A Testbed of Erasure Coding on Video Streaming System over Lossy Networks

Xing-Jun Zhang*† and Xiao-Hong Peng*
*Electronic Engineering, School of Engineering and Applied Science, Aston University, UK
E-mail: x.zhang, x-h.peng@aston.ac.uk
†Department of Computer Science, Xi'an Jiaotong University, China
E-mail: xjzhang@xjtu.edu.cn

***Abstract*— As one of the most challenging aspects of streaming video over lossy networks, the technology for controlling packet losses has attracted more and more attention. Erasure coding is one of the ideal choices to deal with this problem. In most cases, the researchers need an effective method or tool to validate the erasure codes used for dealing with different packet loss patterns. Although some previous work has been done on employing erasure codes in video streaming system, few actual buildups and experiments which involve implementation of erasure codes against real packet loss in streaming systems have been reported. In this paper, we focus on constructing a testbed that integrates loss pattern generation and erasure coding implementation into video streaming services over lossy networks. With this approach, we are able to assess the capability of erasure coding in packet loss control and compare the performances of the video streaming systems with and without erasure coding. As an example, we have implemented the Reed-Solomon (7, 5) code for protecting MPEG streaming data under random packet losses. Experiment results show that the replay quality can be improved significantly by using erasure coding in video streaming systems, and that the testbed can suggest appropriate erasure code parameters for different loss environments.**

I. Introduction

Video has been an important media for communications and entertainment for many decades. The growth and popularity of the Internet motivated video communication over best-effort packet networks. Nowadays, with the development of broadband wireless networks, delivering video over wireless networks is also a popular application. Video over lossy networks is complicated by a number of factors including unknown and time-varying bandwidth, end-to-end delay, jitter, and losses, as well as many additional issues such as how to fairly share the network resources amongst many flows and how to efficiently perform one-to-many communication for popular content [1]. As one of the most challenging aspects of video streaming, the technology for controlling packet losses has attracted increasing attention [2].

Numerous techniques have been suggested for error and loss control over the lossy networks. To recover lost packets, two well-known techniques exist: automatic repeat request (ARQ), which retransmits the lost packets, and forward error correction coding (FEC) in packet level, which transmits redundant packets together with the data protected and requires no retransmission.

In ARQ, if errors have been detected in the data stream transmitted at the receiver, it requests a retransmission of that data from the transmitter. Simple ARQ protocols for multicast suffer from a condition known as feedback implosion, because many receivers attempt to send acknowledgement for a single packet. A number of multicast ARQ protocols have been suggested to avoid or reduce the implosion effect [3]-[8]. While ARQ techniques are effective in providing reliability, they can result in significant and unpredictable delay, making ARQ unsuitable for applications that have stringent real-time constraints, for instance, video conferencing and moving pictures through wide area networks (WAN). Since most real-time application can tolerate some degree of data losses, but can not tolerate long-time delay associated with retransmissions, FEC is often cited as a technique for real-time multicast [9]. Error correcting codes are traditionally applied to correct erroneous bits or symbols [10], but have been proposed, as erasure codes, to recover lost packets due to channel fading, interference and network congestion in video/audio broadcasting, multicasting and real-time Internet communications [11]-[4]. These erasure codes, such as Reed-Solomon (RS) codes, have strong inherent erasure-correction capability [2] [15] and the capability of carrying out error correction and erasure correction simultaneously. A dropped packet can be regarded as an erasure, so we call the FEC code used in the video streaming system as an erasure code and FEC coding as erasure coding correspondingly.

The end-to-end applications and related network performances mainly depend on the transport layer protocols, such as the transmission control protocol (TCP) and the user datagram protocol (UDP). TCP, which is equivalent to the ARQ strategy, could suffer long delays in the scenarios such as poor channel conditions (particularly in wireless networks), multicast and long-distance transmission. UDP, in contrast to TCP, offers speedy data delivery as it has no re-transmission, but can not guarantee for reliable services as it does not recover the lost or corrupted packets. Therefore, it is a big challenge for conventional IP-based networks to meet the increasing demand for supporting the multimedia distribution that requires both real-time and high-quality performances. This leads naturally to the consideration of employing erasure coding techniques, combined with ARQ, to tackle these problems [16].

