Transform-domain Postprocessing of DCT-coded Images

Chung-Nan Tien and Hsueh-Ming Hang

Dept. of Electronics Engineering
National Chiao-Tung University
Hsin-Chu 300, TAIWAN, ROC
Email: hmhang@cc.nctu.edu.tw

ABSTRACT

Data compression algorithms are developed to transmit massive image data under limited channel capacity. When a channel rate is not sufficient to transmit good quality compressed images, a degraded image after compression is reconstructed at the decoder. In this situation, a postprocessor can be used to improve the receiver image quality. Ideally, the objective of postprocessing is to restore the original pictures from the received distorted pictures. However, when the received pictures are heavily distorted, there may not exist enough information to restore the original images. Then, what a postprocessor can do is to reduce the subjective artifact rather than to minimize the differences between the received and the original images. In this paper, we propose two postprocessing techniques, namely, error pattern compensation and inter-block transform coefficient adjustment. Since Discrete Cosine Transform (DCT) coding is widely adopted by the international image transmission standards, our postprocessing schemes are proposed in the DCT domain. When the above schemes are applied to highly distorted images, quite noticeable subjective improvement can be observed.

1. INTRODUCTION

When channel rate is not sufficient to transmit good quality compressed images such as difficult pictures at low bit rate transmission, postprocessing shown in Fig.1 is one of the few means that can improve the decoded picture quality. Most existing postprocessing algorithms are performed in the spatial domain, for example, Ramamurthi [1]. In this approach, image blocks are classified according to their local characteristics into edge blocks and shade blocks. Coding errors associated with different types of image blocks are reduced by adaptive filtering with parameters designed to cope with image local characteristics. Since a typical Discrete Cosine Transform (DCT) image decoder would receive quantized coefficients, the decoder structure may become simpler if our postprocessing is operating in the transform domain rather than in the spatial domain. Two methods are proposed to reduce quantization errors in this paper: error pattern compensation, and inter-block transform coefficient adjustment (IBTCA).

Our basic idea behind the error pattern method is that there exists a strong correlation between the quantized DCT coefficient patterns and the coding error patterns, particularly in the high contrast areas such as edges. This is largely due to the rough truncation of DCT coefficients; only a few low frequency coefficients are retained when the chosen quantization step size is high. This is similar to the so-called Gibbs phenomenon in Fourier series approximation. Since the coefficient truncation errors are particularly noticeable in the high contrast areas, we can improve the decoded picture quality by identifying those areas and by compensating the truncated coefficients.

As for the smooth image areas, a heuristic inter-block transform coefficient adjustment scheme is proposed. It is designed based upon the observation that there exist noticeable distortions on the block boundaries. This is the well-known *blocking effect* due to non-overlapped block coding. A typical postprocessor that reduces such effects in spatial domain is a low pass filter. By blurring the discontinued block boundaries it lowers the blocking effects for human observers. In this paper, another technique is proposed for reducing blocking effects, namely, inter-block transform coefficient adjustment (IBTCA). It is operating in the transform domain. It modifies certain selected DCT coefficient based upon the adjacent block information. Hence, unlike the spatial post-filters that only affect the distorted boundaries, this scheme modifies the entire block with a simple implementation.

0-8194-1369-0/93/$6.00

In Section 2, the basic JPEG coding scheme, a typical DCT compression algorithm, is briefly reviewed. In Section 3, an error pattern compensation approach for postprocessing is developed. The inter-block transform coefficient adjustment scheme is described in Section 4. Finally, a summary of our work and a brief discussion on future directions are presented in Section 5.

2. JPEG-LIKE CODING SYSTEM

Joint Photographic Experts Group (JPEG), a working group of ISO/CCITT, developed an international standard for compressing still images [2]. The baseline coding algorithm, which is the most popular image coding algorithm today, is briefly described below. Each input image component is first partitioned into 8×8 pixel blocks. Each block is then converted by DCT into a block of 64 transform coefficients. These coefficients are quantized by a set of uniform scalar quantizers. The step sizes of these quantizers are decided by a matrix called *quantization table*. The indices of the quantized DCT coefficients are then losslessly coded into variable length codes. At the decoder, the string of variable length codes is decoded into the corresponding quantized DCT coefficients. The decoded image is reconstructed by performing inverse DCT on the quantized DCT coefficients.

The 2-D forward and inverse DCTs in JPEG standard are defined by:

$$S_{vu} = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} P_{yx} \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16}, \text{ and} \tag{1}$$

$$P_{yx} = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} S_{vu} C_u C_v \cos \frac{(2v+1)x\pi}{16} \cos \frac{(2u+1)y\pi}{16}, \tag{2}$$

$$\text{where} \qquad C_u, C_v = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } u, v = 0 \\ 1 & \text{otherwise.} \end{cases} \tag{3}$$

After forward DCT, each of the 64 DCT coefficients is divided by the corresponding quantizer step size and is rounded to the nearest integer to produce the quantized coefficient. The quantizer step size is derived by the quantization table T_{uv} and a quality control parameter Q. The quantized coefficient I_{uv} is produced by quantizing the coefficient S_{uv}. The quantization process is given by the following formula,

$$I_{uv} = round(\frac{S_{uv} \times 8}{T_{uv} \times Q}), \tag{4}$$

and the dequantization equation is:

$$R_{uv} = \frac{I_{uv} \times T_{uv} \times Q}{8}. \tag{5}$$

The effect of our postprocessing depends strongly on the value of Q. Hence, the *quality control parameter* Q plays an important role in the following algorithm descriptions.

