SELECTIVE FEC FOR ERROR-RESILIENT IMAGE CODING AND TRANSMISSION USING SIMILARITY CHECK FUNCTIONS

Tuyet-Trang Lam[1], *Lina J. Karam*[1], *Rida A. Bazzi*[2], *Glen P. Abousleman*[3]

[1]Electrical Engineering Dept., Arizona State University, Tempe, AZ 85287-5706.
[2] Computer Science & Engineering Dept., Arizona State University, Tempe, AZ 85287-5406.
[3] Compression, Comm. & Intelligence Lab, General Dynamics C4 Systems, Scottsdale, AZ 85257.
lamsnow@asu.edu, karam@asu.edu, bazzi@asu.edu, glen.abousleman@gdds.com

ABSTRACT

This paper introduces the concept of similarity check functions that measure the degree of corruption in transmitted multimedia data packets. Rather than directly discarding or directly retransmitting erroneous packets, the degree of corruption contributed by a particular packet is measured by a similarity check function at the receiver, which does not require explicit knowledge of the source data. Accordingly, if the packet is found to be badly corrupted based on a specified similarity criteria, it can be considered as a lost packet and recovered using forward error correcting codes or retransmission. An example of a similarity check function design and a selective forward error correction (S-FEC) scheme that allocates channel protection bits only to those packets that are in most need of correction, is presented to illustrate the proposed concept.

1. INTRODUCTION

In typical data transmission systems, the original source file is compressed, packetized, and transmitted through a noisy channel. Because bit corruption can cause a catastrophic loss of information, the packets are protected with channel codes, and error detection is performed on each packet at the receiver. However, existing error detection schemes do not provide information about the effects of a corrupted packet on the quality of the reconstructed data. Bit errors on significant bits may propagate throughout the reconstructed data, while bit errors on less significant bits may not present visible distortions.

In popular joint source-channel coding schemes [1, 2], the noisy channel is transformed into a packet erasure channel by dropping all packets that cannot be corrected. Reed-Solomon (RS) erasure codes are then used to recover missing packets [3, 4]. Typically, rate-compatible punctured convolutional codes (RCPC) are used to correct packets from random bit errors, while the packets that remain corrupted are detected using a cyclic redundancy check (CRC) code. If the CRC fails, the packet is declared to be erroneous and is discarded. In [1], it has been shown that combining RCPC/CRC with a fixed-rate Reed-Solomon erasure code performs better than using RCPC/CRC only. Unfortunately, a fixed rate RS code cannot recover an arbitrary number of lost packets. This problem has been addressed in [2], where each packet is protected against scattered bit errors by using an RCPC-CRC row code, where the source symbols are distributed across packets, and protected fully using RS codes of different strengths according to the importance of the source symbol. The amount of protection is given by an algorithm that uses the characteristics of the channel to optimally allocate the correct amount of RCPC/CRC and RS codes. Additionally, optimal source-channel rate allocation algorithms have been improved for packet loss over Rayleigh fading channels as shown in [3, 4].

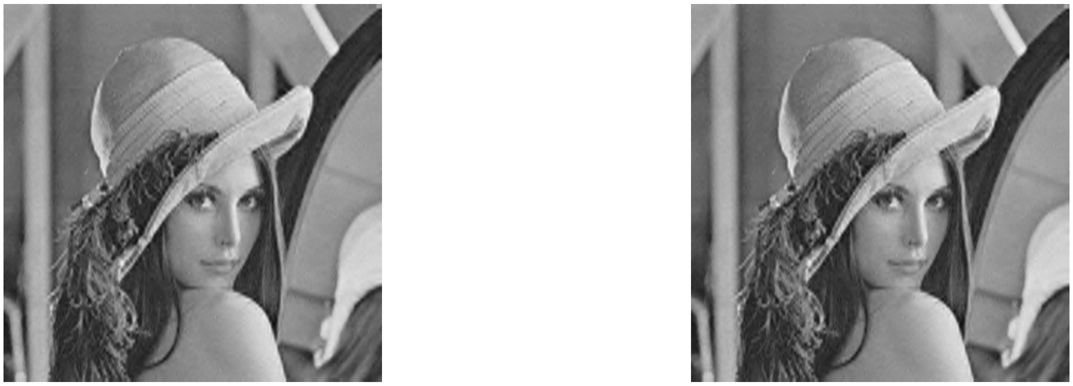
In our proposed scheme, rather than directly discarding erroneous packets, the degree of corruption is measured using a similarity check function that does not require explicit knowledge of the original data at the receiver. Accordingly, if the packet is found to be badly corrupted based on a specified similarity criteria, it can be considered as a lost packet and recovered using forward error correcting codes (FEC). If the applied similarity check function indicates that the degree of corruption of the considered packet is acceptable, the packet is still usable and no FEC is needed. In this way, the total amount of FEC needed to protect the transmitted data can be decreased, and more bits can be allocated to the source coding for enhanced quality.