Although some previous work has been done in employing erasure codes in video streaming system, few tangible buildups and experiments involving actual packet loss and implementation of erasure codes on streaming system have been reported. In most cases, effective methods or tools are vital for research in this field in order to validate the codes against different packet loss patterns. Our project: Optimized Data Storage Caching with High Availability Data Delivery within a Distributed Storage Network, supported by EPSRC (Engineering and Physical Sciences Research Council) and Xyratex, aims to tackle the problem as to how to support real-time and high-quality multimedia distribution applications over IP-based lossy networks. We propose a unique combination of multidiscipline technologies including erasure coding, graph coloring, data aggregation and multi-source data processing, in order to ensure reliable and speedy delivery of high-quality data to mobile computing users. For these purposes, we need a testbed to verify the performances of the algorithms and protocols developed. In this paper, we focus on constructing the testbed to examine the erasure codes in the context of enhancing the received video quality over lossy networks.

The paper is structured as follows. Section II gives a brief introduction to erasure codes. Section III describes FEC-based video communication system. Section IV describes the testbed. In Section V, we give a typical example and analyze the experimental results, system performance characterized by the improved packet loss rate and implementation complexity. Finally, a conclusion is given in Section VI.

II. Erasure Codes

A. An Introduction to Erasure Codes

Erasure codes are a form of FEC used for communication between senders and receivers through a lossy medium. When decoding the encoded data using erasure codes, the receiver is assumed to know the exact location of the lost packets, while this information is not needed in a general FEC technique. Erasure codes are typically used for sending packets through the Internet since the receiver can detect the location of the lost packets by noting the skipped packet sequence number. In a typical erasure code, sender encodes redundant packets before sending both the original and redundant packets to the receiver. Receiver can reconstruct the original packets upon receiving a fraction of the total packets. Standard erasure codes such as the RS (N, K) erasure codes; take K original packets and produces $(N - K)$ redundant packets, resulting in a total of N packets. If K or more packets are received, then all the original packets can be completely reconstructed. Hence, a larger N/K ratio leads to a higher level of protection for data [17]. In this paper, we use RS codes as a typical example.

B. Reed Solomon Codes

RS codes [10] as maximum distance separable (MDS) codes have been suggested to be applied for packet loss protection in many papers, such as [9] [14] [18] [19]. RS code is a media-independent FEC technique that can be applied at the packet level. As shown in Figure 1, an application level video frame is supposed as being transmitted in K packets where K varies with frame type, encoding method, and media content. RS code adds $(N-K)$ redundant packets to the K original packets and sends the N packets over the network. Although some packets may be lost, e.g. packet 3 in Figure 1, the frame still can be completely reconstructed if any K or more packets are successfully received.

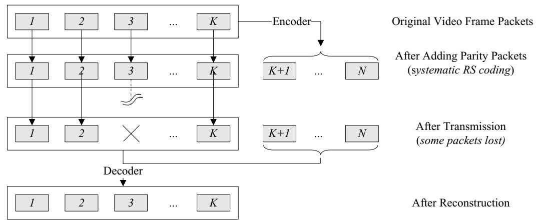


Fig. 1. Reed-Solomon Codes

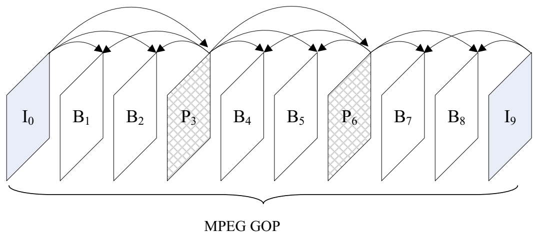


Fig. 2. Prediction dependencies between frames

III. Erasure Coding on Video Streaming

A video sequence consists of a number of video frames or images. There are three basic common types of coded frames: (1) intra-coded frames, or I-frames, where the frames are coded independently of all other frames, (2) predictively coded, or P-frames, where the frame is coded based on a previously coded frame, and (3) bi-directionally predicted frames, or Bframes, where the frame is coded using both previous and future coded frames. Figure 2 illustrates the different coded frames and prediction dependencies in an MPEG Group of Pictures (GOP), as an example. The selection of prediction dependencies between frames can have a significant effect on video streaming performance, e.g. in terms of compression efficiency and error resilience [1].

