Adaptive Image Compression Based on Segmentation and Block Classification

Mahmoud R. El-Sakka, Mohamed S. Kamel

Pattern Analysis and Machine Intelligence Laboratory, Systems Design Engineering Department, University of Waterloo, Waterloo, Ontario N2L-3G1, Canada

ABSTRACT: This article presents a new digital image compression scheme which exploits a human visual system property—namely, recognizing images by their regions—to achieve high compression ratios. It also assigns a variable bit count to each image region that is proportional to the amount of information it conveys to the viewer. The new scheme copes with image nonstationarity by adaptively segmenting the image into variable block–sized regions and classifying them into statistically and perceptually different classes. These classes include a smooth class, a textural class, and an edge class. Blocks in each class are separately encoded. For smooth blocks, a new adaptive prediction technique is used to encode block averages. Meanwhile, an optimized DCT-based technique is used to encode both edge and textural blocks. Based on extensive testing and comparisons with other existing compression techniques, the performance of the new scheme surpasses the performance of the JPEG standard and goes beyond its compression limits. In most test cases, the new compression scheme results in a maximum compression ratio that is at least twice of JPEG, while exhibiting lower objective and subjective image degradations. Moreover, the performance of the new block-based compression is comparable to the performance of the state-of-the-art wavelet-based compression technique and provides a good alternative when adaptability to image content is of interest. Int J Imaging Syst Technol, 10, 33–46, 1999

I. INTRODUCTION

It is widely believed that a picture is worth more than a thousand words. However, dealing with digital pictures (images) requires far more computer memory and transmission time than that needed for plain text. To be able to handle efficiently the huge amount of data associated with images, compression schemes are needed. Image compression is a process intended to yield a compact representation of an image, hence reducing the image storage/transmission requirements.

Generally, images carry three main types of information: redundant, irrelevant, and useful. Redundant information is the deterministic part of the information which can be reproduced, without loss, from other information contained in the image (i.e., interpixel redundancy): for example, low-varying background information. Irrelevant information is the part of information that has enormous details which is beyond the limit of perceptual significance (i.e., psychovisual redundancy). Useful information is the part of information which is neither redundant nor irrelevant.

Decompressed images are usually observed by human beings. Therefore, their fidelities are subject to the capabilities and limitations of the human visual system (HVS). A significant property of the HVS is the fact that it recognizes images by their regions and not by the intensity value of their pixels (Cornsweet, 1971). In addition to this property, when an observer looks to an image, trying to understand it, he or she searches for distinguishing features such as edges (not pixel values) and mentally combines them together into recognizable groupings.

This HVS property (recognizing images by their regions) can be exploited by segmenting images into regions based on the amount of information each region conveys to the viewer. Then, regions of each category could be encoded using a distinct encoding procedure. This encoding procedure should preserve the main visual characteristics of this particular category (focusing on useful information) while reducing the existing correlation (reducing redundant information), and neglecting some of the irrelevant details (omitting irrelevant information).

The idea of block classification has been used in a number of encoding techniques. In Chen (1989) and Nasiopoulos et al. (1991), quadtrees are used to segment a given image into smooth blocks of variable sizes and nonsmooth blocks of a fixed smaller size. In Chen (1989), three-level 4×4 to 16×16 bottom-up quadtrees were used. The DCT followed by a fixed step-size quantizer was used to encode each of these blocks, but with coarser quantization applied to smooth-blocks (large blocks). In Nasiopoulos et al. (1991), two-level 4×4 to 2×2 top-down quadtrees were used. The block average was exploited to encode smooth blocks, while absolute moment block truncation coding (AM-BTC) (Delp and Mitchell, 1979) and AM-BTC with lookup tables (based on block activities) were used to encode nonsmooth blocks. The results reported in both of those papers showed good-quality reconstructed images with compression ratios of $<12{:}1$, i.e., at a bit rate of more than 0.66 bits per pixel (bpp), assuming 8-bit gray levels. The reason for these high bit rates might be either using the same fixed step-size quantizer for all DCT coefficients regardless of their locations within the transformed block, or using the BTC-encoding algorithm which has a limited compression capability.

In Lee and Crebbin (1994), three-level 16×16 to 4×4 top-down quadtrees were used. The nonsmooth blocks were further

Correspondence to: M. S. Kamel
Grant sponsor: the Natural Sciences and Engineering Research Council of Canada

 CCC 0899–9457/99/010033-14

Table I. Performance results of various encoding techniques for the 512 × 512 "Lena" image.

Method	Ref.	CR	RMSE
Segmentation: trilevel quadtrees {4, 8, 16}; encoding: DCT	Chen (1989)	6.62:1	5.50
Segmentation: dilevel quadtrees {4, 2}; encoding: average/AM-BTC	Nasiopoulos et al. (1991)	5.88:1	5.47
		6.40:1	5.70
		10.53:1	9.10
Segmentation: trilevel quadtrees {16, 8, 4}, block classification; encoding: DPCM/VQ in DCT domain	Lee and Crebbin (1994)	22.28:1	5.87
		23.95:1	6.11
		25.89:1	6.37
		28.17:1	6.68
Segmentation: quadruple-level quadtrees {32, 16, 8, 4}, block classification; encoding: VQ	Vaisey and Gersho (1992)	22.04:1	6.90
		28.88:1	7.88
Segmentation: binary trees; encoding: first-order polynomial fitting	Radha et al. (1996)	80.00:1	13.23
		114.00:1	14.84
Image decomposition; encoding: chain encoding/DCT/subband	Ran and Farvardin (1995a)	10.94:1	3.20
		16.43:1	3.94
		32.39:1	5.62
		64.00:1	8.52

classified into eight classes according to the orientation of their edges. Then, VQ in the DCT domain was used to encode the AC coefficients of each class separately, while the DC coefficients were DPCM encoded. Although both textural and edge blocks represent high-frequency data, textural blocks usually convey a low amount of information to the viewer and should therefore be assigned fewer bits. In Vaisey and Gersho (1992), this idea was exploited. In the first stage, an initial four-level 32 × 32 to 4 × 4 top-down quadtree segmentation of the input image was performed to locate regions which have homogeneous mean values. Then, the mean of each block was encoded to form the mean image, which was then smoothed to generate an interpolated image. The interpolated image was subtracted from the input image to obtain a residual image. The residual image was segmented by other four-level 32 × 32 to 4 × 4 top-down quadtrees and their leaves were classified into three classes (smooth, textural, and edge classes). Finally, the classified residual blocks were vector quantized at different bit rates according to their levels of information (i.e., according to the class to which they belong). The reported results show that these two techniques achieved good-quality reconstructed images with compression ratios between 10:1 and 32:1, i.e., at bit rates between 0.8 and 0.25 bpp, depending on the nature of the original image. The main disadvantage of these two techniques is the long time needed to execute them. For example, the latter technique typically takes about 10 min of CPU time on a Sun-3/260 computer with a floating-point accelerator to encode a 512 × 512 image. Most of this time is devoted to searching the codebooks. (The CPU power of a Sun-3/260 computer with a floating-point accelerator is about 60 times less than the CPU power of a single-processor Sun-Ultra-1 computer.)

In Wu (1992) and Radha et al. (1996), instead of using quadtrees binary trees were used, and each leaf region was represented by a first-order polynomial function. The polynomial function coefficients were determined using the least-square approximation method. The main difference between these two techniques is that the former restricts the dividing line orientations to one of four possible orientations—namely, 0°, 45°, 90°, and 135°. Hence, the latter technique provides a more general framework, yet requires higher computational complexity, than the former technique. The reported results show that these two techniques obtained a fair-quality reconstructed images with compression ratios between 80:1 and 160:1, i.e., at bit rates between 0.1 and 0.05 bpp.

Other segmentation techniques, such as region growing (Kunt et al., 1985, 1987), perform a more precise isolation on statistically homogeneous regions. However, both the number of regions and their shapes are determined solely by the contents of the examined image. This implies that a very large number of bits may be needed to represent the shape and location of each region.

Another approach for compressing images (Ran and Farvardin, 1995a) is to decompose images, instead of segmenting them, into three components (namely, strong edge, smooth, and textural components). The model used is the curvature energy minimization model [developed in Ran and Farvardin (1995b)]. The intensity and geometric information of strong edge contours in the strong edge component were encoded separately using the chain-encoding technique (Freeman, 1961; Neuhoff and Castor, 1985). Two alternatives for encoding the smooth and textural components were suggested: namely, entropy-encoded fixed-block-size adaptive DCT encoding and entropy-encoded subband encoding. It has been reported that this technique has better performance over the JPEG continuous-tone image compression standard (Wallace, 1992). However, its main disadvantage is that it is extremely time-consuming. The main bulk of complexity resides in the three-component decomposition. For example, it typically takes about 23 min of CPU time on a Sun-Sparc-station-1 just to decompose a 256 × 256 image. (The CPU power of a Sun-Sparc-station-1 computer is about 15 times less than the CPU power of a single-processor Sun-Ultra-1 computer.) Table I presents performance results of this technique as well as several other encoding techniques mentioned in this section.