This paper is organized as follows. Section 2 describes the new concept of the similarity check function. This function can be used to intelligently allocate channel protection bits to packets that most need to be corrected rather than protecting all packets. Section 3 illustrates the proposed selective-FEC scheme using a particular similarity function design and the wavelet-based TCQ image coder presented in [5]. A conclusion is given in Section 4.

2. SIMILARITY CHECK FUNCTION

The proposed concept of a similarity check function is motivated by the use of hash functions to provide fault toler-

0-7803-8554-3/04/$20.00 ©2004 IEEE.



(a) No noise, PSNR $= 29.88$ dB (b) BSC with BER$= 0.001$, PSNR$= 27.34$ dB

Fig. 1. 512×512 Lena image coded at 0.246 bpp.

ance and security in a distributed storage environment [6]. A hash is computed to uniquely represent a file (or a piece of a file) using only a few bits. It can be thought of as a digital fingerprint of a larger document [7]. Because the hash is much smaller than the original file, it is more efficient to fully protect the hash than the corresponding file. The protected hash is sent along with the file. When the file is received, a new hash is computed from that file. If the received file is identical to the original file, the two hashes will match. If the hashes do not match, the received file is discarded. Similarly, our similarity check function is designed such that, when applied to a piece of data (or packet), it results in an output (i.e., an index) that can be represented with an insignificant number of bits relative to the original piece of data, without the need to have the original piece of data. However, while a hash can only indicate whether there is an exact match between the received and transmitted pieces of data, and thus, cannot provide information about the amount of corruption introduced in a file, the proposed similarity check function is designed to measure the degree of corruption introduced. For multimedia applications, a file does not need to be exactly identical to the original data to be usable, which motivates the proposed similarity check function general framework. Note that a CRC code or a hash function can be considered as a special similarity check function with the similarity criteria set to be the identity function.

Let the original data be transmitted (or stored) as N packets (or pieces), $p_1, \cdots, p_N$, each of size L. Let $p'_1 \cdots, p'_N$ be the corresponding received packets. Let P denote the set containing all packets (data pieces) of size L. The similarity check function is defined as a mapping from the set P to the set $I = \{I_1, \cdots, I_r\}$ as follows:

$$f(p \in P) = I_l,\ l \in \{1, \cdots, r\}, \qquad (1)$$

where I_l in (1) is referred to as the similarity check index. Note that only $c = \log_2(r)$ bits are needed to represent elements in the set I, and c is selected to be very small compared to the packet size. Two packets, p_k and p'_k, are considered similar if and only if $f(p_k) = f(p'_k)$.

The similarity check function can be designed based on MSE or perceptual criteria. For example, Fig. 1 shows the 512×512 Lena image compressed with the wavelet-based TCQ image coder of [5] at 0.25 bits per pixel (bpp) and transmitted through an ideal channel with no noise (Fig. 1a), and through the binary symmetric channel (BSC) with bit error rate (BER), $P_b = 0.001$ (Fig. 1b). For this image, 71 bits have been corrupted, resulting in 65 corrupted packets. Despite the fact that the PSNR for the image in Fig. 1b is 2 dB lower than the image in Fig. 1a, both images are very similar. Thus, it would be wasteful to discard the image in Fig. 1b because it is not identical to the image in Fig. 1a.

The similarity check function can be used to estimate the amount of protection needed to receive acceptable data. In the proposed selective-FEC scheme, the similarity check function is used to evaluate the number of packets to be protected. Then, rather than correcting all corrupted packets, only packets that result in a high degree of degradation will be corrected. In this S-FEC method, the received data will not be exactly the same as the transmitted data, but annoying noise artifacts can be corrected. Note that if not enough channel coding is assigned to fully recover all severely corrupted or lost packets, the similarity check function can also be designed to provide usable information to correct a portion of the lost packets.

