# FEC for VoIP using Dual-Delay Streaming Codes

*Abstract*—**We introduce a new class of forward error correction (FEC) codes for VoIP communications which supports different recovery delay depending on the channel conditions. Specifically, our code recovers challenging long bursts with close to theoretical minimum delay to ensure recovered packetcan meet playback deadlines, while improving conversational interactivity by allowing lower recovery delay when burst length decreases. These DD codes are shown to achieve lower residual loss rates when compared to existing codes over a wide range of parameters of the Gilbert-Elliott channel. Experiments over real world packet traces further show gains in perceptual quality achieved by the DD codes in terms of the ITU-T G.107 E-model.**

*Index Terms*—**VoIP Audio Quality, Interactive Multimedia Applications, Application Layer Forward Error Correction (AL-FEC), Burst Erasures, E-model**

## I. INTRODUCTION

Interactive voice applications are sensitive to both network losses and delay [1]–[4]. Forward error correction (FEC) is a common practice to recover lost packets in a round-trip-time independent manner [5]–[11]. Nevertheless, it incurs a *recovery delay* associated with the time necessary to collect enough parity packets for loss recovery. This recovery delay depends on both the type of FEC used, and the actual loss pattern. In a typical dynamic environment, losses can range from simple isolated losses to occasional long bursts. An important challenge is to design a FEC code that can support recovery from challenging long bursts with the lowest possible recovery delay without large penalties to recovery time of short bursts. In such a way, an application can enjoy low delay communication during times with isolated losses and short bursts, but can still practically recover from long bursts which can severely affect the perceptual audio quality.

Many flavors of FEC with distinct recovery delay characteristics have been developed for use in different communication settings. For file transfer or buffered streaming playback, a common choice is to exploit the relaxed delay requirement by using a digital fountain type code to achieve high rate and low complexity. For interactive applications, Reed Solomon (RS) and Random Linear Codes (RLC) [12] can simultaneously recover $m$ lost packets with only $m$ parity packets. Interestingly, this does not lead to minimum recovery delay. Since every audio packet typically corresponds to a different timestamp, lower recovery delay can be realized by FEC schemes that preferentially recover earlier losses without waiting to simultaneously recover all losses. This is the basis of streaming codes, an example of which is the Maximally Short (MS) codes [13], [14] that achieve the minimum possible recovery delay for single burst of a given length. However, MS cannot recover from as few as two erasures in random positions. Other variations of streaming codes over burst erasure channels are studied in [15]–[17]. In [18]–[20], a family of streaming codes that recover from both patterns, long bursts and few random erasures, were proposed. They achieve recovery delay for long bursts close to the theoretical minimum, but fail to provide low delay even for a single loss.

In practice, VoIP systems are equipped with adaptive playback buffers. Hence, an FEC that recovers quickly from single or two erasures while also capable of recovering long bursts is of interest. In other words, we want an FEC scheme that achieves close to minimum possible recovery delay for long bursts, so that more loss patterns can be practically recovered. At the same time, we want shorter recovery delay for simple one or two losses to improve interactivity over channels with light losses. In this work, we show that this is in fact achievable through a careful combination of two baseline schemes, random linear codes and maximally short codes. The resulting dual-delay (DD) code is shown to outperform existing codes over Gilbert-Elliott channels in terms of residual loss rates and delay adaptability. Furthermore, we use the ITU-T G.107 E-model to quantify perceptual quality gains achieved over real world packet traces.

Adaptive schemes for real-time streaming have been studied in the literature from different perspectives. [21] studied adapting the transmission strategy when feedback is available. A class of codes that adapts its playback rate to the channel erasure probability was proposed in [22]. The most relevant work is [23], [24], which studied the problem of adapting the recovery delay to the burst length. Similar to MS codes, these codes are sensitive to isolated losses. In [25] a robust extension that allows for recovering a small number of isolated losses at both users is proposed. However, these codes incur a relatively long delay when the channel introduces a single erasure.

The rest of the paper is organized as follows. We start by introducing the streaming setup in Section II. Section III illustrates two examples of the proposed DD codes at rate $1/2$. The encoding and decoding steps of the general construction are provided in Section IV. Numerical comparisons to existing codes are discussed in Section V. Simulation results over Gilbert-Elliott channels are provided in Section VI. The E-model $R$-ratings achieved by different codes over real world packet traces is presented in Section VII. Finally, conclusions are provided in Section VIII.

## II. STREAMING SETUP

We consider a source that produces one packet of size $k$ symbols every $t_s$ seconds, namely $\mathbf{s}[t]$ for $t = 0, t_s, 2t_s, \ldots$. In a typical VoIP application, $t_s$ can be $5, 10, 20$ or $30$ milliseconds and $k$ is few tens of bytes. For simplicity, we drop $t_s$ in all packet indices and use $\mathbf{s}[0], \mathbf{s}[1], \mathbf{s}[2], \ldots$ instead. The FEC

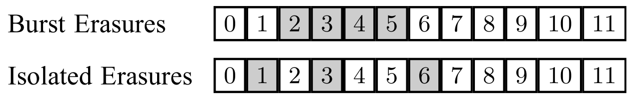
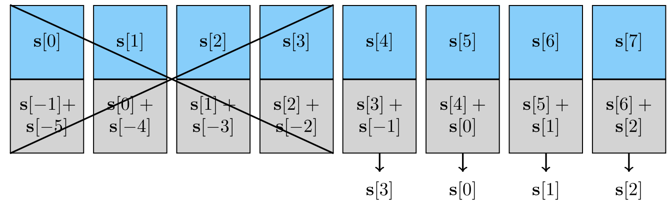
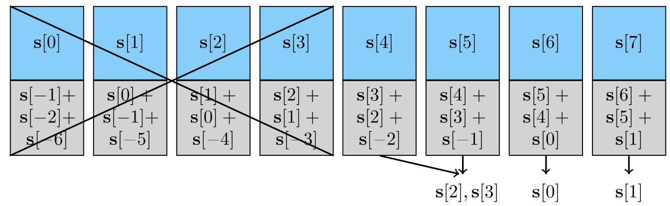



Fig. 1: Examples of two different erasure patterns. The first corresponds to a single erasure burst of length $B = 4$ packets, whereas the second is an isolated erasure pattern with $N = 3$ losses. Gray and white squares correspond to erased and perfectly received packets respectively.




(a) $(R = 1/2, M = 1, \Delta = 4)$ DD Code




(b) $(R = 1/2, M = 2, \Delta = 4)$ DD Code

Fig. 2: Channel packets of DD codes at $R = 1/2$, $\Delta = 4$ and memories $M = 1, 2$. The decoding steps in case of a burst of length $B = 4$ are illustrated for both constructions. We drop the RLC code coefficients $\alpha_j$ for simplicity.

encoder combines the source packet stream in a causal fashion and generates another stream of channel packets $\mathbf{x}[t]$ of size $n \geq k$ symbols. The rate of the code is $R = k/n$. In case of systematic codes, $\mathbf{x}[t] = (\mathbf{s}[t], \mathbf{p}[t])$, where $\mathbf{p}[t]$ denotes the parity-check stream. Systematic codes are of practical interest, since no decoding is needed in case of no losses.

We consider a class of erasure channels that introduces either bursts or random isolated erasures. In particular, the channel can introduce either a single erasure burst of maximum length $B$ packets or up-to $N < B$ erasures in random positions. The channel output is $\mathbf{y}[t] = \star$ when an erasure occurs at time $t$ and $\mathbf{y}[t] = \mathbf{x}[t]$ otherwise. Although the channel model we consider is somewhat simplistic, we will verify that the insights gleaned from their analysis are valuable in our experiments on statistical channels and real world traces.

