On Packetization of JPEG2000 Codestreams

Ali Bilgin, Zhenyu Wu and Michael W. Marcellin

Dept. of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85721

ABSTRACT

We introduce techniques that improve the error resilience of JPEG2000 against packet losses. The presented methods operate on JPEG2000 codestreams and consider the properties of different codestream segments. Experiments indicate that error resilience of JPEG2000 codestreams against packet losses can be improved significantly using these techniques. The performance of the proposed methods compares favorably with existing algorithms.

1. INTRODUCTION

Transmission of images over noisy channels is important for many applications. When network congestion occurs in packet switched networks, some of the transmitted packets are dropped. When images are transmitted over such networks, parts of the codestream might be unavailable at the decoder due to these losses. The decoder should be able to reconstruct the image using only the available packets. Many high performance coders of today rely on efficient entropy coders that are highly dependent on the state of the system. Thus, such coders are highly sensitive to errors in the codestream. If the codec is not designed carefully, a single error, i.e. a single packet loss, can result in disposal of the entire codestream.

Several different strategies exist for combating packet erasure. Retransmission protocols such as Automatic Repeat Request (ARQ) allow the decoder to request the retransmission of a lost packet. However, ARQ techniques introduce transmission delay that may be unacceptable in real-time applications. The requested packets may arrive too late to be useful in reconstruction. Furthermore, when the network experiences heavy congestion, retranmission requests increase, worsening the congestion. ARQ techniques are also unacceptable for applications where the receiver is unable transmit a message to the sender.

An alternative approach to achieve error resilience against packet losses is to partition the codestream into small segments that can be decompressed independently. Such methods have attracted considerable attention recently [1–3] . In [1] , a method that partitions wavelet coefficient trees into a fixed number of groups and compresses each group independently is proposed. Since groups are decompressed independently, errors are confined to a group and do not propogate beyond group boundaries. A similar method called Packetizable Zerotree Wavelet (PZW) was proposed in [2] . Here, the authors were able to form the packets in such a way that complete trees of wavelet coefficients were contained within a packet. In [3] , the authors address the problem of minimizing packetization inefficiency due to codestream alignment. They develop a theoretically optimal packetization scheme, and provide low-complexity suboptimal methods that achieve comparable performance. These strategies provide sufficient resilience when only a small fraction of the packets are lost. Their performance characteristics degrade quickly as packet losses increase.

To achieve robustness under high packet loss rates, Forward Error Correction (FEC) techniques can be used. These techniques introduce controlled redundancy into the codestream. This redundancy is exploited at the decoder to correct some of the transmission errors. The FEC techniques can be divided into two categories depending on their redundancy assignment strategies: Equal Loss Protection (ELP) and Unequal Loss Protection (ULP). ELP techniques assign the same amount of protection to the entire codestream. These techniques provide excellent results when the network conditions do not change, and are known (or can be estimated) apriori. However when packet losses occur less frequently than anticipated, the amount of redundancy in the codestream becomes unnecessary and reduces the compression performance. Furthermore, if the packet

Further author information: (Send correspondence to A.B.) A.B.: bilgin@ieee.org, Z.W. zywu@ece.arizona.edu, M.W.M.: marcellin@ece.arizona.edu losses occur more frequently than anticipated, the amount of protection provided by the ELP schemes become insufficient and catastrophic failures occur. Thus, ELP techniques do not provide a graceful degradation of image quality as network conditions vary. In [4] , the authors design a robust image transmission scheme based on unequal loss protection. The Set Partioning in Hierarchial Trees (SPIHT) algorithm is used as the compression scheme and the output codestream is protected by assigning unequal amounts of FEC. The amount of FEC that will be added to a partial section of the codestream is selected considering the importance of the section on image quality.

In this paper, we develop efficient techniques for transmission of JPEG2000 compressed codestreams over packet switched networks. The presented techniques utilize both smart partitioning of JPEG2000 codestream and FEC techniques to achieve robustness against packet losses. In the next section, we provide a brief review of the JPEG2000 compression algorithm and the anatomy of the codestreams it generates. In Section 3, we discuss the error resilience of JPEG2000 codestreams. Section 4 presents a method for packetizing JPEG2000 codestreams which provides error resilience against packet loss. Section 5 provides experimental results and comparisons of the proposed method with other methods in literature.

2. ANATOMY OF JPEG2000 CODESTREAMS

JPEG2000 is the latest international standard for image compression [5,6] . Besides providing state-of-the-art compression performance, it offers a number of functionalities that address the requirements of emerging imaging applications. The basic block diagram of a JPEG2000 encoder is illustrated in Figure 1. Here, the input image is first divided into non-overlapping rectangular tiles. If the image has multiple components, an optional component transform can be applied to decorrelate the components. The samples of each component that fall into a particular tile are referred to as a *tile-component* . Each tile-component is then transformed using a wavelet transform and the wavelet subbands are partitioned into several different geometric structures, as illustrated in Figure 2. These geometric structures are instrumental in enabling low memory implementations and providing spatial random access. As we will discuss later, they also contribute to the error resilience of the codestream.