3. CODING EXAMPLE USING A SIMILARITY CHECK FUNCTION AND SELECTIVE FEC

In the following coding example, the wavelet-based TCQ coder of [5] is used to code 512×512 images, which are then transmitted through the BSC channel with a bit error rate, P_b. The coder in [5] has been modified to perform a

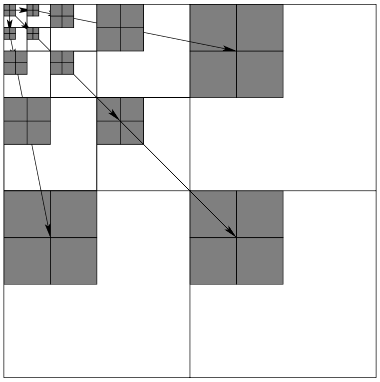


Fig. 2. Packetization of wavelet coefficients for a 13 sub-band decomposition.

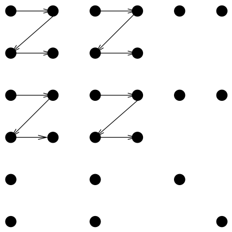


Fig. 3. Scanning of coefficients from the LL sub-band.

4-level dyadic wavelet decomposition to simplify the packetization. Each packet consists of 4 quantized wavelet coefficients from the zero-mean LL sub-band, and their corresponding children in the wavelet decomposition, as shown in Fig. 2. To limit the propagation of bit errors across packets, coefficients from the lowest sub-band are scanned, as shown in Fig. 3, and coded using trellis-coded quantization (TCQ).

Since the coefficients from the LL sub-band are more important than coefficients from the other sub-bands, a simple similarity check function can be designed as follows. Let $\mathbf{v_{LL}}$ denote a vector consisting of the 4 quantized LL coefficients in a packet, p. $\mathbf{v_{LL}}$ can be classified into two classes with centroids, $\mathbf{c_0}$ = [-1 -1 -1 -1] and $\mathbf{c_1}$ = [1 1 1 1]. The similarity check function in this example is defined as

$$f(p) = \begin{cases} 1, & \text{if } \|\mathbf{v_{LL}} - \mathbf{c_1}\| < \|\mathbf{v_{LL}} - \mathbf{c_0}\| \\ 0, & \text{otherwise,} \end{cases} \qquad (2)$$

where $\|.\|$ is the L_1 norm. The S-FEC-based joint source-channel coding scheme proceeds as follows:

1. At the transmitter side, for each packet, p_i, compute the similarity check function as in (2). This will result in a 1-bit output (similarity check index), $I_{p_i} \in \{0, 1\}$.
2. Send the resulting 1-bit similarity check indices, $\{I_{p_i}\}$, to the receiver along with the data packets. Note that these constitute an insignificant portion of the bit budget and can be protected fully without significant added overhead.
3. At the receiver, for each received packet, p'_i, compute the similarity check function as in (2), and compare with the corresponding transmitted similarity check index, I_{p_i}: if $\{f(p_i) \neq f(p'_i)$, then p'_i is discarded$\}$ else $\{p'_i$ is kept and used to reconstruct the image$\}$. Error correcting codes can then be used to recover the discarded packets.

Table 1. Received 512×512 Lena image coded at 0.245 bpp using the wavelet-based TCQ coder in [5].

BSC BER	unprotected PSNR	corrupted packets	protected packets	S-FEC PSNR
0	29.85	0%	0%	29.85
0.001	27.76	23.1 %	0.08 %	29.6
0.005	23.43	71.9 %	2 %	28.36
0.01	21.38	92.2 %	3.1 %	27.21
0.05	15.61	100 %	9.4 %	23.16
0.1	12.93	100 %	14 %	21.11

One advantage of this implementation is that very little overhead is needed to perform packet loss detection as compared to typical CRC codes that can take 8 to 16 bits. If N is the number of packets, L is the packet length, and c is the number of bits output by the similarity check function per packet ($c = 1$ in the example above), the overhead produced by the S-FEC scheme is computed as:

$$\text{overhead } = \lambda LN + 2cN \text{ bits,} \qquad (3)$$

where λ is the maximum fraction of discarded dissimilar packets, and assuming that a rate-1/2 code is used to protect the similarity check indices.

Table 1 shows the number of corrupted packets for the 512×512 grayscale Lena image compressed at 0.246 bpp using the wavelet-based TCQ coder in [5]. The results in this table were obtained by averaging over 10 simulations. The percentage of protected packets is given by the maximum percentage of dissimilar packets over all simulations for a given bit error rate. As shown in Table 1, for the BSC with a bit error rate of 0.01, the proposed S-FEC scheme results in a 6 dB increase in peak signal-to-noise ratio (PSNR) by protecting only 3.1% of the packets, while 92.2% of the received packets are corrupted.

