Video Compression—From Concepts to the H.264/AVC Standard

GARY J. SULLIVAN, SENIOR MEMBER, IEEE AND THOMAS WIEGAND

Invited Paper

Over the last one and a half decades, digital video compression technologies have become an integral part of the way we create, communicate, and consume visual information. In this paper, techniques for video compression are reviewed, starting from basic concepts. The rate-distortion performance of modern video compression schemes is the result of an interaction between motion representation techniques, intra-picture prediction techniques, waveform coding of differences, and waveform coding of various refreshed regions. The paper starts with an explanation of the basic concepts of video codec design and then explains how these various features have been integrated into international standards, up to and including the most recent such standard, known as H.264/AVC.

Keywords*—*Advanced Video Coding (AVC), compression, H.264, H.26x, Joint Video Team (JVT), Moving Picture Experts Group (MPEG), MPEG-4, standards, video, video coding, video compression, Video Coding Experts Group (VCEG).

I. INTRODUCTION

Digital video communication can be found today in many application scenarios, such as:

- broadcast, subscription, and pay-per-view services over satellite, cable, and terrestrial transmission channels (e.g., using H.222.0 | MPEG-2 systems [1]);
- wire-line and wireless real-time conversational services (e.g., using H.32x [2]–[4] or Session Initiation Protocol (SIP) [5]);
- Internet or local area network (LAN) video streaming (using Real-Time Protocol/Internet Protocol (RTP/IP) [6]);
- storage formats (e.g., digital versatile disk (DVD), digital camcorders, and personal video recorders).

The basic communication problem may be posed as conveying source data with the highest fidelity possible within an available bit rate, or it may be posed as conveying the source data using the lowest bit rate possible while maintaining a specified reproduction fidelity [7]. In either case, a fundamental tradeoff is made between bit rate and fidelity. The ability of a source coding system to make this tradeoff well is called its *coding efficiency* or *rate-distortion performance*, and the coding system itself is referred to as a *codec* (i.e., a system comprising a *cod*er and a *dec*oder).

Thus, video codecs are primarily characterized in terms of:

- *Throughput of the channel*: a characteristic influenced by the transmission channel bit rate and the amount of protocol and error-correction coding overhead incurred by the transmission system.
- *Distortion of the decoded video*: distortion is primarily induced by the video codec and by channel errors introduced in the path to the video decoder.

However, in practical video transmission systems the following additional issues must be considered as well.

- *Delay (startup latency and end-to-end delay)*: Delay characteristics are influenced by many parameters, including processing delay, buffering, structural delays of video and channel codecs, and the speed at which data are conveyed through the transmission channel.
- *Complexity (in terms of computation, memory capacity, and memory access requirements)*. The complexity of the video codec, protocol stacks, and network.

Hence, the practical source coding design problem is posed as follows: given a maximum allowed delay and a maximum allowed complexity, achieve an optimal tradeoff between bit rate and distortion for the range of network environments envisioned in the scope of the applications.

The various application scenarios of video communication show very different optimum working points, and these working points have shifted over time as the constraints on complexity have been eased by Moore's law and as higher bit-rate channels have become available. In this paper, we examine the video codec design problem and the evolution of its solutions up to the latest international standard known as H.264 or MPEG-4 Advanced Video Coding (H.264/AVC).

Manuscript received February 19, 2004; revised July 30, 2004.

G. J. Sullivan is with the Microsoft Corporation, Redmond, WA 98052 USA (e-mail: garys@ieee.org).

T. Wiegand is with the Heinrich Hertz Institute (FhG), Berlin D 10587, Germany (e-mail: wiegand@hhi.de).

Digital Object Identifier 10.1109/JPROC.2004.839617

II. Video Source Coding Basics

A digital image or a *frame* of digital video typically consists of three rectangular arrays of integer-valued samples, one array for each of the three components of a tristimulus color representation for the spatial area represented in the image. Video coding often uses a color representation having three components called Y, Cb, and Cr. Component Y is called *luma* and represents brightness. The two *chroma* components Cb and Cr represent the extent to which the color deviates from gray toward blue and red, respectively.[1] Because the human visual system is more sensitive to luma than chroma, often a sampling structure is used in which the chroma component arrays each have only one-fourth as many samples as the corresponding luma component array (half the number of samples in both the horizontal and vertical dimensions). This is called 4:2:0 sampling. The amplitude of each component is typically represented with 8 b of precision per sample for consumer-quality video.

The two basic video formats are *progressive* and *interlaced*. A frame array of video samples can be considered to contain two interleaved *fields*, a *top* field and a *bottom* field. The top field contains the even-numbered rows $0, 2, \ldots, H-2$ (with 0 being top row number for a frame and H being its total number of rows), and the bottom field contains the odd-numbered rows $1, 3, \ldots, H-1$ (starting with the second row of the frame). When interlacing is used, rather than capturing the entire frame at each sampling time, only one of the two fields is captured. Thus, two sampling periods are required to capture each full frame of video. We will use the term picture to refer to either a frame or field. If the two fields of a frame are captured at different time instants, the frame is referred to as an interlaced frame, and otherwise it is referred to as a progressive frame.

Techniques for digital compression for many applications can typically be classified as follows.

- *Prediction:* A process by which a set of prediction values is created (often based in part on an indication sent by an encoder of how to form the prediction based on analysis of the input samples and the types of prediction that can be selected in the system design) that is used to predict the values of the input samples so that the values that need to be represented become only the (typically easier to encode) differences from the predicted values, such differences being called the residual values.
- *Transformation:* A process (also referred to as subband decomposition) that is closely related to prediction, consisting of forming a new set of samples from a combination of input samples, often using a linear combination. Simplistically speaking, a transformation can prevent the need to repeatedly represent similar values and can capture the essence of the input signal by using frequency analysis. A typical benefit of transformation is a reduction in the statistical correlation of the input samples, so that the most relevant aspects of the set of input samples are typically concentrated into a small number of variables. Two well-known examples of transformation are the Karhunen-Loève transform (KLT), which is an optimal decorrelator, and the discrete cosine transform (DCT), which has performance close to that of a KLT when applied to highly correlated auto-regressive sources.
- *Quantization:* A process by which the precision used for the representation of a sample value (or a group of sample values) is reduced in order to reduce the amount of data needed to encode the representation. Such a process is directly analogous to intuitively well-understood concepts such as the rounding off of less significant digits when writing the value of some statistic. Often the rounding precision is controlled by a step size that specifies the smallest representable value increment. Among the techniques listed here for compression, quantization is typically the only one that is inherently noninvertible—that is, quantization involves some form of many-to-few mapping that inherently involves some loss of fidelity. The challenge is to minimize that loss of fidelity in relation to some relevant method of measuring distortion.
- *Entropy coding:* A process by which discrete-valued source symbols are represented in a manner that takes advantage of the relative probabilities of the various possible values of each source symbol. A well-known type of entropy code is the variable-length code (VLC), which involves establishing a tree-structured code table that uses short binary strings to represent symbol values that are highly likely to occur and longer binary strings to represent less likely symbol values. The best-known method of designing VLCs is the well-known Huffman code method, which produces an optimal VLC. A somewhat less well-known method of entropy coding that can typically be more optimal than VLC coding and can also be more easily designed to adapt to varying symbol statistics is the newer technique referred to as arithmetic coding.

One way of compressing video is simply to compress each picture separately. This is how much of the compression research started in the mid-1960s [9], [10]. Today, the most prevalent syntax for such use is JPEG (-1992) [11]. The most common "baseline" JPEG scheme consists of segmenting the picture arrays into equal-size blocks of 8×8 samples each. These blocks are transformed by a DCT [12], and the DCT coefficients are then quantized and transmitted using variable-length codes. We refer to this kind of coding scheme as *intra-picture* or *Intra* coding, since the picture is coded without referring to other pictures in a video sequence. In fact, such Intra coding (often called motion JPEG) is in common use for video coding today in production-quality editing systems.

However, improved compression performance can be achieved by taking advantage of the large amount of temporal redundancy in video content. This was recognized at

[1]The terms luma and chroma are used here (and in H.264/AVC) rather than the terms luminance and chrominance, in order to avoid the implication of the use of linear light transfer characteristics that is often associated with the other terms.

least as long ago as 1929 [13]. Usually, much of the depicted scene is essentially just repeated in picture after picture without any significant change, so video can be represented more efficiently by sending only the changes in the video scene rather than coding all regions repeatedly. We refer to such techniques as *inter-picture* or *Inter* coding. This ability to use temporal redundancy to improve coding efficiency is what fundamentally distinguishes video compression from the Intra compression exemplified by JPEG standards. A historical analysis of video coding can be found in [14].