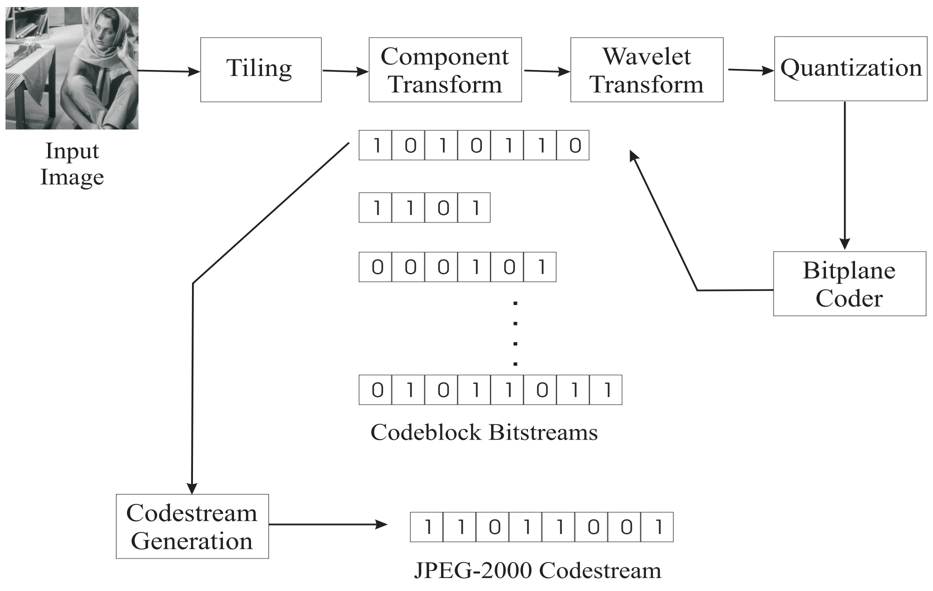


Figure 1: Block diagram of a JPEG2000 encoder.

The smallest geometric structure in JPEG2000 is the *codeblock.* Codeblocks are formed by partitioning the wavelet subbands. As illustrated in Figure 2, the codeblocks of particular resolutions are grouped together to form *precincts.* Once the wavelet subbands are quantized, each codeblock is compressed individually using a

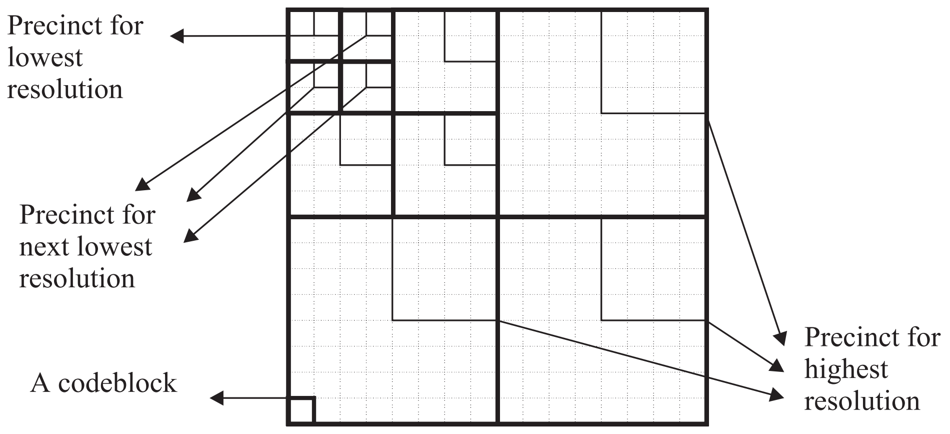

Precinct for lowest resolution
Precinct for next lowest resolution
A codeblock
Precinct for highest resolution

Figure 2: Partitioning of wavelet subbands for a tile-component.

bitplane coder. The bitplane coder makes three passes over each bitplane of a codeblock. These passes are referred to as *coding passes* or *subbitplanes*. The compressed data from each codeblock can be regarded as an embedded bitstream. The JPEG2000 encoder computes and stores the rate-distortion information associated with each coding pass of every codeblock. The codestream is then comprised of a different number of coding passes from each individual codeblock bitstream, based on any desired criteria.

For the purposes of forming the codestream, compressed data from each precinct are arranged to form *packets* *. Packets play an important role in the organization of data within JPEG2000 codestream. Each packet contains a header and a body. The packet header contains information about the contribution of each codeblock in the precinct into the packet, and the body contains coding passes of codeblocks. Packets that belong to a particular tile are grouped together to form a *tile-stream*, and tile-streams are grouped together to form the JPEG2000 codestream. Similar to packets, tile-streams are comprised of a header and a body. It is possible to break tile-streams into multiple *tile-parts*. In this case, the first tile-part contains a tile header and the remaining tile-parts contain tile-part headers. Tile-parts allow the progression concepts to be extended to the entire image. There is also a main header at the beginning of the codestream. The structure of a JPEG2000 codestream is illustrated in Figure 3. The EOC marker denotes the end of the codestream.