In case of isolated erasures, the receiver recovers every erased packet with a maximum delay of $T_1$ packets, i.e.,

$$\mathbf{s}[t] = g_t^{(1)}(\mathbf{y}[0], \ldots, \mathbf{y}[t + T_1]), \tag{1}$$

whereas in the burst erasure case, the receiver tolerates a slightly longer delay of $T_2 > T_1$ packets,

$$\mathbf{s}[t] = g_t^{(2)}(\mathbf{y}[0], \ldots, \mathbf{y}[t + T_2]). \tag{2}$$

The special case of $T_1 = T_2 = T$ is studied in [18], [20], [26].

## III. Dual-Delay Codes: $R = 1/2$ Examples

We start by providing examples for DD codes at $R = 1/2$ where the construction is in its simplest form. In this special case, the parity-check packets are a combination of a Random Linear Code and a shifted Repetition code, i.e.,

$$\mathbf{p}[t] = \mathbf{p}_1[t] + \mathbf{p}_2[t] = \sum_{j=1}^{M} \alpha_j \mathbf{s}[t-j] + \mathbf{s}[t - M - \Delta] \tag{3}$$

where the first term corresponds to the parity-check packets of a random linear code of rate $R = 1/2$ and memory $M$ packets, whereas the second term corresponds to a repetition code with a shift of $M + \Delta$ packets. The coefficients of the random linear code $\alpha_j$ for $j \in \{1, 2, \ldots, M\}$ in (3) are selected such that every $M$ consecutive parity-check packets are linearly independent in the combined source packets. We denote a DD code of rate $R$, memory $M$ and shift $\Delta$ by $(R, M, \Delta)$ DD code. The following are two examples of DD codes at $R = 1/2$ that will be used in the subsequent sections.

### A. $(R = 1/2, M = 1, \Delta = 4)$ *Dual-Delay Code*

Substituting in (3), the overall parity-check of the $(1/2, 1, 4)$ DD code is,

$$\mathbf{p}[t] = \mathbf{s}[t-1] + \mathbf{s}[t-5], \tag{4}$$

where $\alpha_1 = 1$ is used. The channel packets $\mathbf{x}[t] = (\mathbf{s}[t], \mathbf{p}[t])$ for $t \in [0, 7]$ are shown in Fig. 2a. One can easily verify that a single erasure at time $t = 0$ can be recovered using $\mathbf{p}[1]$, i.e., $T_1 = 1$. In case of a burst of length $B = 4$ in the interval $[0, 3]$, shown in Fig. 2a, the decoder uses $\mathbf{p}[4] = \mathbf{s}[3] + \mathbf{s}[-1]$ to recover $\mathbf{s}[3]$ with a delay of one packet. The parities $\mathbf{p}[5], \mathbf{s}[6], \mathbf{p}[7]$ can then be used to recover $\mathbf{s}[0], \mathbf{s}[1], \mathbf{s}[2]$ respectively, i.e., with a delay of $T_2 = 5$ packets.

### B. $(R = 1/2, M = 2, \Delta = 4)$ *Dual-Delay Code*

In this construction, we use a memory $M = 2$ RLC code and a repetition code with shift $\Delta + M = 6$. The overall parity-check of this DD code is,

$$\mathbf{p}[t] = \alpha_1 \mathbf{s}[t-1] + \alpha_2 \mathbf{s}[t-2] + \mathbf{s}[t-6]. \tag{5}$$

The channel packets $\mathbf{x}[t]$ for $t \in [0, 7]$ are shown in Fig. 2b.

In case of a single erasure at time $t = 0$, the erased packet $\mathbf{s}[0]$ can be directly recovered from $\mathbf{p}[1]$, i.e., with a delay of one. If the channel introduces $N = 2$ erasures, they can either be isolated or in a burst. If they are isolated, each packet can be recovered with a delay of one packet, whereas if they happen in a burst, e.g., $\mathbf{s}[0]$ and $\mathbf{s}[1]$ are erased, the decoder first uses,

$$\mathbf{p}[3] = \alpha_1 \mathbf{s}[2] + \alpha_2 \mathbf{s}[1] + \mathbf{s}[-3],$$

to decode $\mathbf{s}[1]$ at time 3, i.e., with a delay of 2 packets. Now the decoder can go back to subtract $\mathbf{s}[1]$ from

$$\mathbf{p}[2] = \alpha_1 \mathbf{s}[1] + \alpha_2 \mathbf{s}[0] + \mathbf{s}[-4]$$

to recover $\mathbf{s}[0]$ and the corresponding delay is 3 packets.

In case of a burst of length $B = 4$ in the interval $[0, 3]$, shown in Fig. 2b, the decoder can use the parities $\mathbf{p}[4]$ and $\mathbf{p}[5]$ to recover $\mathbf{s}[t+2]$ and $\mathbf{s}[t+3]$ with a delay of 3 and 2 packets respectively similar to the case of $N=2$ erasures discussed above. Then $\alpha_1\mathbf{s}[5]+\alpha_2\mathbf{s}[4]$ is computed and subtracted from $\mathbf{p}[6]$ to recover $\mathbf{s}[0]$ with a delay of $T_2=6$ packets. Similarly, $\mathbf{s}[1]$ is decoded from $\mathbf{p}[7]$ with the same delay.

# IV. Dual-Delay Codes: General Construction

The construction of a general rate $(R,M,\Delta)$ DD code requires careful combination of two baseline codes, Random Linear Codes (RLC) and Maximally-Short (MS) codes. Hence, we start by briefly summarizing these constructions.

## *A. Random Linear Convolutional (RLC) Codes*

The parity-check packet of a $(R,M)$ RLC code of rate $R=k/n$ and memory $M$ is given by,

$$\mathbf{p}[t]=\sum_{j=1}^{M}\mathbf{s}[t-j]\cdot\mathbf{P}_j, \tag{6}$$

where the $\alpha_j$ scalars in (3) are replaced by the $k\times(n-k)$ parity-check matrices $\mathbf{P}_j\neq 0$ for $j\in\{1,\dots,M\}$. RLC codes can recover the maximum number of isolated erasures at a fixed rate and delay. A $(R,M)$ RLC code can recover from

$$N=(1-R)(j+1),\qquad j=1,2,\dots,M \tag{7}$$

erasures within a delay of $T=j$ packets. In the dual-delay setup in Section II, this is equivalent to selecting $T_1=j\in\{1,\dots,M\}$ to achieve,

$$N=(1-R)(j+1), B=(1-R)(M+1), T_2=M. \tag{8}$$

## *B. Maximally-Short (MS) Codes*

The general construction of the MS code requires splitting the source packets into two groups, $\mathbf{s}[t]=(\mathbf{u}[t],\mathbf{v}[t])$ whose sizes are $n-k$ and $2k-n$, respectively. The parity-check packet of a $(R,\Delta)$ MS code of rate $R=k/n$ is,

$$\mathbf{p}[t]=\sum_{j=1}^{\Delta-1}\mathbf{v}[t-j]\cdot\mathbf{P}_j+\mathbf{u}[t-\Delta], \tag{9}$$

where $\mathbf{P}_j$ are the $(2k-n)\times(n-k)$ parity-check matrices of a $(\frac{1-R}{R},\Delta-1)$ RLC code. At a given rate $R$ and delay $T=\Delta$, MS codes recover from the longest possible burst,

$$B=\frac{1-R}{R}\Delta. \tag{10}$$

Due to the use of repetition codes, the MS codes incur a long delay of $\Delta$ packets when recovering from a single loss. For the same reason, they fail to recover from as few as two isolated losses. Hence, a $(R,\Delta)$ MS code achieves

$$N=1,\quad B=\frac{1-R}{R}\Delta,\quad T_1=T_2=\Delta \tag{11}$$

in our dual-delay setup which clearly makes it not delay adaptive as it incurs the same delay for both patterns.