A simple method of improving compression by coding only the changes in a video scene is called conditional replenishment (CR) [15], and it was the only temporal redundancy reduction method used in the first version of the first digital video coding international standard, ITU-T Recommendation H.120 [16]. CR coding consists of sending signals to indicate which areas of a picture can just be repeated, and sending new information to replace the changed areas. Thus, CR allows a choice between one of two modes of representation for each area, which we call *Skip* and *Intra*. However, CR has a significant shortcoming, which is its inability to refine the approximation given by a repetition.

Often the content of an area of a prior picture can be a good starting approximation for the corresponding area in a new picture, but this approximation could benefit from some minor alteration to make it a better representation. Adding a third type of "prediction mode," in which a refinement difference approximation can be sent, results in a further improvement of compression performance—leading to the basic design of modern *hybrid* codecs (using a term coined by Habibi [17] with a somewhat different original meaning). The naming of these codecs refers to their construction as a hybrid of two redundancy reduction techniques—using both prediction and transformation. In modern hybrid codecs, regions can be predicted using inter-picture prediction, and a spatial frequency transform is applied to the refinement regions and the Intra-coded regions. The modern basic structure was first standardized in ITU-T Recommendation H.261 [18], and is used very similarly in its successors MPEG-1 [19], H.262 | MPEG-2 [20], H.263 [21], MPEG-4 Part 2 Visual [22], and H.264/AVC [23].

One concept for the exploitation of statistical temporal dependencies that was missing in the first version of H.120 [16] and in [17] was motion-compensated prediction (MCP). MCP dates to the early 1970s [24], and the way it is used in modern video coding standards [18]–[23] was first widely published in [25]. MCP can be motivated as follows. Most changes in video content are typically due to the motion of objects in the depicted scene relative to the imaging plane, and a small amount of motion can result in large differences in the values of the samples in a picture, especially near the edges of objects. Often, predicting an area of the current picture from a region of the previous picture that is displaced by a few samples in spatial location can significantly reduce the need for a refining difference approximation. This use of spatial displacement *motion vectors* (MVs) to form a prediction is known as *motion compensation* (MC), and the encoder's search for the best MVs to use is known as *motion estimation* (ME). The coding of the resulting difference signal for the refinement of the MCP is known as MCP residual coding.

It should be noted that the subsequent improvement of MCP techniques has been the major reason for coding efficiency improvements achieved by modern standards when comparing them from generation to generation. The price for the use of MCP in ever more sophisticated ways is a major increase in complexity requirements. The primary steps forward in MCP that found their way into the H.264/AVC standard were as follows.

- *Fractional-sample-accurate MCP* [26]. This term refers to the use of spatial displacement MV values that have more than integer precision, thus requiring the use of interpolation when performing MCP. A theoretical motivation for this can be found in [27] and [28]. Intuitive reasons include having a more accurate motion representation and greater flexibility in prediction filtering (as full-sample, half-sample, and quarter-sample interpolators provide different degrees of low-pass filtering which are chosen automatically in the ME process). Half-sample-accuracy MCP was considered even during the design of H.261 but was not included due to the complexity limits of the time. Later, as processing power increased and algorithm designs improved, video codec standards increased the precision of MV support from full-sample to half-sample (in MPEG-1, MPEG-2, and H.263) to quarter-sample (for luma in MPEG-4's advanced simple profile and H.264/AVC) and beyond (with eighth-sample accuracy used for chroma in H.264/AVC).
- *MVs over picture boundaries* [29], first standardized in H.263. The approach solves the problem for motion representation for samples at the boundary of a picture by extrapolating the reference picture. The most common method is just to replicate the boundary samples for extrapolation.
- *Bipredictive MCP* [30], i.e., the averaging of two MCP signals. One prediction signal has typically been formed from a picture in the temporal future with the other formed from the past relative to the picture being predicted (hence, it has often been called bidirectional MCP). Bipredictive MCP was first put in a standard in MPEG-1, and it has been present in all other succeeding standards. Intuitively, such bipredictive MCP particularly helps when the scene contains uncovered regions or smooth and consistent motion.
- *Variable block size MCP* [31], i.e., the ability to select the size of the region (ordinarily a rectangular block-shaped region) associated with each MV for MCP. Intuitively, this provides the ability to effectively trade off the accuracy of the motion field representation with the number of bits needed for representing MVs [41].
- *Multipicture MCP* [36], [37], i.e., MCP using more than just one or two previous decoded pictures. This allows the exploitation of long-term statistical dependencies in video sequences, as found with backgrounds, scene cuts, and sampling aliasing.

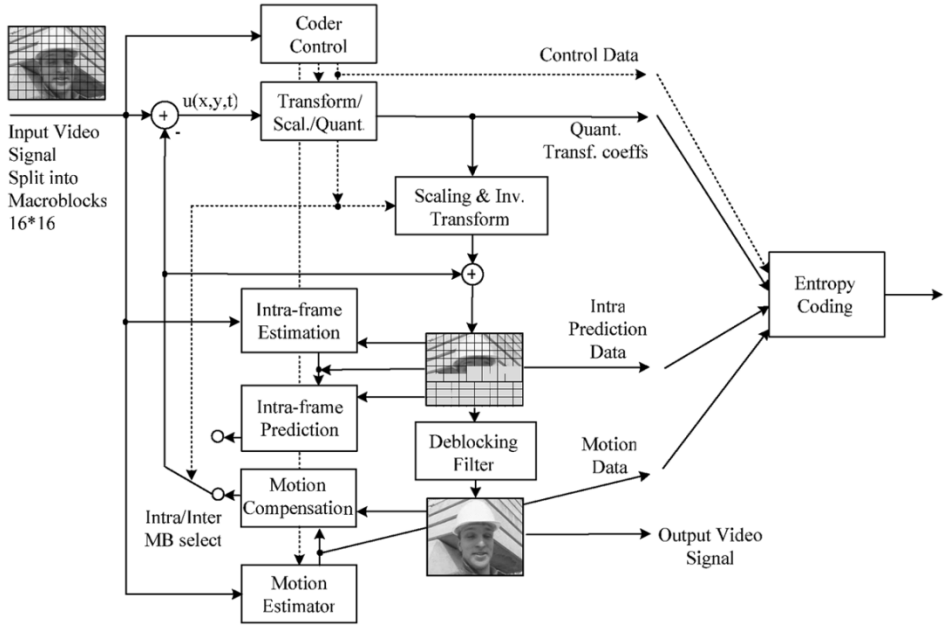

Fig. 1. Hybrid video encoder (especially for H.264/AVC).

- *Multihypothesis and weighted MCP* [32]–[35], i.e., the concept of linearly superimposed MCP signals. This can be exploited in various ways, such as overlapped block MC as in [32] and [33] (which is in H.263 but not H.264/AVC) and conventional bidirectional MCP. The combination of bidirectional MCP, multipicture MCP, and linearly weighted MCP can lead to a unified generalization [34] as found in H.264/AVC. Even the interpolation process of fractional-sample-accurate MCP is a special case of multihypothesis MCP, as it uses a linear superposition of MCP signals from multiple integer MV offsets.

Natural video contains a wide variety of content with different statistical behavior, even from region to region within the same picture. Therefore, a consistent strategy for improving coding efficiency has been to add coding modes to locally adapt the processing for each individual part of each picture. Fig. 1 shows an example encoder for modern video coding standards [18]–[23].

In summary, a hybrid video encoding algorithm typically proceeds as follows. Each picture is split into blocks. The first picture of a video sequence (or for a "clean" random access point into a video sequence) is typically coded in Intra mode (which typically uses some prediction from region to region within the picture but has no dependence on other pictures). For all remaining pictures of a sequence or between random access points, typically inter-picture coding modes are used for most blocks. The encoding process for Inter prediction (ME) consists of choosing motion data comprising the selected reference picture and MV to be applied for all samples of each block. The motion and mode decision data, which are transmitted as side information, are used by the encoder and decoder to generate identical Inter prediction signals using MC.

The residual of the Intra or Inter prediction, which is the difference between the original block and its prediction, is transformed by a frequency transform. The transform coefficients are then scaled, quantized, entropy coded, and transmitted together with the prediction side information.

The encoder duplicates the decoder processing so that both will generate identical predictions for subsequent data. Therefore, the quantized transform coefficients are constructed by inverse scaling and are then inverse transformed to duplicate the decoded prediction residual. The residual is then added to the prediction, and the result of that addition may then be fed into a deblocking filter to smooth out block-edge discontinuities induced by the block-wise processing. The final picture (which is also displayed by the decoder) is then stored for the prediction of subsequent encoded pictures. In general, the order of the encoding or decoding processing of pictures often differs from the order in which they arrive from the source, necessitating a distinction between the *decoding order* and the *output order* for a decoder.