In this article, a new digital image compression technique called adaptive block-based compression based on segmentation and classification (ABC-SC) is proposed (El-Sakka, 1997). The main objectives of ABC-SC are:

1. to achieve a good rate-distortion performance at diverse compression levels while maintaining a practical compression/decompression time, and
2. to achieve very high compression ratios while maintaining, at least, the performance of the JPEG standard at low compression ratios.

ABC-SC exploits one of the HVS properties—the recognition of images by their regions—to achieve high compression ratios. It also assigns a varying bit count to each image region that is proportional

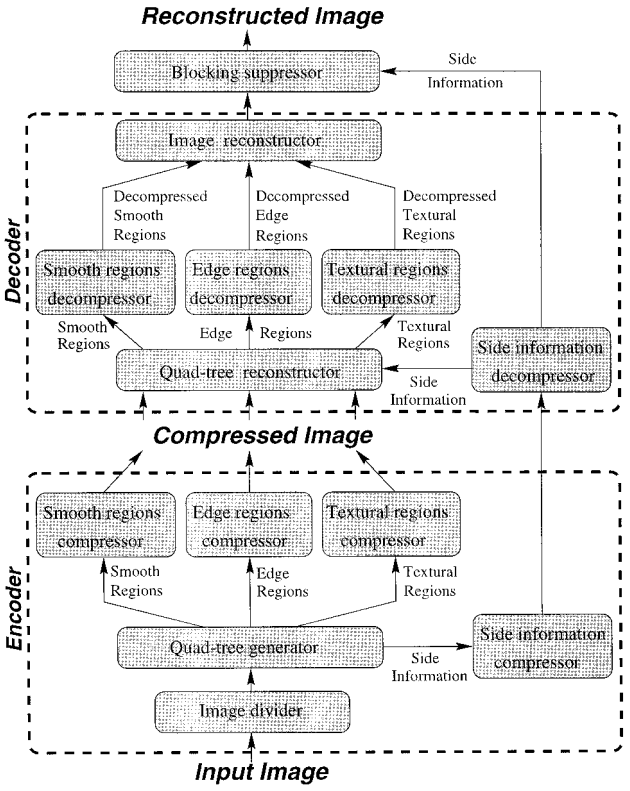

Figure 1. Block diagram of ABC-SC.

to the amount of useful information conveyed in each region. ABC-SC uses three-level 32×32 to 8×8 top-down quadtrees to segment a given image, where the quadtree leaves are classified into three classes (smooth, textural, and edge classes). Unlike Lee and Crebbin's work (1994), ABC-SC uses a new adaptive prediction technique to encode smooth regions. Meanwhile, instead of using a DCT-based technique with a fixed step-size quantizer, as in Chen (1989), an optimized DCT-based technique is used to encode both edge and textural blocks. Unlike VQ-based techniques, ABC-SC does not require training or image ensemble statistics to be encoded. Since ABC-SC is a block-based technique, blocking artifacts might appear in the reconstructed images, especially at very high compression ratios. Therefore, an optional postprocessing filtering scheme may be applied to restore the smooth continuity of the gray-scale intensities over the entire reconstructed image. This postprocessing filtering scheme uses the already-saved region content information. Consequently, no additional bits are needed.

The organization of this article is as follows. Sections II and III describe the proposed technique. In Section IV, performance metrics for evaluating ABC-SC are discussed and the results of ABC-SC are presented. Section V gives some concluding comments.

II. OVERVIEW OF ABC-SC

In the ABC-SC technique (El-Sakka, 1997), as shown in Figure 1, a given input image is first subdivided into superblocks of 32×32 pixels. Then, for each of these superblocks, a quadtree with minimum subblocks of 8×8 pixels is built. A block is declared to be a leaf block if its homogeneity measure satisfies a certain threshold or if the bottom of the tree is reached; otherwise the tree is traversed down further. The leaf blocks are then classified into three perceptual classes according to the amount of useful information each block conveys to the viewer. These perceptual classes are a smooth class, a textural class, and an edge class. As a result of this classification, three segments of the image are produced (one per perceptual class). Each of these segments is then encoded separately. For the smooth image segment, only the quantized average of each block is encoded using a new adaptive differential pulse code modulation (ADPCM) technique. In ADPCM, different linear prediction rules, including second- and third-order two-dimensional prediction rules, are used to predict the current block average—only one rule per prediction is applied. The choice of prediction rule is based on the differences between the neighboring encoded block averages. For the textural and edge image segments, blocks are DCT transformed. Then, the transformed coefficients are quantized, where the quantization matrix for each of the two classes is optimized based on the amount of information conveyed in each region—coarser quantization is applied to the textural blocks than to the edge blocks. This is because the textural blocks convey less information to the viewer than edge blocks. Since the averages of the adjacent blocks are strongly correlated, the DC coefficients are ADPCM encoded. Meanwhile, the AC coefficients of each block are zigzag ordered and then run-length (RL) encoded (Golumb, 1966). Finally, each group of encoded data—the quadtree structures (also called the side information), the encoded average of each smooth block with the encoded DC coefficients, and the encoded AC coefficients—are arithmetically and separately encoded to produce the compressed image.

At the receiver's side, each group of data is arithmetically decoded. Then, using the decoded side information, the quadtrees are reconstructed. Next, the quantized coefficients are multiplied by the quantization step size and decoded to form a reconstructed image segment. For the smooth segment, the inverse ADPCM (I-ADPCM) is applied. Meanwhile, the RL decoding, the inverse zigzag ordering, the inverse DCT (I-DCT), as well as the inverse ADPCM are applied to the edge and textural image segments. Finally, all of these reconstructed image segments are gathered to form the reconstructed image.

At high compression levels, smooth-block boundaries may become visible and viewers might suffer from the annoying intrablocking effect. The reason for having this blocky appearance is that smooth blocks were represented only by their quantized averages, although their pixel intensity values might not be close enough to this quantized average. Since it is always desired to reach high compression ratios and still have a good reconstruction quality, a postprocessing filtering scheme should be applied to reduce this annoying intrablocking effect and return the lost gray-level continuity back to the reconstructed image.

A. Homogeneity Criteria.

1. Smoothness Criterion. Many smoothness criteria have been proposed in previous studies, e.g., the weighted sum of the absolute values of the transform coefficients (Gimlett, 1975), the block variance (Vaisey and Gersho, 1992), and the squared deviation from the neighboring pixels mean (El-Sakka and Kamel, 1995). Since the main focus of this work is to investigate whether the segmentation approach will improve the compression performance, the block variance as a simple and adequate measure is selected to be the smoothness criterion in this work. To identify an $m \times m$ block to be

a smooth block, its variance σ_m^2 must be less than or equal to a certain threshold T_m, i.e.,

$$\sigma_m^2 \leq T_m \tag{1}$$

Since ABC-SC is not sensitive to segmentation errors, it is believed that the variance criterion is sufficient for the purpose of this research. This is the case since even if a segmentation error is made, it would result in only a small change in the bit rate.

2. Textural Criterion. It was found (Vaisey and Gersho, 1992) that a block belonging to the textural class must have a high level of details and exceed a minimum size. Also, the block's contents together with the contents of its surrounding blocks must be homogeneous. It was shown experimentally that 8×8 may be an adequate size for textural blocks. Therefore, the candidates for the textural class are those 8×8 blocks which do not satisfy the smoothness criterion. To test the textural surrounding of a given candidate block, a 16×16 augmented block which has the candidate block at its center is considered. This augmented block is then broken up into four nonoverlapped subblocks of size 8×8. Next, the variances of these four blocks (σ_1^2, σ_2^2, σ_3^2, and σ_4^2), as well as the variance of the candidate block (σ_5^2) are calculated. The candidate block is declared as a textural block if the average of these five variances is greater than or equal to a threshold T_{average}, and the relative absolute deviation between each of them and their average is less than or equal to a threshold $T_{\text{deviation}}$, i.e.,

$$\sigma_{\text{avg}}^2 \geq T_{\text{average}} \tag{2}$$

and

$$\frac{|\sigma_i^2 - \sigma_{\text{avg}}^2|}{\sigma_{\text{avg}}^2} \leq T_{\text{deviation}}, \qquad i = 1, 2, \ldots, 5 \tag{3}$$

where

$$\sigma_{\text{avg}}^2 = \frac{\sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \sigma_4^2 + \sigma_5^2}{5} \tag{4}$$

In other words, these conditions ensure that blocks classified into the textural class are located in a region exhibiting a homogeneous high value variance.

3. Edge Criterion. If a block is classified as neither a smooth block nor a textural block and the bottom of the quadtree is reached, then the block is simply declared an edge block.

B. Adaptive Differential Pulse Code Modulator. In an ADPCM, an attempt is made to predict the block average to be encoded. Note that the DC coefficient is exactly m times the block average. The prediction is made using the already encoded block averages. Only the prediction error, i.e., the difference between the predicted and the current block average values, is quantized. The absolute quantized prediction error is then arithmetically encoded while the sign of each nonzero quantized prediction error is encoded separately in a single bit.

