CRC-Aided High-Rate Convolutional Codes With Short Blocklengths for List Decoding

Wenhui Sui, *Student Member, IEEE,* Brendan Towell, *Student Member, IEEE,* Ava Asmani, *Student Member, IEEE,* Hengjie Yang, *Member, IEEE,* Holden Grissett, *Student Member, IEEE,* and Richard D. Wesel, *Fellow, IEEE*.

***Abstract*—Recently, rate-$1/n$ zero-terminated (ZT) and tail-biting (TB) convolutional codes (CCs) with cyclic redundancy check (CRC)-aided list decoding have been shown to closely approach the random-coding union (RCU) bound for short blocklengths. This paper designs CRC polynomials for rate-$(n-1)/n$ ZT and TB CCs with short blocklengths. This paper considers both standard rate-$(n-1)/n$ CC polynomials and rate-$(n-1)/n$ designs resulting from puncturing a rate-$1/2$ code. The CRC polynomials are chosen to maximize the minimum distance $d_{\min}$ and minimize the number of nearest neighbors $A_{d_{\min}}$. For the standard rate-$(n-1)/n$ codes, utilization of the dual trellis proposed by Yamada *et al.* lowers the complexity of CRC-aided serial list Viterbi decoding (SLVD). CRC-aided SLVD of the TBCCs closely approaches the RCU bound at a blocklength of 128. This paper compares the FER performance (gap to the RCU bound) and complexity of the CRC-aided standard and punctured ZTCCs and TBCCs. This paper also explores the complexity-performance trade-off for three TBCC decoders: a single-trellis approach, a multi-trellis approach, and a modified single-trellis approach with pre-processing using the wrap around Viterbi algorithm.**

I. Introduction

The structure of concatenating a convolutional code (CC) with a cyclic redundancy check (CRC) code has been a popular paradigm since 1994 when it was proposed in the context of hybrid automatic repeat request (ARQ) [2]. It was subsequently adopted in the cellular communication standards of both 3G [3] and 4G LTE [4]. In general, the CRC code serves as an outer error-detecting code that verifies if a codeword has been correctly received, whereas the CC serves as an inner error-correcting code to combat channel errors.

Recently, there has been a renewed interest in designing powerful short blocklength codes. This renewed interest is mainly driven by the development of finite blocklength information theory by Polyanskiy *et al.*, [5] and the stringent requirement of ultra-reliable low-latency communication (URLLC) for mission-critical IoT (Internet of Things) services [6]. In [5], Polyanskiy *et al.* developed a new achievability bound known as the random-coding union (RCU) bound and a new converse bound, known as the meta-converse (MC) bound. Together, these two bounds characterize the error probability range for the best short blocklength code of length N with M codewords. The URLLC for mission-critical IoT requires that the time-to-transmit latency is within 500 μs while maintaining a block error rate no greater than 10^{-5}.

Several short blocklength code designs have been proposed in the literature. Important examples include the tail-biting (TB) convolutional codes decoded using the wrap around Viterbi algorithm (WAVA) [7], extended BCH codes under ordered statistics decoding [8], [9], non-binary low-density parity-check (LDPC) codes [10], non-binary turbo codes [11], and polar codes under CRC-aided successive-cancellation list decoding [12]. Recent advances also include the polarization adjusted convolutional codes proposed by Arıkan [13]. As a comprehensive overview, Coşkun *et al.* [8] surveyed most of the contemporary short blocklength code designs in the recent decade. We refer the reader to [8] for additional information.

In [14], Yang *et al.* proposed CRC-aided CCs as a powerful short blocklength code for binary-input (BI) additive white Gaussian noise (AWGN) channels. The convolutional encoder of interest has rate-$1/n$ and is either zero-terminated (ZT) or TB. Yang *et al.* seek to design a *distance-spectrum optimal* (DSO) CRC polynomial that minimizes frame error rate (FER) at a specified signal-to-noise ratio. For high SNR this is often equivalent to selecting the CRC polynomial that maximizes the minimum distance and minimizes the number of nearest neighbors for the resulting concatenated code. This design follows the approach of Lou *et. al.* [15] for designing CRC polynomials matched to the CC for improved error detection.

