Memory reduction for HDTV decoders

by W.-M. Lam
L. Lu

In this paper, we propose a low-cost memory reduction scheme which can reduce the memory requirement for full HDTV decoding from 12 MB to 4 MB. This scheme uses a switchable architecture that allows HDTV decoding with scalable memory reduction ratios. Depending on the input format and the amount of available memory, the scheme performs 1/4 or 1/2 memory reduction or no memory reduction. The 1/2 memory reduction is achieved by performing a block-based Hadamard transform followed by appropriate scalar quantization. The Hadamard transform has good signal energy compaction and a low computational cost. Appropriately designed nonuniform scalar quantizers take advantage of the statistics of the Hadamard transform coefficients and compress the data to match the 1/2 memory reduction target. The 1/4 memory reduction is achieved by 1/2 horizontal decimation followed by the 1/2 memory reduction scheme. Experimental results show that the proposed memory reduction schemes achieve good performance at very low computational cost, which makes them very attractive for digital TV applications.

1. Introduction

The Advanced Television Systems Committee (ATSC) was formed to establish technical standards for digital television (DTV). In December 1996, the United States Federal Communications Commission adopted the ATSC DTV standards for the nation's next-generation digital television.

The DTV standards consist of video, audio, transport, transmission, and program and system information services. The video portion of ATSC Document A/53 [1] describes a video decompression system based on the MPEG-2 standard [2] and supports 18 different video formats. These formats include pictures of the following sizes: 1920 × 1088, 1280 × 720, 704 × 480, and 640 × 480. The 1920 × 1088 and 1280 × 720 picture sizes are generally referred to as the HDTV (high-definition TV) formats, while the 704 × 480 and 640 × 480 sizes are called the SDTV (standard-definition TV) formats.

Since the cost of DTV receivers will be very high initially, it is expected that in the deployment of terrestrial DTV broadcast there will be two types of products: DTV sets with integrated receivers and digital set-top boxes (STBs) with external outputs connecting to either legacy TVs or HDTV displays. DTV sets with integrated receivers and HDTV displays will have good image quality but at a high cost. Digital STBs will allow customers to connect to legacy TVs initially and upgrade to HDTV displays at a later time. It is also possible that some STBs may be designed to connect only to legacy standard-definition TVs; in this case, some reduction in cost is possible in comparison to STBs that can drive HDTV displays. As we will see, this reduction in cost comes from a reduced-memory implementation of the decoder system. It is important to note that a DTV STB must be able to decode all 18 ATSC formats, since any one of them can be used by broadcasters. This normally requires an MPEG-2 decoder (Main Profile at High Level, or

0018-8646/99/$5.00 © 1999 IBM

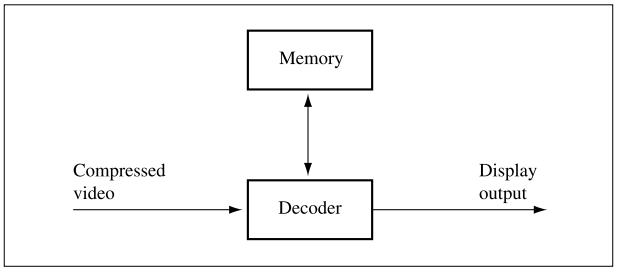

Figure 1

A commercial MPEG-2 decoder.

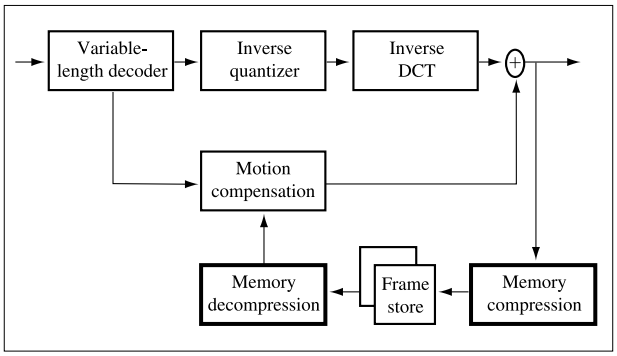

Figure 2

Architecture of a reduced-memory HDTV decoder.

