Unequal Error Protection Using Fountain Codes With Applications to Video Communication

Article *in* IEEE Transactions on Multimedia · February 2011

Impact Factor: 2.3 · DOI: 10.1109/TMM.2010.2093511 · Source: DBLP

3 authors, including:

Shakeel Ahmad
Southampton Solent University
179 PUBLICATIONS **3,186** CITATIONS
SEE PROFILE

Marwan Al-Akaidi
De Montfort University
43 PUBLICATIONS **205** CITATIONS
SEE PROFILE

Unequal Error Protection using Fountain Codes with Applications to Video Communication

Shakeel Ahmad, Raouf Hamzaoui, Marwan Al-Akaidi

Abstract— **Application-layer forward error correction (FEC) is used in many multimedia communication systems to address the problem of packet loss in lossy packet networks. One powerful form of application-layer FEC is unequal error protection which protects the information symbols according to their importance. We propose a method for unequal error protection with a Fountain code. When the information symbols were partitioned into two protection classes (most important and least important), our method required a smaller transmission bit budget to achieve low bit error rates compared to the two state of the art techniques. We also compared our method to the two state of the art techniques for video unicast and multicast over a lossy network. Simulations for the scalable video coding (SVC) extension of the H.264/AVC standard showed that our method required a smaller transmission bit budget to achieve high quality video.**

EDICS: 5-HIDE. **Index terms**: Fountain codes, unequal error protection, video transmission.

I. Introduction

Many multimedia communication systems use application layer forward error correction (FEC) to deal with the problem of packet loss in networks that do not guarantee quality of service. One important class of FEC codes are Fountain codes [1], [2], [3]. Fountain codes are FEC erasure codes with two main advantages over conventional erasure codes such as Reed-Solomon codes. First, Fountain codes have a much lower encoding and decoding complexity. Second, whereas conventional erasure codes have a fixed code rate that must be chosen before the encoding begins, Fountain codes are rateless in the sense that the encoder can generate on the fly as many encoded symbols as needed. This is an advantage when the channel conditions are unknown because the use of a fixed channel code rate would lead to bandwidth waste if the erasure rate is overestimated or to poor performance if it is underestimated.

Luby Transform (LT) codes [2] were the first class of practical Fountain codes. Another class of Fountain codes are Raptor codes [3], which are obtained by concatenating a fixed-rate channel code with an LT code. Raptor codes have been adopted as enhanced application layer FEC by Multimedia Broadcast/Multicast System (MBMS) of the 3rd Generation Partnership Project (3GPP), IP datacast (IPDC) of Digital Video Broadcasting - Handheld (DVB-H), as well as Digital Video Broadcasting Project's (DVB) global IPTV standard.

With the growing interest in Fountain codes, the question of how to achieve unequal error protection (UEP) with these codes has been addressed by several researchers [4], [5], [6], [7]. In contrast to equal error protection (EEP) where the same level of FEC is applied to all information symbols, UEP assigns different levels of protection to different information symbols. Typically, the information symbols are protected according to their importance. This usually allows a better overall system performance than EEP [8]. UEP has been successfully used for the protection of scalable image and video coders such as JPEG2000 [9], 3D SPIHT [10], and the scalable video coding (SVC) [11] extension of the H.264/MPEG-4 AVC video compression standard. In [7], UEP with standard Raptor codes is applied to stereoscopic video streaming. The authors define layers of stereoscopic video and use rate-distortion optimization to jointly determine optimal video encoder bit rates and Raptor code redundancy for the defined layers. In [4], [5], [6], rateless codes with intrinsic UEP characteristics are designed.

This paper has two main contributions. The first one is a method to decrease the bit error rate (BER) of LT codes. The idea is to duplicate the set of information symbols and extend the original degree distribution to the new set of information symbols. This effectively results in a stronger degree distribution, which is shown by experiments to decrease the BER at the cost of a controllable increase in encoding and decoding complexity. The second contribution of the paper is an extension of this idea to UEP with LT codes. In particular, we apply our UEP scheme to the problem of video multicast with heterogeneous receivers. We provide experimental results for SVC and show that our method required a smaller transmission bit budget to achieve higher peak signal to noise ratio (PSNR) than the state of the art techniques of [4] and [5]. The paper synthesizes and extends previous results published in [12], [13]. It provides a more general and more rigorous description of the proposed methods and presents more experimental results.

The paper is organized as follows. Section II contains background material about LT codes. Section III describes the UEP techniques of [4] and [5]. Section IV presents our idea for improving the BER performance of LT codes and explains how it can be exploited to provide UEP. Section V gives simulation results.

II. Background

In this section, we explain the encoding and decoding with LT codes. More details can be found in [2].

This work was supported in part by the DFG Research Training Group GK-1042.
The authors are with the Faculty of Technology, De Montfort University, The Gateway, Leicester LE1 9BH, UK. Email: sahmad@dmu.ac.uk, rhamzaoui@dmu.ac.uk, mma@dmu.ac.uk.

A. Encoding

The LT encoder takes a set of k information symbols (bits or bytes, for example) and generates a potentially infinite sequence of encoded symbols of the same alphabet. Each encoded symbol is computed independently of the other encoded symbols. More precisely, given k information symbols $i_0, \ldots, i_{k-1}$ and a suitable probability distribution $\Omega(x)$ on $\{1, \ldots, k\}$, a sequence of encoded symbols e_m, $m \geq 0$ is generated as follows. For each $m \geq 0$

1) Select randomly a degree $d_m \in \{1, \ldots, k\}$ according to the distribution $\Omega(x)$.
2) Select uniformly at random d_m distinct information symbols and set e_m equal to their bitwise modulo 2 sum.

The relationship between the information symbols and encoded symbols can be described by a graph (see Fig. 1 for an example).

If the number of information symbols is k, the degree of an encoded symbol is given by the degree distribution $\Omega(x) = \sum_{i=1}^{k} \Omega_i x^i$ on $\{1, \ldots, k\}$, where Ω_i is the probability that degree i is chosen. For example, suppose that

$$\Omega_i = \begin{cases} \frac{1}{k} & \text{if } i = 1; \\ \frac{1}{i(i-1)} & \text{otherwise.} \end{cases}$$

Then $\Omega(x)$ is called the ideal soliton distribution [2]. A more practical distribution is the robust soliton distribution [2] $\Delta(x) = \sum_{i=1}^{k} \Delta_i x^i$ given by $\Delta_i = \frac{\Omega_i + \Lambda_i}{d}$, where $\Omega(x)$ is an ideal soliton distribution, $\Lambda(x) = \sum_{i=1}^{k} \Lambda_i x^i$ is a degree distribution on $\{1, \ldots, k\}$ given by

$$\Lambda_i = \begin{cases} \frac{s}{ki} & \text{if } i = 1, \ldots, \frac{k}{s} - 1; \\ \frac{s}{k} \ln(\frac{s}{\delta}) & \text{if } i = \frac{k}{s}; \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

$d = \sum_{i=1}^{k} \Omega_i + \Lambda_i$, and $s = c \ln \frac{k}{\delta} \sqrt{k}$. Here c and δ are parameters (see [2] for an interpretation of these parameters).

