An Image Coding and Reconstruction Scheme for Mobile Computing

Edward Chang
Department of Electrical Engineering
Stanford University
echang@cs.stanford.edu

Abstract

An asynchronous transfer mode (ATM) wireless network has bursty and high error rates. To combat the contiguous bit loss due to damaged or dropped packets, this paper presents a code packetization and image reconstruction scheme. The packetization method distributes the loss in both frequency and spatial domains to reduce the chance that adjacent DCT blocks lose the same frequency components. The image reconstruction takes into consideration the spatial characteristics represented by the frequency components. Combining these two approaches is able to reconstruct the damaged images more accurately, even under very high loss rates. In addition, since the reconstruction technique is computational efficient, it conserves system resources and power consumption, which are restrictive in mobile computers.
Keywords: multimedia, compression, mobile computing.

1 Introduction

Many studies have proposed image and video data delivery schemes over wired channels [3, 4, 5, 6, 10]. Since mobile computing has gained popularity in recent years, it is important also to devise robust techniques that work in a mobile environment. This paper presents image coding and reconstruction techniques that are suitable in a mobile (wireless) environment.

Wireless channels are different from wired ones in at least two respects: capacity and reliability. Regarding the first, the wireless channel's capacity is typically limited to 40 to 100 Kbps. Regarding the second, the compressed video signals delivered by wireless channels are often corrupted in transmission [2]. A compression algorithm suitable for the wireless environment needs not only to have high compression rates but also to maintain a high degree of error tolerance [15].

Some priority transmission schemes have been developed for the wired channels [1]. These schemes, however, do not work in the mobile environment since the signal disturbance can happen to any packet indiscriminately, despite its delivery priority. To address the problem of the lack of resiliency to channel errors, layered coding and error correction schemes such as automatic retransmission query protocols (ARQ) or forward error correction (FEC) have been proposed [14]. However, if the network is already congested, ARQ tends to aggravate the problem by sending out retransmitted packets. Also, retransmission does not help real-time decoding schemes such as motion JPEG or MPEG. As for error correction codes, they are effective only when the loss rate is below the design threshold. Furthermore, the redundant transmissions and codes required to improve reliability decrease the effective channel capacity.

This paper presents research on an image packetization and reconstruction scheme that works at low channel capacity and in high bursty packet loss environments. Specifically, my design objectives have aimed at 1) preventing propagation of errors, 2) smoothing out the effects of bursty loss, and 3) reconstructing damaged images accurately and efficiently. The study is based on the existing JPEG international standards. However, since standards such as H.261 and H.263 use compression schemes similar to JPEG to perform intra-frame encoding, the techniques that work for JPEG also work for video key frames.

To distribute errors, [7] proposed reordering the DCT blocks' delivery sequence. My technique takes a step further in also scrambling the coefficients in the blocks. To smooth out the bursty packet loss, the packetization scheme distributes the 64 frequency components of a DCT block to 64 different transmission units. In other words, each transmission unit contains 64 different frequency components from 64 DCT blocks. Next, these transmission units are packetized into different network packets by a carefully chosen "stride." During both steps, the packetization scheme makes sure that in the case of a bursty error, the loss is not concentrated in one area of the image. Thus, during image reconstruction, the lost frequency components have a better chance to be recovered from the neighboring blocks.

To reconstruct the image, my approach takes the spatial characteristics represented by the DCT coefficients into consideration. I show that interpolating the high frequency components in

