the luminance component. In order to preserve bandwidth, the chrominance channels are normally subsampled. This subsampling produces minimal visual impact due to the decreased HVS sensitivity to the chrominance channels relative to the luminance channel [12].

Table 1.3
ITU-R BT.601 Digital Video Parameters.

	NTSC	PAL
Pictures per second	30	25
Fields per second	60	50
Lines per picture	525	625
Luminance samples/line	858	864
Chrominance samples/line	429	432
Bits/sample	8	8
Total data rate	216 Mbit/sec	216 Mbit/sec
Actives lines per picture	480	576
Active samples per line (Y)	720	720
Active samples per line (Cr,Cb)	360	360

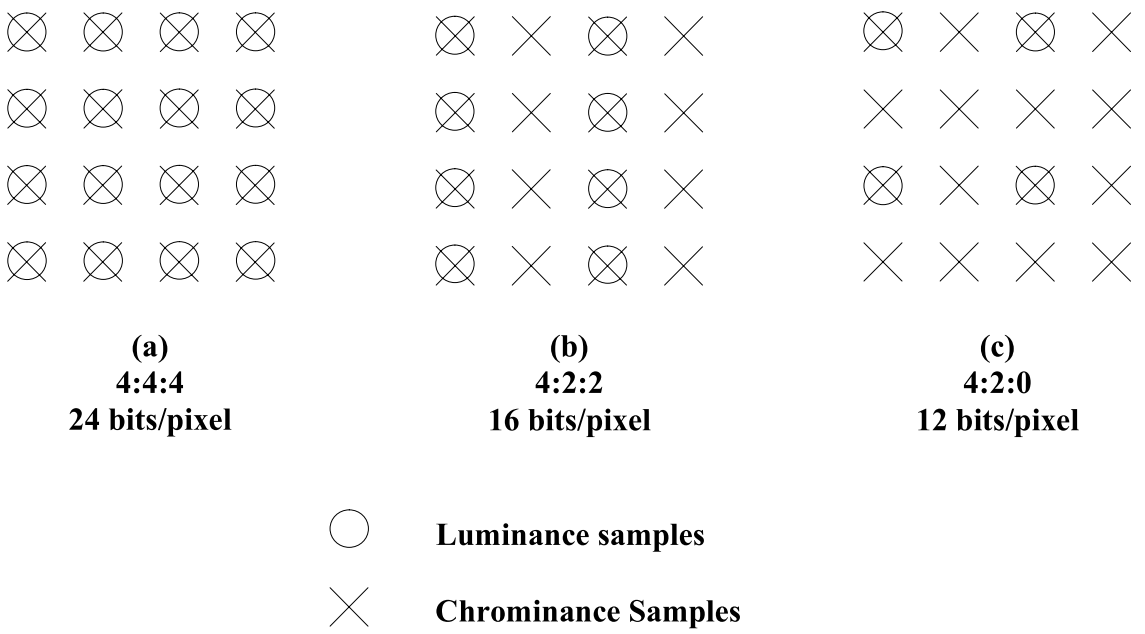


Fig. 1.1. YCrCb sub-sampling formats. The 4:2:0 format is commonly used for digital video.

Digital color video normally has three components. If there are exactly 3 digital samples for each spatial location in the image, the components are said to be 4:4:4 encoded. In 4:2:2 encoding, the chrominance components are sub-sampled 2:1 in the horizontal direction. In 4:2:0 encoding, the chrominance components are sub-sampled 2:1 in both the horizontal and vertical directions. While 4:2:0 encoding offers the least visual fidelity due to the least number of samples, this level of fidelity is sufficient for most consumer applications and is the default encoding in the Main Profile for MPEG-2 video (the profile utilized in consumer DVDs). 4:2:2 encoding offers increased fidelity and is more suited for professional applications, such as the Studio Profile for MPEG-2 video. A figure comparing the different subsampling formats is shown in Figure 1.1 [15].

Frame and Field Pictures

A video can be represented as a ordered sequence of either frames or fields. In progressive video, each picture is represented as a single frame. A frame consists of the set of all horizontal lines of the picture. In interlaced video, each picture is represented as two separate fields. Each field consists of the set of alternating lines as shown in Figure 1.2. A single field is updated every sampling period. Since only half of a picture is updated at a time, it is possible to display field-coded video at twice the temporal frequency of an equivalent progressive video. This gives the perception of smoother motion. Analog broadcast television (NTSC/PAL) are natively interlaced video formats.

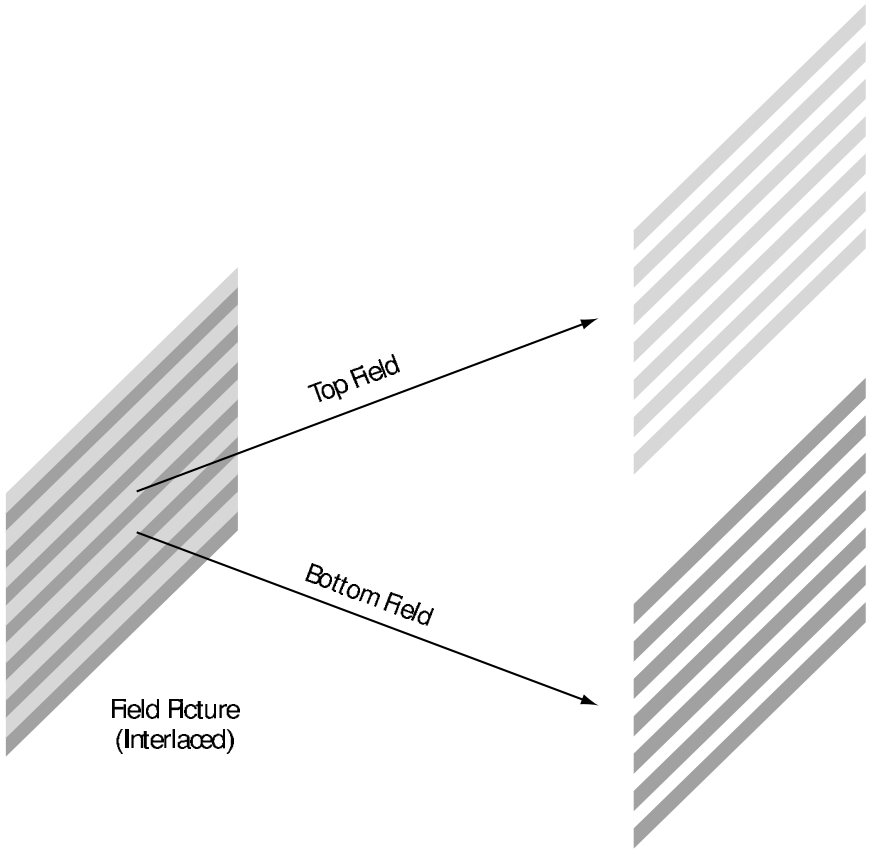


Fig. 1.2. Interlaced video alternates the display of top and bottom fields. Only a single field is displayed at any given time but a complete picture is perceived by the viewer.

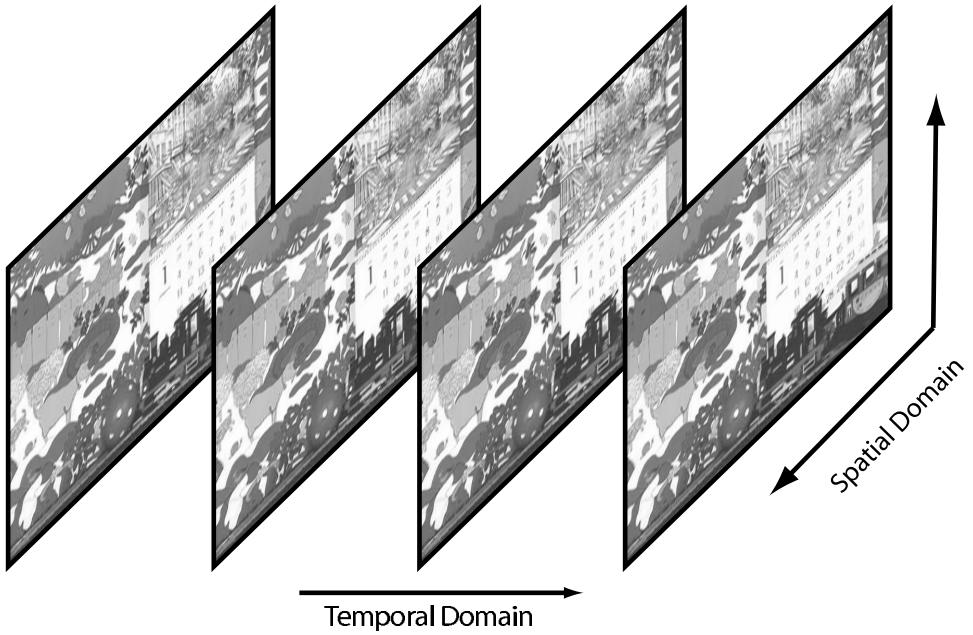


Fig. 1.3. The spatial domain is the set of pixels within a single picture. The temporal domain involves multiple pictures of the video.

1.3.4 Motion Compensated Video Compression Overview

The digital video compression techniques described here are all lossy compression techniques, i.e. the compressed video is not an exact reproduction of the original video. Digital video compression exploits data redundancy in the spatial domain, also known as intracoding, and the temporal domain, which is known as intercoding. The spatial domain is the set of pixels within a single picture. The temporal domain involves multiple pictures of the video, as shown in Figure 1.3. Motion compensation uses the redundancy in consecutive pictures to efficiently code the current picture, which is the main difference between video coding, such as MPEG, and image coding, such as JPEG [16].

A generalized block diagram in Figure 1.4 outlines the complete process of digital video compression, from input, encoding, decoding, and output stages. The first three blocks represent the encoding process. The bottom four blocks represent the video decoding process. It is important to note that post processing is considered

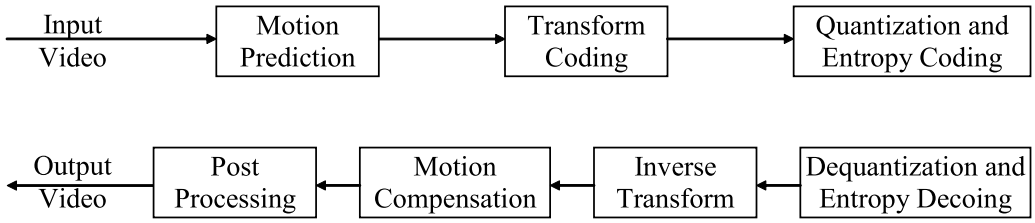


Fig. 1.4. Block diagram outlining steps from video encoding to video decoding.

an optional process and is not defined in the video coding standards described below. Intracoding (transform coding), intercoding (motion prediction), and entropy coding are further detailed below. Throughout this thesis, references will be made to different types of objects encapsulated in a digital video sequence. A hierarchy of these objects is represented in Figure 1.5. The video sequence represents the highest level. Each sequence is comprised of a sequence of pictures (or frames). A continuous subset of pictures is a group of pictures (GOP). Each picture consists of continuous partitions known as slices. Each slice consists of a continuous run of macroblocks. Each macroblock can be divided into sub-macroblocks. Each sub-macroblock consists of blocks (8×8 or 4×4 pixels). Each block consists of samples, or pixels, which is the smallest unit in a video sequence. A stylized figure of a picture in a video sequence is shown in Figure 1.6.

Intracoding

While intercoded pictures are the basis of most of the compression in video coding, intracoded pictures are not dependent upon other pictures. They are the first picture type in a sequence and can aid in encoding scene changes and seeking forwards and backwards. The process of intracoding involves the use of a decorrelating transform, such as the Discrete Cosine Transform (DCT) [17], and quantization of the transform coefficients. The Discrete Cosine Transform will be discussed in more detail in the following chapter.

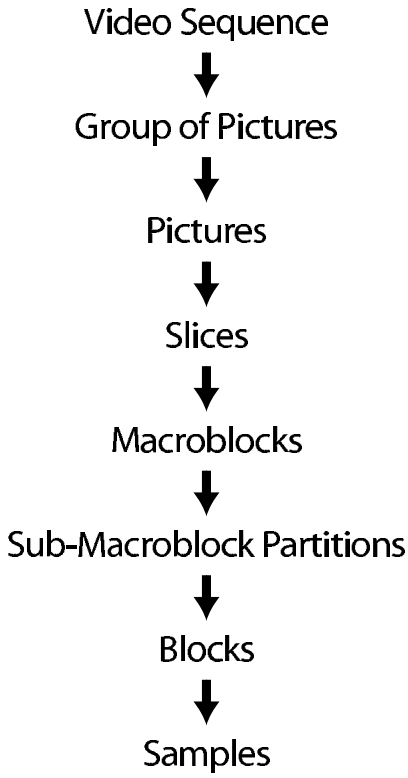


Fig. 1.5. A video sequence consists of a hierarchy of different units, with the sample being the smallest addressable unit.

The quantization of the coefficients maps the input into discrete values, typically uniformly distributed with a wider central "dead zone." The extra wide dead zone allows more coefficients to be quantized to zero. Using a zig-zag scan pattern while encoding the transform coefficients results in long runs of zeros, which can be efficiently run-length encoded. After quantization, the entropy coder converts the quantized transform coefficients into an efficient binary representation.

Intercoding

Given two successive pictures in a video sequence, it is assumed that the pictures are correlated from one picture to the next picture. Instead of coding an entire new picture, intercoding exploits this temporal redundancy by forming a prediction of

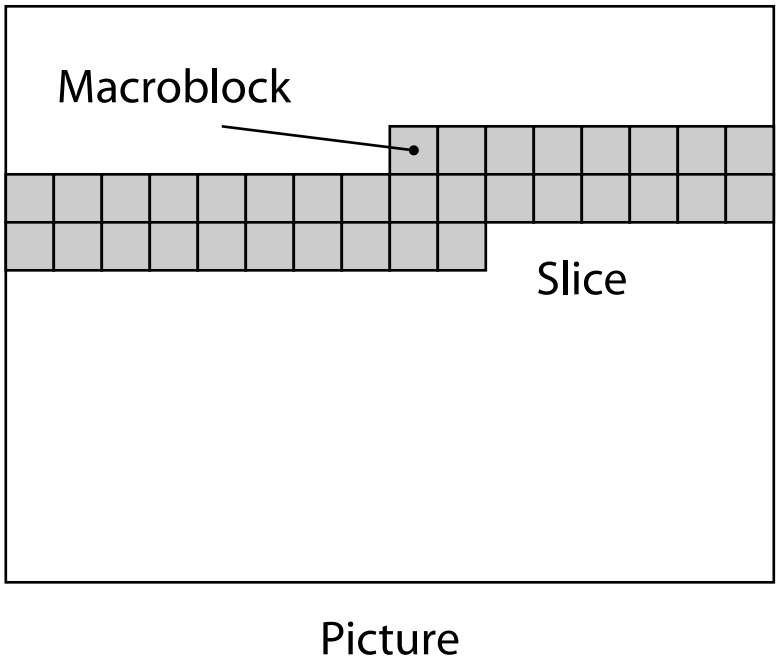


Fig. 1.6. A picture consists of slices. Each slice consists of macroblocks. Macroblocks are 16×16 pixels.

the picture using a picture than has already been coded (reference picture). First, the current picture being coded is partitioned into macroblocks. Then the reference picture is used to create a prediction of the current picture. The encoder searches for the best macroblocks in a previously transmitted picture to create an estimate of the current picture. The encoder can use prediction error as a metric for determining the best macroblocks. Common measures of prediction error include minimum mean square error (MSE) and minimum sum absolute difference (SAD). Given two discrete 2-D signals x_{ij} and y_{ij} of size $M \times N$, the $SAD(x, y)$ is defined as

$$SAD(x, y) = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} = |x_{ij} - y_{ij}| \,. \tag{1.6}$$

The MSE between two signals x and y is defined as

$$MSE(x, y) = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (x_{ij} - y_{ij})^2. \tag{1.7}$$

The encoder associates a motion vector with each macroblock to describe the spatial location of each macroblock in the reference picture relative to the current picture. The process of creating the best prediction for the current picture is known as motion estimation. Little changes from one picture to the next as can be seen in Figure 1.7.

The use of only the previously displayed picture is known as forward prediction. The use of both a previous picture and a future picture is known as bi-directional prediction. Both types of predictive pictures are illustrated in Figure 1.8.

Once a motion compensated prediction picture is created for the current picture, the residual between motion-only prediction and the actual picture is then encoded using intracoding. For each intercoded picture, the motion vectors and the intra-coded residual signal need to be transmitted.

Entropy Coding

In order to efficiently represent each element, e.g. transform coefficient, of the compressed video, each element is represented as a symbol, or a string of bits. It is

Fig. 1.7. Two consecutive pictures from the 'mobile' sequence. The bottom image shows the difference image between the two images. Very little change is seen between the two consecutive pictures.

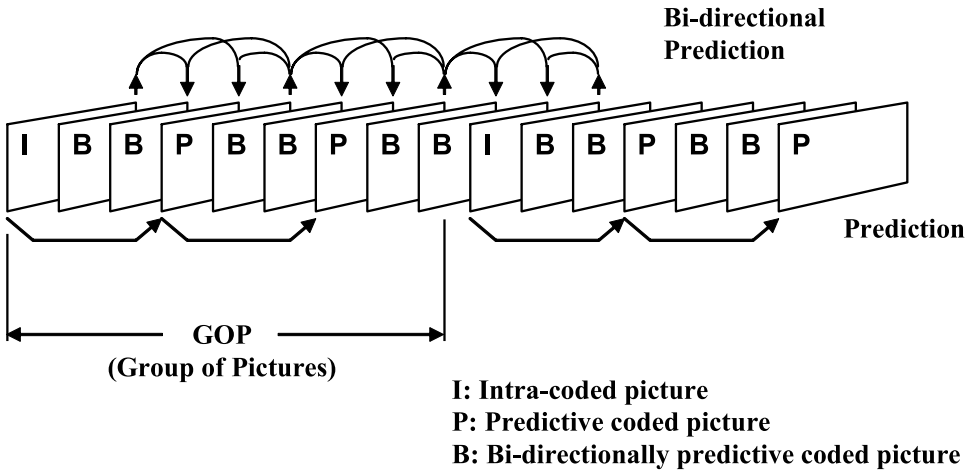


Fig. 1.8. Different types of predictive pictures. Arrows point from the reference picture to the prediction picture.

possible to perform lossless data compression due to redundancy in the data. Since it is known a priori that some elements in the coded video, or syntax elements, occur more often than others, it is more efficient to to use a shorter symbol for elements with a high probability and use a longer symbol for elements with a lower probability. Shannon showed a lower bound for the average codeword length of over many coded elements [18]. A well-known method for constructing a code that approached this lower bound was decribed by Huffman [19]. Since each symbol in such codes has a different length, they belong to a class of codes known as variable length codes (VLC). Another type of VLC is an Exponential Golomb code [20], which is used extensively in H.264.

For illustration, a portion of the motion vector code table in the MPEG-2 standard is shown in Table 1.4. It is assumed that smaller motion displacements are more prevalent than larger motion displacements. Hence, smaller motion displacements are assigned shorter codewords. Zero motion would apply to any static object in a video sequence, such as a stationary background.

Another class of efficient codes is arithmetic codes [21]. Arithmetic codes can more closely approach Shannon's lower bound by assigning a non-integer number of symbols to each codeword, which is not possible with a VLC.

1.4 History of Previous Video Coding Standards

The purpose of this section is to provide a short description of the various video compression standards. Standards allow interoperability between different manufacturers and a variety of equipment worldwide [22]. Two standards bodies have strongly influenced the development of video compression standards: the International Telecommunications Union (ITU) and the Moving Pictures Experts Group (MPEG). The oldest body is the ITU [23]. Of particular interest is the Telecommunication Standardization Sector, or ITU-T sector. Their goals have been transmitting video over both the analog and digital telephone system although in recent years the

Table 1.4
Variable length codes for motion vectors in MPEG-2. The zero motion vector is the most occurring, hence, it is assigned the shortest code.

Variable Length Code	Motion Vector
0000 0011 001	-16
0000 0011 011	-15
...	...
0011	-2
011	-1
1	0
010	1
0010	2
...	...
0000 0011 000	16

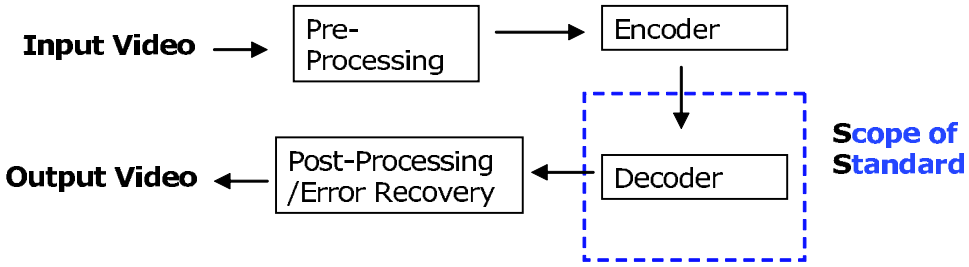


Fig. 1.9. The scope of video coding standard only defines the decoder and not the encoder.

difference between their goals and ISO MPEG are very much blurred (see below). Video compression standards developed by ITU are designated by the label "H.26x."

In the early 1990's the International Organization for Standardization (ISO) began looking at video compression for computer and multimedia applications. ISO formed MPEG to develop video compression standards for ISO. MPEG can be thought of as the "computer guys." Standards developed by MPEG are designated by the label "MPEG-x." The MPEG committee [24] [25] tends to be better known than the ISO although both had an equally important impact on the video compression industry.

It is important to note that both ITU and MPEG video compression standards only describe the decoder and not the encoder structure, as shown in Figure 1.9. The standards describe the syntax of the encoded bit stream as well as behavior of a compliant decoder. Developers then are allowed to design the encoder any way they want as long as it produces a compliant bit stream.

In order to achieve better compression, the video may be pre-processed before being processed by the encoder [26]. Since errors may be introduced to the compressed video data, error concealment [27] or block-artifact reduction techniques [28] may be used after the decoding process to enhance the overall quality of the video. It should be emphasized that such techniques are not part of the video compression standards.

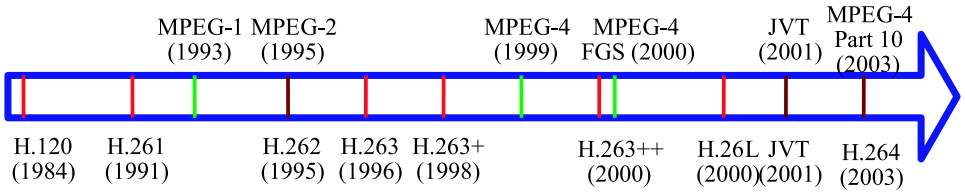


Fig. 1.10. Timeline of video coding standards. MPEG-2/H.262 and MPEG-4: Part 10/H.264 were joint projects of MPEG and ISO.

A timeline of the development of video compression standards is shown in Figure 1.10, with the contributions of MPEG and ISO shown above and below respectively. It should be noted noted that MPEG and ITU jointly developed two standards, these are MPEG-2/H.262 and MPEG-4: Part 10/H.264.

1.4.1 H.261

The H.261 video coding standard described a framework for video that is still widely used today. It was originally adopted in early 1991. This standard supports both I- pictures and P-pictures with a target data rate of 64-2048 Kbit/s. The standard supports two input formats: CIF and QCIF (see Table 1.1). The smallest coding unit in H.261 is the 16×16 pixel macroblock, each containing a header which specifies the address and the compression mode. A Group of Blocks layer (GOB) is defined to be a 3×11 matrix of macroblocks. A picture layer contains GOBs, with a header containing input format and picture number.

Motion compensation is employed in H.261 utilizing forward only prediction, that is, only P-pictures are allowed. The allowed range for each motion vector component (horizontal and vertical displacement) is $[-15, 15]$.

1.4.2 MPEG-1

MPEG-1, formally known as ISO/IEC 11172, is an international standard approved in 1993 [29]. It was aimed at the digital storage of video and audio with a

target data rate of approximately 1.5Mb/s (including both video and audio) [30]. The first three main parts of the MPEG-1 standard includes the following:

Part 1: Systems The first part of the standard described the framework for the multiplexing of the audio and video into a single stream. It was geared towards virtually loss free channels, e.g. reading from a CD-ROM.

Part 2: Video This concerns the video portion of the standard.

Part 3: Audio This concerns the audio portion of the standard.

Some application issues were considered in the MPEG-1 coding standard, include the following:

- Random access of the video stream is important for many applications, including video storage. MPEG-1 utilizes I-pictures to allow quick access to arbitrary points in the video, as well as "trick"-modes such as fast forward and reverse playback.

- Coding and encoding delay has been limited to approximately 1 second for interactive video applications.

The typical input format is CIF (shown in Table 1.1), although input resolutions up to 4096×4096 are supported. The input colorspace is 4:2:0 $YCrCb$, as in H.261.

The MPEG-1 standard specifies a hierarchy of data units, similar to H.261:

1. Blocks are 8×8 pixel, which is the size of the 2D DCT.

2. Macroblocks (MBs) are 16×16 pixels. Some compression parameters can be varied from MB to MB.

3. Slices are groups of continuous MBs.

4. Pictures consist of slices, which consist of I,P, and B picture types.

5. I-pictures utilize only intracoding, similar to JPEG still image compression. P-pictures are motion-compensated encoding utilizing forward prediction. B-pictures are introduced in this standard, which can utilize forward, backwards, or bidirectional prediction.

6. GOPs are groups of pictures or a consecutive sequence of pictures. The first picture of a GOP is always an I-picture. In MPEG, the temporal order of the pictures may not be equal to the order they are stored since P and B pictures are dependent upon pictures already coded.

MPEG-1 Intracoding

Intracoding in MPEG-1 is realized using I-pictures. An I-picture is completely intracoded with no dependencies on other pictures. They do not propagate errors and allow the use of "trick" modes such as fast-forward and reverse playback. The standard specifies at least 1 out of 132 pictures must be an I-picture in order to prevent decoder drift, which is the encoder decoder mismatch due to the accumulation of rounding errors associated with a floating point DCT/IDCT.

For intracoding I-pictures in MPEG-1, the standard utilizes a 8×8 DCT based technique similar to JPEG. The DCT coefficients are quantized by dividing by the specified quantizer step size and rounding to the nearest integer. MPEG-1 allows quantization to varied on a macroblock basis. Each quantized DC coefficient is first differentially encoded, with the residual VLC coding using 8 bits. AC coefficients are zig-zag scanned and converted into [run,level] pairs, similar to JPEG and H.261. A single fixed code table is used for all blocks and color component.

MPEG-1 Intercoding

MPEG-1 supports two types of interpicture prediction: forward prediction and bi- directional prediction.

Forward prediction, or P-pictures, allow motion compensated prediction in MPEG-1. P-pictures utilize previously coded I-pictures or P-pictures. In addition to the temporal prediction, an intracoded residual signal may be transmitted. In the special case of a zero motion vector, the macroblock may be transmitted in SKIP mode, in which neither a motion vector or intracoded residual signal is transmitted.

The second type of interpicture prediction in the MPEG-1 standard is B-pictures, which utilizes bidirectional prediction. Each macroblock in a B-picture can utilize a previous picture, a future picture, or both a previous and future picture as a reference. Just like P-pictures, B-pictures can reference only I-pictures and P-pictures. If two pictures are utilized for reference, the previous and future data are averaged in order to perform the prediction signal.

B-pictures have several unique properties:

- They are not used as a reference for other pictures, hence they do not propagate errors in the video sequence. Hence, B-pictures are generally more heavily quantized than I or P-pictures to provide more coding efficiency.

- The use of bidirectional prediction can provide a better predictor than regular forward prediction.

- B-pictures address the problem of covering and uncovering of parts of the picture due to its bidirectional prediction ability.

- More memory is needed by the decoder since two reference pictures may be needed to be decoded and stored in order to be used as a reference.

- The use of two reference pictures by the decoder requires the picture coded order to differ from the display order. The transmission of an additional reference picture increases the coding delay.

- Too many consecutive B-pictures may actually result in decreased coding efficiency due to decreasing temporal correlation between two pictures.

1.4.3 MPEG-2

The MPEG-2 standard [5], also known as ITU-T Recommendation H.262 or ISO/IEC 13818, was developed jointly by ISO/IEC JTC1 and ITU-T and completed in 1994. The MPEG-2 Video standard was developed for the growing need for video coding standard for applications such as digital storage and broadcast television.

It was designed to be backwards compatible with the MPEG-1 standard. The target data rate for MPEG-2 is 4-30 Mbit/s. It is used extensively today, being the standard compression scheme used in DVDs and broadcast High Definition Television (HDTV). The development of MPEG-3 for HDTV was never finished and the features were instead integrated into the MPEG-2 standard.

MPEG-2 Part 1: Systems

MPEG-2 Systems defines different coding layers to efficiently handle several sources of data simultaneously. More details are provided in the next chapter.

MPEG-2 Part 2: Video

MPEG-2 video coding is very similar to MPEG-1 video coding. The main features introduced in MPEG-2 are better data rate scalability (efficient use of high data rates) and native support for interlaced video. More details on the coding tools in MPEG-2 video can be found in the next chapter.

1.4.4 MPEG-4 Part 2: Visual

This standard is also known as ISO/IEC 14496-2 [31]. The first three parts of the standard are similar to MPEG-1 and MPEG-2:

Part 1: Systems Systems layer coding

Part 2: Video Coding of the video layer

Part 3: Audio Audio coding

Version 1 of the standard was adopted by MPEG in December 1998. Version 2 of the standard was frozen in December 2000. Version 2 and any subsequent versions are designed to be a superset of previous versions and be fully backwards compatible with previous versions.

Amendment 2 to the MPEG-4 Video standard defines a group of streaming video profiles. Of particular interest is the Advanced Simple Profile (ASP), which is the preferred standard of the Internet Streaming Media Alliance (ISMA) for Internet multimedia streaming over broadband networks. MPEG-4 is also the basis for DivX [32]and Xvid [33].

The MPEG-4 Visual standard introduces a variety of new coding tools for natural video scenes, similar to earlier MPEG and ITU standards. The standard also defines several new tools for video coding, including arbitrary shape coding, 2D-mesh coding, and face animation. The MPEG-4 video coding standard defines Video Object Planes (VOP), which are analogous to pictures in previous video coding standards.

For coding of rectangular shaped regions, MPEG-4 Visual improves on previous coding tools. Intracoding uses an 8×8 DCT with zig-zag scanning. The DC coefficient and optionally the first row and column of AC transform coefficients are predictively coded with respect to neighboring macroblocks. P-pictures use 16×16 macroblocks but allow an 8×8 macroblock mode with 4 motion vectors per macroblock. Motion vectors are unrestricted, i.e. they are allowed to reference outside picture boundaries.

The Advanced Simple Profile introduces more tools for greater coding efficiency:

Global Motion Compensation (GMC) This mode attempts to use additional motion models to represent large regions of the video moving together, such as in a camera panning, zooming, and tilting. The use of this option can provide data rate savings as many nearby motion vectors will be highly correlated. This feature has shown 3 dB PSNR improvements in video with fast camera movement, such as ‘foreman’ and ‘coastguard’, and ‘stefan’ [34]. This feature

does not show any improvement in sequences or regions of a sequence without any global motion.

B-VOP MPEG-4 defines Bi-predictive VOPs (B-VOP) which are to similar to B-pictures in previous standards.

Quarter-pixel motion vectors Motion vector accuracy is increased to 1/4-pixel. An 8-tap FIR filter generates the 1/2-pixel samples and bilinear averaging is used to generate the 1/4-pixel samples.

Interlaced Support The standard includes a Field DCT mode for interlaced video.

The MPEG-4 standard also includes tools for error-resilient coding [35]. These tools include the following:

Resynchronization If the video decoder detects an error in the decoded video, it can quickly lose track of its precise location in the decoded video. With the resynchronization feature, the video decoder can seek for a resynchronization marker. These markers can be inserted in a more flexible way than previous standards, such as H.261 and H.263.

Data Partitioning A motion boundary marker (MBM) is inserted in the video bitstream that separates the motion data and the DCT data. If the decoder detects an error in the motion data, it replaces all the current macroblocks with skipped macroblocks. Likewise, if an error is found in the DCT data, all of the current DCT data is unused.

Reversible Variable-Length Codes (RVLC) If the decoder detects an error in VLC data, typically all the current VLC data must be discarded until the next resynchronization marker. If RVLCs are utilized, the decoder can attempt to decode the data in the backwards direction until the error is found.

Header Extension Code (HEC) Important header data is placed redundantly in the video pictures and the video packets as well.

MPEG-4 also has tools for coding interlaced video [36], such as alternate DCT scan orders and field-based prediction.

1.4.5 H.264 / MPEG-4 Part 10: Advanced Video Coding

In early 2000 ITU and MPEG began working jointly again on a new video compression standard. They formed a group known as the Joint Video Team (JVT) to examine new issues in video compression [37]. The official ITU and MPEG designations are H.264 and and MPEG-4 Part 10: Advanced Video Coding respectively. In this thesis, the ITU designation H.264 will be used to refer to the standard. More details on H.264 can be found in [6, 38, 39].

This standard was designed to target a wide variety of applications, including wireless [40], IP networks [41], and digital cinema.

The standard has defined two separate main coding layers: the Video Coding Layer (VCL) and the Network Abstraction Layer (NAL). The VCL will be the focus of this paper, although the NAL will be discussed in some detail.

In comparison to previous standards, this standard is a departure from earlier standards, introducing many new technical features to further increase compression efficiency, such as flexible macroblock sizing, 1/4-pixel interpolation [42], multiple reference picture capabilities [43], and an in-loop filter [44].

2. A COMPARISON OF MPEG-2 AND H.264 VIDEO CODING

H.264 and MPEG-2 video coding both are motion-compensated hybrid coders but possess important differences. This chapter outlines the different tools used in both standards and highlights the differences of the tools found in each standard.

2.1 Transform Video Coding

In order to exploit spatial redundancy, most video coding methods (including MPEG-2 and H.264) incorporate some form of transform coding. To efficiently code the spatial data, first a transform is performed on the spatial data in order to decorrelate it so that it can represented using the fewest coefficients [45]. The most common transform utilized in modern video coding methods is the Discrete Cosine Transform (DCT), which is used in MPEG-2. After the transform, the transform coefficients are quantized and then are entropy coded for efficient binary representation. H.264 uses a new integer transform described below.

2.1.1 Transform Coding in MPEG-2

The transform specified in the MPEG-2 video coding standard is a floating point DCT. This same transform is used in many video coding standards, including MPEG-1, H.263, and MPEG-4: Part 2 Visual, as well as the JPEG image coding standard. The main role of the DCT is to decorrelate the spatial data in the image. The DCT is widely used in transform coding since it is a close approximation to the Karhunen-Loeve Transform (KLT) [17, 45].

Many DCT-based compression techniques partition the image into (8×8) blocks and apply the DCT to each block. For a given image or block $f(x, y)$, the 2-D 8×8 DCT $F(u, v)$ is given by:

$$F(u,v) = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) * cos\frac{(2x+1)u\pi}{16} cos\frac{(2y+1)v\pi}{16}\right], \qquad (2.1)$$

while the inverse discrete cosine transform (IDCT) is given by

$$f(x,y) = \frac{1}{4}\left[\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v) * cos\frac{(2x+1)u\pi}{16} cos\frac{(2y+1)v\pi}{16}\right], \qquad (2.2)$$

where $C(u), C(v) = \frac{1}{\sqrt{2}}$ for $u, v = 0$ and $C(u), C(v) = 1$ otherwise.

DCT block sizes larger than 8×8 are more computationally complex and do not offer significantly greater decorrelation properties [46]. The 8×8 transform size is an appropriate trade off between decorrelation performance and computational complexity.

Once the spatial data is transformed into DCT coefficients, it is quantized using a scalar quantizer, given by (1.2). MPEG-2 defines quantizer matrices which specify the parameter values β in (1.2). For intracoded data (I-pictures), the quantizer matrix is tailored to properties of the human visual system, which is more sensitive to lower frequencies (upper left of the quantizer matrix) and less sensitive to higher frequencies (lower right of the quantizer matrix). The spatial residual data for intercoded pictures (P/B-pictures) is uniformly quantized using a separate uniform quantizer matrix. Both quantizer matrices are illustrated in Figure 2.1.

Entropy Coding in MPEG-2

To efficiently represent the transform coefficients, the MPEG-2 video coding standard utilizes a method very similar to the method used in the original JPEG image compression standard [16].

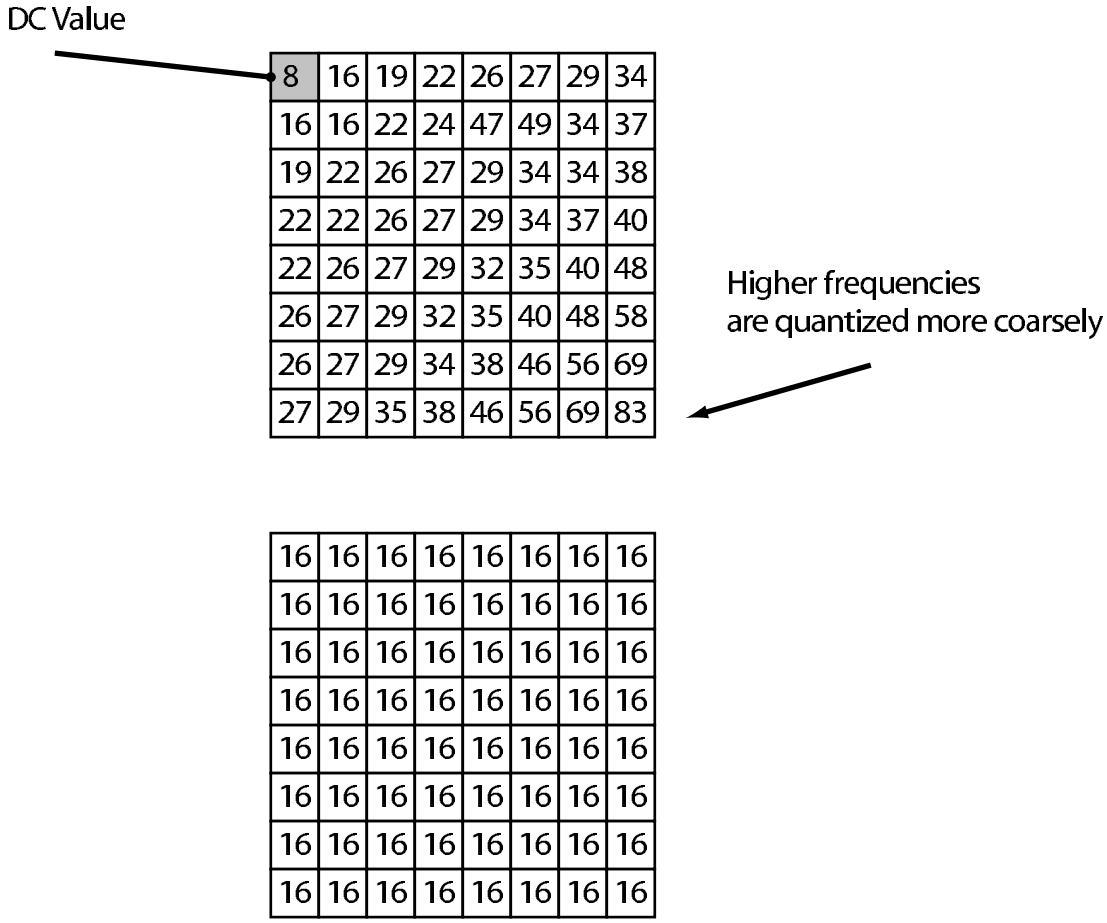


Fig. 2.1. Quantization matrices for DCT coefficients for intracoded data (top) and intercoded data (bottom).

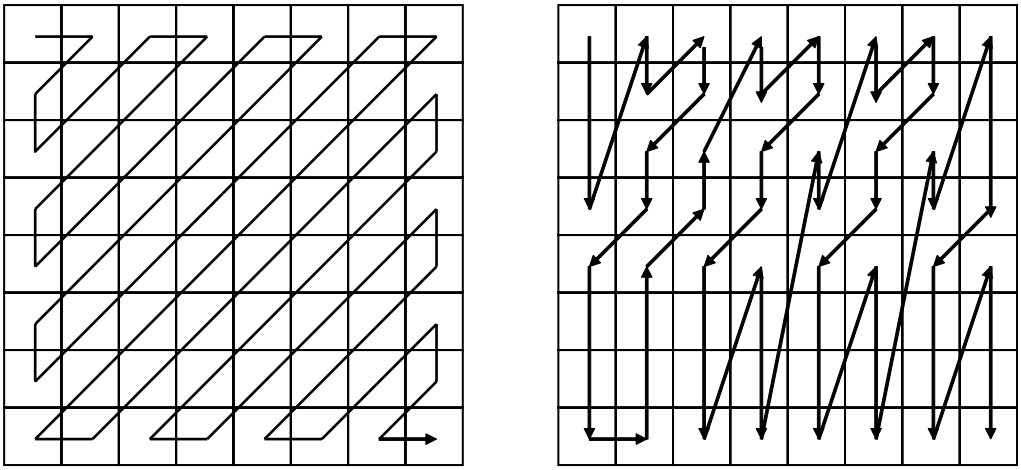


Fig. 2.2. The normal zig-zag scanning order, shown on the left, is used for frame-coded pictures in MPEG-2. The alternate scanning order of transform coefficient for field-coded pictures is shown on the right.