The presence of a CRC facilitates the use of the serial list Viterbi decoding (SLVD), an efficient algorithm originally proposed by Seshadri and Sundberg [16]. Yang *et al.* [14] showed that the expected list rank of SLVD of the CRC-aided CC is small at high SNR, thus achieving a low average decoding complexity at operating points of common interest. Yang *et al.* [14] demonstrated that these concatenated codes can approach the RCU bound. In [17], [18], Schiavone extended this line of work by looking at the parallel list Viterbi decoding and applying this perspective to legacy coding. In our precursor conference paper [1], this framework is extended to rate-$(n-1)/n$ CCs and the resulting concatenated code is able to approach the RCU bound with a low decoding complexity as well.

This research was supported by the National Science Foundation (NSF) under Grant CCF-2008918. An earlier version of this paper was presented in part at the 2022 IEEE International Conference on Communications (ICC) [1].

Wenhui Sui, Brendan Towell, Ava Asmani, Holden Grissett, and Richard D. Wesel are with the Department of Electrical and Computer Engineering, University of California at Los Angeles, Los Angeles, CA, 90095 USA (e-mail: wenhui.sui@ucla.edu; brendan.towell@ucla.edu; ava24@ucla.edu;holdengs@ucla.edu; wesel@ucla.edu).

Hengjie Yang is with Qualcomm Technologies, Inc., San Diego, CA 92121, USA (e-mail: hengjie.yang@ucla.edu).

A. Contributions

This paper presents designs of CRC-aided ZT and TB CCs for rate-$(n-1)/n$ CCs at short blocklengths for the BI-AWGN channel. Each CRC polynomial is selected to maximize the minimum distance and minimizes the number of nearest neighbors for the resulting concatenated code. If SLVD is performed with a sufficiently large maximum list size such that finding a CRC-passing codeword is guaranteed, then SLVD is an implementation of maximum-likelihood (ML) decoding for these concatenated codes.

We consider standard, systematic, rate-$(n-1)/n$ feedback convolutional encoders [19], as well as encoders obtained by puncturing the output of a rate-$1/2$ convolutional encoder, as proposed by Cain *et al.* [20]. The resulting concatenated codes are called a standard or punctured CRC-ZTCC or CRC-TBCC, respectively.

For the standard CCs, SLVD on the primal trellis requires high decoding complexity due to the 2^{n-1} outgoing branches at each node. SLVD implementation becomes exponentially more complicated when there are more than two outgoing branches per state. In order to simplify SLVD implementation and reduce complexity, we utilize the *dual trellis* pioneered by Yamada *et al*. [21]. The dual trellis expands the length of the primal trellis by a factor of n, while reducing the number of outgoing branches at each node from 2^{n-1} to at most two.

For the punctured codes, Cain *et al.* [20] were the first to propose obtaining high-rate CCs through puncturing a rate-$1/2$ code to allow the use of a simple primal trellis for decoding. In [22], Haccoun and Bégin presented optimal encoder generators of the original rate-$1/2$ codes and the corresponding puncturing matrices for the rate-$(n-1)/n$ punctured codes. A list-decoding sieve [23] is used to identify the CRC polynomials that maximize the minimum distance and minimize the number of nearest neighbors for the Haccoun and Bégin punctured codes.

For ZTCCs, a work related to this line of research is that of Karimzadeh and Vu [24]. They considered designing the optimal CRC polynomial for multi-input ZTCCs. In their framework, the information sequence is first divided into $(n-1)$ streams, one for each input rail, and they aim at designing an optimal CRC polynomial for each rail. In contrast, this paper encodes the information sequence with a single CRC polynomial and is then divided into $(n-1)$ streams for the standard rate-$(n-1)/n$ encoder. Simulation results indicate the new approach improves FER performance at high SNRs.