One of the key features of JPEG2000 is that it enables several modes of progression that provide increasing quality, resolution, color components, and/or spatial extent as more bytes are decoded from the codestream. Different types of progression are achieved by selecting the order in which packets appear within the codestream. For a comprehensive review of JPEG2000, including the codestream syntax, the reader is referred to [6] .

3. ERROR RESILIENCE OF JPEG2000 CODESTREAMS

Entropy coding in JPEG2000 is achieved using a context-based arithmetic bitplane coder. The operation of this coder is highly dependent on the state of the system, and it is crucial to maintain synchronization between the encoder and the decoder. A single bit error in the arithmetically coded segments of the bitstream can destroy this synchronization, and could result in erroneous decompression. To combat this problem, several error resilience tools are provided within JPEG2000.

The partitioning of the codestream into different segments is the first line of defense. In terms of error resilience, this partitioning aims to isolate errors made in one segment to that particular segment, and prevent error propagation across segment boundaries. This isolation occurs at several levels, since the codestream is organized in a hierarchical fashion. Each codestream starts with a main header. The main header contains

*It is important to note the distinction between the packets that are used in JPEG2000 codestreams and the packets that are used within the context of packet-switched networks. To avoid misunderstanding, we will refer to the packets used in networking as *network packets* for the remainder of this paper.

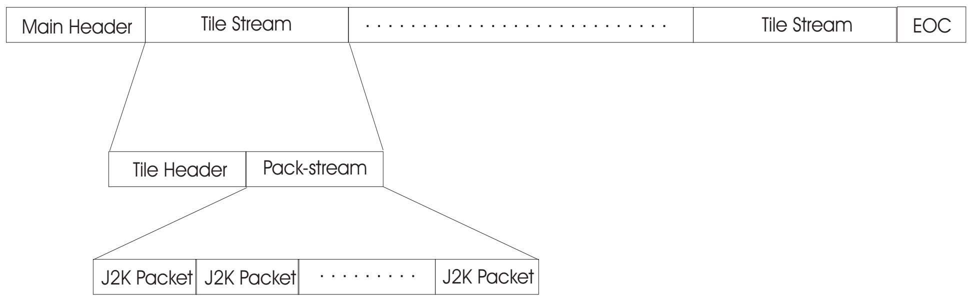


Figure 3: A simple JPEG2000 codestream.

critical information such as image and codeblock sizes, and is essential for correct decompression of the codestream. If sections of the main header are unavailable at the decoder, the decoder will not be able to decode the codestream. Similarly, if a tile-stream header is lost during tranmission, the decoder might be unable to determine several parameters that will be required for correct decompression of that tile. This might result in discarding the entire tile. A similar scenario would occur, if tile-parts were utilized and the first tile-part header was lost. However, if the header of any of the remaining tile-parts were lost, the decoder would be able to decompress the earlier tile-parts and the remaining tile-parts that belong to the current tile might need to be discarded. If a packet header is lost, all of the data in the current and future packets that correspond to the corresponding precinct will have to be discarded. Since codeblocks are coded independently, errors do not propogate between codeblocks.

To complement the hierarchical data partitioning, JPEG2000 provides several resynchronization tools. One of these tools is the SEGMARK switch. If this switch is enabled, a four symbol code is inserted at the end of the third coding pass of each bitplane. Since a bit error in any of the coding passes is likely to corrupt at least one of these symbols, the decoder can detect that an error has occured, and discard the erroneous coding passes. It is also possible to use the ERTERM option. This option creates a separate predictably terminated codeword segment for each coding pass. Thus, the decoder can detect that an error has occurred in a particular coding pass, and discard the current and future coding passes for that codeblock. It is also important to note that the JPEG2000 arithmetic coder uses byte-stuffing and is not allowed to produce certain values inside coding passes. These values are reserved for codestream markers. Unexpected detection of one of these values would also indicate that an error has occurred.

JPEG2000 provides a mechanism where the packet headers can be extracted from every packet and stored in tile-part headers or the main header. This is referred to as *packed packet headers*. Packed packet headers can provide significant advantages for error resilience if the main and tile-part headers can be transmitted in a lossless fashion. Since the packet headers contain the lengths in bytes of all coding passes in the packet, the decoder can utilize this information to isolate errors.

4. PACKETIZATION OF JPEG2000 CODESTREAMS

The problem of packetization is to find an efficient method that partitions a large JPEG2000 codestream into smaller segments corresponding to network packets. The packetization scheme should enable the decoder to resynchronize and continue decompression even when some of the network packets are lost during transmission. In this paper, we assume that the codestream is already generated and we are not allowed to modify it during the packetization process. Our ongoing work is directed to joint codestream formation/network packetization.