During the adaptive prediction process, different linear prediction rules are used to predict the current block average (only one rule per prediction is applied). These rules include second-order two-dimensional prediction rules, (e.g., $\frac{1}{4}A + \frac{3}{4}C$, $\frac{3}{4}A + \frac{1}{4}C$, $1\frac{1}{4}A - \frac{1}{4}C$, and $-\frac{1}{4}A + 1\frac{1}{4}C$), and third-order two-dimensional prediction rules (e.g., $A - \frac{1}{4}B + \frac{1}{4}C$, $\frac{1}{4}A - \frac{1}{4}B + C$, $A - \frac{1}{2}B + \frac{1}{2}C$, $\frac{1}{2}A - \frac{1}{2}B + C$, and $\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$), where A, B, and C are the neighboring block averages.

The choice of a prediction rule is based on encoded block averages local statistics, namely, the differences between the neighboring encoded block averages. Table II shows all the different cases of A, B, and C with all the possible relative distances between them; Table III shows the corresponding prediction rule, the prediction range, and the expected shape of this area for each case. Note that the terms "small" and "large" in Table II mean that the difference between the neighboring encoded block averages is less or greater, respectively, than a certain value v. In this work, the value of v is empirically determined so that the prediction rule usage is partitioned as evenly as possible (Section IIIB).

C. Quantizers.

1. ADPCM Quantizer. Typically, ADPCM prediction errors have a greatly reduced variance compared to the variance of the original average values. However, more bit rate reduction can be achieved by quantizing these prediction errors prior to encoding them. For this purpose, a simple scalar quantizer is designed. In this quantizer, each prediction error value is quantized using a simple scalar value defined by

$$\text{ADPCM quantization step} = \left\lfloor \frac{256}{K_{\text{ADPCM}}^{\text{class}}(QF)} \right\rfloor \tag{5}$$

Table II. All possible relative pixel values of A, B, and C.

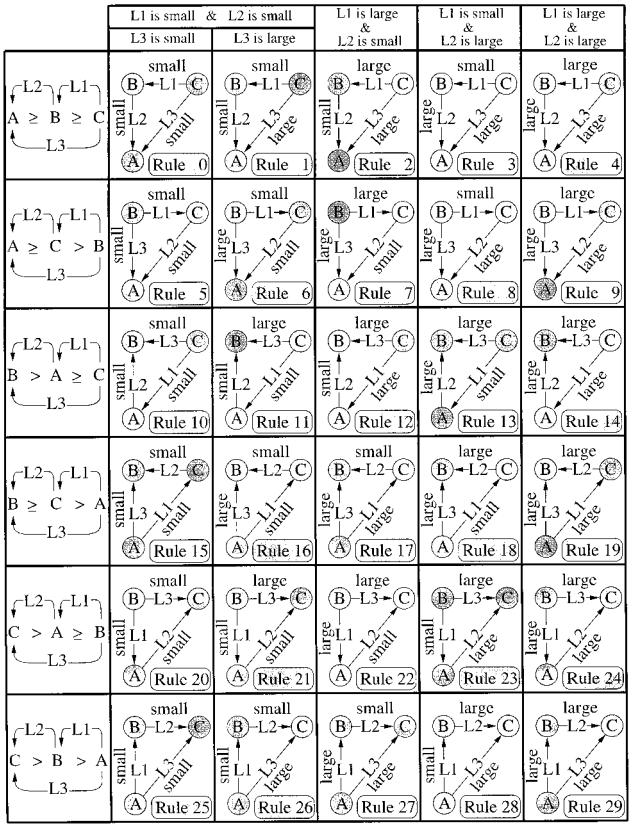

Table III. ADPCM prediction rules.

	Rule No.	Prediction Rule	Prediction Range	Expected Shape
$A \geq B \geq C$	0	$\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$	$\in[C,A]$	Flat
	1	$\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$	$\in (C, A)$	Flat
	2	$\frac{1}{2}A - \frac{1}{2}B + C$	$\in [C, B)$	Edge
	3	$A - \frac{1}{2}B + \frac{1}{2}C$	$\in (B, A]$	Edge
	4	$\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$	$\in (C, A)$	Strong edge/texture
$A \geq C > B$	5	$\frac{3}{4}A + \frac{1}{4}C$	$\in [C, A]$	Flat
	6	$A - \frac{1}{2}B + \frac{1}{2}C$	$> A$	Edge
	7	$1\frac{1}{4}A - \frac{1}{4}C$	$\geq A$	Edge/texture
	8	$A - \frac{1}{2}B + \frac{1}{2}C$	$> A$	Edge
	9	$A - \frac{1}{4}B + \frac{1}{4}C$	$> A$	Strong edge/texture
$B > A \geq C$	10	$\frac{1}{4}A + \frac{3}{4}C$	$\in [C, A]$	Flat
	11	$\frac{1}{2}A - \frac{1}{2}B + C$	$< C$	Edge
	12	$\frac{1}{2}A - \frac{1}{2}B + C$	$< C$	Edge
	13	$-\frac{1}{4}A + 1\frac{1}{4}C$	$\leq C$	Edge/texture
	14	$\frac{1}{4}A - \frac{1}{4}B + C$	$< C$	Strong edge/texture
$B \geq C > A$	15	$\frac{3}{4}A + \frac{1}{4}C$	$\in (A, C)$	Flat
	16	$A - \frac{1}{2}B + \frac{1}{2}C$	$< A$	Edge
	17	$A - \frac{1}{2}B + \frac{1}{2}C$	$\leq A$	Edge
	18	$1\frac{1}{4}A - \frac{1}{4}C$	$< A$	Edge/texture
	19	$A - \frac{1}{4}B + \frac{1}{4}C$	$< A$	Strong edge/texture
$C > A \geq B$	20	$\frac{1}{4}A + \frac{3}{4}C$	$\in (A, C)$	Flat
	21	$\frac{1}{2}A - \frac{1}{2}B + C$	$> C$	Edge
	22	$-\frac{1}{4}A + 1\frac{1}{4}C$	$> C$	Edge/texture
	23	$\frac{1}{2}A - \frac{1}{2}B + C$	$\geq C$	Edge
	24	$\frac{1}{4}A - \frac{1}{4}B + C$	$> C$	Strong edge/texture
$C > B > A$	25	$\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$	$\in (A, C)$	Flat
	26	$\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$	$\in (A, C)$	Flat
	27	$A - \frac{1}{2}B + \frac{1}{2}C$	$\in (A, B)$	Edge
	28	$\frac{1}{2}A - \frac{1}{2}B + C$	$\in (B, C)$	Edge
	29	$\frac{3}{4}A - \frac{1}{2}B + \frac{3}{4}C$	$\in (A, C)$	Strong edge/texture

where $K_{\mathrm{ADPCM}}^{\mathrm{class}}(QF)$ is a class-dependent, positive, nonzero integer function of the quality factor (QF), and "class" can be any of the three perceptual classes. The $K_{\mathrm{ADPCM}}^{\mathrm{class}}$ is used as a scaling factor to arrive at a compromise between the compression ratio and the quality of the reconstructed images. The range of $K_{\mathrm{ADPCM}}^{\mathrm{class}}$ is

$$2 \leq K_{\mathrm{ADPCM}}^{\mathrm{class}} \leq 256 \tag{6}$$

On the other hand, the $K_{\mathrm{ADPCM}}^{\mathrm{class}}$ domain is arbitrarily set between 1 and 256, where a maximum reconstruction quality with a minimum compression ratio is achieved when $QF = 256$, while a maximum compression ratio with a minimum reconstruction quality is achieved when $QF = 1$.

2. AC Quantizer. For most natural scene images, the majority of the AC coefficients have small magnitudes. Hence, these coefficients have the smallest impact on the reconstructed image quality. Consequently, they can be coarsely quantized or even discarded entirely with the trade-off of having a little image distortion.

The role of the AC quantizer is to eliminate selectively or quantize coarsely the AC coefficients that carry the least information. In this quantizer, the AC coefficient in row i and column j is quantized using a scalar value defined by

$$\text{AC quantization step} = \begin{cases} \left\lfloor \dfrac{q_{ij}}{K_{AC}^{\mathrm{class}}(QF)/256} \right\rfloor & \text{if } q_{ij} \times 256 > K_{AC}^{\mathrm{class}}(QF) \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

where $q_{ij} \in Q$, Q is an 8×8 normalization matrix and $K_{AC}^{\mathrm{class}}(QF)$ is a class-dependent, positive, nonzero integer function of the quality

factor QF while "class" is restricted only to either textural or edge class. The Q normalization matrix, shown in Equation (8) is adopted after the JPEG standard (Wallace, 1992). The role of this Q matrix, which was heuristically determined, is to weight the AC coefficients according to their perceptual importance.

$$Q = \begin{pmatrix} * & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix} \quad (8)$$

Similar to the $K_{\mathrm{ADPCM}}^{\mathrm{class}}$ the K_{AC}^{class} is used as a scaling factor to arrive at a compromise between the compression ratio and the quality of the reconstructed images. As the smallest and the largest element in Q are equal to 10 and 121, respectively, and the range of the AC coefficients is between $\pm(2^8 - 1) \times$ block_size/2, the range of K_{AC}^{class} will be

$$2 \leq K_{AC}^{\mathrm{class}} \leq 30976 \quad (9)$$

At $K_{AC}^{\mathrm{class}} = 2$, all of the AC coefficients will be quantized to zero, i.e., a given image block belongs to this class will be represented by its DC coefficient only. On the other hand, at $K_{AC}^{\mathrm{class}} = 30976$, all of the AC coefficients will be unquantized. Meanwhile, the K_{AC}^{class}

Figure 2. Original images.