For TBCCs, simulations show that for both the standard and punctured rate-$(n-1)/n$ TBCCs with blocklength $N=128$, the FER performance of our CRC-aided TBCCs approaches the RCU bound as the degree m of the CRC polynomial increases.

This paper considers three architectures to enforce the TB condition for CRC-TBCCs. One approach performs list decoding on a single trellis that allows all initial states. As the list decoder identifies possible trellis paths, non-tail-biting paths are rejected. At low SNR, SLVD on the single trellis requires a large list size to identify the ML TB codeword, with a majority of trellis paths not satisfying the TB condition.

An alternative is a multi-trellis approach that initializes multiple copies of the dual trellis, one for each possible starting and ending state pair. The multi-trellis approach requires a much smaller list size because only TB paths are added to the list. This approach avoids the decoding time complexity of potentially exploring a large list at the cost of the computational and space complexity of creating and storing multiple trellises. It can provide a benefit over the single-trellis approach in low-SNR environments.

A third approach to TB decoding that can be applied in the context of the single-trellis list decoder is the wrap-around Viterbi algorithm (WAVA). Introduced in [25], WAVA is a near-ML decoding algorithm for TBCCs. To achieve the balance of decoding time and space efficiency, this paper combines the wrap-around behavior of WAVA with SLVD for TBCCs. The decoding process is completed in two steps: the WAVA step with at most 2 trellis iterations, and the list decoding step with a sufficiently large list size such that there are no negative acknowledgment signals. Simulation results demonstrate that this decoding method reduces the average list size as compared with the single-trellis decoder without WAVA, but this reduction comes at a cost of degraded FER performance.

This paper culminates by providing plots that show the trade-off between decoder complexity and FER performance computed as the gap from the RCU bound at FER 10^{-4} and FER vs. SNR results that show both punctured and standard TBCCs performing within 0.1 dB of the RCU bound with a degree-10 CRC polynomial designed to maximize the minimum distance and minimize the number of nearest neighbors.

B. Organization

The remainder of this paper is organized as follows. Sec. II reviews systematic encoding with feedback for $(n, n-1, v)$ convolutional codes and describes the dual trellis construction. Sec. III introduces various serial list decoders for rate-$(n-1)/n$ CCs. Sec. IV presents optimal CRC polynomial designs for standard and punctured high-rate codes using a trellis event enumeration method and a sieve method. Sec. V analyzes the trade-off between complexity and decoding performance of these designs. Sec. VI presents simulation results comparing FER performance of our designs to the RCU bound and to the Karimzadeh and Vu approach. Finally, Sec. VII concludes the paper.

II. The Dual Trellis

This section describes systematic encoding for $(n, n-1, v)$ convolutional codes and introduces the dual trellis proposed by Yamada *et al.* [21] for high-rate CCs generated with an $(n, n-1, v)$ convolutional encoder, where v is the number of memory elements in the encoder. This section also discusses the tree-trellis algorithm for list decoding and its benefits.

Let K and N denote the information length and blocklength in bits. Let $R=K/N$ denote the rate of the CRC-aided CC. A degree-m CRC polynomial is of the form $p(x)=1+p_1x+\cdots+p_{m-1}x^{m-1}+x^m$, where $p_i\in\{0,1\}$, $i=1,2,\ldots,m-1$.

For brevity, a CRC polynomial is represented in hexadecimal where its binary coefficients are written from the highest to lowest order. For instance, 0xD represents $x^3 + x^2 + 1$. The codewords are BPSK modulated. The SNR is defined as $\gamma_s \triangleq 10\log_{10}(A^2)$ (dB), where A represents the BPSK amplitude and the noise is distributed as a standard normal. This SNR can also be written as $2\frac{E_c}{N_o} = 2R\frac{E_b}{N_o}$, where E_c is the channel symbol energy, E_b is the bit energy, and $\frac{N_o}{2}$ is the two-sided power spectral density of noise.