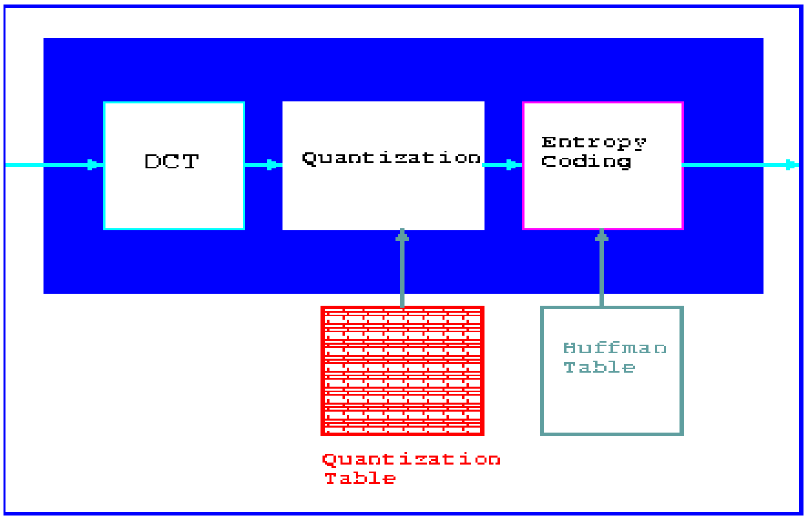


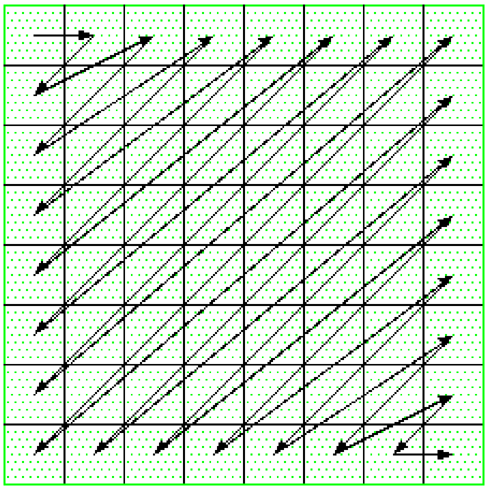
Figure 1: JPEG Encoding Procedure

the horizontal direction by using only the top and bottom blocks, and in the vertical direction by using only the left and right blocks, repairs the damaged images more accurately than a scheme that takes an average over all neighboring blocks. (For instance, to reconstruct a standing tree in an image, the relevant information is in the DCT blocks above and below the damaged block. The information in the blocks left and right of the damaged block is irrelevant to reconstructing the standing tree.) Moreover, the reconstruction scheme, since is not computing intensive, conserves battery power for mobile computers.

The rest of this paper is organized as follows. Section 2 offers a brief overview of JPEG standards. Section 3 describes the error characteristics of the wireless network and related work to improve its transmission resiliency. Section 4 details the packetization and image reconstruction schemes proposed to combat the bursty packet loss problem. Section 5 presents the evaluation of the proposed schemes by demonstrating that the damaged images can be better reconstructed with these techniques than with others even under high loss rates.

2 A Brief Overview of JPEG

To understand the techniques proposed in this paper, this section gives a brief overview of JPEG [16]. JPEG encoding procedure consists of three steps: DCT (Discrete Cosine Transform),



(a) DCT Coefficient Matrix

16	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

(b) Quantization Table

Figure 2: DCT Coefficients

quantization, and entropy coding. JPEG decoding, conversely, consists of entropy decoding, dequantization, and IDCT (Inverse Discrete Cosine Transform). Figure 1 depicts the encoding steps; the decoding steps are executed in the opposite direction. The following describes the encoding steps only.

JPEG deals with colors in the YUV (one luminance and two chrominances) space. For each separate color component, the image is broken into 8 by 8 pixel blocks of picture elements, which join end to end across the image. JPEG transforms each block into a two-dimensional DCT matrix by performing a one-dimensional DCT on the columns and on the rows. Figure 2(a) shows an 8 by 8 coefficient matrix generated by the DCT step. The arrows link the frequency components from DC to the highest. Note that at this point the information about the original image is mostly preserved, except for the rounding errors of the DCT coefficients.

The coefficients of the DCT are quantized to reduce their magnitude, to increase the number of zero value coefficients, and to reduce the bit rate. Figure 2(b) shows the uniform midstep quantizer that is used for the JPEG baseline method, where the stepsize varies according to the coefficient location in the DCT matrix. As one can see, the low frequency components (at the upper left corner of the quantization matrix) have smaller steps, while the high ones have larger steps. This is because human eyes are less sensitive to high frequency components. Quantization is the lossy stage in the JPEG coding scheme. If we quantize too coarsely, we achieve a higher compression rate with poorer image quality. On the other hand, if we quantize too finely, we may spend extra bits on coding noise.

The final step is entropy coding. JPEG uses the Huffman or arithmetic coding scheme to compress the quantized Discrete Cosine Transform (DCT) coefficients to approach the entropy rate. Since both the Huffman and the arithmetic codes are variable-rate, any bit errors or loss in the transmission propagates and destroys the entire image. One remedy to this problem is to introduce resynchronization points in the code to limit the effect of the loss. Another way is to use fixed-rate coding schemes [15] that trade bit rates for error containment. With minimum changes to the current JPEG baseline, I use resynchronization bits to contain the loss effect. The proposed packetization and reconstruction schemes work orthogonally with variable or fixed-rate codes.