The design and operation of an encoder involves the optimization of many decisions to achieve the best possible tradeoff between rate and distortion given the constraints on delay and complexity. There has been a large amount of work on this optimization problem. One particular focus has been on Lagrangian optimization methods [38]–[40]. Some studies have developed advanced encoder optimization strategies with little regard for encoding complexity (e.g., [41]–[51]), while others have focused on how to achieve a reduction in complexity while losing as little as possible in rate-distortion performance.

Above we have described the major technical features of a modern video coder. An example of the effectiveness of these features and the dependence of this effectiveness on video content is shown in Fig. 2. The plot shows performance for a sequence known as *Foreman*, with heavy object motion and an unstable hand-held moving camera. The sequence was encoded in *common intermediate format* (CIF) resolution (352×288 in luma with 4:2:0 sampling) at 15 frames/s, using well-optimized H.263 and MPEG-4 part

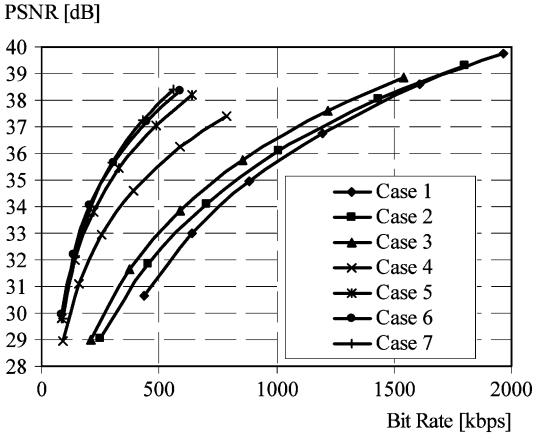

Fig. 2. Effectiveness of basic technical features.

2 video encoders (using optimization methods described in [51]). H.263 and MPEG-4 part 2 use 8×8 DCT-based residual coding and (as with all other standards starting with H.261) 16×16 prediction mode regions called *macroblocks*.

Gains in performance can be seen in Fig. 2 when adding various enhanced Inter coding modes to the encoder.

- Case 1) The performance achieved by spatial-transform Intra coding only (e.g., as in JPEG coding).
- Case 2) Adding Skip mode to form a CR coder.
- Case 3) Adding residual difference coding, but with only zero-valued MVs.
- Case 4) Adding integer-precision MC with blocks of size 16×16 luma samples.
- Case 5) Adding half-sample-precision MC.
- Case 6) Allowing some 16×16 regions to be split into four blocks of 8×8 luma samples each for MC.
- Case 7) Increasing MV precision to quarter-sample.

The addition of more and more such cases must be done carefully, or the complexity of selecting among them and the amount of coded data necessary indicate that selection could exceed the benefit of having more choices available. The amount of benefit for each technique will also vary dramatically for different video scene content.

III. Video Transmission Over Error-Prone Channels

In many cases, the errors of transmission channels can be efficiently corrected by classical channel coding methods such as forward error correction (FEC) and automatic repeat request (ARQ) or a mixture of them. This is achieved at the cost of reduced throughput and increased delay. Applications that typically fall into this category are broadcast, streaming, and video mail, and most of the problems related to error-prone transmission channels do not affect the design of video codecs for these applications.

However, these channel coding techniques sometimes require too much of a reduction in data throughput from the transmission channel and add too much delay to provide a negligible bit-error and packet-loss rate for some applications. Examples are video conferencing with its demanding delay requirements, slow fading mobile channels, congested Internet routers, and broadcast with varying coverage. Therefore, some amount of data losses or residual errors must often be tolerated.

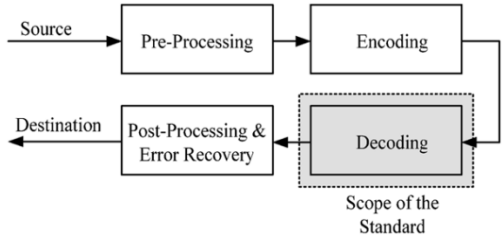

Fig. 3. Scope of video coding standardization.

However, when MCP is used in a hybrid video codec, data losses can cause the reference pictures stored at the encoder and decoder to differ in that the encoder's reference storage contains the transmitted video pictures while the decoder's reference storage contains corrupted or concealed content for the parts of the pictures that are affected by the errors. MCP can then cause the error to propagate to many subsequently decoded pictures. Because errors remain visible for much longer than a single picture display period, the resulting artifacts are particularly annoying to viewers. Quick recovery can only be achieved when picture regions are encoded in Intra mode or when Inter prediction is modified to ensure that no reference is made to the parts of the reference pictures that differ.

The bitstream and its transport layer must provide frequent access points at which a decoder can restart its decoding process after some loss or corruption, and it can also be beneficial to separate more important data (such as header information, prediction modes, MVs, and Intra data) from less important data (such as the fine details of the Inter prediction residual representation) in the bitstream so that the more important data can still be decoded when some of the less important data has been lost. Providing greater protection against losses of the more important parts of the data can also be beneficial.

Work in this area often focuses on modifying syntax and encoder operation to minimize error propagation, or improving the decoder's ability to conceal errors. Recently, there has also been some work on changing the basic structure of a low-delay video codec to using distributed coding (reviewed in [52]). Approaches to modify encoder operation either concentrate on the use of Intra coding (e.g., [53]–[57]) or modify MCP in Inter coding (e.g., [58]–[63]) or both (e.g., [37], [64]). Methods to improve error concealment at the decoder have included approaches with and without dedicated side information (e.g., see [65]–[68]).

IV. Video Coding Standards

A typical video processing chain (excluding the transport or storage of the video signal) and the scope of the video coding standardization are depicted in Fig. 3. For all ITU-T and ISO/IEC JTC 1 video coding standards, only the central decoder is standardized. The standard defines a specific bitstream syntax, imposes very limited constraints on the values of that syntax, and defines a limited-scope decoding process. The intent is for every decoder that conforms to the standard to produce similar output when given a bitstream that conforms to the specified constraints. Thus, these video coding standards are written primarily only to ensure interoperability (and syntax capability), not to ensure quality. This

limitation of scope permits maximal freedom to optimize the design of each specific product (balancing compression quality, implementation cost, time to market, etc.). It provides no guarantees of end-to-end reproduction quality, as it allows even crude encoding methods to be considered in conformance with the standard.

V. H.264/AVC Video Coding Standard

To address the requirement of flexibility and customizability to various applications, the H.264/AVC [23], [69] design covers a video coding layer (VCL), which is designed to efficiently represent the video content, and a network abstraction layer (NAL), which formats the VCL representation of the video and provides header information to package that data for network transport.

A. H.264/AVC NAL

The NAL is designed to enable simple and effective customization of the use of the VCL for a broad variety of systems. The full degree of customization of the video content to fit the needs of each particular application is outside the scope of the H.264/AVC standard itself, but the design of the NAL anticipates a variety of such mappings.

Some key building blocks of the NAL design are NAL units, parameter sets, and access units. A short description of these concepts is given below, with more detail including error resilience aspects provided in [70] and [71].

1) NAL Units: The coded video data is organized into NAL units, each of which is effectively a packet that contains an integer number of bytes. The first byte of each NAL unit is a header byte that contains an indication of the type of data in the NAL unit, and the remaining bytes contain payload data of the type indicated by the header.

Some systems (e.g., H.320 and H.222.0 | MPEG-2 systems) require delivery of the entire or partial stream of NAL units as an ordered stream of bytes or bits. For use in such systems, H.264/AVC specifies a byte stream format, where each NAL unit is prefixed by a specific pattern of three bytes called a start code prefix which can be uniquely identified in the byte stream. A finite-state machine prevents accidental emulation of start code prefixes. In other systems (e.g., RTP/IP systems), the coded data is carried in packets that are framed by the system transport protocol, and identification of the boundaries of NAL units within the transport packets can be established without use of start code prefix patterns.

There are two classes of NAL units, called VCL and non-VCL NAL units. The VCL NAL units contain the data that represents the values of the samples in the video pictures, and the non-VCL NAL units contain all other related information such as parameter sets (important header data that can apply to a large number of VCL NAL units) and supplemental enhancement information (timing information and other supplemental data that may enhance usability of the decoded video signal but are not necessary for decoding the values of the samples in the video pictures).

2) Parameter Sets: A parameter set contains important header information that can apply to a large number of VCL NAL units. There are two types of parameter sets:

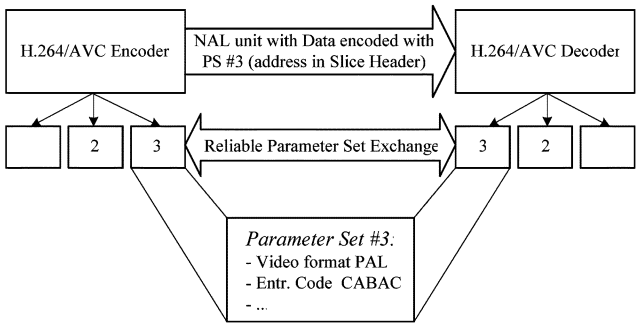

Fig. 4. Parameter set use with reliable "out-of-band" parameter set exchange.