3. ERROR PATTERN COMPENSATION

The goal of postprocessing is to reduce the distortion in the decoded image. To archive this goal, we are looking for the relationship between quantization errors (including errors due to coefficients truncation) and coded pictures. This relationship can be observed from the error image (Fig. 4), i.e., the differences between the original image (Fig. 2) and the decoded image (Fig. 3). In Fig. 4, the large-magnitude errors with line structures locate mostly in the high contrast image areas and thus they are strongly correlated to the local structures of the coded (or the original) image. On the other hand, quantization errors in the smooth areas are rather random and thus have little correlation with the structures of the original pictures. The line structure of quantization errors is mainly contributed to the truncation of transform coefficients, similar to the Gibbs phenomenon in Fourier series approximation.

3.1 Error Pattern Estimation and Compensation

Error patterns are the quantization errors with regular structures produced in the coding process. They are strongly correlated to the quantized coefficients. To illustrate our algorithm, a simplified overall encoding and decoding scheme is shown in Fig. 5. E is the error pattern, the differences between the original and the reconstructed images. If error

pattern can be estimated from the quantized coefficients precisely, then the distortion can be significantly reduced. An intuitive way of associating the quantized coefficient patterns with the corresponding error patterns is to classify the coefficient patterns into clusters in which each cluster represents patterns of similar characteristics. A simple clustering algorithm, LBG algorithm [3], is adopted.

Our pattern classification is done in the transform domain. Since the shape information of an image block is mostly represented by its AC coefficients. DC values are not included in the clustering process. The number of clusters is preselected. The DCT coefficients after quantization, $\tilde{C}$, is used as training vectors. A set of representatives of quantized coefficient patterns is then obtained. In total, nine pictures are used as the training set, each of size 720×480 and 256 gray level per pixel.

The error block representatives are generated by averaging all the error patterns associated with a quantized coefficient cluster. For each quantized image block $\tilde{C}_i$, it has an associated error block E_i produced in quantization procedure. Assuming that $\tilde{C}_i$ belongs to the jth cluster in our cluster analysis; in other words, $\tilde{C}_i$ is represented by QCB_j, the jth quantized coefficient representative. Then, the error pattern representative EB_j associated with QCB_j is calculated by averaging all the error blocks associated with the jth cluster. The training process is described below.

Step 1 For each image block S_i, compute E_i — quantized error pattern,

$$E_i = C_i - \tilde{C}_i \ . \tag{6}$$

Step 2 Among all quantized coefficient representatives, $QCB_k, k = 0, 1, ..., NR-1$, (NR is the number of clusters), find the index j such that

$$||\tilde{C}_i - QCB_j||^2 \quad is \quad minimum \tag{7}$$

Step 3 Add error patterns E_i to the corresponding accumulated error pattern GE_j

$$\begin{aligned} GE_j &= GE_j + E_i \\ counter_j &= counter_j + 1 \end{aligned} \tag{8}$$

Step 4 Repeat Step 1 to Step 3 until all the training blocks are processed.

Step 5 Compute the error pattern representative EB_k for the kth cluster by

$$EB_k = \frac{GE_k}{counter_k} \tag{9}$$

Each error pattern representative EB_k can be viewed as an average error pattern associated with the quantized coefficient representative QCB_k. Hence, for every received quantized coefficient block $\tilde{D}_i$, we can find its closest quantized coefficient representative from $\{QCB_k\}$ by searching for the best match between $\tilde{D}_i$ and $\{QCB_k\}$. This process is called *error pattern estimation*. And then the error pattern compensation is performed by adding the associated error pattern representative EB_k to $\tilde{D}_i$. The scheme of postprocessing using error pattern estimation and compensation is shown in Fig. 6. In the rest of this paper, error pattern representatives will be abbreviated as error patterns if there is no ambiguity in meaning.

3.2 Performance

There are three important parameters involved in our simulations. They are coding picture quality, training picture quality, and the number of error pattern representatives. Coding picture quality (PQ) is the value of quality control parameter Q used for the test image in postprocessing. Training picture quality (TQ) is the Q value used in the error pattern training process. The number of representatives (NR) is the number of quantized coefficient representatives, which is also equal to the number of error pattern representatives. These parameters have a strong impact on the postprocessing performance. The test picture used in simulation is "lena", 512×512, 256 gray level of a pixel, and it is outside the training set. Although PSNR does not well represent the subjective quality of an image, it is widely used in image coding literature and is defined as

$$PSNR(dB) = 10 \log \frac{(255)^2}{\sum_{i=1}^{M} \sum_{j=1}^{N} [x_{ij} - y_{ij}]^2}, \tag{10}$$

where x_{ij} and y_{ij} are the (i,j) th pixels in the original and processed images, respectively, and the image is of size $M \times N$.

An error pattern set developed using quality Q_1 may not work well on the test images quantized using quality Q_2. When the test picture quality (PQ) is different from the training picture quality (TQ), the error pattern compensation scheme becomes less effective. This performance degradation is largely due to the increase or decrease of the number of truncated coefficients. We found that a scaling factor (defined below) of adjusting the magnitude of error patterns is somewhat effective for PQ smaller than TQ.

$$E'_{adjust} = E' \times \frac{Test\ Picture\ Quality}{Training\ Picture\ Quality}. \tag{11}$$

However, it does not work for PQ greater than TQ. This is expected since truncated coefficient variation is a highly nonlinear effect and thus cannot be compensated by a simple linear scaling factor. Hence, at the present, the above adjustment formula is used only when PQ is less than TQ.

The number of representatives affects the preciseness of pattern clustering and error pattern estimation. Although a larger number of representatives should probably provide more accurate error estimation, the set of representatives may be overly trained to fit into the training images. It is likely that many clusters would have too few blocks. On the other hand, if the representative size is chosen small, there may be insufficient number of representatives to represent different shapes of image blocks. Simulation results indicate that representative size around 512 is close to the optimum.