We note that in the special case of $R=1/2$, the MS code in (9) is a shifted repetition code, $\mathbf{p}[t]=\mathbf{s}[t-\Delta]$ which was used in the two DD code examples in Section III.

## *C. Dual-Delay Encoder*

The encoding steps of a $(R,M,\Delta)$ DD code, for $R\geq 1/2$, $M<\Delta$, are as follows:

1) Apply a $(R,M)$ RLC code to the source packets $\mathbf{s}[\cdot]$ to generate the parity-check packets $\mathbf{p}_1[\cdot]$ of size $n-k$ symbols.
2) Apply a $(R,\Delta)$ MS code to the source packets $\mathbf{s}[\cdot]$ to generate another set of parities $\mathbf{p}_2[\cdot]$ of size $n-k$ symbols.
3) Concatenate both streams of parity-checks after shifting the latter by $M$ time slots. Using (6) and (9), the overall parity-check packet at time $t$ can be written as,

$$\begin{aligned}\mathbf{p}[t]&=\mathbf{p}_1[t]+\mathbf{p}_2[t-M]\\&=\sum_{j=1}^{M}\mathbf{s}[t-j]\cdot\mathbf{P}^1_j+\sum_{j=1}^{\Delta-1}\mathbf{v}[t-M-j]\cdot\mathbf{P}^2_j\\&\quad+\mathbf{u}[t-M-\Delta],\end{aligned} \tag{12}$$

where $\mathbf{P}^1_j$ and $\mathbf{P}^2_j$ are $k\times(n-k)$ and $(2k-n)\times(n-k)$ parity-check matrices of $(R,M)$ and $(\frac{1-R}{R},\Delta-1)$ RLC codes, respectively.
4) The channel packet at time $t$ is $\mathbf{x}[t]=(\mathbf{s}[t],\mathbf{p}[t])$ and the overall rate of the code is $R$.

## *D. Dual-Delay Decoder*

Consider a channel that introduces a single erasure burst in the interval $[t,t+B-1]$. The decoder first skips the parity-check packets in the interval $[t+B,t+B+M-1]$. From (12), the parity-check packets $\mathbf{p}_1[t+B+M],\dots,\mathbf{p}_1[t+B+M+\Delta-1]$ combine source packets from time $t+B$ and later which are not part of the burst. Hence, they are subtracted from the corresponding $\mathbf{p}[\cdot]$ to recover $\mathbf{p}_2[t+B],\dots,\mathbf{p}_2[t+B+\Delta-1]$. These parities together with the source packets in the interval $\mathbf{s}[t+B],\dots,\mathbf{s}[t+B+\Delta-1]$ are used by the $(R,\Delta)$ MS code to recover all erased packets with a delay of $T_2=M+\Delta$ packets provided that the burst length does not exceed,

$$B=\frac{1-R}{R}\Delta. \tag{13}$$

In case of $N$ isolated erasures starting at time $t$, we first consider the interval $[t,t+M]$. The parities $\mathbf{p}_2[t-M],\dots,\mathbf{p}_2[t]$ in the interval $[t,t+M]$ combine source packets from time $t-1$ and earlier which are not erased. These parities can be subtracted from the corresponding $\mathbf{p}[\cdot]$ in (12) to recover the $(R,M)$ RLC parities $\mathbf{p}_1[t],\dots,\mathbf{p}_1[t+M]$. The interval $[t,t+M]$ has at most $N$ erasures and $\mathbf{s}[t]$ is thus recoverable within a delay of $T_1=M$ packets provided that the number of erasures is not more than,

$$N=(1-R)(M+1). \tag{14}$$

The effect of $\mathbf{s}[t]$ can now be canceled from all future parity-check packets and the next interval $[t+1,t+M+1]$ can be considered to recover $\mathbf{s}[t+1]$ by time $t+M+1$ in a similar fashion. The decoder repeats the previous step recursively until all $N$ erased packets are recovered. We point the reader to [20] discussion on recursive recovery.

**Theorem 1.** *There exists a $(R,M,\Delta)$ DD code which recover from $N$ isolated erasures within a delay of $T_1=M$ and an erasure burst of length $B$ within a delay of $T_2=M+\Delta$ where $N$ and $B$ satisfy the following inequality,*

$$\frac{RB}{1-R}+\frac{N}{1-R}\geq M+\Delta+1. \tag{15}$$

We note that the values of $B$ and $N$ in (13) and (14) satisfy (15) with equality.

**Remark 1.** *At a given rate $R$, increasing $\Delta$ of the DD code while fixing $M$ increases the burst erasure correction capability $B$ and its corresponding delay $T_2$ without affecting $N$ and $T_1$. Moreover, varying the parameters $M$ and $\Delta$ while keeping the sum $M+\Delta$ fixed allows for achieving different $(B, N)$ pairs that satisfy* (15) *at a fixed $T_2$ and $R$.*

While our construction holds for any value of $R$, in the rest of the paper we will study the special case of $R = 1/2$. Practical systems such as Skype are known to use this level of redundancy in the presence of packet losses [10].

### *E. Improved Construction*

The DD construction in the previous section was shown to achieve $B$ and $N$ values that satisfy (15) with equality. However, we believe that larger values of $B$ and $N$ can be achieved by reducing the memory of the constituent RLC code used. Let us consider the DD example in Section III-B. Note that according to (13) and (14), the values $N = 2$, $T_1 = 3$, $T_2 = 6$ at $R = 1/2$ can be achieved using $M = 3$ and $\Delta = 3$ which results in $B = 3$. In the $(1/2, 2, 4)$ DD in Section III-B, we show that one can use $M = 2$ to achieve the same $N$ and $T_1$ while increasing $B$ to 4 at $T_2 = 6$ by using $\Delta = 4$.

In other words, reducing the memory of the RLC code to $M = 2$ in this construction allowed for a longer $B$ without changing $N$, $T_1$ or $T_2$. A similar argument can be applied for all $(R = 1/2, M = 2, \Delta)$ codes to achieve $N = 2$, $T_1 = 3$, $B = \Delta$ and $T_2 = \Delta+2$. We conjecture that this result extends to all $(R, M, \Delta)$ DD code for $R \geq 1/2$ and $M < \Delta$ and the trade-off in (15) can be enhanced to,

$$\frac{R}{1-R}(B+N) \geq M+\Delta. \tag{16}$$

Numerical search reveals that a $(1/2, M)$ RLC code for $M \in \{1, \ldots, 8\}$ achieves[1],

$$N = M, \qquad T_1 = 2M-1. \tag{17}$$

This helps constructing $(1/2, M, \Delta)$ DD codes that achieve (17) and

$$B = \Delta, \qquad T_2 = \Delta+M, \tag{18}$$

which satisfies the trade-off in (16) at $R = 1/2$ and $M \in \{1, \ldots, 8\}$. Compared to (15), this is equivalent to doubling the value of $N$ at the same $B$, $T_1$ and $T_2$ when $R = 1/2$.

## V. Numerical Comparisons

In this section, we compare the erasure correction capabilities and recovery delays of existing codes in the literature to DD codes. We focus on rate $R = 1/2$ systematic codes, i.e, $\mathbf{x}[t] = (\mathbf{s}[t], \mathbf{p}[t])$ where $\mathbf{s}[t]$ and $\mathbf{p}[t]$ are of the same size. The following FEC codes are considered.