A. Systematic Encoding

We briefly follow [19, Chapter 11, p. 482] in describing a systematic $(n, n-1, v)$ convolutional encoder with reverse input and output labelling[1]. A systematic $(n, n-1, v)$ convolutional encoder can be represented by its parity check matrix

$$H(D) = [h^{(n-1)}(D), h^{(n-2)}(D), \ldots, h^{(0)}(D)], \quad (1)$$

where each $h^{(i)}(D)$ is a polynomial of degree up to v in delay element D associated with the i-th code stream, i.e.,

$$h^{(i)}(D) = h_v^{(i)} D^v + h_{v-1}^{(i)} D^{v-1} + \cdots + h_0^{(i)}, \quad (2)$$

where $h_j^{(i)} \in \{0, 1\}$. For convenience, we represent each $h^{(i)}(D)$ in octal form. For instance, $H(D) = [D^3 + D^2 + D + 1, D^3 + D^2 + 1, D^3 + D + 1]$ can be concisely written as $H = (17, 15, 13)$. Let $\boldsymbol{h}^{(i)} \triangleq [h_v^{(i)}, h_{v-1}^{(i)}, \ldots, h_0^{(i)}]$, $i = 0, 1, \ldots, n-1$. Since the 0th code stream corresponds to the parity check bit and the ith code stream corresponds to the ith input bit for $1 \le i \le n-1$, the systematic encoding matrix $G(D)$ associated with $H(D)$ is thus given by

$$G(D) = \begin{bmatrix} \frac{h^{(1)}(D)}{h^{(0)}(D)} & 1 & 0 & \cdots & 0 \\ \frac{h^{(2)}(D)}{h^{(0)}(D)} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{h^{(n-1)}(D)}{h^{(0)}(D)} & 0 & 0 & \cdots & 1 \end{bmatrix}. \quad (3)$$

To satisfy the TB condition for the feedback encoder, we use a similar procedure as that for turbo codes [26]. First, we use a pre-encoding operation to encode from the all-zero state and obtain a final state. Depending on that final state, an initial state is selected and the message re-encoded from this state satisfies the TB condition.

B. Constructing the Dual Trellis

The primal trellis associated with a rate-$(n-1)/n$ ZTCC has 2^{n-1} outgoing branches per state. Performing SLVD over the primal trellis when $n > 2$ is highly complex. In [14], the low decoding complexity of SLVD for rate-$1/n$ convolutional codes relies on the fact that each state only has 2 outgoing branches. In order to efficiently perform SLVD, we consider the dual trellis proposed by Yamada *et al.* [21].

We briefly explain the dual trellis construction for parity check matrix

[1]The reverse input (resp. output) labeling means that the labels from top to bottom of the input (resp. output) streams are arranged in decreasing order.

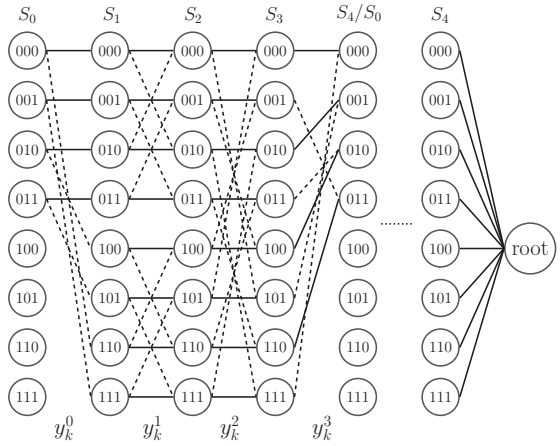

Fig. 1. Dual trellis diagram for rate-$3/4$ TBCC with a root node at the end for encoder $H = (7, 5, 2, 6)$ with $v = 2$. Solid lines represent 0 paths and dashed lines represent 1 paths.