The simplest packetization approach is to divide the codestream into sections that correspond to the payload of the network packets, and fill the network packets sequentially. However, as discussed in the previous section, JPEG2000 codestreams consist of several different segments. Error resilience of each segment varies and should be taken into account during packetization. Header segments contain crucial information and are necessary for proper functioning of the decoder. These segments should be available at the decoder, even when some packets are lost during transmission. Thus, we propose a packetization scheme that partitions the JPEG2000 codestream into two sections. The data that are labeled for the first section are protected using FEC techniques and interleaved across network packets. This section contains all of the data that is critical for the operation of the decoder. This includes header segments of the codestream, as well as certain coding passes that are considered more important. For example, if a Region-Of-Interest (ROI) needs to be transmitted with a higher degree of confidence, the coding passes that correspond to the ROI can be included in this section. We refer to this section as the *protected* section. The data in the second section is not protected with FEC techniques, however certain alignment requirements are still imposed to improve error resilience. This section is refered to as the *unprotected* section. An illustration of this packetization strategy is provided in Figure 4.

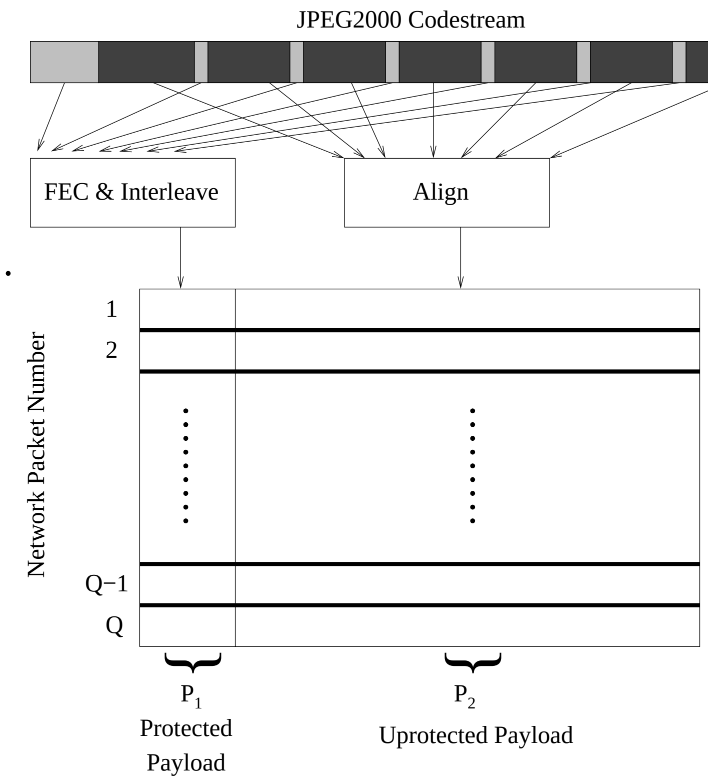


Figure 4: Illustration of the packetization method.

4.1. Packetization of the Protected Payload

The FEC in this work is achieved using Reed-Solomon (RS) codes. RS codes are a class of maximum distance separable codes that are effective in recovering from erasures [7] . An (N, k) RS code can recover from $N - k$ erasures, where N denotes the block length and k denotes the number of source symbols. In this work, we set $N = 255$ and adjust k to achieve the desired level of protection for a given channel.

Let L denote the length (in bytes) of the JPEG2000 codestream that will be packetized, and let L_1 and L_2 be the lengths of the protected and unprotected sections, respectively, such that $L = L_1 + L_2$. Let P denote the length of the network packet payload, and let P_1 and P_2 denote the lengths of the protected and unprotected payloads, respectively. Thus, $P = P_1 + P_2$. For FEC implementation, the protected section of the codestream is divided into blocks of length k. Each of these $\lceil \frac{L_1}{k} \rceil$ blocks are then encoded using an (N, k) RS code to yield length N blocks. These blocks are interleaved across the first P_1 bytes of the network packets. Given the probability of packet loss p_l of the channel, the block length k is selected such that the probability of having more than $N - k$ erasures in one or more of the $\lceil \frac{L_1}{k} \rceil$ protected blocks is less than a desired value ϵ. Since ϵ is an upper bound on the probability of uncorrectable error in the protected section of the codestream, it can be selected as large as can be tolerated in a particular application.

4.2. Packetization of the Unprotected Payload

The unprotected section of the codestream corresponds to individual codeblock codestreams that were not included in the protected section. Let M denote the number of such codeblocks. Let $B_i, i = 1, \cdots, M$ denote the length of the coding passes that belong to codeblock i. Since each of these codestreams are encoded independently, they can be decompressed independently as well. An erasure effecting one codeblock will not propogate to another during decompression. Thus, our goal in packetization of these codestreams is to impose alignment constraints to minimize network packet dependencies.