- **Reed-Solomon Code:** Reed-Solomon (RS) codes are block codes but can be adapted to the streaming setup. A $(n, k)$ RS code can be applied to a block of $k$ consecutive source packets and the generated $n-k$ parity packets can be concatenated to the next block of $k$ packets. Here is an example of a $(6, 3)$ RS whose codewords starting at time $t = 0$ and $t = 3$ are,
$$(\mathbf{s}[0], \mathbf{s}[1], \mathbf{s}[2], \mathbf{p}[3], \mathbf{p}[4], \mathbf{p}[5])$$
$$(\mathbf{s}[3], \mathbf{s}[4], \mathbf{s}[5], \mathbf{p}[6], \mathbf{p}[7], \mathbf{p}[8]).$$
The parity-check packets of the first codeword $\mathbf{p}[3]$, $\mathbf{p}[4]$ and $\mathbf{p}[5]$ are concatenated to the source packets of the second codeword $\mathbf{s}[3]$, $\mathbf{s}[4]$ and $\mathbf{s}[5]$, respectively, before transmission.
- **Random Linear Code:** The parity-check equation of a $(R = 1/2, M)$ RLC code is given by,
$$\mathbf{p}[t] = \sum_{j=1}^{M} \alpha_j \cdot \mathbf{s}[t-j] \tag{19}$$
We use a $(1/2, 5)$ RLC code which achieves $B = N = 3$ and $T_2 = T_1 = 5$ (cf. (7))
- **Maximally-Short Code:** At $R = 1/2$, MS codes are shifted repetition codes. We use a $(1/2, 5)$ MS whose parities are $\mathbf{p}[t] = \mathbf{s}[t-5]$. It recovers from $B = 5$ with delay $T_2 = 5$ and $N = 1$ with the same delay $T_1 = 5$.
- **Embedded-RLC:** Embedded-RLC (ERLC) codes are proposed in [18] to capture the trade-off between $N$ and $B$ for a given $T$. At $R = 1/2$, the $(R, M, \Delta)$ ERLC code is a shifted RLC code, i.e.,
$$\mathbf{p}[t] = \sum_{j=\Delta}^{M} \alpha_j \cdot \mathbf{s}[t-j]. \tag{20}$$
This is similar to setting $\alpha_j = 0$ for $j < \Delta$ in the $(1/2, M)$ RLC code in (19). Varying $\Delta$ between 0 and $M$ achieves different $N$ and $B$ pairs at a fixed delay and rate. In this example, we consider a $(1/2, 5, 4)$ ERLC code which achives $N = 2$, $B = 4$, $T_1 = 5$ and $T_2 = 4$.
- **Dual-Delay Code:** We use the $R = 1/2$ DD codes in Sections III-A and III-B.

[1]The $(1/2, M)$ RLC code in Section IV-A was shown to recover from $(1-R)(M+1)$ within a delay of $T_1 = M$. We show that a shorter memory $(1/2, (M-1)/2)$ RLC code can achieve the same values of $N$ and $T_1$.

TABLE I: Parameters and correction capabilities of FEC codes in Section V. For each code, the following is indicated: the maximum perfectly recoverable burst length and its corresponding delay, delay of a single loss, two losses, three losses and the number of recoverable packets in case of a long burst $B \geq 6$.

| Code | B,T | T(N=1) | T(N=2) | T(N=3) | B≥6 |
|---|---|---|---|---|---|
| $(6, 3)$ RS | 3,5 | 3 | 4 | 5 | 1 |
| $(1/2, 5)$ RLC | 3,5 | 1 | 3 | 5 | 1 |
| $(1/2, 5)$ MS | 5,5 | 5 | $\infty$ | $\infty$ | 5 |
| $(1/2, 5, 4)$ ERLC | 4,5 | 4 | 5 | $\infty$ | 1 |
| $(1/2, 1, 4)$ DD | 4,5 | 1 | 5 | $\infty$ | 4 |
| $(1/2, 2, 4)$ DD | 4,6 | 1 | 3 | $\infty$ | 4 |

Table I shows the achievable delays over different erasure patterns for the $(6, 3)$ RS, $(1/2, 5)$ RLC, $(1/2, 5)$ MS, $(1/2, 5, 4)$ ERLC, $(1/2, 1, 4)$ DD and $(1/2, 2, 4)$ DD codes. The MS code recovers from the longest burst $B = 5$ but requires the same relatively long delay when recovering a single loss and fails to recover from two isolated losses. The ERLC code is more robust to isolated losses at a slight reduction in the maximum recoverable burst. However, its recovery delay is still long $T = 4$ in case of single loss. The RLC code adapts its recovery delay to the number of erasures at a price of short burst correction capability. The RS code has similar properties but is less adaptive because of the block code structure. The proposed DD codes have similar decoding capabilities as the ERLC code while adapting its delay to the number of erasures. Furthermore, the repetition codes used in MS and DD codes allow them to partially recover from long bursts. In particular, a repetition code with shift $\Delta$ can recover the last $\Delta$ packets in any burst of length $B > \Delta$. In such patterns, RS, RLC and ERLC codes can only recover the last packet within the deadline.

## VI. Simulation Results

In this section, we compare the performance of codes in Section V, as well as other codes, over a wide range of parameters of the Gilbert-Elliott (GE) channel. The GE channel is a two state Markov model, with transition probabilities from good-to-bad and bad-to-good given by $\alpha$ and $\beta$, respectively. This results in geometric burst and inter-burst length distributions of mean $1/\beta$ and $1/\alpha$, respectively. The erasure probability in the good-state is $\epsilon \in [0, 1)$, and in the bad-state is 1. The overall packet loss rate of a GE$(\alpha, \beta, \epsilon)$ is,

$$PLR(\alpha, \beta, \epsilon) = \frac{\alpha}{\alpha + \beta} + \frac{\beta}{\alpha + \beta}\epsilon. \tag{21}$$

### A. Residual Loss Rates

Fig. 3 provides the results of thre experiments over a set of Gilbert-Elliott channel realizations. In Fig. 3a, we study the effect of increasing the i.i.d. loss rate in the good-state in the range $\epsilon \in [0, 0.05]$ on the performance of different codes while fixing $\beta = 1/2$, i.e., the mean burst length is 2 packets. The percentage of packet erasures in the good-state, $100\epsilon(\%)$, is on the x-axis whereas the percentage of residual packet losses at $T = 5$ packets is plotted on the y-axis for different codes. The MS code that recovers from the maximum burst length, $B = 5$, achieves the minimum loss rate at small values of $\epsilon$. However, as $\epsilon$ increases, its performance deteriorates faster than any other code since it is designed to recover single isolated losses only. It is also worth noting that the MS code fails to recover any erased packet within a delay shorter than $T = 5$ packets. The performance of ERLC, RS and RLC codes do not deteriorate as quickly when $\epsilon$ increases. However, their higher loss rate is determined by the failure to recover (even partially) from long bursts. On the other hand, the proposed DD code is robust to isolated losses and thus maintains a slow deterioration in the performance as $\epsilon$ increases. Its capability to partially recover from bursts longer than 4 allows it to start at a low loss rate. This is besides recovering the largest fraction of packets within a short delay, $T = 1$ packet as will be shown in the next section.

Fig. 3b studies the effect of the mean burst length on the performance of the codes in Table I. Here, we again fix $\alpha = 0.01$, and we choose $\epsilon = 0.025$ (the middle point in Fig. 3a) and vary $\beta \in [1/3, 4/5]$ to achieve mean burst lengths (MBL) of 1.25 to 3 packets. The residual packet loss rate is plotted on the y-axis vs. MBL on the x-axis. We again see that at $T = 5$, the DD code outperforms all other codes in almost all the range of $\beta$. The MS code is considered second best at this delay, however, this code is not delay adaptive as it fails to recover any erased packet at a shorter delay. RLC and RS codes are delay adaptive but they achieve higher loss rates at $T = 5$ compared to the proposed DD code. The ERLC code fail to partially recover any of the packets when lost in a burst and is also not delay adaptive.