Since adjacent 8×8 blocks are often highly correlated, e.g. sky or solid color background, the DC coefficient (upper left corner) is differentially coded with respect to the previously coded 8×8 block. The DC coefficient can be encoded with 8 to 11 bits of precision.

The coding of the quantized DC coefficient uses a variable-length code (VLC) and a variable-length integer (VLI) specified in the MPEG-2 standard. First a VLC is coded that specifies the length (and also the dynamic range). The VLI then follows the VLC, specifying the actual value in the given range.

To entropy code the quantized AC coefficients, the coefficients are scanned in a normal zig-zag order, in the case of frame-coded pictures, or an alternate scan order, in the case of field pictures. The two different scan orders are shown in Figure 2.2. The resulting [run,level] pairs are then binary coded using one of the two available VLC tables specified in the MPEG-2 standard. Each block is terminated using an End Of Block (EOB) symbol.

2.1.2 Undesirable Properties of the DCT

Although the DCT is used in many coding standards, it does possess some undesirable properties:

- The DCT transforms an integer signal into a real-valued signal. This implies the use of floating point arithmetic, which can be expensive to implement in hardware or software.
- The use of floating point numbers introduces rounding errors during the transform and inverse transform process. The encoder and decoder may mismatch due to rounding, which can not be completely avoided.

In order to address these shortcomings, the MPEG-2 standard specifies the amount of precision required by a compliant decoder. The error propagation due to mismatch error of present frames predicted from previous frames can be minimized by the periodic insertion of intracoded frames, which do not rely on previously coded frames. Accumulated drift can be terminated by the insertion of an intracoded frame.

2.1.3 Transform Coding in H.264

Intracoding is done in a much more efficient manner as compared to MPEG-2. Table 2.1 presents a short summary of differences between the tools in H.264 and MPEG-2.

The H.264 standard introduces a new transform that is similar to the DCT [47, 48]. This new transform has the following properties:

- The transform block size is 4×4, as opposed to the 8×8 DCT transform found in MPEG-2.
- It is an integer-to-integer transform and does not require floating-point arithmetic.

Table 2.1
Intracoding feature comparsion between MPEG-2 and H.264.

	MPEG-2	**H.264**
Transform	DCT	4×4 integer transform
Spatial Prediction	DC Only	9 ways
DC Coding	Differential	Second level transform

- The integer-based transform allows for exact reconstruction and completely eliminates any decoder drift resulting from encoder/decoder mismatch.

- The inverse transform requires ony 16 bits of integer precision to obtain.

- It can be obtained using only addition and shift operations which are much faster in hardware than multiplication and division operations.

Due to the spatial prediction and other intracoding tools in H.264, the intracoding in H.264 outperforms JPEG by 3.83 dB and approaches the JPEG2000 standard [49] in performance (within 0.9 dB) [50]. A more in-depth analysis of the H.264 intracoding performance is provided in [48].

Development of the 4x4 Integer Transform

This section gives a brief outline of the motivation behind the development of the new transform in H.264 [47]. Consider a 1-D vector x be transformed to X by the following relationship:

$$X = Hx, \tag{2.3}$$

where x is the input, X is the output, and H is the 2-D square transformation matrix. Let H' be a 4×4 DCT matrix defined by

$$H' = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ c & s & -s & -c \\ 1 & -1 & -1 & 1 \\ s & -c & c & -s \end{bmatrix}, \tag{2.4}$$

where $c = \sqrt{2}\cos(\frac{\pi}{8})$ and $s = \sqrt{2}\sin(\frac{\pi}{8})$. H is an orthogonal matrix. The matrix H' contains real numbers but it would be advantageous for H' to contain only integers for simplified computation. Let H represent an integer approximation of H' defined by

$$H = round(\alpha H'), \tag{2.5}$$

where α represents an arbitrary scaling factor. The only values for α that produces an orthogonal matrix H with equinorm rows are 2 and 26. The value of $\alpha = 2$ results in a Hadamard transform matrix which is unsuitable due to its poor decorrelation properties. The value of $\alpha = 26$ was originally proposed for the H.264 standard [47], which results in

$$H = \begin{bmatrix} 13 & 13 & 13 & 13 \\ 17 & 7 & -7 & -17 \\ 13 & -13 & -13 & 13 \\ 7 & -17 & 17 & -7 \end{bmatrix}, \tag{2.6}$$

which is a close approximation of a DCT matrix (with some scaling). The main problem with this choice of α is the increase of dynamic range of the output given the input. Assuming $max|x(n)| = A$, where A is an arbitrary constant, then it can be seen $max|X(n)| = 52A$, which results in a dynamic range increase by a factor of 52. Since the input is two dimensional, the transform needs to be performed twice which results in an overall gain of $52^2 = 2704$, which requires an additional $\log_2(2704) = \lceil 11.4 \rceil = 12$ bits of storage. This would require the use of 32-bit arithmetic for the transform and inverse transform.

The value for α chosen in the standard is $\alpha = 2.5$, which results in

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}, \tag{2.7}$$

which does not posses the property of equinormal rows. This is compensated for in the quantization process by the use of non-uniform scaling of the coefficients.

Properties of the 4x4 Integer Transform

Using the transform in (2.7), it can be seen the dynamic range increases by a factor of 6. Given the 2-D transform and $\log_2(6^2) = 5.17$, H only requires an additional 6 bits.

The transform specified in the standard possesses many desirable properties. First it should be noted that each coefficient in the transform matrix is either ± 1 or ± 2, which can obtained very efficiently using only shifts. The inverse transform $\widetilde{H}_I$ is defined by

$$\widetilde{H}_I = \begin{bmatrix} 1 & \frac{1}{2} & 1 & \frac{1}{2} \\ 1 & 1 & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 2 & -\frac{1}{2} \end{bmatrix}, \tag{2.8}$$

in which the tilde is shown since $\widetilde{H}_I H \neq I$. $\widetilde{H}_I$ is not a true inverse but rather a scaled inverse defined by

$$\widetilde{H}_{inv} \begin{bmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & \frac{1}{5} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{5} \end{bmatrix} H = I, \tag{2.9}$$

where I is an identity matrix. The reader should note that the division by 2 is not exactly a right shift by 1-bit, but the shift can be used with virtually no degradation in performance [47] and guarantees identical results from decoder to decoder.

H.264 Second Level Transform of DC Coefficients

After transforming each 4×4 block, the upper left most component is still the DC value of the block. In adjacent 4×4 blocks, the DC values tend to be highly correlated in flat areas, such as sky regions. DC values of adjacent blocks undergo a second level transform, or hierarchal transform [51]. It was originally proposed to have these DC values undergo the same transform specified in (2.7), it was later simplified to a simple Hadamard transform. Experimental results showed that this simplification greatly reduced complexity while sacrificing very little coding efficiency. This transform is specified by (2.4), with $c = s = 1$.

Spatial Prediction in H.264

In H.264 intracoding, spatial data is coded with respect to previously coded spatial data, or spatially predicted. First, spatial data within a picture is predicted from a number of different directions, shown in Figure 2.3. Then the prediction error signal is transformed and encoded. Similar to interframe prediction, spatial prediction reduces the energy of the coded signal, which results in greater coding efficiency.

Context Adaptive Variable Length Coding

For the coding of the quantized transform coefficients, H.264 offers two entropy coding methods: Context-Adaptive Variable Length Coding (CAVLC) and Context-Adaptive Binary Arithmetic Coding (CABAC). CAVLC is related to the entropy coding method in MPEG-2 in the sense that fixed code tables are used, but the

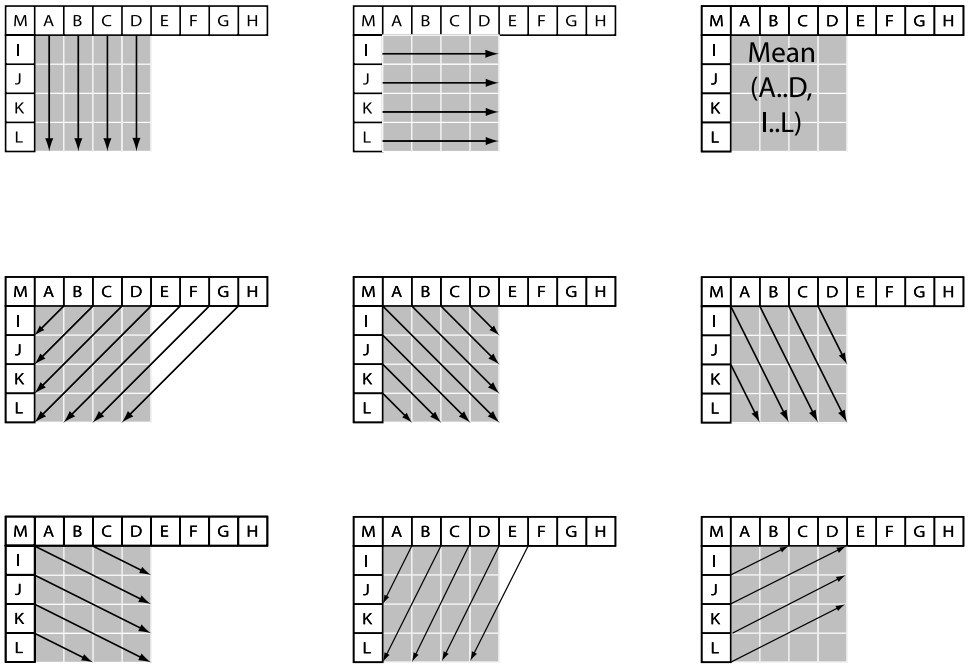


Fig. 2.3. Different methods of spatial prediction in H.264. More prediction leads to less explicitly-coded data.

scheme in H.264 is much more flexible. A detailed example of CAVLC entropy coding is shown in [38].

After the zig-zag scan, the number of trailing zeros and trailing ones are considered. Depending on the statistics of the neighboring blocks, it uses one of four available VLC tables to encode this data. The encoder then codes the coefficients in reverse order, choosing one of six possible Exp-Golumb code tables defined in the standard.

Context Adaptive Binary Arithmetic Coding

The second form of entropy coding in the H.264 coding standard is CABAC. This method has the following desirable properties:

- It is possible to assign a non-integer number of bits to each symbol, e.g. transform coefficient, which is advantageous for symbols with higher probability than 0.5.

- The adaptive arithmetic code allows the code to change as the statistics of the input changes.

The implementation complexity is kept to a minimum by utilizing only shifts and table look-ups. This form of entropy coding can achieve reductions of data rates of approximately 5-15% according to recent work [52].

On the downside, CABAC entropy coding is significantly more computationally intensive than CAVLC. Recent work has shown CABAC entropy coding to require 4 times the clock cycles compared to CAVLC on a Texas Instruments C64x processor [53], which accounts for 35-75% of the total cycles required by the H.264 decoder.

2.2 Motion Compensated Video Coding

Motion compensation is used to exploit temporal redundancy within the video sequence to reduce the data rate. Table 2.2 provides an overview of various tools for motion compensation available in MPEG-2 and H.264. The following two sections will discuss the tools in more detail.

2.2.1 Motion Compensation in MPEG-2

Basic Macroblock Types for Forward Prediction

MPEG-2 specifies a 16×16 macroblock, which is the basic unit that is coded based on previous pictures. For P-pictures in progressive video, the standard allows for the following 3 basic types of macroblocks:

- INTRA-16×16 - The 16×16 macroblock is intracoded, without any references to any previously coded pictures.

Table 2.2
Motion compensation comparsion between MPEG-2 and H.264.

	MPEG-2	H.264
MV Precision	1/2 pixel	1/4 pixel
MB Sizes	16×16 macroblocks	4×4 to 16×16 macroblocks
P-Pictures	Max 1 ref. frame Refers past only	Multiple ref. frame Can refer future
B-Pictures	Max 2 ref. frames Not used as reference MB Skip Mode	Multiple ref. frames Can be used as reference MB Skip/Direct Mode Weighted prediction (fades)

- INTER-16 × 16 - A 16 × 16 macroblock is coded with a single motion vector referring a previously coded picture.

- SKIP - The 16 × 16 macroblock is simply copied from the reference picture. This is also known as a zero motion vector.

Sub-Pixel Motion Prediction

In order to achieve better modeling of the motion vector field, the motion vectors are not limited to being integers. Sub-pixel samples are interpolated for a more accurate motion-compensated prediction, which leads to less energy in the error signal [54].

The sub-pixel samples are created in the decoder by using a bilinear averaging, which results in 1/2-pixel sample accurate motion compensation. An example is shown in Figure 2.4, where the dashed pixels are interpolated using a linear average of neighboring shaded pixels.

Interlaced Motion Compensation in MPEG-2

MPEG-2 supports additional modes of motion compensation to efficiently represent the motion present in interlaced video. Due to the nature of alternating fields updating every cycle, adjacent rows in field-pictures are not as correlated as in frame-pictures when motion is present. In additional to standard frame-based prediction, MPEG-2 provides additional coding tools for field-based prediction:

Field Prediction for Field-Pictures This mode is conceptually similar to frame-prediction, except that the predictive signal originates from a single-field (top or bottom).

Dual Prime for P-Pictures In this mode, two initial predictions are initially created, one from the same field as the target and another from the opposite

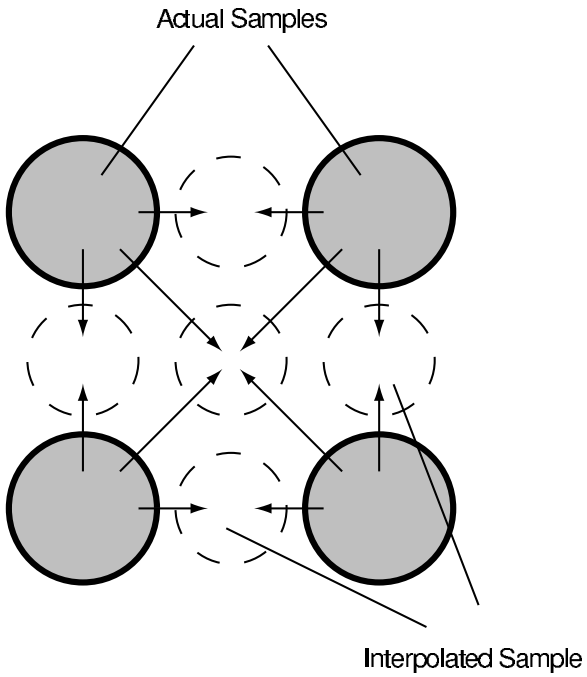


Fig. 2.4. MPEG-2 interpolates half-pixel positions for more accurate motion prediction.

field. These predictions are then averaged to create a final prediction. Each macroblock has one associated motion vector.

16×8 **Motion Compensation** Each 16×16 macroblock is split into two 16×8 field-based halves. Each macroblock half has an associated motion vector. Therefore, each P macroblock has two associated motion vectors and each B macroblock has two or four associated motion vectors.

MPEG-2 Bi-Directional Prediction

In order to achieve further gains in motion compensation, MPEG-2 supports bi-directional prediction, or B-pictures. B-pictures utilize not only the previous picture as a reference picture (as done in P-pictures), but also a subsequent, or future picture in the video sequence. B-pictures can be most advantageous in natural video scenes with moderate motion, such as slow camera panning [55]. B-pictures in MPEG-2

show measurable quality improvement, from 0.3 to 0.6 dB in terms of average overall PSNR, as compared to similarly encoded sequences without B-pictures or dual-prime prediction [56]. Subjective tests have shown greater perceived quality improvement than the numbers would suggest.

Since B-pictures utilize an extra reference picture and hence offer improved prediction, they can require less data to achieve same quality as the otherwise same P-picture.

For reference pictures, B-pictures utilize only previously coded I-pictures or P-pictures. In order to prevent excess propagation of coding error, MPEG-2 video never uses B-pictures as a reference for other coded pictures.

While B-pictures offer greater coding efficiency, they have some disadvantages:

- B-pictures introduce extra decoding delay. Since the (temporally) future reference picture must be coded previously to the current picture, the display order and coding order differ. This introduces delay in the reconstruction process since the temporally future picture must be decoded before the current picture. In many applications, such as entertainment applications, the delay may be ignored or go unnoticed. In applications such as video conferencing where minimal delay is paramount, bi-directional encoding delay may be unsuitable.

- B-pictures are more expensive to encode than P-pictures. A current picture depends on two reference pictures instead of the normal single reference picture. Also, each macroblock can be predictively use the first, the second, or both reference pictures, which creates more modes for the encoder to choose from, increasing encoding complexity. Also, memory usage is increased since the encoder requires both reference pictures.

MPEG-2 Motion Vector Entropy Coding

In order to more efficiently to represent the values of each component (x, y) of a motion vector, MPEG-2 differentially encodes each motion vector with the previously encoded motion vector. The motion vector difference is defined by the following:

$$\Delta_{MV} = 2(MV - PMV), \tag{2.10}$$

where MV represents the magnitude of the current motion vector, PMV represents the value of the most recently coded motion vector within the same slice, and the scalar constant 2 is to account for the half-pixel accurate motion vectors. If the previous macroblock belonged to a different slice that the current macroblock, the value PMV is reset to $PMV = 0$.

By using this simple prediction for coding of the motion vectors, the actual magnitude of the value that has to be encoded in the bitstream is reduced. This exploits the correlation of movement between neighboring macroblocks.

For coding of motion vectors in B-pictures, the current forward motion vector is coded with respect to the previously coded forward motion vector. Similarly, the current backward motion vector is coded with respect to the previously coded backward motion vector. This applies even there are two backward or two forward motion vectors in the current macroblock.

Once Δ_{MV} is obtained for the current motion vector, it is encoded according to a fixed VLC table specified in the MPEG-2 standard. Since smaller values are assigned shorter VLC code words, smaller values of Δ_{MV} are desirable for greater coding efficiency.

2.2.2 Motion Compensation in H.264

The H.264 motion model is much more flexible compared to MPEG-2. The H.264 standard allows for a much wider variety of encoding modes as compared to previous video coding standards.

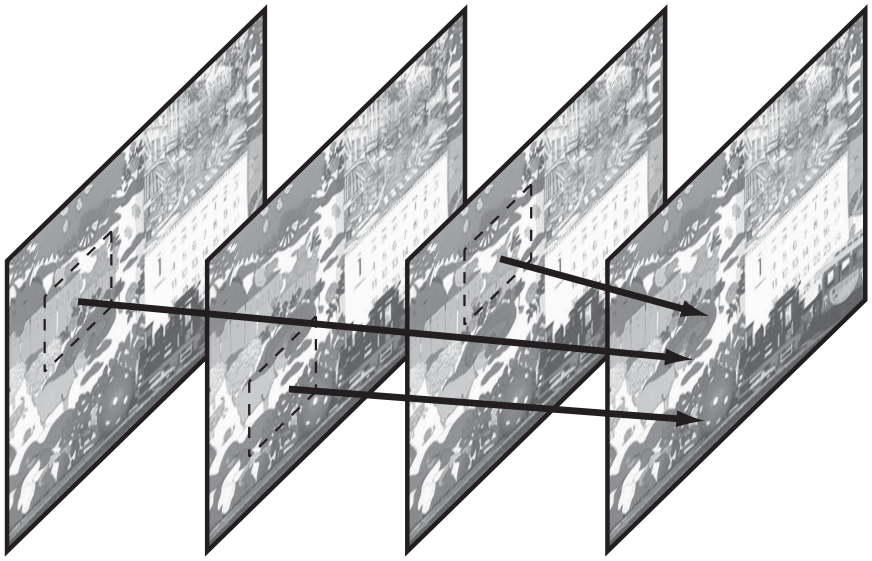


Fig. 2.5. H.264 can use multiple reference pictures for prediction. Early standards were limited to a maximum of two reference pictures.

Multipicture Prediction

H.264 introduces the capability of using multiple reference pictures for prediction, up to a maximum of 15. This does not limit the current picture to serve as a reference to only the immediate proceeding or immediate subsequent picture. An example of multipicture prediction is shown in Figure 2.5. The usage of more temporally further pictures as a reference has been shown to provide significant gains in many situations, such as scene cuts, uncovered background, texture with aliasing, and similar references in the presence of noise [57].

Previous work compared the use of 5 references pictures with the use of two and three references pictures. The use of 2 reference pictures has shown to achieve 62% of the total gain achieved by 5 reference pictures and 3 reference pictures achieve 83% of the total gain [58].

Macroblock Types in H.264

H.264 is similar to the MPEG-2 in the respect that it uses the basic 16×16 macroblock, along with similar INTRA-$16x16$, INTER-$16x16$, and SKIP modes in P-pictures. The H.264 standard also allows a 16×16 macroblock to be partitioned in smaller submacroblocks. Other macroblock modes in H.264 P-pictures include the following: INTER-$8x8$, INTER-$8x4$, INTER-$4x8$, INTER-$4x4$, and INTRA-$4x4$. The different macroblock sizes are illustrated in Figure 2.6.

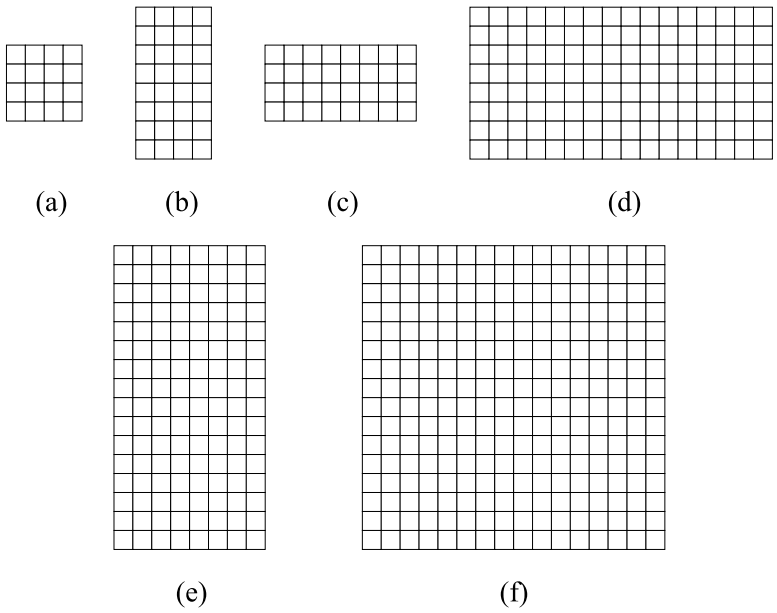


Fig. 2.6. H.264 allows several different macroblock sizes: (a) 4×4, (b) 4×8, (c) 8×4, (d)16×8, (e) 8×16, and (f) 16×16. Each box represents a single pixel.

The flexibility in macroblock sizing allows the most efficient macroblock size to be chosen for a given area of the current encoded picture [59].

Quarter Pixel Accurate Motion

H.264 utilizes 1/4-pixel interpolative motion compensation which allows for greater motion compensation accuracy than the 1/2-pixel motion compensation in MPEG-2. The motion compensation in H.264 allows for more accurate motion compensated prediction, which results in less energy in the residual signal. The use of 1/4-pixel interpolation has been shown to provide up to 2 dB PSNR improvement over 1/2-pixel interpolation in testing [42]. 1/8-pixel interpolation was initially proposed for H.264 but was not adopted due to complexity issues. The 1/4-pixel interpolation is performed as a two step process. First, 1/2-pixel predictions are first created by the following 6-tap filter in the horizontal and vertical directions:

$$b1 = (E - 5F + 20G + 20H - 5I + J), \tag{2.11}$$

$$h1 = (A - 5C + 20G + 20M - 5R + T), \tag{2.12}$$

where the pixel locations are shown in Figure 2.7. The final result for b and h are found by adding 16 and clipping to fall within $0 - 255$:

$$b = (b_1 + 16) >> 5 \tag{2.13}$$

$$h = (h_1 + 16) >> 5 \tag{2.14}$$

Once the half-pixel samples are interpolated, the final prediction values at the quarter-pixel positions are interpolated using a linear averaging with the adjacent pixels, e.g.

$$a = (G + b + 1) >> 1 \tag{2.15}$$

$$c = (b + h + 1) >> 1. \tag{2.16}$$

In Figure 2.7, non-shaded areas represent interpolated pixel positions and shaded areas represent actual pixels.

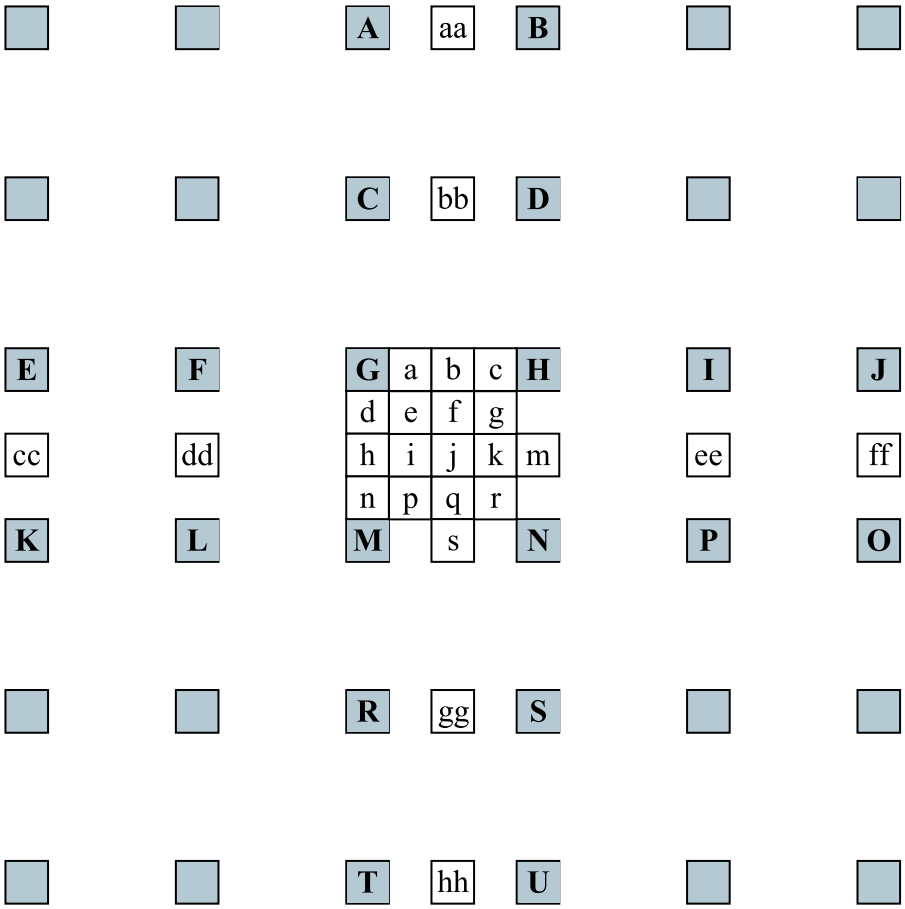
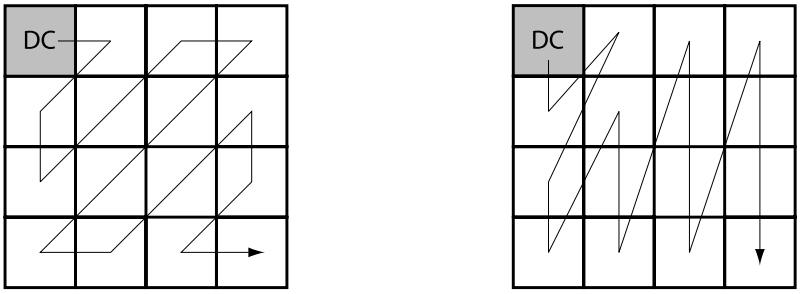


Fig. 2.7. Interpolation positions of 1/4 pixels in H.264.



(a) Normal scan order for frame-based video

(b) Alternate scan order for field-based video

Fig. 2.8. Transform scan orders in H.264.

Interlaced Support in H.264

As in MPEG-2, the H.264 standard includes special tools to code interlaced video. There are a few key differences between frame and field coding in H.264:

- The transform coefficients are scanned in an alternate order, similar to the alternate scan order in MPEG-2. The normal and alternate scan orders are shown in Figures 2.8(a) and 2.8(b) respectively.

- The predictively coded macroblocks refer to reference fields as opposed to reference pictures.

- Due to the nature of the interlaced material, the deblocking filter does not strongly filter horizontal edges between macroblocks. Since each alternating row is displayed at any given time, the rows undergoing the filtering are now twice as far apart, which would cause a larger filter.

Similar to MPEG-2 interlaced coding support, H.264 supports a mix of interlaced and progressive video:

Picture Adaptive Frame-Field Coding (PAFF) Each picture in a sequence can be adaptively field coded or frame coded, depending on which leads to a more efficiently coded result. PAFF has been shown to lead to a 16-20% reduction in data rate for ITU-R 601 sequences, e.g. 'canoa' and 'rugby' [38].

Macroblock Adaptive Frame-Field Coding (MAFF) This mode allows each vertical pair of macroblocks (16×32) to be adaptively coded as interlaced. Note that this is different than MPEG-2, in which an interlaced macroblock consists of a 16×8 region. This mode can benefit video which contains moving regions mixed with static regions. The moving areas can be coded as fields and the static regions can be efficiently represented with frame coding. MAFF can provide a data rate savings of 14-16 % over PAFF when coding ITU-R 601 sequences, e.g. 'mobile' and 'news' [38].

H.264 Bi-Directional Prediction

Bi-directional prediction in H.264 is more flexible as compared to previous coding standards [60]. There are several key differences between B-pictures in H.264 and B-pictures in MPEG-2 and other previous video coding standards:

- The H.264 standard frees the restriction in the use of B-pictures as reference pictures. P and B-pictures are permitted to use B-pictures as a reference.
- Direct coding modes allow sophisticated motion vector prediction.
- The motion compensated prediction signals can be weighted, which can help in certain situations such as scene fades. In this mode, the predictive signal is multiplied by a weighing factor.

The macroblock sizes and partitions include those outlined previously for P-picture macroblock types. B-pictures in H.264 fall into two general categories:

- Multi-Hypothesis Mode Macroblock - A predictive picture can be viewed as a linear combination of two general signals with side information, as is the case

in MPEG-2 and other previous coding standards. The two general signals, or hypothesises, are from a previous picture and a subsequent picture. The side information consists of the rest of of the information required to form the prediction, such as the motion vector data, block sizes, and reference frames.

- Direct Mode / Skip Mode - This is mode is different from the previous mode in that it does not require side information. Data rate savings can be realized by deriving this information from already coded data instead of explicitly coding it. This side information can be derived using the temporal direct or spatial direct prediction. In temporal direct mode, the side information is derived from a macroblock in a previously coded picture in the same spatial location. In spatial direct, the side information is derived from neighboring macroblocks in the same coded picture. The compression performance of both direct modes is virtually identical [61]. In addition, spatial direct has the advantage of less memory requirements. A special case of Direct Mode is Skip Mode, in which no residual signal is transmitted.

2.2.3 In-Loop Deblocking Filter in H.264

H.264 includes an in-loop deblocking filter to help reduce blocking artifacts caused by the block-shaped integer transform [44]. This filtering occurs immediately after the inverse quantization in the decoder. The in-loop filtering is different from a post-filter in that the in-loop filter is utilized during the motion-compensation process [62, 63]. Filtered frames are used as references for future frames during the motion compensation, which can help improve the quality of interframe predictions.

An adaptive filter has the advantage of smoothing blocking artifacts while preserving natural detail. The operation of the filter in H.264 adapts according to local encoding parameters such as prediction type, motion vector data, prediction error energy, and quantization level.

The inclusion of an in-loop deblocking filter was somewhat controversial during the development of the H.264 standard due to its added computational complexity. The in-loop deblocking filter has been shown to increase subjective quality and provide a 9% data rate reduction for equal PSNR [44].

Samples illustrating the use of the in-loop filter are shown in Figures 2.9 through 2.11. In Figure 2.9, the quantization level is lower, resulting in a lower deblocking filter strength. Figure 2.11 shows the effects when the quantization level is high. With higher quantization levels, blocking artifacts are much more likely to be visible.

2.3 Transmission of Compressed Video

This section provides a short overview of transmitting compressed MPEG-2 and H.264 video over a network.

2.3.1 Transmission of MPEG-2 Compressed Video

MPEG-2 Systems [64] defines different coding layers to efficiently handle several sources of data simultaneously.

A single source of compressed data, e.g. compressed audio or video, forms what is known as an *Elementary Stream* in the MPEG Systems. The elementary stream is then partitioned into fixed or variable length packets to form a Packetized Elementary Stream (PES). These PES packets are then multiplexed together to form a Program Stream or a Transport Stream.

Program Streams This is similar to MPEG-1 Systems layer. It is primarily suitable for non error-prone environments. A Program Stream consists of one or more PES packet streams.

Transport Streams A Transport Stream is designed for error-prone environments. A Transport Stream consists of one or more programs, each of which may have

Fig. 2.9. Picture #8 of the CIF foreman sequence, encoded with H.264 QP=24 without and with the in-loop filtering enabled. The resulting PSNR and data rates are 38.97 dB at 806.0 kb/s and 38.93 dB at 701.6 kb/s respectively.

Fig. 2.10. Picture #8 of the CIF foreman sequence, encoded with H.264 QP=32 without and with the in-loop filtering enabled. The resulting PSNR and data rates are 33.97 dB at 271.4 kb/s and 33.97 dB at 220.1 kb/s respectively.

Fig. 2.11. Picture #8 of the CIF foreman sequence, encoded with H.264 QP=40 without and with the in-loop filtering enabled. The resulting PSNR and data rates are 29.46 dB at 99.4 kb/s and 29.54 dB at 78.3 kb/s respectively.

a time base independent with one another. Each Transport Stream packet is exactly 188 bytes in length.

The Program and Transport streams were both designed with different goals in mind. It should be noted that one is not a subset or superset of the other.

2.3.2 Transmission of H.264 Compressed Video

Network Adaptation Layer

The H.264 standard defines a Network Abstraction Layer (NAL) [65] in addition to the Video Coding Layer (VCL). The scope of each layer is outlined in Table 2.3. The coded data is partitioned into separate NAL units. This allows easy mapping of H.264 coded video to be mapped onto a variety of different network layers. The relationship between the VCL and the NAL is illustrated in Figure 2.12.

Table 2.3
Scope of Video Coding Layer and Network Abstraction Layer

VCL	**NAL**
Block Layer	Slice Layer
Macroblock Layer	Picture Layer
	GOP Layer
	Sequence Layer

Transmission of H.264 Video using MPEG-2 Systems

The flexibility of the standard allows H.264 VCL data to be carried over popular the MPEG-2 Systems layer via Program Streams or Transport Streams.

One main issue that has arisen is the mapping of the variable length NAL units into fixed-length packets, such as the case in MPEG-2 Systems Transport Streams

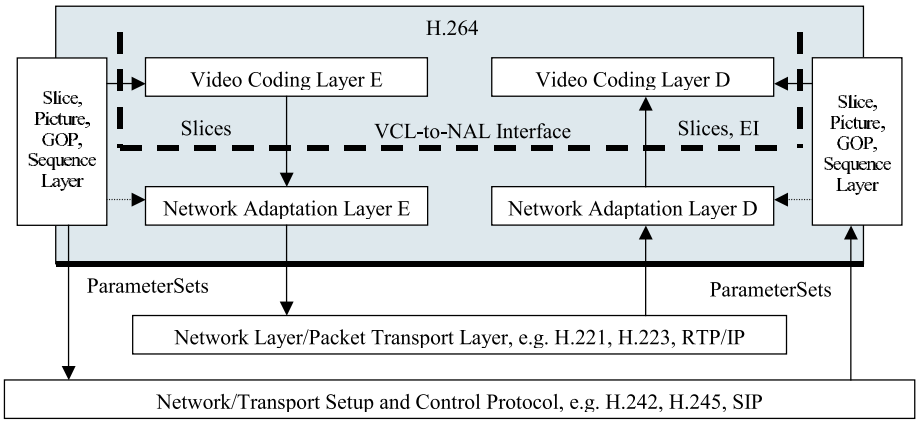


Fig. 2.12. Decoupling of the transmission of setup and control information (the picture/GOP, and Sequence information) from the lower layers.

[66]. MPEG-2 TS packets are always 188 bytes in length. A simple approach would simply align the start of each NAL unit with the start of a new TS packet. This would waste bits, however, since the end of the NAL unit would not always coincide with the end of TS packet. The TS packet would have to be padded in order to meet the exact length requirement of 188 bytes, which would result in much wasted bandwidth.

A solution would be to assign unique start codes for each NAL unit. In this way, the start of the NAL unit would not have to be aligned with the start of the TS packet. The decoder is able to uniquely identify the start of each NAL unit.

Amendment 3 to the MPEG-2 Systems standard [67] provides provisions to allow the transmission of H.264 video. The Amendment assigns a new stream type to H.264 video.

2.3.3 Error Resilient Coding Tools in H.264

Slice Coding

H.264 allows grouping of macroblocks known as slices. A slice is a subset of a picture. Analogous to I, P, and B frames, the H.264 standard includes I, P, B slices. The standard also specifies switching P (SP) and switching I (SI) slices. They are used to aid in the dynamic switching of different bitstreams encoded at different data rates. The definition of different slice types are very flexible:

I Slice All of the macroblocks in an I slice are coded using intra prediction.

P Slice An I slice except that some of the macroblocks can be coded using inter prediction, using a maximum of one prediction signal.

B Slice A P slice except that some macroblocks can be coded using interprediction, using a maximum of two signals

Flexible Macroblock Ordering (FMO)

The normal raster scan ordering (left to right, top to bottom) of macroblocks is normally utilized, but it may not the most optimal in terms of error resilience. FMO allows for a variety of scanning patterns, including chess boards or completely random patterns. By transmitting the macroblocks in a non-raster fashion, any errors in the transmission will manifest itself throughout the picture, instead of consecutive runs of macroblocks. In turn, the correct macroblocks can be used to efficiently conceal the errors in the damaged macroblocks.

Arbitrary Slice Ordering (ASO)

The slice structure in the H.264 standard allows slices within a picture to be decoded independently. Each slice can be then independently transmitted in an ar-

bitrary order. This feature can improve transmission delay on networks that exhibit out-of-order delivery of data, such as internet protocol (IP) networks.

Redundant Slices

The H.264 encoder can choose to transmit redundant information for specific regions of pictures [68] [69]. This feature allows extra added resilience to errors. If the primary information of the given area is lost during the transmission, the redundant information can be used to help reconstruct the lost region [70].

Data Partitioning

Similar to MPEG-4, H.264 utilizes data partitioning to add increased error resilience. The H.264 adds more partitions than defined in MPEG-4 Part 2. This additional partitioning adds to the coding flexibility and error-resilience. The data partitions in H.264 are listed in Table 2.4.

Table 2.4
Data partitions in H.264.

Partition	Data
0	Picture or Slice Headers
1	Macroblock Header Information
2	Motion Vector Data
3	Coded Block Pattern
4	DC Coefficients
5	Luma AC Coefficients
6	Chroma AC Coefficients
7	End_of_stream_symbol

2.4 Profiles and Applications

In order to achieve compliance of video coding standards, the standards define various levels and profiles. A profile specifies the coding tools allowed under that level, such as B-pictures, entropy coding type, or interlaced support. A level specifies limits on coding parameters, such as sample depth, resolution, and coded data rate.

2.4.1 MPEG-2 Profiles

The MPEG-2 standard defines several profiles to suit almost any target application. Table 2.5 outlines the profiles (and levels) defined in the standard. Of interest is the MPEG-2 Main Profile at Main Level, or MP@ML, which is used in commercial DVDs. Broadcast high-definition television uses the Main Profile at High Level, or MP@HL.

2.4.2 H.264 Profiles

The H.264 standard defines three profiles:

Baseline Profile The Baseline profile contains the aforementioned features, with the exception of B-slices, CABAC, and interlaced coding tools. Its target applications are low in complexity, such as video conferencing.

Main Profile In applications that require the highest coding efficiency and allow a larger coding delay, such as video streaming and entertainment applications, the Main profile is used. The Main Profile adds the B-slices, CABAC, and interlaced coding tools.

Extended Profile The goal of the Extended profile was to combine the flexibility of the Baseline Profile with the efficient coding tools found in the Main profile. Target applications, such as flexible video streaming, would require the use of "trick modes."

