Unequal Protection of JPEG2000 Code-Streams in Wireless Channels

Ambarish Natu and David Taubman

School of Electrical Engineering and Telecommunications
University of New South Wales, Sydney
Australia.

Email: a.natu@ee.unsw.edu.au, d.taubman@unsw.edu.au

Abstract- The rapid growth of wireless communication has resulted in a demand for robust transmission of compressed images over wireless channels. The challenge of robust transmission is to protect the compressed image data against loss, in such a way as to maximize the received image quality. The paper addresses this problem; investigating unequal error protection of JPEG2000 compressed imagery. More particularly, the results reported in this paper provide guidance concerning the selection of JPEG2000 coding parameters and appropriate combinations of RS (Reed-Solomon) codes, for typical wireless bit error rates in the range 10^{-4} to 10^{-3} .

1. INTRODUCTION

This paper is concerned with the development of unequal error protection schemes for JPEG2000 compressed imagery. More particularly, we are interested in maximizing the received image quality, in the presence of random bit errors. Such conditions might be expected in the context of wireless image transmission.

The JPEG2000 standard offers a variety of coding modes and syntax features, which may be used to provide some degree of protection against errors. Most notable amongst these are: a) the option to include resync markers in the code-stream; b) the ability to modify the arithmetic codeword segments so as to allow detection and concealment of errors in individual code-blocks; and c) the ability to partition the compressed data into independently decodable elements. Although these techniques help, they are unable to recover lost data. Error protection schemes were deliberately excluded from the scope of JPEG2000 Part 1, but some effort was invested in ensuring that the standard would not preclude the later development of effective error protection mechanisms.

In [5], Hamming Codes are used to provide unequal error protection for JPEG2000 code-streams, using the same memoryless channel model as in the present work. In fact, this work had a significant influence on the strategy for embedding resync markers in JPEG2000 code-streams [7]. In [6], turbo codes are proposed to protect JPEG2000 code-streams. These earlier works do not consider the problem of optimizing the JPEG2000 coding parameters themselves; nor do they consider the application of different levels of protection to different quality layers in the code-stream. Our selection of Reed Solomon (RS) codes for the present work is

motivated by their ability to offer strong protection without excessive complexity. The use of block codes particularly simplifies switching between different codes to protect different parts of the code-stream unequally.

In section 2, we give a brief description of the existing wireless channel models. In section 3, properties of the RS codes are mentioned. In section 4, a high level description of the JPEG2000 is presented. In section 5, existing error resilience tools are briefly mentioned along with a description concerning the selection of JPEG2000 coding parameters. Section 6, presents the results and interpretation and section 7 provides conclusion.

2. WIRELESS CHANNELS

Wireless channels are characterized by their limited bandwidths and high bit error rates. Without protection they cannot provide the necessary quality of service guarantees for compressed image data. Two models that exist in developing robust image transmission are: 1) *bit error loss* models and 2) *packet loss* models. *Bit error* models assume random bit errors, occurring at some specified bit error rate (BER). On the other hand *packet loss* models assume that the data is partitioned into either fixed- or variable-length packets. Commonly it is assumed that lost packets are detected, and a lost packet does not disrupt the reception of subsequent packets. Such a model is valid for a wireless channel when Forward Error Correction (FEC) codes are applied in the transport layer to protect the contents of each packet; when the capabilities of the FEC codes are exceeded, the packet is considered lost. For this work, we restrict our attention to the bit error models, assuming a memoryless error process. This allows us to tailor the FEC codes to the properties of the compressed data source that is being protected.

3. REED SOLOMON CODES

Reed Solomon (RS) codes [1] are a special case of BCH codes. An (n, k) RS code takes a group of k data symbols and generates $n-k$ parity symbols; n is the codeword size, measured in symbols. The error correcting capability of the code is directly related to the number of symbols, which differ between any two different codewords. In particular, a t -error correcting RS code from the Galois Field $GF(2^m)$ has the following parameters.

Number of codeword symbols = $n = 2^m - 1$.

Number of parity symbols = $n - k = 2 \times t$.

Here m is the number of bits per symbol and t is the number of symbols, containing one or more bit errors, which the code can correct.

For the present study, we have implemented the RS codes over GF(16). This means that each symbol is a *nibble*, having $m = 4$ bits. Thus, for example, the (15,9) code produces codewords with 15 nibbles (60 bits) each, and is able to locate and correct errors in any 3 nibbles.