3 Wireless Network Error Characteristics

The transmission errors in a wireless channel may not occur independently of digit positions. Rather, the electrical disturbance of interfering radiation sources and of attenuation due to distance and obstacles frequently affects adjacent transmitted packets [11]. The packets that are damaged beyond repair are discarded and considered a loss by the network. According to the statistics collected by the Stanford University Wireless Computing Group [2], the average packet loss rate is about 3.6% in a wireless environment and occurs in a bursty fashion. According to the study conducted by the Stanford Multimedia Network Group for a teleconferencing application [8, 9], the duration of a burst, or the number of successive packets lost, ranges from one to seven packets. In terms of an ATM network, since each packet contains 56 bytes or 448 bits of data, each burst of loss translates to a loss of 448 to more than 3,000 consecutive bits, which can be a significant chunk of an image.

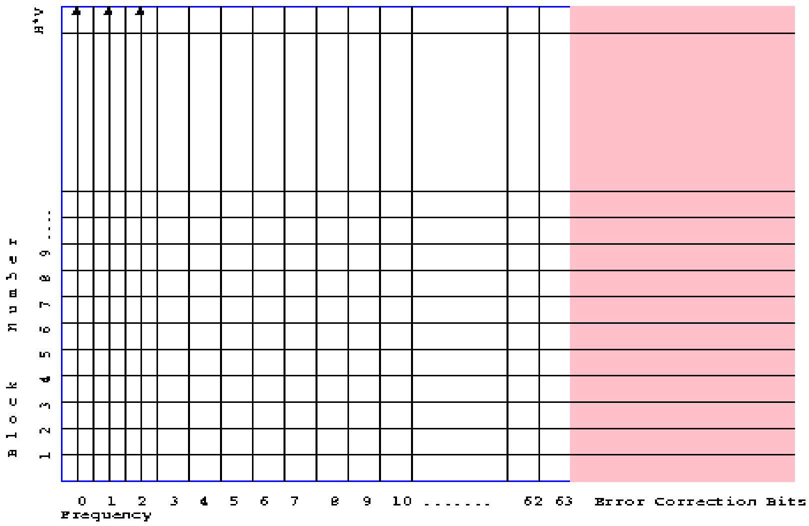


Figure 3: Interleaved Code

Total Bits	*Information Bits*	*Maximum Correction Bits*	*Overhead %*
127	*113*	*2*	10%
127	*106*	*3*	12%
127	*99*	*4*	29%
127	*92*	*5*	40%
127	*64*	*10*	100%
255	*107*	*22*	150%
255	*99*	*23*	170%

Table 1: Bit Efficiency for Interleaved Code

One approach to recover loss is introducing redundancy to correct errors. Figure 3 shows the DCT blocks in an image organized into a two-dimensional array. Each row represents a DCT block with 64 frequency components (from 0 to 63), and the number of rows represents the number of DCT blocks in an image. For each DCT block the Hamming code for instance can be used to correct its bit errors. The number of bit errors the Hamming code is able to correct depends on the number of parity bits (the shaded area in the figure) added to the code. Now, in order to deal with bursty errors, the data are sent out by the columns of Figure 3. The data transmission starts from the first bit (the first column), second bit (the second column), and so on of each block. If one ATM packet (448 bits) is lost, one may lose only one to two bits in each row (DCT block). The loss can be corrected by the Hamming code. This interleaved code effectly uses a scheme that is efficient at correcting random bit errors.

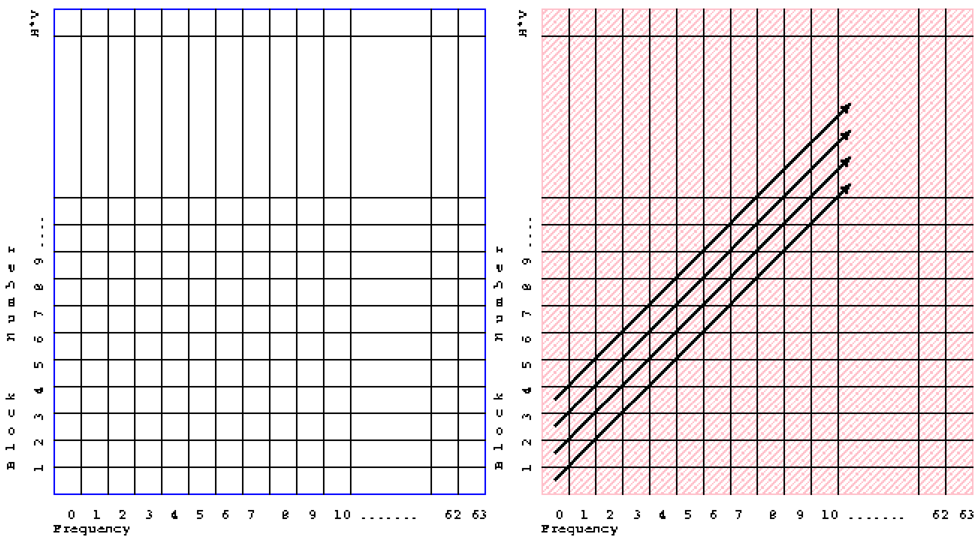
However, the overhead of this interleaved code can be high. Assuming that each DCT block takes up about 100 bits (since the Huffman code is a variable-rate code, the actual bit number may vary), Table 1 lists the number of parity bits needed to correct different number of errors. The number of additional bits needed to cover additional errors grows super-linearly. The effectiveness of this interleaved code depends on the degree of redundancy, listed in Table 1. For example, the fifth and the sixth rows in Table 1 show that correcting a 20% loss rate (20 bits for each 100 bits block) doubles the code length (the total bits over the information bits). If a code is designed for up to 20% loss rate and the actual error rate is higher than 20%, the scheme breaks down. In an environment where loss is infrequent and the increasing bit rate is not a concern, the interleaved code may be a better choice. However, because of the channel capacity constraint and high error rates, the smart packetization and heuristic reconstruction techniques proposed below are better alternatives for the wireless network.