Table 2.5
MPEG-2 Profiles. This chart shows supported chrominance subsampling, maximum resolution, maximum data rates, supported picture types, and picture rate.

High	-	4:2:0 1920x1152 80 Mb/s I,P,B 60 Hz	-	-	-	4:2:0, 4:2:2 1920x1152 100 Mb/s I,P,B 60 Hz
High 1440	-	4:2:0 1440x1152 60 Mb/s I,P,B 60 Hz	-	-	4:2:0 1440x1152 60 Mb/s I,P,B 60 Hz	4:2:0, 4:2:2 1920x1152 100 Mb/s I,P,B 60 Hz
Main	4:2:0 720x576 15 Mb/s I,P 30 Hz	4:2:0 720x576 15 Mb/s I,P,B 30 Hz	4:2:0 720x608 50 Mb/s I,P,B 30 Hz	4:2:0 720x576 15 Mb/s I,P,B 30 Hz	-	4:2:0, 4:2:2 720x576 20 Mb/s I,P,B 60 Hz
Low	-	4:2:0 352x288 4 Mb/s I,P,B 30 Hz	-	-	-	-
Level/ Profile	Simple	Main	4:2:2 Profile	SNR	Spatial	High

Table 2.6
H.264 defines three profiles designed to suit a wide variety of applications.

H.264 Feature	H.264 Profile		
	Baseline	Main	Extended
I/P-pictures	•	•	•
B-pictures		•	•
CAVLC	•	•	•
CABAC		•	
Interlaced Support		•	
FMO, ASO, RS	•		•
Data Partitioning			•
SP/SI-pictures			•

Table 2.6 outlines the major differences between the three defined profiles in the H.264 standard. It should be noted that each of these profiles is neither a subset nor superset of each other. Specifications for a Professional Extension Amendment [71] is already being proposed, with more coding tools to allow higher video fidelity. Some key features being researched include support for RGB / XYZ color spaces, 4:4:4 encoding, 12-bit sample accuracy, and a 8×8 transform. A general overview of potential technical hurdles is given in [72].

An efficient way to represent film grain is also being proposed [73]. By transmitting the film grain information separately as side information, bandwidth is not wasted trying to compress the film grain. Initial results show compression gains up to 6.8× (85% reduction in data rate) for equal quality video [74].

3. EXPERIMENTAL RESULTS

This chapter presents experiments that were performed to measure the performance of H.264 relative to MPEG-2. A variety of test sequences were used, including those used by the Video Quality Experts Group (VQEG) in [75]. Each sequence was compressed with MPEG-2 and H.264 at several spatial and temporal resolutions and varying data rates. The main results can be seen in the form of rate-distortion (RD) curves for each sequence at a given resolution.

3.1 Testing Method

3.1.1 Software Used

The reference software used in this experiment was limited to standard software used in video coding research. By using these standard software packages, experimental results can be compared and reproduced by others.

MPEG-2 Reference Software

For the creation of MPEG-2 compliant bitstreams, the MPEG-2 reference model Test Model 5 (TM5) [76] was used. The latest version 1.2 was utilized. The reference software accepts 4:2:0 uncompressed $YCrCb$ video as input and creates MPEG-2 video elementary streams in the format of `*.m2v` files. Through an input directive file, most of the common parameters can be controlled, including GOP structure. For rate-control, the TM5 algorithm accepts a data rate in terms of bits per second as input. Using its own rate-control model, TM5 dynamically adjusts the quantization level of each macroblock to control overall data rate.

Table 3.1
Overview of MPEG-2 configuration.

Parameter	Progressive Video	Interlaced Video
DC precision	8 bits	8 bits
DCT Scan Order	Normal Zig-Zag	Alternate
Interprediction	Frame	Field

The reference software was modified slightly to accept a single concatenated $YCrCb$ file as input, as opposed to one file per input frame. This allows both the TM5 reference software and the JM reference software to use the same input file.

For encoding parameters, the GOP length was set to $N = 15$ with $M = 3$. This inserts a B-picture in between every I/P-picture. A single GOP would have the following structure: IBPBPBPBPBPBPBP. 8-bit precision was used for the encoding of the DC coefficient. Error resilient features, such as intra slice refresh and concealment motion vectors, were all disabled. Rate control parameters were all initialized to the default values of zero. Frame prediction and normal zig-zag scanning were used for all progressive input sequences. Field prediction and alternate scanning were used for all interlaced input sequences. An overview of these features is listed in Table 3.1.

H.264 Reference Software

The H.264 joint-reference model (JM) was utilized for the creation of H.264 video streams. The version used for these experiments was JM v7.5b (the latest version at the time of writing this thesis was v9.0). Many features in the JM reference software are unstable. Generally, the encoding parameters were chosen to maximize encoding quality at the expense of increased encoding complexity. Error-resilient tools were disabled. The rate control algorithm in the reference model [77] controls data rate at

the macroblock level by automatically adjusting the quantization parameter (QP), which is an index to a table of quantization step sizes defined in the standard. This can provide up to 0.67 dB [78] PSNR over constant QP encoding for equivalent data rate. This rate control scheme was not utilized due to instability reasons. More details on encoding techniques employed in the JM reference software is described in [79].

Similar to TM5, the GOP structure in the H.264 encoder we used was IBPBPBPBPBPBPBP. Temporal Direct prediction for B-pictures was enabled. For maximum encoding quality, rate-distortion optimized macroblock mode decision was enabled [80] and the Hadamard transform was utilized for motion estimation. Unless otherwise specified, 5 reference frames were used (the maximum allowed by the JM reference software). Quantization for each picture was set at the picture level, with the quantization parameter (QP) for each B-picture was set to 2 higher than I/P QP. That is, $B_{QP} - 2 = I_{QP} = P_{QP}$, which results in the quantizer for B-pictures to be approximately 25% larger.

3.1.2 Measuring Quality of Digital Video

Since digital video is considered to be "consumable" information, that is, humans are said to consume video, the ultimate judge of video quality is a human observer. But for scientific purposes, a good scientific evaluation involving human subjects is very costly. An alternative method will be used for these experiments.

In order to provide some objective and reproducible estimate of perceived video, two full-reference (FR) metrics will be used. An FR metric is a metric that uses the original uncompressed source video to compare against the target compressed video. One such metric that is very widely used today is peak signal-to-noise ratio (PSNR). It is used because it requires a simple calculation and is well-suited mathematically for optimization purposes.

The other metric utilized is a relatively newer metric and is less widely used. The second metric is based on the structural similarity index (SSIM) of a video sequence [81].

For all experiments and measurements of distortion, only the luminance component (Y) was considered for simplicity.

Obtaining of MSE and PSNR of Compressed Sequences

Given two discrete, 2-D signals $x(m,n)$ and $y(m,n)$, each with dimensions $M \times N$, the mean square error (MSE) of the two signals is obtained by

$$MSE(x,y) = \frac{1}{MN} \sum_{j=1}^{N} \sum_{i=1}^{M} (x_{ij} - y_{ij})^2. \tag{3.1}$$

The peak signal-to-noise ratio is given by

$$PSNR(x,y) = 10 \log_{10} \frac{L^2}{MSE}, \tag{3.2}$$

where L is the dynamic range of the input signals. For all of the input video sequences here, each sample is represented with 8-bits sampling precision and therefore $L = (2^8 - 1) = 255$. Higher PSNR implies better visual quality.

Determination of the SSIM for a Video Sequence

For this thesis, a variant of the algorithm described in [81] was employed. SSIM is a metric with values between 0.0 and 1.0, with 1.0 representing the best video quality. The SSIM of two signals x and y is given by the following:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{3.3}$$

where

$$\mu_x \quad = \quad \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{3.4}$$

$$\mu_y = \bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{3.5}$$

$$\sigma_x^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2 \tag{3.6}$$

$$\sigma_y^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{y})^2 \tag{3.7}$$

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y}) \tag{3.8}$$

and C_1 and C_2 are arbitrary constants given by

$$C_1 = (K_1 L)^2 \tag{3.9}$$

$$C_2 = (K_2 L)^2, \tag{3.10}$$

where $L = 255$ represents the dynamic range of the signals. $K_1 = 0.01$ and $K_2 = 0.03$ were set to match the values used in the original paper [81].

In order to obtain the SSIM for a single picture in a video sequence, the SSIM for every possible 8×8 overlapping window is determined and the mean is determined over the total number of overlapping windows in the picture. In order to obtain the SSIM for the overall video sequence, each picture SSIM value is uniformly averaged over the total number of pictures in the sequence.

For simplification purposes, only the luminance channel was considered and all 8×8 windows were weighted equally. The windows were not weighted according to local motion. Results in [81] imply that motion-weighted windows provide marginal gains over non-motion-weighted windows.

3.2 Testing Each Feature in H.264

These first set of tests are used to roughly illustrate the coding gain of various new features in H.264.

In order to illustrate the gains provided by each new main coding feature in H.264, first a "baseline" or basic configuration is created. Many of of the advanced

coding features are disabled within the encoder. Several tests then were performed, with each a new main feature enabled in addition to the baseline profile.

The main coding features will be categorized into these main categories: advanced entropy coding, in-loop filtering, flexible motion compensation, and multiple reference frame capabilities. Although it is the sum of these smaller improvements add together to provide increased coding gain, these tests are useful to illustrate the contribution of each new feature.

The input sequences for these tests consist of the 'akiyo' sequence and the 'foreman' sequence. These two sequences cover two main types of video. The first is a head and shoulder shot with very little motion. The second is considered to have complex motion and therefore it is more challenging to be efficiently encoded using motion-compensated video coding standards, e.g. MPEG-2 and H.264.

3.2.1 H.264 Test Configurations

The baseline configuration uses only 16×16 macroblocks, with the other smaller sizes disabled. The number of reference frames are limited to 2 pictures (to accommodate for B-pictures). Context Adaptive Variable Length Coding (CALVLC) is used for entropy coding. The in-loop filter is disabled.

The common encoding parameters include rate-distortion optimized macroblock mode selection, hadamard transform, and 1 B-picture coded between every I/P-picture. All of the error resilient encoding features, such as redundant slices, were disabled. An overview of the H.264 encoder configurations for these tests is shown in Table 3.2.

An summary of the results of testing each coding feature in H.264 can be found in Table 3.3. Test show a small but measurable improvement for each coding feature in H.264. In each case, The last row titled 'All Features' compares the "base" H.264 profile to the "full" profile.

Table 3.2
Overview of H.264 Configurations.

	H.264 Configuration					
H.264 Feature	Base	#1	#2	#3	#4	Full
Smaller MBs		•				•
Entropy Coding	CAVLC	CAVLC	CAVLC	CABAC	CAVLC	CABAC
# of Reference pictures	2	2	2	2	5	5
In-Loop Filter			•			•

Table 3.3
Each feature in H.264 contributes a small but measurable improvement. Larger numbers indicate better performance.

Coding Feature	Δ PSNR (dB)	Data rate Savings (% kbps)
CABAC Entropy Coding	0.0	8.2
In-Loop Filter	0.1	1.2
Flexible Macroblock Sizing	0.2	3.8
Multiple Reference Pictures	0.0	2.2
All Features	0.5	0.2

Each sequence was encoding using the reference picture profile and compared to the results using the baseline profile.

3.3 Experiment 1: Test of the Additional Macroblock Sizing

The first test attempts to illustrate the coding gain achieved by the increased number of macroblock sizes. For each sequence in this test, two R-D curves were created: one curve for the base H.264 configuration and one curve with the additional macroblock sizing enabled. Figures 3.1 and 3.2 illustrate the coding efficiency achieved by permitting the use of additional macroblock sizes.

3.4 Experiment 2: Test of the H.264 In-Loop Filter

The second test attempts to illustrate the gains achieved by the usage of the loop filter. The loop-filter H.264 configuration is identical to the base configuration, with the exception of the usage of the in-loop filter during the motion-compensation process.

Figures 3.3 and 3.4 illustrate the coding efficiency achieved by use of the in-loop deblocking filter.

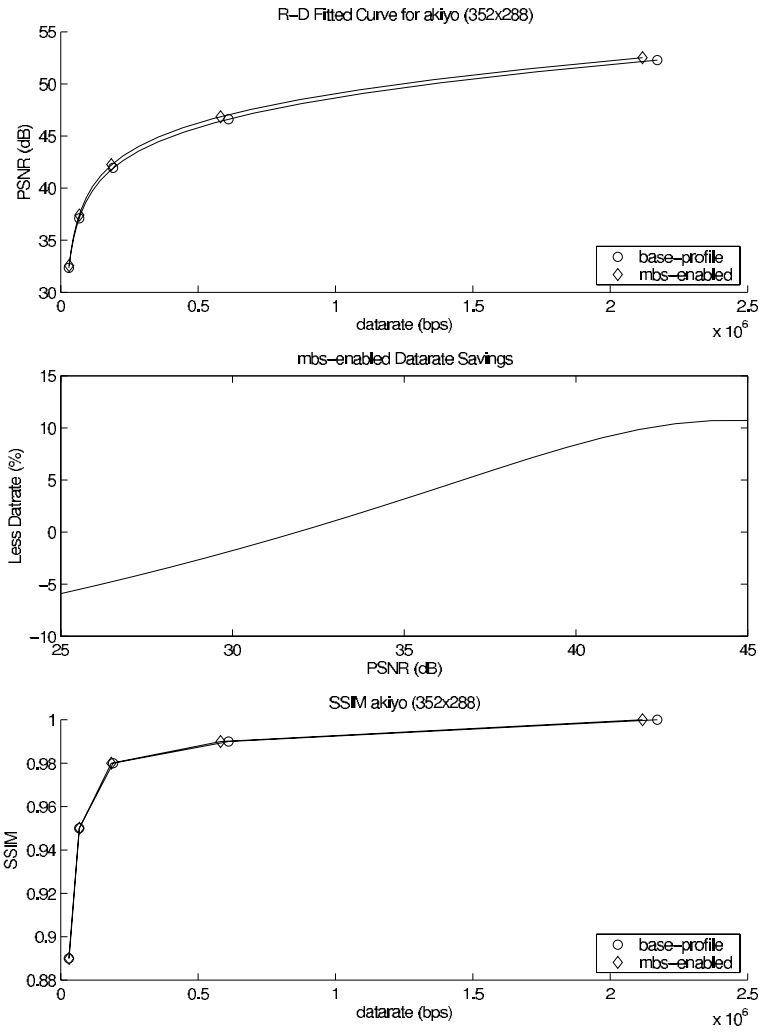


Fig. 3.1. Data rate savings of additional macroblock sizing in ‘akiyo’.

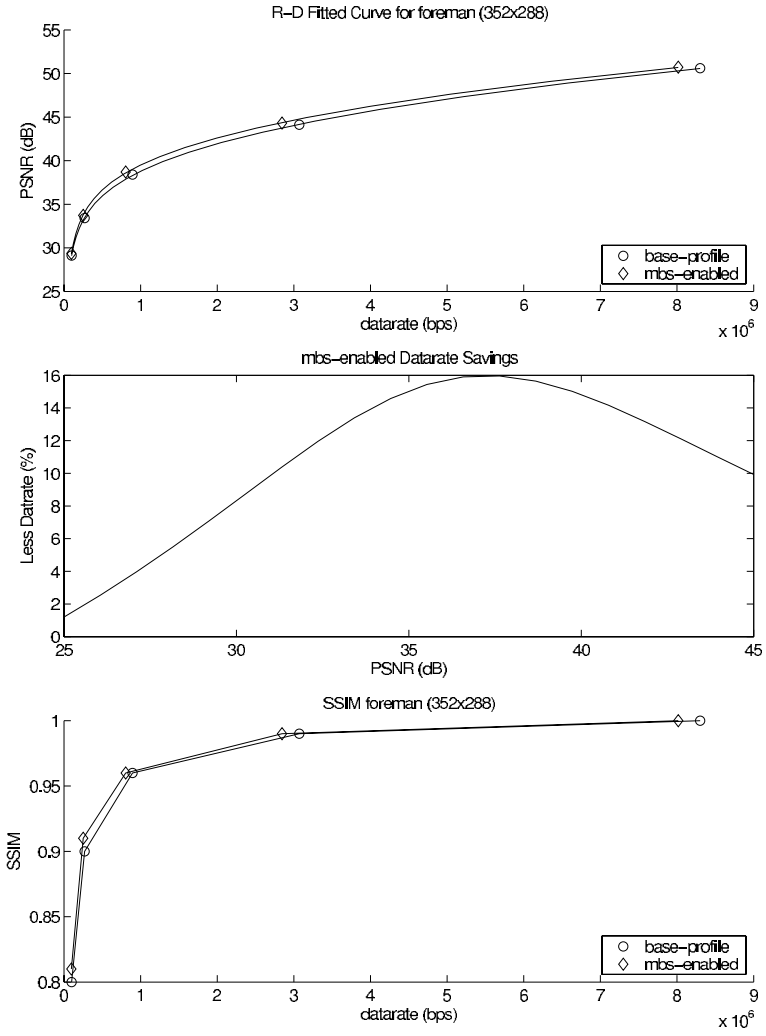


Fig. 3.2. Data rate savings of additional macroblock sizing in 'foreman'

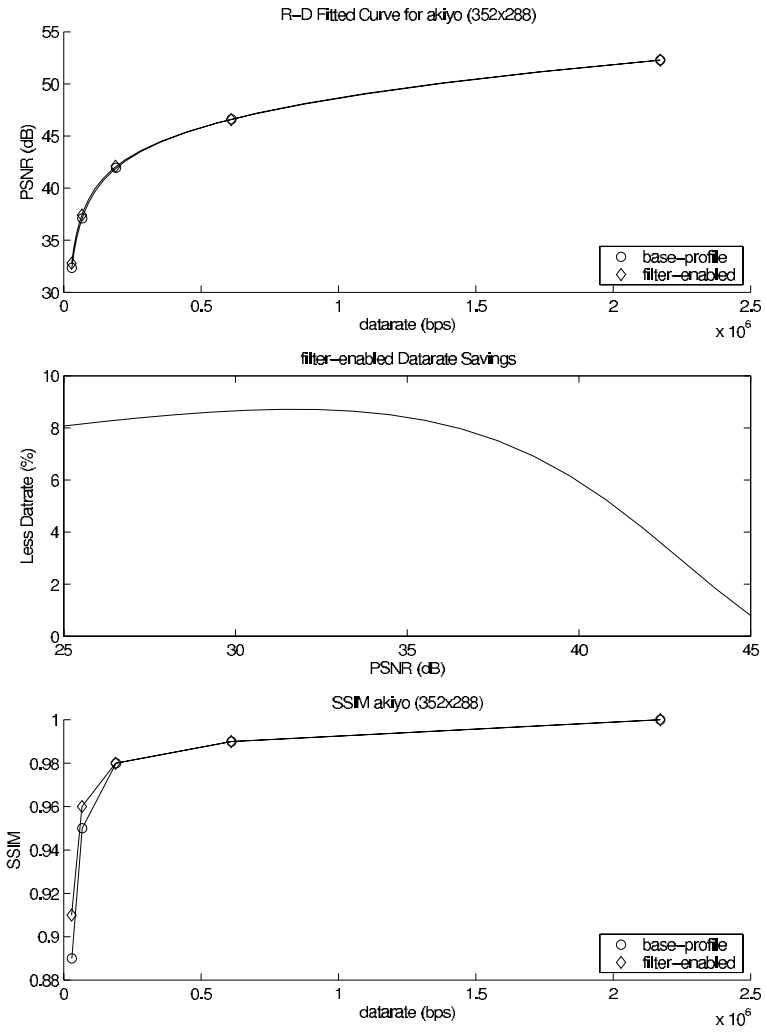


Fig. 3.3. Data rate savings of in-loop filter in ‘akiyo’.

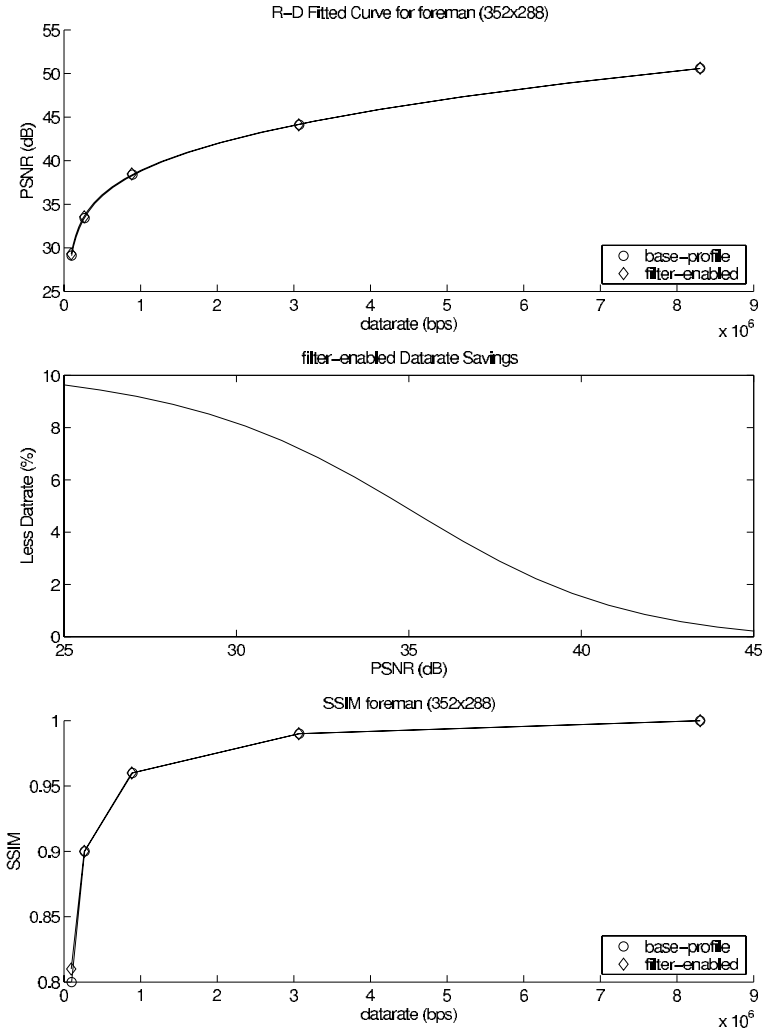


Fig. 3.4. Data rate savings of in-loop filter in 'foreman'

3.5 Experiment 3: Test of Context-Adaptive Binary Arithmetic Coding

The next test attempts to illustrate the coding gain achieved by the usage of the more advanced entropy coding available in H.264. The H.264 standard specifies two types of entropy coding of the transform coefficients. The first is context adaptive variable length coding, or CAVLC, which is used in the normative H.264 Baseline Profile. The other entropy method specified in the standard is context adaptive binary arithmetic coding, or CABAC, which is used in the normative H.264 Main Profile. CABAC provides increased coding efficiency by requiring less bits to represent the same information as CABAC at the expense of more complex encoding and decoding.

Each sequence was encoding using the CABAC-enabled profile and compared to the results using the baseline profile. Figures 3.5 and 3.6 illustrate the coding efficiency achieved by use of CABAC entropy coding compared with CAVLC entropy coding.

3.6 Experiment 4: Test Using Increased Number of Reference Pictures

The next test attempts to illustrate the increased efficiency provided by increased number of reference pictures. Previous standards permitted only a maximum of 2 reference pictures for each predictively coded picture (1 for P-pictures and 2 for B-pictures). The H.264 standard relaxes this restriction and permits up to 15 reference pictures for predictively encoded pictures. The increased number of reference pictures should provide increased coding efficiency. Each predictive picture has a larger amount of references available, which should provide more accurate prediction and thereby reduce the energy of the prediction error signal. Less energy in the prediction error signal Each sequence was encoding using the reference picture profile and compared to the results using the baseline profile. translates into less required residual coding of the error signal. Figures 3.7 and 3.8 illustrate the coding efficiency achieved by increasing the number of reference pictures from 2 to 5.

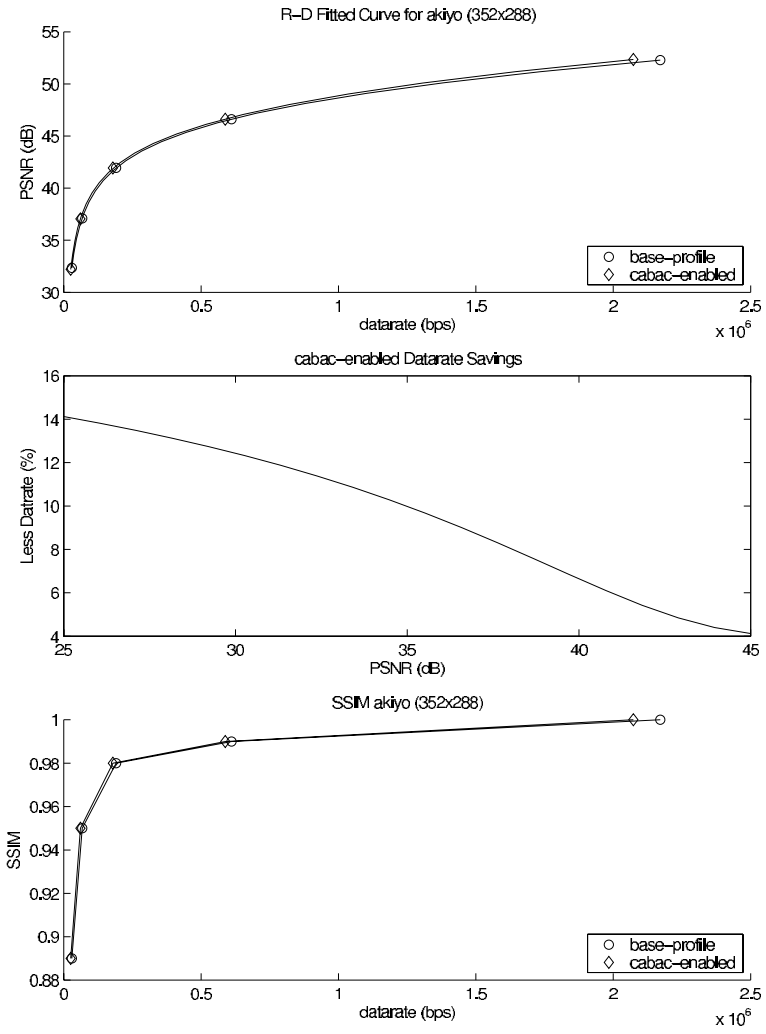


Fig. 3.5. Data rate savings of CABAC in 'akiyo'.

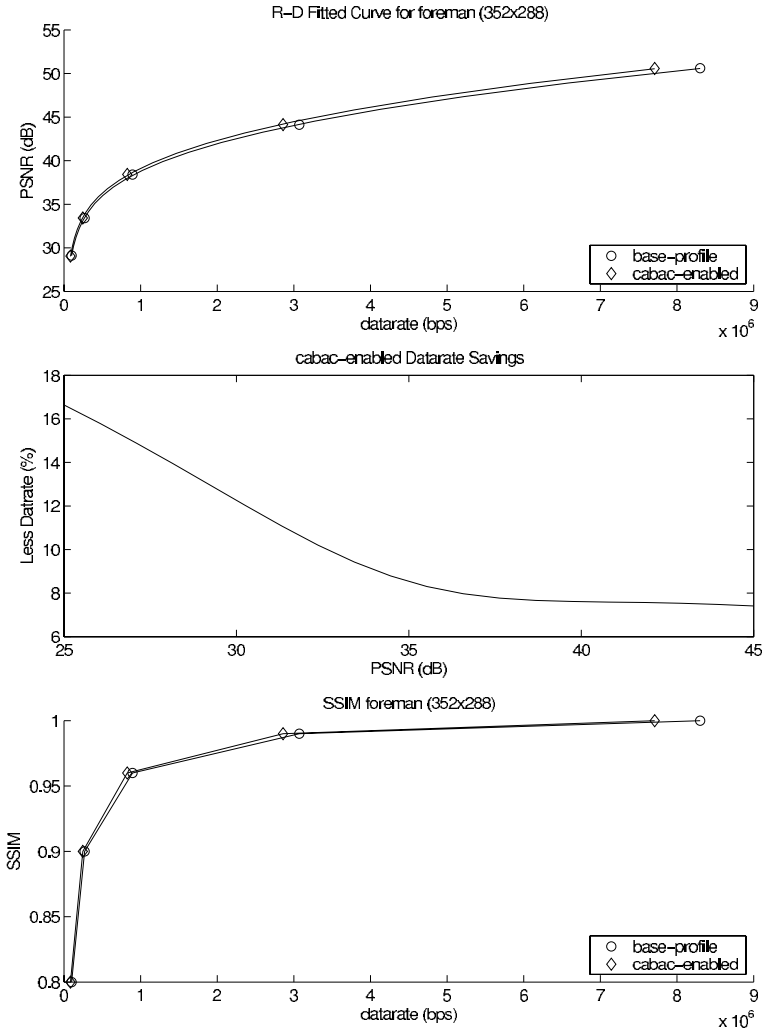


Fig. 3.6. Data rate savings of CABAC in 'foreman'.

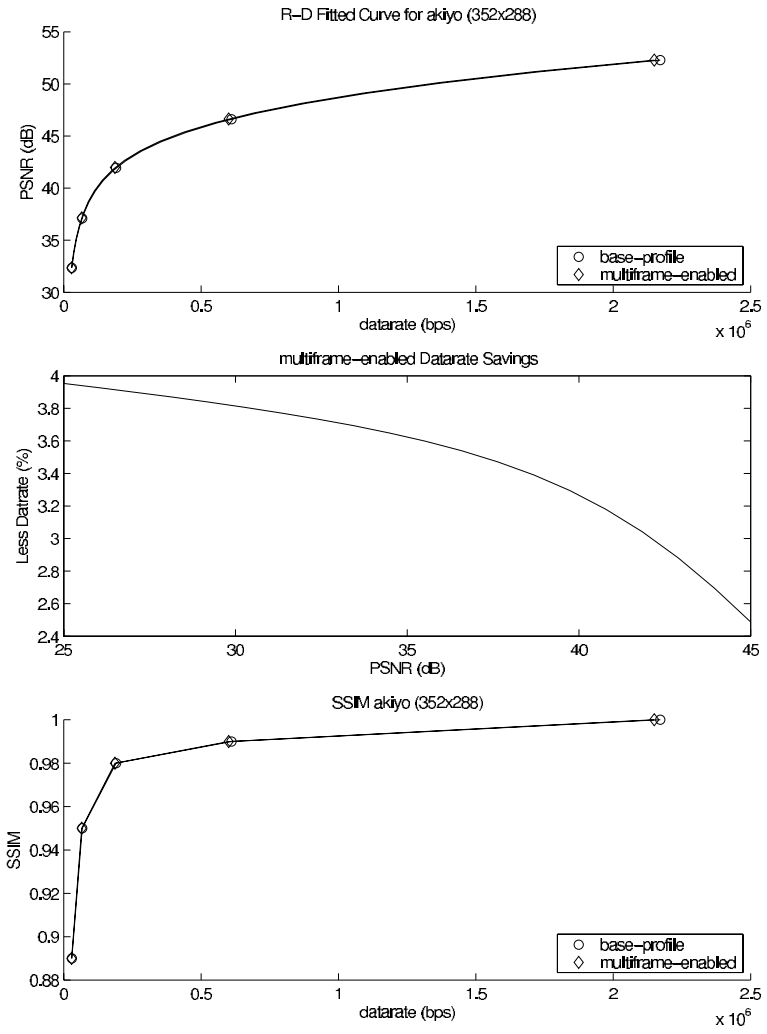


Fig. 3.7. Data rate savings of additional reference pictures in ‘akiyo’.

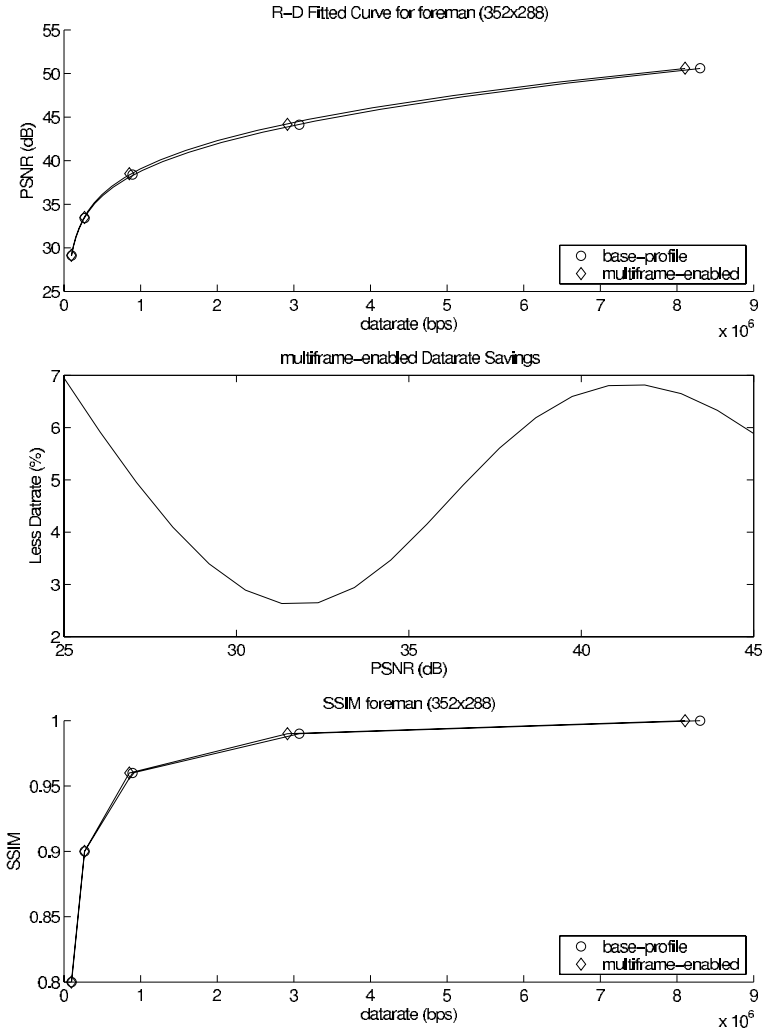


Fig. 3.8. Data rate savings of additional reference pictures in ‘foreman’.

3.7 Experiment 5: Test of Combination of Features of H.264

For this experiment, Figures 3.9 and 3.10 illustrate the coding efficiency achieved by use of the "full" profile, which enables additional macroblock sizing, in-loop filtering, CABAC, and multiple reference pictures.

3.8 Experiment 6: MPEG-2 versus H.264 Video Compression

The final suite of experiments compares H.264 with MPEG-2. For these set of experiments, each input sequence was compressed using the JM video encoder with the QP pre-selected due to lack of stable rate control functionality in the JM reference software. QP for I and P pictures set to 8, 16, 24, 32, and 40. QP for B-pictures was set to 2 higher than the I/P QP, e.g. if $I_{QP} = P_{QP} = 16$, then $B_{QP} = 18$. These values were chosen to cover a wide span of input data rates. Each resulting H.264 sequence data rate was used as an input into the TM5 software's built in rate control. Therefore, the MPEG-2 and H.264 encoders created similar data rate streams which allows for more direct PSNR comparison.

Experimental results show that H.264 is far superior to MPEG-2 in terms of rate-distortion performance. Full results are given in the appendix. Table 3.4 shows the average quality improvement of H.264 compared with MPEG-2 measured in PSNR. The average was obtained by the arithmetic mean of the change in PSNR values. Note that is not exactly representative of the results due to the highly non-linear nature of the rate-distortion curves. For more accurate data, the reader is invited to refer to the appendix for complete rate distortion plots and full table listings of all data.

Informal subjective tests show H.264 to be far superior to MPEG-2, especially at lower data rates, e.g. less than 500 kb/s. This is in agreement with our objective evaluation. At low data rates, the video appears to be very blocky due to the heavy quantization of the DCT coefficients. The blockiness manifests itself in the video as visible blocks in the image. At lower data rates, the blockiness is not as

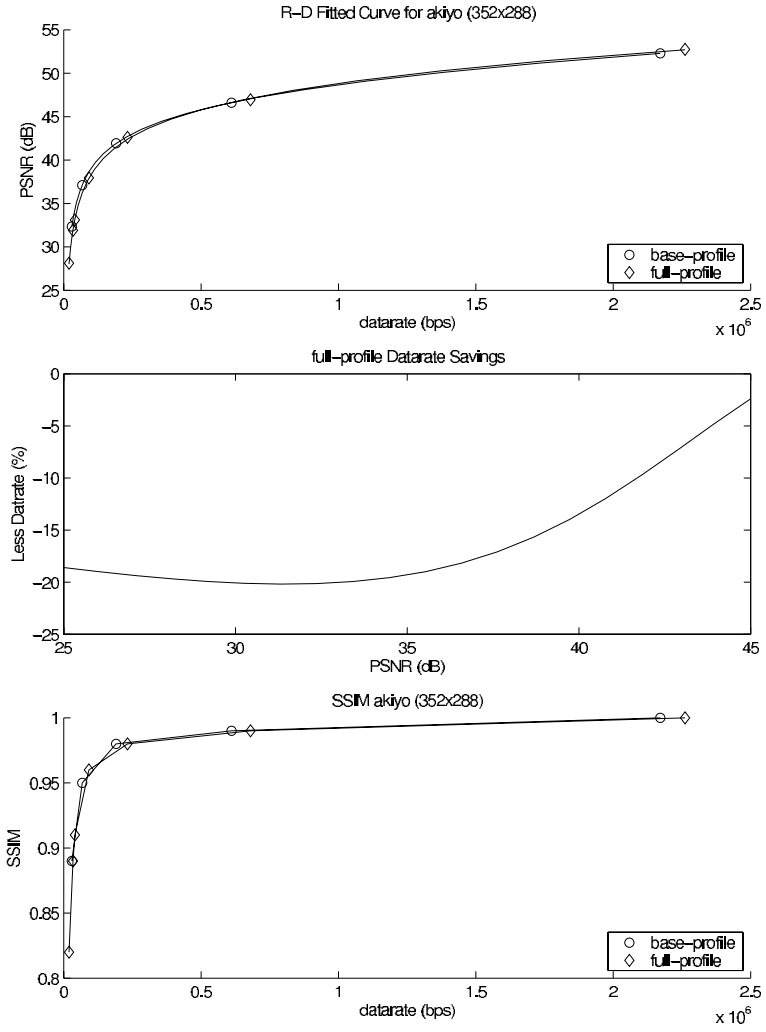


Fig. 3.9. Data rate savings of full features in 'akiyo'.

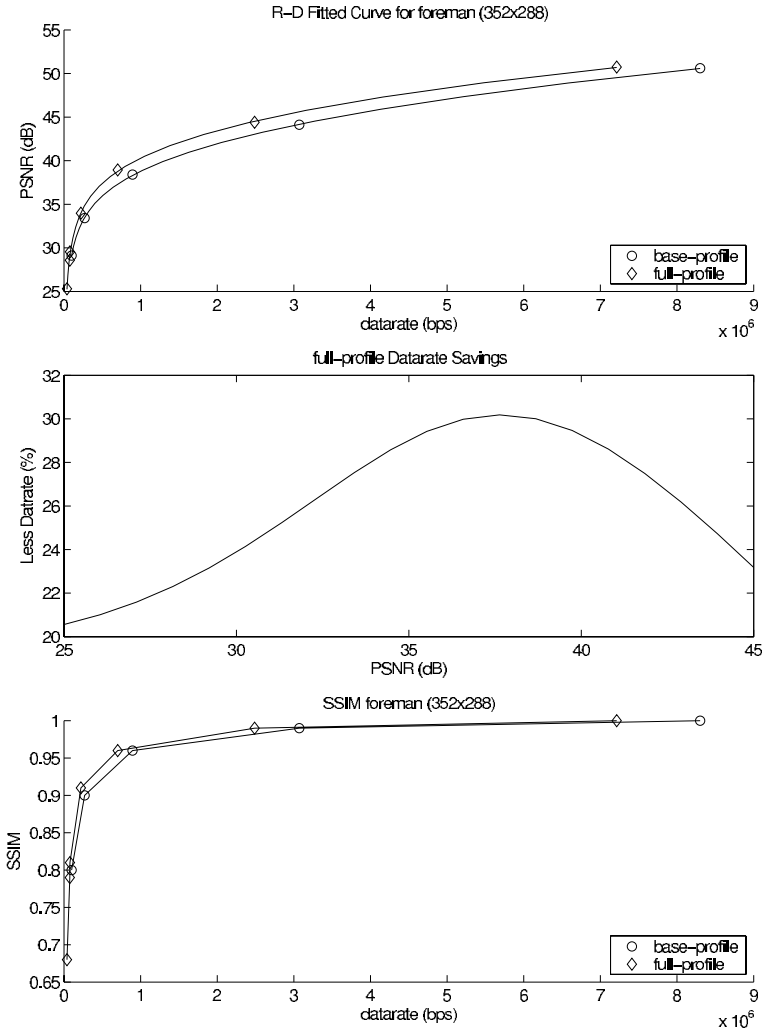


Fig. 3.10. Data rate savings of full features in 'foreman'.