- sequence parameter sets, which apply to a series of consecutive coded video pictures;
- picture parameter sets, which apply to the decoding of one or more individual pictures.

Key VCL NAL units for a picture each contain an identifier that refers to the content of the relevant picture parameter set, and each picture parameter set contains an identifier that refers to the relevant sequence parameter set. In this manner, a small amount of data (the identifier) can be used to establish a larger amount of information (the parameter set) without repeating that information within each VCL NAL unit.

The sequence and picture parameter set mechanism decouples the transmission of infrequently changing information from the transmission of coded representations of the values of the samples in the video pictures. This design for extra robustness for parameter sets is especially important, as the loss of certain syntax elements can have a catastrophic impact on the ability to decode the video.

Sequence and picture parameter sets can be sent well ahead of the VCL NAL units that they apply to, and can be repeated to provide robustness against data loss. In some applications, parameter sets may be sent within the channel that carries the VCL NAL units (termed "in-band" transmission). In other applications (see Fig. 4) it can be advantageous to convey the parameter sets "out-of-band" using a more reliable transport mechanism.

3) Access Units: The set of VCL and non-VCL NAL units that is associated with a single decoded picture is referred to as an access unit. The access unit contains all macroblocks of the picture, possibly some redundant approximations of some parts of the picture for error resilience purposes (referred to as *redundant slices*), and other supplemental information associated with the picture.

B. H.264/AVC VCL

As in all prior ITU-T and ISO/IEC JTC 1 video standards since H.261 [18], the VCL design follows the so-called block-based hybrid video coding approach (as depicted in Fig. 1). There is no single coding element in the VCL that provides the majority of the significant improvement in compression efficiency in relation to prior video coding standards. It is rather a plurality of smaller improvements that add up to the significant gain. A more detailed description of the VCL design is given below.

1) Macroblocks, Slices, and Slice Groups: A coded video sequence in H.264/AVC consists of a sequence of coded pictures. Each picture is partitioned into fixed size

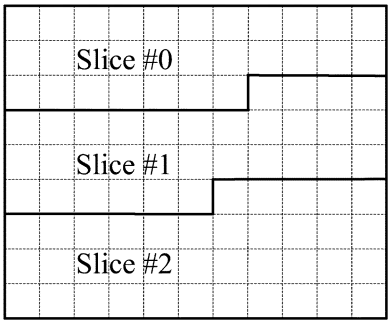

Fig. 5. Subdivision of a picture into slices (when not using FMO).

macroblocks that each contain a rectangular picture area of 16×16 samples for the luma component and the corresponding 8×8 sample regions for each of the two chroma components. Macroblocks are the basic building blocks for which the decoding process is specified. The luma and chroma samples of a macroblock are predicted—either spatially or temporally—and the resulting prediction residual is transmitted using transform coding. Each color component of the residual is subdivided into blocks, each block is transformed using an integer transform, and the transform coefficients are quantized and entropy coded.

The macroblocks of the picture are organized into *slices*, which represent regions of a given picture that can be decoded independently. Each slice is a sequence of macroblocks that is processed in the order of a raster scan, i.e., a scan from top-left to bottom-right, (although they are not necessarily always consecutive in the raster scan, as described below for the flexible macroblock ordering (FMO) feature). A picture may contain one or more slices (for example, as shown in Fig. 5). Each slice is self-contained, in the sense that, given the active sequence and picture parameter sets, its syntax elements can be parsed from the bitstream and the values of the samples in the area of the picture that the slice represents can basically be decoded without use of data from other slices of the picture (provided that all previously decoded reference pictures are identical at encoder and decoder for use in MCP). However, for completely exact decoding, some information from other slices may be needed in order to apply the deblocking filter across slice boundaries. Slices can be used for:

- error resilience, as the partitioning of the picture allows spatial concealment within the picture and as the start of each slice provides a resynchronization point at which the decoding process can be reinitialized;
- creating well-segmented payloads for packets that fit the maximum transfer unit (MTU) size of a network (e.g., MTU size is 1500 B for Ethernet);
- parallel processing, as each slice can be encoded and decoded independently of the other slices of the picture.

The error resilience aspect of slices can be further enhanced (among other uses) through the use of the FMO technique, which modifies the way macroblocks are associated with slices. Using FMO, a picture can be split into many macroblock scanning patterns such as interleaved slices, a dispersed macroblock allocation, one or more "foreground" slice groups and a "leftover" slice group, or a checkerboard mapping. For more details on the use of FMO, see [70]; concealment techniques for FMO are exemplified in [71].

Since each slice of a picture can be decoded independently of the others, no specific ordering of the decoding for the various slices of a picture is strictly necessary. This gives rise to a concept closely related to FMO that can be used for loss robustness and delay reduction, which is *arbitrary slice ordering* (ASO). When ASO is in use, the slices of a picture can be in any relative order in the bitstream, and when it is not, the slices must be ordered such that the first macroblock in each subsequent slice is increasing in the order of a raster scan within the picture.

Loss robustness can also be enhanced by separating more important data (such as macroblock types and MV values) from less important data (such as inter residual transform coefficient values) and reflecting data dependencies and importance by using separate NAL unit packets for data of different categories. This is referred to as *data partitioning*.

Further loss robustness can be provided by sending duplicative coded representations of some or all parts of the picture. These are referred to as *redundant slices*.

2) Slice Types: There are five fundamental slice types.

- **I slice:** A slice in which all macroblocks of the slice are coded using Intra prediction.
- **P slice:** In addition to the coding types of the I slice, macroblocks of a P slice can also be coded using Inter prediction with at most *one* MCP signal per block.
- **B slice:** In addition to the coding types available in a P slice, macroblocks of a B slice can also be coded using Inter prediction with *two* MCP signals per prediction block that are combined using a weighted average.
- **SP slice:** A so-called switching P slice that is coded such that efficient and exact switching between different video streams (or efficient jumping from place to place within a single stream) becomes possible without the large number of bits needed for an I slice.
- **SI slice:** A so-called switching I slice that allows an exact match with an SP slice for random access or error recovery purposes, while using only Intra prediction.

The first three slice types listed above are very similar to coding methods used in previous standards, with the exception of the use of reference pictures as described below. The other two types are new. For details on the novel concept of SP and SI slices, the reader is referred to [72]; the other slice types are further described below.

3) Intra-Picture Prediction: In all slice-coding types, two primary types of Intra coding are supported: Intra_4×4 and Intra_16×16 prediction. Chroma Intra prediction is the same in both cases. A third type of Intra coding, called I_PCM, is also provided for use in unusual situations.

The Intra_4×4 mode is based on predicting each 4×4 luma block separately and is well suited for coding of parts of a picture with significant detail. The Intra_16×16 mode, on the other hand, does prediction and residual coding on the entire 16×16 luma block and is more suited for coding very smooth areas of a picture. In addition to these two types of luma prediction, a separate chroma prediction is conducted.

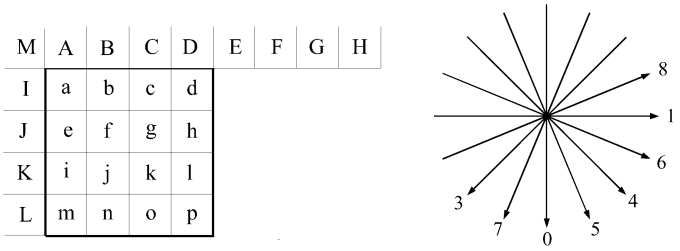

Fig. 6. Left: Intra_4×4 prediction is conducted for samples a-p using samples A-M. Right: Eight selectable "prediction directions" for Intra_4×4.

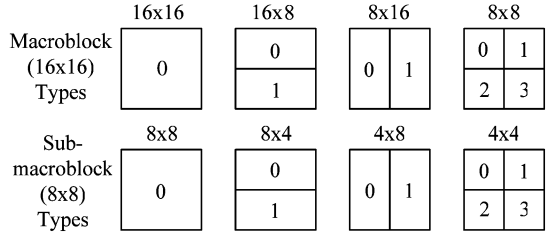

Fig. 7. Segmentations of the macroblock for MC. Top: segmentation of macroblocks, bottom: segmentation of 8×8 partitions.

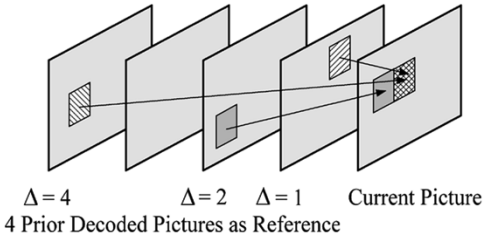

Fig. 8. Multipicture MCP. In addition to the MV, reference indexes (Δ) are transmitted. The concept is similarly extended for B slices.