4 Packetization and Reconstruction

My proposed scheme to combat contiguous bit loss consists of two parts: packetization and reconstruction. The objective of the packetization step is to scatter the bursty loss in the spatial domain so that adjacent DCT blocks do not lose the same frequency components. This aids to the reconstruction quality since 1) a block does not lose all its coefficients and 2) most lost frequency components can be recovered from adjacent blocks. In the reconstruction step, I also take spatial characteristics of the frequency components into consideration to repair the damaged image more accurately.

4.1 Packetization

To minimize the effect of bursty packet loss on an image, the packetization scheme must achieve two goals in the spatial domain. First, for each burst of error, the lost information must not be clustered on the image. Second, the spatially adjacent 8 by 8 blocks of the image must not lose the same frequency components. If both objectives are met, each block will lose only a small number of coefficients, and the lost frequency components will have high chance of being



(a) DCT Coefficient Array

(b) Block Packetization

Figure 4: DCT Coefficient Packetization

recovered from the neighboring blocks (which do not lose the same frequency components).

Without loss of generality, one can assume that each frame is divided into an H (horizontal) times V (vertical) number of 8 by 8 blocks. Designating the code of each block as C(h,v), one can construct the H times V code array shown in Figure 4(a). The horizontal axis in Figure 4(a) represents the frequency components of each block from 0 to 63. The vertical axis of the figure represents the blocks ordered by their spatial locations from the top left-handed corner (h = 0 and v = 0) of the image to the bottom right-handed (h = H and v = V).

To make sure that the lost coefficients are shared by as many blocks as possible, each transmission unit (the new block constructed from the DCT blocks) includes frequency components from 64 blocks. Figure 4(b) illustrates a simple way to achieve this objective by packing frequency components diagonally. Note that entropy coding is performed after this packetization step, and hence the packetization scheme is not affected by whether the resulting code is fixed or variable-rate.

Packing coefficients from different blocks achieves only part of the objective. Next, one

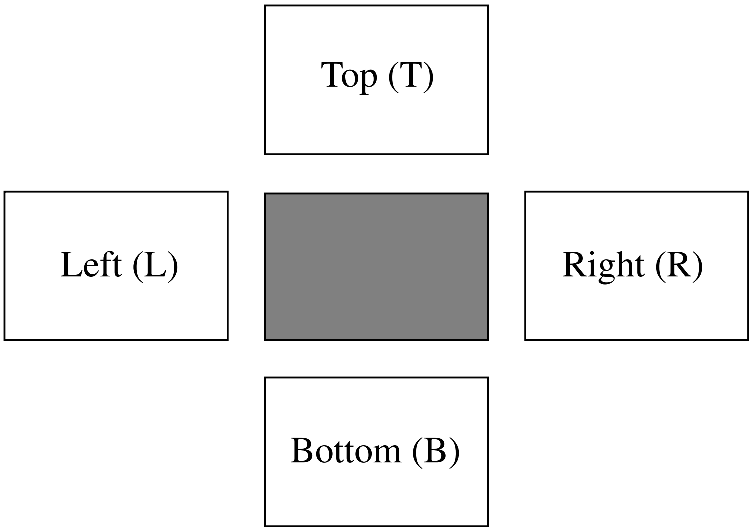


Figure 5: Spatial Adjacent Blocks

wants to ensure that the spatially adjacent blocks do not lose the same frequency components when packets are dropped. Let T (top), B (bottom), L (left), and R (right) denote the spatial adjacent DCT blocks, as depicted in Figure 5. Section 4.2 will show that the lost coefficients can be reconstructed from a block's four neighboring blocks (except for the blocks on the edges). However, to ensure that the neighboring blocks do not lose the same frequency components, i.e., to optimize the reconstruction, I propose packetizing these transmission units in "strides."