Algorithm: *Non-texture-related-functions-identification*

Step 0: **Begin algorithm**:
Step 1: **Let** *functions* **be** { K_{AC}^{edge}, K_{ADPCM}^{edge}, K_{ADPCM}^{smooth}, T_8, T_{16}, and T_{32} }.
Step 2: **Let** *QF-values* **be** { 1, 8, 16, 32, 64, 96, 128, 160, 192, 224, 240, 248, and 255 }.
Step 3: **Repeat:**
Step 3.1: **For each** *function* **in** *functions* **do**:
Step 3.1.1: **For each** *QF-value* **in** *QF-values* **do**:
Step 3.1.1.1: **Let** *valid-outputs* **be** samples from all possible valid outputs for *function* (*QF-value*).
Step 3.1.1.2: **For each** *valid-output* **in** *valid-outputs* **do**:
Step 3.1.1.2.1: Compress the 8 images at quality-factor = *QF-value*, while assuming that *function* (*QF-value*) = *valid-output*.
Step 3.1.1.2.2: Calculate the average compression ratio over the 8 images.
Step 3.1.1.2.3: Decompress the 8 images.
Step 3.1.1.2.4: Calculate the average Root Mean Squared Error (*RMSE*) over the 8 images.
Step 3.1.1.3: **End for**.
Step 3.1.1.4: Generate *rate-distortion* (*QF-value*) curve over the 8 images.
Step 3.1.2: **End for**.
Step 3.1.3: Generate *envelope* (*function*) for all the *rate-distortion* (*QF-value*) curves generated in this iteration.
Step 3.1.4: **For each** *QF-value* **in** *QF-values* **do**:
Step 3.1.4.1: **Let** the current value of *function* (*QF-value*) **be** the value of *valid-output* corresponding to the tangent point between *envelope* (*function*) and *rate-distortion* (*QF-value*).
Step 3.1.5: **End for**.
Step 3.2: **End for**.
Step 4: **Until** no change is happened to any function during a one full iteration.
Step 5: **End algorithm**.

Figure 3. Non–texture-related functions' identification algorithm.

domain is arbitrarily set between 1 and 256, where a maximum reconstruction quality with a minimum compression ratio is achieved when $QF = 256$, while a maximum compression ratio with a minimum reconstruction quality is achieved when $QF = 1$.

III. PARAMETERS ADJUSTMENT

A. Thresholding and Quantization Functions. There is no doubt that the impact of the quantization functions on the ABC-SC performance is affected by the values of the thresholding parameters. The *T* thresholding parameters control the routing of the blocks to the different compression modules. In the same time, the impact of the *T* thresholding parameters on the ABC-SC performance is also affected by the definition of the *K* quantization functions, which control the allowed amount of distortions that might occur to the blocks of each class. Therefore, care should be taken when identifying these parameters and functions.

In this work, the parameters and functions' identification process is divided into two phases. In the first phase, all the nonsmooth blocks are dealt with as if they are edge blocks, i.e., only two classes of blocks are considered (smooth and edge classes). Consequently, only the non–texture-related parameters and functions (i.e., T_m, $K_{\text{ADPCM}}^{\text{smooth}}$, $K_{\text{ADPCM}}^{\text{edge}}$, and K_{AC}^{edge}) are identified in this phase. Then, in the second phase, the rest of the parameters and functions (i.e., the texture-related parameters functions, namely, T_{average}, $T_{\text{deviation}}$, $K_{\text{ADPCM}}^{\text{texture}}$, and K_{AC}^{texture}) are identified based on the value of the already identified non–texture-related parameters and functions.

1. Non–Texture-Related Functions. To add more adaptability to ABC-SC, each of the T_m thresholding parameters is considered a function of the *QF* rather than just a scalar value. The role of these T_m thresholding functions depends on the value of the *QF*. For high *QF* values, the role of these functions is to facilitate the conditions on blocks to be classified as nonsmooth blocks. Hence, the blocks gain less distortion. On the other hand, for low *QF* values, their role is to ease the conditions on blocks to be classified as smooth blocks. Hence, they can be compressed further.

Theoretically speaking, any decreasing function can be considered as a valid definition for any of the T_m thresholding functions. Also, any increasing function can be considered as a valid definition for any of the non–texture-related *K* quantization functions: namely, $K_{\text{ADPCM}}^{\text{smooth}}$, $K_{\text{ADPCM}}^{\text{edge}}$, and K_{AC}^{edge}. However, as the performance of ABC-SC strongly depends on the definition of these functions, they should be chosen carefully.

Since optimizing both of the *T* and the *K* functions simultaneously is not an easy optimization problem, we propose an iterative solution for identifying them sequentially. The basic idea is to adjust only one function at a time while fixing the rest of the functions, and repeat doing this adjustment in sequence until no change occurs during one full iteration.

During this identification process, each of these functions is assumed to be a continuous piecewise linear function between some predetermined *QF* values—namely, 1, 8, 16, 32, 64, 96, 128, 160, 192, 224, 240, 248, and 255. At $QF = 256$, all the T_m thresholding functions are set to zero so that all blocks are classified as edge blocks. In the same time, the K_{AC}^{edge} and $K_{\text{ADPCM}}^{\text{edge}}$ quantization functions are set to a large number. This allows all the AC and the DC coefficients to be preserved without quantization. Hence, the lowest reconstruction degradation is achieved.

Note that the piecewise linear assumption is a reasonable assumption, as each of these functions is either an increasing or a decreasing function. Also note that the domains of the piecewise segments are selected to be equal, except at low and high *QF* values, where the changing rate of the T_m thresholding functions and the *K*

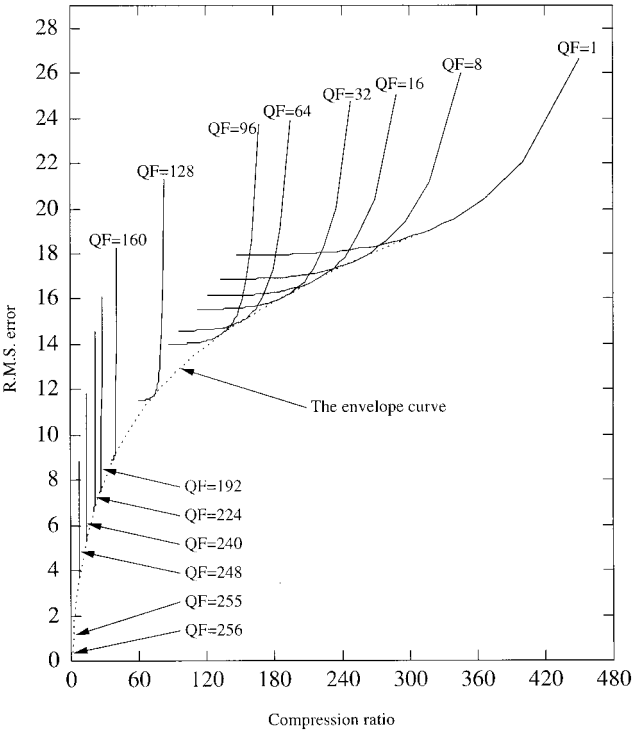

Figure 4. Example of an envelope curve for rate-distortion curves which are generated for $K_{\text{ADPCM}}^{\text{smooth}}$ during the functions' second identification iteration.

Table IV. Converged values of all non–texture-related functions.

QF	K_{AC}^{edge}	$K_{\text{ADPCM}}^{\text{edge}}$	$K_{\text{ADPCM}}^{\text{smooth}}$	T_8	T_{16}	T_{32}
1	10	12	12	4000	700	58
8	12	12	13	2600	600	57
16	13	12	16	2000	500	56
32	14	13	16	1700	400	50
64	16	15	19	1400	240	36
96	18	16	20	1216	176	26
128	32	26	32	550	112	16
160	64	36	48	224	48	12
192	96	52	64	128	36	8
224	128	68	92	80	20	4
240	208	88	128	40	12	2
248	512	160	208	20	6	1
255	4096	256	256	0	0	0

quantization functions are expected to be higher than the changing rate for the rest of the QF values. With this assumption, our goal now is to determine the output values of each function at these predetermined QF values.

During this phase, determining the function's output values is based on eight different 512 × 512 images, shown in Figure 2(a). These eight images are selected to represent different image classes including, edge images (e.g., the "Monarch" and "Boats" images); textural images (e.g., the "Rocks" and "Barbara" images); smooth images (e.g., the "Zelda" and "Peppers" images); and natural scene images (e.g., the "Goldhill" and "F16" images).