As discussed earlier, when the test picture quality value (PQ) does not match the training quality value (TQ), the performance of error pattern compensation is degraded. From the PSNR gain plot shown in Fig. 7, we find that each training quality has its best performed range. For $TQ = 140$, its best performed range is from $PQ = 190$ to $PQ = 150$. For PQ is under 150, the error pattern set trained with $TQ = 100$ has a better performance. Again, if PQ is under 95, the $TQ = 60$ error pattern set is the better performer. The reason for the locality of error pattern compensation is that the quantization (truncation) process is nonlinear. When PQ is far away from TQ, the error patterns become very poor approximations to the true quantization errors even with scaling. Under our assumption, error pattern compensation affects mainly the high contrast regions, or the said edge areas. This can be verified by looking at the difference image (Fig. 8) between the JPEG standard decoded image (Fig. 3) and the image postprocessed by error pattern compensation.

4. INTER-BLOCK TRANSFORM COEFFICIENT ADJUSTMENT

Among the various types of degradations in low bit rate image coding, the blocking effect is generally considered to be the most disturbing one in the reconstructed images. In the international image standards such as JPEG, the blocking effect is mostly due to the fact that the coded images are partitioned into square-shaped blocks and that every block is treated independently in the coding process. The quantization of transform coefficients introduces discontinuities between neighboring blocks. These discontinuities are particularly noticeable in the monotone (smooth) areas, for example, shoulder and background in Fig 2. A more clearly illustration is shown by Fig. 9. Fig. 9(a) is a segment of 9 blocks in Fig.2, and Fig. 9(b) is a 9-block segment of Fig. 4. Comparing with Fig. 9(a) to 9(b), the blocking effect regions are shown clearly in the standard coding-decoding process.

Our objective is to reduce the discontinuity caused by block doing so that the processed pictures have better subjective apparence to human eyes. The to-be-processed image blocks can be classified roughly into two groups, *the edge areas* and *the monotone areas*. Since monotone clocks are not compensated by the error patterns in the previous section, in this section we will focus on the processing of monotone areas.

An image block classification scheme in transform domain similar to that in [4] is adopted. Our goal is to separate high contrast (edge) areas from shade (smooth) areas. The DCT domain is chosen here partially because the data received at decoder are the quantized DCT coefficients and partially because the classification can be done with rather simple schemes in DCT domain. The DCT blocks are first classified into four groups according to the AC energy distribution in the transform domain. Then, all the blocks are further divided in accordance with the locations of the largest energy group in DCT domain.

4.1 DC Value Adjustment

In monotone areas of the original image, where the intensity is changing gradually, the coded image tends to change abruptly from one block to another. Continuity of the tone within one block is often preserved by the coder, but it is

not assured between blocks. If we want to reduce the blocking effect on the coded images, we need to reestablish the continuity among the neighboring blocks.

Since DC value represents the average intensity of an image block, the entire image block intensity level can be changed by adjusting the DC value. To abridge the discontinuity between neighboring blocks, we use the weighted average method to adjust DC value of the current block based on the information from the neighboring blocks, as illustrated by Fig. 10. We assume DC_{ij} is the DC value of $Block_{ij}$ in Fig. 10. A heuristic approach to make a smooth transition between $Block_{i,j}$ and its neighbors is to replace $DC_{adjust_{i,j}}$ for DC_{ij}, where $DC_{adjust_{ij}}$ is computed as follows.

$$DC_{adjust_{i,j}} = \frac{\sum_{g=-1}^{1}\sum_{h=-1}^{1}(DC_{i+g,j+h} \times WM_{gh})}{\sum_{g=-1}^{1}\sum_{h=-1}^{1} WM_{gh}}, \tag{12}$$

where the weighting mask WM is selected from experience and is

$$WM = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 13 & 1 \\ 1 & 1 & 1 \end{bmatrix}. \tag{13}$$

4.2 VAC and HAC Value Adjustment

The DC value adjustment changes only the average image intensity level; it does not change the block shape. Although the discontinuity of DC values among adjacent blocks has been lowered and that often produces noticeable improvement for reducing blocking effect, there remains discontinuity and grid effect that are not eliminated completely. Hence, we further adjust two additional transform coefficients. They are VAC and HAC, the (0,1) and (1,0) DCT coefficients. Since the operations on HAC is similar to those on VAC, only the VAC operations are described here.

In the forward DCT, eq.(1) with $v = 0$ and $u = 1$,

$$\begin{aligned} VAC &= S(0,1) \\ &= \tfrac{1}{4}\tfrac{1}{\sqrt{2}}\sum_{x=0}^{7}\sum_{y=0}^{7} P(y,x)\cos\tfrac{(2x+1)\pi}{16}. \end{aligned} \tag{14}$$

The coefficients allocated on the first horizontal line in the transform domain correspond to the vertical line variations in the spatial domain. Hence, $S(0,1)$ is called VAC (vertical AC coefficient). The amount of pixel values in spatial domain affected by VAC can be calculated by the inverse DCT in eq.(2),

$$P(y,x) = \frac{1}{4}\frac{1}{\sqrt{2}} VAC \cos\frac{(2x+1)\pi}{16}. \tag{15}$$

From eq.(15), it is clear that VAC only relates to the magnitude variation in the horizontal direction, modulated by a cosine function. Thus, we can adjust VAC to eliminate the grid effect on the horizontal boundaries and to reestablish the continuity in horizontal direction. The adjacent blocks affected by VAC adjustment are $Block_{i,j-1}$, $Block_{i,j}$, $Block_{i,j+1}$, as shown in Fig. 11.