A stride is the number of blocks skipped between packing blocks. For instance, if the stride is 2, we packetize blocks in the order of 1, 3, 5, 7, . . . and then 2, 4, 6, 8, . . ., etc. This way, for example, if blocks 3, 5 and 7 are lost in a burst, the DC components of the lost blocks can be reconstructed from blocks 2 and 4 for 3, 4 and 6 for 5, and so forth in the horizontal direction of the image. To make sure that in the vertical direction no spatially adjacent blocks lose the same coefficients, one must shift the starting point of the stride from row to row. For instance, one can let the stride on the odd rows start from the first image column while that on the even rows starts from the second column. This way, even if the contiguous loss is as high as 50%, the neighbors of a lost block are still intact for reconstruction. In an environment where the loss rate is much lower, one can choose a larger stride to disperse the artifacts as long as the stride is prime to the width (in blocks) of the image.

To summarize, the coding scheme first packetizes the blocks diagonally to spread the frequency components to 64 transmission units. Then, it uses a stride method to pack these transmission units in such a way as to turn the potential consecutive block loss into a pseudo-random loss. To invert the packetization, the only information the decoder needs is the stride. This number can be sent with every packet with a negligible overhead (less than 4 bits) compared to the size of a compressed image.

4.2 Reconstruction

The studies of [12, 13] show two typical approaches to reconstruct a damaged image. The first approach, introduced in [13] is a decoder side reconstruction that interpolates the T, B, L, and R blocks to reconstruct the lost block. The technique used by [13] adds only 20% in computational overhead to the decoder, and the reconstruction is of higher quality than that achieved in the previous attempts. The reconstruction formula can be depicted as follow:

$$C_z = C_z + W_t \times C_t + W_b \times C_b + W_l \times C_l + W_r \times C_r, \tag{1}$$

where the weights, W_t, W_b, W_l, and W_r, are the weighting factors for averaging the neighboring blocks' coefficients to reconstruct coefficient C_z.

In a later study [12], Hemami proposes an encoder side approach in which the "optimal" weight vectors are computed for each DCT block based on 15 different combinations of available adjacent blocks (none available, one available, etc.). During the reconstruction step, for each block lost, the proper weighted vector is selected to perform linear interpolation. These vectors, since they take up large space ($200 - 600\%$ of the image data), are compressed using a vector quantizer before the image is shipped to the clients. The resulting space overhead, as reported, is reduced to about 10% of the image size. The drawback of this approach is that either the disk space requirement at the server side increases by 110% (to keep two different compressed images), or the bit rate decreases by 10% (the inflated version is transmitted regardless of the channel's characteristics). Another concern is that this approach greatly increases encoding time. Moreover, the scheme, although achieving better reconstruction quality, may not be optimal after all. I argue that the optimal weight set must be at the frequency component

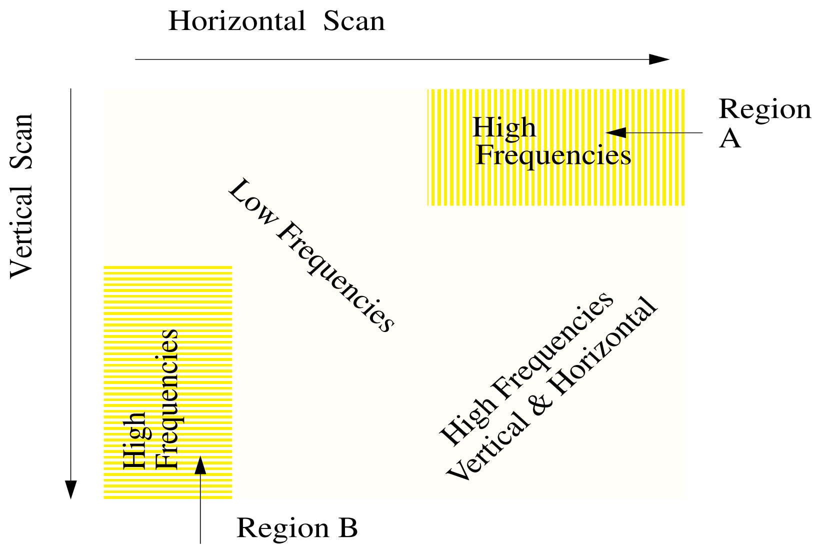


Figure 6: Frequency Component Characteristics

level rather than the block level. Since the optimal weight set is computational intensive and takes up large space, it may not be practical to expect an optimal reconstruction, especially in a mobile computing environment where the system resources and power supply are restrictive. Consequently, I suggest a simple and intuitive method to set the weight factors.