Another useful degree distribution is the fixed degree distribution [3] given by

$$\begin{aligned} \Omega(x) = {} & 0.007969x + 0.493570x^2 + 0.166220x^3 + 0.072646x^4 \\ & + 0.082558x^5 + 0.056058x^8 + 0.037229x^9 \\ & + 0.055590x^{19} + 0.025023x^{64} + 0.003135x^{66}. \end{aligned} \tag{2}$$

B. Decoding

When an encoded symbol is transmitted over an erasure channel, it is either received correctly or lost. The LT decoder tries to recover the original information symbols from the received encoded symbols. We assume that for each received encoded symbol, the decoder knows the indices of the information symbols it is connected to. This is possible, for example, by using a pseudo-random generator with the same seed as the one used by the encoder.

The decoding process is as follows:

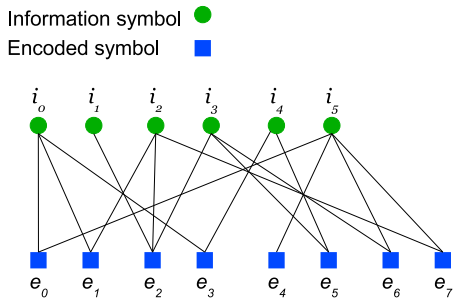

Fig. 1. Graph of an LT code. Eight encoded symbols are generated from $k = 6$ information symbols. The degree of an encoded symbol is the number of information symbols that were used to generate it. For example, the degree of e_0 is equal to two.

1) Find an encoded symbol e_m that is connected to only one information symbol i_j. If this is not possible, stop the decoding.
   a) Set $i_j = e_m$.
   b) Set $e_x = e_x \oplus i_j$ for all indices $x \neq m$ such that e_x is connected to i_j. Here $\oplus$ denotes the bitwise modulo 2 sum.
   c) Remove all edges connected to i_j.
2) Go to Step 1.

The probability of successfully decoding all information symbols increases with increasing number of received encoded symbols.

III. Previous work

In this section, we describe the two previous UEP techniques with LT codes.

A. Rahnavard, Vellambi, and Fekri's method [4]

Rahnavard, Vellambi, and Fekri [4] were the first to propose a method to provide UEP with LT codes. For simplicity, we describe their method when two levels of protection are used. Consider a source block having k information symbols. Partition this block into two blocks S_1 and S_2 of length $|S_1| = \alpha k$ and $|S_2| = (1-\alpha)k$, respectively, where $0 < \alpha < 1$. The block S_1 is called the block of most important bits (MIB) while the block S_2 is called the block of least important bits (LIB). Define probabilities p_1 and p_2 ($p_1 + p_2 = 1$) to select S_1 and S_2, respectively. Given a suitable probability distribution $\Omega(x)$ on $\{1, \ldots, k\}$, a sequence of encoded symbols e_m, $m \geq 0$ is generated as follows. For each m

1) Select randomly a degree $d_m \in \{1, \ldots, k\}$ according to the distribution $\Omega(x)$.
2) Select d_m distinct information symbols successively. To select a symbol, first select one of the two blocks S_1 or S_2 (S_1 with probability p_1 and S_2 with probability p_2). Then choose randomly a symbol from the selected block.
3) Set e_m equal to the bitwise modulo 2 sum of the d_m selected information symbols.

Fig. 2 illustrates the process.

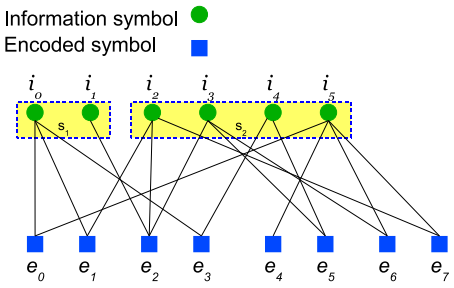

Fig. 2. UEP scheme proposed in [4]. Two levels of protection are used. The MIB block contains two information symbols while the LIB block contains four information symbols.

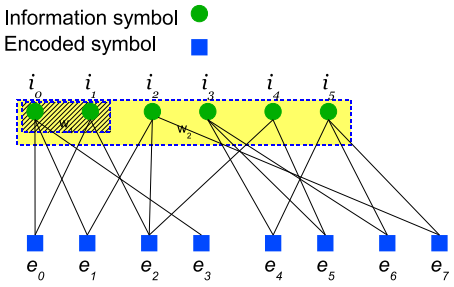

Fig. 3. UEP scheme proposed in [5]. Two windows W_1 and W_2 are used. The encoded symbols e_0 and e_3 are generated from W_1 while the remaining encoded symbols are generated from W_2.

To ensure that the MIB symbols have lower BER than the LIB symbols, the probability of selecting an MIB symbol should be larger than the probability of selecting an LIB symbol [4], that is, $p_1 \frac{1}{|S_1|} > p_2 \frac{1}{|S_2|}$. To achieve this, one can set $p_1 = \frac{k_M |S_1|}{k}$ and $p_2 = \frac{k_L |S_2|}{k}$ for $0 < k_L < 1$ and $k_M = (1 - (1-\alpha)k_L)/\alpha$. Here the parameter k_M gives the relative importance of the MIB symbols.

B. Method of Sejdinovic et al. [5]

A source block having k information symbols is partitioned into L blocks $S_1, S_2, \ldots, S_L$ such that the first $|S_1|$ information symbols of the source block are the most important bits, the next $|S_2|$ information symbols are the next most important bits and so on. Then L windows $W_1, W_2, \ldots, W_L$ are defined such that W_i is the concatenation of the blocks $S_1, \ldots, S_i$. Thus the size of the ith window is $|W_i| = \sum_{j=1}^{i} |S_j|$. For every window W_i, an LT code with a degree distribution on the set $\{1, \ldots, |W_i|\}$ is defined. To generate an encoded symbol, a window W_i is selected according to a probability distribution $\Gamma(x) = \sum_{i=1}^{L} \Gamma_i x^i$. Here Γ_i is the probability that window W_i is chosen. Then the LT code defined on W_i is applied. UEP is achieved by choosing appropriate values for $\Gamma_1, \ldots, \Gamma_L$. Fig. 3 illustrates the encoding process for $k = 6$ and $L = 2$.

IV. Proposed method

We first explain how to improve the BER performance of an LT code. We then present a method to build an LT code with UEP property.