In this phase, the determination of the output values of a given function is achieved by considering all the valid output values of this function. Then, for each QF value, a rate-distortion curve, based on compressing the eight images, is generated. Note that each point on this curve corresponds to one of the valid output values. Then, an envelope curve for all these rate-distortion curves, which maximizes the compression ratio and, at the same time, minimizes the root mean squared error, is generated. The envelope curve may be approximated by a continuous piecewise linear curve. The first segment of this curve is the line tangent to the rate-distortion curve corresponding to $QF = 255$ starting from the rate-distortion point corresponding to $QF = 256$. The second segment of this curve is the line tangent to the next rate-distortion curve (i.e., the curve corresponding to $QF = 248$) starting from the current tangent point (which is locating on the curve corresponding to $QF = 255$), and so on. Finally, the output values corresponding to the tangent points between the envelope and each rate-distortion curve are determined and assigned to the value of the function at the corresponding QF values. This procedure is repeated for all the non–texture-related functions until no change occurs to any of them over a one full iteration. Note that after the adjustment of each function, the performance of ABC-SC is either improved from a rate-distortion point of view, or at worst it remains the same if the unchanged function case is encountered. Hence, the procedure convergence is guaranteed. Figures 3 and 4 show the algorithmic steps of the non–texture-related functions identification procedure, and an example of an envelope curve for rate-distortion curves which are generated for $K_{\text{ADPCM}}^{\text{smooth}}$ during the functions' second identification iteration, respectively.

Note that this identification procedure is insensitive to initial conditions; i.e., any increasing function can be considered as a valid initial definition for any of the T_m thresholding functions, and any decreasing function can be considered as a valid initial definition for any of the K quantization functions. In this work, the initial definitions of $K_{\text{ADPCM}}^{\text{smooth}}$ and $K_{\text{ADPCM}}^{\text{edge}}$ quantization functions are selected to be constant functions, as given by Equations (10) and (11), respectively.

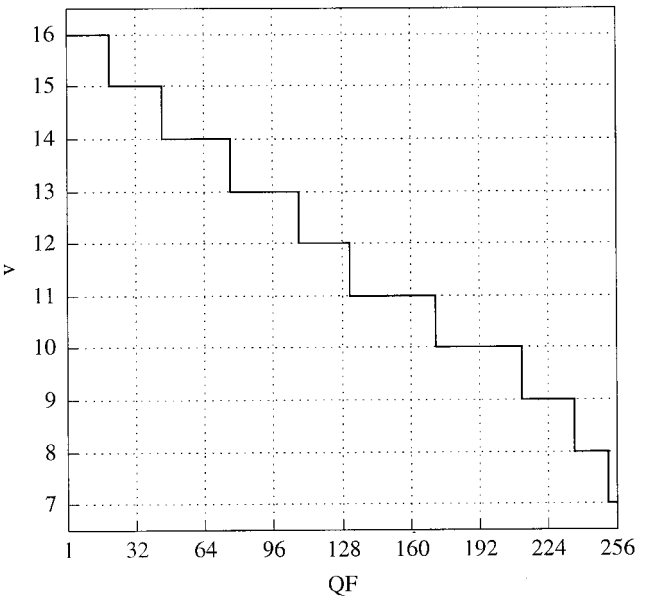

Figure 5. Values of v as a function of QF.

$$K_{\text{ADPCM}}^{\text{smooth}} = 256 \tag{10}$$

$$K_{\text{ADPCM}}^{\text{edge}} = 256 \tag{11}$$

Meanwhile, the initial definitions of T_{32}, T_{16}, and T_8 functions are selected to be continuous piecewise linear decreasing functions, as given by Equations (12)–(14), respectively.

$$T_{32} = \begin{cases} \lfloor -3/8 \times QF + 60 \rfloor & \text{if} \quad 1 \le QF < 64 \\ \lfloor -2/8 \times QF + 52 \rfloor & \text{if} \quad 64 \le QF < 160 \\ \lfloor -1/8 \times QF + 32 \rfloor & \text{if} \quad 160 \le QF < 255 \end{cases} \tag{12}$$

Figure 6. Original "Lena" image.

$$T_{16} = \begin{cases} \lfloor -9/2 \times QF + 528 \rfloor & \text{if} \quad 1 \le QF < 64 \\ \lfloor -4/2 \times QF + 368 \rfloor & \text{if} \quad 64 \le QF < 160 \\ \lfloor -1/2 \times QF + 128 \rfloor & \text{if} \quad 160 \le QF < 255 \end{cases} \tag{13}$$

$$T_{8} = \begin{cases} \lfloor -54 \times QF + 5184 \rfloor & \text{if} \quad 1 \le QF < 64 \\ \lfloor -16 \times QF + 2752 \rfloor & \text{if} \quad 64 \le QF < 160 \\ \lfloor -2 \times QF + 512 \rfloor & \text{if} \quad 160 \le QF < 255 \end{cases} \tag{14}$$

After the second full iteration, all functions are converged to their final definitions. Table IV shows the converged values of each function.

2. Texture-Related Functions. In general, the objective of the T and K parameter and function identification process is to maximize the compression ratios and at the same time minimize the reconstruction errors. However, the texture-related parameters and functions (i.e., T_{average}, $T_{\text{deviation}}$, $K_{\text{ADPCM}}^{\text{texture}}$, and K_{AC}^{texture}) might have different objectives according to the application. For example, in some image compression applications, more reconstruction errors in the textural regions might be allowed to achieve higher compression ratios. The reason is that textural regions usually convey a low amount of information to the viewer. However, there are image-processing applications where the textural regions might be of interest. Hence, reducing their reconstruction errors may be more significant to the viewer than achieving higher compression ratios.

ABC-SC deals with these different situations by allowing users to define the quality of the reconstructed textural regions relative to the quality of the reconstructed edge regions [texture–quality ratio (TQR)]. If this ratio is <1, this means that the user is willing to sacrifice some of the reconstruction quality in the textural regions in order to achieve higher compression ratios. On the other hand, if this ratio is >1, this means that the user is interested in the textural regions. Hence, less degradation is allowed in these regions even if the compression ratio is decreased. Finally, if this ratio $= 1$, this means that both of the edge and the textural regions have the same

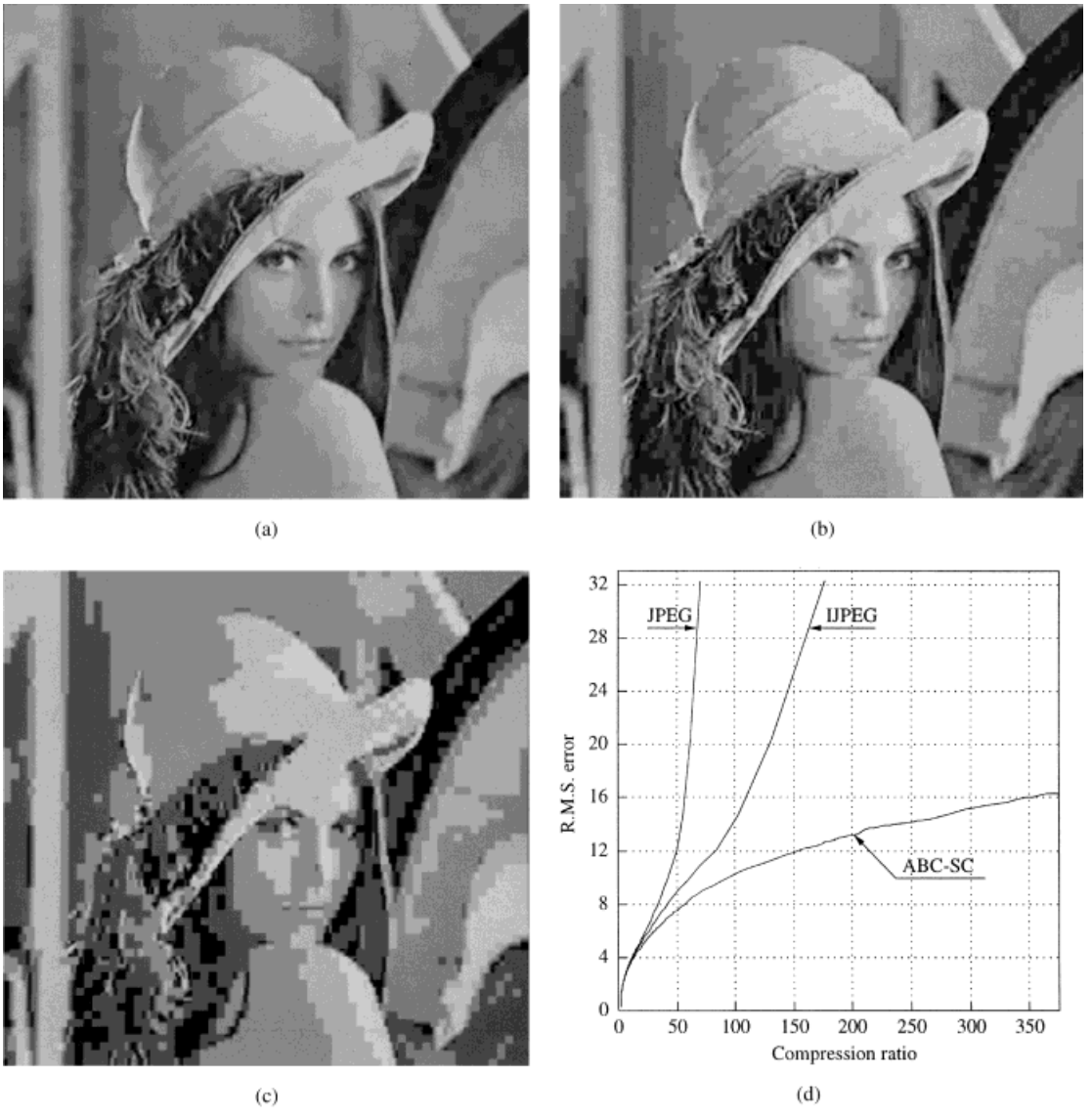

Figure 7. ABC-SC, IJPEG, and JPEG compression results for the "Lena" image.