In contrast to previous video coding standards (especially H.263+ and MPEG-4 Visual), where Intra prediction has been conducted in the transform domain, Intra prediction in H.264/AVC is always conducted in the spatial domain, by referring to neighboring samples of previously decoded blocks that are to the left and/or above the block to be predicted. Since this can result in spatio-temporal error propagation when Inter prediction has been used for neighboring macroblocks, a constrained Intra coding mode can alternatively be selected that allows prediction only from Intra-coded neighboring macroblocks.

In Intra_4×4 mode, each 4×4 luma block is predicted from spatially neighboring samples as illustrated on the left-hand side of Fig. 6. The 16 samples of the 4×4 block, marked a–p, are predicted using position-specific linear combinations of previously decoded samples, marked A–M, from adjacent blocks. The encoder can select either "DC" prediction (called mode 2, where an average value is used to predict the entire block) or one of eight directional prediction types illustrated on the right-hand side of Fig. 6. The directional modes are designed to model object edges at various angles.

In Intra_16×16 mode, the whole 16×16 luma component of the macroblock is predicted at once, and only four prediction modes are supported: vertical, horizontal, DC, and plane. The first three are similar to the modes in Intra_4×4 prediction except for increasing the number of samples to reflect the larger block size. Plane prediction uses position-specific linear combinations that effectively model the predicted block as a plane with an approximate fit for the horizontal and vertical variation along the block edges.

The chroma samples of an Intra macroblock are predicted using similar prediction techniques as for the luma component in Intra_16×16 macroblocks.

For the I_PCM Intra macroblock type, no prediction is performed and the raw values of the samples are simply sent without compression. This mode is primarily included for decoder implementation reasons, as it ensures that the number of bits needed for any macroblock will never need to be much larger than the size of an uncompressed macroblock, regardless of the quantization step size and the values of the particular macroblock samples. As a side benefit, it also enables lossless coding of selected regions.

4) Inter-Picture Prediction: Inter-Picture Prediction in P Slices: Various "predictive" or motion-compensated coding types are specified as P macroblock types. P macroblocks can be partitioned into smaller regions for MCP with luma block sizes of 16×16, 16×8, 8×16, and 8×8 samples. When 8×8 macroblock partitioning is chosen, an additional syntax element is transmitted for each 8×8 partition, which specifies whether the 8×8 partition is further partitioned into smaller regions of 8×4, 4×8, or 4×4 luma samples and corresponding chroma samples (see Fig. 7). The prediction signal for each predictive-coded $M \times N$ luma block is obtained by MC, which is specified by a translational MV and a picture reference index. The syntax allows MVs to point over picture boundaries.

The accuracy of MC is in units of one-quarter of the horizontal or vertical distance between luma samples. If the MV points to an integer-sample position, the prediction signal consists of the corresponding samples of the reference picture; otherwise, the corresponding sample is obtained using interpolation. The prediction values at half-sample positions are obtained by applying a one-dimensional six-tap finite impulse response (FIR) filter horizontally and/or vertically. Prediction values at quarter-sample positions are generated by averaging two samples at integer- and half-sample positions. For further analysis, refer to [73].

The MV values are differentially coded using either median or directional prediction from neighboring blocks. No MV value prediction (or any other form of prediction) takes place across slice boundaries.

The syntax supports multipicture MCP [36], [37]. That is, more than one previously decoded picture can be used as a reference for MCP. Fig. 8 illustrates the concept. Previously decoded pictures are stored in a decoded picture buffer (DPB) as directed by the encoder, and a DPB reference index is associated with each motion-compensated 16×16, 16×8, 8×16, or 8×8 luma block. MCP for smaller regions than 8×8 uses the same reference index for predicting all blocks in an 8×8 region.

A P macroblock can also be coded in the so-called P_Skip mode. For this coding mode, neither a quantized prediction error signal nor an MV with a reference index is sent. The reconstructed signal is obtained using only a prediction signal

like that of a P_16×16 macroblock that references the picture located at index 0 in the list (referred to as list 0) of pictures in the DPB. The MV used for reconstructing the P_Skip macroblock is similar to the MV predictor for the 16×16 block. The useful effect of this P_Skip coding type is that large areas with no change or constant motion (like slow panning) can be represented with very few bits.

Inter-Picture Prediction in B Slices: In comparison to prior video coding standards, the concept of B slices is generalized in H.264/AVC. This extension refers back to [32]-[34] and is further studied in [74]. For example, other pictures can use reference pictures containing B slices for MCP, depending on whether the encoder has selected to indicate that the B picture can be used for reference. Thus, the substantial difference between B and P slices is that B slices are coded in a manner in which some macroblocks or blocks may use a weighted average of two distinct MCP values for building the prediction signal. B slices use two distinct lists of reference pictures in the DPB, which are referred to as the first (list 0) and second (list 1) reference picture lists, respectively.

B slices use a similar macroblock partitioning as P slices. Beside the P_16×16, P_16×8, P_8×16, P_8×8, and the Intra coding types, bipredictive prediction and another type of prediction called Direct prediction are provided. For each 16×16, 16×8, 8×16, and 8×8 partition, the prediction method (list 0, list 1, bipredictive) can be chosen separately. An 8×8 partition of a B macroblock can also be coded in Direct mode. If no prediction error signal is transmitted for a Direct macroblock mode, it is also referred to as B_Skip mode and can be coded very efficiently, similar to the P_Skip mode in P slices. The MV coding is similar to that of P slices with the appropriate modifications because neighboring blocks may be coded using different prediction modes.

Weighted Prediction in P and B Slices: In previous standards, biprediction has typically been performed with a simple (1/2, 1/2) averaging of the two prediction signals, and the prediction in the so-called P macroblock types has not used weighting. However, in H.264/AVC, an encoder can specify scaling weights and offsets to be used for each prediction signal in the P and B macroblocks of a slice. The weighting and offset values can be inferred from temporally related relationships or can be specified explicitly. It is even allowed for different weights and offsets to be specified within the same slice for performing MCP using the same particular reference picture.

5) Transform, Scaling, and Quantization: Similar to previous video coding standards, H.264/AVC uses spatial transform coding of the prediction residual. However, in H.264/AVC, the transformation is applied to 4×4 blocks (instead of the larger 8×8 blocks used in previous standards), and instead of providing a theoretical inverse DCT formula to be approximated by each implementer within specified tolerances, a separable integer transform with similar properties to a 4×4 DCT is used. Its basic matrix is

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

The transform coding process is similar to that in previous standards, but since the inverse transform is defined by very simple exact integer operations, inverse-transform mismatches are avoided and decoding complexity is minimized. There are several reasons for using a smaller transform size (4×4) than was used in prior standards (8×8).

- One of the main improvements of the present standard is the improved prediction process both for Inter and Intra. Consequently, the residual signal has less spatial correlation. This generally means that the transform has less to offer concerning decorrelation, so a 4×4 transform is essentially as efficient.
- With similar objective compression capability, the smaller 4×4 transform has visual benefits resulting in less noise around edges (referred to as "mosquito noise" or "ringing" artifacts).
- The smaller transform requires less computation and a smaller processing word length.

For the luma component in the Intra_16×16 mode and for the chroma components in all Intra macroblocks, the DC coefficients of the 4×4 transform blocks undergo a second transform, with the result that the lowest-frequency transform basis functions cover the entire macroblock. This additional transform is 4×4 for the processing of the luma component in Intra_16×16 mode and is 2×2 for the processing of each chroma component in all Intra modes. Extending the length of the lowest-frequency basis functions by applying such a secondary transform tends to improve compression performance for very smooth regions.

A quantization parameter (QP) is used for determining the quantization of transform coefficients in H.264/AVC. It can take on 52 values. The quantization step size is controlled logarithmically by QP rather than linearly as in previous standards, in a manner designed to reduce decoding complexity and enhance bit rate control capability. Each increase of six in QP causes a doubling of the quantization step size, so each increase of one in QP increases the step size by approximately 12%. (Often a change of step size by approximately 12% also means roughly a reduction of bit rate by approximately 12%.)

The quantized transform coefficients of a block generally are scanned in a zigzag fashion and transmitted using entropy coding. The 2×2 DC coefficients of the chroma component are scanned in raster-scan order.

All inverse transform operations in H.264/AVC can be implemented using only additions, subtractions, and bit-shifting operations on 16-b integer values, and the scaling can be done using only 16 b as well. Similarly, only 16-b memory accesses are needed for a good implementation of the forward transform and quantization processes in the encoder. For more information, see [75].

6) Entropy Coding: In H.264/AVC, two alternatives for entropy coding are supported. These are called context-adaptive variable-length coding (CAVLC) and context-adaptive binary arithmetic coding (CABAC). CABAC has higher complexity than CAVLC, but has better coding efficiency.