MP@HL) with 12 MB of memory to support the largest picture size specified by the ATSC, i.e., 1920 × 1088. In this paper we propose a decoder which requires significantly less memory than the 12 MB of a full implementation. However, decoding with reduced memory comes at a price in video quality degradation, commensurate with the amount of memory reduction. This degradation is less noticeable when the display is a conventional standard-definition legacy TV, so this is best suited for SDTV displays. In Section 2 of this paper we formally introduce the problem of designing an HDTV decoder with reduced memory. In Section 3 we present an HDTV decoder architecture with a switchable memory reduction scheme. The decoder can accept all of the 18 input formats, and, depending on the actual format received, the decoder operates with 1/2 or 1/4 memory reduction, or no reduction, as appropriate. The goal is to reduce the memory requirement for HDTV decoding from 12 MB to 4 MB while keeping the degradation of picture quality to a minimum. In Section 4 we present the details of the memory reduction scheme we propose. In Section 5 we give simulation results using multiple HDTV sequences. Finally, in Section 6, we present our conclusions.

Table 1 Maximum memory usage in an HDTV decoder with full- and reduced-memory schemes.

	No reduction	*1/2 reduction*	*1/4 reduction*
Frame store	8.96	4.48	2.24
VBV buffer	0.95	0.95	0.95
Four-bit OSD	0.99 (1920 × 1080)	0.16 (720 × 480)	0.16 (720 × 480)
Total	10.9	5.59	3.35

2. A proposal for an HDTV decoder with reduced memory

Figure 1 is a block diagram of a commercial MPEG-2 decoder and its associated memory. This memory is used to store two video frames used as references in motion compensation of MPEG P- and B-pictures. Sometime it is necessary to store more than two frames for the decoders which cannot decode B-pictures instantaneously. It is also used as temporary storage for display output, to store compressed video data (MPEG VBV, or video buffer verifier), and to store a graphics overlay plane commonly called the on-screen display (OSD). Thus, an HDTV MPEG-2 decoder requires about 12 MB of memory, in contrast to the 2 MB required by an MP@ML standard-definition decoder [2]. **Table 1** summarizes the memory requirements of a full HDTV decoder, as well as the amount of memory utilized with our proposed 1/2 and 1/4 memory reduction schemes. The table shows data for only the 1920 × 1088 format, since this format demands the largest amount of memory. One can see that less than 4 MB are required with the 1/4 scheme.

Figure 2 shows a detailed block diagram of our proposal for an HDTV decoder with a reduced-memory implementation. This is quite similar to a normal MPEG-2 decoder, except for the addition of "memory compression" and "memory decompression" blocks. The frame store shown in this figure contains two reference frames used in MPEG motion compensation. However, and as suggested by the new blocks, these reference frames are now stored at a reduced resolution.

- *1/2 Memory reduction*

MPEG-2 frames are segmented into macroblocks of 16 × 16 pixels. To achieve memory compression, we propose to reduce the size of frames stored in memory by dividing each of their macroblocks into thirty-two 8 × 1 horizontal blocks, each of which is compressed by applying an 8 × 1 Hadamard transform, followed by scalar quantization of

the Hadamard coefficients to a pre-assigned number of bits. The total number of pre-assigned bits is half the number of bits in the original 8 × 1 block of data, thus achieving a 1/2 reduction in storage size.

The decompression process first reverses the quantization of the Hadamard coefficients and then applies the inverse Hadamard transform to restore blocks of 8 × 1 pixel values. Since the restored pixels are only an approximation of the original pixel values, there is a degradation in video quality as a result of memory reduction. Both processes are shown in **Figure 3**.

- *1/4 Memory reduction*

In this case, we propose that each 16 × 16 macroblock be first reduced horizontally by a two-tap filter. The filter averages each pair of pixels to form an 8 × 16 block, which is then divided into sixteen 8 × 1 horizontal blocks. Each 8 × 1 block is now processed as in the previous section. The decompression process first inverse-quantizes the Hadamard coefficient values and then applies the inverse Hadamard transform in order to recover, in the end, an 8 × 16 block of pixels. Each pixel is then replicated once in the horizontal direction to form the 16 × 16 decompressed block. Clearly image degradation is worse in the memory reduction case. These processes are shown in **Figure 4**.

3. Issues in designing a reduced-memory decoder

The following subsections describe various difficulties in designing a reduced-memory HDTV decoder and provide a justification for our algorithmic choices.

- *Pixel processing rates*

In decoding MPEG-2 compressed video, multiple frames are transferred—in one frame period—between the decoder itself and its associated memory. This means that in a reduced-memory HDTV decoder, with its memory compression and decompression blocks inserted between the decoder and its associated memory, the pixel rate that must be processed by these blocks can be enormous. It is thus important that the memory compression and decompression schemes be of very low computational cost for them to operate in real time. Our 1/2 algorithm uses the Hadamard transform and nonuniform scalar quantizers for compression. The Hadamard transform does not require multiplications and uses only integer operations. As we see in the next section, the scalar quantizers use at most seven comparisons. Our 1/4 algorithm uses an additional two-tap reduction filter and a zero-order (repeated-pixel) interpolation filter which are extremely easy to implement.