The following presents a much simpler heuristic approach that incurs no extra bit rate or computational overhead (crucial for mobile computing) for image reconstruction. Figure 6 shows the 8 by 8 DCT coefficient matrix by its spatial characteristics. The high frequency components corresponding to the horizontal DCT scan (region A in the figure) represent the vertical edges in the image (e.g., a tree), and the high frequency components corresponding to the vertical DCT scan (region B in the figure) represent the horizontal edges (e.g., a roof top). To take advantage of this relationship, the interpolation function (Eq. 1) should assign different weights for different frequency components. The higher the frequencies in the vertical scan (representing horizontal lines), the smaller the weight one should assign to the top and bottom blocks. On the other hand, the higher the frequencies in the horizontal scan (representing vertical lines), the smaller the weight one should assign to the left and right blocks. For example, for reconstructing a vertical line (e.g., a tree), only the top (T) and bottom (B) blocks of the missing block are relevant. The left (L) and right (R) blocks of the missing block do not have the same tree! If one added left and right blocks into the computation for reconstructing the vertical line, one would dilute the recovered coefficients. Of course, one cannot know whether a DTC block contains a tree or not. However, if a tree passes through the missing block, the high frequencies of the horizontal scan must not be zero, and this value can be recovered from the blocks above and below. On the other hand, if no tree passes through the missing block (in other words if the high frequency components in the horizontal scan are zero), there is a high probability that the same high frequency components in the blocks above and below are also zero. Therefore, the reconstruction scheme works accurately without the content of the image being known. Based on this heuristic, the weights of the reconstruction function

$$C_z = C_z + W_t \times C_t + W_b \times C_b + W_l \times C_l + W_r \times C_r,$$

can be assigned based on which portion of the DCT block is being reconstructed:

- For the high frequency components in the horizontal scan (region A in Figure 6):

  $W_t = W_b = \frac{1}{2}, W_l = W_r = 0.$

- For the high frequency components in the vertical scan (region B in Figure 6):

  $W_t = W_b = 0, W_l = W_r = \frac{1}{2}.$

- For the rest of the coefficients:

  $W_t = W_b = W_l = W_r = \frac{1}{4}.$

Again, the advantage of this technique is its simplicity. The examples in Section 5 shows its effectiveness.

Also note that, after the packetization scheme, not only does a block not lose all coefficients, but its neighboring blocks do not lose the same frequency components. This minimizes 1) the number of coefficients that need to be reconstructed in each block, and 2) the chance that consecutive blocks lose the same frequency components. Without this packetization scheme, reconstruction may interpolate using blocks many (e.g., 16 or 24) pixels away, blocks that may have no spatial correlation whatsoever.

5 Experimental Results

In my experiment, I first compare a damaged image that does not distribute the loss of the frequency components to other blocks (Figure 7) with one that uses my packetization scheme (Figure 8). Both images were generated under a loss rate of 20%. The major difference is that in the blocks where DC is lost, one can still see high frequency components in Figure 8. Although both images are damaged badly, Figure 8 looks slightly better. (The submitted images are colored.)

Next, I compared the reconstructed images from the first set of damaged images using our reconstruction technique, which takes spatial characteristics into consideration. Figure 9 is the image reconstructed from Figure 7, and Figure 10 is that reconstructed from Figure 8. It is clear that Figure 10 has better visual quality. The distant trees and lamps in Figure 10 are almost lossless as seen when Figure 10 is compared with the original image in Figure 11. The flower bed of the image in Figure 10 has much better continuity than that of Figure 9, which suffers from severe blocking effects. The distribution of the loss by the proposed packetization scheme significantly reduces the blocking effects for which DCT has often been criticized.

5.1 Reconstruction Under High Loss Rates

Our technique is effective even under conditions of very high bursty errors. Figures 12, 13, and 14 show the reconstructed images at the loss rates of 30%, 40%, and 50% respectively.

6 Conclusion

This paper has shown a packetization approach to transform bursty errors into pseudo-random errors. Since each DCT block loses only a limited number of frequency components, which are preserved in its neighboring blocks, the image can in this way be reconstructed to be closer to the original one. Moreover, the approach takes advantage of the spatial characteristics of the DCT coefficients to reconstruct the lost blocks. The scheme is simple and efficient to implement, compatible with the current JPEG standards, and achieves satisfactory results. The techniques proposed in this study are applicable to other DCT-based standards such as H261 and H263 for video conferencing, and H262 for motion pictures. Finally, since the scheme does not increase bit rate and reconstructs damaged images efficiently (hence keeps power consumption low), it is attractive for use in mobile computing environments.