Most of the current video coding schemes, such as MPEG-1/2/4 and H.261/263/264, are compressed by applying the same basic principles [1]. The temporal redundancy is exploited by applying motion compensated prediction, the spatial redundancy is exploited by applying the Discrete Cosine Transform (DCT), and the color space redundancy is exploited by a color space conversion. The resulting DCT coefficients are quantized, and the nonzero quantized DCT coefficients are runlength and Huffman coded to produce the compressed bitstream. After compression, strong spatiotemporal dependency in video data is created. When these compressed data are transmitted over lossy networks, packet losses can severely affect streaming video quality. For example, as little as 3% MPEG packet loss can cause 30% of the frames to be undecodable [19].

A video communication system is designed with error control to combat the effect of losses. There are four rough classes of approaches for error control: retransmissions, FEC, error concealment, and error-resilient video coding. The last two classes of approaches are source coding approaches for error control. A video streaming system is typically designed using a number of these different approaches. In addition, joint design of the source coding and channel coding is very important. FEC provides a number of advantages [1]. For example, compared to retransmissions, FEC does not require a back-channel and may provide lower delay since it does not depend on the round-trip-time of retransmits. Most importantly, FEC-based approaches are designed to overcome a predetermined amount of losses and they are quite effective if they are appropriately matched to the channel. If the losses are less than a threshold, then the transmitted data can be perfectly recovered from the received data with losses. However, if the losses are greater than the threshold, then only a portion or none of the data can be recovered, depending on the type of FEC used. Although it is difficult to find those thresholds, constructing a testbed or framework that can investigate the packet loss effects on FEC-based video communication system is very useful to verify if the new algorithms and schemes work properly.

IV. Description of the Testbed

A. System Overview

Figure 3 shows the system architecture. The whole system is mainly composed of five parts: (1) streaming server, (2) erasure code encoder, (3) network emulation platform: NTuner, (4) erasure code decoder and (5) video player. The footage is streamed by Live555 streaming server. The encoding part includes packet receiver and sender, in/out buffer, code algorithm library and encoder. The erasure code encoder gets the video packets and applies certain encoding algorithm to the video data. Then, the real video data is sent to NTuner, a network emulation platform which we created by encapsulating the Linux advanced networking traffic controller into the Web-based application. NTuner can alter the network traffic according to the configuration. The decoding part composed of the packet receiver and sender, in/out buffer, code algorithm library and decoder. The decoder decodes the received video streaming data with some packets lost, and then sends them to VLC player. The streaming server and client all adopt UDP as the transfer protocol.

The main challenges presented in this system include how to apply and select different coding algorithms to video streaming, how to control network traffic within the emulation platform, and how to implement the whole integrated system.

B. Packet Loss Mechanism

We build up the network emulator upon the traffic control component of Fedora 4 that supports a number of advanced networking features [20]. Figure 4 shows how the Linux kernel processes incoming packets, and how it generates packets to be sent to the network. The input de-multiplexer examines the

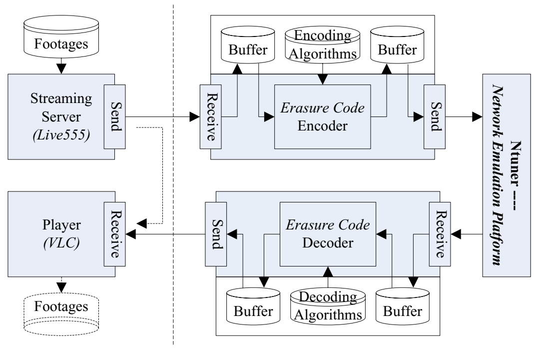


Fig. 3. System Architecture

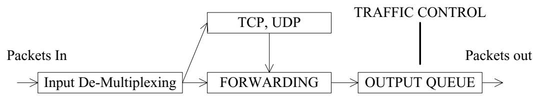


Fig. 4. Linux Traffic Control

incoming packets and determines if the packets are destined for the local node. If so, they are sent to the higher layer for further processing, otherwise to the forwarding block. The forwarding block, which may have also received locally generated packets from the higher layer, looks up the routing table and determines the next hop for the packets. After this, it queues the packets to be transmitted on the output interface. It is at this point that Linux traffic control comes into play, which can be used to build a complex combination of queuing disciplines, classes and filters, which control the packets that are sent onto the output interface.