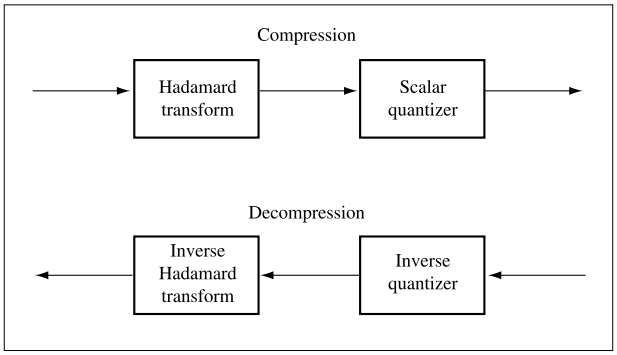

Figure 3

Block diagram of the compression and decompression scheme.

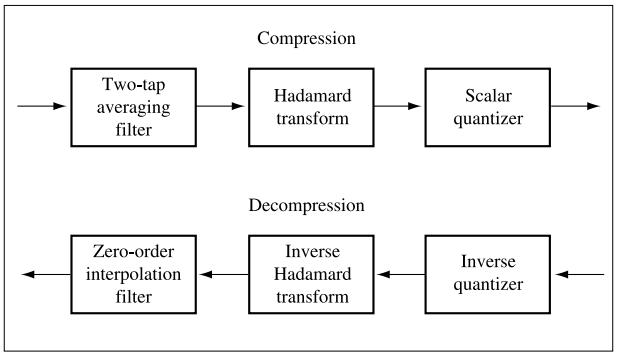

Figure 4

Block diagram of the 1/4 compression and decompression scheme.

- *Locating a pixel in memory*

During decoding of a motion-compensated macroblock, an MPEG decoder must read one or two 16 × 16 blocks from the reference frames stored in memory. Although the locations of these reference blocks do not usually correspond to macroblock boundaries, it is relatively simple to read the corresponding pixels from memory when there is no memory reduction. In this case, a decoder knows the exact location of a pixel in memory once it knows its relative position in a frame. This may no longer be true with a memory compression scheme, since some compression schemes, such as the ones we propose here, are block-based. To read a particular pixel in block-based memory compression schemes, a decoder must first know the location of the compressed block containing the pixel in question, and then decompresses the whole block, before it can address the individual pixel. In order to reduce the number of calculations a decoder must perform

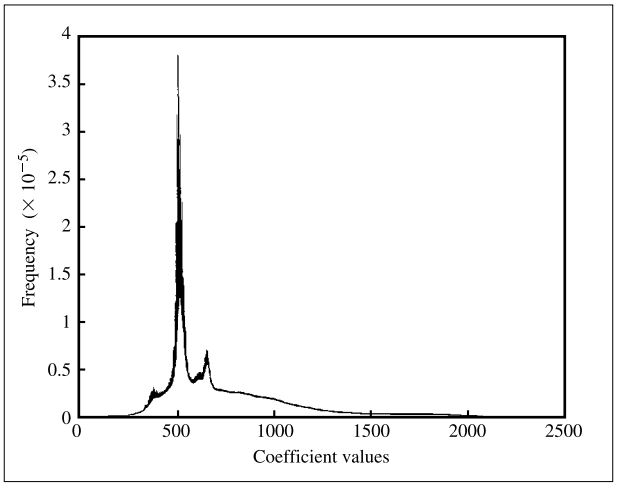

Figure 5

Histogram of dc coefficient distribution of 8 × 1 Hadamard transform for HDTV sequence "March."

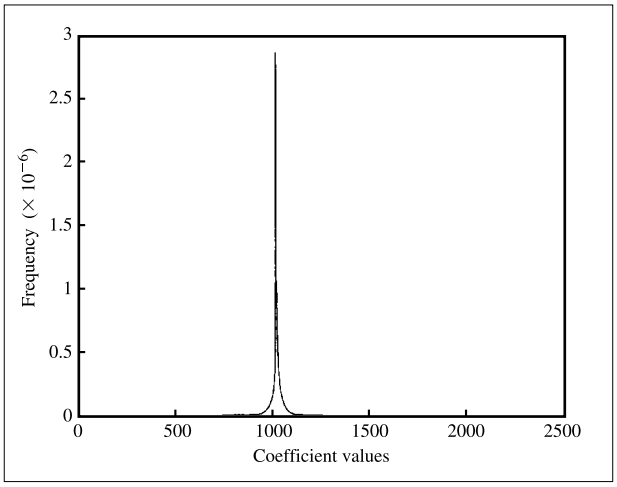

Figure 7

Histogram of last ac coefficient distribution of 8 × 1 Hadamard transform for HDTV sequence "March."