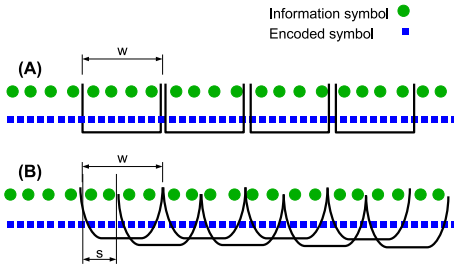

Fig. 4. Sliding window technique of [14]. (A): without window overlap. (B) with window overlap.

A. Virtual increase of source block size

Given an LT code, we propose to extend the set of information symbols by duplicating it. This idea has similarities with the sliding window (SW) technique introduced in [14] for LT codes and extended in [15] for Raptor codes (see Fig. 4). The SW technique defines a window and applies LT encoding to the information symbols within the window. The window has a size of w symbols and is shifted by s symbols until all information symbols are covered. Thus, the number of windows is $N_w = \frac{k-w}{s} + 1$ and each symbol is covered about w/s times (except for the few first and last ones). For example, when $s = w$, the windows do not overlap and every information symbol is covered once. When $s < w$, some information symbols are covered by more than one window. As the size of the overlap increases, the virtual size of the source block increases, resulting in higher decoding efficiency [14].

As in the method of [14], we virtually increase the size of the source block. However, we do not use windows. Instead, we duplicate all information symbols and extend the original degree distribution to the new set of information symbols. Simulations in Section V show that our approach gives in general better BER performance. In the following, we describe our approach in detail.

Consider a source block $S = i_0 * \cdots * i_{k-1}$ consisting of k information symbols $i_0, \ldots, i_{k-1}$. Let $\Omega(x)$ be the degree distribution of an LT code on $\{1, \ldots, k\}$. We expand the source block S by repeatedly appending the same k information symbols at the end of the block. The new (virtual) source block can be written $\underbrace{S * S * \cdots * S}_{EF}$ where the *expanding factor* EF denotes the number of times the original source block occurs in the new source block. This new source block has a length of $EF \times k$ and its information symbols have indices ranging from 0 to $EF \times k - 1$ (Fig. 5). Next, we extend the original degree distribution $\Omega(x)$ from $\{1, \ldots, k\}$ to $\{1, \ldots, EF \times k\}$ and use a standard LT encoder with this new degree distribution to generate the encoded symbols. For the robust soliton distribution, this is done by replacing k by $EF \times k$ in (1). An encoding graph using the original k information symbols $i_0, \ldots, i_{k-1}$ is obtained by replacing the index $j \in \{0, \ldots, EF \times k - 1\}$ of a selected information symbol by $j \mod k$.

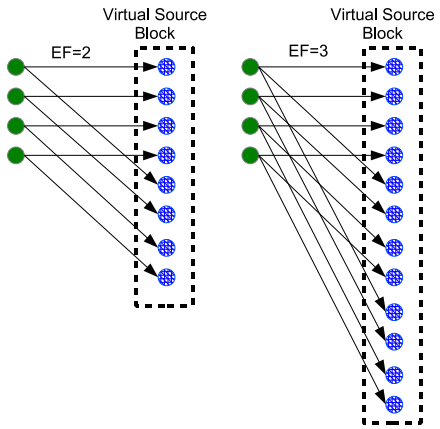

Fig. 5. Virtual increase of the source block size for $k = 4$, (left) $EF = 2$ and (right) $EF = 3$.

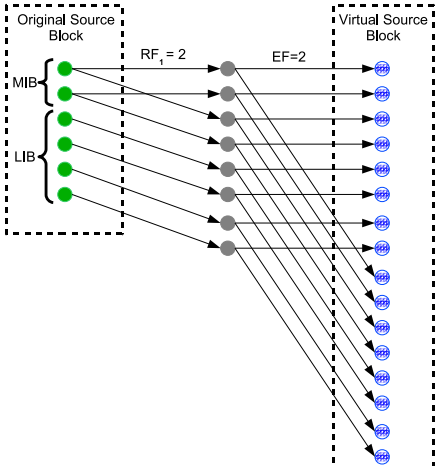

Fig. 6. Building a virtual source block with the proposed UEP with $k = 6$, $EF = 2$, and $RF = 2$.

B. Unequal error protection

The concept of virtually increasing the size of the source block by duplicating information symbols has a natural application to UEP. Suppose that a source block $S = i_0 * \cdots * i_{k-1}$ is partitioned into L adjacent blocks $S_1, S_2, \ldots, S_L$ such that the first block S_1 consists of the most important bits, the next block S_2 consists of the next most important bits and so on. We can assign different levels of protection to these blocks by duplicating them according to a sequence of repeat factors $RF_i,\ i = 1, \ldots, L$. That is, we build a (virtual) source block

$$\underbrace{S_1 * S_1 * \cdots * S_1}_{RF_1} * \underbrace{S_2 * S_2 * \cdots * S_2}_{RF_2} * \ldots * \underbrace{S_L * S_L * \cdots * S_L}_{RF_L}$$

whose information symbols have indices ranging from 0 to $\sum_{i=1}^{L} RF_i|S_i| - 1$.

We next extend the degree distribution of the LT code from $\{1, \ldots, k\}$ to $\{1, \ldots, \sum_{i=1}^{L} RF_i|S_i|\}$. To generate an encoded symbol, we find its degree d using the new degree distribution and then select d information symbols from the virtual source block. An encoding graph using the original k information symbols $i_0, \ldots, i_{k-1}$ is obtained by replacing the index $j \in \{0, \ldots, \sum_{i=1}^{L} RF_i|S_i| - 1\}$ of a selected information symbol by an index l as follows:

For example, suppose that $k = 6$, $L = 2$, $|S_1| = 2$, and $|S_2| = 4$. If we duplicate block S_1 as in the first step of Fig. 6, the virtual size of the source block becomes 8, corresponding to repeat factors $RF_1 = 2$ and $RF_2 = 1$. We next extend the degree distribution of the LT code from $k = 6$ to $k = 8$. To generate an encoded symbol, we find its degree d using the new degree distribution and then select d information symbols from the 8 virtual symbols. If the index of a selected information symbol is, for example, 5 we map it to $5 \mod 4 + 2 = 3$.

This UEP technique can be combined with the method proposed in Section IV-A by duplicating the virtual source block with an expanding factor EF. For example, for $RF_1 = 2$, $RF_2 = 1$, and $EF = 2$, the original source block consisting of two MIB symbols and four LIB symbols is transformed into a virtual block of size $EF(RF_1 \times 2 + RF_2 \times 4) = 16$ (Fig. 6).