Because of the location-dependent variation of cosine function associated with VAC adjustment, it is difficult to control it very precisely; hence, we only use the VAC adjustment to eliminate the remaining defects after DC adjustment. Therefore, we need to estimate the gaps between two adjacent blocks at their boundaries. Because IBTCA is operated in the monotone areas, we assume there is no acute variation within one block; that is, the transform coefficients are centered in the low frequency region of the classified monotone areas. This assumption, indeed, is generally satisfied in most cases. Hence, eq.(15) is used to estimate the gaps by the following procedure.

Let $P_{i,j}(left)$ is the intensity of the left boundary of $Block_{i,j}$. Then,

$$\begin{aligned} P_{i,j}(left) &= \tfrac{1}{4}\tfrac{1}{\sqrt{2}} VAC_{i,j}\cos\tfrac{\pi}{16} \\ &= 0.17654 VAC_{i,j}. \end{aligned} \tag{16}$$

Similarly, $P_{i,j}(right)$ is the intensity of the right boundary of $Block_{i,j}$,

$$\begin{aligned} P_{i,j}(right) &= \tfrac{1}{4}\tfrac{1}{\sqrt{2}} VAC_{i,j}\cos\tfrac{15\pi}{16} \\ &= 0.17654 VAC_{i,j}. \end{aligned} \tag{17}$$

For $Block_{i,j-1}$, its right block boundary is

$$P_{i,j-1}(right) = 0.17654 VAC_{i,j-1}. \tag{18}$$

And the left block boundary of $Block_{i,j+1}$ is

$$P_{i,j+1}(left) = 0.17654 VAC_{i,j+1}. \tag{19}$$

Thus, the gaps between adjacent block boundaries can be computed by

$$gap_{i,j}(left) = P_{i,j-1}(right) - P_{i,j}(left) \tag{20}$$

and,

$$gap_{i,j}(right) = P_{i,j}(right) - P_{i,j+1}(left). \tag{21}$$

The cross section plot of boundaries intensity and gaps is illustrated by Fig. 12.

The average gap between the current block boundaries and its neighboring blocks is,

$$gap_{average_{i,j}} = \frac{gap_{i,j}(left) - gap_{i,j}(right)}{2}. \tag{22}$$

If the current block and its adjacent blocks have continuous pixel values along the horizontal direction, the $gap_{average}$ be close zero; otherwise, a large $gap_{average}$ implies a large amount of discontinuity. We try to lower the discontinuity by reducing the value of $gap_{average}$. The gap adjustment is half of $gap_{average}$, $gap_{adjust} = gap_{average}/2$, since the reduction of discontinuity is across two block boundaries. To close up the boundary gaps around $Block_{i,j}$,

$$VAC_{adjust} = VAC_{i,j} + \frac{4\sqrt{2} gap_{adjust}}{\cos(\pi/16)}. \tag{23}$$

The cross section of VAC adjustment is shown in Fig. 13. The adjustment of HAC to reduce vertical direction discontinuity is defined in a similar manner.

In addition, to assure that IBTCA is properly operated in the monotone areas, the processed segment is consist of the current block and its 8 neighbors so that the local image shape can be identified. In our scheme if one or more blocks in a segment is classified as an edge block, then this segment is said lying in edge areas. Therefore, the current block is not processed using IBTCA; otherwise, IBTCA is applied. The postprocessing algorithm using IBTCA is shown in Fig. 14, and the overall postprocessing scheme that combines error pattern compensation and inter-block transform coefficient adjustment is shown in Fig. 15.

4.3 Performance

The effect of IBTCA with DC value adjustment can be seen by comparing Fig. 16(a) to Fig. 9(b). The gaps across the block boundaries are reduced clearly by using DC value adjustment. The numerical reduction is not significant here since the objective of IBTCA is to produce more pleasant images not to recover the original images. Therefore, the processed images may not match the original images exactly, but they have better visual quality.

Fig. 17 shows the result of using both error pattern compensation and IBTCA for $PQ = 140$. Since error pattern compensation mainly operates in the edge areas and IBTCA mainly in the monotone areas, together the fully processed images have much more superior subjective and objective improvement than the picture before processing.

5. DISCUSSIONS

In low bit rate transmission, there exists significant signal distortion in the coding-decoding process. To reduce coding distortion, several techniques have been proposed at either the encoder or the decoder. Due to the constraints posed by image standards, we focus on the improvement of the processing only at the decoder and in the transform domain.

Two techniques are proposed for postprocessing in this paper, namely, error pattern compensation and inter-block transform coefficient adjustment. Error pattern estimation is a model that estimates the lost information due to coefficients quantization (truncation). It tries to restore the distorted images at the receiver. Its performance is most clearly seen in high contrast areas. Inter-block transform coefficient adjustment is a method of reducing blocking effects

in the monotone areas of distorted images. It adjusts three transform coefficient values to reduce the discontinuity between neighboring blocks caused by block coding scheme. Its performance is mainly on improving the subjective quality, not on the numerical errors.

Initial investigation shows rather promising results using these two techniques. However, detailed analysis of selecting optimal parameters needs to be further explored. Particularly, we like to search for a robust set of error patterns that can operate in a wider range of quantization step size.

6. ACKNOWLEDGMENT

This research has been supported by a grant from National Science Council, Taiwan.

7. REFERENCES

[1] B. Ramamurthi et al., "Non-Linear Space-Variant Postprocessing of Block Coded Images," IEEE Trans. on ASSP, pp.1258-1267, Vol 34, No. 5, 1986.

[2] "JPEG Draft Technical Specification Revision 8," Aug. 1990.