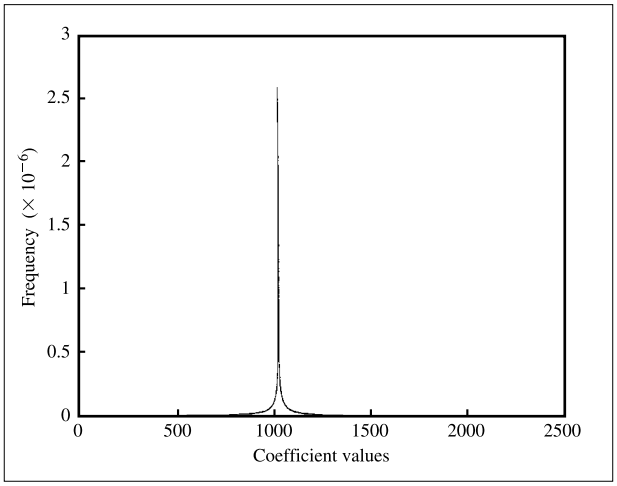

Figure 6

Histogram of first ac coefficient distribution of 8 × 1 Hadamard transform for HDTV sequence "March."

in order to locate the required compressed blocks, our memory compression scheme keeps the same compression ratio for every block. This is why our 1/2 algorithm reduces each 8 × 1 block of pixels to exactly half its size in memory.

- *Distortion introduced by memory compression*

In normal operation, an MPEG decoder operates one macroblock at a time. When a reference macroblock is decoded, for example, it must first be stored in memory prior to decoding of the next reference macroblock. When memory compression is applied to a decoded macroblock, the size of the macroblock imposes a natural constraint on the size of the memory-compressed block. For compression by a factor of 2, for example, the compressed macroblock should be of size 8 × 16. Coding theory tells us that distortion can be minimized by compressing the full 16 × 16 macroblock as a whole instead of dealing with several smaller segments, such as the 8 × 1 segments of our scheme. On the other hand, the bigger the compressed block, the harder it is to locate and extract individual pixels from memory. We must then balance the need to minimize pixel distortion with the need to easily locate and read an individual pixel from memory. Our choice of 8 × 1 blocks for memory compression addresses this balance.

- *Interlace formats*

The DTV standard consists of an interlaced 1920 × 1088 HDTV format. Designing a vertical filter for interlace pictures is a difficult task, since frame vertical filtering can produce flickering artifacts and field filtering can lose vertical resolution. Our compression scheme uses only horizontal filtering and a horizontal 8 × 1 transform scheme to avoid these problems.

4. Algorithmic details

Since the memory reduction is used in the decoder, it is important to keep the complexity of the algorithm as low as possible so that a low-cost, real-time implementation

Table 2 Estimated standard deviations of Hadamard transform coefficients of sequence "March."

Coefficient	dc_0	ac_1	ac_2	ac_3	ac_4	ac_5	ac_6	ac_7
Y standard deviation	323.24	119.04	74.99	57.07	31.98	34.24	35.56	24.82
U standard deviation	62.99	21.82	14.60	11.41	6.91	7.00	6.48	5.01
V standard deviation	96.65	24.83	16.93	13.37	8.61	8.29	7.51	5.71

can be realized. This is a challenge because we must also simultaneously achieve good performance, which translates into good-quality results. In this section, we elaborate on the elements of our proposed algorithms that address these objectives.

Linzer et al. [3] were the first to propose the use of Hadamard transforms [4] for memory reduction in MPEG decoding. They used a 4×1 Hadamard transform followed by uniform quantization to achieve a 1/2 reduction ratio. Their proposed scheme had three shortcomings: 1) Their 4×1 block size was too small to effectively utilize the correlations among pixels. 2) Their uniform quantization scheme did not match the probability distribution of the Hadamard coefficients, and thus could not compress the coefficients efficiently. 3) Their scheme could not provide the higher memory reduction requirement in HDTV decoding.

Our proposed memory reduction schemes address the above shortcomings by using an 8×1 Hadamard transform with matched nonuniform quantizers to achieve a 1/2 reduction, and with a spatial decimation filter for a higher reduction ratio.