V. Experimental results

We present four sets of experiments. The first one shows the effect of increasing the expanding factor on the encoding and decoding time of the method proposed in Section IV-A. The second one compares the BER of this method to that of the SW approach of [14]. The third one compares the BER performance of our UEP scheme (Section IV-B) to that of [4] and [5]. The fourth one compares the PSNR performance of our UEP scheme to that of [4] and [5] in video transmission experiments. The BER was calculated as the average (over the number of simulations) of $(k - d)/k$, where k is the number of original information symbols and d is the number of (correctly) decoded symbols. The PSNR was calculated as the average (over the number of simulations) of the mean (over all frames) of the PSNR of the luminance (Y) component. For a given reconstructed frame, the PSNR was calculated as $10 \log_{10} \frac{255^2}{MSE}$ where MSE is the mean square error between the original frame and this frame. The BER and the PSNR were computed for various values of the transmission overhead $t = (n - k)/k$, where k is the number of original information symbols and n is the number of transmitted symbols.

A. Time complexity of block duplication

Fig. 7 and 8 show respectively the average encoding time and the average decoding time as a function of the expanding factor for $k = 1000$ information symbols (bytes) and various values of the transmission overhead. The channel was lossless. The average time was computed over 1000 simulations. Throughout the paper, the decoding time was calculated as the average decoding time for all runs (both successful and unsuccessful decodings). The time was measured on a PC running an Intel(R) Core(TM)2 CPU 1.66GHz with 1GB RAM. The results show that even for large values of the expanding factor, both the encoding and the decoding are fast. Note also that the time complexity does not increase linearly with the expanding factor.

$$l = \begin{cases} j \mod |S_1| & \text{if } 0 \le j \le RF_1|S_1| - 1; \\ [(j - RF_1|S_1|) \mod |S_2|] + |S_1| & \text{if } RF_1|S_1| \le j \le RF_1|S_1| + RF_2|S_2| - 1; \\ \dots & \\ [(j - \sum_{i=1}^{L-1} RF_i|S_i|) \mod |S_L|] + |S_{L-1}| + \ldots + |S_1| & \text{if } \sum_{i=1}^{L-1} RF_i|S_i| \le j \le \sum_{i=1}^{L} RF_i|S_i| - 1; \end{cases}$$

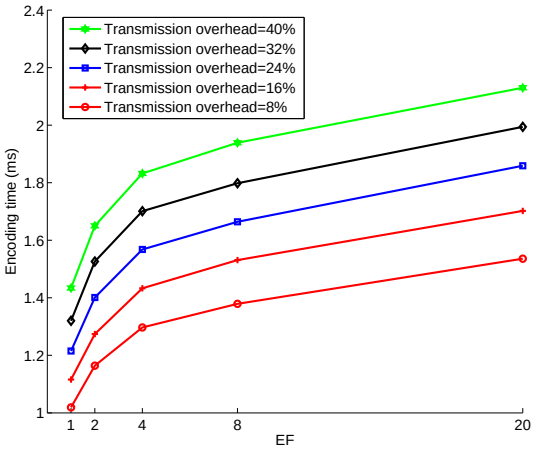

Fig. 7. Average encoding time vs. expanding factor for $k = 1000$ and various transmission overheads.

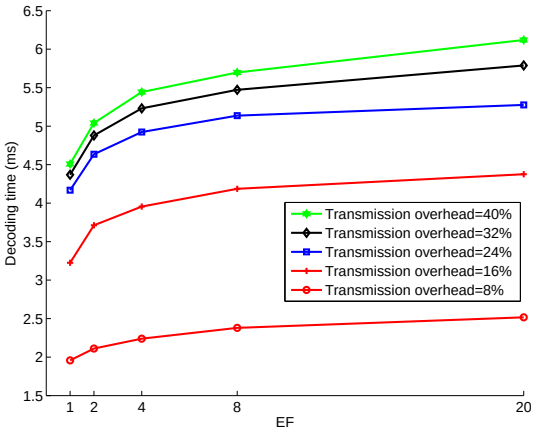

Fig. 8. Average decoding time vs. expanding factor for $k = 1000$ and various transmission overheads.

B. Comparison with the sliding window approach of [14]

Fig. 9 compares the BER of our method to that of the SW technique [14] for an input source block of size $k = 20,000$ information symbols (bytes). A robust soliton distribution with parameters $c = 0.1$ and $\delta = 0.5$ was used as the underlying degree distribution of the LT code. The results are for 100 simulations. The channel was lossless.

The performance of the two methods improved by increasing the virtual number of information symbols. However, there was a limit beyond which no improvement was observed (50 % overlap for SW and $EF = 8$ for our approach). The BER performance of our approach was better than that of the SW approach when the transmission overhead was larger than about 0.05. When the transmission overhead was smaller than this value, SW gave a lower BER. However, in this range, the

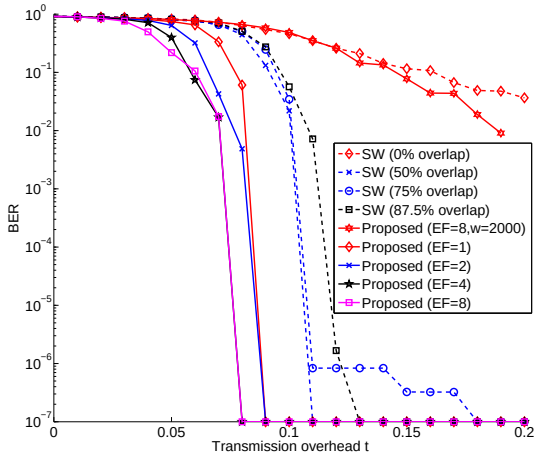

Fig. 9. BER vs. transmission overhead for our approach and the sliding window method (SW) of [14]. The number of information symbols is $k = 20,000$. For the SW method, the window size is $w = 2000$. The curve "Proposed ($EF = 8$, $w = 2000$)" shows results for our method when it was applied on windows of size 2000 with $EF = 8$.

BER is too high for the method to be useful. Note that the BER improvement obtained with our method is penalized by a higher coding complexity. The complexity can be reduced by, for example, applying our method on windows of size $2,000$. However, as shown in Fig. 9, the BER performance would then significantly decrease.

Fig. 10 and 11 compare the encoding and decoding times of our method to those of the SW approach [14] for various values of EF and the window overlap. Since SW applies LT coding on smaller blocks, its encoding time was lower. The decoding time of SW was also lower when the transmission overhead was small. Note that the decoding time of our scheme almost stops increasing when the transmission overhead reaches about 8 %. This is because the decoder generally can recover all information symbols at this overhead and stops without processing further encoded symbols. For SW with window overlap 50 % and 75 %, successful decoding is achieved at about 11 % overhead. However, the decoding time keeps on increasing with increasing overhead because the SW decoder needs to process the windows in sequential order and cannot therefore recover all information symbols before almost all encoded symbols are processed (except for the encoded symbols of the last window).