[3] Y. Linde, A. Buzo and R. Gray, "An Algorithm for Vector Quantizer Designs," IEEE Trans. on Commu., pp. 84-95, COM-2, Jan. 1980.

[4] Y. S. Ho and A. Gersho, "Classified Transform Coding of Images Using Vector Quantization," Intl. Conf. on Acoust., Speech, and Signal process., pp.1890 - 1893, Glasgow, Scotland, May 23 - 26, 1989.

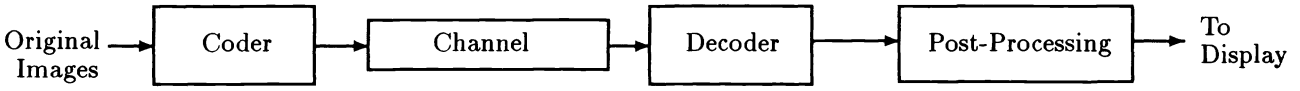

Figure 1: An Image Coding System with Postprocessing

Figure 2: Original Picture — *Lena*

Figure 3: JPEG Decoded Picture, PQ=140

Figure 4: The Difference Picture between the Original Picture (Fig. 2) and the JPEG Decoded Picture (Fig. 4)

Figure 7: The Difference Picture between the JPEG Decoded Picture and the Error Compensated Picture (PQ=140, TQ=140, and NR=512)