We encapsulate traffic control into a Web-based application, named NTuner, which can be used easily to configure parameters for network bandwidth, delay, jitter, packet loss rate, and packet loss pattern.

C. Erasure Coding Implementation

Rizzo had showed the feasibility of FEC coding in software at high speeds [21]. As a typical example, we implement a systematic RS code in our testbed. RS codes are described in numerous coding theory books and papers [10] [22] [24]-[26]. Given a data polynomial *a(x)* of degree $k < n$, $n = 2^m$ (*k* is the number of information symbols and *n* is the code length), in Galois field GF(2^m) and a code generating polynomial *g(x)* of degree *p*, where $p \leq n - k$ and

$$g(x) = \prod_{i=0}^{p-1} \left(x + \alpha^i\right), \alpha(x) = \sum_{i=0}^{k} \alpha_i x^i \quad (1)$$

with α^i successive unity roots in GF(2^m) and α_i elements of the same field, the systematic encoding of $\alpha(x)$ is given by

$$C(x) = \alpha(x)x^{n-k} - R(x) \quad (2)$$

where *R(x)* is the remainder of the division $\alpha(x)x^{n-k}$ by *g(x)*. Galois field arithmetic is fundamental to RS encoding and decoding. There are several approaches to performing Galois field multiplication in software [22]. In our implementation, we represent the GF elements either in the index or polynomial format. In the index format, the number is the power of the GF primitive element. This format is convenient for multiplication, which just needs to add the powers modulo 2^{m-1}. In the polynomial format, the bits represent the coefficients of the polynomial representation of the number. This is the most convenient format for addition. The two formats are swapped via lookup tables. Encoding implementation is based on the use of a Linear Feedback Shift Register (LFSR) that provides a convenient way to perform polynomial division [27]. Decoding utilizes the Berlekamp-Massey algorithm [28].

The RS code offers optimal efficiency such that any available parity element can be substituted for any erased data element in the block. Parity-check information is generated through operations on a Galois field [23]. The computational cost of this process is related to the size of the field, where typical RS code implementations operate in a field of size 2^8, or one with 255 elements. The implementation in our testbed is for a generic RS code; user can configure the parameters *n*, *k* and *t* (*t* is the error correction capability of the code) according to different requirements for video source codes and bit rates. In the following experiments, the Galois field size is 2^3, so *n*=7 and *k*=5.

V. EXPERIMENTS

A. Experiment Setup

As an example, the system uses the standard video sequence Paris in the 4:2:0 YUV CIF format (352x288), which is encoded to the MPEG2 code, for the experiments on the testbed. Any video packet, with a size of 188 bytes, is aggregated into a single UDP packet. The RS (7, 5) code with the capability of recovering 2 lost packets per 7 packets is employed in the payload of each UDP packet. Figure 5 shows the UDP payload format for erasure correction using the RS (7, 5) code. Each packet includes 8 bytes header and 188 bytes payload (figure 6 shows the packet format). Those packets are encoded group by group, with each group having five packets. Each codeword (a column in figure 5) composed of seven symbols with 3 bits each. In these seven symbols, five of them come from five source video packets respectively and two of them are generated by applying the RS (7, 5) code algorithm on those five data symbols. The symbols generated by the RS (7, 5) code are organized into two new packets, called parity packets. On the NTuner platform, the random packet loss pattern is adopted and the packet loss rate is tuned from 0.2% to 5%.

B. Results Analysis

Video data are divided into groups for encoding and each group is composed of 5 video data packets. When the packets are transmitted over the platform, if the number of the lost packets is smaller or equal to 2 in any one group, and then the client can recover all the lost packets since this is within the erasure correction capacity of the RS (7, 5) code.