For $i = 1, \cdots, M$, the initial $\lfloor B_i/P_2 \rfloor P_2$ bytes of each codeblock bitstream can entirely fill $\lfloor B_i/P_2 \rfloor$ network packets. Note that none of these $\sum_{i=1}^{M} \lfloor B_i/P_2 \rfloor$ network packets will exhibit inter-packet dependencies due to their unprotected payloads. Furthermore, since the codeblock codestreams are embedded collections of coding passes, these initial $\lfloor B_i/P_2 \rfloor P_2$ bytes of the codeblock codestreams can be considered to be "more important" than the remaining $B_i^r = B_i - \lfloor B_i/P_2 \rfloor P_2$ bytes. Loss of the remaining B_i^r bytes of a codeblock codestream does not prevent the decompression of the coding passes that are completely contained in the initial $\lfloor B_i/P_2 \rfloor P_2$ bytes.

Let LIST denote the list of remaining bytes B_i^r, $i = 1, \cdots, M$. The B_i^r are then packetized using the following algorithm that aims to minimize the inter-packet dependencies without introducing significant packetization inefficiency:

```
Remove from LIST all $B_i^r$ that are zero;
while( LIST not empty ) {
    Initialize the available packet payload $R = P_2$;
    Find $j = \underset{i}{\mathrm{argmax}}(B_i^r)$
    Place $B_j^r$ into the next network packet;
    Remove $B_j^r$ from LIST;
    Set $R = R - B_j^r$;
    while( $R \neq 0$ ) {
        Select $j = \underset{i}{\mathrm{argmax}}(B_i^r)$ subject to the constraint that $R \geq B_j^r$;
        If no such $j$ exists {
            set $R = 0$;
        } else {
            Remove $B_j^r$ from LIST.
            Set $R = R - B_j^r$.
        }
    }
    Send network packet.
}
```

4.3. Extensions for Progressive Transmission

It is quite straight forward to extend the presented method to progressive transmission. If the JPEG2000 codestream contains several layers, the above method could be applied to each layer separately. Thus, the first group of network packets may contain the information for the first layer. The protected sections of these packets can contain the main header, together with other headers that are required for correct decompression of the initial layer. Subsequent groups of network packets will contain the additional layers, with the codestream for each layer partitioned into protected and unprotected sections.

5. EXPERIMENTAL RESULTS

In this section, we presents results of the experiments obtained using the presented method. Throughout these experiments, we have selected ϵ to be 10^{-5}. In other words, the probability of incorrect decompression of the protected payload is under 1 in 100, 000. We consider 53-byte ATM packets with payloads of 48 bytes. We assume that the decoder has access to p_l, ϵ, and the desired rate (the number of network packets). Experiments were repeated for 1000 random realizations of packet losses. The 512×512 grayscale Lena image was used as a single-tile image for the experiments. The codec used in the experiments was JPEG2000 Verification Model (VM) 9.0. The codestreams were single-layer, and packed packet headers were used to store the packet headers within the main header.

We first consider the effects of different codeblock sizes and number of wavelet decomposition levels. For these experiments, the main header, the tile-header, and the coding passes of the codeblocks that belong to the LL subband were protected and 683 network packets were used corresponding to a rate of 1.0005 bits/pixel. The results are presented in Figures 5, 6, and Table 1. Figure 5 presents results for different codeblock sizes with the number of wavelet decomposition levels equal to 3. Figure 6 presents results for different numbers of wavelet decomposition levels when the codeblock size equals 8×8. In both figures, the x-axis denotes PSNR, and the y-axis denotes the cumulative percentage of simulations that resulted in PSNR values lower than the corresponding x value. Both of these figures illustrate results for a mean packet loss rate of 10%. Table 1 presents average PSNR values for different network packet loss rates, codeblock sizes, and wavelet decomposition levels.

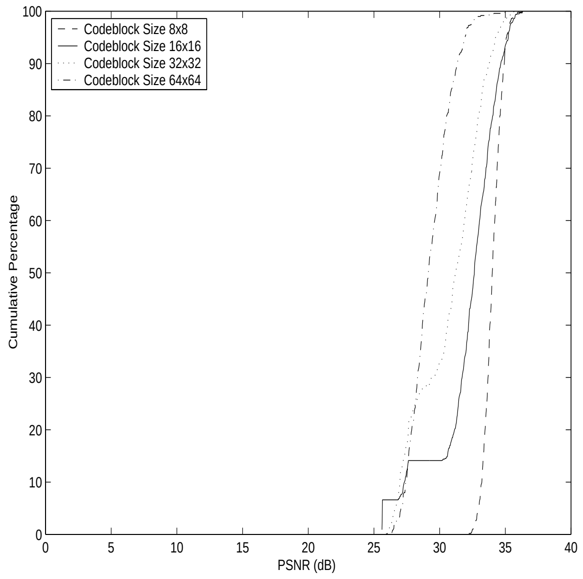


Figure 5: Comparison of different codeblock sizes. The number of wavelet decomposition levels is 3.