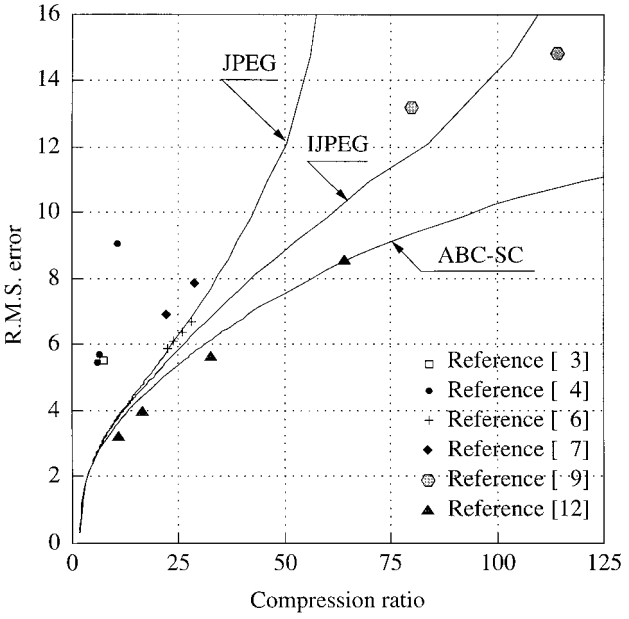

Figure 8. Comparison of rate distortion results between ABC-SC, IJPEG, JPEG, and some other recent segmentation-based encoding techniques for the "Lena" image.

interest to the user and there is no need to differentiate between them (two-class case), i.e., as done in the first phase.

The main difference between edge and textural blocks is the level of activities within the block and not in its average (i.e., the AC coefficients and not the DC coefficient). This suggests that the texture-related K quantization functions, $K_{\mathrm{ADPCM}}^{\mathrm{texture}}$ and $K_{AC}^{\mathrm{texture}}$, may be defined as follows.

$$K_{\mathrm{ADPCM}}^{\mathrm{texture}} = K_{\mathrm{ADPCM}}^{\mathrm{edge}} \tag{15}$$

$$K_{AC}^{\mathrm{texture}} = \mathrm{TQR} \times K_{AC}^{\mathrm{edge}} \tag{16}$$

The values of T_{average} and $T_{\mathrm{deviation}}$ have been determined experimentally by visually examining many textural and nontextural blocks in the eight images. Based on these examinations, the following values are suggested.

$$T_{\mathrm{average}} = 400 \tag{17}$$

$$T_{\mathrm{deviation}} = 0.96 \tag{18}$$

According to experimentations, we found that these thresholding values are quite successful in separating textural blocks from edge blocks.

B. Neighboring Block Averages Parameter. To achieve a maximum benefit from all prediction rules, it is required to choose the value of v so that all these prediction rules are used evenly. To do so empirically, the eight images of Figure 2(a) are reconsidered. This time, these images are used for studying the effect of v on partitioning the usage of the prediction rules. This can be done by compressing these images and counting the number of times each prediction rule is used. Since the variance of these counts gives an indication about how even the use of the prediction rules is (the higher/lower the variance value, the less/more even use of the prediction rules is), the required v value is chosen to be the value corresponding to the minimum variance for each QF value. Figure 5 shows the chosen v values as a function of QF.

IV. RESULTS

A. Performance Metrics. Since ABC-SC is a lossy technique, its reconstructed image may have some loss of information which may be visually useful. To assess the loss of fidelity in reconstructed images, a measurement of distortion should be used. As most decompressed images ultimately are viewed by human beings, it is appropriate to measure their qualities by subjective evaluations of human observers. Unfortunately, a simple, convenient, subjective evaluating mechanism does not exist.

As alternative measures, root mean squared error (RMSE) and peak signal-to-noise-ratio (PSNR),

$$\mathrm{RMSE} = \sqrt{\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(\hat{f}(x,y) - f(x,y))^2} \tag{19}$$

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255}{\mathrm{RMSE}}\right)^2 dB \tag{20}$$

are usually used to accomplish this task, where $f(x, y)$ and $\hat{f}(x, y)$ represent the original and reconstructed images, respectively, whereas M and N represent the image height and width, respectively. During this work, the RMSE measure is used to assess the reconstructed images degradation.

Finally, the compression ratio (CR) is used to assess the compression level. In this work, the CRs are calculated from the actual size of the compressed files, not entropy estimates, as follows:

$$\mathrm{CR} = \frac{\text{Image width} \times \text{Image height}}{\text{Actual compressed file size}} \tag{21}$$

B. Experimental Setup. The validity of ABC-SC is demonstrated by compressing, then decompressing several images and comparing the input with the reconstructed images. Also other existing compression techniques are used in the comparison, including the Joint Photographic Experts Group (JPEG) (Wallace, 1992) and set partitioning in hierarchical trees (SPIHT) (Said and Pearlman, 1996). JPEG is a widely accepted industrial standard for continuous-tone natural scene image compression, whereas SPIHT is believed to be the state-of-the-art still-image lossy-compression method. While JPEG is a fixed-block-size, DCT-based compression technique, SPIHT is an embedded wavelet-based compression technique which is based on the embedded zerotree wavelet approach (EZW) (Shapiro, 1993).

The JPEG version that is used in this work is obtained via the Internet from the Independent JPEG Group; the source code is available by anonymous FTP from ftp.uu.net:/graphics/jpeg/jpegsrc.v6.tar.gz. This JPEG version has two modes of operation: the baseline JPEG mode and the improved JPEG (IJPEG) mode. The only difference between these two modes of operations is that the

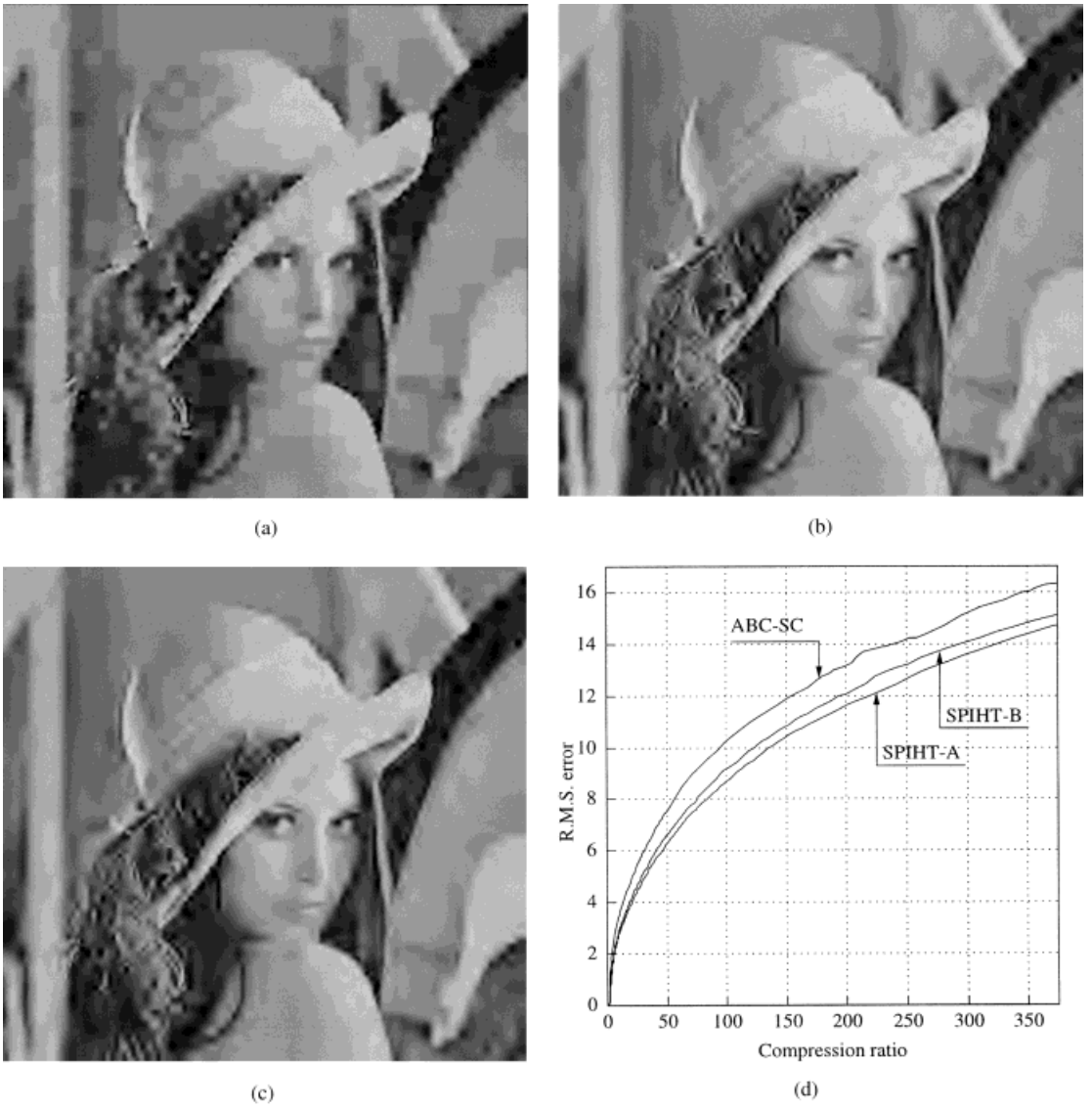

Figure 9. ABC-SC, SPIHT-A, and SPIHT-B compression results for the "Lena" image.