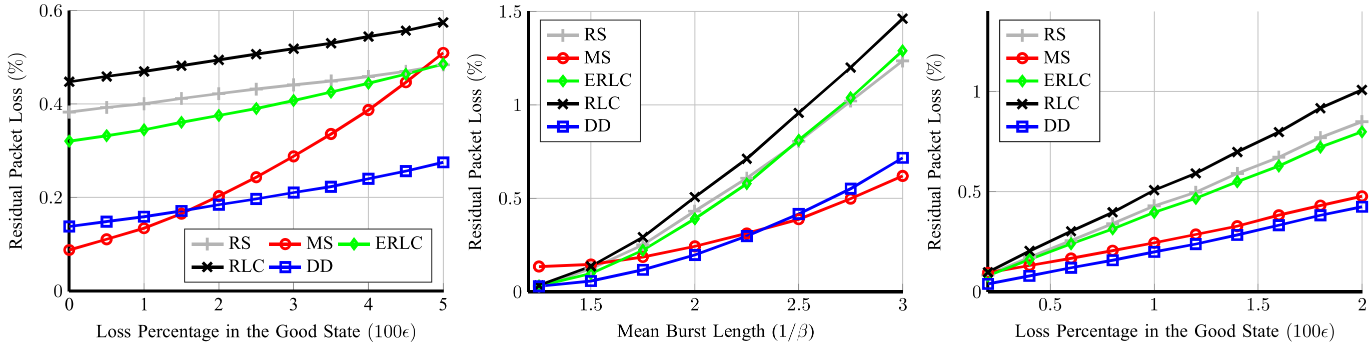
The effect of the guard period separation between bursts is studied in Fig. 3c. As $\alpha$ increases on the x-axis, the average separation between bursts $1/\alpha$ decreases. Code with a long memory, e.g., RLC, ERLC and RS, tries to recover all lost packets simultaneously. As the bursts get closer to each other, this incurs long delays. On the other hand, MS and DD codes, can partially recover from such pattern with no extra delay due to their constituent repetition codes.

Fig. 4 is similar to Fig. 3 but with a different set of channel and code parameters. The simulated codes are $(12, 6)$ RS, $(1/2, 11)$ RLC, $(1/2, 11)$ MS, $(1/2, 11, 9)$ ERLC and $(1/2, 9, 2)$ DD codes. Similar to Fig. 3a, the DD code starts at a low loss rate and increases slowly as $\epsilon$ increases. At the same delay, its ability to partially recover from long bursts allows it to outperform all other codes for all the MBL range in Fig 4b. It constituent repetition code can recover some of the lost packets when bursts are close to each other and hence the low loss rate in Fig. 4c.

### B. Delay Adaptability

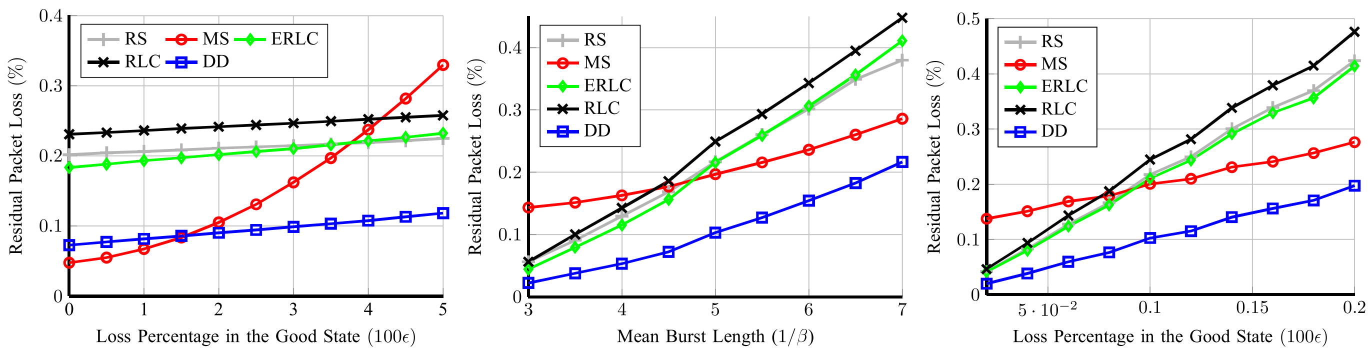
Fig. 5 illustrates the simulation results over a Gilbert-Elliott channel at $\alpha = 0.01$. Fig. 5a, 5b and 5c correspond to three different $(\beta, \epsilon)$ pairs, namely, $(2/3, 0.025)$, $(1/2, 0.015)$ and $(2/5, 0.005)$, respectively. The figure compares different codes in terms of fraction of recovered packets out of all losses with respect to varying recovery delay threshold $T$. As expected, a larger fraction of lost packets can be recovered as the maximum recovery delay is increased. In particular, if the channel introduces isolated losses, the DD code can recover the largest fraction of lost packets among all other codes within a delay of $T = 1$. On the other hand, when the channel introduces burst losses, increasing the allowable delay to $T = 5$ allows the DD code to recover more than $90\%$ of the lost packets. RLC and RS codes achieve a smooth performance improvement while increasing the maximum delay $T$. However, these codes fail to recover when the channel introduces long bursts as shown in Fig. 5c where the average burst length is $1/\beta = 3$ packets. Moreover, their fraction of lost packets recovered at $T = 1$ and $T = 2$ is less than that of the DD code. MS and ERLC codes achieve a comparable performance to that of the DD code at $T = 5$ but both codes cannot adapt their decoding delay. The MS code fail to recover any lost packet with a delay shorter than $T = 5$ because of the shifted repetition code used. Similarly, ERLC codes with a shift of $\Delta = 4$ can only start decoding with a delay of $T = 4$ packets.

From Fig. 5a to 5b to 5c, the i.i.d. loss rate $\epsilon$ is decreasing and the mean burst length $1/\beta$ is increasing while fixing $\alpha$,




(a) $\alpha = 0.01, \beta = 1/2, \epsilon \in [0, 0.05]$ (b) $\alpha = 0.01, \beta \in [1/3, 4/5], \epsilon = 0.025$ (c) $\alpha \in [0.002, 0.02], \beta = 1/2, \epsilon = 0.025$

Fig. 3: Simulations results over a set of Gilbert Elliott channel realizations. The residual loss rates of different codes at $R = 1/2$ and $T = 5$ are plotted vs. $\epsilon$ in Fig. 3a, MBL $(1/\beta)$ in Fig. 3b and $\alpha$ in Fig. 3c.




(a) $\alpha = 0.001, \beta = 1/5, \epsilon \in [0, 0.05]$ (b) $\alpha = 0.001, \beta \in [1/7, 1/3], \epsilon = 0.035$ (c) $\alpha \in [0.0002, 0.002], \beta = 1/5, \epsilon = 0.035$

Fig. 4: Simulations results over a set of Gilbert Elliott channel realizations. The residual loss rates of different codes at $R = 1/2$ and $T = 11$ are plotted vs. $\epsilon$ in Fig. 4a, MBL $(1/\beta)$ in Fig. 4b and $\alpha$ in Fig. 4c.