$H(D) = [h^{(n-1)}(D), h^{(n-2)}(D), \ldots, h^{(0)}(D)]$. First, we define the maximum instant response order λ as

$$\lambda \triangleq \max\{j \in \{0, 1, \ldots, n-1\} : h_0^{(j)} = 1\}. \quad (4)$$

The state of the dual trellis is represented by the partial sums of $(v+1)$ adders in the observer canonical form of $H(D)$. At time index j, $j = 0, 1, \ldots, n-1$, the state is given by

$$\boldsymbol{s}^{(j)} = [s_v^{(j)}, s_{v-1}^{(j)}, \ldots, s_0^{(j)}]. \quad (5)$$

Next, we show how the state $\boldsymbol{s}^{(j)}$ evolves in terms of the output bits $\boldsymbol{y}_k = [y_k^{(0)}, y_k^{(1)}, \ldots, y_k^{(n-1)}]$, $k = 1, 2, \ldots, N/n$, so that a dual trellis can be established.

Dual trellis construction for $\boldsymbol{y}_k = [y_k^{(0)}, y_k^{(1)}, \ldots, y_k^{(n-1)}]$:

1) At time $j = 0$, $\boldsymbol{s}^{(0)} = [0, s_{v-1}^{(j)}, s_{v-2}^{(j)}, \ldots, s_0^{(j)}]$, where $s_i^{(0)} \in \{0, 1\}$. Namely, only 2^v states exist at $j = 0$.
2) At time j, $j < n-1$, draw branches from each state $\boldsymbol{s}^{(j)}$ to the states $\boldsymbol{s}^{(j+1)}$ that satisfy
$$\boldsymbol{s}^{(j+1)} = \boldsymbol{s}^{(j)} + y_k^{(j)} \boldsymbol{h}^{(j)}, \quad y_k^{(j)} \in \{0, 1\}. \quad (6)$$
3) At time $j = n-1$, draw branches from each state $\boldsymbol{s}^{(n-1)}$ to state $\boldsymbol{s}^{(n)}$ by
$$\boldsymbol{s}^{(n)} = \left(\boldsymbol{s}^{(n-1)} + y_k^{(n-1)} \boldsymbol{h}^{(n-1)}\right)^r, \; y_k^{(n-1)} \in \{0, 1\}, \quad (7)$$
where $(a_v, a_{v-1}, \ldots, a_1, a_0)^r = (0, a_v, a_{v-1}, \ldots, a_1)$.
4) For time $j = \lambda$, draw a branch from each state $\boldsymbol{s}^{(\lambda)}$ according to (6) only for $y_k^{(\lambda)} = s_0^{(\lambda)}$.

After repeating the above construction for each $\boldsymbol{y}_k$, $k = 1, 2, \ldots, N/n$, we obtain the dual trellis associated with the $(n, n-1, v)$ convolutional code. Since the primal trellis is of length N/n, whereas the dual trellis is of length N, the dual trellis can be thought of as expanding the primal trellis length by a factor of n, while reducing the number of outgoing branches per state from 2^{n-1} to less than or equal to 2. Fig. 1 illustrates the dual trellis structure for a rate-$3/4$ code with 2 memory elements.