The RS family of codes provides a wide range of codes with different code lengths and error correction capabilities, from which the most appropriate codes may be selected. The RS codes are simpler to decode than turbo or convolutional codes. When an error cannot be corrected, $m \times k$ original source bits are lost together; however, for the current application, loss of multiple consecutive source is not substantially worse than the loss of a single source bit, since the code-stream elements which are being protected generally exhibit at least local dependencies, which usually render the bits immediately following a corrupted bit useless. In this paper we investigate the use of (15,7), (15,9), (15,11) and (15,13) RS codes, for the protection of various elements in the JPEG2000 code-stream.

4. JPEG2000

The JPEG2000 codec borrows heavily on ideas from the EBCOT paradigm [2]. Although JPEG2000 is able to process an image in smaller independent tiles, we are not interested in this feature for the present study. Instead, the entire image is processed as a single tile. Its samples are transformed into spatial frequency subbands with the aid of the Discrete Wavelet Transform (DWT). Each subband is partitioned into small blocks, known as *code-blocks*. Initial quantization and bit-plane coding are performed on these code-blocks. The bit-plane coder makes three passes over each magnitude bit-plane of the quantized subband sample indices. Each code-block is coded independently.

Each resolution in the DWT is partitioned into so-called *precincts*. Precincts have no impact on sample data transformations or coding. Precincts are not a partition of image data, but of the compressed code-block bit-streams, which are used to reconstruct that resolution. The precinct size can be chosen independently for different resolutions. Compressed data from each precinct is collected into so-called *packets* (the use of the term *packet* here is not to be confused with the packets which might be employed by a network transport layer). Each packet consists of a packet head and a packet body, which together identify and contain incremental contributions from the code-block bit-streams belonging to the relevant precinct. In order to extract the code-block bit-stream contributions from a packet body, it is necessary to correctly decode the header of that packet and all preceding packets from the same precinct.

Figure 1: Code-stream Structure.

Figure 1 illustrates the organization of a JPEG2000 code-stream, having only one tile and one tile-part. As seen in the figure, the code-stream commences with a sequence of marker segments, each of which commences with a specific marker code and contains parameters describing the coding options, which have been employed. The last of these marker segments is followed by a concatenated list of code-stream packets, each with its own header, as mentioned previously.

5. ERROR RESILIENCE IN JPEG2000

Partitioning the code-stream into different segments helps to isolate errors in one segment and prevent them from propagating through the entire code-stream. For the purpose of this investigation, we assume that all of the marker segments which precede the concatenated list of code-stream packets remain intact. As already mentioned, these marker segments contain crucial coding parameters, without which nothing can be decoded.

JPEG2000 provides several error resilience tools, to help minimize the impact of corruption in the packet data. One of these tools is the *ETERM* predictable termination strategy. When the arithmetic codeword generation process, used to produce each code-block bit-stream, is terminated at the end of each coding pass, using this predictable termination

strategy, a suitably equipped decoder can localize any errors encountered in the bit-stream with reasonably high confidence. Once an error is detected, the decoder can discard the affected coding pass and all future coding passes, thereby minimizing the impact of the error upon reconstructed image quality. We refer to this strategy as *error concealment*.

JPEG2000 also offers an additional feature to assist the decoder in localizing errors in a code-block's bit-stream. In this case, a special four-symbol code, known as a *SEGMARK*, is inserted immediately before the first new coding pass in each magnitude bit-plane. A bit error in any of the preceding coding passes is likely to corrupt at least one of the four *SEGMARK* symbols, allowing the decoder to detect the error. When used together, these two mechanisms allow most code-block bit-stream errors to be detected and concealed.

As discussed earlier the main component of the code-stream is the packet. Encoding and decoding of code-blocks are independent processes, so bit errors in the bit stream of any one code-block will not affect other code-blocks. The code block contributions inside a packet body can be independently decoded. However, if a packet header is corrupt, the code-block contributions from that packet's body cannot generally be correctly recovered. Moreover, subsequent packet headers for the same precinct are often not decodable, without the previous ones. This suggests that errors in a packet header can be expected to have a more devastating impact on image quality than errors in the packet body. For this reason, our experimental investigation considers the benefits of protecting packet headers with stronger codes than the corresponding bodies.