C. Comparison of BER performance for UEP schemes

Fig. 12 and 13 show the BER performance of the three UEP schemes for $k = 1000$ and $k = 5000$ symbols (bytes), respectively. The results were obtained from 1000 simulations. The channel was lossless. The information symbols were

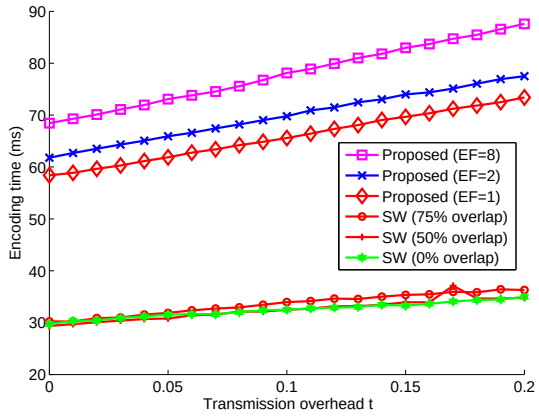

Fig. 10. Encoding time for our approach and the sliding window method (SW) of [14]. The number of information symbols is $k = 20,000$. For the SW method, the window size is $w = 2,000$.

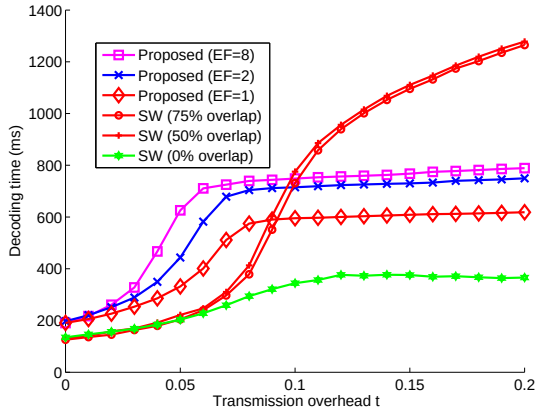

Fig. 11. Decoding time for our approach and the sliding window method (SW) of [14]. The number of information symbols is $k = 20,000$. For the SW method, the window size is $w = 2,000$.

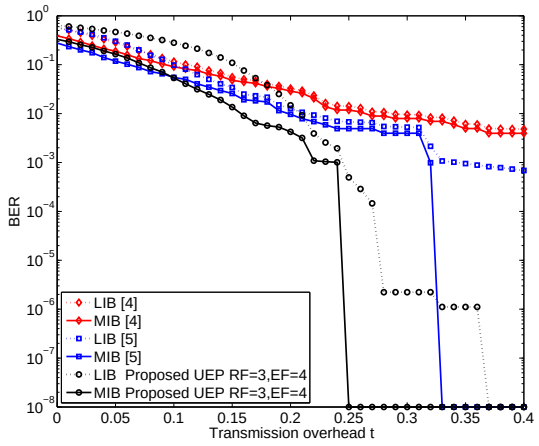

Fig. 12. BER vs. transmission overhead. There are $k = 1,000$ information symbols, 100 of which belong to the MIB block. Our scheme is used with the robust soliton distribution. The scheme of [4] is used with the fixed degree distribution (2). The scheme of [5] is used with the robust soliton distribution for the MIB block and the fixed degree distribution for the LIB block.

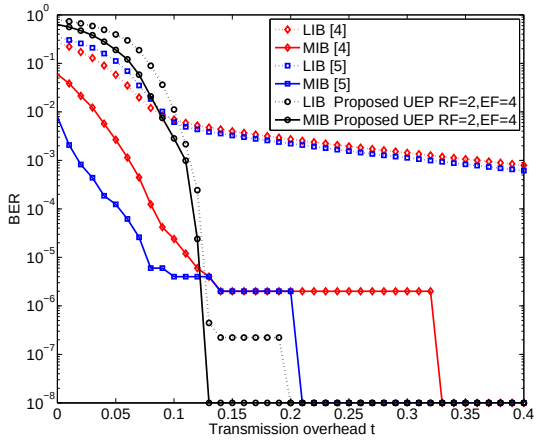

Fig. 13. BER vs. transmission overhead. There are $k = 5,000$ information symbols, 500 of which belong to the MIB block. Our scheme is used with the robust soliton distribution. The scheme of [4] is used with the fixed degree distribution (2). The scheme of [5] is used with the robust soliton distribution for the MIB block and the fixed degree distribution for the LIB block.

partitioned into two blocks. The MIB block consisted of the first 10 % information symbols. For the UEP scheme of [4], we used the fixed degree distribution (2) as in [4]. For the UEP scheme of [5], we followed [5] and used the robust soliton distribution with $c = 0.03$ and $\delta = 0.5$ for the MIB block and the fixed degree distribution for the LIB block. The parameter k_M was set to 2 as in [4], and the parameter Γ_1 was set to 0.084 as in [5]. For our scheme, we used the robust soliton distribution with $c = 0.1$ and $\delta = 0.5$, set $RF_2 = 1$, and run simulations to determine the best values for RF_1 and EF. For simplicity of notation RF_1 will be denoted by RF. Compared to the UEP methods of [4] and [5], our scheme provided better performance for both the MIB and the LIB blocks when a low BER was targeted. For example, when $k = 1000$, the BER with our scheme was below 10^{-3} at transmission overhead $t = 0.25$, while the best previous scheme did not provide such a BER before $t = 0.33$. The BER performance gain of our scheme was larger for smaller block lengths. This can be seen in Fig. 13 ($k = 5000$) where at low transmission overhead, the UEP methods of [4] and [5] yielded a lower MIB BER than our scheme.

Fig. 14 and 15 show the average encoding and decoding time of the three UEP schemes for $k = 1000$. Compared to our UEP scheme, the schemes of [4] and [5] have an additional decision making step in the encoding. However, the average degree of an encoded symbol is larger in our scheme. Consequently, both the encoding and decoding time of our scheme were higher than those of [4] and [5]. The additional decision making step is taken each time an encoded symbol is computed in [5] and each time an edge is selected as in [4]. This explains why the encoding complexity of scheme of [4] was higher than that of [5].

D. Comparison of PSNR results for transmission of H.264 SVC

We compare our UEP technique to the UEP techniques of [4] and [5] for video unicast as well as video multicast to heterogeneous receiver classes.

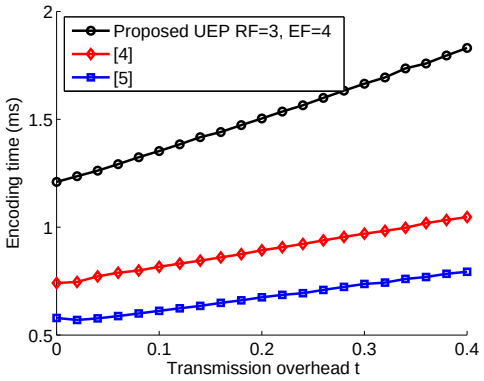

Fig. 14. Average encoding time vs. transmission overhead for $k = 1000$.