For an $(n, n-1, v)$ CC, zero termination over the dual trellis requires at most $n\lceil v/(n-1)\rceil$ steps. In our implementation,

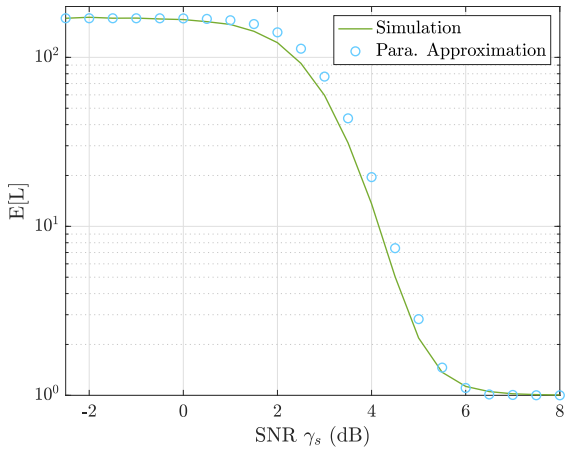

Fig. 2. Comparison of the parametric approximation [14] of the expected list rank $\mathsf{E}[L]$ with the simulated results for the $v = 4, m = 3$ CRC-TBCC generated by $H = (33, 25, 37, 31)$ with blocklength of 128. The optimal CRC used is 0x9.

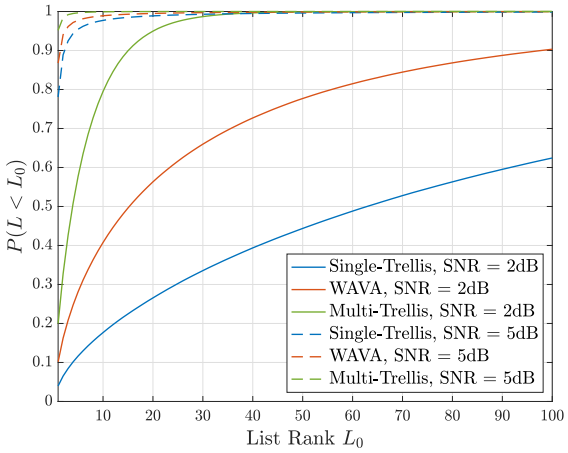

Fig. 3. Cumulative distribution function (CDF) of list ranks for the single-trellis, multi-trellis, and WAVA decoding approaches at SNR $\gamma_s = 2$ and 5 dB for the $(33, 25, 37, 31)$ TBCC with blocklength of 128.

a breadth-first search identifies the zero-termination input and output bit patterns that provide a trajectory from each possible state $\boldsymbol{s}$ to the zero state. The input and output bit patterns have lengths $(n-1)\lceil v/(n-1)\rceil$ and $n\lceil v/(n-1)\rceil$ respectively.

III. Serial List Viterbi Decoding for TBCCs

This section considers three SLVD methods that apply to both the standard and punctured high-rate TBCCs. SLVD enumerates possible paths through the trellis, starting from the lowest weight path, stopping once the first path that satisfies both the CRC and the TB condition is reached. Information about the previously investigated paths and path metrics is required to find the next optimal path.

To efficiently implement SLVD, we use the tree-trellis algorithm (TTA) [27], which maintains a sorted list of nodes indexed by path metric. These nodes either correspond to a previously unexplored ending state in the trellis or to a previously explored path and a detour. This approach allows the efficient determination of the next path to be explored if the current one does not satisfy both the CRC and TB conditions. In order to efficiently maintain the sorted list of nodes, this paper uses a Min Heap [28], which is easier to implement than the Red-Black tree [29] [30] and has the same $\mathcal{O}(\log \ell)$ time complexity, where ℓ is the number of elements in the heap. This simpler Min Heap implementation comes at the cost of an increased memory requirement compared to the Red-Black Tree since the heap has to maintain every element, as opposed to the best L elements for the tree.

A. Single Trellis Decoder

To adapt SLVD to handle the multiple terminating states that are possible with a TBCC, a root node is added as shown in Fig. 1. The root node connects to all terminating states of the trellis. The Hamming distance of the branch metric for the branch connecting any state to this root node is zero. This additional root node allows the trellis to end in a single state.

B. Multi-Trellis Decoder

Decoding on a single dual trellis (single-trellis approach) leads to complexity issues, because the decoder goes through a number of paths that pass neither the TB check nor CRC, resulting in high expected list ranks at low SNRs. To decode more efficiently, we propose a multi-trellis approach that includes only TB paths in the list.