It should also be noted that although the packetization scheme is designed for a particular network packet loss rate, its operation under lower network packet loss rates is possible. The performance of the presented method under different design and operational network packet loss rates is illustrated in Table 2.

Next, we compare the performance of different packetization strategies to illustrate the effectiveness of the presented method. Method I is the simplest packetization strategy. It partitions the codestream into network packets sequentially without considering the underlying structure of the JPEG2000 codestream. Method II protects the headers and the LL subband coding passes as described in section 4.1. However, the remainder of the codestream is segmented and placed into network packets sequentially. Method III uses the proposed algorithm, however, the LL subband coding passes are transmitted in the unprotected section of the network packets. The last method is the proposed algorithm with all the headers and the LL subband coding passes protected. Figures 7 and 8 present comparisons of these different methods for 683 (1.0005 bits/pixel) and 143 (0.2095 bits/pixel) network packets, respectively. A mean network packet loss rate of 10% was used to generate the figures. We also present the average PSNR results in Tables 3 and 4 For comparison, the tables include results from PZW [2] and optimal packetization (OP) [3] schemes. It should also be noted that unlike [2] and [3] , no

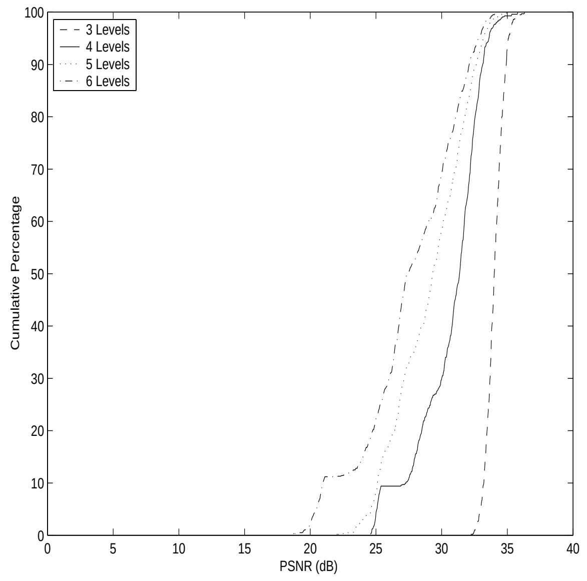


Figure 6: Comparison of different wavelet decomposition levels. The codeblock size is 8×8.

	Network Packet Loss Rate					
Block Size / Decomp. Levels	1%	2%	5%	10%	20%	30%
8×8 / 3	37.54	36.79	35.41	34.04	29.45	30.36
8×8 / 4	37.19	36.32	34.18	30.63	29.07	25.89
8×8 / 5	36.77	35.06	31.74	29.10	24.98	24.56
8×8 / 6	36.41	34.46	31.39	27.60	23.29	22.52
16×16 / 3	38.12	37.20	35.24	32.10	29.99	28.70
16×16 / 4	37.99	36.63	33.22	30.36	27.94	25.97
16×16 / 5	37.21	35.17	31.68	29.42	25.38	23.66
32×32 / 3	38.06	36.73	34.05	30.77	29.00	28.30
32×32 / 4	37.37	35.36	31.66	29.04	26.47	25.08
64×64 / 3	36.91	35.07	31.23	29.30	27.89	26.78

Table 1. Average PSNR results of the presented method for different network packet loss rates, codeblock sizes and wavelet decomposition levels (683 network packets, 1.0005 bits/pixel).

error concealment is used in our simulations. Error concealment techniques can be utilized to provide additional subjective and objective improvement over the presented results.

Figure 9 presents four representative images generated using the presented method for network packet loss rates of 2%, 10%, 20%, and 50%. The image was encoded into 143 network packets, corresponding to 0.2095 bits/pixel. Notice that the image quality degrades gracefully and the image remains recognizable even at the high loss rate of 50%.

	Design Network Packet Loss Rate						
Operational Network Packet Loss Rate	0%	1%	2%	5%	10%	20%	30%
0%	31.85	31.85	31.56	31.56	31.32	30.92	30.11
1%	-	31.57	31.35	31.38	31.15	30.77	29.94
2%	-	-	31.04	31.04	30.95	30.62	29.79
5%	-	-	-	30.53	30.40	30.22	29.35
10%	-	-	-	-	29.70	29.63	28.69
20%	-	-	-	-	-	28.69	27.75
30%	-	-	-	-	-	-	26.88

Table 2. Average PSNR results of the presented method for different design and operational network packet loss rates. Codeblock size is 16×16 and the number of wavelet decomposition levels is 3. (143 network packets, 0.2095 bits/pixel).