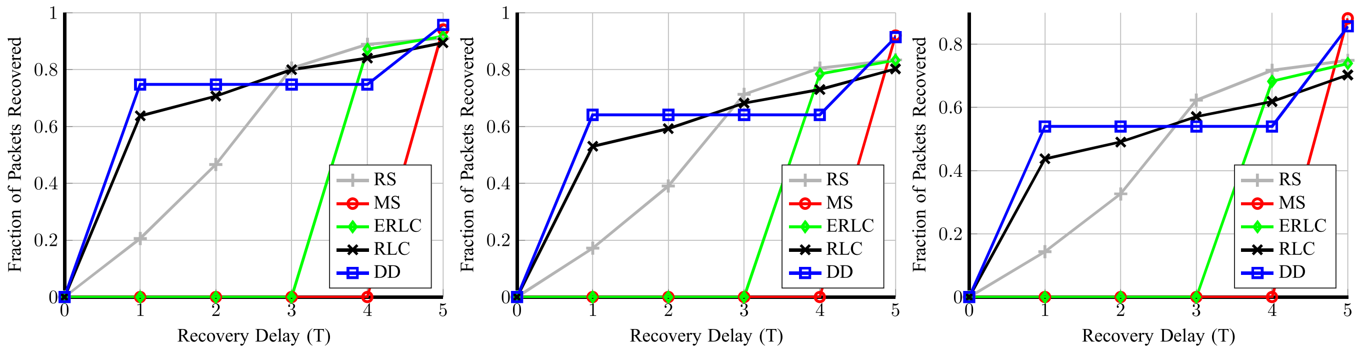
i.e., the channel is getting more bursty. Hence, the fraction of recoverable packets with a short delay is decreasing (75%, 65% and 55% respectively) since recovering burst losses require a longer delay. The DD recovers more than 90% of the packets in all three cases by adapting its delay. In other words, if the channel is bursty, adapting the adaptive playback buffer to accommodate packets recovered within a longer delay can boost the number of recovered packets.

We use the same set of codes in Fig. 4 in Fig. 6 to study their performance while changing the maximum delay $T$. We set $\alpha = 0.001$ and use $(\beta, \epsilon) = (1/4, 0.035)$ in Fig. 6a, $(\beta, \epsilon) = (1/5, 0.025)$ in Fig. 6b and $(\beta, \epsilon) = (1/6, 0.015)$ in Fig. 6c. The maximum delay $T$ in the range 0 to 11 is plotted on the x-axis whereas the fraction of recovered packets is on the y-axis. The delay of $T = 11$ packets can correspond to a delay of 110 ms in a VoIP system that uses packets with 10 ms duration. Note that an RLC with a reduced memory is used in constructing the $(1/2, 11, 2)$ DD code to achieve $N = 2$, $T_1 = 3$, $B = 9$ and $T_2 = 11$ as suggested in Section IV-E.

Increasing the maximum delay $T$ allows for recovering a larger fraction of lost packets. The performance of RLC and RS codes improves smoothly with $T$, however, they are slightly slower than DD codes due to the ability of the latter to recover the last two packets in any long burst within a delay of 3 packets (see Section III-B). Moreover, the DD codes outperform all other codes at the delay of $T = 11$ packets specially in the case of large mean burst length $1/\beta = 6$ in Fig. 6c. This is similar to the conclusion made in Fig. 5. As the i.i.d. loss rate $\epsilon$ decreases and the mean burst length $1/\beta$ increases, a larger fraction of lost packets is recovered with a longer delay. For example, more than 30% of the lost packets are recovered at $T = 11$ using the DD code in Fig. 6c.

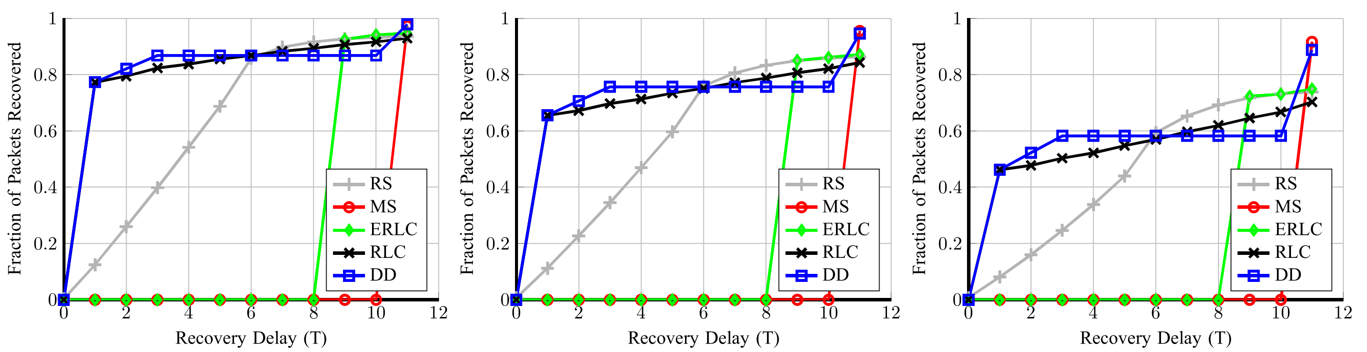
## VII. Experimental Results

In this section, we study the performance of different FEC codes in Section V over a set of real world packet traces [27]. These traces consist of around 150 million packets collected over a wireless sensor network while varying 32 different parameters, e.g., packet inter-arrival time, payload-size, distance between nodes. We consider all packets with inter-arrival time equal to 20 ms which is used by many VoIP applications. These are a total of 18.75 million packets with loss rate 8.3%. We parse the trace files and generate a vector of ones and zeros that correspond to lost and received packets respectively. We divide the trace into non-overlapping windows of length 15000 packets, each corresponding to 5 minutes of audio. The FEC codes in Section V are then simulated over each window.




(a) $\alpha = 0.01, \beta = 1/2, \epsilon = 0.03$ (b) $\alpha = 0.01, \beta = 2/5, \epsilon = 0.025$ (c) $\alpha = 0.01, \beta = 1/3, \epsilon = 0.02$

Fig. 5: Simulation results over Gilbert Elliot channels with $\alpha = 0.01$ and different $(\beta, \epsilon)$ pairs. For each pair, the y-axis shows the fraction of erased packets recovered within a delay of $T$ packets on the x-axis for different codes.




(a) $\alpha = 0.001, \beta = 1/4, \epsilon = 0.005$ (b) $\alpha = 0.001, \beta = 1/5, \epsilon = 0.01$ (c) $\alpha = 0.001, \beta = 1/6, \epsilon = 0.015$

Fig. 6: Simulation results over Gilbert Elliot channels with $\alpha = 0.01$ and different $(\beta, \epsilon)$ pairs. For each pair, the the fraction of erased packets recovered vs. the recovery delay in packets $T$ is plotted for different codes.

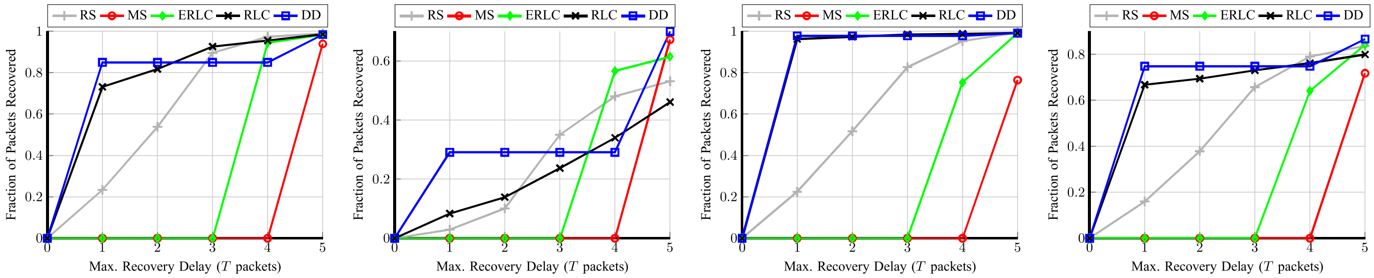
We use the transmission rating factor $R$ of the ITU-T G.107 E-model [28] to compare the performance of these codes. The $R$-factor is a single metric which predicts the subjective quality experienced by an average listener by combining different effects such as, signal-to-noise ratio, equipment impairments, echo, delay, packet loss rate, burstiness of the losses, etc. According to [28], the $R$-factor lies in the range from 0 to 100 and can be interpreted in terms of user satisfaction. A value less than 60 is considered unacceptable whereas values above 90 are very good. Values in the range $[60, 70]$, $[70, 80]$ and $[80, 90]$ can be interpreted as “Many Users Dissatisfied”, “Some Users Dissatisfied” and “Satisfied”, respectively.