The multi-trellis approach constructs 2^v trellises. Each multi-trellis follows the same structure as the original punctured or dual trellis, but with only one starting and ending state to enforce the TB condition. In a conceptually similar manner to how a root node was added to the dual trellis in Fig. 1, a root node is also added to the multi-trellis approach, but paths to the root node only come from the single ending state in each trellis that guarantees the TB condition.

Since all paths found using this approach will be TB, this significantly reduces the expected list size at low SNRs. However, at high SNRs, the multi-trellis approach has a substantially higher decoding complexity due to the additional upfront cost of constructing the dual trellises. In this case, the extra resources taken to initialize the multi-trellis approach bring down the overall decoder efficiency.

C. Wrap-Around Viterbi Algorithm Decoder

As the constraint length of a TBCC increases, the number of states grows exponentially. The multi-trellis approach becomes impractical due to both time and memory for constructing the trellises. Thus, we consider a non-ML single-trellis decoder that uses WAVA [25] to reduce the average list rank. In [31], the authors show that a WAVA-inspired parallel list Viterbi decoder achieves good performance with low complexity.

Fig. 2 extends the parametric approximation of the expected list rank shown in [14] to the single-trellis decoding of high-rate CRC-TBCCs. The approximation lines up well with the simulation results for a $v = 4$ TBCC with a degree-3 CRC. In addition, an initial examination of the list rank distributions for the three decoding schemes at SNR points of 2 and 5

dB is presented in Fig. 3. At a low SNR, the multi-trellis approach maintains an extremely small list rank compared to the other approaches. The WAVA approach also has a substantially larger probability of a small list rank compared to the single-trellis, but not as small as the multi-trellis. Although the ordering is preserved as SNR increases, additional pre-processing to reduce list rank is inconsequential since the list ranks of all three approaches are low. Considering the extra complexity of constructing the multi-trellis approach and the non-ML nature of the WAVA approach, we use the single-trellis SLVD for simulations. Different distributions lead to different $\mathsf{E}[I]$ and $\mathsf{E}[L]$ values when evaluating the decoding complexity, where $\mathsf{E}[I]$ is the expected number of insertions to maintain the sorted list of path metric differences.

The non-ML decoder with WAVA proceeds in two steps. In the first step, the algorithm initializes each state of a single dual or punctured trellis with all zero metrics. It then performs two iterations of add-compare-select (ACS) along the trellis. Each time the end of the trellis is encountered, the initial states of the trellis are initialized to the cumulative metrics in the final states. At the end of the first iteration, if the optimal path satisfies TB and CRC conditions, the algorithm outputs this path and stops decoding. In the second step, SLVD runs on the ending metrics of the second trellis iteration. This decoding algorithm improves the reliability of the final decision for the optimal traceback path and decreases the expected list rank while keeping the complexity low. The WAVA metrics are not ML and this algorithm has slightly worse decoding performance than the other two ML approaches.

IV. Optimal CRC Polynomial Design

This section presents two approaches for designing CRC polynomials that maximize the minimum distance and minimize the number of nearest neighbors: a trellis enumeration method extended from [14] and a list decoding sieve method proposed in [23]. As a case study, this paper mainly focuses on the rate-$3/4$ systematic feedback convolutional codes in [19, Table 12.1(e)] and the punctured rate-$3/4$ convolutional codes in [22].

A. CRC Polynomials for Standard Zero-Terminated Codes

In this paper, we focus on the low FER regime. Thus, the CRC polynomials identified in this paper simply maximize the minimum distance $d_{\min}$ of the concatenated code and minimizes the number of nearest neighbors. Examples in [14] indicate that CRC polynomials designed in this way can provide optimal or near-optimal performance for a wide range of SNRs.