Partitioning resolutions into multiple precincts can also contribute to error resilience. The use of smaller precincts results in a large number of packets being formed. This also results in an increase in the total amount of packet header information, which slightly reduces the overall compression efficiency. On the other hand, selecting smaller precincts substantially reduces the number of code-blocks affected by a single packet header error. For this reason, one would expect to find that smaller precincts yield more robust code-streams. While this appears to be true when the data is not protected by FEC codes, our current experience suggests that the impact of smaller precincts on error resilience may not be significant when FEC codes are carefully assigned to the protection of the various packet header and body components.

It is worth pointing out that it is not sufficient simply to apply an FEC coding strategy directly to the compressed packet data bytes [4]. The reason for this is that no two consecutive packet data bytes are permitted to form a code in the range FF90 through FFFF, this range being restricted for the signaling of code-stream markers. In particular, the resync marker (*SOP=FF91*) plays an important role in error resilient parsing of the code-stream. *SOP* marker segments

may optionally be inserted in front of each code-stream packet. In the event of a corrupt packet header, the length of that packet's header and/or body are likely to be misread, so that the next packet's *SOP* marker will not be encountered in the expected location. The *SOP* marker segment contains a sequence number, which may be used to recover from errors of the form described above.

For the present study, the RS encoded data is used to build a modified code-stream, whose packet data consist of the codewords produced by the RS coder. Packets are then aligned on RS codeword boundaries, rather than byte boundaries. The parity bits generated by RS coding may result in packet data, which violates the requirement that no two consecutive bytes may form a value in the range FF90 to FFFF. To avoid this problem, the parity bits are collected into multiple of 7, packing each such multiple into a whole packet data byte, whose most significant bit is 0. Of course, the result is not a legal JPEG2000 code-stream, but it is still decodable by a slightly modified decoder. For practical applications, it would be better to pack the parity bits into a completely separate data stream with its own resync markers. Our packing will still work in this case and results should be similar to those obtained here.

Another factor contributing to the error resilience of a JPEG2000 code-stream is the number and spacing of *quality layers*. The layering concept imparts quality scalability to JPEG2000. The first layer in a code-stream is the collection of all first packets from each precinct in the image. The second layer consists of the second packet from each precinct, etc. The code-block contributions associated with each quality layer successively reduce the image distortion in an optimal way, as a function of the number of packet bytes in those layers.

The six-layered code-streams used in this paper have cumulative sizes corresponding to overall image bit-rates of 0.03125, 0.0625, 0.125, 0.25, 0.5 and 1.0 bits per sample. Since the information in packets belonging to layer 1 is required to decode the information in layer 2 and so forth, the sensitivity of the data to corruption clearly increases as we move from lower to higher quality layers. As a result, we would expect to find that a protection scheme which assigns stronger FEC codes to the lower layers than the higher layers can outperform a uniform protection strategy, assuming a fixed total coded data size.

By adjusting the number of quality layers we can vary the number of packets that are formed. More significantly, by spacing the cumulative layer sizes more closely, each layer and each precinct packet represent a smaller incremental contribution to the image quality.

6. RESULTS AND INTERPRETATION

To investigate the impact of various RS coding strategies on error resilience, we work with Kakadu v3.2, modifying

this implementation so as to perform the coding and parity bit packing, and also to account for the impact of coding on rate control. We simulate a Binary Symmetric Channel (BSC) with bit error rates (BER) of 10^{-4} and 10^{-3} . Error rates of these magnitudes are typical in wireless communication systems. The 2560×2048 monochrome test image "WOMAN" is coded to a total size corresponding to 1.0 bits per sample, including the cost of all FEC codes. All noisy simulations are repeated 100 times for each set of test conditions. Image quality is assessed in terms of the Mean Squared Error (MSE), taken over all 100-simulation runs, and expressed in terms of the PSNR¹.

Tables 1 shows the results of our simulations, using only the existing error resilience features offered by JPEG2000. In particular, we compare the reconstructed image quality obtained when no error resilience features are enabled, with the quality obtained when resync (SOP) marker segments are inserted, and then when both resync markers and decoder error concealment are employed. In the last case, both the ERTerm and SEGMark features mentioned previously are employed when constructing the original code-stream.