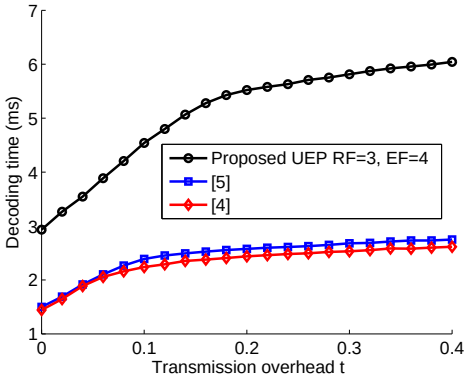

Fig. 15. Average decoding time vs. transmission overhead for $k = 1000$.

As in [6] (this paper applies the method of [5] to SVC transmission), we used the Stefan video sequence which has a spatial resolution of 352×288 and a temporal resolution of 30 fps. The first group of pictures (GOP) of the sequence was encoded using the SVC reference software (JVSM) into one base layer (BL) and 14 enhancement layers (ELs). SVC can provide quality, resolution, and temporal scalability. In the experiments, we used it in the quality (SNR) scalability mode to encode a GOP of 16 frames. The resulting layer sizes, bit rates, and Y-PSNR are summarized in Table I. A source block corresponding to this GOP was transmitted 250 times.

Decoded layers	Size	Bitrate	PSNR
BL	400	292.37	25.79
BL + 1 EL	700	510.65	27.25
BL + 2 EL	875	636.56	28.14
BL + 3 EL	1155	839.82	29.00
BL + 4 EL	1550	1127.10	29.51
BL + All ELs	3800	2764.55	40.28

TABLE I

SVC ENCODING OF THE FIRST GOP OF THE STEFAN VIDEO SEQUENCE (352X288,30 FPS) INTO ONE BASE LAYER (BL) AND 14 ENHANCEMENT LAYERS (EL). THE TABLE SHOWS THE NUMBER OF SYMBOLS, THE BITRATE IN KBPS, AND THE Y-PSNR IN DB.

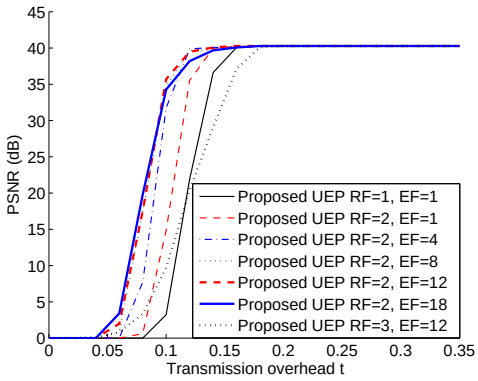

Fig. 16. PSNR as a function of the transmission overhead for the transmission of the Stefan sequence with the proposed UEP scheme. The performance of the scheme is shown for different settings of EF and RF.

Since the GOP size is not constant, the number of information symbols in the following source blocks may be different. To address this problem, the sender has to inform the receiver about the number of information symbols in each GOP. This can be done using either out-of-band signaling (using a control channel) or in-band signaling (where the information is sent in the header of the data packets).

As in [6], each symbol was equal to 50 bytes, giving $k = 3800$ symbols per source block. In each source block, the base layer (containing 400 symbols) was chosen as the MIB block, whereas the remaining symbols built the LIB block. The source block was transmitted 250 times. At the receiver side, we assumed that a layer can be used to enhance the video quality only if it was decoded fully, and all the layers before this layer were also decoded fully. In this way, the number of consecutively decoded information symbols, starting from the first information symbol, determined the number of decoded layers.

Fig. 16 shows the PSNR performance of our UEP scheme for various settings of $RF = RF_1$ and EF (RF_2 was fixed to 1). The video was transmitted to one receiver. The channel was lossless. The robust soliton distribution was used with parameters $c = 0.1$ and $\delta = 0.5$ The results show that increasing RF from 2 to 3 increases the likelihood of decoding the BL successfully, but decreases the likelihood of successfully decoding the ELs, leading to a decrease of the overall PSNR. Increasing EF beyond 12 increases the average degree and leads to duplicate selection of the information symbols, which decreases the performance.

Fig. 17 and 18 show the PSNR performance of the UEP scheme of [4] and the UEP scheme of [5], respectively, as a function of the transmission overhead. As in the previous experiment, the video was transmitted to one receiver and the channel was lossless. For the UEP scheme of [4], we tested the fixed degree distribution (2) as in [4] but also the robust soliton distribution with $c = 0.1$ and $\delta = 0.5$. For the UEP scheme of [5], we used the robust soliton distribution with parameters $c = 0.03$ and $\delta = 0.5$ for the MIB block and the fixed degree distribution (2) for the LIB block as in [6]. Moreover, we did an experiment where for the LIB block we used the robust

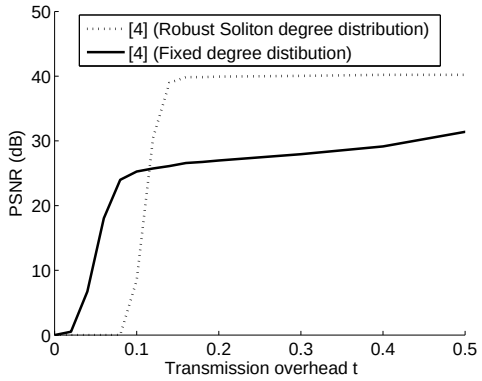

Fig. 17. PSNR as a function of the transmission overhead for the transmission of the Stefan sequence with the UEP scheme of [4].

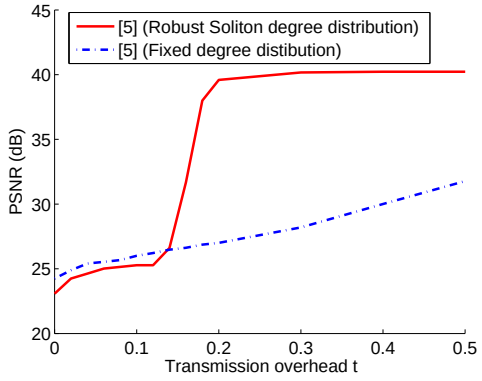

Fig. 18. PSNR as a function of the transmission overhead for the transmission of the Stefan sequence with the UEP scheme of [5].