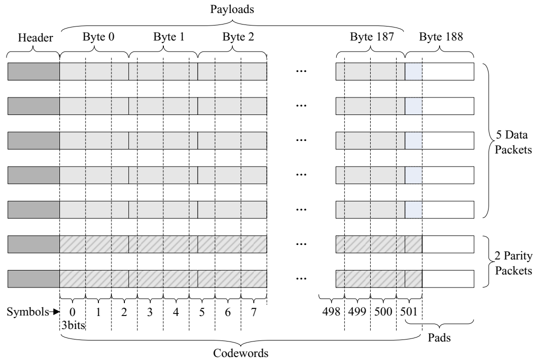


Fig. 5. UDP Payload Format for RS (7, 5)

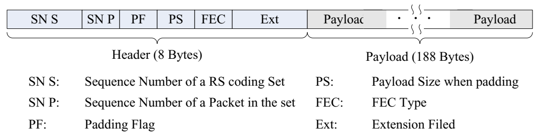


Fig. 6. Packet Format

In Table 1, parameter *R* is defined as the rate of successfully receiving at the receiver. Those groups that have some packets lost are divided into three Bands, i.e., Band A with one loss (or $R = 6/7$), Band B with two losses (or $R = 5/7$), and Band C with more than two losses (or $R < 5/7$), respectively. PLR here represents the original network packet loss rate, i.e., the packet loss rate for the system without erasure coding. The improved packet loss rate (I-PLR), with regards to PLR, is the packet loss rate measured at the output of the decoder or before the video replay for the system employing erasure coding. Based on Table 1, the packet loss statistics and comparison between I-PLR and PLR are shown in Figure 7 and 8, respectively. It is clearly from Table 1 and Figure 8 that the RS (7, 5) code can significantly enhance the replay quality for video streaming systems by effectively controlling packet losses. For example, the system with erasure

TABLE I
PACKET LOSS STATISTICS

PLR	*Band A(R=6/7)*	*Band B(R=5/7)*	*Band C(R<5/7)*	*I-PLR*
0.2%	66	5	0	0
0.5%	198	3	0	0
0.8%	319	8	0	0
1%	394	14	0	0
1.5%	560	22	1	0.007%
2%	801	36	1	0.007%
2.5%	886	64	5	0.036%
3%	1033	112	5	0.036%
3.5%	1168	127	9	0.065%
4%	1273	172	8	0.058%
4.5%	1405	185	16	0.116%
5%	1545	272	26	0.188%

Comments: ***PLR*** : Packet Loss Rate
R=6/7: No. of groups that have received 6 packets
R=5/7: No. of group that have received 5 packets
R<5/7: No. of group that have received less than 5 packets
I-PLR: Improved Packet Loss Rate