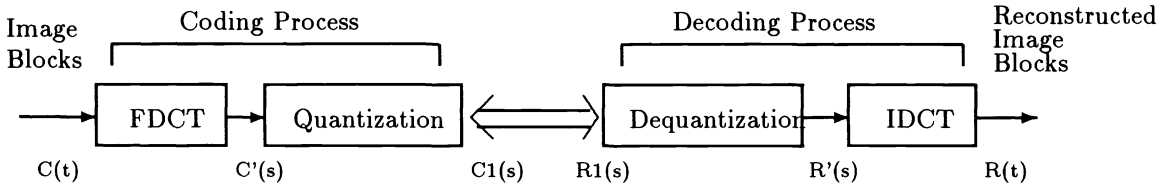

Figure 5: A Simplified Coding-Decoding Diagram

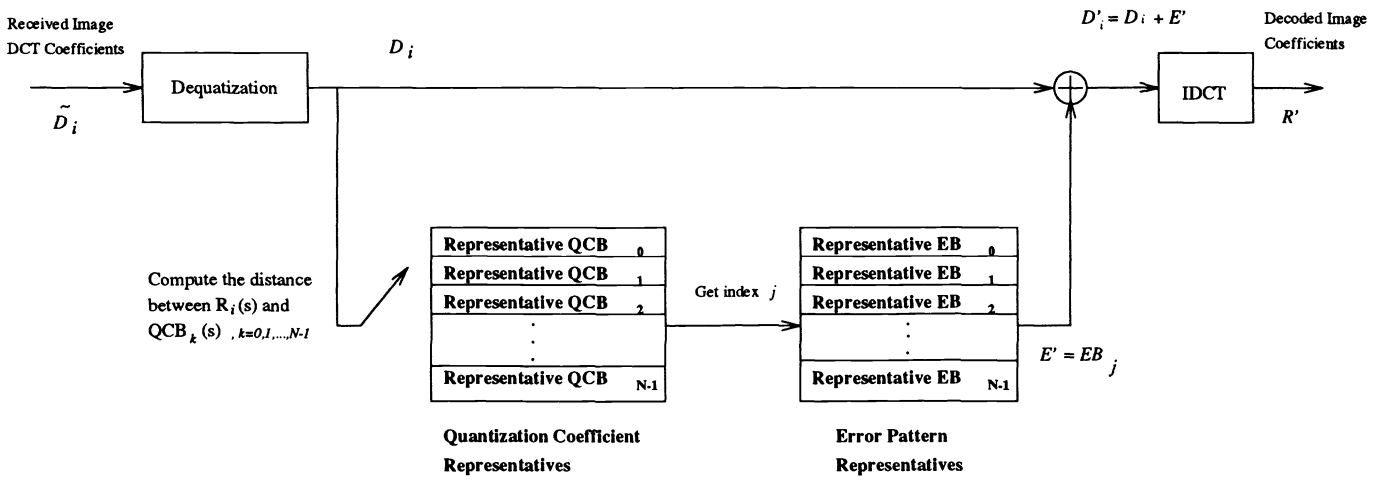

Figure 6: Error Pattern Estimation and Compensation

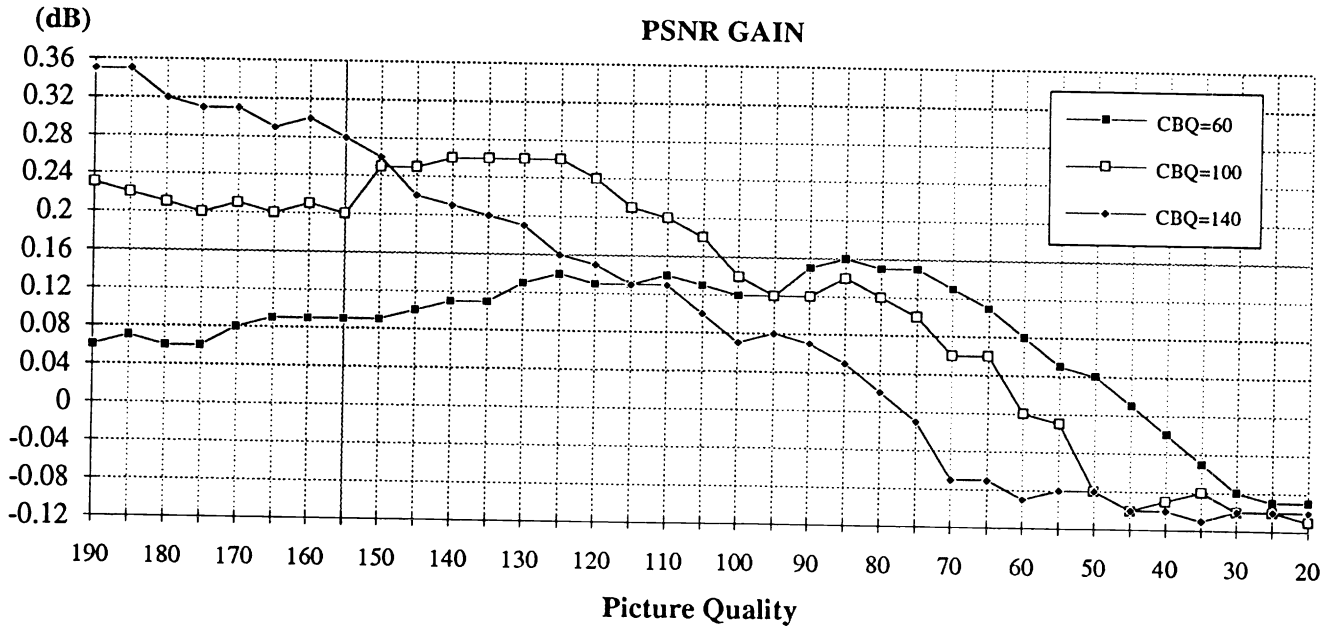

Figure 8: Gain of Postprocessing Using Error Pattern Compensation

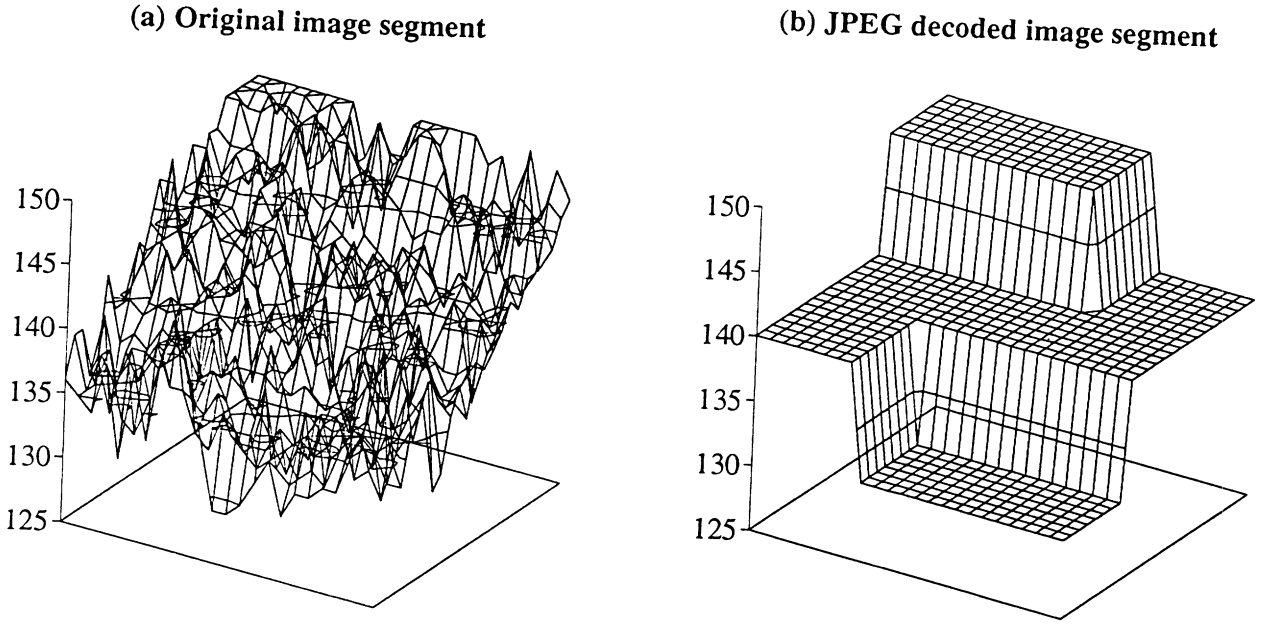

Figure 9: 3D View of Segments of (a) the Original Picture and (2) the Decoded Picture