- *Hadamard transform*

The elements of the basis vectors of the Hadamard transform matrix consist of only $+1$ and -1 values. The transform calculations require no multiplications. This property makes the Hadamard transform a fast operation which can be implemented in $O(N \log_2 N)$ additions and subtractions for an $N \times 1$ one-dimensional transformation. Furthermore, the Hadamard transform **H** is real, symmetric, and orthogonal; thus, the forward and inverse Hadamard transforms are identical. Let x_k, $0 \le k \le N - 1$, be the $N \times 1$-image pixel samples and c_k, $0 \le k \le N - 1$, denote the Hadamard transform coefficients. The $N \times 1$ one-dimensional Hadamard transform is then defined as

$$c_k = \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} x_l h_{k,l} \qquad 0 \le k \le N-1, \tag{1}$$

where $h_{k,l}$ is the (k, l)th element of **H**. Using the real, symmetric, and orthogonal properties of the transform, it is straightforward to show that the inverse Hadamard transform can be carried out as

$$x_l = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} c_k h_{k,l} \qquad 0 \le l \le N-1. \tag{2}$$

Another important property of the Hadamard transform is its good energy compaction for highly correlated images, which is an essential requirement for signal compression. In our applications, the high-definition images have, in general, higher correlation than the standard-definition images of similar content. Therefore, the Hadamard transform matches very well our need for low computational complexity and good performance. We employ an 8×8 Hadamard matrix to transform an 8×1-pixel block into eight Hadamard coefficients so that subsequent compression processes can be made more efficient. The 8×8 Hadamard transform matrix is defined as

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}. \tag{3}$$

- *Scalar quantization*

Scalar quantization is the most widely used quantization technique in real applications. It has low computational complexity and is easy to implement. It can also achieve reasonably good compression performance if applied properly. Since our objectives require good performance as well as low complexity, we use scalar quantization to compress the Hadamard transform coefficients in our memory reduction algorithm. In designing the quantizers, the first requirement is to obtain knowledge of the statistics of the Hadamard transform coefficients.

Since we had examined several HDTV sequences and their ac Hadamard coefficients showed similar statistical distributions, we use an HDTV sequence called "March" as an example. This sequence consists of athletes marching into a stadium at moderate speed with a lot of background detail. **Figures 5**, **6**, and **7** are the histograms of the dc

Table 3 Bit allocation to quantize the eight Hadamard transform coefficients.

Y, U, V coefficients	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7
Bit allocation	7	5	4	4	3	3	3	3

Table 4 The three symmetric nonuniform scaler quantizers (only positive levels shown).

Q_1		Q_2		Q_3	
q_k	r_k	q_k	r_k	q_k	r_k
8	0	9	0	9	0
25	16	30	19	32	19
43	33	53	40	68	46
62	52	81	66		91
83	72	117	97		
106	94	165	137		
131	118	240	193		
159	144		287		
191	174				
221	207				
269	246				
319	292				
383	347				
468	418				
603	518				
	687				

coefficient c_0, the first ac coefficient c_1, and the last ac coefficient c_7 of the luminance signal, respectively.

The histograms show two characteristics which give us important information for designing the quantizers:

1. The Hadamard transform ac coefficients are nearly identically distributed after normalization and resemble a Laplacian density function as defined by

$$p(x) = \frac{1}{\omega^2} e^{-2|x-m|/\omega^2}. \tag{4}$$

2. The distribution of the dc coefficient is not symmetric, and its shape depends on the brightness of the pictures in the sequence. On the basis of these characteristics, we decided to use a uniform quantizer to compress the dc coefficient and a set of nonuniform scalar quantizers to encode the ac coefficients.

The next step in designing the quantizers is to determine the bit allocations among the 8×1 transform coefficients. Since the bit allocations depend on the required memory reduction ratio, and in our application the typical ratios will be 1/2 or 1/4, we concentrate on these two cases. The extension to other cases is straightforward.

1/2 Memory reduction

In this case the memory requirement is reduced by half. Each picture pixel is represented by a 8-bit integer which ranges from 0 to 255. Therefore, each 8×1-pixel block which originally has 64 bits will be compressed down to 32 bits after Hadamard transform and quantization.

In **Table 2** we can see that the dc coefficient has a very large variance; i.e., most of the signal energy is concentrated in the dc coefficient. Therefore, sufficient bits should be allocated to quantize the dc coefficient in order to achieve good results. We decided from experiments that a 7-bit (128-level) uniform quantizer is necessary for our purposes. The dc coefficient after the Hadamard transform varies from 0 to 2040, so the uniform quantizer can easily be determined as

$$t_k = \frac{2040(k-1)}{128} \qquad k = 1, \cdots, 129;$$

$$r_k = t_k + \frac{1020}{128} \qquad k = 1, \cdots, 128.$$

To avoid any floating-point calculation in our algorithm, we round the quantization levels and reconstruction levels to integers. This minor adjustment results in a nearly uniform quantizer with step size of 16, except for the four cells in the center, which have a step size of 14.