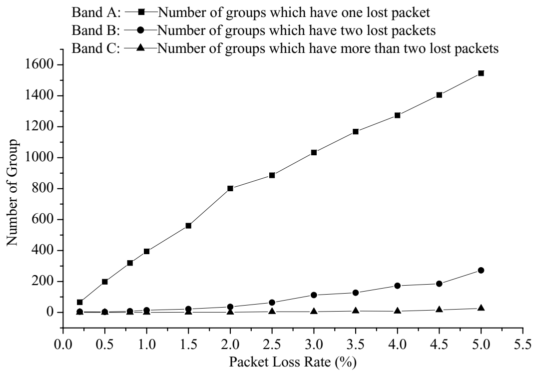


Fig. 7. Packet Loss Statistics

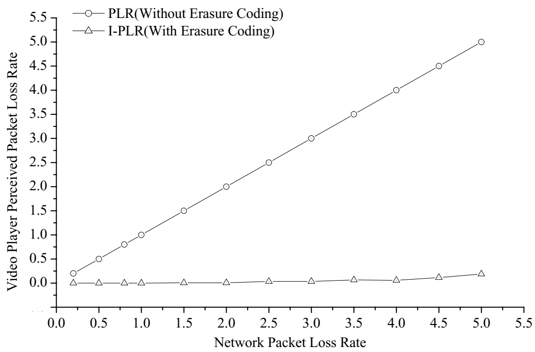
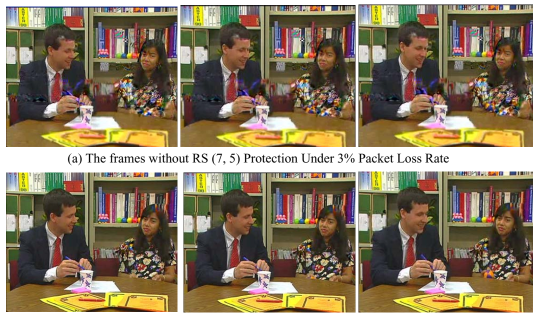


Fig. 8. PLR vs. I-PLR

coding can reduce the packet loss rate from 4.5% to 0.116%. We have also calculated the PSNR (Peak Signal to Noise Ratio) values, a metric normally used for objective quality assessment. However, the PSNR value is not consistent with the user perceived quality [31] sometimes, although for source coding quality assessment PSNR maybe is a good choice. Figure 9 shows that the user perceived quality gets more improvements, where three frames are used to compare the replay quality between the systems without and with erasure coding.

We assess the performance of the RS (7, 5) code by using the uncorrectable probability, which is the probability of an uncorrectable error that occurs when more than two packets are lost within one group. Figure 10 shows both the analytical values and experimental results for the uncorrectable probability. Although the two curves are not identical, mainly because the packet loss patterns generated are not real random, they all demonstrate that the performance improvement through erasure coding is significant. The decoding complexity of RS codes depends on a number of factors, including code length, error correcting ability and the decoding algorithm. Several techniques exist for solving the key equation, such as the Peterson-Gorenstein-Zierler (PGZ) algorithm, the Berlekamp-Massey Algorithm (BMA), Euclid's algorithm, and the Galois Field Fourier Transform approach [29]. Our system uses the BMA to solve the RS key equation. The total numbers of GF



(a) The frames without RS (7, 5) Protection Under 3% Packet Loss Rate

(b) The frames with RS (7, 5) Protection Under 3% Packet Loss Rate

Fig. 9. Playing Effect Comparison of With Coding and Without Coding

multiplications (*#GF-mult*) and GF additions (*#GF-adds*) for decoding the BMA are calculated as follows [30],

$$\#GF-mult = \frac{t}{2}(19t + 7n - 3) \quad (3)$$

$$\#GF-adds = \frac{t}{2}(19t + 7n - 5) \quad (4)$$

The above GF operations are roughly convert [30] to Binary Operations (BOPs) by assuming that a GF addition costs m BOPs and a GF multiplication $2m(2m - 1)$ BOPs. This gives a decoding cost of:

$$BOPs = \frac{tm}{2}(76tm + 28nm - 12m - 19t - 7n + 1) \quad (5)$$

Each successfully decoded vector results in mk decoded bits. Hence, one can write the decoding cost in BOPs per bit as:

$$BOPs/bit = \frac{t}{2k}(76tm + 28nm - 12m - 19t - 7n + 1) \quad (6)$$

which makes a good figure of merit for comparing RS codes of disparate lengths. We can calculate the decoding cost in BOPs per bit for the RS (7, 5) code, where n=7, k=5, t=1 and m=3:

$$\frac{t}{2k}(76tm+28nm-12m-19t-7n+1) = 71BOPs/bit \quad (7)$$

With the developing of the computer hardware technology, the decoding complexity is rapidly becoming a non-issue in many circumstances due to the availability of inexpensive high speed CPU or microchips for decoding.

VI. Conclusions

In this paper we present a testbed of erasure coding on video streaming system over lossy networks. In order to evaluate the performance of erasure coding applied to video streaming services, we have integrated the streaming server, erasure encoder/decoder, and network emulation platform into a controllable system that provides an effective means for fulfilling our objectives. In particular, we have implemented a RS code and tested its responses to difference network conditions in terms of packet loss rates. The experiment results have been analyzed and performance validation is carried out using the uncorrectable probability and processing complexity.