In both of these modes, many syntax elements are coded using a single infinite-extent codeword set referred to as an Exp-Golomb code. Thus, instead of designing a different

VLC table for each syntax element, only the mapping to the single codeword table is customized to the data statistics. The Exp–Golomb code has a simple and regular structure.

When using CAVLC, the quantized transform coefficients are coded using VLC tables that are switched depending on the values of previous syntax elements. Since the VLC tables are context conditional, the coding efficiency is better than for schemes using a single VLC table, such as the simple "run+level" or "run+level+last" coding found in previous standards. More details can be found in [23] and [69].

The efficiency can be improved further using CABAC [76]. CABAC not only uses context-conditional probability estimates, but adjusts its probability estimates to adapt to nonstationary statistical behavior. Its arithmetic coding also enables the use of a noninteger number of bits to encode each symbol of the source alphabet (which can be especially beneficial when the source symbol probabilities are highly skewed). The arithmetic coding core engine and its associated probability estimation use low-complexity multiplication-free operations involving only shifts and table lookups. Compared to CAVLC, CABAC typically reduces the bit rate 10%–15% for the same quality.

7) In-Loop Deblocking Filter: One annoying characteristic of block-based coding is the production of visible block artifacts, especially at low bit rates. Block edges are typically predicted by MCP with less accuracy than interior samples, and block transforms also produce block edge discontinuities. Blocking is generally considered to be one of the most visible artifacts with the present compression methods. For this reason, H.264/AVC defines an adaptive in-loop deblocking filter. A detailed description of the deblocking filter can be found in [77].

The filter reduces blockiness while basically retaining the sharpness of the true edges in the scene. Consequently, the subjective quality is significantly improved. The filter typically reduces bit rate by 5%–10% for the same objective quality as the nonfiltered video, and improves subjective quality even more. Fig. 9 illustrates the visual effect.

8) Adaptive Frame/Field Coding Operation: Interlaced frames often show different statistical properties than progressive frames. H.264/AVC allows the following interlace-specific coding methods:

- frame mode: combine the two fields together as a frame and to code the entire frame as a picture;
- field mode: not combining the two fields and instead coding each single field as a separate picture;
- macroblock-adaptive frame/field mode (MBAFF): coding the entire frame as a picture, but enabling the selection of individual pairs of vertically adjacent macroblocks within the picture to be split into fields for prediction and residual coding.

The choice between the three options can be made adaptively for each frame in a sequence. Choosing just between the first two options is referred to as picture-adaptive frame/field (PAFF) coding. When a picture is a single field, each field is partitioned into macroblocks and is coded in a manner very similar to a frame, except MCP uses reference fields rather than reference frames, the zigzag scan for transform coefficients is different, and the strongest deblocking strength is not used for filtering across horizontal edges of macroblocks in fields, because the field rows are spatially twice as far apart as frame rows (effectively lengthening the filter).

For MBAFF coding, the frame/field encoding decision can also be made for each vertical pair of macroblocks in a frame (a 16×32 luma region). For a macroblock pair that is coded in frame mode, each macroblock contains lines from both fields. For a field mode macroblock pair, one macroblock contains top field lines and the other contains bottom field lines. Each macroblock of a field macroblock pair is processed in essentially the same way as a macroblock within a field in PAFF coding. Note that, unlike in MPEG-2, the MBAFF frame/field decision is made at a macroblock pair level rather than within the macroblock level. This keeps the basic macroblock processing structure the same for each prediction or residual coding operation, and permits field mode MCP block sizes as large as an entire macroblock.

During the development of the H.264/AVC standard, for key ITU-R BT.601 [8] resolution sequences chosen as representative for testing, PAFF coding was reported to reduce bit rates roughly 15%–20% over frame-only coding for sequences like "Canoa," "Rugby," etc.; and MBAFF coding was reported to reduce bit rates roughly 15% over PAFF for sequences like "Mobile & Calendar" and "News."

9) Hypothetical Reference Decoder: A key benefit provided by a standard is the assurance that all decoders that conform to the standard will be able to decode any conforming compressed video bitstream (given the appropriate profile and level capabilities as discussed below). To achieve that, it is not sufficient to just specify the syntax of the data and how to interpret it. It is also important to constrain how fast the bitstream data can be fed to a decoder and how much buffering of the bitstream and decoded pictures is required to build a decoder. Specifying input and output buffer models and developing an implementation-independent idealized model of a decoder achieves this. That receiver model is also called a *hypothetical reference decoder* (HRD) (see [78]).

The H.264/AVC HRD specifies operation of an idealized decoder with two buffers having specified capacity constraints: the coded picture buffer (CPB) and the DPB. The CPB models the arrival and removal timing of the coded bits and the DPB models the storage for decoded pictures. The HRD design is similar in spirit to what MPEG-2 had, but is more flexible for sending video at a variety of bit rates and without excessive delay, and it provides flexible DPB management for highly generalized multipicture buffering.

10) Profiles and Levels: Profiles and levels specify conformance points to facilitate interoperability for various applications. Ordinarily a *profile* defines a syntax that can be used in generating a conforming bitstream, whereas a *level* places constraints on the values of key parameters (such as maximum bit rate, buffering capacity, or picture resolution).

All decoders conforming to a specific profile must support all features in that profile. Encoders are not required to make use of any particular set of features supported in a profile but must provide conforming bitstreams, i.e., bitstreams that can

Fig. 9. Performance of the deblocking filter for highly compressed pictures. Top: without deblocking filter; bottom: with deblocking filter.

be decoded by conforming decoders. In H.264/AVC, three profiles are defined. These are the *Baseline, Main*, and *Extended* profiles.

The features of the H.264/AVC design can be segmented into the following five elemental sets.

- **Set 0** (basic features for efficiency, robustness, and flexibility): I and P slices, CAVLC, and other basics.
- **Set 1** (enhanced robustness/flexibility features): FMO, ASO, and redundant slices.
- **Set 2** (further enhanced robustness/flexibility features): SP/SI slices and slice data partitioning.
- **Set 3** (enhanced coding efficiency features) B slices, weighted prediction, field coding, and macroblock adaptive frame/field coding.
- **Set 4** (a further coding efficiency feature): CABAC.

The Baseline profile, which emphasizes coding efficiency and robustness with low computational complexity, supports the features of sets 0 and 2. The Main profile, which empha-

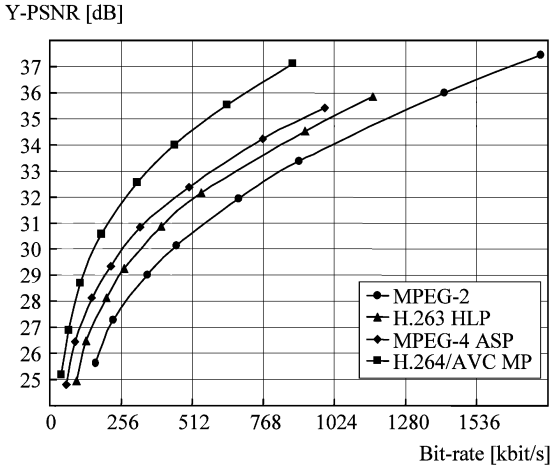

Fig. 10. PSNR-rate curves for the test sequence "Tempete" in video streaming applications.

sizes primarily coding efficiency alone, supports the features of sets 0, 3, and 4. The Extended profile, which emphasizes robustness and flexibility with high coding efficiency, supports the features of sets 0, 1, 2, and 3 (all features except CABAC).

Since the Main profile does not support the FMO, ASO, and redundant slice feature of set 1, some bitstreams that are decodable by a Baseline profile decoder are not decodable by a Main profile decoder. Similarly, because of noncommonality for sets 3 and 4, some bitstreams that are decodable by a Main profile decoder are not decodable by an Extended profile decoder and *vice versa*. To address this issue, flags in the sequence parameter set are used to indicate which profiles can decode each video sequence.

In H.264/AVC, the same set of levels is used with all profiles, and individual implementations may support a different level for each supported profile. Fifteen levels are defined, specifying upper limits for picture size (from 99 to 36 864 macroblocks per picture), decoder-processing rates (from 1485 to 983 040 macroblocks per second), CPB size, DPB size, bit rate (from 64 kb/s to 240 Mb/s), etc.

C. Performance Comparisons

To illustrate the performance gains that can be achieved when using H.264/AVC, we report the results of an experiment targeting video streaming applications (one of several application experiments reported in [51]). The measure of fidelity is luma peak signal-to-noise ratio (PSNR), which is the most widely used such objective video quality measure

$$\text{PSNR} = 10 \log 10(255^2/\text{MSE})$$

where MSE is the mean squared error between the original and the corresponding decoding sample values.