Table 3.4
Summary of PSNR improvement for H.264 video. Results were summarized according to resolution. VQEG input sequence results were separated for comparison purposes.

Input	Mean PSNR Δ (dB)
QCIF 176×144	5.8
CIF 352×240	4.76
CIF 352×288	3.98
720×576 VQEG	4.39
720×480 VQEG	5.74
720×480 Other	4.90

apparent due to the superior compression of H.264 as displayed in Figure 3.11(a) and Figure 3.11(b). As the data rate is increased, the visual difference between MPEG-2 and H.264 is less apparent, as shown in Figure 3.12(a) and Figure 3.12(b). This observation is true over all the input sequences.

Of interest is the trends in the rate-distortion plots. At higher data rates, both MPEG-2 and H.264 scale fairly linearly. At lower data rates, however, the behavior of each coding standard differs. Generally, MPEG-2 quality falls off at a faster exponential rate as the data rate is decreased as compared to H.264. Therefore, at extremely high data rates, a viewer may or may not be able to distinguish between H.264 and MPEG-2 video, but at very low data rates, the difference should be quite obvious. It should also be noted that MPEG-2 can not achieve the extremely low data rates possible with H.264.

The results have been grouped for each resolution in Figures 3.13–3.18 to show the overall data rate savings of H.264 over MPEG-2. The RD curve for each individual sequence can be found in the appendix.

(a) MPEG-2

(b) H.264

Fig. 3.11. The CIF 'carphone' sequence compressed to 246 kb/s using MPEG-2 and H.264. Severe blocking is visible in the MPEG-2 sequence.

(a) MPEG-2

(b) H.264

Fig. 3.12. The CIF 'carphone' sequence compressed to 708 kb/s using MPEG-2 and H.264. The visual difference between the two sequences is less apparent at higher data rates.

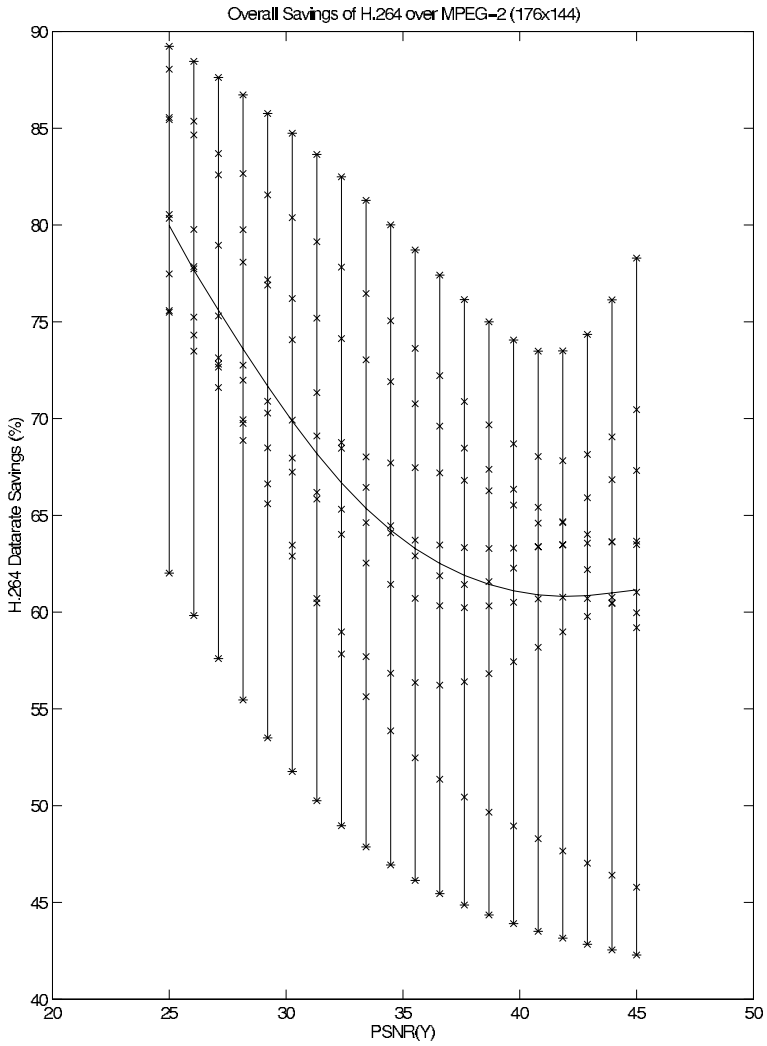


Fig. 3.13. Overall data rate savings for QCIF (176×144) 30 fps sequences.

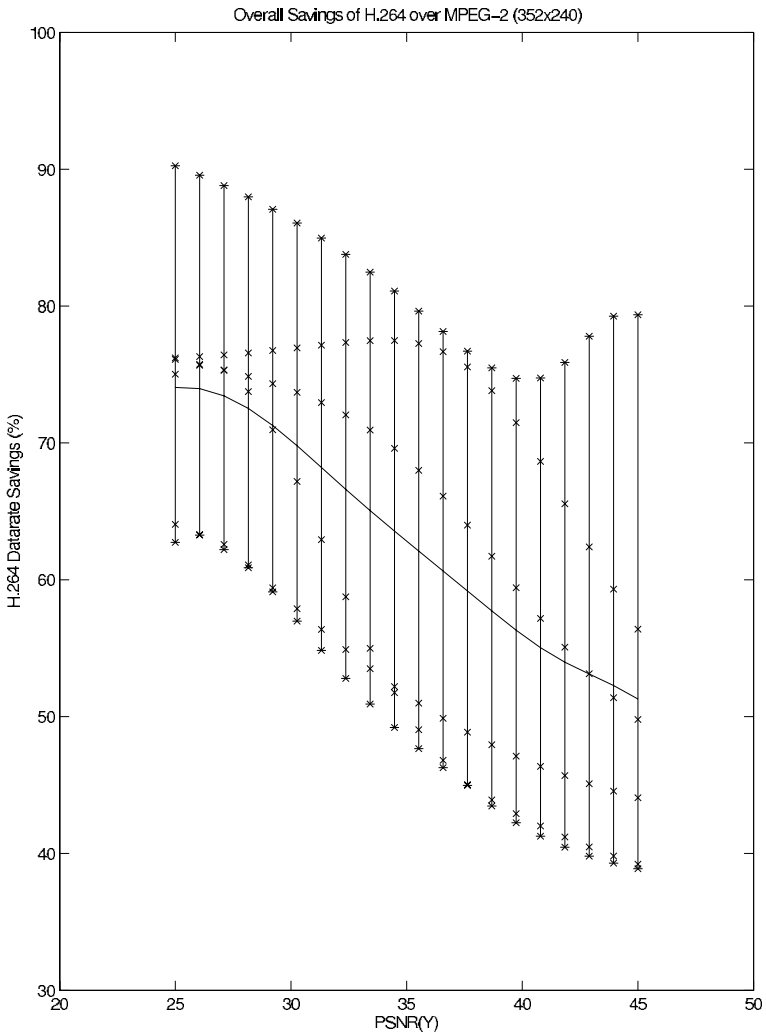


Fig. 3.14. Overall data rate savings for CIF (352×240) 30 fps sequences.

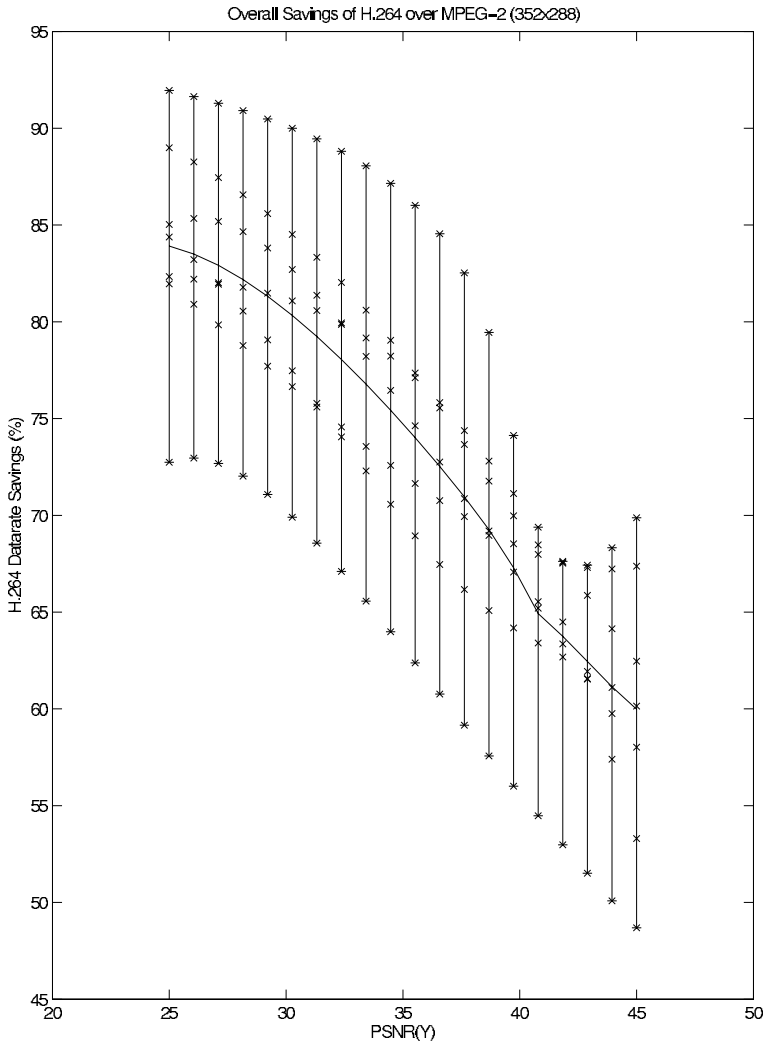


Fig. 3.15. Overall data rate savings for (352×288) 30 fps sequences.

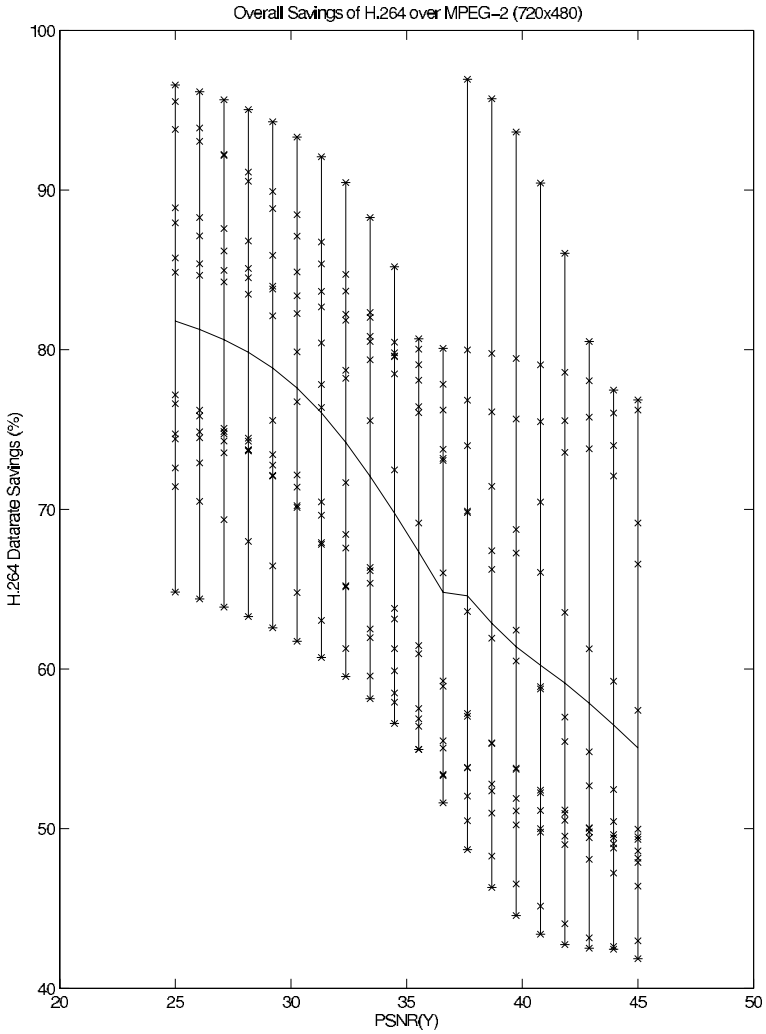


Fig. 3.16. Overall data rate savings for (720×480) 30 fps sequences.

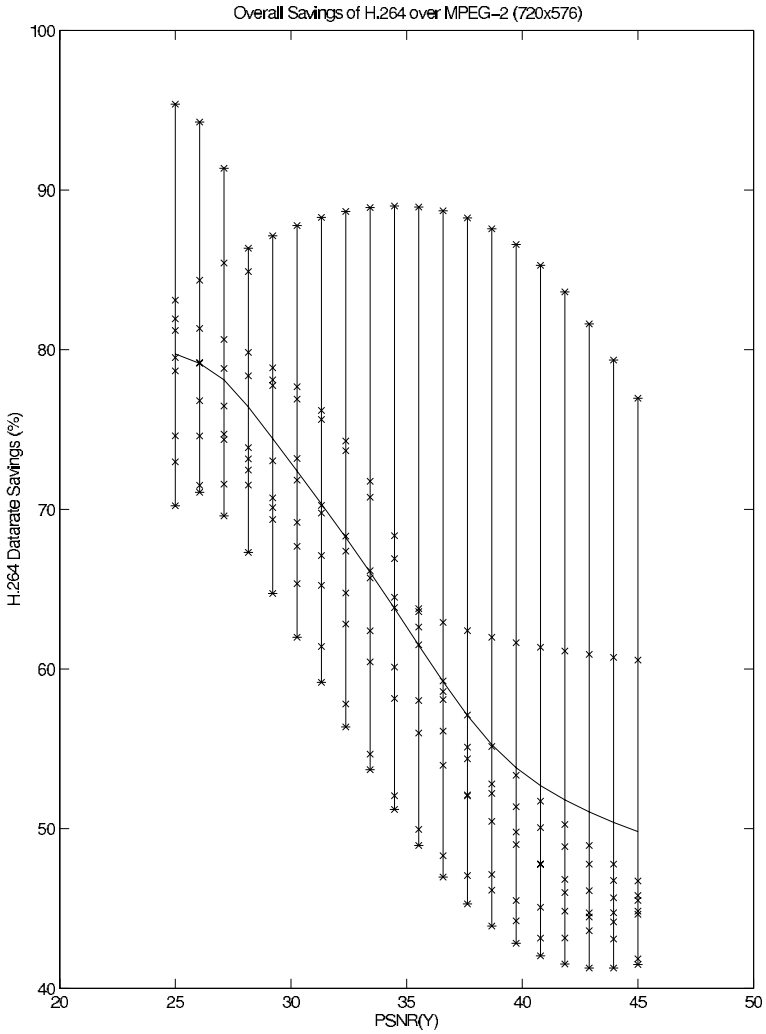


Fig. 3.17. Overall data rate savings for (720×576) 30 fps sequences.

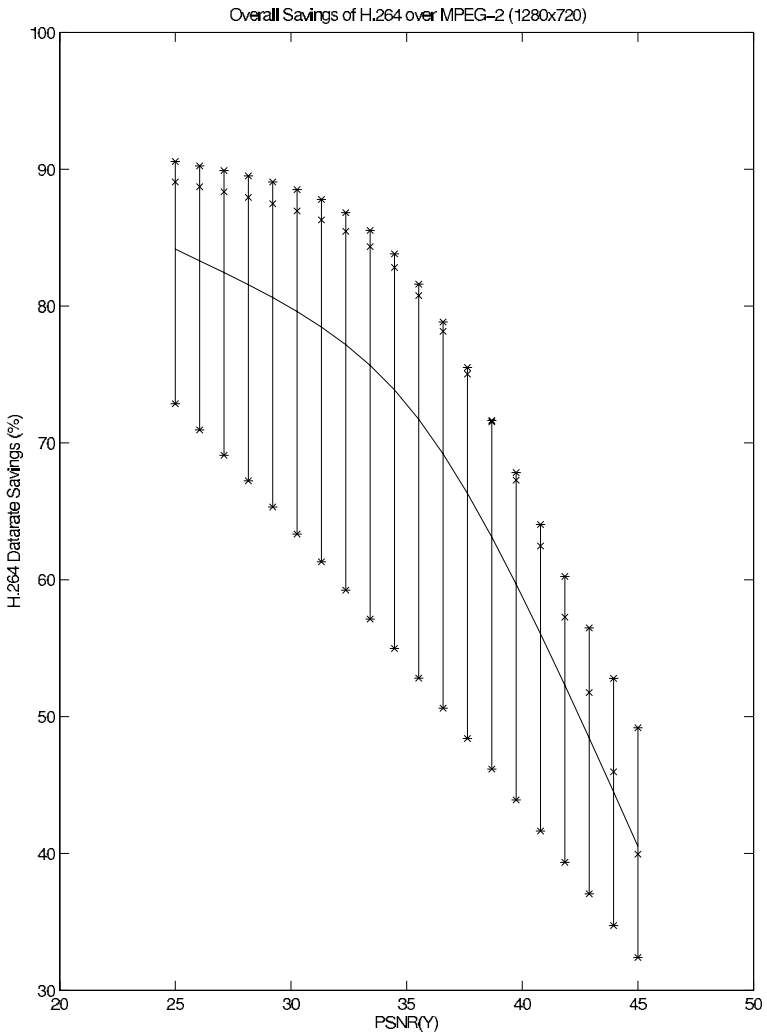


Fig. 3.18. Overall data rate savings for (1280 × 720) 25 fps sequences.

4. CONCLUSIONS

This thesis presents an overview of the differences between the MPEG-2 video coding standard and the recently finalized H.264 video compression standard. MPEG-2 made popular the general framework of a hybrid coder. The H.264 standard takes the basic building blocks of MPEG-2 and improves on every aspect, from the drift-free transform to the advanced entropy coding.

From the experiments it is shown that H.264, compared to MPEG-2, offers significant data rate savings for similar quality video. In turn, it offers significant higher quality video given similar data rates. These results have been shown through two objective video quality metrics: the popular PSNR metric and the newer SSIM metric.

4.1 Suggestions for Future Work

Possibly the largest improvement that could be made to the experiments presented here is the inclusion of formal human observer experiments. Since the humans are the final deciding factor in the overall quality of a compressed video, it is only logical that human subjects be employed as a quality metric. As an experiment, numerous human subject viewers could be used to experimentally determine the required data rate to achieve similar quality H.264 video given a compressed MPEG-2 video. This experiment would also take into consideration factors that were not considered in the experiments here, e.g. subjective preference.

Another area of concern is the inclusion of the in-loop filter. Informal tests indicated some tested subjects actually prefer H.264 video without the loop filter, when comparing two identical input sequences with equal compressed measured distortion and data rate. This preference was more prevalent at extremely low data rates when

the filtering was extremely visible. This may indicate more work may be needed to be performed in optimizing the parameters for the loop filter, e.g. optimization of the filtering strength and thresholding, which are supported by the H.264 standard.

One area that was not emphasized during these experiments was encoding complexity. Encoding complexity was not included for simplicity. These experiments utilized two different software packages utilizing different encoding strategies, i.e. TM5 MPEG-2 software uses rate control to automatically set quantization while JM H.264 software uses fixed quantization. Generally speaking, H.264 encoding is much more complex than MPEG-2 encoding due to the vast amount of encoding options. For real world applications, encoding complexity is usually an issue.

APPENDICES

APPENDICES

Appendix A: Input Sequences

The following information describes the input video sequences used for the compression tests. When necessary, each video frame was cropped to ensure the spatial resolutions were even multiples of 16. Each video sequence used was in an uncompressed 4:2:0 $YCrCb$ format with 8 bits per sample (1 byte). The VQEG sequences listed in Tables A.7 and A.6 were downsampled from their original uncompressed 4:2:2 format. The components of each video frame is stored in a planar format. The Y, Cr, and Cb components were concatenated together for each frame. All the frames were concatenated to form the entire sequence.

The high definition sequences in Table A.5 are available from [82].

Table A.1
QCIF (176×144) resolution sequences.

Name	Frames	Description
akiyo	300	Head and shoulder shot, low movement
carphone	382	Head and shoulder shot, low movement
claire	494	Head and shoulder shot, low movement
coastguard	300	Ship moving along coastline
container	150	Ship in water, low movement
foreman	400	Man talking, complex movement
glasgow	200	Vehicle moving, complex movement
mthr-dgthr	300	Woman talking with daughter
news	300	News anchor sequence
salesman	448	Man sitting and talking

Table A.2
CIF (352×240) resolution sequences.

Name	Frames	Description
flowergarden	151	Fast panning landscape
tennis	149	Men playing table tennis
tv18_new	1200	News broadcast sequence with scene cuts
foot_cif_cr	150	Football, fast movement
news2_cr	301	Woman talking
salescut_cr	300	Man sitting and talking

Table A.3
CIF (352×288) resolution sequences.

Name	Frames	Description
akiyo	300	Head and shoulder shot, low movement
bus150	150	Bus moving
carphone	300	Head and shoulder shot, low movement
foreman	300	Man talking and pointing
mobile	300	Slow pan, global movement
news	300	News anchor
missA_cr	300	Head and shoulder shot

Table A.4
ITU-R 601 (720×480) resolution interlaced sequences.

Name	Frames	Description
football2	150	American football
flowergarden_cr	150	Fast pan of house and flowers
hockey1_cr	150	Ice hockey
mobilcal_cr	150	Mobile train and calendar
tennis2_cr	150	Table tennis

Table A.5
High Definition (1280×720) resolution sequences.

Name	Frames	Description
mobcal	252	Toy train on moving background
parkrun	252	Slow pan of man running in park
shields	252	Fast moving pan shot

Table A.6
VQEG PAL (720×576) resolution interlaced sequences.

Name	Frames	Description
src2	220	Parade with global zoom
src3	220	Musicians with instruments
src4	220	Computer generated screen
src5	220	Man on kayak
src6	220	Automobile racing
src7	220	Restaurant kitchen
src8	220	Computer generated text
src9	220	Soccer game
src10	220	Mobile train with calendar

Table A.7
VQEG NTSC (720×480) resolution interlaced sequences.

Name	Frames	Description
src13	260	Fairground
src14	260	Pan over city
src15	260	Mobile train with calendar
src16	260	Computer animation
src17	260	Computer generated text
src18	260	Waterfall
src19	260	American football
src20	260	Ship in harbor
src21	260	Woman talking on phone
src22	260	Flowers with zoom out

(a) akiyo

(b) carphone

(c) claire

(d) coastguard

(e) container

(f) foreman

(g) glasgow

(h) mthr-dgthr

(i) news

(j) salesman

Fig. A.1. QCIF (176×144) sequences.

(a) flowergarden

(b) foot_cif_cr

(c) news2_cr

(d) salescut_cr

(e) tennis

(f) tv18_new

Fig. A.2. CIF (352×240) sequences.

(a) akiyo

(b) bus150

(c) carphone

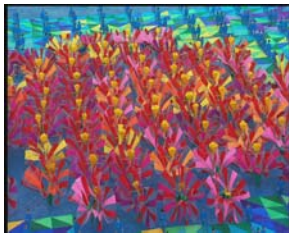
(d) foreman

(e) mobile

(f) news

(g) missA

Fig. A.3. CIF (352×288) sequences.



(a) src2

(b) src3

(c) src4

(d) src5

(e) src6

(f) src7

(g) src8

(h) src9

(i) src10

Fig. A.4. ITU-R 601 (720×576) sequences.

(a) src13

(b) src14

(c) src15

(d) src16

(e) src17

(f) src18

(g) src19

(h) src20

(i) src21

Fig. A.5. ITU-R 601 (720 × 480) NTSC sequences.

(a) mobcal

(b) parkrun

(c) shields

Fig. A.6. High Definition (1280×720) sequences.

Appendix B: Full Experimental Results

This section contains the rate-distortion plots of all of the tested sequences in Figures B.1–B.43 as well as the complete experimental results in tabular format in Tables B.1–B.18.

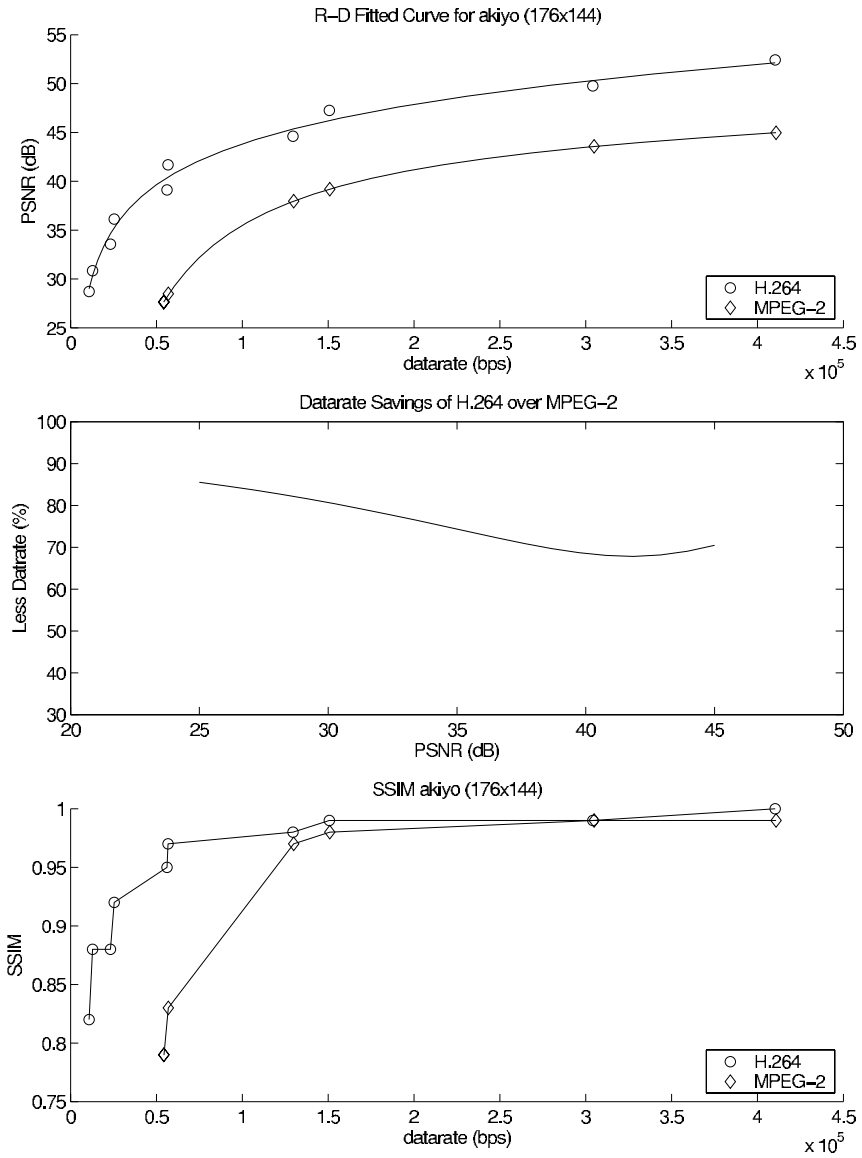


Fig. B.1. R-D Plot for akiyo sequence QCIF 30 fps.

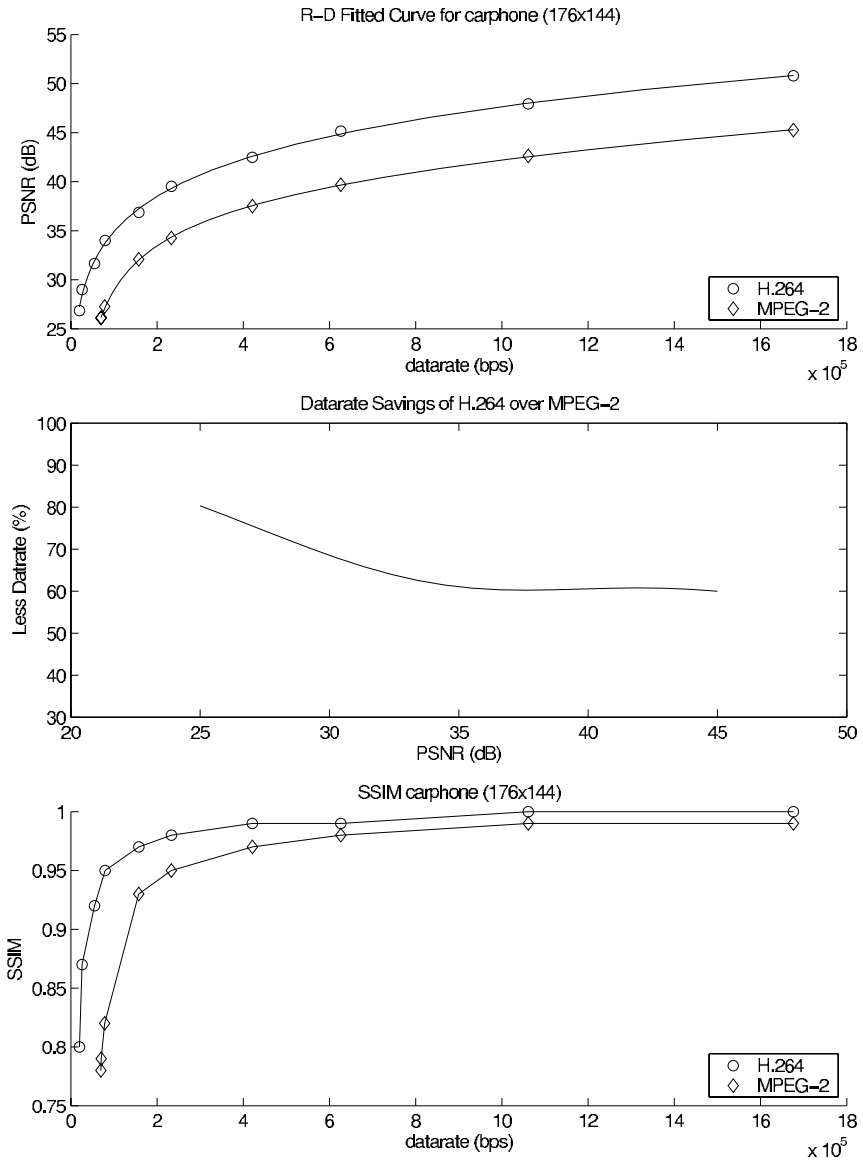


Fig. B.2. R-D Plot for carphone sequence QCIF 30 fps.

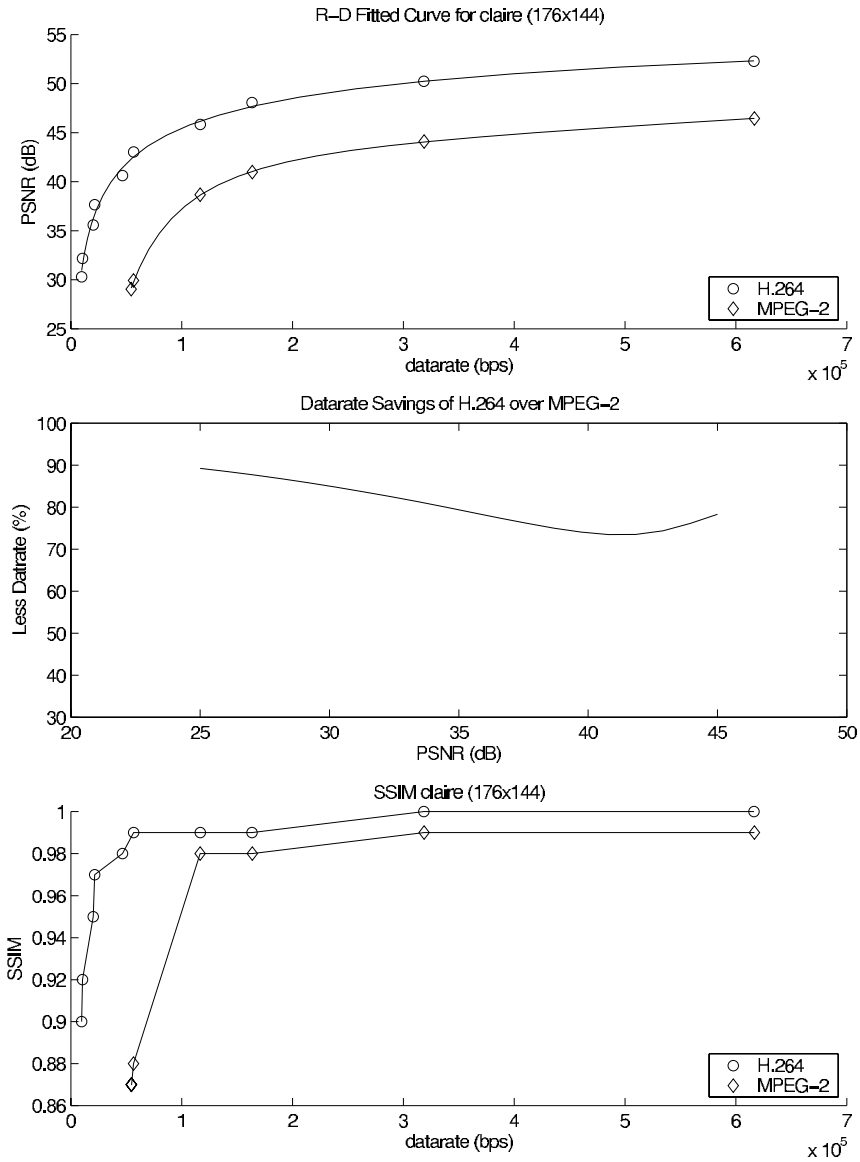


Fig. B.3. R-D Plot for claire sequence QCIF 30 fps.

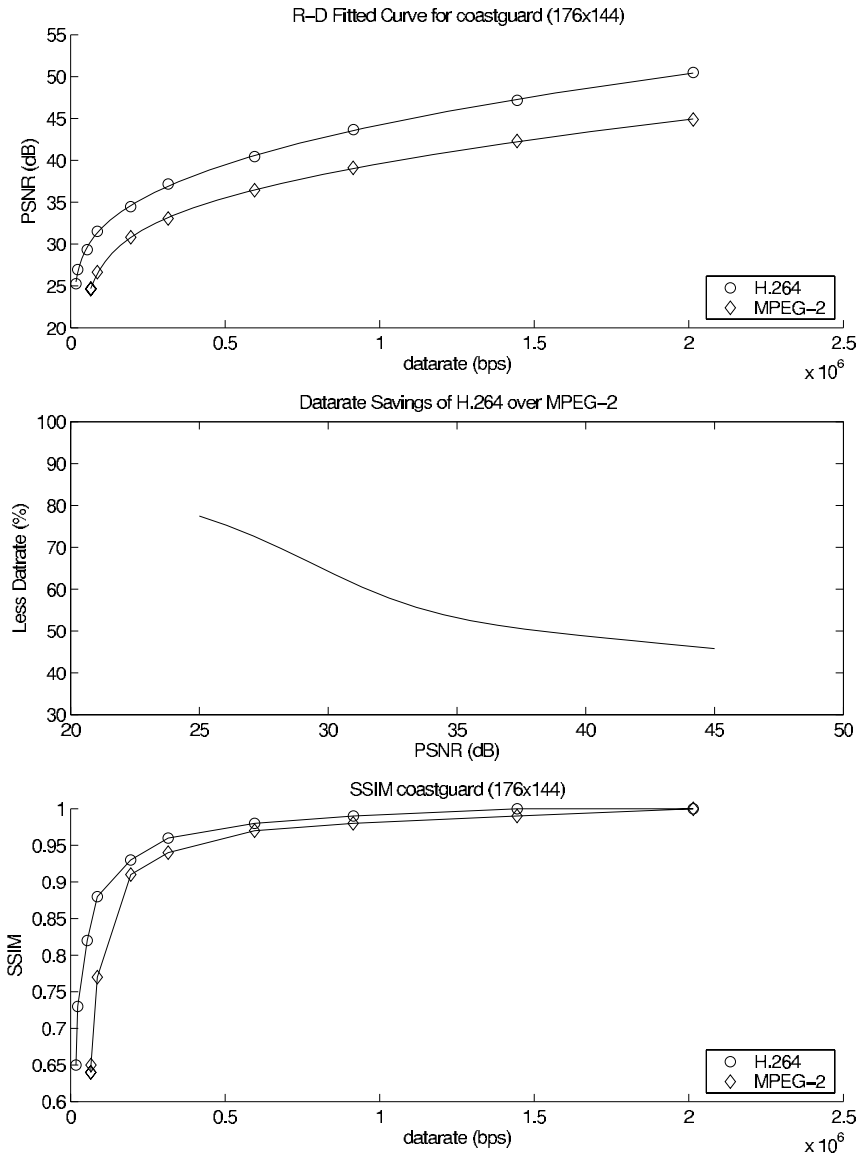


Fig. B.4. R-D Plot for coastguard sequence QCIF 30 fps.

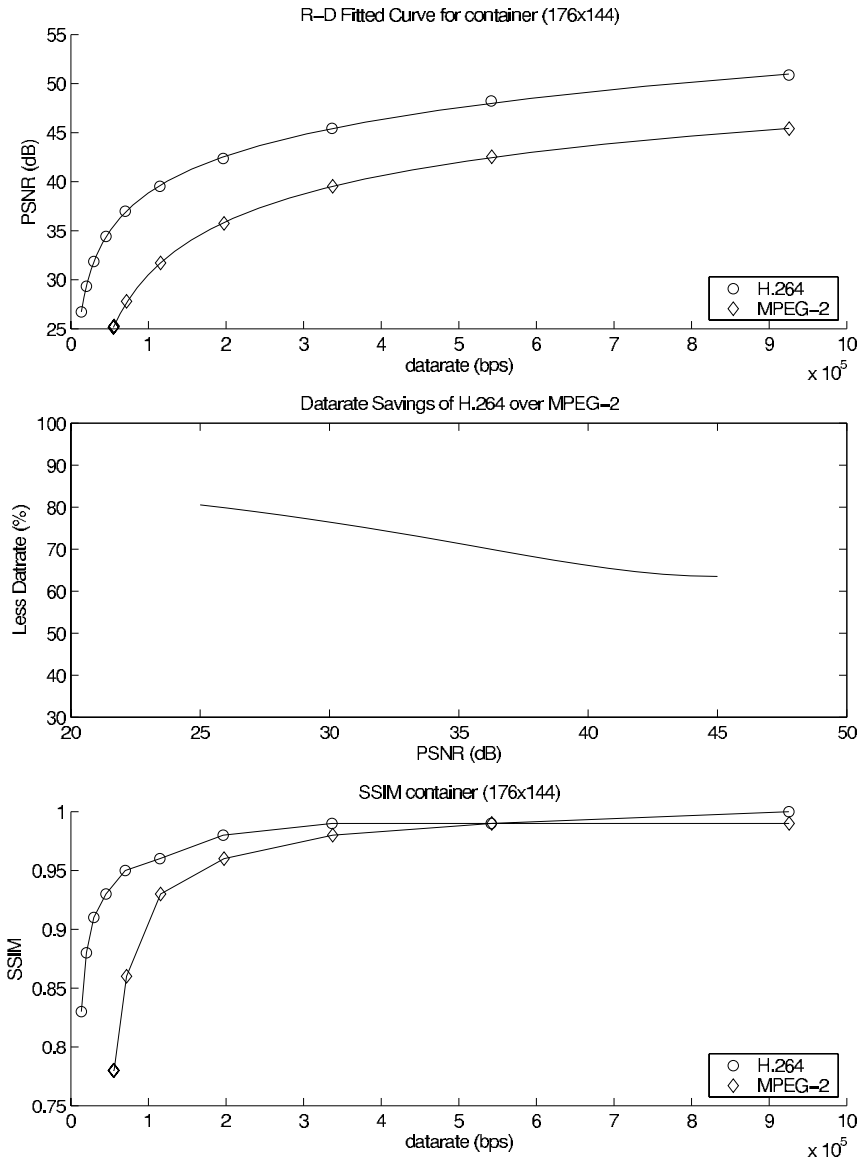


Fig. B.5. R-D Plot for container sequence QCIF 30 fps.