Once the quantizer for the dc coefficient is determined, we can allocate the remaining 25 bits among the seven ac coefficients. Since we do not use variable-length coding in order to keep the complexity and cost low, each coefficient can be allocated only an integer number of bits. Furthermore, as we noticed before, the seven ac coefficients have nearly identical distributions when they are normalized by the corresponding standard deviation and the bit rate is relatively high. We can use the following formula [5] to achieve nearly optimal bit allocation:

Table 5 Estimated standard deviations of 8×1 Hadamard transform coefficient statistics obtained from closed-loop (including quantization) operation.

Coefficient	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7
Y standard deviation	321.42	115.93	70.57	54.94	29.57	31.89	34.09	24.52

$$b_i = \frac{B}{k} + \log_2 \frac{\omega_i^2}{\alpha^2}, \tag{5}$$

where B is the total number of available bits and k is the number of coefficients, and

$$\alpha^2 = \left(\prod_{i=1}^{k} \omega_i^2 \right)^{1/k} \tag{6}$$

is the geometric mean of the variances of the coefficients. After calculation using Equation (5) and rounding the results to integers, we obtain the same bit allocation for both luminance Y and chrominances U and V in **Table 3**.

To keep the number of quantizers as small as possible, we combine the seven ac coefficients into three groups according to the bit allocation and design a nonuniform scalar quantizer for each group. This quantizer is used to compress the luminance signal as well as the chrominance signals. Thus, we need only design three nonuniform scalar quantizers. Specifically, on the basis of the Laplacian density assumption with the range truncated from −1020 to 1020, we designed a 5-bit quantizer with 31 levels for the ac coefficient c_1, a 4-bit quantizer with 15 levels for the coefficients c_2 and c_3, and a 3-bit quantizer with seven levels for the coefficients c_4, c_5, c_6, and c_7. In designing the 4-bit quantizer, we obtained an estimated sample variance of 66.63 using a combined histogram of c_2 and c_3. Similarly, in designing the 3-bit quantizer we used an estimated sample variance of 31.92 from the combined histogram of c_4, c_5, c_6, and c_7. Since we use only integer numbers, we truncate all floating numbers to integer values. The three nonuniform symmetric scalar quantizers are designed to minimize the mean squared distortion and are presented in **Table 4**. The quantizers are mid-level quantizers which have a reconstruction level of value 0 to avoid error in small values which are close to zero.

1/4 Memory reduction

In this case, the eight Hadamard transform coefficients must be compressed into 16 bits in total. If we spend six or seven bits in coding the dc coefficient, the remaining nine or ten bits will not be adequate to achieve reasonably good quality in quantizing the seven ac coefficients unless some coding technique other than the scalar quantization is used. Also, the high-bit-rate assumption for bit allocation in Equation (5) is not valid here. However, the requirements for low complexity, low cost, and easy implementation have prevented us from adopting a more sophisticated coding method, such as vector quantization. In order to achieve good picture quality while meeting the requirements in the 1/4 memory reduction case, we propose a simple scheme: horizontal filtering to reduce the image size by 2, followed by the 8 × 1 Hadamard transform and scalar quantization.

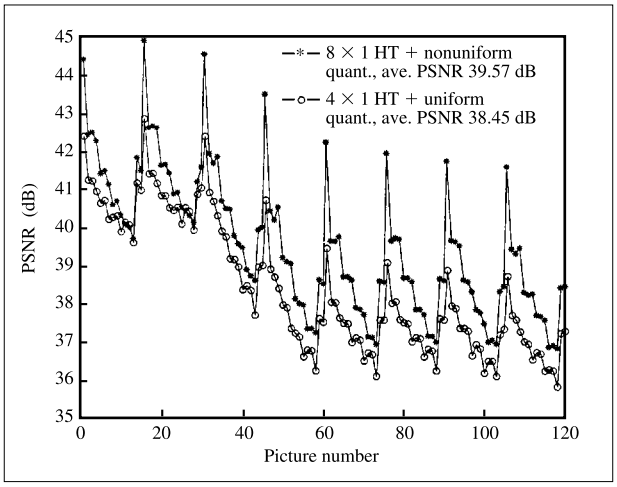

Figure 8

Performance comparison of 8 × 1 Hadamard transform with nonuniform quantization and 1 × 4 Hadamard transform with uniform quantization in 1/2 memory reduction for HDTV sequence "Dome."