Acknowledgement

I would like to thank professors Robert Gray and Hector Garcia-Molina at Stanford and professor Sheila Hemami at Cornell for their comments on the early draft. I would also like to thank Ken Lin for his help with the experiment.

References

[1] A. Albanese, J. Bloemer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *Proc. 35th Annual Symposium on Foundations of Computer Science*, pages 604–612, 1994.

[2] M. Baker. Wireless network characteristics. *Stanford Technical Report*, 1997.

[3] E. Chang and H. Garcia-Molina. Bubbleup - low latency fast-scan for media servers. *Proceedings of the 5th ACM Multimedia Conference*, pages 87–98, November 1997.

[4] E. Chang and H. Garcia-Molina. Effective memory use in a media server. *Proceedings of the 23rd VLDB Conference*, pages 496–505, August 1997.

[5] E. Chang and H. Garcia-Molina. Medic: Memory and disk cache for media servers. *Stanford Technical Report SIDL-WP-1997-0076 URL: http://www-diglib.stanford.edu*, November 1997.

[6] E. Chang and H. Garcia-Molina. Reducing initial latency in media servers. *IEEE Multimedia*, 4(3):50–61, July-September 1997.

[7] P.-C. Chang. Interleaving and error concealment for mpeg video over atm networks. *SPIE, Issue 1898*, pages 271–282, 1996.

[8] I. Dalgic and F. Tobagi. Characterization of quality and traffic for various video encoding schemes and various encoder control schemes. *Stanford Technical Report CSL-TR-96-701*, August 1996.

[9] I. Dalgic and F. Tobagi. Performance evaluation of ethernets and atm networks carrying video traffic. *Stanford Technical Report CSL-TR-96-702*, August 1996.

[10] J. Danskin, G. Davis, and X. Song. Fast lossy internet image transmission. *ACM Conference on Multimedia*, pages 321–332, 1995.

[11] D. Eckhardt and P. Steenkiste. Measurement and analysis of an in-building wireless network. *Computer Communication Review*, pages 243–43, August 1996.

[12] S. Hemami and R. Gray. Reconstruction-optimized lapped orthogonal transforms. *IEEE International Conf. on Acoustics Speech and Signal Processing*, 3:1542–45, 1996.

[13] S. Hemami and T. Meng. Transform coded image reconstruction exploiting interblock correlation. *IEEE Transaction on Imagine Processing*, 4(7):1023–27, July 1995.

[14] S. Lin and D. Costello. *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, 1983.

[15] T. Meng. Portable video-on-demand in wireless communication. *Proceedings of the IEEE*, 83(4), April 1997.

[16] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann, 1996.