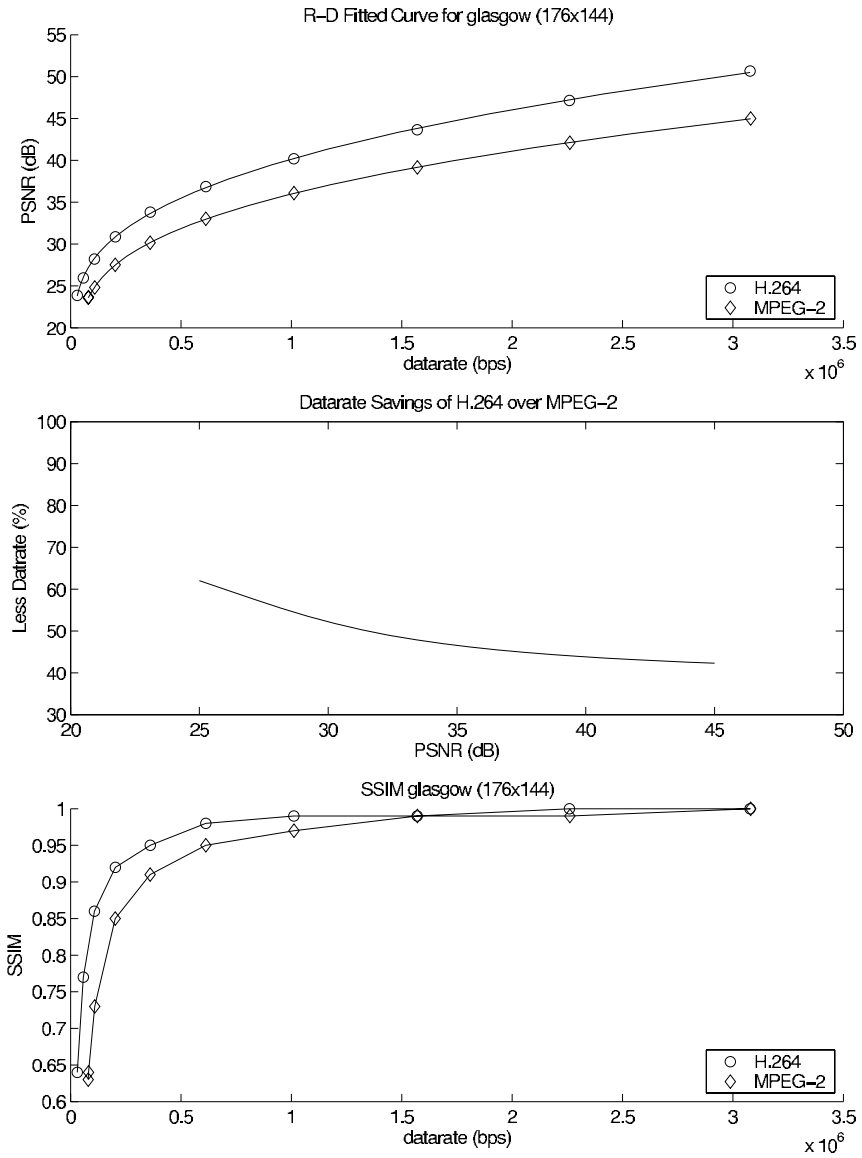


Fig. B.6. R-D Plot for glasgow sequence QCIF 30 fps.

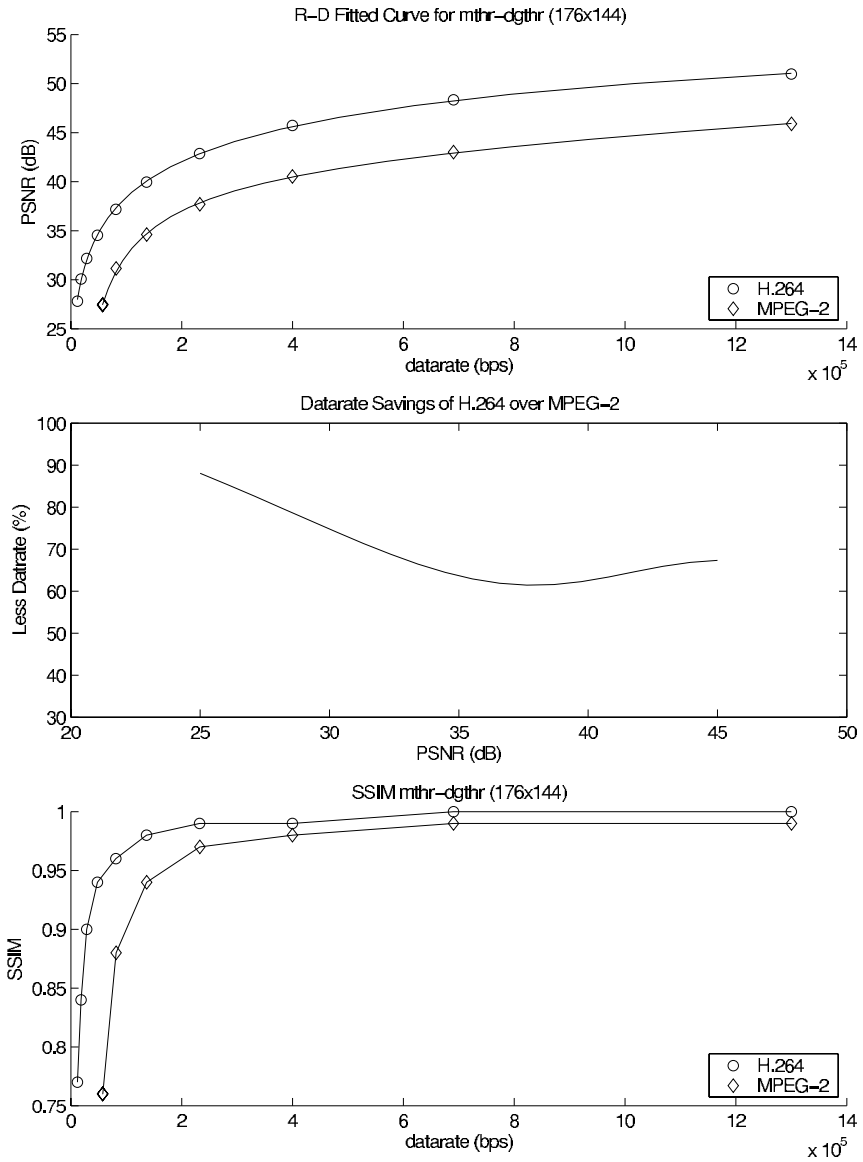


Fig. B.7. R-D Plot for mthr-dgthr sequence QCIF 30 fps.

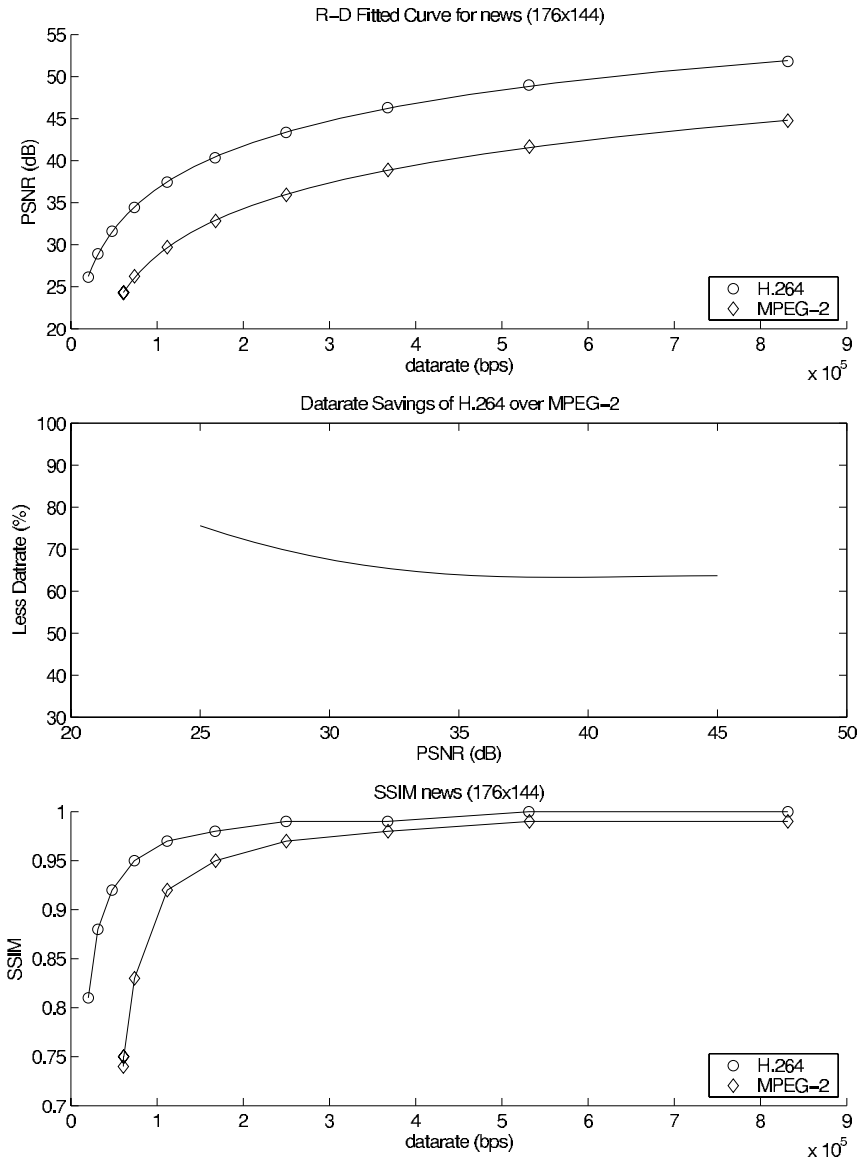


Fig. B.8. R-D Plot for news sequence QCIF 30 fps.

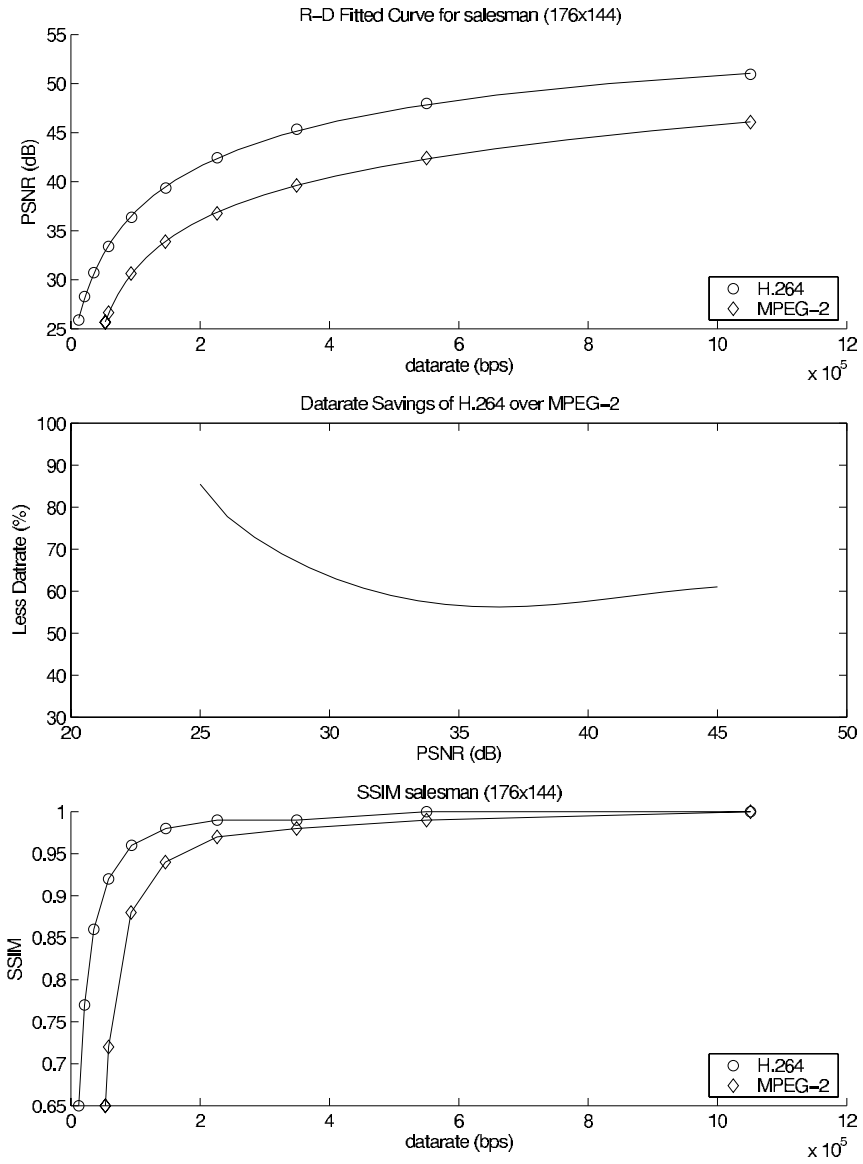


Fig. B.9. R-D Plot for salesman sequence QCIF 30 fps.

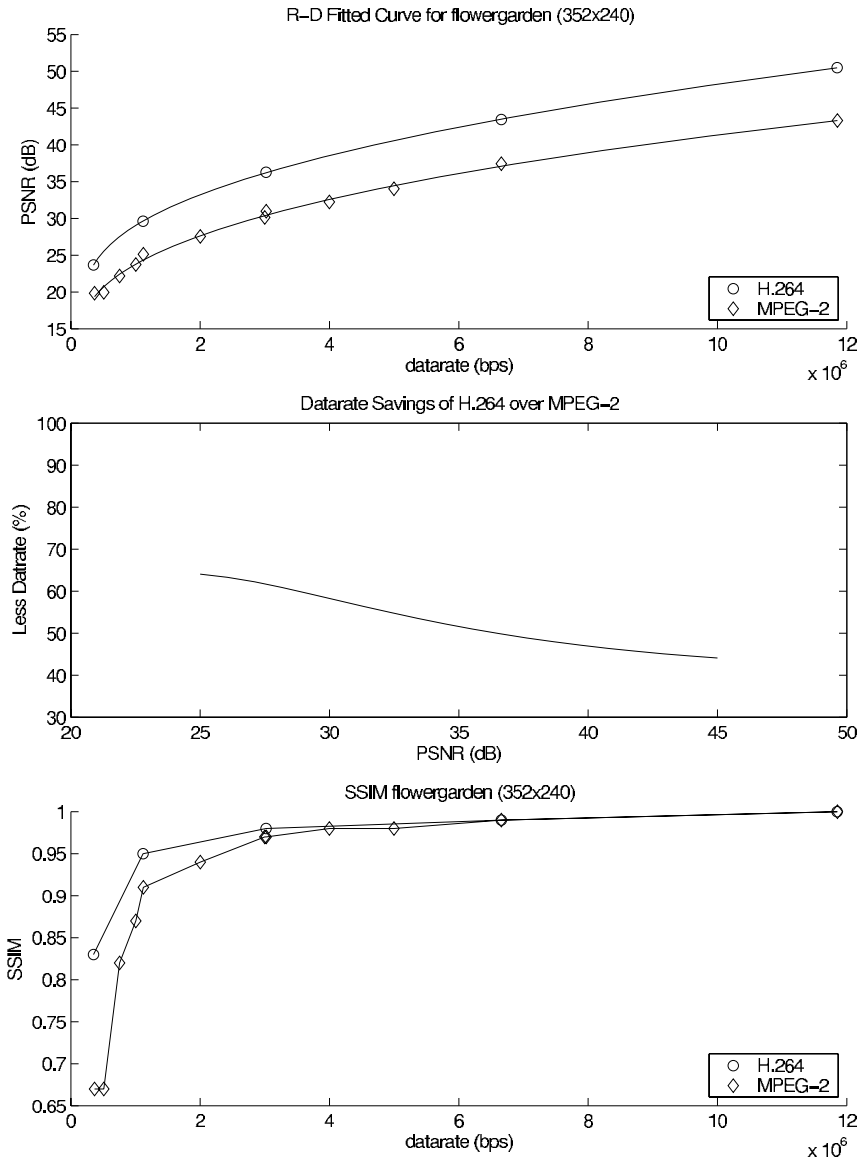


Fig. B.10. R-D Plot for flowergarden sequence CIF 30 fps.

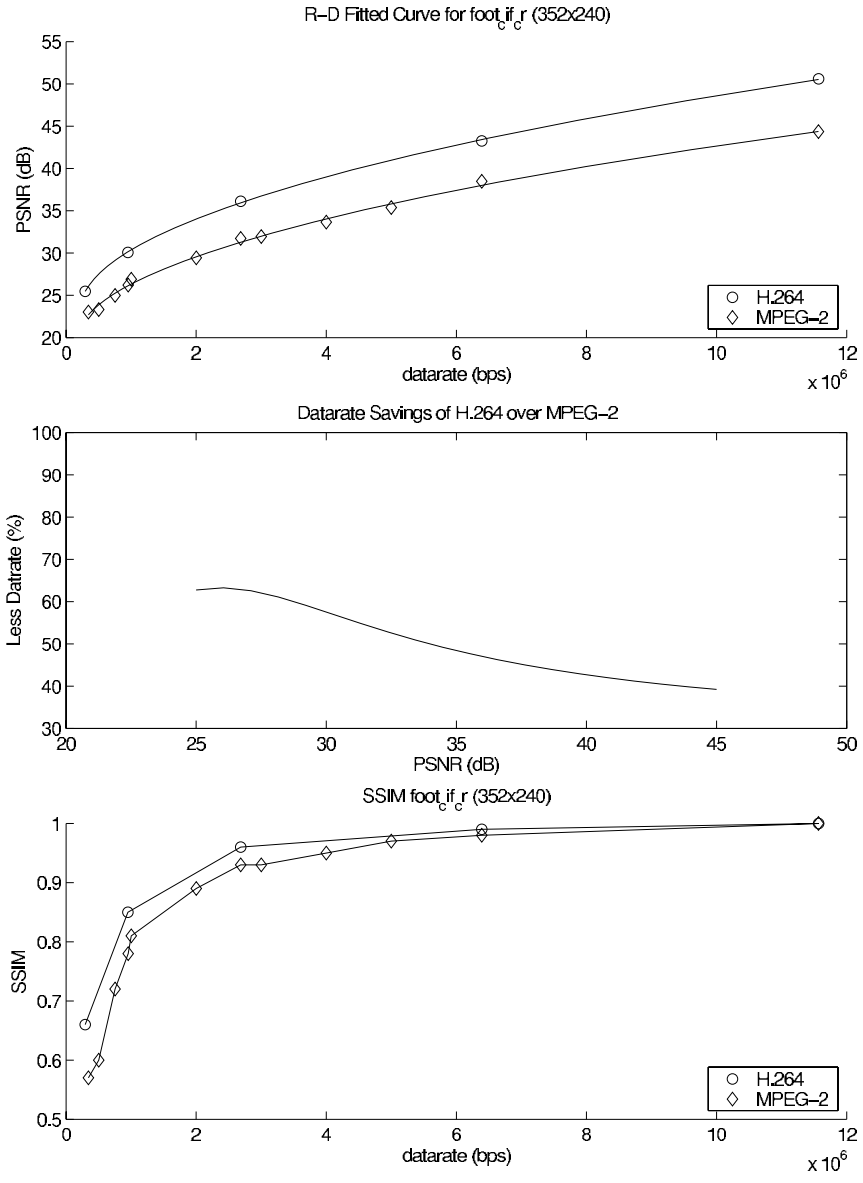


Fig. B.11. R-D Plot for foot sequence CIF 30 fps.

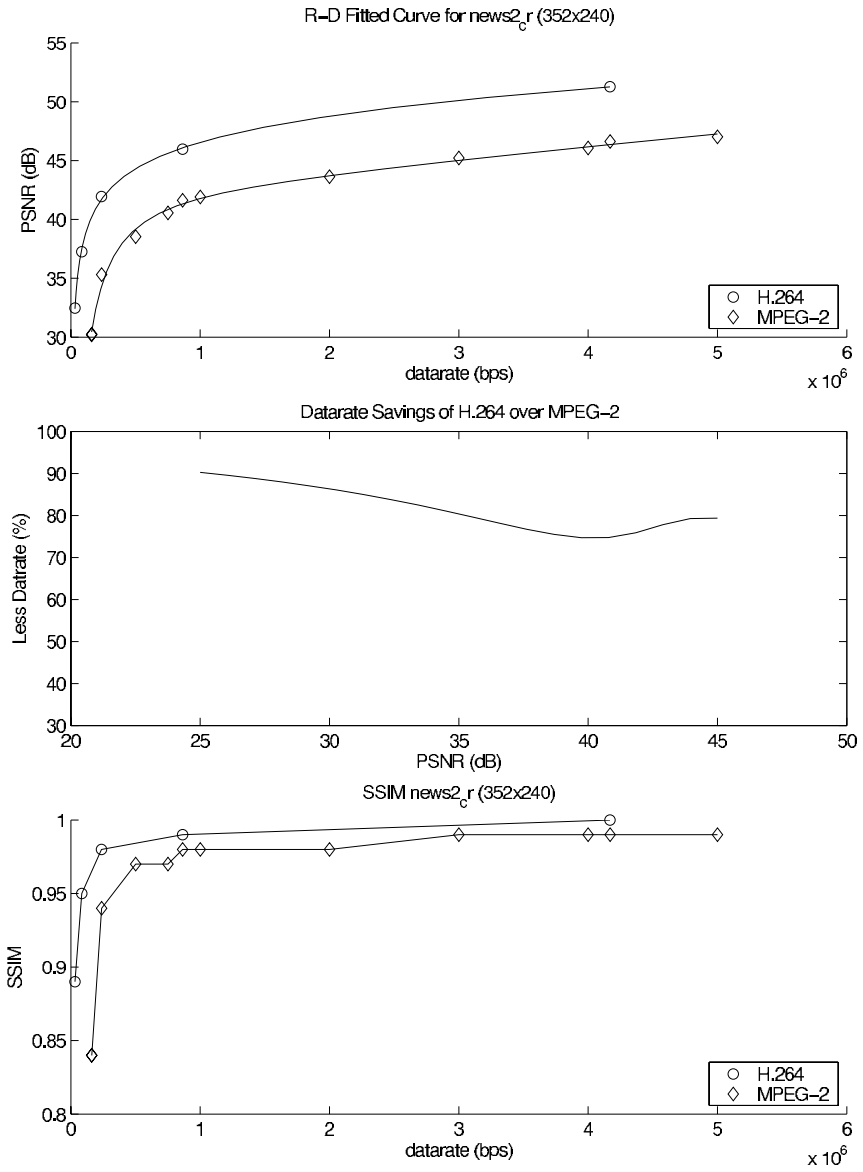


Fig. B.12. R-D Plot for sequence news2 CIF 30 fps.

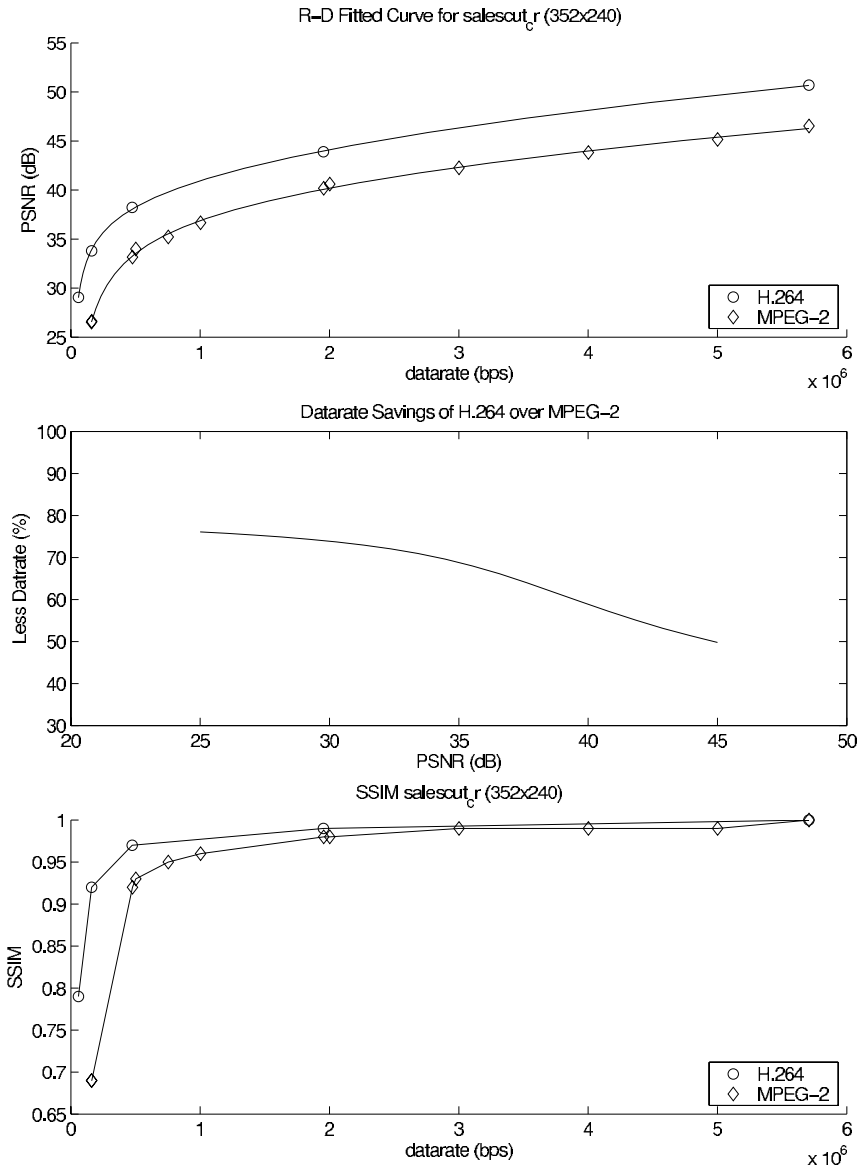


Fig. B.13. R-D Plot for sequence salescut CIF 30 fps.

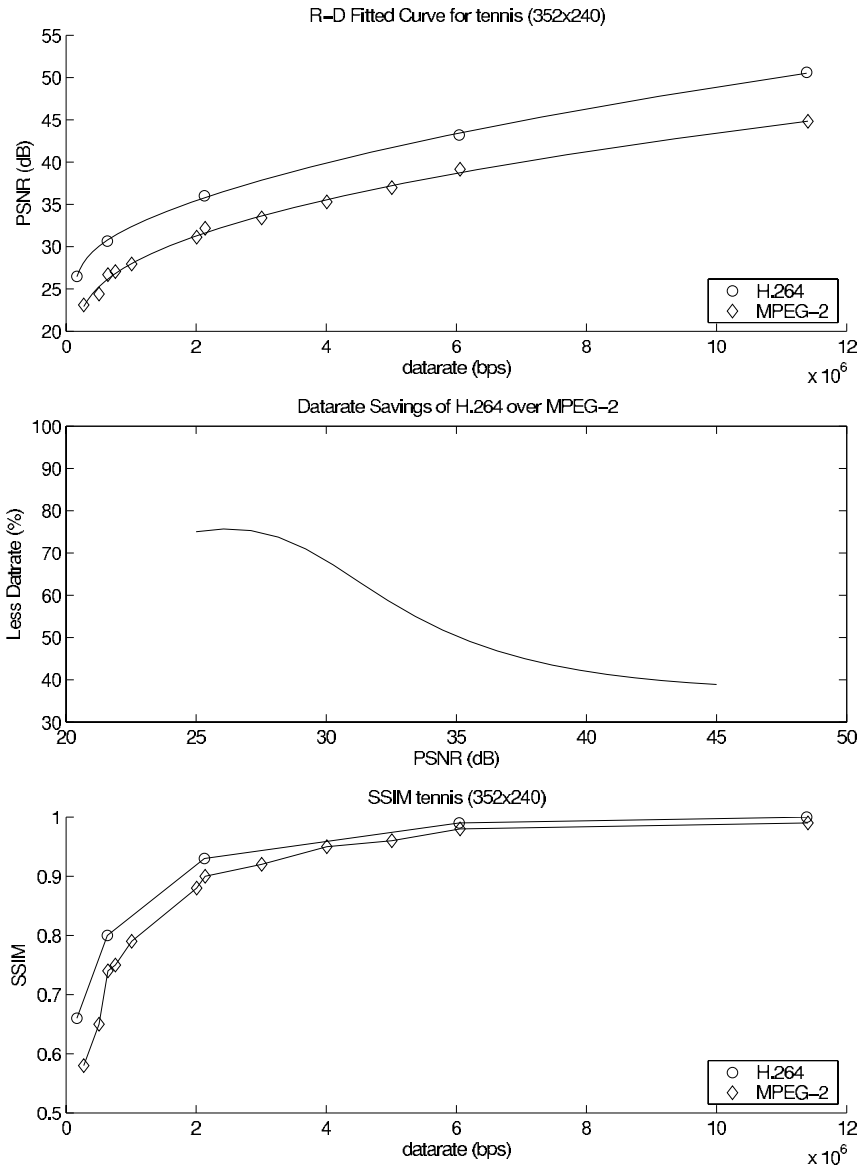


Fig. B.14. R-D Plot for sequence tennis CIF 30 fps.

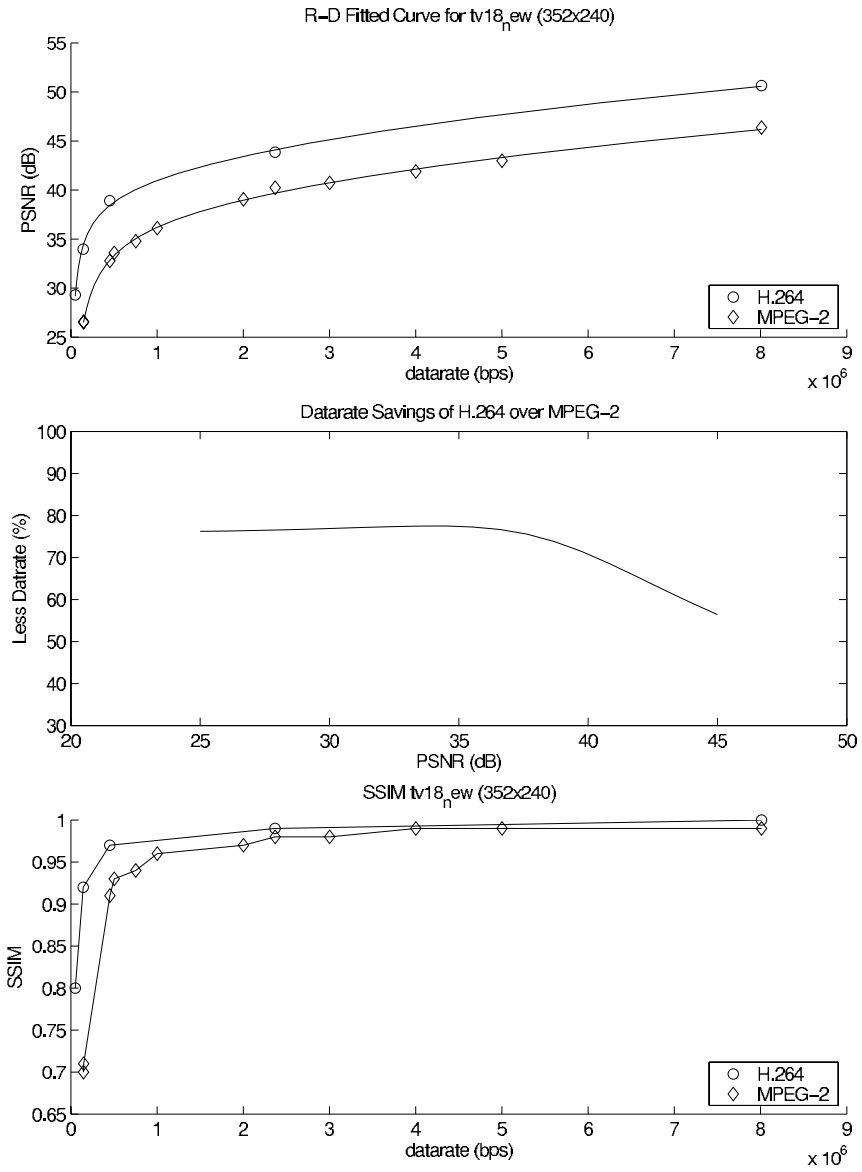


Fig. B.15. R-D Plot for sequence tv18_new CIF 30 fps.

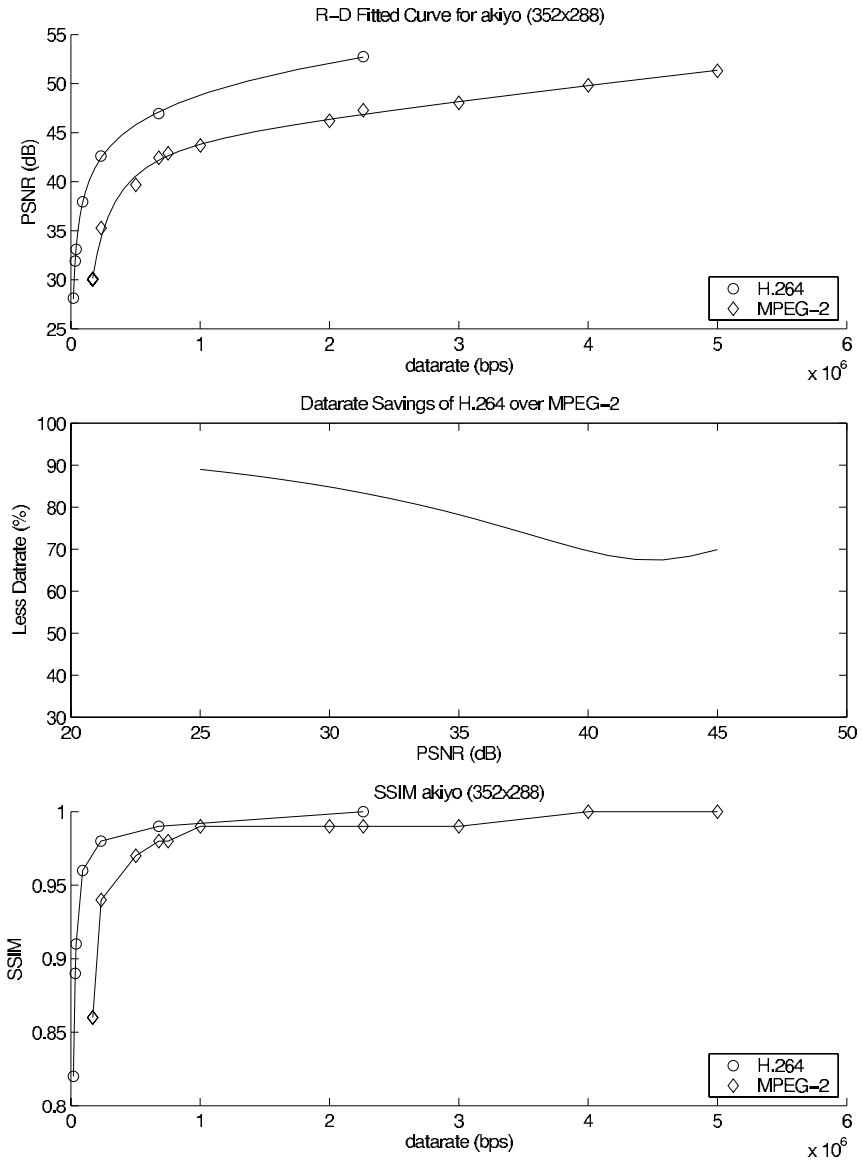


Fig. B.16. R-D Plot for sequence akiyo CIF 30 fps.

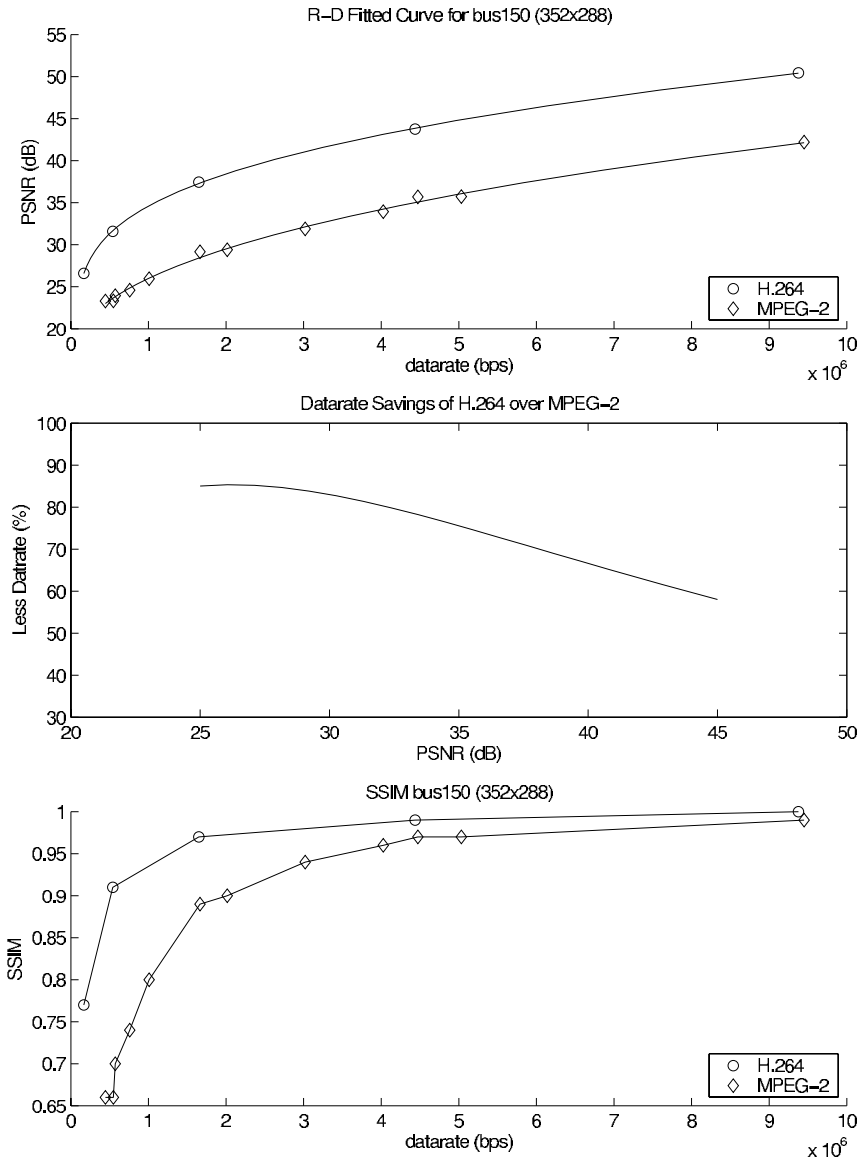


Fig. B.17. R-D Plot for sequence bus150 CIF 30 fps.

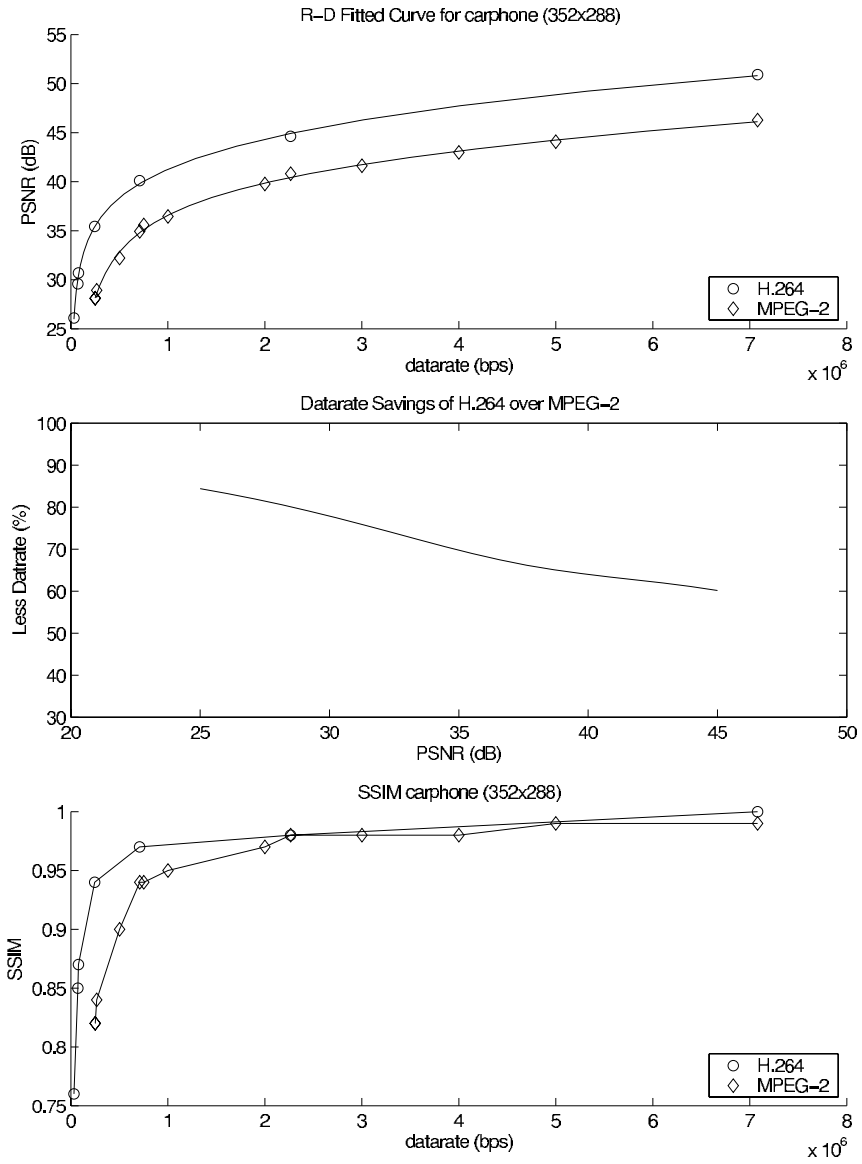


Fig. B.18. R-D Plot for sequence carphone CIF 30 fps.