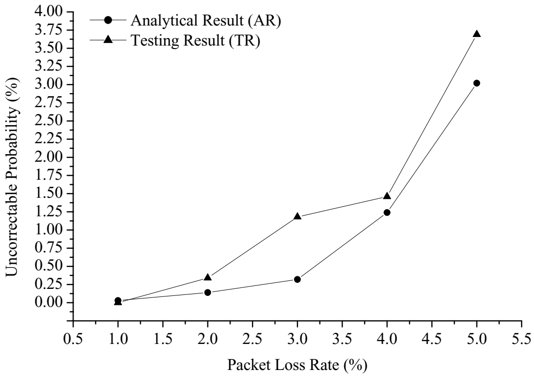


Fig. 10. Uncorrectable Probability

In our system, the erasure code employed can greatly reduce the packet loss rate at the client end in a wide range of network conditions. For example, given the original network packet loss rate of 4.5% it can be reduced to 0.116%, equivalent to a 97% improvement rate, by using an erasure code that requires around 28.57% redundant data. Obviously, this achievement can significantly and cost-effectively contribute to the quality enhancement of video streaming systems. We can also use the testbed to tune the code parameters to meet the requirements for different applications, bandwidth availability and network operating conditions.

We have also been investigating new code construction methods [16] and joined the PlanetLab [32], a global research network. It is expected that our current work will be extended to address challenges in a large-scale distributed network environment, including FEC-based MPEG4/H.264 video transmissions, unequal/adaptive erasure coding, and combined erasure coding and other error resilient techniques such as error concealment.

ACKNOWLEDGMENT

This work was supported by the grants from the Engineering and Physical Sciences Research Council of the United Kingdom and Xyratex Ltd. The authors wish to thank Tim Porter and Richard Haywood for useful discussions.

REFERENCES

[1] John Apostolopoulos, Wai-Tian Tan, Susie Wee, "Video Streaming: Concepts, Algorithms, and Systems", *Hewlett-Packard Laboratories Technical Report,* September 2002.

[2] J.H. Jeng and T.K. Truong, "On decoding of both errors and erasures of a Reed-Solomon code using an inverse-free Berlekamp-Massey algorithm", *IEEE Trans. Commun.,* vol. 47, no. 10, Oct. 1999, pp. 1488-1494.

[3] S. Floyd, V. Jacobson, S. McCanne, "A Reliable Multicast Framework for Light-weight Sessions and Application level Framing", In *Proceeding SIGCOMM '95,* the conference on Applications, technologies, architectures, and protocols for computer communication, Aug. 1995, Cambridge, MA.

[4] S. K. Kasera, J. Kurose and D. Towsley, "Scalable Reliable Multicast Using Multiple Multicast Groups", *ACM Sigmetrics performance evaluation review,* Vol. 25, No. 1, June 1997.

[5] P. Kauff, B. Makai, S. Rauthenberg, U. Golz, J.L.P. De Lamerillieure and T. Sikora, "Functional coding of video using a shapeadaptive DCT algorithm and an object-based motion prediction toolbox", *IEEE Transactions on Circuits and Systems for Video Technology,* Vol. 7, Issue: 1, Feb. 1997, pp.181 - 196.

[6] B. N. Leviene, D. B. Lavo and J. J. Garcia-Luna-Aceves, "The Case for Reliable Concurrent Multicasting Using Shared ACK Trees", *AVM Mulitmedia'96,* Boston MA USA.

[7] B. N. Leviene and J. J. Garcia-Luna-Aceves, "A comparison of known Classes of Reliable Multicast Protocols", In *Proceeding ICNP'96,* Columbus, Ohio, Oct. 1996.

[8] B. N. Leviene and J. J. Garcia-Luna-Aceves, "A comparison of Reliable Multicast Protocols", *Multimedia Systems (ACM/Springer),* Vol. 6. No. 5, August 1998.

[9] T. Zhang and Y. Xu, "Unequal packet loss protection for layered video transmission", *IEEE Trans. Broadcast.,* June 1999.

[10] S. Lin and D. J. Costello, "Error Control Coding: Fundamentals and Applications", *Englewood Cliffs,* New. Jersey: Prentice Hall, 1983.