Table 1: PSNR (dB) results for test image "Woman", coded at 1.0 bpp and 6 quality layers, using only existing JPEG2000 error resilience tools

BER	No Conceal.; No Resync	No Conceal., Resync	Concealment, Resync
Noiseless	38.40 dB	38.40 dB	38.38 dB
10^{-4}	11.01 dB	18.44 dB	25.31 dB
10^{-3}	7.85 dB	11.09 dB	18.68 dB

Table 2 shows the results of our simulations, for two possible precinct size combinations. We also compare the reconstructed image quality of a code-stream when only one quality layer is used, with the quality obtained when the code-stream is organized in six quality layers. From Table 1 and Table 2 it becomes clear that using multiple precincts can improve the error resilience of an unprotected JPEG2000 code-stream. While multiple quality layers have some benefit, this benefit largely disappears when multiple precincts are also used. As we shall see, the real advantage of layers for error resilience is realized only when we introduce error correction codes.

We may categorize FEC protection schemes as either *Equal Protection* (EP) schemes or *Unequal Protection* (UP) schemes. EP schemes assign the same amount of protection to the entire code-stream. On the other hand, UP schemes provide different levels of protection to different elements in

Table 2: PSNR results for test image "Woman", coded at 1.0 bpp, using only existing JPEG2000 error resilience tools such as variable precincts sizes and additional quality layers.

BER	No. of layers = 1		No. of layers = 6	
	Precinct sizes {256,128,64}	Single precinct	Precinct sizes {256,128,64}	Single precinct
Noiseless	38.38 dB	38.41 dB	38.24 dB	38.38 dB
10^{-4}	26.86 dB	23.15 dB	26.89 dB	25.31 dB
10^{-3}	19.43 dB	17.13 dB	19.46 dB	18.68 dB

Table 3: PSNR (dB) results for test image "Woman" coded at 1.0 bpp and 6 quality layers using EP with RS Codes.

BER	Scheme-1	Scheme-2
Noise-less	34.34 dB	36.87 dB
10^{-4}	34.25 dB	30.58 dB
10^{-3}	27.70 dB	20.15 dB

the code-stream; data which is more sensitive to errors is protected more heavily than the less sensitive data.

Table 3 shows results obtained using two different protection schemes. For *Scheme-1*, the (15,9) RS code is applied to all packet data uniformly. In *Scheme-2*, all packet data is uniformly coded with the weaker (15,13) code. For both of these EP schemes we employ resync markers together with the error concealment tools mentioned previously. Comparing Table 1 with Table 3 we see that RS coding has a considerable impact on the reconstructed image quality.

As noted above, we would expect to benefit from the layered prioritization of information in a JPEG2000 code-stream, by protecting earlier quality layers with stronger codes than the later layers. Table 4 gives results for such a scheme, *Scheme-3*. Among the several schemes investigated we find that, if we code the first three layers using the (15,9) code, the next 2 layers with the (15,11) code and the last layer with the (15,13) code, we can achieve improved results at lower bit error rates. Also worth noting is the fact that the noiseless compression performance improves substantially. This is because the strongest codes are used to protect only the initial quality layers, which contain many fewer data bytes than the later layers, which are protected with less redundancy.

Table 4: PSNR (dB) results for test image "Woman" coded at 1.0 bpp using UP with RS Codes

Scheme-3	BER	PSNR
Scheme -3 with Concealment	Noiseless	36.04 dB
	10^{-4}	35.05 dB
	10^{-3}	26.91 dB
Scheme-3 without Concealment	Noiseless	36.06 dB
	10^{-4}	32.13 dB
	10^{-3}	23.33 dB

¹ PSNR (Peak Signal to Noise Ratio) is defined as $10 \log_{10}(P^2 / \text{MSE})$, where P is the peak-to-peak signal amplitude. In this case, $P = 255$, since we are working with 8-bit images.

Of course, at higher bit error rates, the fact that we have protected some quality layers with weaker codes in Scheme-3 than in Scheme-1 can damage the PSNR performance. This can be seen by comparing Tables 3 and 4. For higher BER conditions, an unequal error protection scheme is still preferred, but we must select yet stronger codes for the initial quality layers. Results for such a scheme, *Scheme-4*, are presented in Table 5. For this scheme, we code the first layer with a (15,7) code, the next three layers with the (15,9) code and the final two layers with a (15,11) code. This strategy was found to yield the best results, out of several schemes investigated at the higher BER. Comparing the results presented in Tables 3 and 5, we see that the use of unequal error protection can improve the PSNR at high error rates, while also improving the noiseless compression efficiency.