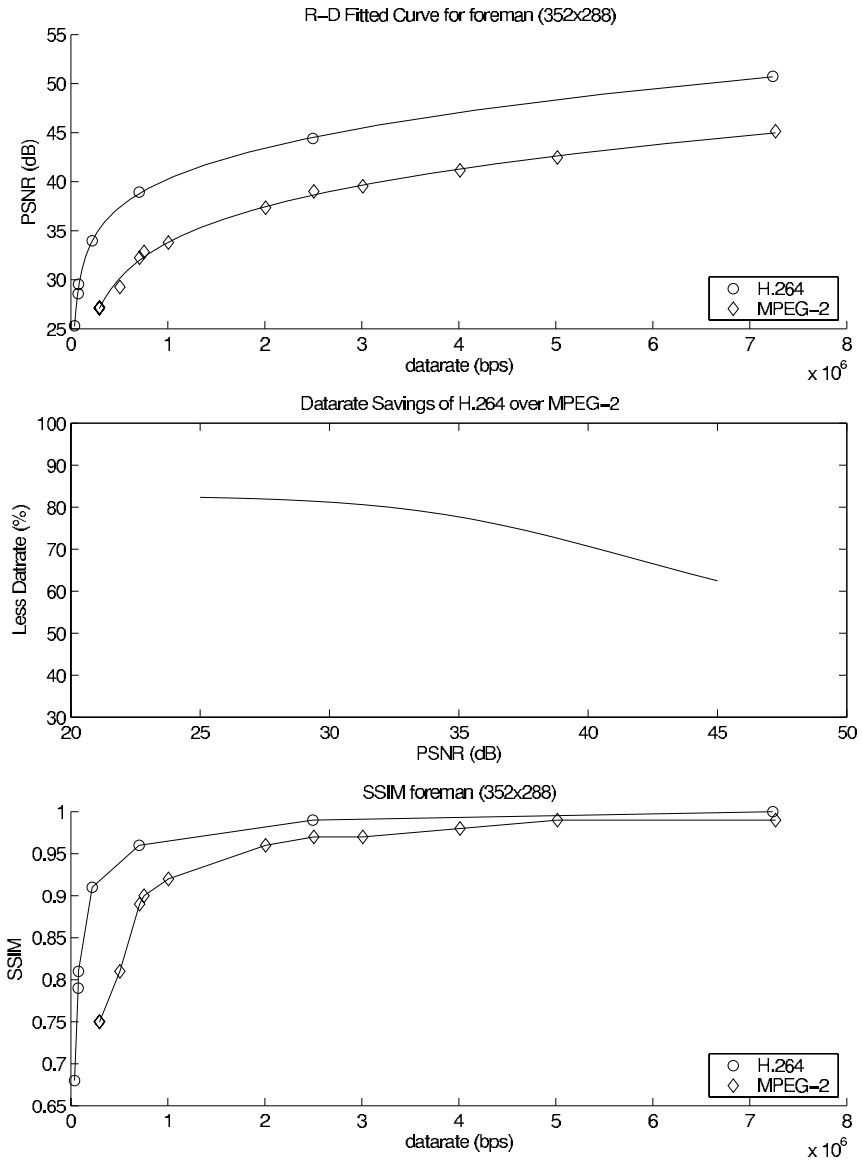


Fig. B.19. R-D Plot for sequence foreman CIF 30 fps.

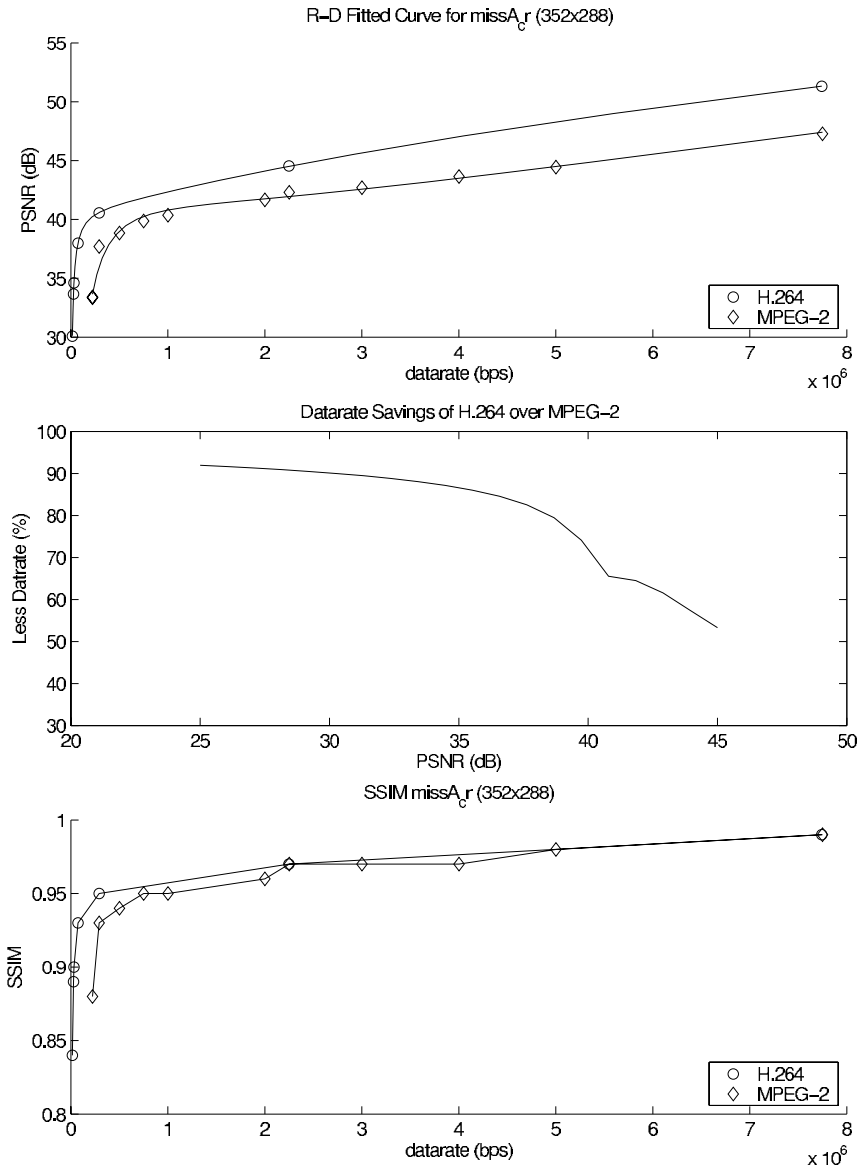


Fig. B.20. R-D Plot for sequence missA CIF 30 fps.

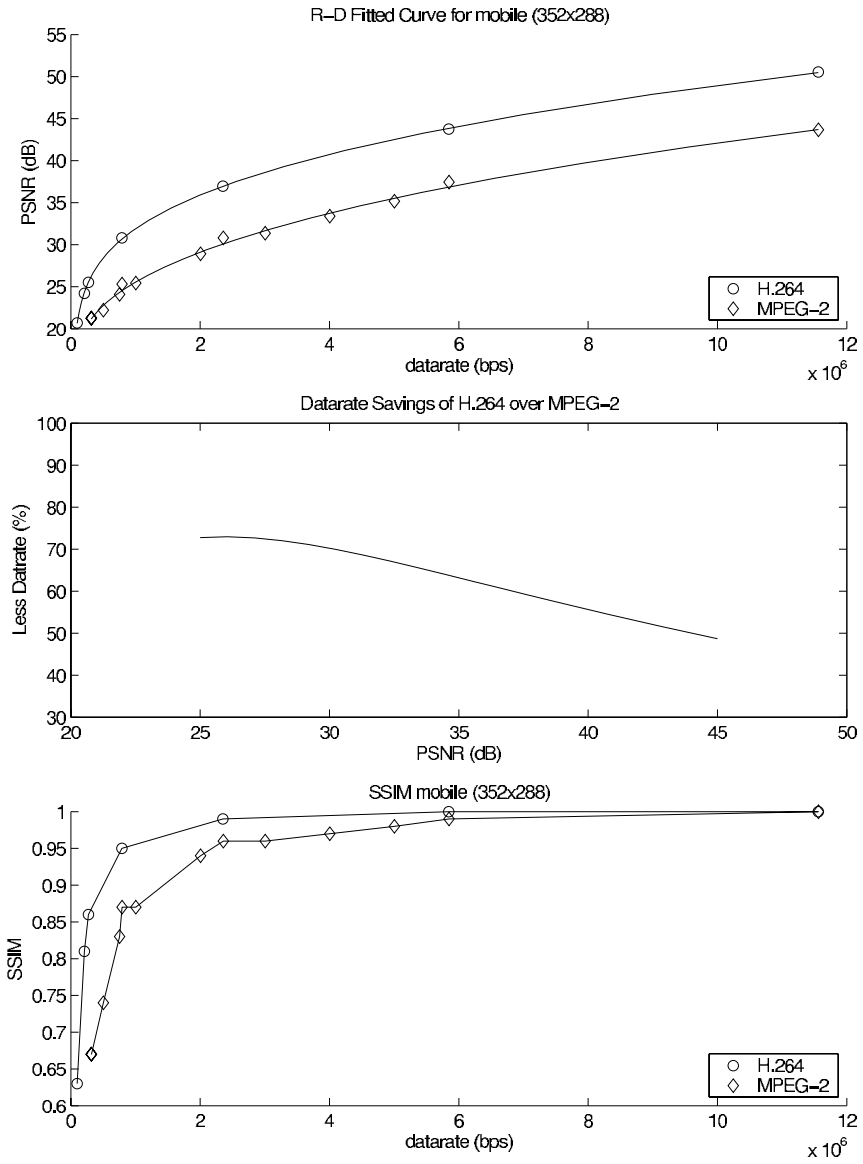


Fig. B.21. R-D Plot for sequence mobile CIF 30 fps.

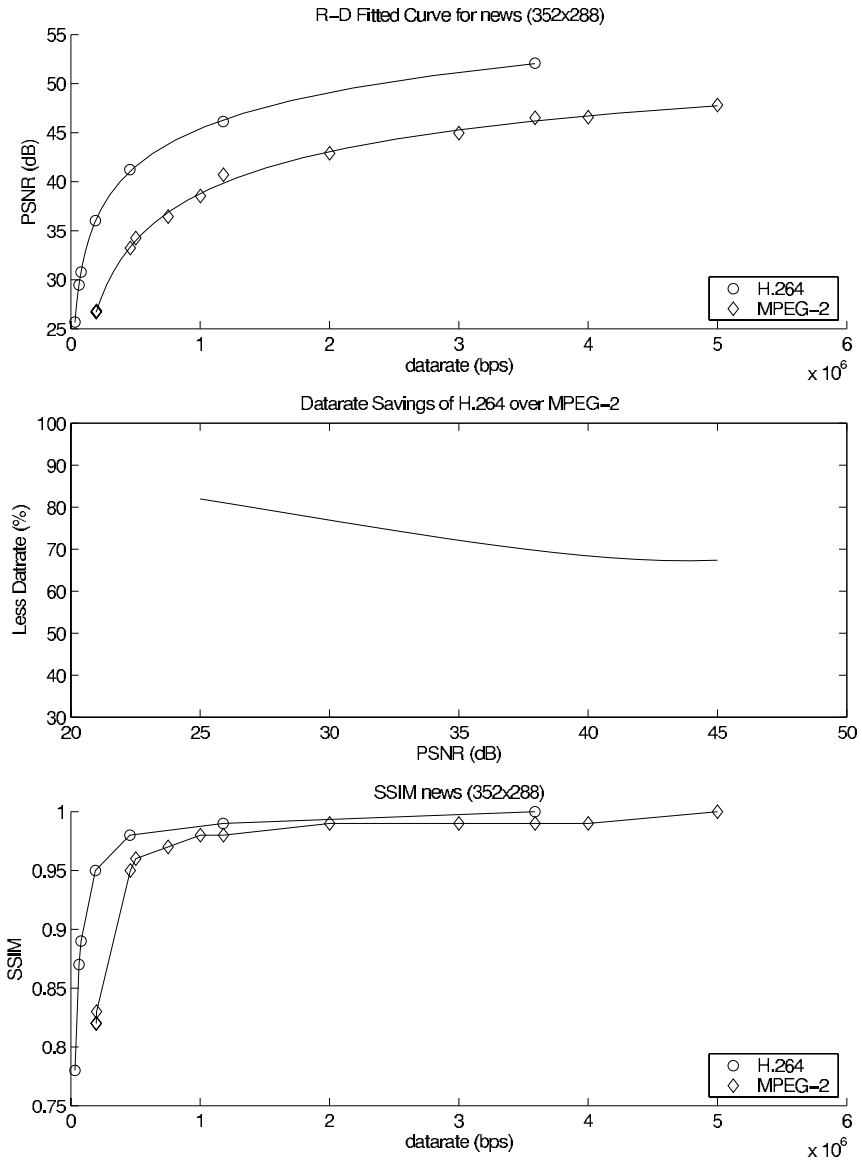


Fig. B.22. R-D Plot for sequence news CIF 30 fps.

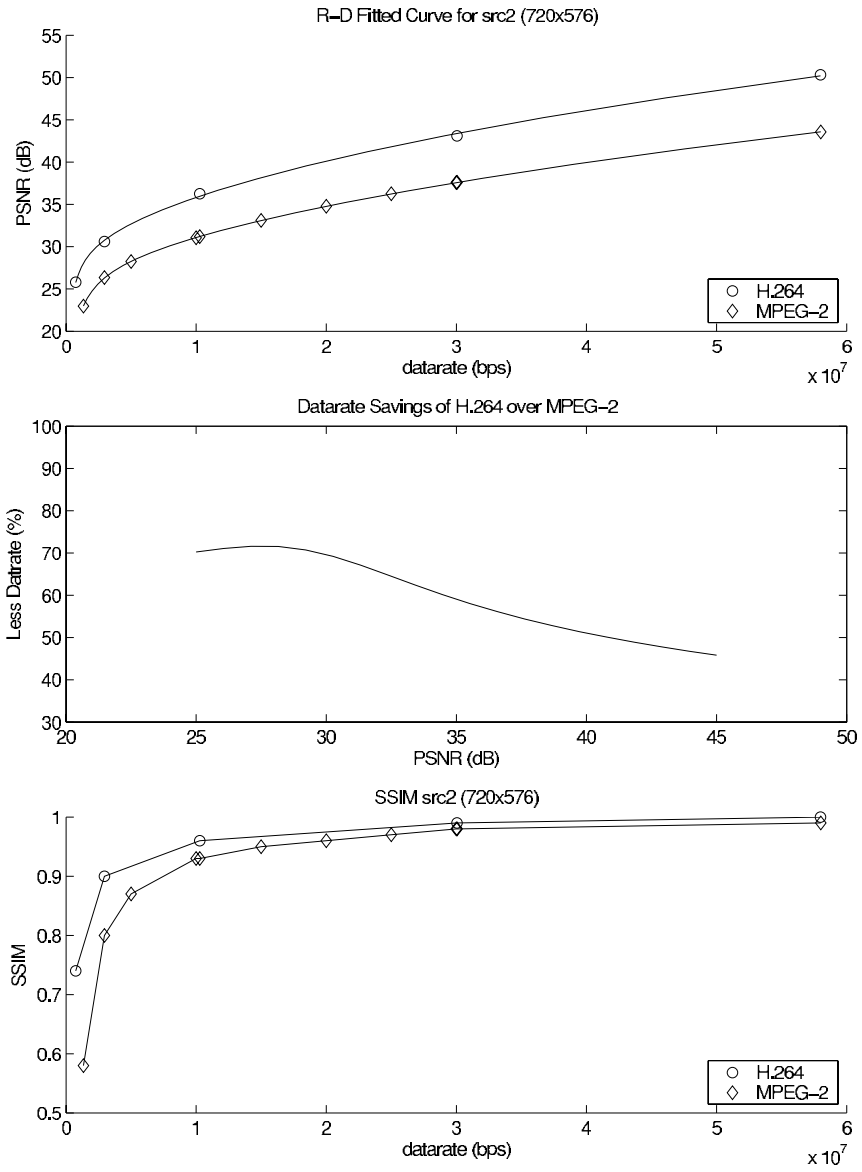


Fig. B.23. R-D Plot for src2 sequence 720×576 interlaced 30 fps.

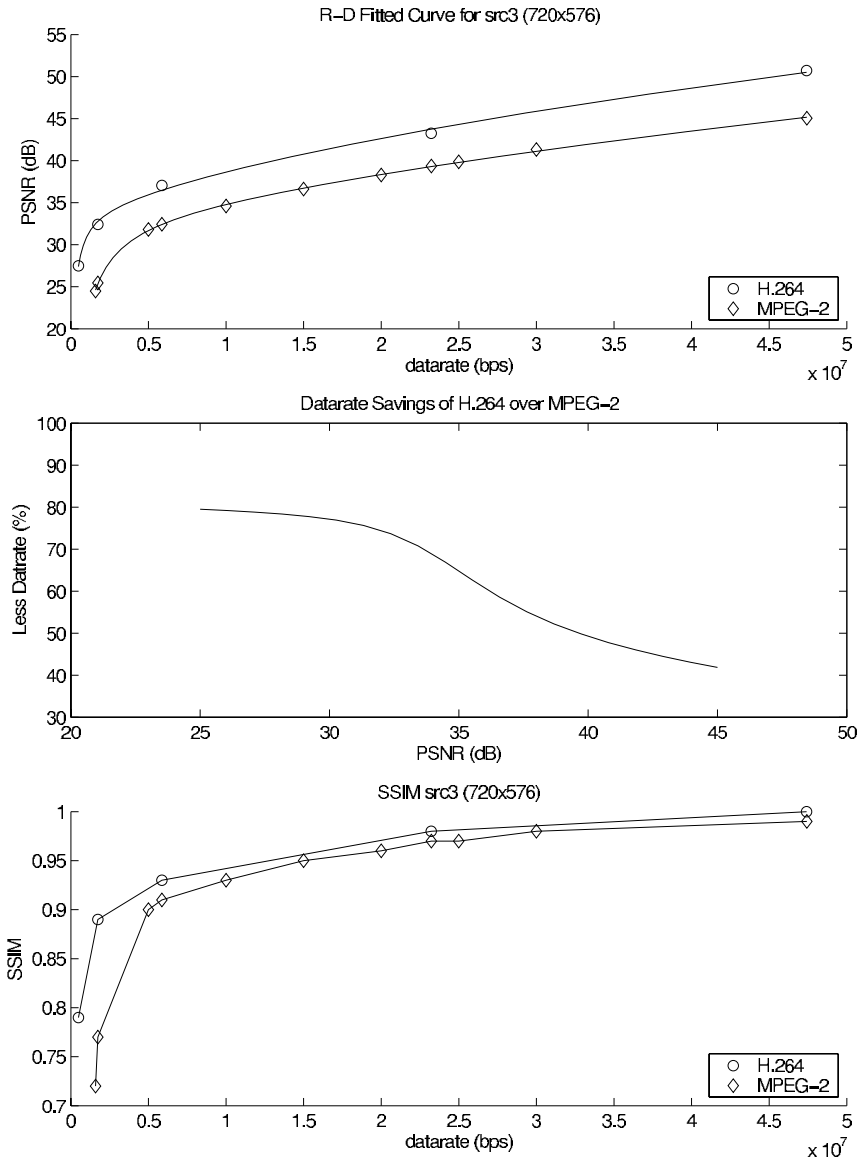


Fig. B.24. R-D Plot for src3 sequence 720×576 interlaced 30 fps.

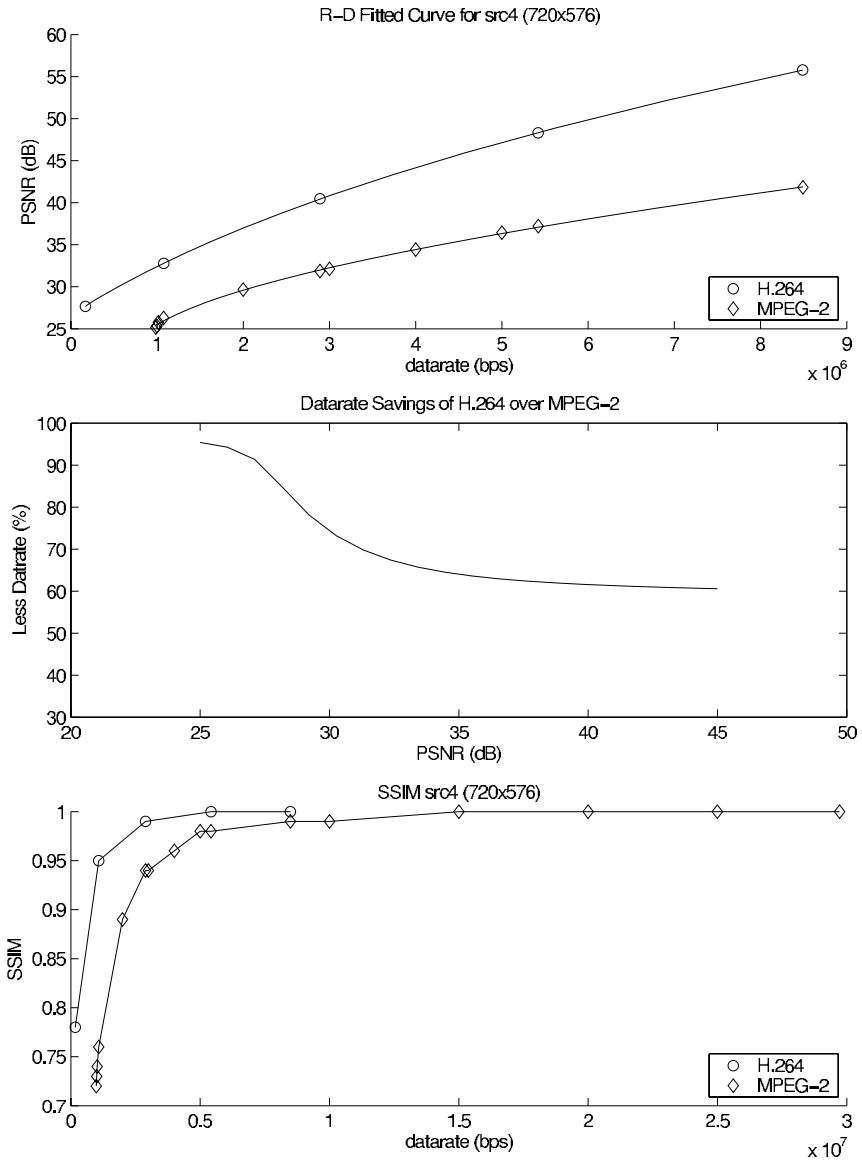


Fig. B.25. R-D Plot for src4 sequence 720×576 interlaced 30 fps.

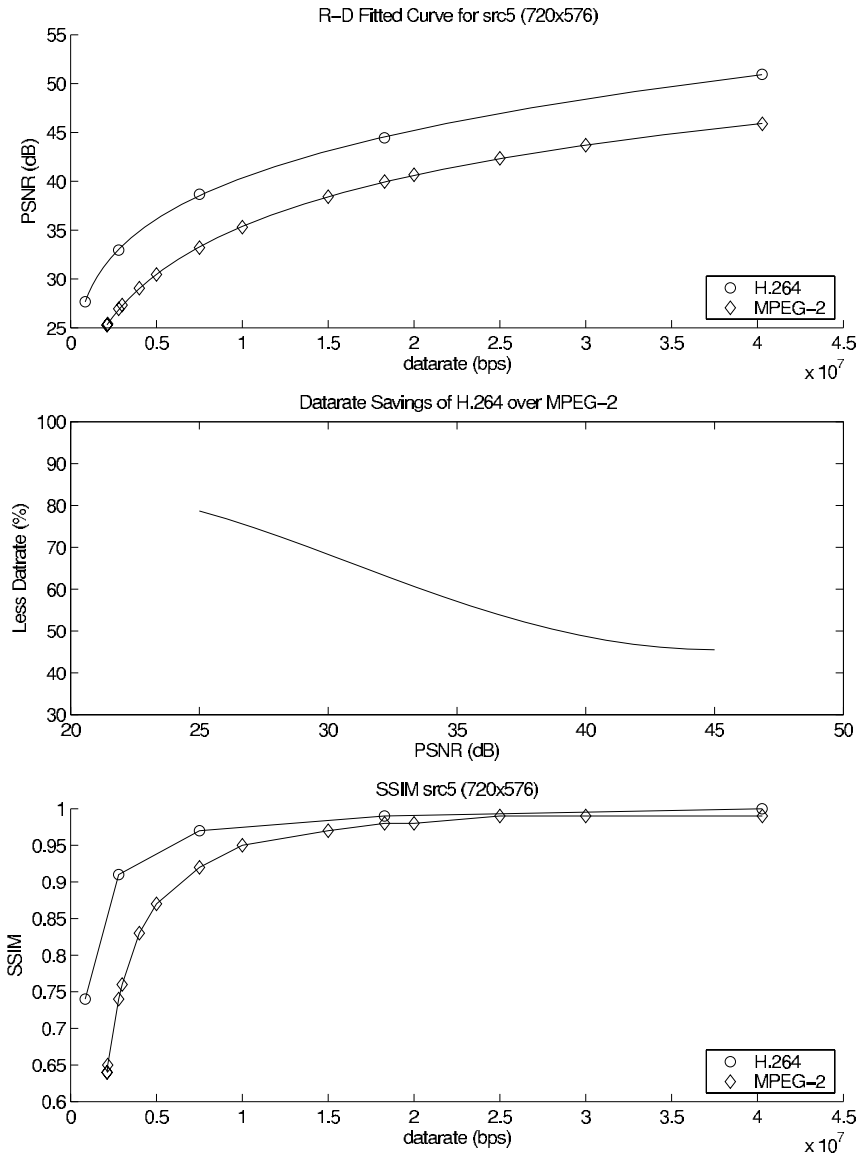


Fig. B.26. R-D Plot for src5 sequence 720×576 interlaced 30 fps.

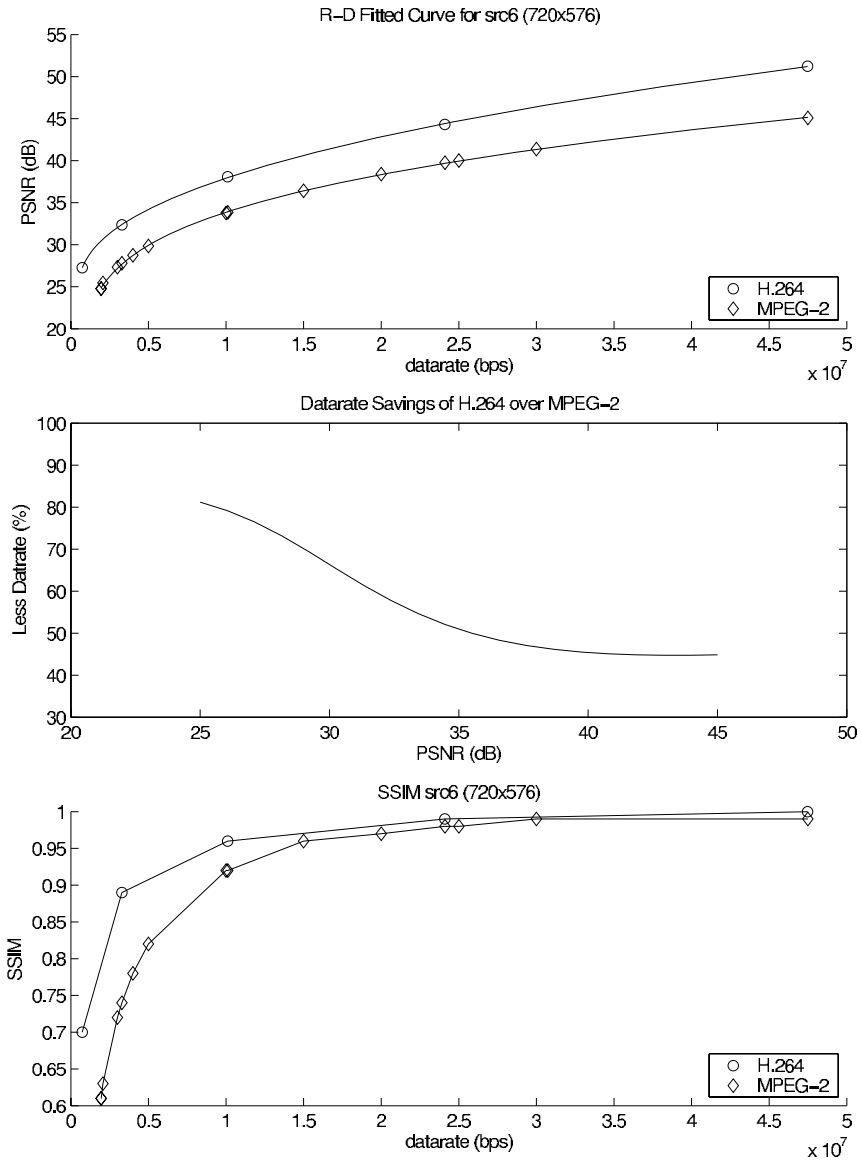


Fig. B.27. R-D Plot for src6 sequence 720×576 interlaced 30 fps.

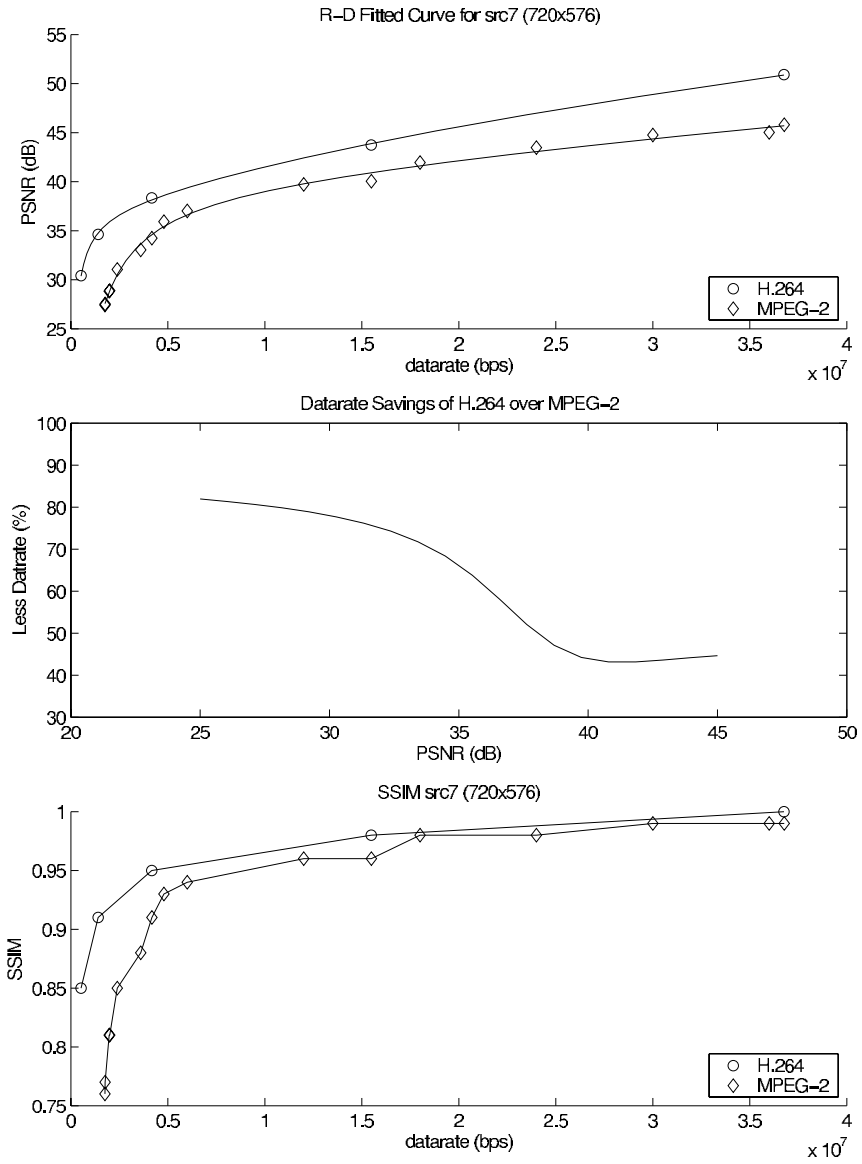


Fig. B.28. R-D Plot for src7 sequence 720×576 interlaced 30 fps.

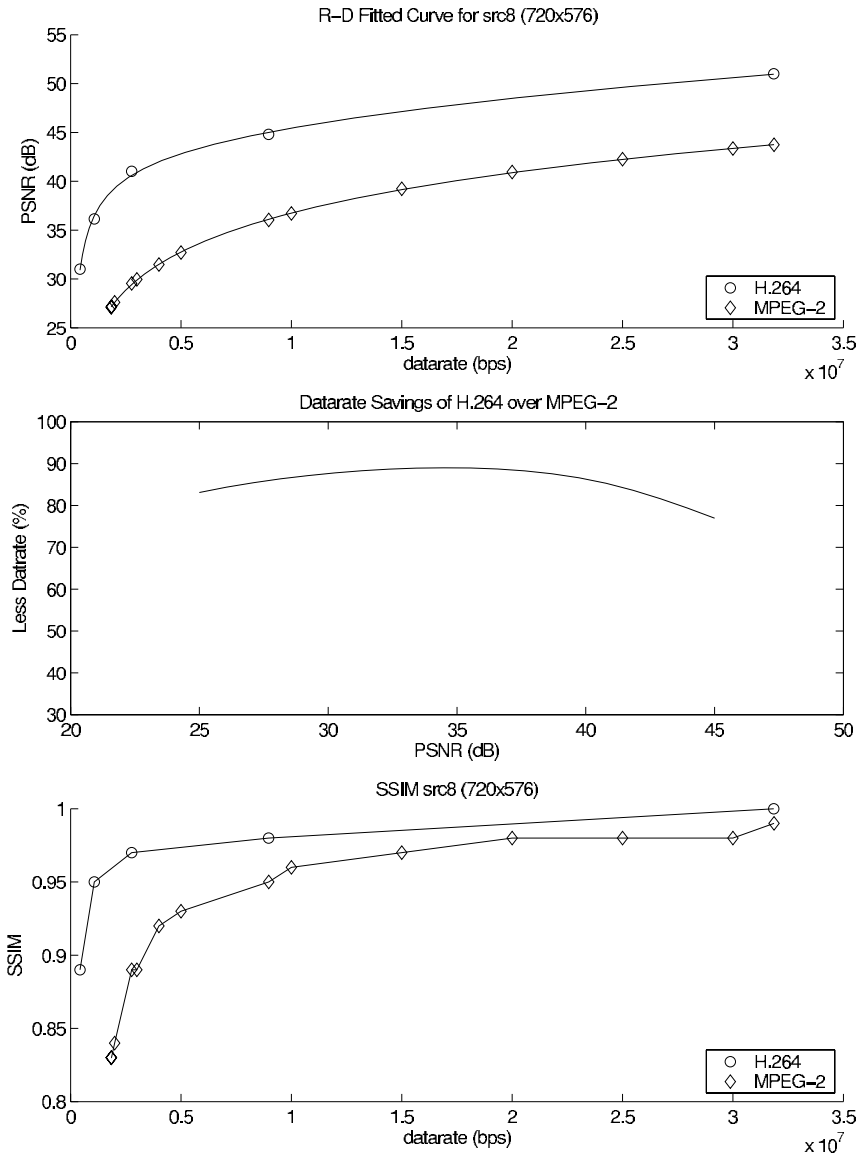


Fig. B.29. R-D Plot for src8 sequence 720×576 interlaced 30 fps.

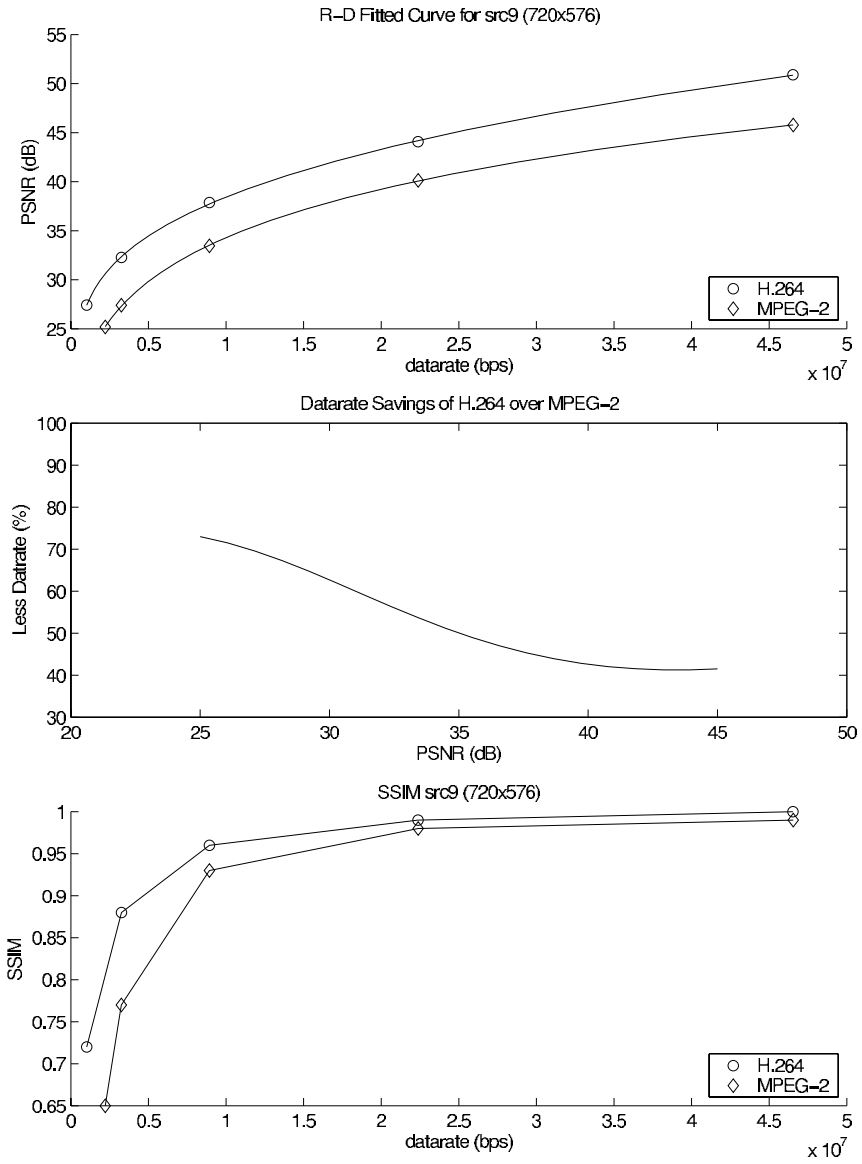


Fig. B.30. R-D Plot for src9 sequence 720×576 interlaced 30 fps.