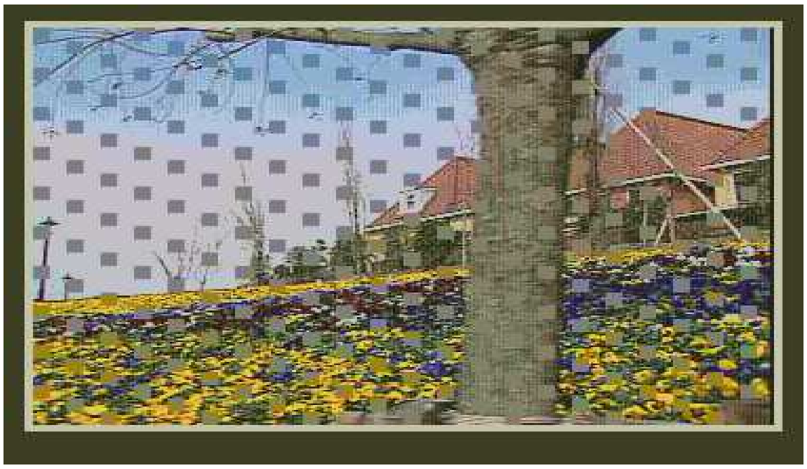


Figure 7: 20% Damaged Image Without Coefficients Spread Out

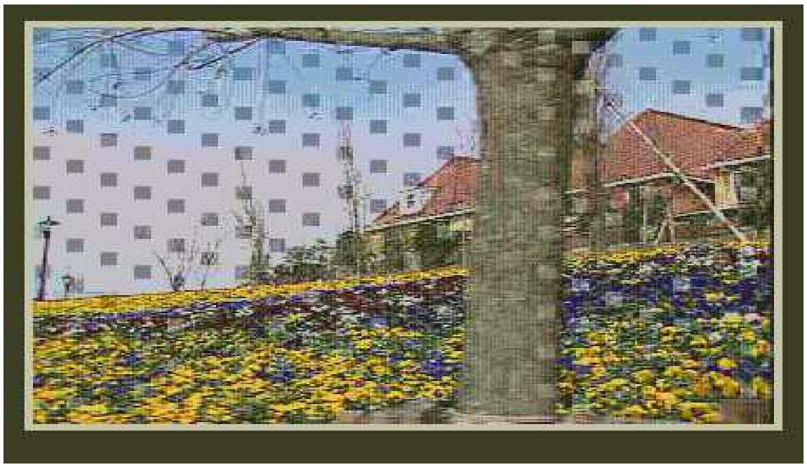


Figure 8: 20% Damaged Image With Coefficients Spread Out

Figure 9: Image Reconstructed from Figure 7

Figure 10: Image Reconstructed from Figure 8

Figure 11: Original Undamaged Image

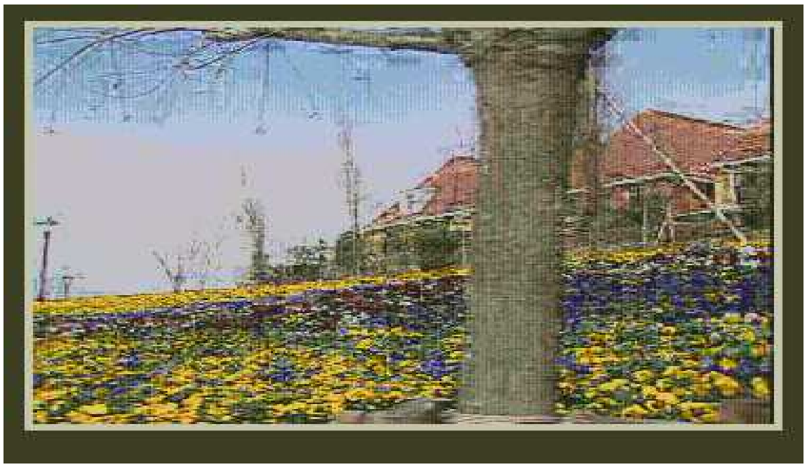


Figure 12: Reconstructed Image from 30% Loss

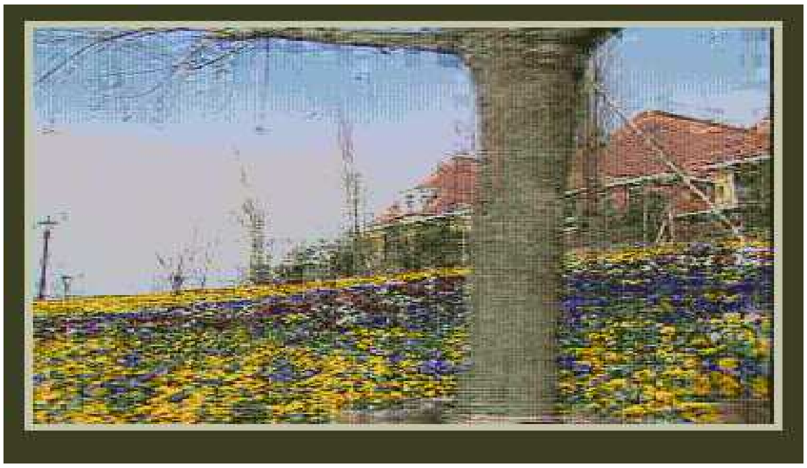


Figure 13: Reconstructed Image from 40% Loss

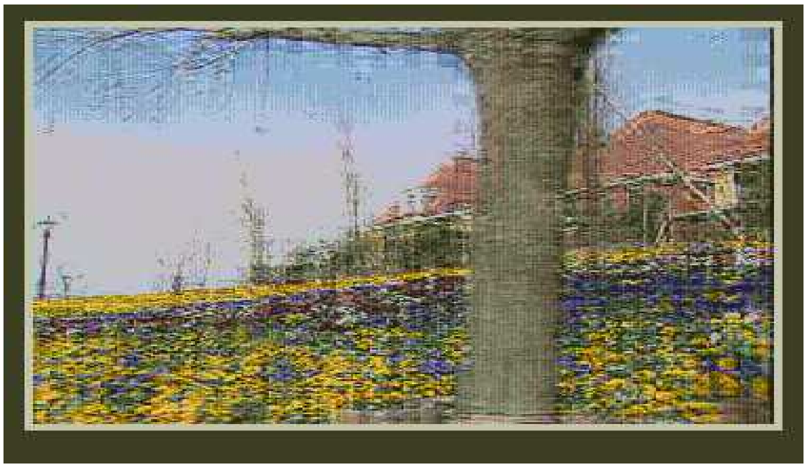


Figure 14: Reconstructed Image from 50% Loss