Table 5: PSNR (dB) results for test image "Woman" coded at 1.0 bpp using UP with RS Codes.

BER	Scheme-4
Noise-less	35.23 dB
10^{-4}	34.70 dB
10^{-3}	28.20 dB

Interestingly, the performance of *Scheme-4* at the lower BER of 10^{-4} is not dramatically worse than that of *Scheme-3*. This suggests that simple code assignments can be devised which are able to offer reasonable efficiency simultaneously over a wide range of bit error rates, without excessively damaging the noiseless compression efficiency.

We also observe that spacing layers more closely (two layers for each factor of 2, change in the cumulative layer bit-rate), we can improve the image quality by 1 dB at lower bit error rates as compared to scheme-1 and by 0.2 dB as compared to scheme-3.

In compiling the results presented in Tables 3 through 5, three different combinations of precinct dimension were considered: one precinct per resolution; the {256}, {128}, {64} combination used for Table 2; and a third combination in which all resolutions were partitioned into precincts measuring 128x128. Interestingly, we find that the use of multiple precincts has no significant benefit. In fact, image quality appears to deteriorate slightly when multiple precincts are used. This result appears to be at odds with that shown in Table 2 for the case where FEC codes are not employed. Upon closer investigation, we found that the loss in performance associated with multiple precincts is due to the additional cost associated with independently coding each packet header and aligning the packet on a whole codeword boundary. This cost reduces the noiseless compression performance by 0.3dB and 0.5dB, respectively, for the {256}, {128}, {64} and 128x128 precinct cases considered. While some of this performance loss is generally recovered when bit errors are introduced, the effect is insufficient to warrant a conclusion that multiple precincts are beneficial in the context of FEC coding.

In the lower portion of Table 4, we present additional results showing that error concealment tools play an important role, even when FEC codes are employed to protect against loss.

7. CONCLUSION

In this paper, we have examined firstly the impact of the existing error resilience tools offered by JPEG2000, including error concealment, resync markers, layering and precincts. We find that these tools are of substantial benefit and that dividing the image resolutions into multiple precincts is advantageous for error resilience.

The availability of multiple quality layers is of little benefit unless we introduce forward error correction codes, to protect the layers unequally. Using a family of RS block codes having codeword sizes of 15 nibbles (60 bits), we have found that JPEG2000 code-streams can be made substantially resilient to errors. Unequal protection of the quality layers is definitely beneficial and allows a code-stream to be efficiently protected over a range of different channel error conditions. To realize these benefits, one may get away with layers having octave bit-rate spacing, although somewhat better performance may be achieved if there are two layers for every factor of two changes in the bit-rate.

Somewhat counter-intuitively, we have not found the use of multiple precincts to be of any benefit when RS codes are used to protect the compressed data, either equally, or unequally across quality layers.

REFERENCES

- [1] Shu, L.; and Costello, D. Jr., *Error Control Coding: Fundamentals and Applications*, Englewood Cliffs, N.J.: Prentice-Hall, 1978
- [2] Taubman, D., "High Performance Scalable Image Compression with EBCOT," *IEEE Trans. on Image Processing*, vol. 9, no.7, pp. 1158-1170, July 2000.
- [3] Taubman, D.S. and Marcellin, M.W., *JPEG2000: Image Compression Fundamentals, Standards, and Practices*, Norwell, MA: Kluwer Academics Publishers, 2001.
- [4] Natu, Ambarish. S, *Error Resilience in JPEG2000*, Master of Engineering thesis. The University of New South Wales, Sydney, Australia. To be published.
- [5] Zhang, D, *Error Resilience in JPEG2000*, Bachelor of Engineering Thesis. The University of New South Wales, Sydney, Australia, 1999.
- [6] Banister, B.A.; Belzer, B.; Fischer, T.R., "Robust image transmission using JPEG2000 and turbo-codes," *Proceedings of the International Conference on Image Processing*, vol. 1, pp 375-378. 2000.
- [7] Taubman, D., "Elimination of Byte Stuffing for Error Resilience", ISO/IEC JTC1/SC29/WG1 N1480, Nov 1999