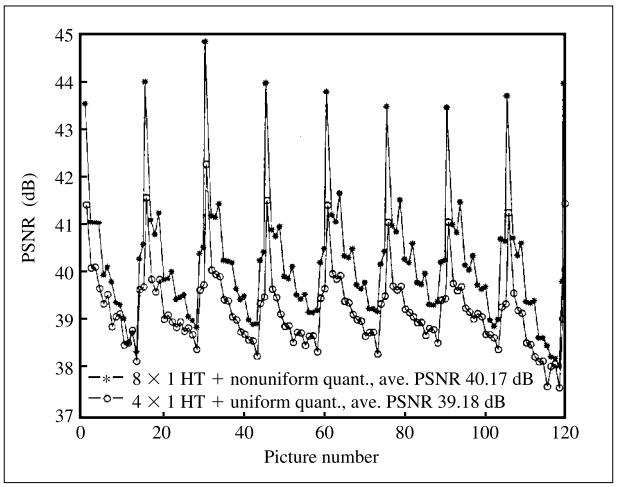

Figure 9

Performance comparison of 8 × 1 Hadamard transform with nonuniform quantization and 1 × 4 Hadamard transform with uniform quantization in 1/2 memory reduction for HDTV sequence "Whale."

The 8 × 1 Hadamard transform and the scalar quantization operation are exactly the same as described above. The horizontal filtering operation uses a simple two-tap filter which takes two pixels as inputs and outputs their average value. This filtering process requires only one addition and one shift operation.

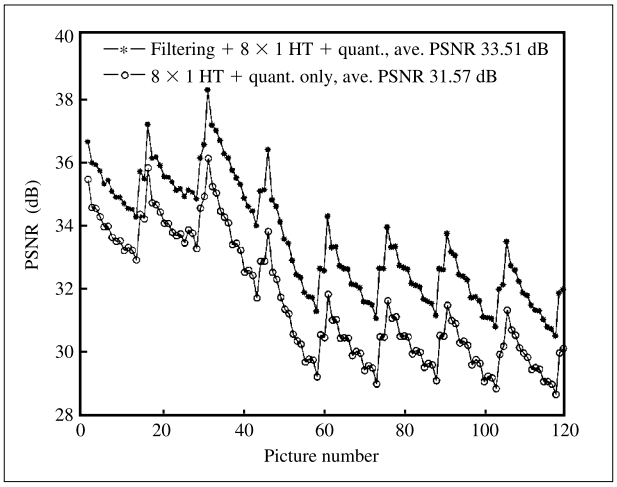

Figure 10

Performance comparison of 8 × 1 Hadamard transforms with filtering and without filtering in 1/4 memory reduction for HDTV sequence "Dome."

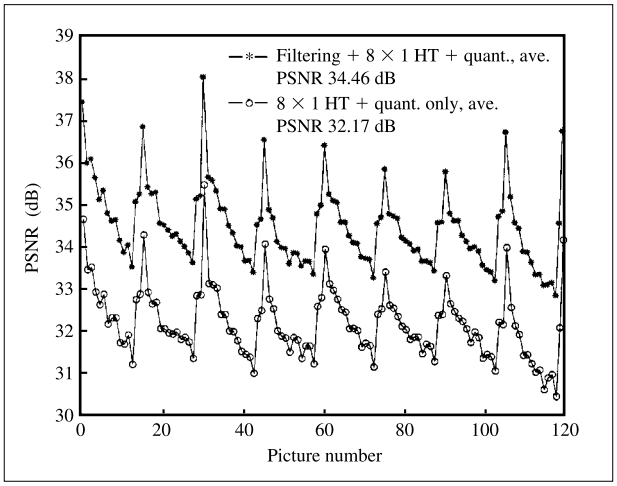

Figure 11

Performance comparison of 8 × 1 Hadamard transforms with filtering and without filtering in 1/4 memory reduction for HDTV sequence "Whale."

For clarity, we summarize our memory reduction scheme as follows. For 1/4 memory reduction, the decoded picture is first processed by half-horizontal filtering, resulting in a picture with the horizontal resolution reduced by half. The resulting picture undergoes the 8 × 1 Hadamard transform and is subsequently compressed with scalar quantization. For 1/2 memory reduction, the horizontal filter is switched off; the decoded pictures need only go through the operations of the 8 × 1 Hadamard transform and scalar quantization.