IJPEG employs a multiple-pass Huffman encoder to determine an optimal symbol code.

Similarly, the SPIHT version that is used in this work is obtained via the Internet, where the source code is available by anonymous FTP from ipl.rpi.edu:/pub/EW_Code/codetree.tar.gz. SPIHT exists in slow and fast versions. The slow version (SPIHT-A) uses an arithmetic encoder to improve compression and the fast version (SPIHT-B) produces a binary-uncoded bitstream. Hence, the former version has the ability to compress images slightly more than that in the latter version.

During this work, ABC-SC is extensively tested and compared with JPEG, IJPEG, SPIHT-A, and SPIHT-B. (In most image compression literature, researchers compare their compression results with JPEG results, but not with IJPEG results, even though the latter outperforms the former.) However, bear in mind that while testing against JPEG is fair enough (since both of ABC-SC and JPEG are DCT-based techniques), testing against SPIHT (the state of the art) is a kind of an ultimate comparison. The 512 × 512 "Lena" image shown in Figure 6 is chosen to demonstrate the results. This 8-bpp gray-level version of the "Lena" image is the Y luminance component of the original 24-bpp red, green, and blue (RGB) "Lena" image, where the Y pixel values are obtained from the RGB pixel values using the following linear transform.

$$Y(i, j) = 0.299 \times R(i, j) + 0.587 \times G(i, j) + 0.114 \times B(i, j)$$

The other way of getting an 8-bpp gray-level version from a 24-bpp RGB image is by using only the G component. However, the G component is darker and has slightly less fidelity than the Y component. Note that the results reported here were obtained by setting TQR = 1.

C. ABC-SC Versus JPEG/IJPEG. Figure 7(a–c) shows the decompressed "Lena" images using ABC-SC at $QF = 147$, IJPEG at

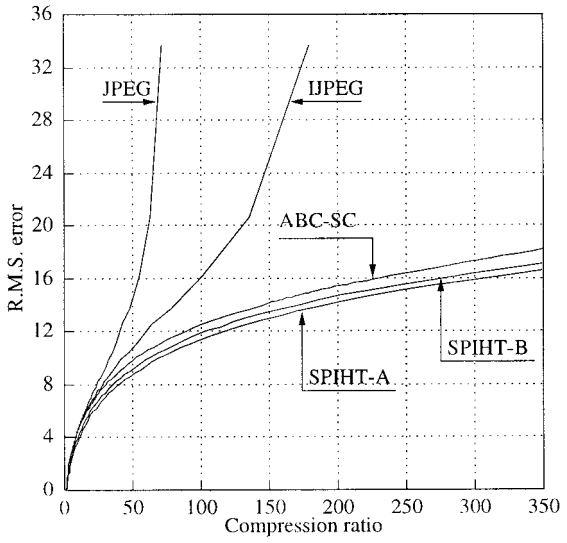

Figure 10. Rate-distortion curves for eight different images, using ABC-SC, JPEG, IJPEG, SPIHT-A, and SPIHT-B.

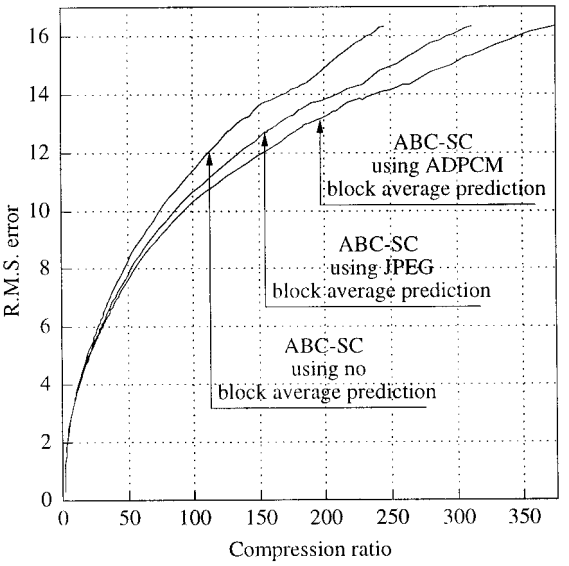

Figure 11. ADPCM effect for the "Lena" image.

$QF = 6$, and JPEG at $QF = 2$, respectively. While the compression ratios of these three reconstructed images are almost the same (62.47:1, 59.89:1, and 62.22:1, respectively), ABC-SC has the smallest RMSE reconstruction error (8.43, 9.86, and 20.42, respectively). From the subjective quality point of view, the ABC-SC reconstructed image is much better than the IJPEG reconstructed image. This appears especially at smooth areas, where the blocking effect is almost removed completely with ABC-SC. The ABC-SC reconstructed image is also far much better than the JPEG reconstructed image.

Figure 7(d) shows the rate-distortion curves for the "Lena" image using ABC-SC, IJPEG, and JPEG. As the curves indicate, ABC-SC outperforms both JPEG and IJPEG for all compression ratios under consideration. Moreover, ABC-SC can achieve high compression ratios—with a reasonable RMSE reconstruction error—which cannot be achieved by either JPEG or IJPEG.

D. ABC-SC Versus a Number of Other Segmentation-Based Compression Techniques. Figure 8 shows the rate-distortion performance of ABC-SC, IJPEG, JPEG, and a number of other segmentation-based compression techniques—which have been reported in the recent compression literature (Section I). As seen, ABC-SC outperforms all these techniques except for the one which was reported in Ran and Farvardin. Nevertheless, the improvement in the RMSE reconstruction error is minor (<0.5) and takes place in the perceptually lossless range. This means that it is really hard to see a difference between the two reconstructed images. On the other hand, from the practical point of view, the technique in Ran and Farvardin is practically unacceptable, since it takes about 23 min of CPU time to decompose a 256×256 image. The ABC-SC average execution time on a comparable task is <1 s (see Section IVI).

E. ABC-SC Versus SPIHT. Figure 9(a–c) shows the decompressed "Lena" images using ABC-SC at $QF = 32$, SPIHT-A, and SPIHT-B, respectively. While the compression ratio of any of these three reconstructed images is 235.11:1, the RMSE reconstruction errors are 13.97, 12.29, and 12.95, respectively. This does not represent a major difference. From the subjective quality point of view, while ABC-SC start suffering from the blocking artifact, both SPIHT-A and SPIHT-B are suffering from a ringing effect—which is a well-known artifact in subband encoders in general, especially near strong edges. However, in ABC-SC, there is always a chance to restore the blocking artifact—since the exact deformity locations are known. At the same time, in SPIHT-A and SPIHT-B there is no chance to restore the ringing effect.

Figure 9(d) shows the rate-distortion curves for the "Lena" image using ABC-SC, SPIHT-A, and SPIHT-B. The figure shows that the ABC-SC rate-distortion curve is following both SPIHT-A and

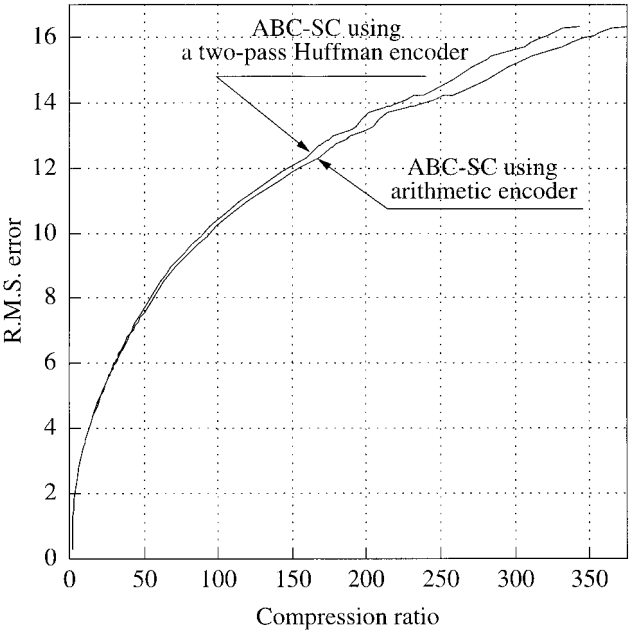

Figure 12. Effect of the lossless encoder on the performance of ABC-SC for the "Lena" image.