Using the residual loss pattern of each FEC in every window, we compute the packet loss rate $Ppl$ and the burst factor $BurstR$ to compute the $R$-factor, $R(Ppl, BurstR, D)$, where $D$ is the delay in milliseconds and is computed as follows. We let the total network delay between the transmitter and the receiver be $t_n = 50$ ms and hence, $D = t_n + T \cdot t_s$, where $t_s = 20$ ms is the packet duration and $T$ is the maximum allowed FEC recovery delay in packets. We note that the default values for all other parameters in [28, Table 3] are used while computing the $R$-factor.

Fig. 7 and 8 illustrate the fraction of packets recovered and the $R$-factor for every code vs. the maximum recoverable delay $T$ in packets. This implicitly assumes that a playback buffer with a fixed delay $T$ is used. However, the figure studies how the buffer should be adapted depending on the channel conditions. In other words, sometimes allowing for a longer delay helps as most the lost packets are recovered with a long delay. On the other hand, when most the packets are recovered with a short delay, increasing the allowed delay might affect the interactivity of the application.

Since different FEC codes are optimized for different erasure patterns, we divide the 1250 windows of the trace into 4 different categories and compare the performance of these codes in each category. The 4 categories are:

1) **Mild Windows:** This category contains 363 windows with short bursts of length $B = 3$ or less and a small fraction of isolated losses, $< 1\%$. RLC and DD codes recover more than $75\%$ of the lost packets within a delay of a single packet. A longer delay $T = 5$ allows all codes to recover $\geq 95\%$ as shown in Fig. 7a. However, Fig. 8a shows that the fraction of packets recovered with a delay of $T = 1$ is indeed enough to get a $R$-factor above 90 (very good quality). Increasing the delay can help recover a larger fraction of the lost packets but reduces the value of the $R$-factor in the case of mild windows as this extra delay affects the interactivity of the application.



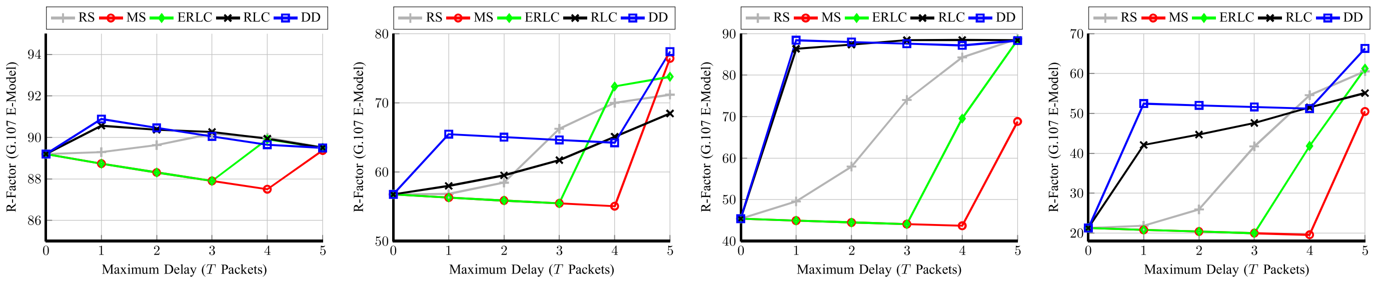

(a) Mild Windows: Isolated Loss $<$ 1%, Burst Length $\leq 3$

(b) Bursty Windows: Isolated Loss $<$ 1%, Burst Length $\in [4, 20]$

(c) Isolated Windows: Isolated Loss $\geq$ 1%, Burst Length $\leq 3$

(d) Mixed Windows: Isolated Loss $\geq$ 1%, Burst Length $\in [4, 20]$

Fig. 7: The fraction of packets recovered vs. the recovery delay $T$ in all four categories of the the packet traces in Section VII.




(a) Mild Windows: Isolated Loss $<$ 1%, Burst Length $\leq 3$

(b) Bursty Windows: Isolated Loss $<$ 1%, Burst Length $\in [4, 20]$

(c) Isolated Windows: Isolated Loss $\geq$ 1%, Burst Length $\leq 3$

(d) Mixed Windows: Isolated Loss $\geq$ 1%, Burst Length $\in [4, 20]$

Fig. 8: The $R$-factor achieved by different codes vs. delay $T$ in all four categories of the the packet traces in Section VII.

2) **Bursty Windows:** This category consists of 307 windows with maximum bursts lengths $B \in [4, 20]$ but small number of isolated losses $< 1\%$. The MS code designed for burst erasure channels recovers around 65% of the lost packets but all at $T = 5$ as shown in Fig. 7b. The DD code achieves a slightly better performance at $T = 5$. More importantly, the DD recovers more than 30% of the lost packets within a delay of $T = 1$ packet. This is more than all other simulated codes including RLC code. This results in being the only code with $R$-factor in the acceptable range ($\geq 60$) at $T = 1$ and a rating close to 80 at $T = 5$ as shown in Fig. 8b. Since the channel introduces many burst losses, allowing for a longer delay can significantly increase the performance specially when using the DD code. It is worth noting that the RLC code is the worse at $T = 5$ for windows in this category as it can only recover from bursts of length 3 or less.

3) **Isolated Windows:** This category consists of 63 windows with short bursts $B \leq 3$ but a large number of isolated losses $\geq 1\%$. As shown in Fig. 7c, the RLC code designed for isolated erasures recovers from more than 95% of the losses with a delay of 1 packet and the percentage increases to around 98% at $T = 5$. Interestingly, the proposed DD codes achieves a similar performance. This is translated to a large gain in terms of $R$-factor as shown in Fig. 8c. Allowing for a single delay increases the $R$-factor from 45 to around 87, i.e., from "unacceptable" to "satisfied". Extra delay is not needed in this category of windows as they do not include long bursts.

4) **Mixed Windows:** This category contains 339 windows with long bursts $B \in [4, 20]$ and large fraction of isolated losses $\geq 1\%$. Again, RLC and DD are the only codes which recovers more than 60% of the lost packets within a single packet of delay. The latter achieves a slightly higher percentage as shown in Fig. 7d. Codes with a competitive performance to DD codes at $T = 5$ are RS and ERLC codes. However, these codes do not provide satisfactory performance at shorter delays. With high loss rates and long bursts, the $R$-factor is always lower than 70. In such challenging windows, Fig. 8d suggests allowing for a long delay to achieve ratings outside the "unacceptable" range.

The main conclusion from Fig. 7 can be explained as follows. Every FEC code is designed for a set of erasure patterns and delay distribution. The MS code is designed for channels with long bursts but requires a long delay even when recovering a single erasure. RLC and RS codes recover quickly from isolated losses but fail to recover from long bursts. ERLC codes can recover from both patterns, isolated losses and burst losses. However, these codes still require a delay of at least $T = 4$ packets. The proposed DD codes achieves fast recovery in case of isolated losses without compromising its burst erasure correction capability. This allows it to achieve the best performance – in terms of fraction of recovered packets, fast recovery of isolated losses, and $R$-factor – among all other codes in all four categories.