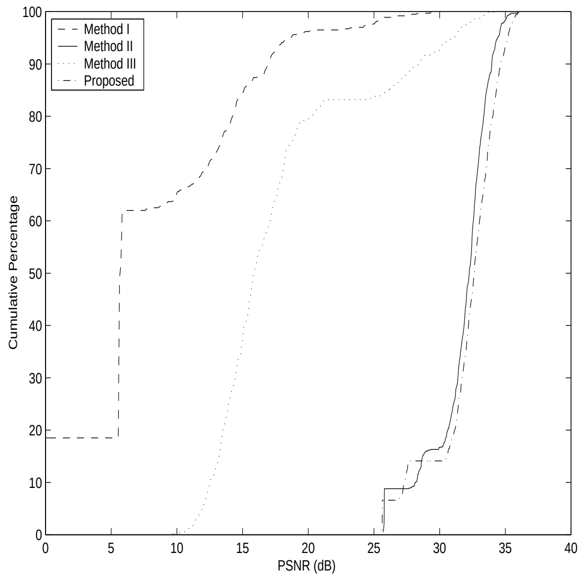


Figure 7: Comparison of different packetization strategies (683 network packets, 1.0005 bits/pixel).

	Packet Loss Rate						
Packetization Method	0%	1%	2%	5%	10%	20%	30%
Method I	39.40	29.21	21.67	13.61	8.26	4.20	3.01
Method II	39.40	38.08	36.91	34.43	31.65	29.68	28.07
Method III	39.40	34.46	30.45	24.11	17.82	13.30	10.53
Proposed	39.40	38.12	37.20	35.24	32.10	29.99	28.70
OP [3]	39.71	36.28	34.13	30.62	27.64	24.44	-

Table 3: Average PSNR results for different packetization strategies (683 network packets, 1.0005 bits/pixel).

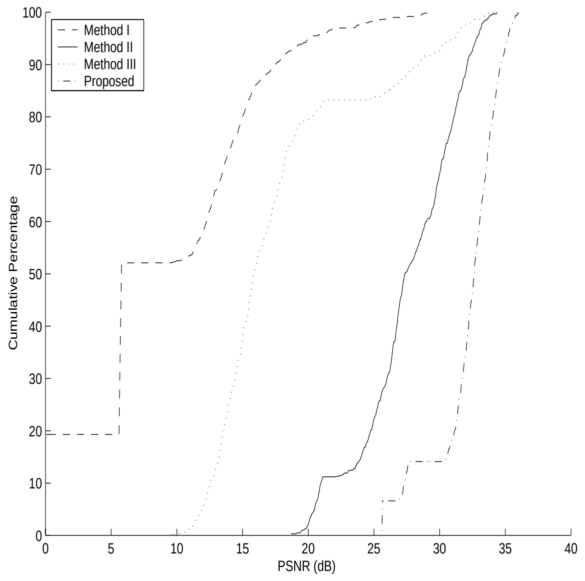


Figure 8: Comparison of different packetization strategies (143 network packets, 0.2095 bits/pixel).

	Packet Loss Rate						
Packetization Method	0%	1%	2%	5%	10%	20%	30%
Method I	31.85	26.57	22.28	15.48	9.25	4.79	3.02
Method II	31.85	31.59	31.05	30.31	29.54	28.27	26.80
Method III	31.85	29.59	27.27	22.70	17.89	13.17	10.75
Proposed	31.85	31.57	31.04	30.53	29.70	28.69	26.88
PZW [2]	32.19	31.33	-	-	26.29	24.63	-
OP [3]	32.20	31.50	30.86	29.36	27.53	25.08	-

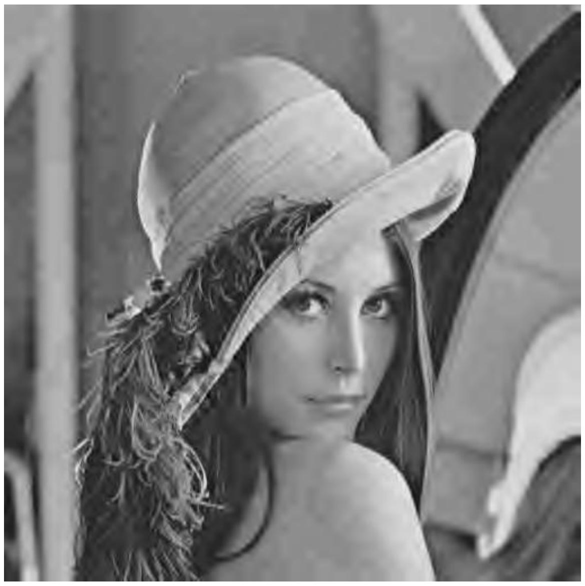
Table 4: Average PSNR results for different packetization strategies (143 network packets, 0.2095 bits/pixel).