Table V. Compression/decompression time for the "Lena" image at different compression ratios.

	Compression Time (s)	Decompression Time (s)	Enhancement Time (s)
ABC-SC at $QF = 147$, i.e., $CR = 62.47{:}1$	0.53	0.47	0.27
IJPEG at $QF = 6$, i.e., $CR = 59.89{:}1$	0.18	0.07	
JPEG at $QF = 2$, i.e., $CR = 62.22{:}1$	0.14	0.06	
SPIHT-A at $CR = 62.47{:}1$	0.58	0.60	
SPIHT-B at $CR = 62.47{:}1$	0.51	0.53	
ABC-SC at $QF = 32$, i.e., $CR = 235.11{:}1$	0.38	0.35	0.30
SPIHT-A at $CR = 235.11{:}1$	0.53	0.53	
SPIHT-B at $CR = 235.11{:}1$	0.49	0.52	

SPIHT-B rate-distortion curves. Although ABC-SC maintains a minor increase in the RMSE reconstruction error, it does not really make a major difference in the subjective quality of the reconstructed images.

F. ABC-SC Versus Both JPEG/IJPEG and SPIHT. Figure 10 shows the rate-distortion curves for eight test images—different from the images used for parameter adjustment in Section III—using ABC-SC, JPEG, IJPEG, SPIHT-A, and SPIHT-B. These eight images, shown in Figure 2(b), are "Man," "Woman," "Playgirl," "Couple," "Crowd," "Hotel," "Eye," and "Bridge." The figure confirms the results presented in Figures 7(d) and 9(d).

G. ADPCM Effect. Figure 11 shows the rate-distortion curves for the "Lena" image using three different versions of ABC-SC. The only difference between these three versions is the DC coefficient predictor. In the first version, the ADPCM predictor introduced in Section IIB is used. In the second version, the JPEG/IJPEG DC coefficient predictor is used, where the predicted value is always the previous DC coefficient value of the block to the left of the current block. Finally, in the third version there is no prediction at all. Figure 11 shows the superiority of the ADPCM predictor version over the two other versions. We believe that this superiority justifies the additional complexity introduced by the ADPCM predictor, which is an insignificant extra complexity after all.

H. Lossless Encoder Effect. As an alternative to the arithmetic encoder, we also considered the Huffman encoder. Figure 12 shows the rate-distortion curves for the "Lena" image using two different versions of ABC-SC. The only difference between these two versions is the lossless encoder used. While the first version uses an arithmetic encoder, a two-pass Huffman encoder is used in the second version. The figure shows that a minor rate-distortion improvement is achieved when the arithmetic encoder is used.

I. Execution Time. Table V shows the exact amount of time (in seconds) required to execute ABC-SC, IJPEG, JPEG, SPIHT-A, and SPIHT-B on a Sun-Ultra-1 computer for the "Lena" image at different compression ratios. Table VI shows the average amount of time (in seconds) required to execute ABC-SC, SPIHT-A, SPIHT-B, JPEG, and IJPEG on a Sun-Ultra-1 computer for the "Lena" image. These averages are calculated based on actual runs over all QF values in the case of ABC-SC, JPEG, and IJPEG, and equivalent compression ratios in the case of SPIHT-A and SPIHT-B. From Table VI, we can conclude the following:

1. On average, the ABC-SC compression/decompression time falls between the SPIHT-A and SPIHT-B compression/decompression times.
2. The enhancement time is almost half the decompression time. However, this time always can be waived, especially at low compression ratios, where the blocking artifact is unnoticeable.
3. The execution time of JPEG/IJPEG is, dramatically, less by an order of magnitude than the execution time of the rest of the techniques.

Finally, it is worth mentioning that we have not made a major effort to reduce the complexity or optimize the implementation of ABC-SC. In fact, we believe that ABC-SC could be further optimized and we anticipate that its execution time could be reduced to a value in the vicinity of JPEG/IJPEG execution time, since both ABC-SC and JPEG/IJPEG are DCT-based techniques. However, further investigation is needed to support this belief.

V. CONCLUSIONS

In this work, a new adaptive compression technique is proposed. Adaptability has been incorporated into many aspects of this technique, including adaptive block size determination, adaptive multiblock-encoding techniques, adaptive block-average prediction, adaptive arithmetic encoding, and adaptive postprocessing. The proposed compression technique also exploits one of the HVS properties, which is recognizing images by their regions, to get high

Table VI. Average compression/decompression time (in seconds) for the "Lena" image.

Average Execution Time	ABC-SC	SPIHT-A	SPIHT-B	JPEG	IJPEG
Compression	0.9945	1.253	0.763	0.109	0.122
Decompression	0.8305	1.051	0.532	0.084	0.081
Enhancement	0.414				

compression ratios. Based on extensive test results and comparisons with other existing compression techniques, the following conclusions can be made:

1. ABC-SC produces good-quality reconstructed images at both high and low bit rates.
2. It also produces a good rate-distortion performance as well as good subjective-quality reconstructed images.
3. Although ABC-SC and the JPEG/IJPEG techniques are both DCT-based techniques, the former technique consistently exhibits a significantly better performance than the latter, especially at high compression ratios.
4. ABC-SC moves block-based compression beyond the limits of JPEG/IJPEG—usually, the maximum compression ratio that can be achieved with ABC-SC is at least twice that of IJPEG and five times that of JPEG, with even less objective and subjective degradations.
5. In all of the test cases, the ABC-SC performance is comparable to that of the wavelet compression technique from both the rate-distortion and the subjective-quality points of view.
6. The execution time of ABC-SC is somewhere between the execution time of SPIHT-A and SPIHT-B, the slow and fast versions of SPIHT. However, ABC-SC has a potential to be further optimized so that its execution time can be reduced by an order of magnitude or so.
7. ABC-SC provides users with the ability to give certain classes of image segments more importance than others, so that they are less affected by the compression process.
8. ABC-SC provides a good alternative to wavelet-based compression techniques, especially when adaptability to image content is of interest.

REFERENCES

C. Chen, Adaptive transform coding via quadtree-based variable blocksize DCT, IEEE Int Conf Acoustics Speech Signal Process (ICASSP'89), vol 3, May 1989, pp. 1854–1857.

T. Cornsweet, Visual perception, Academic Press, New York, 1971.

E. Delp and R. Mitchell, Image compression using block truncation coding, IEEE Trans Commun 27 (1979), 1335–1342.

M. El-Sakka, Adaptive digital image compression based on segmentation and block classification, Ph.D. Dissertation, Systems Design Engineering, University of Waterloo, Waterloo, Ontario, Canada, 1997.

M. El-Sakka and M. Kamel, A segmentation criterion for digital image compression, IEEE Int Conf Acoustics Speech Signal Process (ICASSP'95) (1995), 2551–2554.

H. Freeman, On the encoding of arbitrary geometric configuration, IRE Trans Electronic Comput EC-10 (1961), 260–268.

J. Gimlett, Use of "activity classes" in adaptive transform image coding, IEEE Trans Commun COM-23 (1975), 785–786.

S. Golomb, Run-length encodings, IEEE Trans Inform Theory IT-12 (1966), 399–401.

M. Kunt, M. Benard, and R. Leonardi, Recent results in high-compression image coding, IEEE Trans Circuits Syst CAS-34 (1987), 1306–1336.

M. Kunt, A. Ikonomopoulos, and M. Kocher, Second-generation image-coding, Proc IEEE 73 (1985), 549–574.

M. Lee and G. Crebbin, Classified vector quantisation with variable block-size DCT models, IEEE Proc Vis Image Signal Process 141 (1994), 39–48.

P. Nasiopoulos, R. Ward, and D. Morse, Adaptive compression coding, IEEE Trans Commun 39 (1991), 1245–1254.

D. Neuhoff and K. Castor, A rate and distortion analysis of chain codes for line drawings, IEEE Trans Inform Theory IT-31 (1985), 53–67.

H. Radha, M. Vetterli, and R. Leonardi, Image compression using binary space partitioning trees, IEEE Trans Image Process 5 (1996), 1610–1624.

X. Ran and N. Farvardin, A perceptually motivated three-component image model—Part II: Applications to image compression, IEEE Trans Image Process 4 (1995a), 430–447.

X. Ran and N. Farvardin, A perceptually motivated three-component image model—Part I: Description of the model, IEEE Trans Image Process 4 (1995b), 401–415.

A. Said and W. Pearlman, A new, fast, and efficient image codec based on set partitioning in hierarchical tree, IEEE Trans Circuits Syst Video Technol 6 (1996), 243–250.

J. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, IEEE Trans Signal Process 41 (1993), 3445–3462.

J. Vaisey and A. Gersho, Image compression with variable block size segmentation, IEEE Trans Signal Process 40 (1992), 2040–2060.

G.K. Wallace, The JPEG Still Picture Compression Standard, IEEE Trans Consumer Electronics 38 (1992), xviii–xxxiv.

X. Wu, Image coding by adaptive tree-structured segmentation, IEEE Trans Inform Theory IT-38 (1992), 1755–1767.