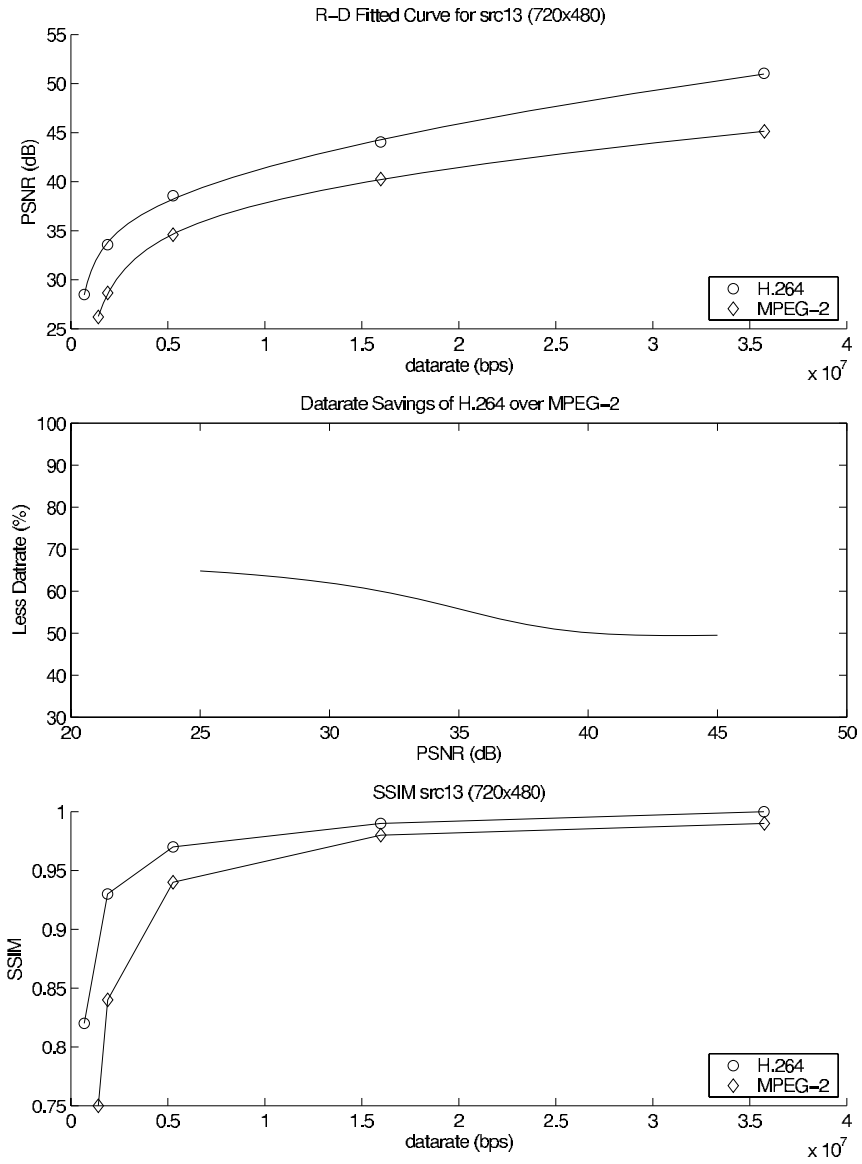


Fig. B.31. R-D Plot for src13 sequence 720 × 480 interlaced 30 fps.

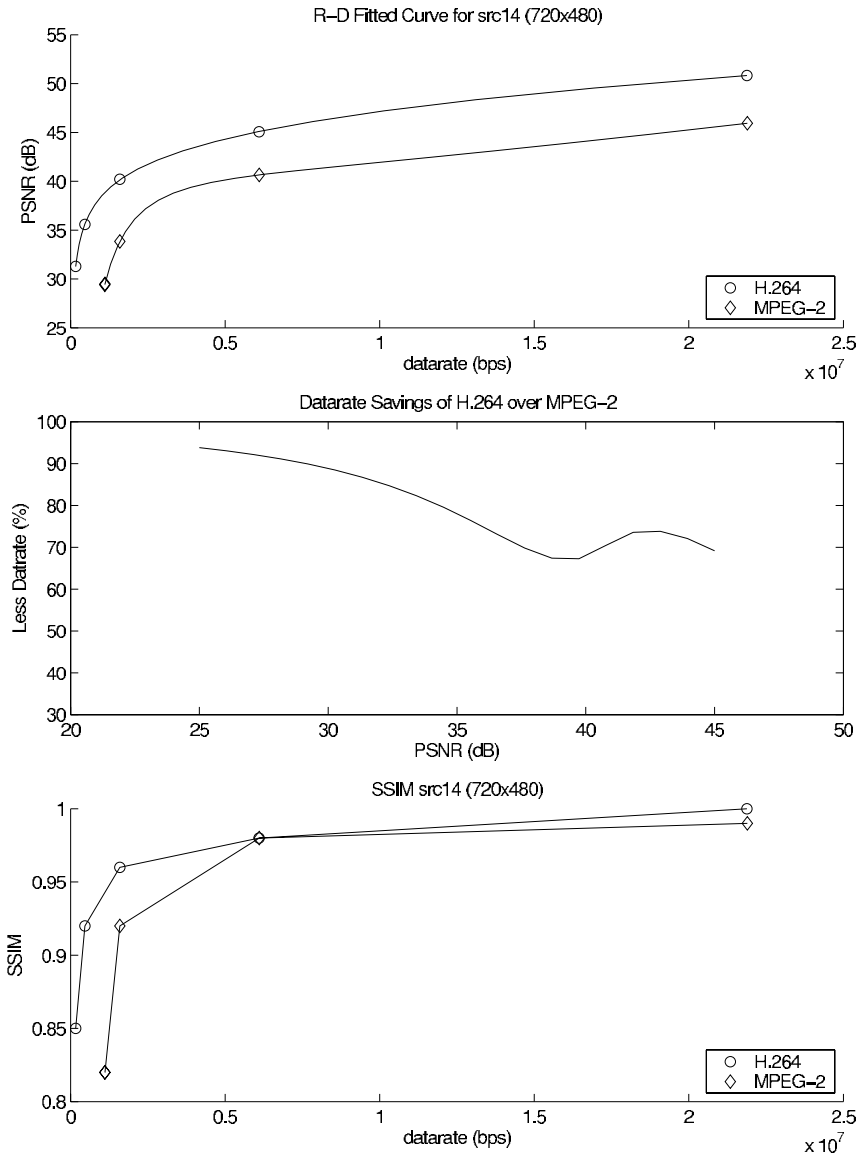


Fig. B.32. R-D Plot for src14 sequence 720×480 interlaced 30 fps.

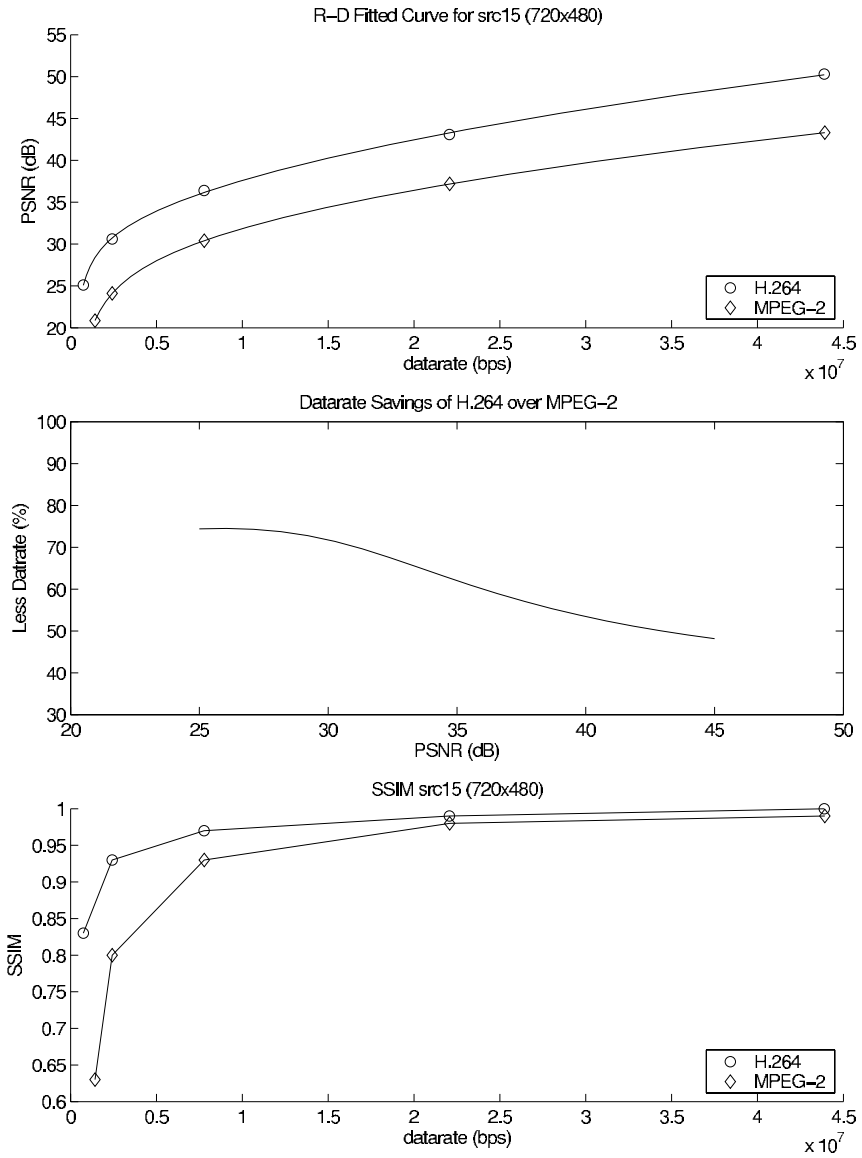


Fig. B.33. R-D Plot for src15 sequence 720×480 interlaced 30 fps.

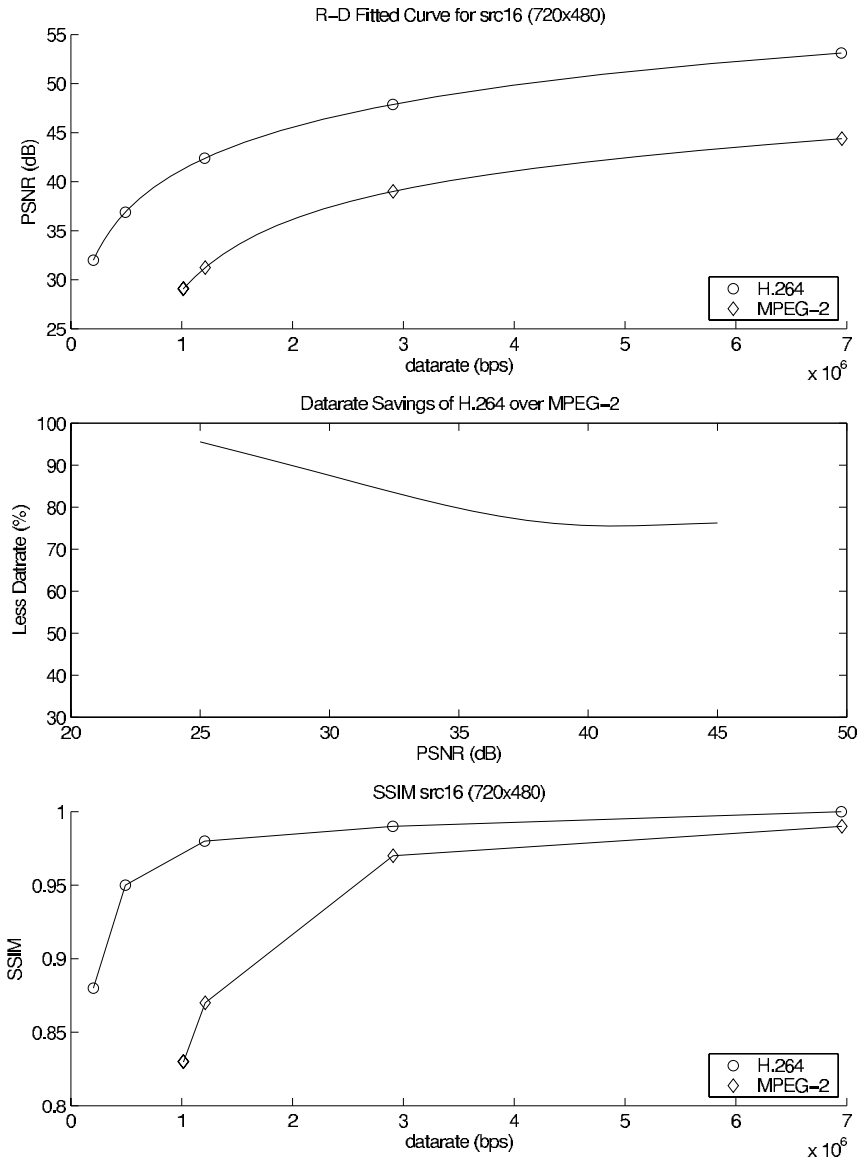


Fig. B.34. R-D Plot for src16 sequence 720 × 480 interlaced 30 fps.

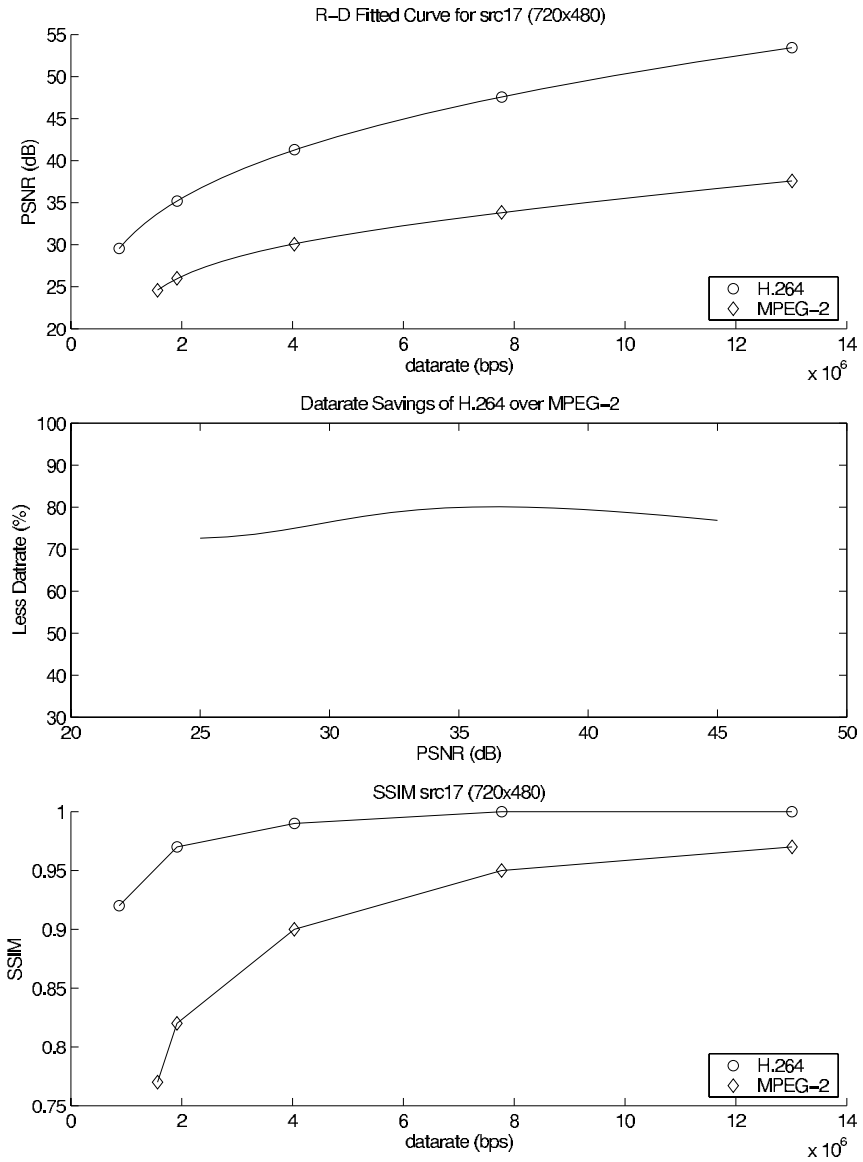


Fig. B.35. R-D Plot for src17 sequence 720 × 480 interlaced 30 fps.

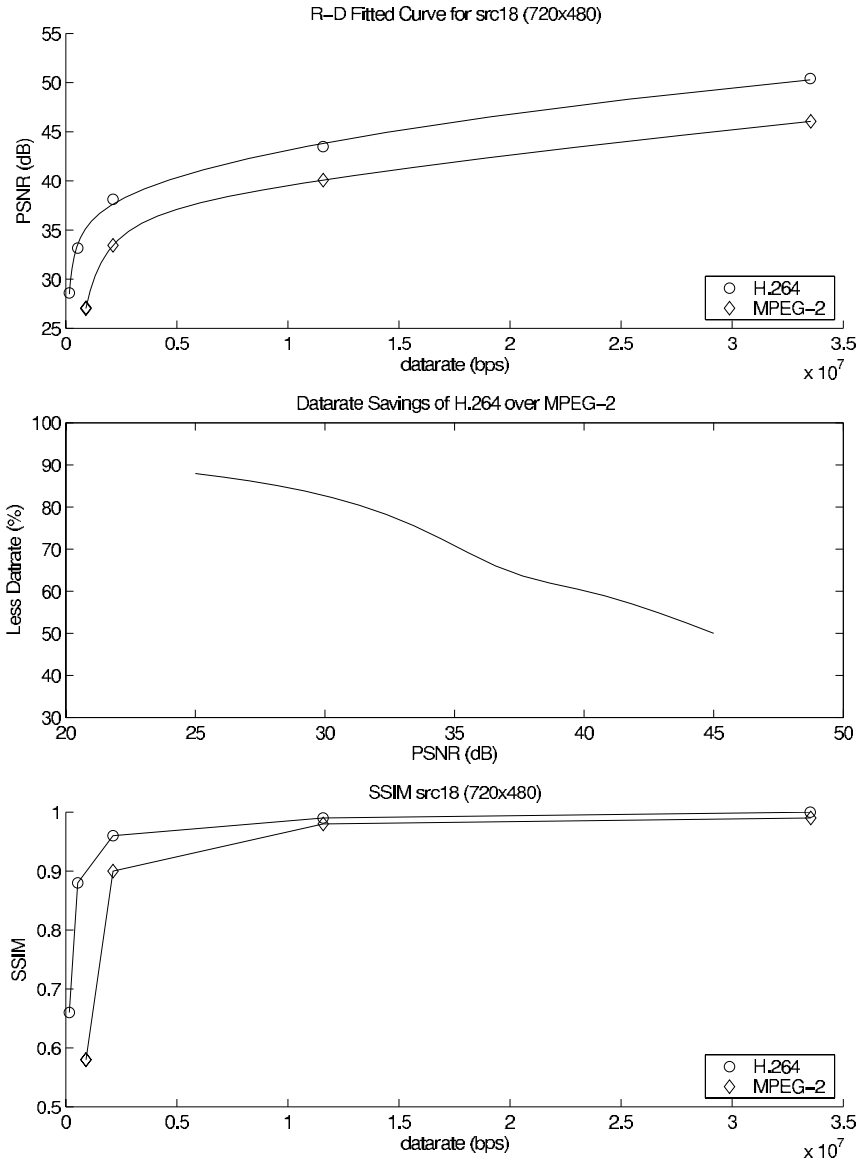


Fig. B.36. R-D Plot for src18 sequence 720 × 480 interlaced 30 fps.

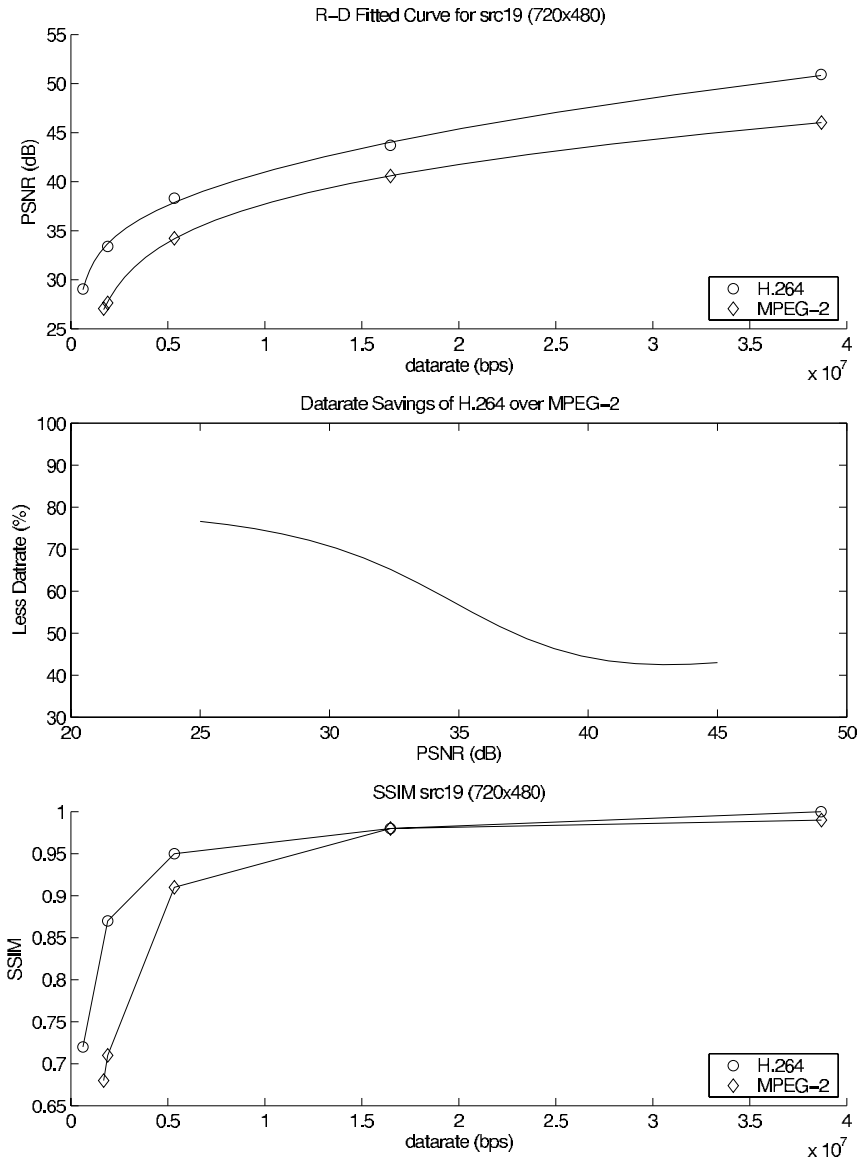


Fig. B.37. R-D Plot for src19 sequence 720 × 480 interlaced 30 fps.

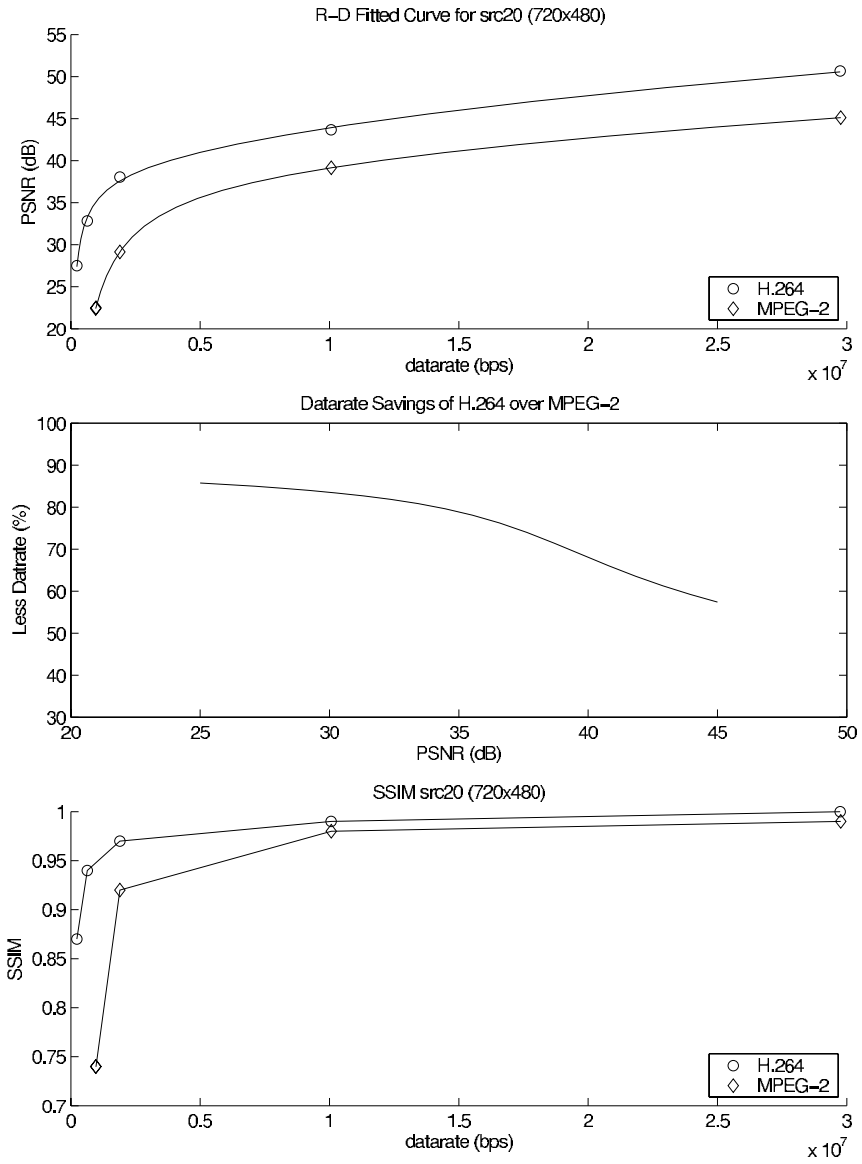


Fig. B.38. R-D Plot for src20 sequence 720 × 480 interlaced 30 fps.

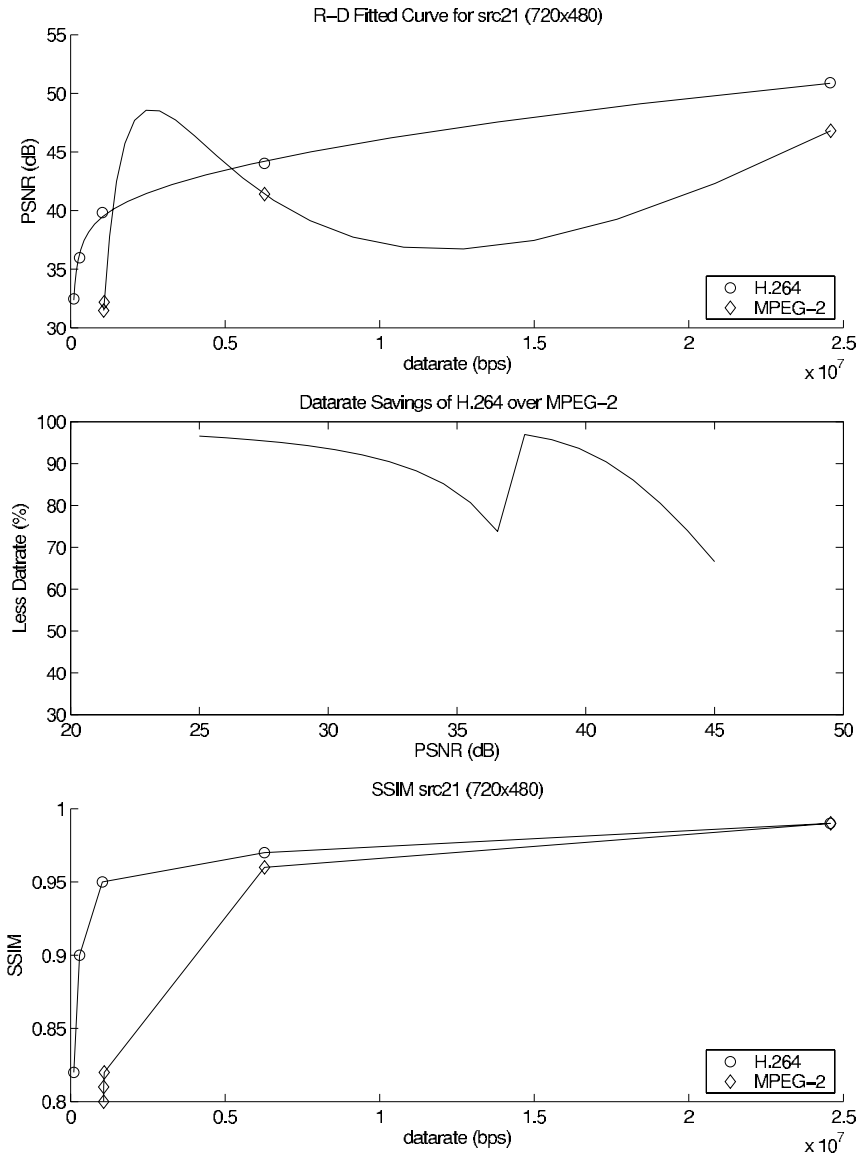


Fig. B.39. R-D Plot for src21 sequence 720×480 interlaced 30 fps.

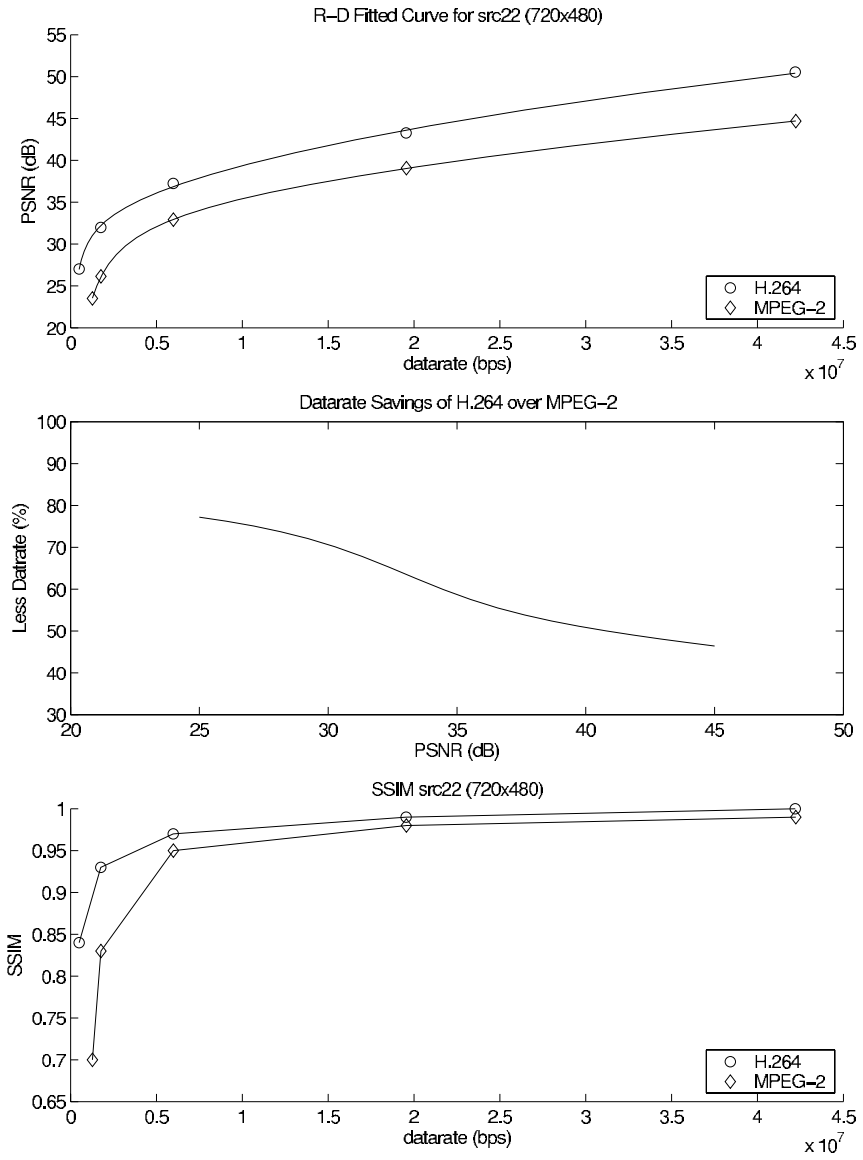


Fig. B.40. R-D Plot for src22 sequence 720×480 interlaced 30 fps.

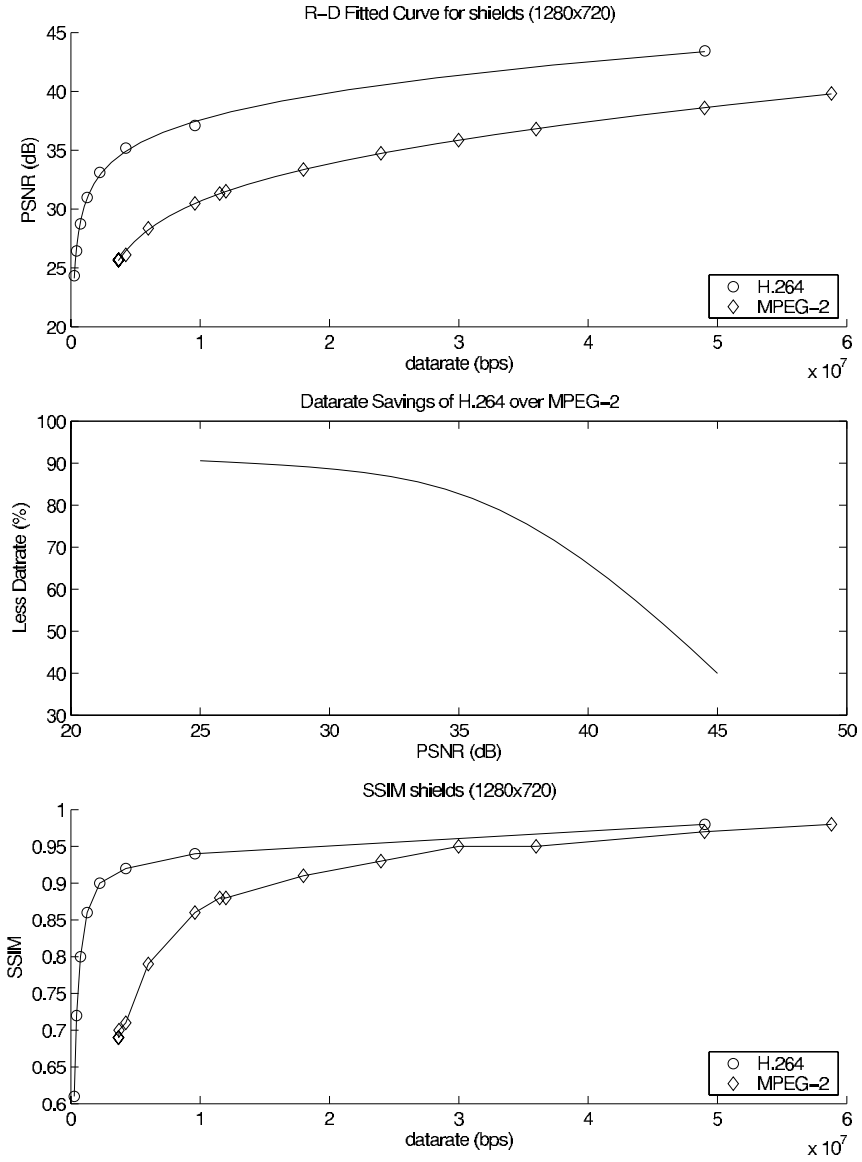


Fig. B.41. R-D Plot for shields sequence 1280×720 25 fps.

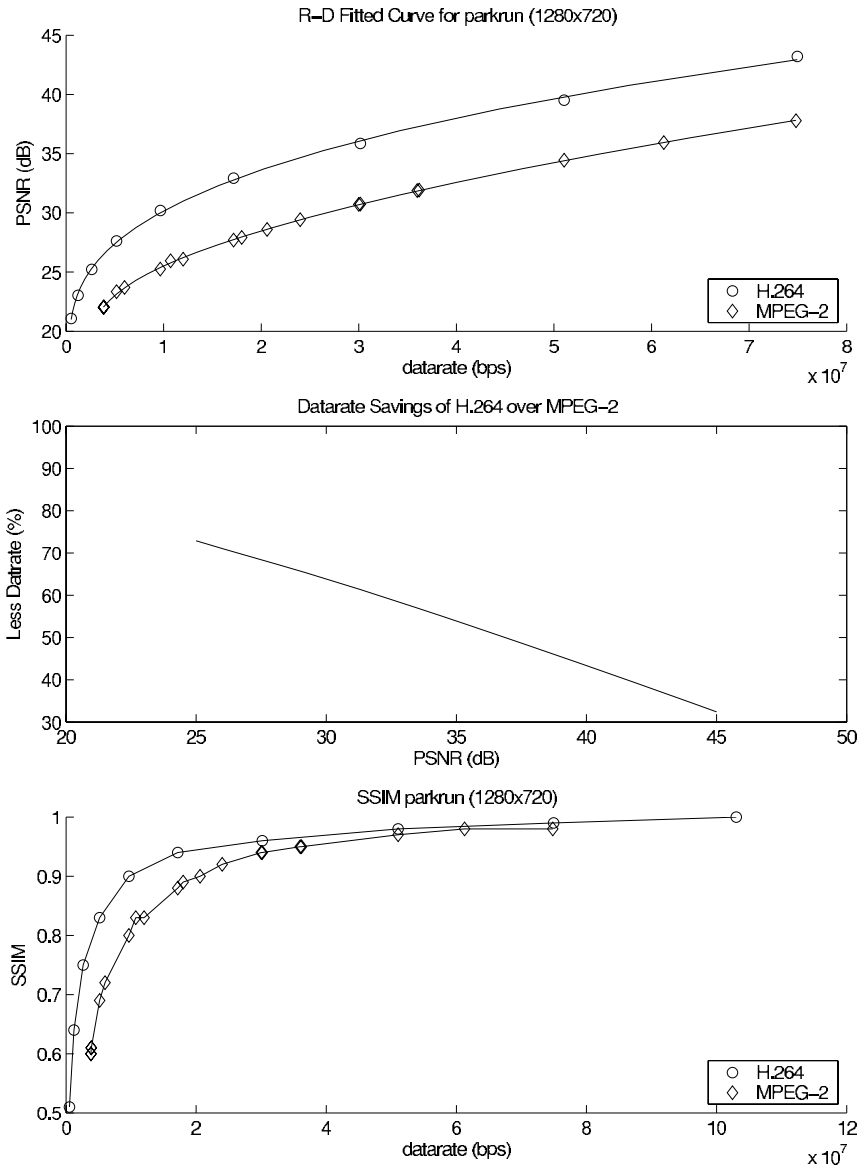


Fig. B.42. R-D Plot for parkrun sequence 1280 × 720 25 fps.

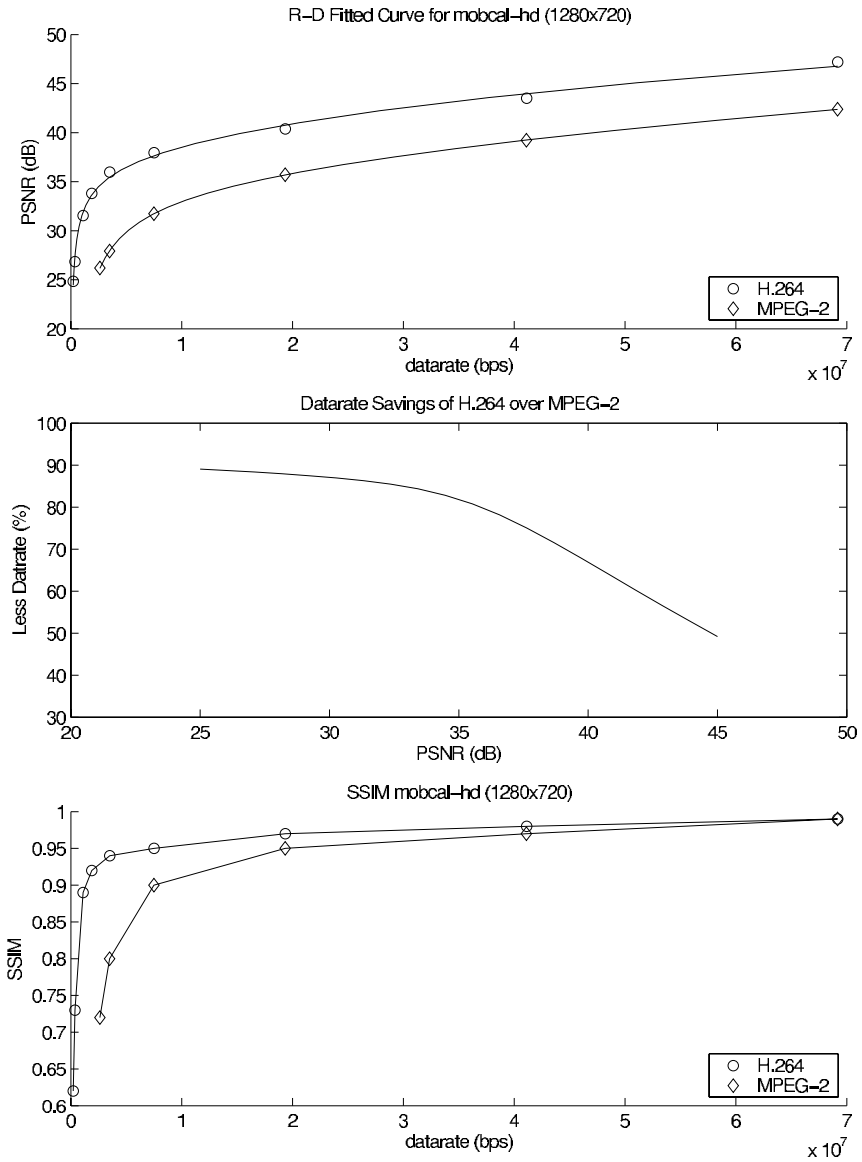


Fig. B.43. R-D Plot for mobcal sequence 1280 × 720 25 fps.