Using the same set of codes in Sections VI and VII helps building a connection between the simulation results over Gilbert-Elliott channels and that over real-packet traces. In Fig. 5, it was shown that as the mean burst length $1/\beta$ increases, larger fraction of erased packets are decoded with a longer delay. A similar conclusion can be drawn by looking at the simulation results over the real packet traces in Fig. 7. Moreover, when the channel introduces short bursts, RLC codes are the main competitors with the DD code (cf. Fig. 3b). However, as the mean burst length increases the performance of the RLC codes deteriorate and the MS code becomes the only competitor. The same conclusion can be made by comparing Fig. 7b and 7c.

## VIII. Conclusions

We introduce a new class of FEC, dual-delay (DD) codes, for low-delay streaming applications such as VoIP. When recovering from long burst losses, our codes outperform baseline schemes such as random linear codes and Reed-Solomon codes. When recovering from isolated loss patterns, our codes provide considerably faster recovery than previously proposed streaming codes. We propose a general construction for DD codes and characterize the performance trade-off between burst-loss correction and isolated-loss correction, as well as the delays associated with each of these patterns. Using simulations over statistical channels, we demonstrate that DD codes not only achieve significant performance gains over previously proposed codes in the overall packet loss rate, but also recover a large fraction of packets with a considerably shorter delays. Finally by using real-world packet loss traces, and the E-model performance metric recommended in ITU-T G.107, we demonstrate that DD codes can be a promising candidate for interactive streaming applications such as VoIP.

Our proposed DD codes provide a rich design space that can be valuable in many interactive streaming applications. For example, many practical VoIP systems use adaptive playback buffers where the playback deadline can be adjusted. Using the delay adaptivity of DD codes, the deadline can be kept low to improve interactivity when there are isolated losses over the channel. In the presence of longer bursts, the deadline can be increased to improve error resiliency. A thorough investigation of such systems appears to be a promising avenue of future work. On the theoretical side, DD codes require large column distance in smaller windows for isolated loss correction and larger column span in longer windows for burst loss correction. Studying fundamental trade-offs underlying these metrics along the lines of [20] is also a promising direction for future work.

## References

[1] W. Jiang and H. Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *PROCEEDINGS OF NOSSDAV '2000*, 2000.

[2] M. Manousos, S. Apostolacos, I. Grammatikakis, D. Mexis, D. Kagklis, and E. Sykas. Voice-quality monitoring and control for VoIP. *IEEE Internet Computing*, 9(4):35–42, July 2005.

[3] A. F. Ribadeneira. *An Analysis of the MOS under Conditions of Delay, Jitter and Packet Loss and an Analysis of the Impact of Introducing Piggybacking and Reed Solomon FEC for VoIP*. PhD thesis, 2007.

[4] X. Chen, C. Wang, D. Xuan, Z. Li, Y. Min, and W. Zhao. Survey on QoS management of VoIP. In *Computer Networks and Mobile Computing, 2003. ICCNMC 2003. 2003 International Conference on*, pages 69–77, Oct 2003.

[5] J. C. Bolot, S. Fosse-Parisis, and D. Towsley. Adaptive FEC-based error control for Internet telephony. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1453–1460 vol.3, Mar 1999.

[6] C. Perkins, O. Hodson, and V. Hardman. A survey of packet loss recovery techniques for streaming audio. *IEEE Network*, 12(5):40–48, Sept 1998.

[7] A. Lapidoth. The performance of convolutional codes on the block erasure channel using various finite interleaving techniques. *IEEE Transactions on Information Theory*, 40(5):1459–1473, Sep 1994.

[8] C. Padhye, K. J. Christensen, and W. Moreno. A new adaptive fec loss control algorithm for voice over ip applications. In *Performance, Computing, and Communications Conference, 2000. IPCCC '00. Conference Proceeding of the IEEE International*, pages 307–313, Feb 2000.

[9] M. Podolsky, C. Romer, and S. McCanne. Simulation of fec-based error control for packet audio on the internet. In *INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 505–515 vol.2, Mar 1998.

[10] T. Huang, P. Huang, K. Chen, and P. Wang. Could Skype be more satisfying? a QoE-centric study of the FEC mechanism in an internet-scale VoIP system. *IEEE Network*, 24(2):42–48, 2010.

[11] E. Soljanin. Some coding and information theoretic problems in contemporary (video) content delivery. In *Information Theory Workshop (ITW), 2013 IEEE*, pages 1–5, Sept 2013.

[12] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006.

[13] E. Martinian and C. W. Sundberg. Burst Erasure Correction Codes with Low Decoding Delay. *IEEE Transactions on Information Theory*, 50(10):2494–2502, 2004.

[14] E. Martinian and M. Trott. Delay-optimal burst erasure code construction. In *Proc. International Symposium on Information Theory (ISIT)*, Nice, France, July 2007.

[15] D. Leong and T. Ho. Erasure coding for real-time streaming. In *Proc. International Symposium on Information Theory (ISIT)*, 2012.

[16] N. Adler and Y. Cassuto. Low-delay erasure-correcting codes with optimal average delay. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2613–2617, June 2015.

[17] Y. Wei, Y. Mao, S. Leng, and T. Ho. Priority-Based Real-Time Stream Coding under Burst Erasures. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, Dec 2015.

[18] A. Badr, A. Khisti, W. Tan, and J. Apostolopoulos. Streaming Codes for Channels with Burst and Isolated Erasures. In *Proc. International Conference on Computer Communications (INFOCOM)*, 2013.

[19] A. Badr, A. Khisti, W. Tan, and J. Apostolopoulos. Robust Streaming Erasure Codes based on Deterministic Channel Approximations. In *Proc. International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, 2013.

[20] A. Badr, P. Patil, A. Khisti, W. Tan, and J. Apostolopoulos. Layered Constructions for Low-Delay Streaming Codes. *CoRR*, abs/1308.3827, 2013.

[21] G. Joshi. On playback delay in streaming communication. Master's thesis, Massachusetts Institute of Technology (MIT), 2012.

[22] C. Li, S. Yang, and R. W. Yeung. Multi-rate sequential data transmission. In *Proc. IEEE Information Theory Workshop (ITW)*, 2013.

[23] A. Badr, A. Khisti, and E. Martinian. Diversity Embedded Streaming erasure Codes (DE-SCo): Constructions and Optimality. *IEEE Journal on Selected Areas in Communications (JSAC)*, 29(5):1042–1054, May 2011.

[24] A. Badr, D. Lui, and A. Khisti. Streaming codes for multicast over burst erasure channels. *IEEE Transactions on Information Theory*, 61(8):4181–4208, Aug 2015.

[25] A. Badr, R. Mahmood, and A. Khisti. Embedded mds codes for multicast streaming. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2276–2280, June 2015.

[26] A. Badr, A. Khisti, W. Tan, and J. Apostolopoulos. Streaming Codes with Partial Recovery over Channels with Burst and Isolated Erasures. *submitted to IEEE Journal of Selected Topics in Signal Processing (JSTSP)*, 2014.

[27] S. Fu and Y. Zhang. CRAWDAD dataset due/packet-delivery (v. 2015-04-01). Downloaded from http://crawdad.org/due/packet-delivery/20150401/delay, April 2015. traceset: delay.

[28] ITU-T. *The E-model: a computational model for use in transmission planning*. ITU-T Recommendation G.107, June 2015.