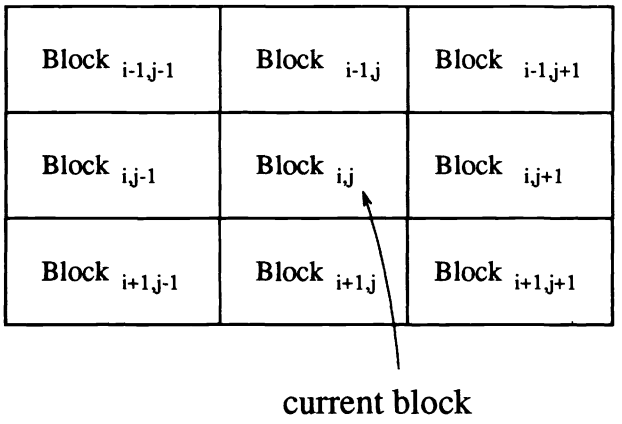

Figure 10: The 9 Neighboring Blocks

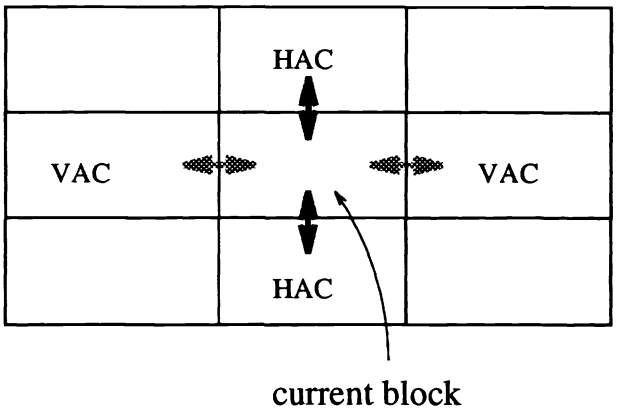

Figure 11: The Adjacent Blocks Affected by VAC and HAC Adjustment

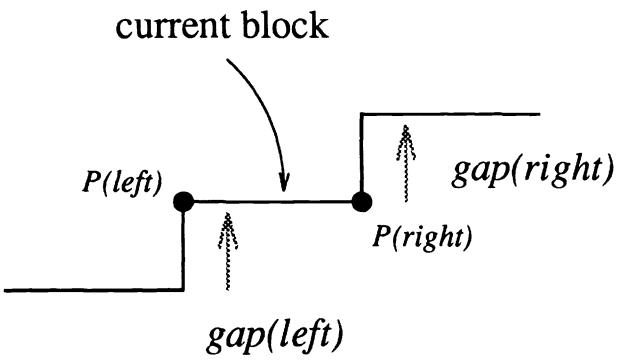

Figure 12: The Cross Section of Boundary Intensities and Gaps

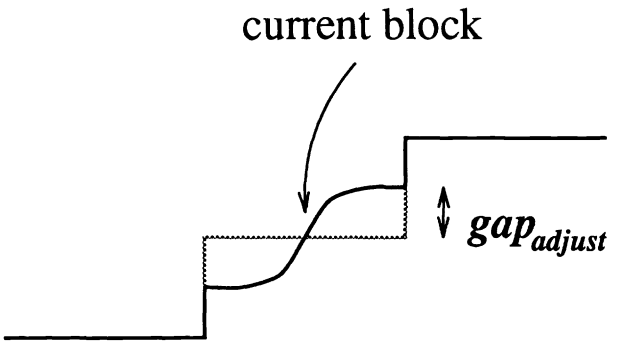

Figure 13: The Cross Section of VAC Adjustment

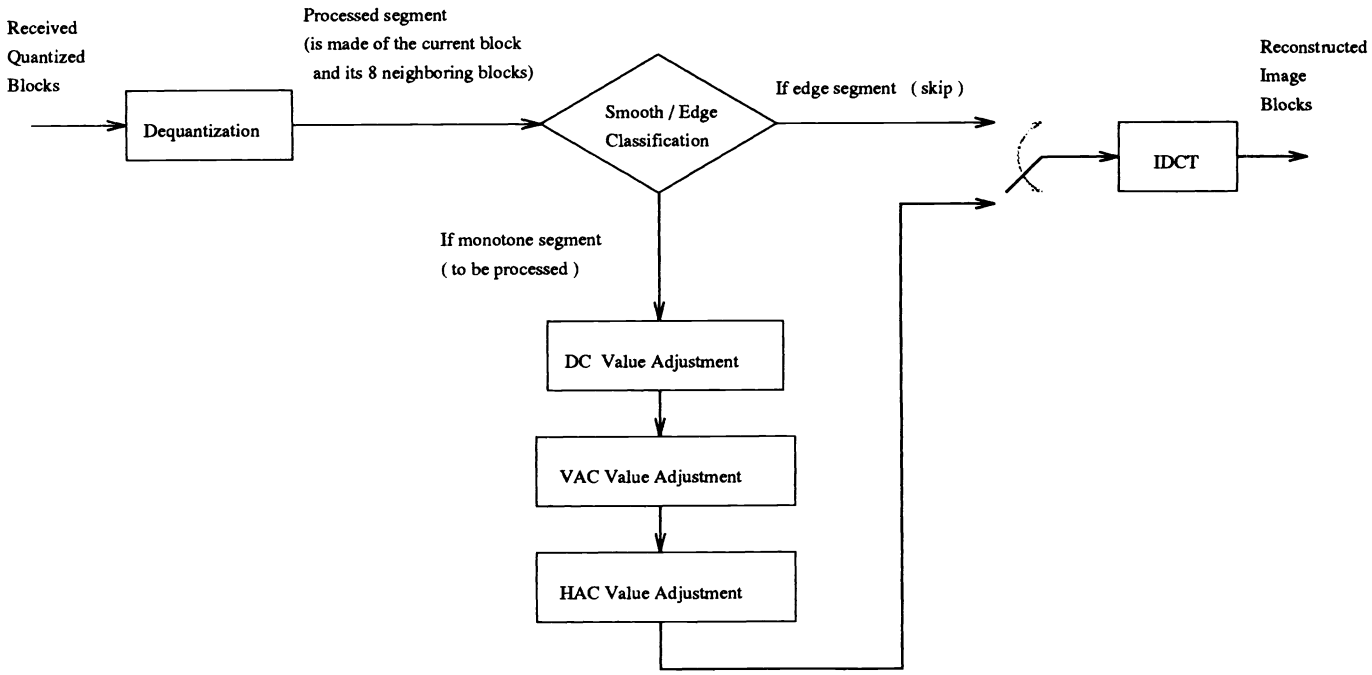