Table B.1
Full results for QCIF (176 × 144) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
akiyo	54.3	27.62	0.79	10.8	30.84	0.88	3.22	0.09
-	54.4	27.65	0.79	23.2	36.13	0.95	8.48	0.16
-	56.9	28.48	0.83	56.7	41.68	0.98	13.2	0.15
-	150.9	39.19	0.98	150.7	47.25	0.99	8.06	0.01
-	410.8	44.96	0.99	410.3	52.42	1	7.46	0.01
carphone	69.9	26.09	0.78	26.0	28.99	0.87	2.9	0.09
-	78.5	27.27	0.82	79.5	34.01	0.95	6.74	0.13
-	233.2	34.26	0.95	233.5	39.53	0.98	5.27	0.03
-	626.1	39.69	0.98	626.4	45.15	0.99	5.46	0.01
-	1676.2	45.26	0.99	1676.1	50.8	1	5.54	0.01
claire	54.4	29.04	0.87	9.7	32.17	0.92	3.13	0.05
-	54.5	29.06	0.87	21.5	37.65	0.97	8.59	0.1
-	56.5	29.92	0.88	56.7	43.03	0.99	13.11	0.11
-	163.7	40.98	0.98	163.4	48.07	0.99	7.09	0.01
-	572.7	46.2	0.99	572.8	52.19	1	5.99	0.01
coastguard	27.5	25.15	0.64	23.2	26.96	0.73	1.81	0.09
-	27.9	25.29	0.77	86.5	31.52	0.88	6.23	0.11
-	57.8	31.71	0.94	315.7	37.17	0.96	5.46	0.02
-	168.8	39.53	0.98	914.9	43.67	0.99	4.14	0.01
-	462.9	45.41	1	2014.4	50.5	1	5.09	0
container	55.0	25.15	0.78	20.1	29.33	0.88	4.18	0.1
-	55.7	25.29	0.78	45.2	34.41	0.93	9.12	0.15
-	115.7	31.71	0.93	114.8	39.52	0.96	7.81	0.03
-	337.5	39.53	0.98	336.6	45.44	0.99	5.91	0.01
-	925.9	45.41	0.99	925.4	50.87	1	5.46	0.01
foreman	74.8	25.53	0.74	33.0	28.63	0.84	3.1	0.1
-	85.1	26.72	0.78	86.0	33.24	0.93	6.52	0.15
-	251.2	32.84	0.93	251.1	38.38	0.97	5.54	0.04
-	760.2	38.51	0.98	760.2	44.34	0.99	5.83	0.01
-	1946.3	44.44	0.99	1945.8	50.72	1	6.28	0.01

Table B.2
Full results for QCIF (176×144) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
glasgow	81.2	23.66	0.64	58.1	25.95	0.77	2.29	0.13
-	201.9	27.53	0.85	202.7	30.86	0.92	3.33	0.07
-	612.6	33.02	0.95	612.5	36.87	0.98	3.85	0.03
-	1570.9	39.16	0.99	1570.2	43.66	0.99	4.5	0
-	3080.2	45	1	3077.7	50.67	1	5.67	0
mthr-dgthr	57.2	27.42	0.76	18.3	30.08	0.84	2.66	0.08
-	57.6	27.5	0.76	48.1	34.53	0.94	7.03	0.18
-	136.8	34.61	0.94	136.5	39.96	0.98	5.35	0.04
-	399.8	40.53	0.98	399.9	45.73	0.99	5.2	0.01
-	1300.4	45.89	0.99	1300.1	50.98	1	5.09	0.01
news	61.1	24.29	0.75	31.4	28.91	0.88	4.62	0.13
-	73.8	26.24	0.83	73.7	34.44	0.95	8.2	0.12
-	167.9	32.83	0.95	167.2	40.34	0.98	7.51	0.03
-	368.0	38.89	0.98	367.4	46.3	0.99	7.41	0.01
-	831.6	44.75	0.99	831.8	51.81	1	7.06	0.01
salesman	53.2	25.67	0.65	21.2	28.28	0.77	2.61	0.12
-	58.3	26.65	0.72	58.4	33.4	0.92	6.75	0.2
-	146.2	33.89	0.94	146.8	39.35	0.98	5.46	0.04
-	349.2	39.62	0.98	349.4	45.34	0.99	5.72	0.01
-	1051.2	46.07	1	1051.1	50.95	1	4.88	0

Table B.3
Full results for CIF (352 × 240) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
flowergarden	366.2	19.96	0.67	349.6	23.67	0.83	3.71	0.16
-	509.4	19.83	0.67	1117.3	29.63	0.95	9.8	0.28
-	3020.6	30.98	0.97	3019.7	36.28	0.98	5.3	0.01
-	6658.6	37.45	0.99	6657.6	43.45	0.99	6	0
-	11 854.7	43.32	1	11 851.7	50.5	1	7.18	0
tennis	272.9	23.11	0.58	167.9	26.48	0.66	3.37	0.08
-	643.3	27.05	0.75	635.8	30.66	0.8	3.61	0.05
-	2137.9	32.18	0.9	2127.3	36.01	0.93	3.83	0.03
-	6057.7	39.15	0.98	6042.9	43.2	0.99	4.05	0.01
-	11 404.8	44.82	0.98	11 385.9	50.61	1	5.79	0.02
tv18_new	147.1	26.51	0.7	51.7	29.3	0.8	2.79	0.1
-	148.7	26.58	0.71	144.4	33.98	0.92	7.4	0.21
-	453.5	33.59	0.93	451.2	38.9	0.97	5.31	0.04
-	2369.5	40.24	0.98	2368.6	43.86	0.99	3.62	0.01
-	8011.3	46.37	0.99	8010.3	50.65	1	4.28	0.01
foot_cif_cr	344.2	23.02	0.57	294.4	25.48	0.66	2.46	0.09
-	955.5	26.91	0.81	952.2	30.07	0.85	3.16	0.04
-	2686.1	31.95	0.93	2685.4	36.12	0.96	4.17	0.03
-	6389.9	38.49	0.98	6391.6	43.26	0.99	4.77	0.01
-	11 565.6	44.36	1	11 568.3	50.59	1	6.23	0
news2_cr	160.8	30.19	0.84	31.5	32.47	0.89	2.28	0.05
-	161.4	30.25	0.84	84.1	37.26	0.95	7.01	0.11
-	238.2	35.31	0.94	236.7	41.94	0.98	6.63	0.04
-	864.5	41.91	0.98	864.2	45.97	0.99	4.06	0.01
-	4170.2	46.62	0.99	4168.7	51.28	1	4.66	0.01

Table B.4
Full results for CIF (352 × 240) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
salescut_cr	159.7	26.53	0.69	58.3	29.06	0.79	2.53	0.1
-	161.6	26.62	0.69	161.0	33.8	0.92	7.18	0.23
-	476.6	33.99	0.93	474.9	38.23	0.97	4.24	0.04
-	502.8	33.16	0.92	-	-	-	-	-
-	753.0	35.22	0.95	-	-	-	-	-
-	1003.0	36.66	0.96	-	-	-	-	-
-	1955.5	40.6	0.98	1953.0	43.9	0.99	3.3	0.01
-	2002.6	40.17	0.98	-	-	-	-	-
-	3002.0	42.26	0.99	-	-	-	-	-
-	4001.8	43.82	0.99	-	-	-	-	-
-	5001.6	45.16	0.99	-	-	-	-	-
-	5707.9	46.53	1	5706.5	50.7	1	4.17	0

Table B.5
Full results for CIF (352 × 288) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
akiyo	168.3	30.02	0.86	19.4	28.12	0.82	-1.9	-0.04
-	168.4	30.03	0.86	33.7	31.91	0.89	1.88	0.03
-	168.6	30.04	0.86	40.9	33.1	0.91	3.06	0.05
-	169.2	30.1	0.86	91.8	37.95	0.96	7.85	0.1
-	234.3	35.27	0.94	232.4	42.61	0.98	7.34	0.04
-	502.4	39.68	0.97	-	-	-	-	-
-	681.0	42.9	0.98	679.8	46.94	0.99	4.04	0.01
-	752.1	42.43	0.98	-	-	-	-	-
-	1001.7	43.69	0.99	-	-	-	-	-
-	2000.6	46.2	0.99	-	-	-	-	-
-	2261.6	47.28	0.99	2261.3	52.75	1	5.47	0.01
-	3000.4	48.03	0.99	-	-	-	-	-
-	4000.7	49.82	1	-	-	-	-	-
-	5000.5	51.3	1	-	-	-	-	-
bus150	443.2	23.31	0.66	166.2	26.59	0.77	3.28	0.11
-	543.9	23.93	0.7	536.7	31.59	0.91	7.66	0.21
-	567.9	23.3	0.66	-	-	-	-	-
-	753.4	24.6	0.74	-	-	-	-	-
-	1002.5	25.96	0.8	-	-	-	-	-
-	1653.0	29.16	0.89	1637.9	37.44	0.97	8.28	0.08
-	2001.4	29.4	0.9	-	-	-	-	-
-	3000.6	31.88	0.94	-	-	-	-	-
-	3999.8	33.94	0.96	-	-	-	-	-
-	4442.9	35.69	0.97	4408.0	43.75	0.99	8.06	0.02
-	4999.4	35.72	0.97	-	-	-	-	-
-	9387.1	42.2	0.99	9314.7	50.45	1	8.25	0.01

Table B.6
Full results for CIF (352 × 288) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
carphone	250.7	28.1	0.82	32.0	26.1	0.76	-2	-0.06
-	251.1	28.12	0.82	71.8	29.58	0.85	1.46	0.03
-	251.2	28.12	0.82	79.2	30.7	0.87	2.58	0.05
-	267.0	28.92	0.84	246.3	35.43	0.94	6.51	0.1
-	502.9	32.22	0.9	-	-	-	-	-
-	709.9	35.58	0.94	708.6	40.11	0.97	4.53	0.03
-	751.9	34.94	0.94	-	-	-	-	-
-	1001.8	36.46	0.95	-	-	-	-	-
-	2001.4	39.77	0.97	-	-	-	-	-
-	2266.2	40.81	0.98	2265.8	44.62	0.98	3.81	0
-	3000.8	41.62	0.98	-	-	-	-	-
-	4000.6	42.98	0.98	-	-	-	-	-
-	5000.3	44.08	0.99	-	-	-	-	-
-	7080.2	46.28	0.99	7080.3	50.91	1	4.63	0.01
foreman	292.9	27.11	0.75	38.7	25.31	0.68	-1.8	-0.07
-	293.3	27.12	0.75	75.7	28.58	0.79	1.46	0.04
-	293.3	27.12	0.75	78.3	29.54	0.81	2.42	0.06
-	295.2	27.2	0.75	220.1	33.97	0.91	6.77	0.16
-	503.6	29.26	0.81	-	-	-	-	-
-	706.5	32.84	0.9	701.6	38.93	0.96	6.09	0.06
-	752.7	32.25	0.89	-	-	-	-	-
-	1002.7	33.79	0.92	-	-	-	-	-
-	2001.8	37.34	0.96	-	-	-	-	-
-	2497.9	39.01	0.97	2487.1	44.4	0.99	5.39	0.02
-	3000.6	39.52	0.97	-	-	-	-	-
-	4000.0	41.15	0.98	-	-	-	-	-
-	4999.7	42.46	0.99	-	-	-	-	-
-	7241.5	45.14	0.99	7213.7	50.74	1	5.6	0.01

Table B.7
Full results for CIF (352 × 288) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
missA_cr	223.0	33.36	0.88	16.2	30.08	0.84	-3.28	-0.04
-	223.1	33.37	0.88	27.0	33.66	0.89	0.29	0.01
-	223.1	33.36	0.88	31.9	34.61	0.9	1.25	0.02
-	223.4	33.42	0.88	73.7	37.98	0.93	4.56	0.05
-	291.9	37.7	0.93	291.4	40.56	0.95	2.86	0.02
-	500.8	38.85	0.94	-	-	-	-	-
-	751.0	39.86	0.95	-	-	-	-	-
-	1001.2	40.36	0.95	-	-	-	-	-
-	2001.4	41.66	0.96	-	-	-	-	-
-	2251.5	42.3	0.97	2248.5	44.54	0.97	2.24	0
-	3001.6	42.7	0.97	-	-	-	-	-
-	4001.7	43.65	0.97	-	-	-	-	-
-	5001.6	44.45	0.98	-	-	-	-	-
-	7750.1	47.28	0.99	7743.0	51.31	0.99	4.03	0
mobile	315.3	21.23	0.67	97.9	20.7	0.63	-0.53	-0.04
-	317.6	21.26	0.67	208.5	24.22	0.81	2.96	0.14
-	319.1	21.28	0.67	271.2	25.52	0.86	4.24	0.19
-	503.9	22.21	0.74	-	-	-	-	-
-	754.3	24.1	0.83	-	-	-	-	-
-	792.6	25.32	0.87	788.9	30.81	0.95	5.49	0.08
-	1004.8	25.43	0.87	-	-	-	-	-
-	2005.6	28.91	0.94	-	-	-	-	-
-	2358.2	30.81	0.96	2352.1	36.96	0.99	6.15	0.03
-	3005.8	31.36	0.96	-	-	-	-	-
-	4005.1	33.39	0.97	-	-	-	-	-
-	5004.5	35.18	0.98	-	-	-	-	-
-	5848.1	37.45	0.99	5844.4	43.76	1	6.31	0.01
-	11 561.9	43.67	1	11 560.8	50.54	1	6.87	0

Table B.8
Full results for CIF (352 × 288) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
news	195.5	26.7	0.82	32.1	25.69	0.78	-1.01	-0.04
-	196.0	26.71	0.82	63.1	29.47	0.87	2.76	0.05
-	196.3	26.72	0.82	79.3	30.79	0.89	4.07	0.07
-	198.8	26.83	0.83	190.2	36.02	0.95	9.19	0.12
-	459.3	34.25	0.96	457.0	41.22	0.98	6.97	0.02
-	502.4	33.23	0.95	-	-	-	-	-
-	752.3	36.44	0.97	-	-	-	-	-
-	1002.1	38.54	0.98	-	-	-	-	-
-	1180.2	40.71	0.98	1178.1	46.13	0.99	5.42	0.01
-	2001.4	42.9	0.99	-	-	-	-	-
-	3000.8	44.95	0.99	-	-	-	-	-
-	3590.0	46.52	0.99	3589.0	52.08	1	5.56	0.01
-	4001.0	46.57	0.99	-	-	-	-	-
-	5000.5	47.81	1	-	-	-	-	-

Table B.9
Full results for (704 × 480) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
football2	1447.0	24.54	0.60	758.9	26.81	0.68	2.27	0.68
-	2386.4	27.53	0.78	2376.5	31.55	0.87	4.02	0.87
-	7032.9	33.58	0.94	7026.7	37.17	0.96	3.59	0.96
-	20 848.7	39.95	0.98	20 849.6	43.39	0.99	3.44	0.99
-	43 861.4	45.58	0.99	43 861.4	50.48	1	4.9	1

Table B.10
Full results for VCEG (720 × 480) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src2	1117.8	22.97	0.58	623.8	25.8	0.74	2.83	0.16
-	2461.1	26.35	0.8	2460.0	30.61	0.9	4.26	0.1
-	4166.9	28.25	0.87	-	-	-	-	-
-	8333.3	31.06	0.93	-	-	-	-	-
-	8564.1	31.19	0.93	8560.9	36.26	0.96	5.07	0.03
-	12 499.9	33.1	0.95	-	-	-	-	-
-	16 666.5	34.78	0.96	-	-	-	-	-
-	20 833.2	36.26	0.97	-	-	-	-	-
-	24 999.7	37.59	0.98	-	-	-	-	-
-	25 050.6	37.61	0.98	25 040.3	43.1	0.99	5.49	0.01
-	48 347.1	43.57	0.99	48 325.7	50.32	1	6.75	0.01
src3	1324.8	24.49	0.72	408.7	27.49	0.79	3	0.07
-	1452.4	25.46	0.77	1454.2	32.42	0.89	6.96	0.12
-	4166.6	31.82	0.9	-	-	-	-	-
-	4890.8	32.44	0.91	4888.8	37.05	0.93	4.61	0.02
-	8332.9	34.6	0.93	-	-	-	-	-
-	12 499.9	36.62	0.95	-	-	-	-	-
-	16 666.4	38.3	0.96	-	-	-	-	-
-	19 360.0	39.36	0.97	19351.7	43.25	0.98	3.89	0.01
-	20 833.0	39.86	0.97	-	-	-	-	-
-	25 000.0	41.32	0.98	-	-	-	-	-
-	39 525.3	45.04	0.99	39507.3	50.72	1	5.68	0.01

Table B.11
Full results for VCEG (720 × 480) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src4	821.2	25.15	0.72	140.6	27.66	0.78	2.51	0.06
-	832.1	25.33	0.73	-	-	-	-	-
-	845.3	25.76	0.74	-	-	-	-	-
-	897.2	26.28	0.76	897.9	32.78	0.95	6.5	0.19
-	1666.0	29.66	0.89	-	-	-	-	-
-	2408.1	31.87	0.94	2407.6	40.46	0.99	8.59	0.05
-	2499.9	32.13	0.94	-	-	-	-	-
-	3333.3	34.41	0.96	-	-	-	-	-
-	4166.7	36.43	0.98	-	-	-	-	-
-	4517.7	37.21	0.98	4516.3	48.3	1	11.09	0.02
-	7076.2	41.82	0.99	7073.0	55.78	1	13.96	0.01
-	8334.1	43.6	0.99	-	-	-	-	-
-	12 501.1	48.73	1	-	-	-	-	-
-	16 667.5	53.25	1	-	-	-	-	-
-	20 830.6	57.37	1	-	-	-	-	-
-	24 769.9	60.69	1	-	-	-	-	-
src5	1768.5	25.27	0.64	708.1	27.67	0.74	2.4	0.1
-	1770.3	25.28	0.64	-	-	-	-	-
-	1773.0	25.29	0.64	-	-	-	-	-
-	1799.7	25.39	0.65	-	-	-	-	-
-	2335.8	26.94	0.74	2335.1	32.96	0.91	6.02	0.17
-	2500.0	27.33	0.76	-	-	-	-	-
-	3333.1	29.05	0.83	-	-	-	-	-
-	4166.1	30.46	0.87	-	-	-	-	-
-	6257.6	33.2	0.92	6255.7	38.67	0.97	5.47	0.05
-	8332.6	35.3	0.95	-	-	-	-	-
-	12 499.0	38.42	0.97	-	-	-	-	-
-	15 235.5	39.97	0.98	15230.3	44.44	0.99	4.47	0.01
-	16 665.3	40.66	0.98	-	-	-	-	-
-	20 831.9	42.35	0.99	-	-	-	-	-
-	24 998.1	43.7	0.99	-	-	-	-	-
-	33 568.9	45.88	0.99	33 555.8	50.95	1	5.07	0.01

Table B.12
Full results for VCEG (720 × 480) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src6	1621.9	24.76	0.61	607.3	27.27	0.7	2.51	0.09
-	1622.3	24.76	0.61	-	-	-	-	-
-	1627.2	24.8	0.61	-	-	-	-	-
-	1726.9	25.46	0.63	-	-	-	-	-
-	2500.2	27.33	0.72	-	-	-	-	-
-	2745.2	27.79	0.74	2743.9	32.36	0.89	4.57	0.15
-	3334.0	28.74	0.78	-	-	-	-	-
-	4167.5	29.85	0.82	-	-	-	-	-
-	8334.6	33.78	0.92	-	-	-	-	-
-	8428.8	33.85	0.92	8424.0	38.09	0.96	4.24	0.04
-	12 501.3	36.41	0.96	-	-	-	-	-
-	16 667.8	38.39	0.97	-	-	-	-	-
-	20 091.1	39.75	0.98	20 081.3	44.31	0.99	4.56	0.01
-	20 834.3	40.02	0.98	-	-	-	-	-
-	25 000.8	41.38	0.99	-	-	-	-	-
-	39 578.7	45.08	0.99	39 560.9	51.24	1	6.16	0.01
src7	1465.1	28.84	0.76	439.8	30.42	0.85	1.58	0.09
-	1473.0	28.89	0.77	1174.9	34.62	0.91	5.73	0.14
-	1649.1	27.41	0.81	-	-	-	-	-
-	1668.4	27.53	0.81	-	-	-	-	-
-	1992.9	31.05	0.85	-	-	-	-	-
-	3000.4	34.24	0.88	-	-	-	-	-
-	3481.0	33.05	0.91	3477.5	38.35	0.95	5.3	0.04
-	4001.6	35.93	0.93	-	-	-	-	-
-	5001.8	37.01	0.94	-	-	-	-	-
-	10 001.2	40.04	0.96	-	-	-	-	-
-	12 909.6	39.72	0.96	12903.0	43.74	0.98	4.02	0.02
-	15 000.8	41.95	0.98	-	-	-	-	-
-	20 000.4	43.47	0.98	-	-	-	-	-
-	25 001.0	44.76	0.99	-	-	-	-	-
-	30 000.5	45.8	0.99	0.0			-45.8	-0.99
-	30 647.2	45.02	0.99	30633.1	50.92	1	5.9	0.01

Table B.13
Full results for VCEG (720 × 480) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src8	1530.6	27.09	0.83	356.9	31	0.89	3.91	0.06
-	1536.4	27.14	0.83	895.8	36.14	0.95	9	0.12
-	1541.9	27.19	0.83	-	-	-	-	-
-	1543.8	27.2	0.83	-	-	-	-	-
-	1666.2	27.61	0.84	-	-	-	-	-
-	2305.9	29.54	0.89	2304.9	41.02	0.97	11.48	0.08
-	2500.2	29.96	0.89	-	-	-	-	-
-	3333.8	31.48	0.92	-	-	-	-	-
-	4167.1	32.7	0.93	-	-	-	-	-
-	7479.3	36.04	0.95	7476.1	44.8	0.98	8.76	0.03
-	8333.5	36.7	0.96	-	-	-	-	-
-	12 500.2	39.23	0.97	-	-	-	-	-
-	16 667.2	40.95	0.98	-	-	-	-	-
-	20 833.8	42.26	0.98	-	-	-	-	-
-	25 000.7	43.35	0.98	-	-	-	-	-
-	26 549.2	43.73	0.99	26 536.8	51.01	1	7.28	0.01
src9	1843.8	25.19	0.65	853.5	27.41	0.72	2.22	0.07
-	2714.2	27.4	0.77	2715.2	32.28	0.88	4.88	0.11
-	7443.8	33.46	0.93	7437.7	37.87	0.96	4.41	0.03
-	18 651.9	40.14	0.98	18 644.7	44.08	0.99	3.94	0.01
-	38 791.1	45.78	0.99	38 773.8	50.9	1	5.12	0.01

Table B.14
Full results for VCEG (720 × 480) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src10	1285.3	21.73	0.64	606.1	25.55	0.8	3.82	0.16
-	1978.7	24.51	0.79	1979.5	31.02	0.92	6.51	0.13
-	6692.8	31.09	0.94	6690.1	36.91	0.97	5.82	0.03
-	20 194.4	38.2	0.98	20 186.0	43.35	0.99	5.15	0.01
-	42 022.7	44.29	0.99	42 005.1	50.38	1	6.09	0.01

Table B.15
Full results for (720 × 480) sequences.

Sequence	MPEG-2 kbps	MPEG-2 PSNR	MPEG-2 SSIM	H.264 Kb/s	H.264 PSNR	H.264 SSIM	Δ PSNR	Δ SSIM
flowergarden_cr	1580.7	21.92	0.7	786.2	25.32	0.82	3.4	0.12
-	2559.9	24.96	0.86	2546.7	31.04	0.94	6.08	0.08
-	7324.0	31.41	0.96	7313.9	37.07	0.98	5.66	0.02
-	18 744.2	38.09	0.99	18 744.4	43.6	0.99	5.51	0
-	37 811.5	43.85	1	37 814.0	50.49	1	6.64	0
hockey1_cr	1527.0	29.51	0.81	314.4	31.68	0.84	2.17	0.03
-	1539.6	29.62	0.81	811.9	36.12	0.92	6.5	0.11
-	2425.1	34.05	0.9	2421.9	40.43	0.96	6.38	0.06
-	8340.3	42.11	0.98	8341.2	45.7	0.99	3.59	0.01
-	21 149.0	47.6	0.99	21 150.0	51.6	1	4	0.01
mobilcal_cr	1399.9	20.9	0.62	687.7	25.08	0.82	4.18	0.2
-	2182.6	23.75	0.79	2162.0	30.65	0.93	6.9	0.14
-	6853.3	30.24	0.93	6814.3	36.37	0.97	6.13	0.04
-	20 083.4	37.31	0.98	20 055.8	43.08	0.99	5.77	0.01
-	39 964.9	43.4	0.99	39 954.7	50.29	1	6.89	0.01
tennis2_cr	1246.6	24.19	0.6	348.0	27.39	0.67	3.2	0.07
-	1306.0	24.46	0.63	1331.4	31.46	0.83	7	0.2
-	5318.3	31.69	0.88	5303.7	36.33	0.93	4.64	0.05
-	19 144.1	39.06	0.97	19138.6	43.17	0.98	4.11	0.01
-	38 059.5	44.77	0.99	38059.8	50.47	1	5.7	0.01

Table B.16
Full results for VCEG (720 × 576) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src13	1416.7	26.2	0.75	691.1	28.5	0.82	2.3	0.07
-	1897.6	28.65	0.84	1895.0	33.58	0.93	4.93	0.09
-	5268.5	34.59	0.94	5268.2	38.56	0.97	3.97	0.03
-	15 977.0	40.26	0.98	15 970.9	44.04	0.99	3.78	0.01
-	35 751.2	45.13	0.99	35 732.4	51.05	1	5.92	0.01
src14	1108.3	29.39	0.82	167.3	31.28	0.85	1.89	0.03
-	1111.0	29.5	0.82	465.6	35.59	0.92	6.09	0.1
-	1595.9	33.84	0.92	1595.1	40.2	0.96	6.36	0.04
-	6102.3	40.65	0.98	6097.5	45.07	0.98	4.42	0
-	21 895.8	45.94	0.99	21 878.4	50.83	1	4.89	0.01
src15	1420.7	20.86	0.63	739.9	25.11	0.83	4.25	0.2
-	2425.5	24.11	0.8	2422.4	30.61	0.93	6.5	0.13
-	7783.7	30.4	0.93	7779.8	36.4	0.97	6	0.04
-	22 074.8	37.19	0.98	22 058.7	43.08	0.99	5.89	0.01
-	43 916.4	43.31	0.99	43 882.9	50.3	1	6.99	0.01
src16	1012.2	29.07	0.83	203.8	31.99	0.88	2.92	0.05
-	1018.1	29.12	0.83	492.3	36.89	0.95	7.77	0.12
-	1212.4	31.24	0.87	1207.6	42.4	0.98	11.16	0.11
-	2907.9	39.01	0.97	2905.1	47.88	0.99	8.87	0.02
-	6955.8	44.39	0.99	6951.0	53.12	1	8.73	0.01

Table B.17
Full results for VCEG (720 × 576) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src17	1563.3	24.58	0.77	872.7	29.55	0.92	4.97	0.15
-	1914.0	26.01	0.82	1916.2	35.19	0.97	9.18	0.15
-	4032.8	30.06	0.9	4035.3	41.3	0.99	11.24	0.09
-	7772.6	33.84	0.95	7774.7	47.57	1	13.73	0.05
-	13 012.5	37.58	0.97	13 011.1	53.44	1	15.86	0.03
src18	903.8	26.99	0.58	161.2	28.59	0.66	1.6	0.08
-	908.3	27.09	0.58	537.7	33.16	0.88	6.07	0.3
-	2129.1	33.44	0.9	2127.4	38.13	0.96	4.69	0.06
-	11 589.6	40.08	0.98	11 580.6	43.48	0.99	3.4	0.01
-	33 539.8	46.05	0.99	33 513.9	50.41	1	4.36	0.01
src19	1694.5	27.06	0.68	626.9	29.05	0.72	1.99	0.04
-	1899.5	27.63	0.71	1897.6	33.39	0.87	5.76	0.16
-	5338.7	34.22	0.91	5336.6	38.31	0.95	4.09	0.04
-	16 482.1	40.58	0.98	16 471.1	43.71	0.98	3.13	0
-	38 699.7	46.03	0.99	38 671.1	50.93	1	4.9	0.01
src20	962.3	22.4	0.74	234.6	27.51	0.87	5.11	0.13
-	969.5	22.5	0.74	634.7	32.83	0.94	10.33	0.2
-	1895.8	29.15	0.92	1895.3	38.06	0.97	8.91	0.05
-	10 067.6	39.16	0.98	10 060.8	43.67	0.99	4.51	0.01
-	29 768.1	45.12	0.99	29 745.4	50.66	1	5.54	0.01

Table B.18
Full results for VCEG (720 × 576) sequences.

	MPEG-2			H.264			Δ	Δ
Sequence	kbps	PSNR	SSIM	Kb/s	PSNR	SSIM	PSNR	SSIM
src21	1070.1	31.47	0.81	103.9	32.47	0.82	1	0.01
-	1070.3	31.36	0.8	291.6	35.97	0.9	4.61	0.1
-	1087.9	32.18	0.82	1029.8	39.83	0.95	7.65	0.13
-	6275.1	41.4	0.96	6271.0	44.03	0.97	2.63	0.01
-	24 589.9	46.81	0.99	24 571.3	50.91	0.99	4.1	0
src22	1276.0	23.5	0.7	497.8	27.02	0.84	3.52	0.14
-	1767.6	26.14	0.83	1763.7	31.98	0.93	5.84	0.1
-	5983.1	32.91	0.95	5978.9	37.25	0.97	4.34	0.02
-	19 554.2	39.07	0.98	19 539.1	43.27	0.99	4.2	0.01
-	42 232.0	44.69	0.99	42 200.1	50.54	1	5.85	0.01

LIST OF REFERENCES

LIST OF REFERENCES

[1] *Television – 1280 × 720 Progressive Image Sample Structure – Analog and Digital Representation and Analog Interface*, Society of Motion Picture and Television Engineers Std. 296M, 2001.

[2] J. Taylor, *DVD Demystified*. Dallas, TX: McGraw-Hill, 1998.

[3] K. B. Benson and D. G. Fink, *HDTV: Advanced Television for the 1990s*. Dallas, TX: McGraw-Hill, 1990.

[4] *Generic Coding of Moving Pictures and Associated Audio Information - Part 2: Video*, ITU-T and ISO/IEC JTC 1 Std. 13 818-2, 1994.

[5] B. G. Haskell, A. Puri, and A. N. Netravalli, *Digital Video: An Introduction to MPEG-2*. New York, NY: Chapman and Hall, 1997.

[6] *Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding*, ITU-T and ISO/IEC JTC 1 Std. ITU-T Recommendation H.264 and ISO/IEC 13 818-10 (MPEG-4), 1994.

[7] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[8] J. G. Proakis and D. Manolakis, *Digital Signal Processing: Principles, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[9] G. Wyszecki and W. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*. New York, NY: Wiley, 1982.

[10] R. M. Boynton, *Human Color Vision*. New York, NY: Holt, Rinehard and Winston, 1979.

[11] K. Jack, *Video Demystified*. San Deigo, CA: HighText Interactive, 1996.

[12] C. A. Poynton, *A Technical Introduction to Digital Video*. New York, NY: Wiley, 1996.

[13] *Encoding Parameters of Digital Televisions for Studios*, International Telecommunication Union Std. Recommendation ITU-R BT.601-4, 1990.

[14] G. Sullivan, "Approximate theoretical analysis of RGB to YCbCr to RGB conversion error," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-I017*, July 2003.

[15] A. M. Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[16] G. Wallace, "The JPEG still picture compression standard," *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, Sept. 1991.

[17] N. Ahmed, T. R. Natarajan, and K. R. Rao, "On image processing and a discrete cosine transform," *IEEE Transactions on Computers*, vol. 23, no. 1, pp. 90–93, Jan. 1974.

[18] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 7, pp. 379–423, July 1948.

[19] D. Huffman, "A method for the construction of minimum redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, Sept. 1952.

[20] S. W. Golomb, "Run-length encoding," *IEEE Transactions on Information Theory*, vol. IT-12, no. 3, pp. 399–401, July 1966.

[21] I. Witten, R. Neal, and J. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 857–865, June 1987.

[22] R. Schafer and T. Sikora, "Digital video coding standards and their role in video communications," *Proceedings of the IEEE*, vol. 83, no. 6, pp. 907–924, June 1995.

[23] International Telecommunication Union. `http://www.itu.int/home/`

[24] L. Chiariglione, "Impact of MPEG on multimedia industry," *Proceedings of the IEEE*, vol. 86, no. 6, pp. 1222–1227, June 1998.

[25] ——, "MPEG and multimedia communcations," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 5–17, Feb. 1997.

[26] C. Segall and A. Katsaggelos, "Pre- and post-processing algorithms for compressed video enhancement," *Conference Record of the Thrity-Fourth Asilomar Conference on Signals, Systems, ad Computers*, vol. 2, Oct. 2000, pp. 1369–1373.

[27] P. Salama, N. B. Shroff, and E. J. Delp, "Error concealment in encoded video," *IEEE Journal on Selected Areas in Commuincations*, vol. 18, no. 6, pp. 1129–1144, June 2000.

[28] A. Zakhor, "Iterative procedures for reduction of blocking effects in transform image coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 91–95, Mar. 1992.

[29] *Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - Part 2: Video*, ISO/IEC JTC1 Std. ISO/IEC 11 172-2 (MPEG-1), Mar. 1993.

[30] D. LeGall, "MPEG: A video compression standard for multimedia applications," *Communications of the ACM*, vol. 34, no. 4, pp. 46–58, Apr. 1991.

[31] *Coding of audio-visual objects-Part 2: Visual*, ITU-T and ISO/IEC JTC 1 Std. ISO/IEC 14 496-2 (MPEG-4 Visual Version 1), 1999.

[32] Divx. `http://www.divx.com`

[33] Xivd. http://www.xvid.org

[34] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, and H. Watanabe, "Two-stage motion compensation using adaptive global MC and local affine MC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 75–85, Feb. 1997.

[35] T. Talluri, "Error-resilent video coding in the ISO MPEG-4 standard," *IEEE Communcations Magazine*, vol. 36, no. 6, pp. 112–119, June 1988.

[36] K. Panusopone, X. Chenand, R. Eifrig, and A. Luthra, "Coding tools in MPEG-4 for interlaced video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 5, pp. 755–766, August 2000.

[37] Joint Video team (JVT). http://www.itu.int/ITU-T/studygroups/com16/jvt/

[38] T. Wiegand, G. J. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[39] I. E. Richardson, *H.264 and MPEG-4 Video Compression*. Hoboken, NJ: Wiley, 2003.

[40] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 657–673, July 2003.

[41] S. wenger, "Low-complexity transform and quantization in H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 645–656, July 2003.

[42] T. Wedi and H. Musmann, "Motion- and aliasing-compensated prediction for hybrid video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 577–586, July 2003.

[43] T. Wiegand, X. Zhang, and B. Girod, "Long-term memory motion-compensated prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 70–84, Feb. 1999.

[44] P. List, A. Joch, J. Lainema, G. Bjntegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 614–619, July 2003.

[45] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Boston, USA: Academic Press, 1990.

[46] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*. Norwell, MA: Kluwer, 1997.

[47] H. Malvar, "Low-complexity length-4 tranform and quantization with 16-bit arithmetic," *ITU-T Q.6/SG16 (VCEG) Document*, Sept. 2001.

[48] H. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 598–603, July 2003.

[49] *Information technology – JPEG 2000 image coding system: Core coding system*, ISO/IEC JTC1 Std. ISO/IEC 15 444-1, Jan. 2000.

[50] T. Halbach, "Performance comparison: H.26L intra coding vs. JPEG2000," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-D039*, July 2002.

[51] H. S. Malvar, *Signal Professing with Lapped Transforms*. Boston, USA: Artech House, 1992.

[52] D. Marpe, H. Schwarz, and T. Wiegand, "Context-base adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620–636, July 2003.

[53] A. Joch, "Demonstration of main profile decoder on TI C64x," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-F075*, Dec. 2002.

[54] B. Girod, "Motion-compensating prediction with fractional-pel accuracy," *IEEE Transactions on Communications*, vol. 41, no. 4, pp. 604–612, Apr. 1993.

[55] H. G. H. G. Musmann, P. Pirsh, "Advances in picture coding," *Proceedings of the IEEE*, vol. 73, no. 4, pp. 523–548, Apr. 1985.

[56] R. L. Schmidt, A. Puri, and B. G. Haskell, "Performance evaluation of nonscalable MPEG-2 video coding," *Proceedings of the SPIE Visual Communications and Image Processing*, vol. 2308, Sept. 1994, pp. 296–310.

[57] T. Wiegand, X. Zhang, and B. Girod, "Long-term memory motion-compensated prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 70–84, Feb. 1999.

[58] J. Boyce, "Coding efficiency of various numbers of reference frames," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-B060*, Feb. 2002.

[59] M. Chan, Y. Yu, and A. Constantinides, "Variable size block matching motion compensation with applications to video coding," *IEEE Proceedings on Communication, Speech, and Vision*, vol. 2, no. 4, pp. 205–212, Aug. 1990.

[60] M. Flierl and B. Girod, "Generalized B pictures and the draft H.264/AVC video-compression standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 587–597, July 2003.

[61] T. Suzuki, K. Sato, and Y. Yagasaki, "Study of direct mode," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-E071*, Oct. 2002.

[62] K. K. Pang and T. Tan, "Optimum loop filter in hybrid coders," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 2, pp. 158–167, Apr. 1994.

[63] Y. Lee and H. Park, "Loop filtering and post-filtering for low-bit-rates moving picture coding," *Signal Processing: Image Communications*, vol. 16, no. 9, pp. 871–890, June 2001.

[64] *Generic Coding of Moving Pictures and Associated Audio Information - Part 1: Systems*, ITU-T and ISO/IEC JTC 1 Std. ITU-T Recommendation H.262 and ISO/IEC 13 818-1 (MPEG-2), 1994.

[65] S. Wenger, "An overview of the H.26L NAL concept," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-B028*, Feb. 2002.

[66] A. S. MacInnis, "Start codes and mapping to MPEG-2 systems," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-B049*, Feb. 2002.

[67] *Amendment 3: Transport of AVC video data over ITU-T Rec H.222.0 —ISO/IEC 13818-1 streams*, ITU-T and ISO/IEC JTC 1 Std. ISO/IEC 13 818-1, 2003.

[68] M. M. Hannuksela, "Sync pictures," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-D101*, July 2002.

[69] Y.-K. Wang, M. M. Hannuksela, and T. Walker, "On redundant slices and NAL decoding order," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-E130*, Oct. 2002.

[70] Y.-K. Wang, M. M. Hannuksela, and M. Gabbouj, "Video coding using unequally protected key pictures," *International Workshop on Very Low Bitrate Video 2003 (VLB03)*, Sept. 2003.

[71] T. McMahon, T. Wiegand, and G. Sullivan, "Draft professional extention amendment," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-I047*, July 2003.

[72] C. T. Pankaj Topiwala, "High sample depth, high resolution, and color coding issues," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-H023*, May 2003.

[73] C. Gomila and A. Kobilansky, "SEI message for film grain encoding," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-H022*, May 2003.

[74] C. Gomila, "SEI message for film grain encoding: syntax and results," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-I013*, July 2003.

[75] A. M. Rohaly, J. Libert, P. Corriveau, and A. Webster, "Final report from the video quality experts group on the validation of objective models of video quality assessment," *ITU-T Q.6/SG16 (VCEG) Document*, June 2000.

[76] *Test model 5*, ISO/IEC JTC/SC29/WG11 Std. MPEG 93/457, document AVC-491, Apr. 1993.

[77] S. Ma, Z. Li, and F. Wu, "Proposed draft of adaptive rate control," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-H017*, May 2003.

[78] Z. Li, W. Gao, F. Pan, S. Ma, K. P. Lim, G. Feng, X. Lin, S. Rahardja, H. Lu, and Y. Lu, "Adaptive rate control with HRD consideration," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-H014*, May 2003.

[79] G. Sullivan, T. Wiegand, and K.-P. Lim, "Joint model reference encoding methods and decoding concealment methods," *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Doc. JVT-I049*, July 2003.

[80] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, pp. 74–90, Nov. 1998.

[81] Z. Wang, L. Lu, and A. C. Bovik, "Video quality assessment based on structural distortion measurement," *Signal Processing: Image Communication*, vol. 19, pp. 121–132, Feb. 2004.

[82] Sveriges television. `ftp://ftp.ldv.e-technik.tu-muenchen.de/pub/test_sequences/720p`