The four codecs compared use bitstreams conforming to the following standards:

- H.262 | MPEG-2 Visual, Main Profile (MPEG-2);
- H.263, High Latency Profile (HLP);
- MPEG-4 Visual, Advanced Simple Profile (ASP);
- H.264/AVC, Main Profile (MP).

Such applications generally support low to medium bit rates and picture resolutions, with quarter-CIF (QCIF) (176×144) resolution at 10–256 kb/s and CIF (352×288) resolution at 128–1024 kb/s being common. The set of test sequences for this comparison consists of four QCIF (10 and 15 Hz) and four CIF (15 Hz and 30 Hz) sequences. See details and further test results in [51].

Fig. 10 shows PSNR versus bit rate curves for the sequence "Tempete." For this sequence and for all others in the test set, H.264/AVC significantly outperforms the other codecs.

VI. Conclusions and Further Developments

H.264/AVC has been developed and standardized collaboratively by both the ITU-T VCEG and ISO/IEC MPEG organizations. H.264/AVC represents a number of advances in standard video coding technology, in terms of coding efficiency improvement, error/loss robustness enhancement, and flexibility for effective use over a broad variety of network types and application domains. Its VCL design is based on conventional block-based motion-compensated hybrid video coding concepts, but with some important differences relative to prior standards, which include:

- enhanced motion prediction capability;
- use of a small block-size exact-match transform;
- adaptive in-loop deblocking filter;
- enhanced entropy coding methods.

When used well together, the features of the new design provide approximately a 50% bit rate savings for equivalent perceptual quality relative to the performance of prior standards (especially for higher latency applications which allow some use of reverse temporal prediction). The performance of the H.264/AVC compliant encoder in experiments reported here and elsewhere (e.g., in [51]) clearly demonstrates the potential importance of this standard in future applications of video broadcast and streaming as well as interactive video coding.[2]

Since the completion of the first version of the H.264/AVC standard, the JVT experts group has done further work to extend the capabilities of H.264/AVC with important new enhancements known as the Fidelity Range Extensions (FREext), including four new profiles (the High, High 10, High 4:2:2, and High 4:4:4 profiles). The FREext enhancements could not be included in the scope of this paper due to the scheduling of the publication process and the need for brevity.

Acknowledgment

The authors would like to thank the experts of ITU-T VCEG, ISO/IEC MPEG, and the ITU-T/ISO/IEC Joint Video Team for their contributions. The authors would especially like to thank G. Bjøntegaard, A. Joch, F. Kossentini, A. Luthra, T. Hinz, D. Marpe, H. Schwarz, and D. Zier.

References

[1] "Generic coding of moving pictures and associated audio information—Part 1: Systems," Int. Telecommun. Union-Telecommun. (ITU-T) and Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) JTC 1, Recommendation H.222.0 and ISO/IEC 13 818-1 (MPEG-2 Systems), Nov. 1994.

[2] "Narrow-band visual telephone systems and terminal equipment," Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.320, 1999.

[2]Further information and documents of the project are available at http://ftp3.itu.int/av-arch in the directories video-site and jvt-site.

[3] "Packet-based multimedia communications systems," Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.323, 1998.
[4] "Terminal for low bit rate multimedia communication," Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.324, 1996.
[5] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," Internet Eng. Task Force (IETF), Request for Comments (RFC) 3261, 2002.
[6] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," Internet Eng. Task Force (IETF), Request for Comments (RFC) 1889, 1996.
[7] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conv. Rec.*, pt. 4, pp. 142–163, 1959.
[8] "Studio encoding parameters of digital television for standard 4 : 3 and wide-screen 16 : 9 aspect ratios," Int. Telecommun. Union-Radiocommun. (ITU-R), Recommendation BT.601-5, 1995.
[9] H. Enomoto and K. Shibata, "Features of Hadamard transformed television signal," presented at the Nat. Conf. IECE Jpn., 1965, Paper 881.
[10] H. C. Andrews and W. K. Pratt, "Fourier transform coding of images," in *Proc. Hawaii Int. Conf. System Sciences*, 1968, pp. 677–679.
[11] "Digital Compression and Coding of Continuous-Tone Still Images," Int. Telecommun. Union-Telecommun. (ITU-T) and Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) Joint Tech. Comm. (JTC) 1, Recommendation T.81 and ISO/IEC 10 918-1 (JPEG), Sep. 1992.
[12] N. Ahmed, T. Natarajan, and K. R. Rao, "On image processing and a discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–93, Jan. 1974.
[13] R. D. Kell, "Improvements relating to electric picture transmission systems," British Patent 341 811, 1929.
[14] C. Reader, "History of MPEG video compression—Ver. 4.0," Joint Video Team (JVT), JVT-E066, Oct. 2002.
[15] F. W. Mounts, "A video encoding system with conditional picture element replenishment," *Bell Syst. Tech. J.*, vol. 48, no. 7, pp. 2545–2554, Sep. 1969.
[16] "Codec for videoconferencing using primary digital group transmission," Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.120, version 1, 1984; version 2, 1988; version 3, 1993.
[17] A. Habibi, "Hybrid coding of pictorial data," *IEEE Trans. Commun.*, vol. COM-22, no. 5, pp. 614–624, May 1974.
[18] "Video codec for audiovisual services at $p \times 64$ kbit/s," Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.261, version 1, 1990; version 2, 1993.
[19] "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s—Part 2: Video," Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) JTC 1, ISO/IEC 11 172-2 (MPEG-1), Mar. 1993.
[20] "Generic coding of moving pictures and associated audio information—Part 2: Video," Int. Telecommun. Union-Telecommun. (ITU-T) and Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) JTC 1, Recommendation H.262 and ISO/IEC 13 818-2 (MPEG-2 Video), Nov. 1994.
[21] "Video coding for low bit rate communication," Int. Telecommun. Union-Telecommun. (ITU-T), Recommendation H.263, version 1, 1995; version 2, 1998; version 3, 2000.
[22] "Coding of audio-visual objects—Part 2: Visual," Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) JTC 1, ISO/IEC 14 496-2 (MPEG-4 visual version 1), 1999–2003.
[23] "Advanced video coding for generic audiovisual services," Int. Telecommun. Union-Telecommun. (ITU-T) and Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) JTC 1, Recommendation H.264 and ISO/IEC 14 496-10 (MPEG-4) AVC, 2003.
[24] Y. Taki, M. Hatori, and S. Tanaka, "Interframe coding that follows the motion," in *Proc. Institute of Electronics and Communication Engineers Jpn. Annu. Conv. (IECEJ)*, 1974, p. 1263.
[25] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COM-29, no. 12, pp. 1799–1808, Dec. 1981.
[26] S. Brofferio and F. Rocca, "Interframe redundancy reduction of video signals generated by translating objects," *IEEE Trans. Commun.*, vol. COM-25, pp. 448–455, Apr. 1977.
[27] B. Girod, "The efficiency of motion-compensating prediction for hybrid coding of video sequences," *IEEE J. Sel. Areas Commun.*, vol. 5, no. 7, pp. 1140–1154, Aug. 1987.
[28] ——, "Motion-compensating prediction with fractional-pel accuracy," *IEEE Trans. Commun.*, vol. 41, no. 4, pp. 604–612, Apr. 1993.
[29] G. J. Sullivan and R. L. Baker, "Motion compensation for video compression using control grid interpolation," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 1991, pp. 2713–2716.
[30] T. Hidaka, "Description of the proposing algorithm and its score for moving image (A part of the proposal package)," Int. Standards Org./Int. Electrotech. Comm. (ISO/IEC) JTC 1, ISO/IEC JTC 1/SC 2/WG 8 MPEG 89/188, Oct. 1989.
[31] F. Giorda and A. Racciu, "Bandwidth reduction of video signals via shift vector transmission," *IEEE Trans. Commun.*, vol. 23, no. 9, pp. 1002–1004, Sep. 1975.
[32] G. J. Sullivan, "Multi-hypothesis motion compensation for low bit-rate video coding," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 1993, pp. 437–440.
[33] M. T. Orchard and G. J. Sullivan, "Overlapped block motion compensation: An estimation-theoretic approach," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 693–699, Sep. 1994.
[34] M. Flierl, T. Wiegand, and B. Girod, "A locally optimal design algorithm for block-based multi-hypothesis motion-compensated prediction," in *Proc. IEEE Data Compression Conf. (DCC)*, 1998, pp. 239–248.
[35] B. Girod, "Efficiency analysis of multi-hypothesis motion-compensated prediction," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 173–183, Feb. 2000.
[36] T. Wiegand, X. Zhang, and B. Girod, "Long-term memory motion-compensated prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 1, pp. 70–84, Feb. 1999.
[37] T. Wiegand and B. Girod, *Multi-Frame Motion-Compensated Prediction for Video Transmission*. Norwell, MA: Kluwer, 2001.
[38] H. Everett, "Generalized Lagrange multiplier method for solving problems of optimum allocation of resources," *Oper. Res.*, vol. 11, pp. 399–417, 1963.
[39] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, pp. 1445–1453, Sep. 1988.
[40] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 1, pp. 31–42, Jan. 1989.
[41] G. J. Sullivan and R. L. Baker, "Rate-distortion optimized motion compensation for video compression using fixed or variable size blocks," in *Proc. IEEE Global Telecommunications Conf. (GLOBECOM)*, 1991, pp. 85–90.
[42] S.-W. Wu and A. Gersho, "Enhanced video compression with standardized bitstream syntax," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, 1993, pp. 103–106.
[43] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 533–545, Sep. 1994.
[44] A. Ortega, K. Ramchandran, and M. Vetterli, "Optimal trellis-based buffered compression and fast approximations," *IEEE Trans. Image Process.*, vol. 3, no. 1, pp. 26–40, Jan. 1994.
[45] T. Wiegand, M. Lightstone, D. Mukherjee, T. G. Campbell, and S. K. Mitra, "Rate-distortion optimized mode selection for very low bit rate video coding and the emerging H.263 standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 2, pp. 182–190, Apr. 1996.
[46] M. C. Chen and A. N. Willson, "Design and optimization of a differentially coded variable block size motion compensation system," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, 1996, pp. 259–262.
[47] G. M. Schuster and A. K. Katsaggelos, "A video compression scheme with optimal bit allocation among segmentation, motion, and residual error," *IEEE Trans. Image Process.*, vol. 6, pp. 1487–1502, Nov. 1997.
[48] M. C. Chen and A. N. Willson, "Rate-distortion optimal motion estimation algorithms for motion-compensated transform video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 2, pp. 147–158, Apr. 1998.
[49] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression: An overview," *IEEE Signal Process. Mag.*, pp. 23–50, Nov. 1998.
[50] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, pp. 74–90, Nov. 1998.