5. Experimental results

Our algorithm for both 1/2 and 1/4 memory reduction has been tested and evaluated on high-definition image sequences. To verify that the statistics of the Hadamard transform coefficients that we collected in the open loop (without quantization) process are close to the statistics of the closed loop (with quantization), we collected the statistics of the Hadamard transform coefficients, again using the designed quantizers. The standard deviations of the closed-loop statistics are given in **Table 5**. Since the results are very close to the data in Table 2, our quantizer design appears to be valid.

To reduce the computational complexity further, we implemented the encoding operation of the scalar quantization with a binary search, so that there are at most only $\log_2 2^{b_i} = b_i$ comparisons needed to find the code-word index, where b_i is the number of bits for the quantizer. The decoding process is just a simple table lookup operation. In the following we present the comparison and evaluation results on the two high-definition image sequences "Dome" and "Whale," which are not used to design our quantizers. "Dome" has a resolution of 1920 × 1088 and "Whale" is 1920 × 1024.

The peak signal-to-noise (PSNR) results of our 1/2 memory reduction scheme on "Dome" and "Whale" are plotted in **Figures 8** and **9**, respectively. For reference and comparison purposes, we also obtained the PSNR results of a 1/2 memory reduction scheme using the 4 × 1 Hadamard transform followed by uniform scalar quantization, as in [3]. The results have shown that all of the pictures in our scheme have a better picture quality in terms of PSNR values, with an average gain of more than 1 dB in both sequences. The subjective visual picture quality of our scheme is also better.

The PSNR results of our 1/4 memory reduction algorithm on the two test sequences are presented in **Figures 10** and **11**. We also implemented a 1/4 memory reduction scheme which uses only the 8 × 1 Hadamard transform and 1/4 scalar quantization for purposes of comparison. The 1/4 scalar quantization uses three quantizers to achieve the compression ratio; they are seven bits uniform quantizer for dc_0, three bits for ac_1, and two bits for ac_2, ac_3, and ac_4. All of the other ac coefficients are set to 0. The results have demonstrated that the scheme with half-horizontal filtering followed by the 8 × 1 Hadamard transform and 1/2 scalar quantization works significantly better than the scheme with the 8 × 1 Hadamard transform and 1/4 scalar quantization. A 2-dB gain in the PSNR is achieved, and the visual quality is also better. The tradeoff of lower-resolution images with fewer

quantization artifacts has paid off both in PSNR and in picture quality.

6. Conclusion

This paper presents an algorithm for memory reduction in an HDTV decoder. The algorithm uses an 8×1 Hadamard transform, scalar quantizers, and a two-tap horizontal filter for compression. It requires only simple hardware with low computational cost and can be performed in real time during the HDTV decoding process. Memory can be reduced from 12 MB to 4 MB to make HDTV STBs cost-effective.

References

1. Advanced Television Systems Committee, "ATSC Digital Television Standard," *Document A/53*, September 1995.
2. International Organization for Standardization, "Coding of Moving Pictures and Associated Audio," *ISO/IEC 13818-2*, November 1994.
3. E. Linzer, M. West, and P. Westerink, "Reduced Memory MPEG-2 Decoding on Computers," IBM internal report, January 2, 1996; available on request from Peter H. Westerink, IBM Research Division, 30 Saw Mill River Road, 790-H3-J24, Hawthorne, NY 10532.
4. A. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1989.
5. A. Gersho and R. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Norwell, MA, 1992.

Received June 5, 1998; accepted for publication July 23, 1999

Wai-Man Lam *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (lamw@us.ibm.com).* Dr. Lam received his B.Sc. degree from Johns Hopkins University in 1987 and his Ph.D. degree in electrical engineering from Princeton University in 1992. He worked at Thomson Consumer Electronics, Indianapolis, Indiana, for several years, joining the IBM Thomas J. Watson Research Center in 1996 as a Research Staff Member. He currently manages the Video and Image Communications group. Dr. Lam's principal areas of interest have included digital video processing, embedded devices development, and wireless communications.

Ligang (Larry) Lu *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (lul@us.ibm.com).* Dr. Lu is a Research Staff Member at the IBM Thomas J. Watson Research Center in the Video and Image Systems group. He received his Ph.D. in 1995 from Rensselaer Polytechnic Institute. He joined IBM in 1997 and has been working in the areas of visual communication systems and multimedia technologies. From 1995 to 1997, he was with the Thomson Consumer Electronics research center in Indianapolis, Indiana. Dr. Lu's research interests include digital video processing, visual communication and multimedia technologies, structured quantizer design, etc. He received the Alan Dumont Award from Rensselaer in 1996 for his academic achievement.