6. SUMMARY

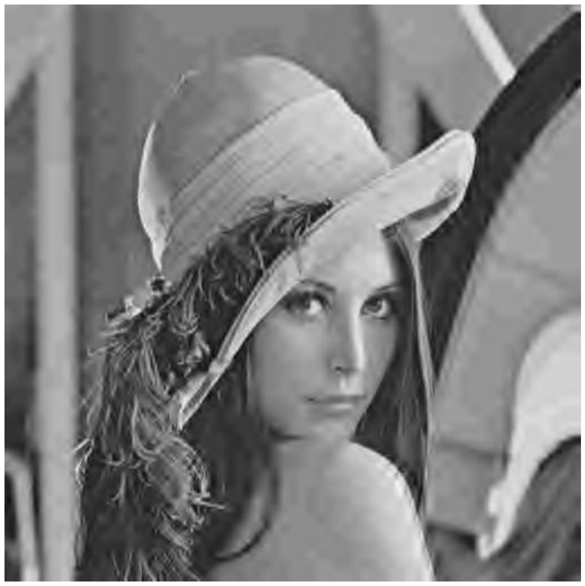
JPEG2000 includes tools to build error resilience into the compressed codestream. In this work, we have presented a technique that improves the error resilience of JPEG2000 against packet losses, when a codestream is transmitted over packet erasure channels. The presented technique takes the properties of different segments of the codestream into account during packetization. The performance of the proposed method compares favorably with existing methods.

REFERENCES

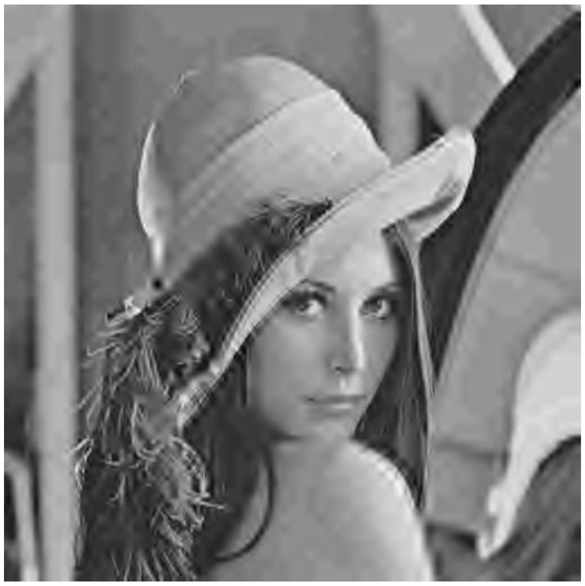
1. C. D. Creusere, "A new method for robust image compression based on embedded zerotree wavelet algorithm," *IEEE Transactions on Image Processing* **6**, pp. 1436–1442, Oct. 1997.
2. J. K. Rogers and P. C. Cosman, "Wavelet zerotree image compression with packetization," *IEEE Signal Processing Letters* **5**, pp. 105–107, May 1998.
3. X. Wu, S. Cheng, and Z. Xiong, "On packetization of embedded multimedia bitstreams," *IEEE Transactions on Multimedia* **3**, pp. 132–140, Mar. 2001.
4. A. E. Mohr, E. A. Riskin, and R. E. Ladner, "Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction," *IEEE Journal on Selected Areas in Comm.* **18**, pp. 819–828, June 2000.
5. "JPEG 2000 Part I Final Draft International Standard," *ISO/IEC JTC 1/SC 29/ WG1, Doc. No. N1855* , Aug. 2000.
6. D. S. Taubman and M. W. Marcellin, *JPEG2000: Image Compression Fundamentals, Practice and Standards*, Kluwer Academic Publishers, Massachusetts, 2002.
7. S. Lin and D. J. Costello Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, New Jersey, 1983.



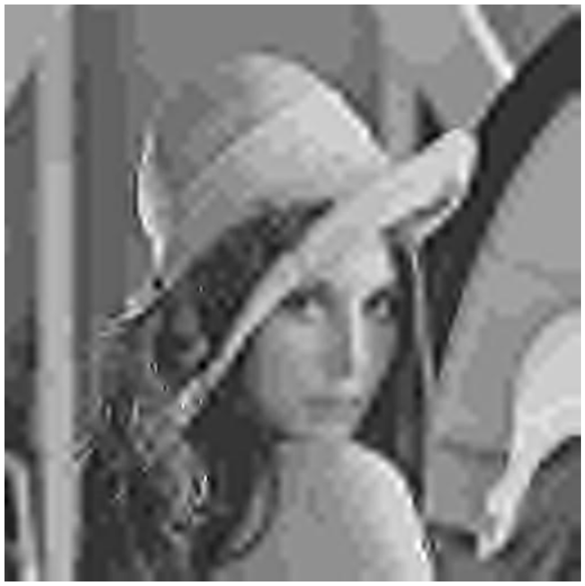
(a) 2% Packet Loss (PSNR = 31.13 dB).



(b) 10% Packet Loss (PSNR = 29.90 dB).



(c) 20% Packet Loss (PSNR = 28.75 dB).



(d) 50% Packet Loss (PSNR = 25.33 dB).

Figure 9: Image quality at 0.2095 bits/pixel for different network packet loss rates.