[51] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 688–703, Jul. 2003.
[52] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding," *Proc. IEEE*, vol. 93, no. 1, pp. 71–83, Jan. 2005.
[53] E. Steinbach, N. Färber, and B. Girod, "Standard compatible extension of H.263 for robust video transmission in mobile environments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 6, pp. 872–881, Dec. 1997.
[54] B. Girod and N. Färber, "Feedback-based error control for mobile video transmission," *Proc. IEEE*, vol. 97, no. 10, pp. 1707–1723, Oct. 1999.
[55] Q. F. Zhu and L. Kerofsky, "Joint source coding, transport processing, and error concealment for H.323-based packet video," *Proc. SPIE, Visual Commun. Image Process.*, vol. 3653, pp. 52–62, Jan. 1999.
[56] R. O. Hinds, T. N. Pappas, and J. S. Lira, "Joint block-based video source/channel coding for packet-switched networks," in *Proc. SPIE, Visual Commun. Image Process.*, vol. 3309, Jan. 1998, pp. 124–133.
[57] R. Zhang, S. L. Regunathan, and K. Rose, "Video coding with optimal Inter/Intra mode switching for packet loss resilience," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 966–976, Jun. 2000.
[58] G. Bjøntegaard, "An error resilience method based on back channel signaling and FEC," Telenor R&D, ITU-T, SG15/WP15/1, LBC-95-033, Jan. 1996. Also submitted to ISO/IEC JTC 1/SC 29/WG 11 as MPEG96/M0616.
[59] S. Fukunaga, T. Nakai, and H. Inoue, "Error-resilient video coding by dynamic replacing of reference pictures," in *IEEE Global Telecommunications Conf. (GLOBECOM)*, vol. 3, 1996, pp. 1503–1508.
[60] Y. Tomita, T. Kimura, and T. Ichikawa, "Error resilient modified inter-frame coding system for limited reference picture memories," in *Proc. Picture Coding Symp.*, 1997, pp. 743–748.
[61] M. Budagavi and J. D. Gibson, "Multiframe block motion compensated video coding for wireless channels," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, vol. 2, 1996, pp. 953–957.
[62] ——, "Error propagation in motion compensated video over wireless channels," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, vol. 2, 1997, pp. 89–92.
[63] ——, "Random lag selection in multi-frame motion compensation," presented at the IEEE Int. Symp. Information Theory, Boston, MA, 1998.
[64] T. Wiegand, N. Färber, K. Stuhlmüller, and B. Girod, "Error-resilient video transmission using long-term memory motion-compensated prediction," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 1050–1062, Jun. 2000.
[65] P. Salama, N. B. Shroff, and E. J. Delp, "Error concealment in encoded video," in *Image Recovery Techniques for Image Compression Applications*. Norwell, MA: Kluwer, 1998.
[66] W. M. Lam, A. R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, 1993, pp. 417–420.
[67] V. Varsa, M. M. Hannuksela, and Y.-K. Wang, "Non-normative error concealment algorithms," Int. Telecommun. Union-Telecommun. (ITU-T), VCEG-N62, Sep. 2001.
[68] Y.-K. Wang, M. M. Hannuksela, V. Varsa, A. Hourunranta, and M. Gabbouj, "The error concealment feature in the H.26L test model," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, vol. 2, 2002, pp. 729–732.
[69] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
[70] S. Wenger, "H.264/AVC over IP," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 645–656, Jul. 2003.
[71] T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 657–673, Jul. 2003.
[72] M. Karczewicz and R. Kurçeren, "The SP and SI frames design for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 637–644, Jul. 2003.
[73] T. Wedi and H. G. Musmann, "Motion- and aliasing-compensated prediction for hybrid video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 577–586, Jul. 2003.
[74] M. Flierl and B. Girod, "Generalized B pictures and the draft JVT/H.264 video compression standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 587–597, Jul. 2003.
[75] H. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 598–603, Jul. 2003.
[76] D. Marpe, H. Schwarz, and T. Wiegand, "Context-adaptive binary arithmetic coding for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620–636, Jul. 2003.
[77] P. List, A. Joch, J. Lainema, G. Bjøntegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 614–619, Jul. 2003.
[78] J. Ribas-Corbera, P. A. Chou, and S. L. Regunathan, "A generalized hypothetical reference decoder for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 674–687, Jul. 2003.

Gary J. Sullivan received the B.S. and M.Eng. degrees from the University of Louisville, Louisville, KY, in 1982 and 1983, respectively, and the Ph.D. and Engineer degrees in electrical engineering from the University of California, Los Angeles, in 1991.

He was the Manager of Communications Core Research at PictureTel Corporation, the former world leader in videoconferencing communication. He was a Howard Hughes Fellow and Member of the Technical Staff in the Advanced Systems Division of Hughes Aircraft Corporation and was a Terrain-Following Radar System Software Engineer for Texas Instruments. He is currently a Video Architect in the Core Media Processing Team of the Windows Digital Media Division of Microsoft Corporation, Redmond, WA. At Microsoft he designed and remains active in the extension of the DirectX Video Acceleration (DXVA) API/DDI feature of the Microsoft Windows operating system platform. He is the Chairman of the Joint Video Team (JVT) for the development of the latest international video coding standard known as H.264/AVC, which has recently been completed as a joint project between the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/WG 11, MPEG). He is also the Rapporteur of Advanced Video Coding in the ITU-T, where he has led VCEG (ITU-T Q.6/SG16) for about eight years. He is also the ITU-T video liaison representative to MPEG and he formerly served as MPEG's video chairman from March of 2001 to May of 2002. His research interests and areas of publication include image and video compression, rate-distortion optimization, motion representation, scalar and vector quantization, and error and packet loss resilient video coding.

Thomas Wiegand received the Dr.-Ing. degree from the University of Erlangen-Nuremberg, Germany, in 2000 and the Dipl.-Ing. degree in electrical engineering from the Technical University of Hamburg-Harburg, Germany, in 1995.

He is the head of the Image Communication Group in the Image Processing Department of the Heinrich Hertz Institute, Berlin, Germany. From 1993 to 1994, he was a Visiting Researcher at Kobe University, Kobe, Japan. From 1997 to 1998, he was a Visiting Researcher at Stanford University, Stanford, CA, and served as a consultant to 8×8, Inc., Santa Clara, CA. He started his research on video compression and transmission in 1995, as a Visiting Scholar at the University of California, Santa Barbara. Since then, he has published several conference and journal papers on the subject and has contributed successfully to the ITU-T Video Coding Experts Group (ITU-T SG16 Q.6, VCEG), ISO/IEC Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/WG 11, MPEG), and Joint Video Team (JVT) standardization efforts and holds various international patents in this field. He has been appointed as the Associate Rapporteur of the ITU-T VCEG (October 2000), the Associate Rapporteur/Cochair of the JVT that was created by ITU-T VCEG and ISO/IEC MPEG for finalization of the H.264/AVC video coding standard (December 2001), and as the Editor of the H.264/AVC video coding standard (February 2002).