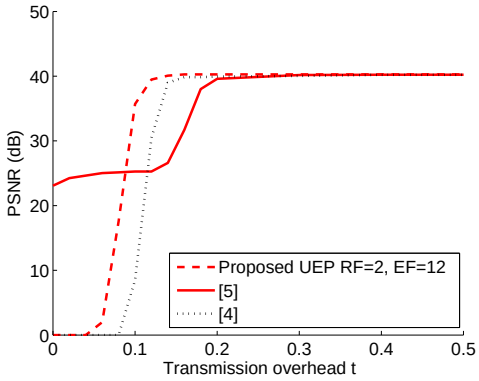

Fig. 19. PSNR as a function of the transmission overhead for the proposed UEP scheme, the UEP scheme of [4], and the UEP scheme of [5] with their best settings.

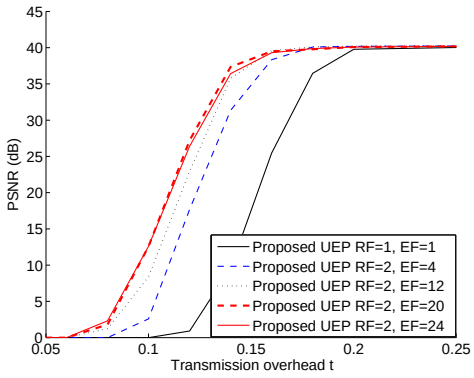

Fig. 20. Average PSNR of $m = 2$ receiver classes as a function of the transmission overhead for our UEP scheme for different settings of EF and ER.

soliton distribution with parameters $c = 0.1$ and $\delta = 0.5$ instead of the fixed degree distribution. On average, the robust soliton distribution provided better results than the fixed degree distribution for both UEP schemes.

Fig. 19 compares the PSNR perceived by a receiver class with a zero loss rate as a function of the transmission overhead for our UEP scheme, the UEP scheme of [4], and the UEP scheme of [5]. The PSNR curve of our UEP scheme had a similar shape to the one of [4]. However, for the same PSNR our scheme required lower transmission overhead. Compared to the UEP scheme of [5], our scheme had a better performance at average and high overhead but worse performance at low overhead. The scheme of [5] has a good performance at low overhead because it ensures that a certain number of encoded symbols are produced exclusively from the MIB block. However, this strategy penalizes the scheme when a higher video quality is required.

Fig. 20, 21, 22 show the average PSNR performance for a multicast transmission to $m = 2$ receiver classes with symbol loss rates 0.02 and 0.04. For our scheme, the best results were achieved with $RF = 2$ and $EF = 20$. Compared to Fig. 16, more transmission overhead was needed to achieve the same PSNR because of symbol loss in the channel. For the schemes of [4] and [5], the fixed degree distribution gave higher PSNR at low overhead but much lower PSNR at high overhead compared to the robust soliton distribution.

Fig. 23 compares the average PSNR performance of the three schemes with their best settings for the multicast scenario. Our UEP scheme always outperformed the UEP scheme of [4]. It also outperformed the UEP scheme of [5] over the range of transmission overheads that are required to provide high video quality. In particular, our UEP scheme achieved an average video quality of about 37 dB with 50 % less transmission overhead (or 10 % less bandwidth).

Fig. 24 and 25 show the PSNR results for each receiver class. Compared to our UEP scheme, the scheme of [5] achieved an acceptable video quality (about 25 dB) for the receiver with the worst channel conditions at a lower transmission overhead (Fig. 24). However, the PSNR for this receiver did not exceed this value up to a transmission overhead of $t = 0.18$. In contrast, the PSNR of our scheme rose sharply beyond 25 dB at $t = 0.12$.

VI. Conclusion

We proposed a method that improves the BER performance of LT codes by strengthening the degree distribution in a simple way. We used the same approach to develop a UEP

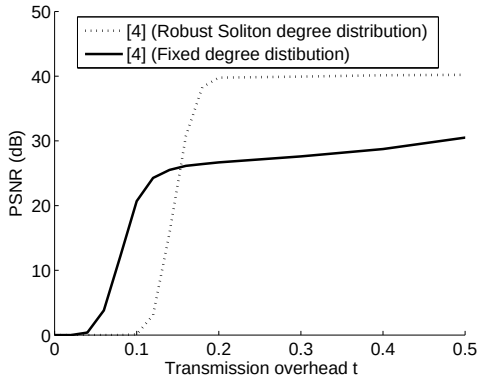

Fig. 21. Average PSNR of $m = 2$ receiver classes as a function of the transmission overhead for the UEP scheme of [4].

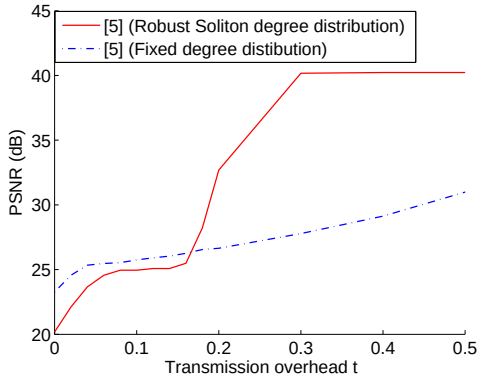

Fig. 22. Average PSNR of $m = 2$ receiver classes as a function of the transmission overhead for the UEP scheme of [5].

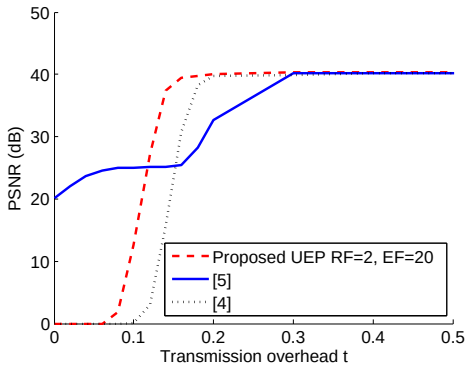

Fig. 23. Average PSNR of $m = 2$ receiver classes as a function of the transmission overhead for the proposed UEP scheme, the UEP scheme of [4], and the UEP scheme of [5] with their best settings.

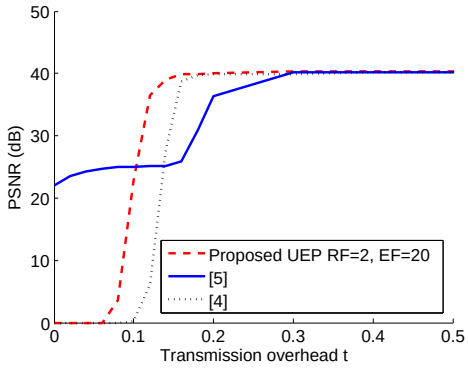

Fig. 24. PSNR of receiver class with symbol loss rate 0.02 as a function of the transmission overhead.

scheme for LT codes. Compared to the UEP schemes of [4] and [5], our scheme had higher coding complexity but required lower transmission overhead to achieve low BER. Since the encoding and decoding time of our scheme are low enough for real-time applications (the times reported in the experimental results can be further reduced by optimizing the software implementation or relying on a hardware-based one), the benefits of our scheme outweigh its drawbacks.