Figure 14: IBTCA Used in Postprocessing

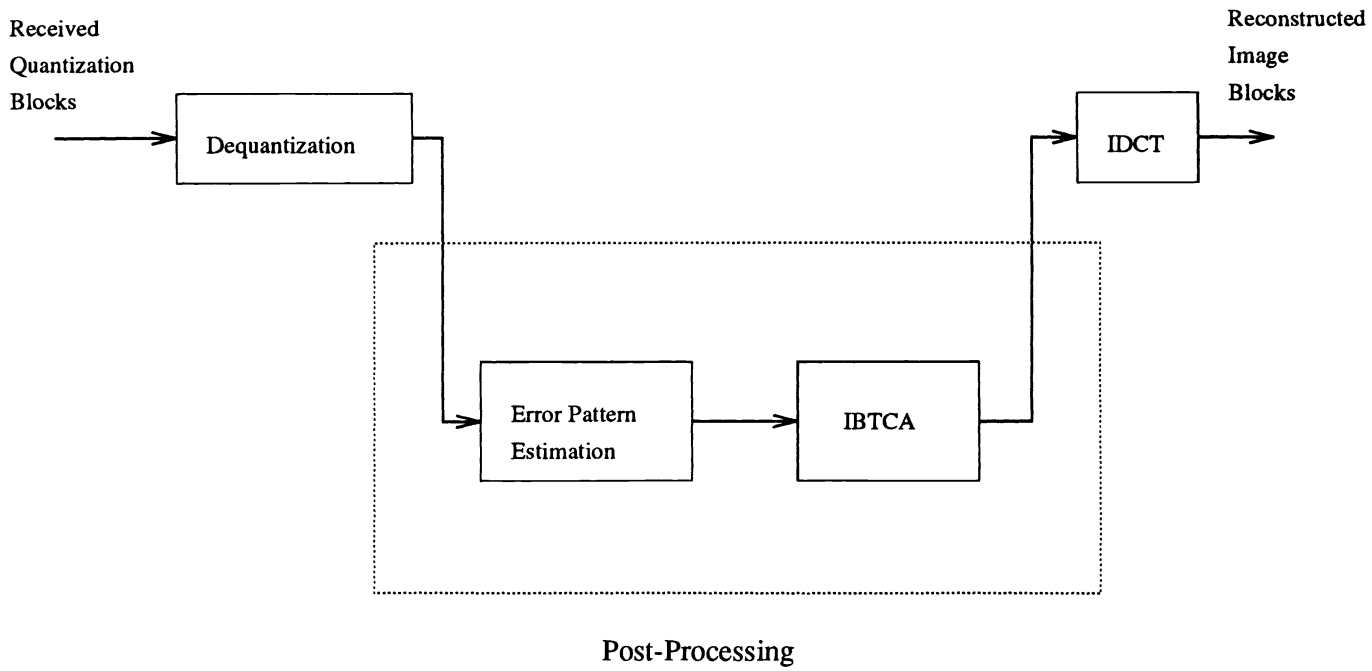

Figure 15: Postprocessing Using Both Error Pattern Compensation and IBTCA

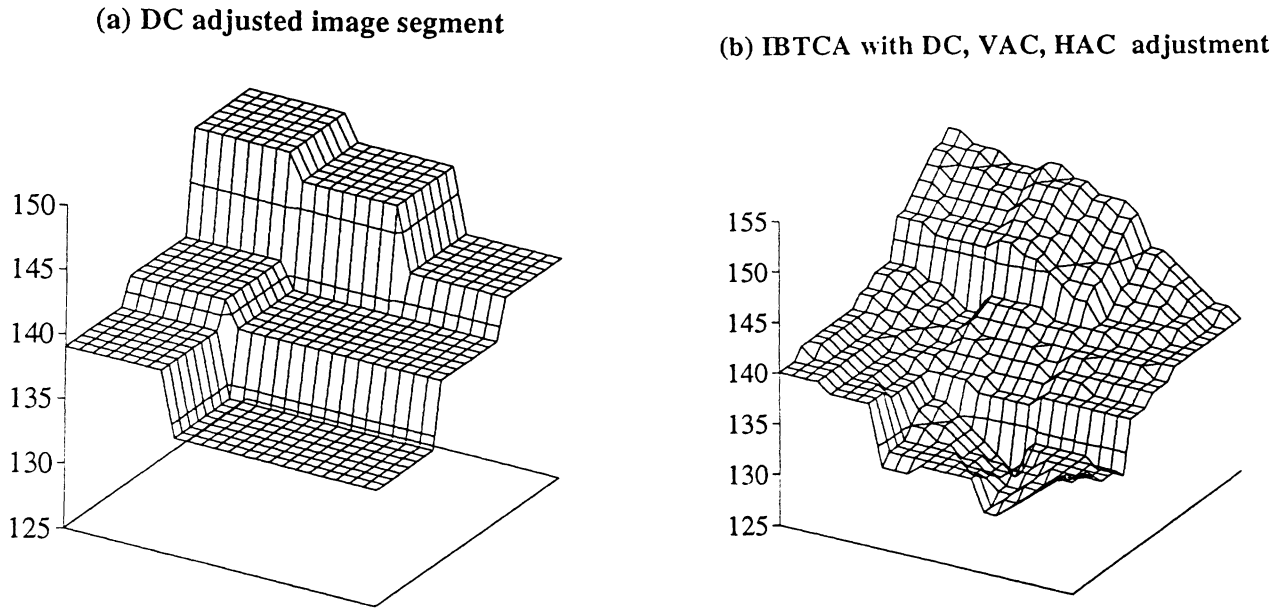

Figure 16: 3D View of Segments Processed by (a) DC Adjustment and (b) DC, VAC, and HAC Adjustment

Figure 17: Picture Postprocessed Using Both Error Pattern Compensation and IBTCA with PQ=140, TQ=140, and NR=512