[11] J. Nonnenmacher, E. W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission", *IEEE/ACM Trans. Networking,* vol. 6, no. 4, Aug. 1998.

[12] S. Paul, K. K. Sabnani, J. C. Lin, and S. Bhattacharyya, "Reliable multicast transport protocal (RMTP)", *IEEE J. Select. Areas Commun.,* vol. 15, no. 3, Apr. 1997.

[13] L. Rizzo, "Effective erasure codes for reliable computer communication protocols", *ACM Computer Communication Review,* vol. 27, no. 2, April 1997.

[14] XU Y, ZHANG T. "Variable shortened-and-punctured Reed-Solomon codes for packet loss protection",J. *IEEE Transactions on Broadcasting,* 2002, 48(3): 237-245.

[15] Donghui Chen, Bo Rong, N. Shayan, M. Bennani, J. Cabral, M. Kadoch, and A.K. Elhakeem. "Interleaved FEC/ARQ coding for QoS multicast over the Internet", *CAN. J. ELECT. COMPUT. ENG.,* VOL. 29, NO. 3, JULY 2004.

[16] X.-H. Peng, "Erasure-control coding for distributed networks", *IEE Proceedings Communications,* December 2005, Volume 152, Issue 6, p. 1075-1080.

[17] T. Nguyen and A. Zakhor, "Distributed Video Streaming with Forward Error Correction", *Packet Video Workshop,* April 2002.

[18] Y. Xu and T. Zhang, "An adaptive redundancy technique for wireless indoor multicasting", in *Proc. Fifth IEEE Symp. Computers and Commun.,* Antibes-Juan les Pins, France, July 3-6, 2000.

[19] Huahui Wu, Mark Claypool, Robert E. Kinicki, "Adjusting forward error correction with quality scaling for streaming MPEG", *NOSSDAV* 2005: 111-116.

[20] Saravanan Radhakrishnan, "Linux - Advanced Networking Overview Version 1", 22 August,1999.

[21] L. Rizzo. On the Feasibility of Software FEC. Technical Report *DEIT Technical Report LR-97013,* University of Pisa, 1997. Available at http://www.iet.unipi.it/ luigi/softfec.ps.

[22] S. Mamidi, M. Schulte, D. Iancu, A. Iancu, and J. Glossner, "Instruction Set Extensions for Reed-Solomon Encoding and Decoding," *IEEE 16th International Conference on Application-specific Systems, Architectures and Processors,* pp. 364-369, July 2005.

[23] J. A. Cooley, J. L. Mineweaser, L. D. Servi, and E. T. Tsung, "Software-based Erasure Codes for Scalable Distributed Storage", 20th *IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies,* San Diego, CA, 2003.

[24] E. R. Berlekamp, "Algebraic Coding Theory," *Aegean Park Press,* 1984.

[25] R. E. Blahut, "Algebraic Codes for Data Transmission," *Cambridge University Press,* 2003.

[26] S. B. Wicker, "Error Control Systems for Digital Communication and Storage," Prentice Hall Inc., 1995.

[27] Y. Katayama and S. Morioka, "One-Shot Reed-Solomon Decoder," *Annual Conference on Information Science and Systems,* vol. 33, pp. 700-705, 1999.

[28] Richard E. Blahut, "Theory and Practice of Error Control Codes", Addison-Wesley Publishing Company, 1983.

[29] Andreas G. Yankopolus. Adaptive Error Control for Wireless Multimedia. PhD thesis (Georgia Tech's), April 2004.

[30] L. K. Rasmussen, M. J. Bartz, and S. B. Wicker, "A comparison of trellis coded and reed-solomon coded hybrid-ARQ protocols over slowly fading rayleigh channels," *Wireless Personal Communications,* vol. 2, pp. 393-413, April 1996.

[31] Y Zhong, I Richardson, A Sahraie and P Mcgeorge, "Qualitative and quantitative assessment in video compression", 12th *European Conference on Eye Movements,* 20-24 August 2003, Dundee,Scotland.

[32] http://www.planet-lab.org/