While our scheme is conceptually similar to the scheme of [4], the two schemes differ in one important aspect. Unlike the scheme of [4], our scheme changes the degree distribution of the original LT code, which results in overall better BER performance.

Experimental results for SVC showed that our scheme can provide up to 13 dB improvement in PSNR over the UEP schemes of [4] and [5] for unicast video transmission. For multicast video transmission to a set of receivers with heterogeneous channel conditions, results showed that our scheme had a better average PSNR performance than the best previous scheme when the transmission overhead was large and a worse performance when it was low. This makes our scheme more appropriate in applications where a high video quality is desired.

We conclude by emphasizing the importance of the degree distribution in UEP code design. For example, our paper showed that the results in [6] can be significantly improved by replacing the fixed degree distribution used for the LIB block with a robust soliton distribution.

VII. Acknowledgment

The authors thank the anonymous reviewers for their constructive comments and suggestions that helped improve the paper.

References

[1] J.W. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE J. Selected Areas Commun.*, vol. 20, pp. 1528–1540, Oct. 2002.

[2] M. Luby, "LT codes," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci.*, Vancouver, BC, Canada, Nov. 2002, pp. 271–280.

[3] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[4] N. Rahnavard, B. N. Vellambi, and F. Fekri, "Rateless codes with unequal error protection property," *IEEE Trans. Inf. Theory*, vol. 53, no. 53, pp. 1521–1532, Apr. 2007.

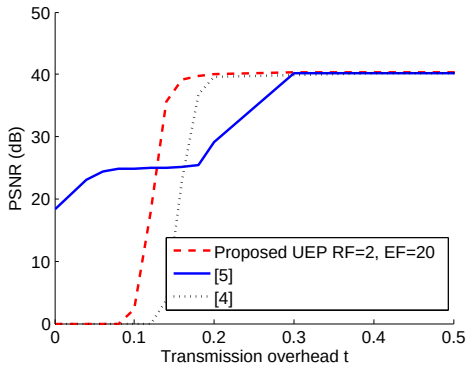

Fig. 25. PSNR of receiver class with symbol loss rate 0.04 as a function of the transmission overhead.

[5] D. Sejdinovic, D. Vukobratovic, A. Doufexi, V. Senk, and R.J. Piechocki, "Expanding window fountain codes for unequal error protection," in *Proc. 41st Asilomar Conf. Sig. Syst. Comp.*, Pacific Grove, CA, Nov. 2007, pp. 1020–1024.

[6] D. Vukobratovic, V. Stanković, D. Sejdinovic, L. Stanković, and Z. Xiong, "Scalable video multicast using expanding window fountain codes," *IEEE Trans. Multimedia*, vol. 11, pp. 1094–1104, Oct. 2009.

[7] A. S. Tan, A. Aksay, G. Bozdagi Akar, and E. Arikan, "Rate-distortion optimization for stereoscopic video streaming with unequal error protection," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, Article ID 632545, 15 pages, 2009. doi:10.1155/2009/632545.

[8] R. Hamzaoui, V. Stanković, and Z. Xiong, "Optimized error protection of scalable image bitstreams," *IEEE Signal Proc. Mag.*, vol. 22, pp. 91–107, Nov. 2005.

[9] D. S. Taubman and M. Marcellin, JPEG2000: Image Compression Fundamentals, Standards and Practice, Norwell, MA: Kluwer, 2001.

[10] B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 1365–1374, Dec. 2000.

[11] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, pp. 1103–1120, Sep. 2007.

[12] S. Ahmad, R. Hamzaoui, and M. Al-Akaidi, "Unequal error protection using LT codes and block duplication," in *Proc. MESM 08, Middle Eastern Multiconference on Simulation and Modelling*, Amman, Jordan, Aug. 2008, pp. 104–108.

[13] S. Ahmad, R. Hamzaoui, and M. Al-Akaidi, "Video multicast using unequal error protection with Luby Transform codes," in *Proc. MESM 09, Middle Eastern Multiconference on Simulation and Modelling*, Beyrouth, Sep. 2009, pp. 72–76.

[14] M.C.O. Bogino, P. Cataldi, M. Grangetto, E. Magli, and G. Olmo,"Sliding-window digital fountain codes for streaming of multimedia contents," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, New Orleans, LA, May 2007, pp. 3467–3470.

[15] P. Cataldi, M. Grangetto, T. Tillo, E. Magli, and G. Olmo, "Sliding-window Raptor codes for efficient scalable wireless video broadcasting with unequal loss protection," IEEE Transactions on Image Processing, vol. 19, pp. 1491–1503, Jun. 2010.

BIOGRAPHIES

Shakeel Ahmad received the B.S. degree in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2000, the M.Sc. degree in information and communication systems from Hamburg University of Technology, Hamburg, Germany, in 2005, and the Dr. Ing. degree from the University of Konstanz, Konstanz, Germany, in 2008. He is currently working as a Research Fellow with the Faculty of Technology, De Montfort University, Leicester, U.K. His research interests include multimedia communication, error-resilient video streaming, peer-to-peer networks, and channel coding.

Raouf Hamzaoui received the Master's degree in mathematics from the Faculty of Sciences of Tunis, Tunis, Tunisia, in 1986, the M.Sc. degree in mathematics from the University of Montreal, Montreal, QC, Canada, in 1993, the Dr. rer. nat. degree from the Faculty of Applied Sciences, University of Freiburg, Freiburg, Germany, in 1997, and the Habilitation degree in computer science from the University of Konstanz, Konstanz, Germany, in 2004. He was an Assistant Professor with the Department of Computer Science, University of Leipzig, Leipzig, Germany, and with the Department of Computer and Information Science of the University of Konstanz, Germany. He is currently a Professor of Media Technology in the Department of Engineering and Technology at De Montfort University, Leicester, U.K. His research interests include image and video compression, multimedia communications, error control systems, and algorithms.

Marwan M. Al-Akaidi (M96-SM03) received the B.S. degree in electrical and electronic engineering in 1982, the M.Sc. degree in digital communications systems in 1984, and the Ph.D degree in optical communications in 1988. He joined the Department of Electronic Engineering, De Montfort University, Leicester, UK, in 1991. In November 2000, he became a Professor of Communication and Signal Processing. His main research interest is in the field of digital signal processing and digital communications. This includes speech coding, processing, recognition, and wireless and mobile communication. Professor Al-Akaidi was appointed as the Chairman of the IEEE United Kingdom and Republic of Ireland (UKRI) Signal Processing Society in September 1999, and as the IEEE UKRI Conferences Chair in March 2000. In December 2000, he joined the board of the IEEE Industrial Relations. He received the award of the IEEE UKRI in recognition of outstanding leadership as a Chapter Chair in 2001 and 2002.