Fig. 4 provides a comparison of the proposed S-FEC scheme with the conventional FEC scheme used in [8]. In [8], the output bit stream is partitioned into blocks of length $L = 200$ bits, with $c = 16$ appended parity bits, and $m = 6$ extra bits to terminate the convolutional codes. For the BSC with a bit error rate, $P_b = 0.001$, a rate-8/9 RCPC code with memory 6 was then used on the obtained block of $N + c + m$ $(= 200 + 16 + 6 = 222)$ bits, which results

(a) Not protected, PSNR = 26.56
source coding: 0.246 bpp (100%)
channel coding: 0 bpp (0%)

(b) FEC from [8] , PSNR = 28.65
source coding: 0.1875 bpp (75%)
channel coding: 0.0625 bpp (25%)

(c) Proposed S-FEC, PSNR = 29.56
source coding: 0.246 bpp (98%)
channel coding: 0.005 bpp (2%)

Fig. 4. PSNR (in dB) of 512×512 Lena image coded at 0.25 bpp over BSC channel with bit error rate, $P_b = 0.001$.

in a total channel coding overhead of 25%. Fig. 4 shows the received Lena image coded at 0.25 bpp using the wavelet-based TCQ coder over the BSC with a bit error rate of 0.001. We can see that the noisy channel introduces severe impulsive artifacts, as shown in Fig. 4a, resulting in a drop in PSNR to 26.56 dB. Figs. 4b and 4c show the 512×512 Lena image obtained using the FEC scheme of [8] and our proposed S-FEC scheme, respectively. The proposed S-FEC scheme results in a PSNR of 29.56 while the FEC scheme of [8] results in a PSNR of 28.65 dB. In addition, the image produced using the scheme of [8] is more blurred as compared to the one obtained using the proposed S-FEC scheme. This is due to the fact that the scheme of [8] allocates 25% of the total bit rate to the channel coding, leaving only 75% of the total bit rate to the source coding. In contrast, the proposed S-FEC scheme allocates only 2% of the total bit rate to the channel coding, while the remaining 98% is used by the source coder, which results in a sharper image and higher PSNR. For our simulations, the 2% channel coding overhead was computed using (3) for the proposed scheme. In addition, all the corrupted packets were assumed to be exactly recovered by the channel coding for the result of Fig. 4b, while only the non-similar corrupted packets were assumed to be recovered for the result of Fig. 4c.

4. CONCLUSION

In this paper, we have presented the concept of a similarity check function that measures the amount of corruption introduced in an image without explicit knowledge of the original data at the receiver. This similarity check function is used to detect severely corrupted packets in the proposed S-FEC scheme and can, thus, allow a fast and intelligent allocation between source and channel bits. Future work will consider the design of similarity check functions that are optimized based on MSE or perceptual distortion measures, and the design of practical selective forward error correcting codes that are optimized based on similarity check functions. We will also consider channels with burst errors such as Rayleigh-fading channels.

5. REFERENCES

[1] P.G. Sherwood and K. Zeger, “Error protection for progressive image transmission over memoriless and fading channels,” in *IEEE ICIP*, Oct. 1998.

[2] D. Sachs, A. Raghavan, and K. Ramchandran, “Wireless image transmission using multiple-description based concatenated codes,” *IEEE DCC*, p. 569, Mar. 2000.

[3] A. Mohr, E. Riskin, and R. Ladner, “Unequal loss protection: graceful degradation of image quality over packet erasure channels through forward error correction,” *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 819–828, June 2000.

[4] J. Kim, R. Mersereau, and Y. Altunbasak, “Error-resilient image and video transmission over internet using unequal error protection,” *IEEE Trans. on Image Processing*, vol. 12, no. 2, Feb. 2003.

[5] T.-T. Lam, G.P. Abousleman, and L.J. Karam, “Image coding with robust channel-optimized trellis-coded quantization,” *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 940–951, June 2000.

[6] D. MacKay, *Information theopry, Inference and learning algorithms*, chapter 12, Cambridge University Press, 2003.

[7] R. Rivest, “The MD5 message digest algorithm,” Internet RFC 1321, Apr. 1992.

[8] P. G. Sherwood and K. Zeger, “Progressive image coding for noisy channels,” *IEEE Signal Processing Letters*, vol. 4, pp. 189–191, July 1997.