We apply the CRC polynomial design algorithm in [14] to identify CRC polynomials for high-rate ZTCCs. The first step is to collect the irreducible error events (IEEs), which are ZT paths on the trellis that deviate from the zero state once and rejoin it once. IEEs with a very large output Hamming weight do not affect the choice of optimal CRC polynomials. In order to reduce the runtime of the CRC optimization algorithm, IEEs with output Hamming weight greater than or equal to a threshold $\tilde{d}$ are not considered. Dynamic programming constructs all ZT paths of length equal to N/n and output weight less than $\tilde{d}$. Finally, we use the resulting set of ZT paths to identify the degree-m optimal CRC polynomial for the rate-$(n-1)/n$ CC.

TABLE I
Optimal CRC Polynomials for Standard Rate-$3/4$ ZTCC at Blocklength $N = 128$ Generated by $H = (33, 25, 37, 31)$ With $v = 4$, by $H = (47, 73, 57, 75)$ With $v = 5$, and by $H = (107, 135, 133, 141)$ With $v = 6$

K	m	$v=4$			$v=5$			$v=6$		
		CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$
90	0	0x1	4	60	0x1	5	200	0x1	6	736
89	1	0x3	4	30	0x3	5	113	0x3	6	331
88	2	0x7	5	85	0x7	5	56	0x7	6	106
87	3	0x9	5	1	0x9	5	1	0xB	6	34
86	4	0x1B	6	251	0x15	6	54	0x1D	6	3
85	5	0x25	6	32	0x25	7	156	0x25	7	27
84	6	0x4D	7	155	0x7B	7	76	0x6F	7	1
83	7	0xF3	7	45	0xED	8	194	0x97	8	12
82	8	0x1E9	8	145	0x1B7	8	25	0x1B5	9	375
81	9	0x31B	8	27	0x3F1	8	1	0x2F1	9	65
80	10	0x5C9	9	168	0x66F	9	2	0x59F	10	490
79	11	0xC2B	10	1015	0xE8D	10	293	0xD2D	10	42

In [14], Yang *et al.* provided a useful result for selecting threshold $\tilde{d}$.

Theorem 1 (Th. 2, [14]): Define the higher-rate code $\mathcal{C}_h$ by

$$\mathcal{C}_h \triangleq \left\{ \boldsymbol{c} \in \{0,1\}^n : \boldsymbol{c} = \boldsymbol{v}\boldsymbol{G}, \forall \boldsymbol{v} \in \{0,1\}^{k+m} \right\}, \quad (8)$$

where $\boldsymbol{G} \in \{0,1\}^{(k+m)\times n}$ is the matrix representation of the convolutional encoder. Given a specified CRC degree m and a higher-rate code $\mathcal{C}_h$ with distance spectrum $B_{d^h_{\min}}, \ldots, B_n$, define w^* as the minimum w for which $\sum_{d=d^h_{\min}}^{w} B_d \geq 2^m$. For any degree-$m$ CRC polynomial, we have $d^l_{\min} \leq 2w^*$.

Theorem 1 shows that it suffices to choose $\tilde{d} = 2w^* + 1$ to identify the degree-m CRC polynomial that maximizes the minimal distance. In practice, the weight w^* can be efficiently determined from the weight enumerating function of a convolutional code [19, p. 488].

Table I presents the optimal CRC polynomials for ZTCCs generated with $H = (33, 25, 37, 31)$, $H = (47, 73, 57, 75)$, and $H = (107, 135, 133, 141)$. Table I also shows the minimum distance $d_{\min}$ and number of nearest neighbors $A_{d_{\min}}$ of the CRC-ZTCCs. The design assumes a fixed blocklength $N = 128$ bits. Due to the overhead caused by the CRC bits and by zero termination, the rates of CRC-ZTCCs are less than $3/4$. Specifically, for a given information length K, CRC degree m and an $(n, n-1, v)$ encoder, the blocklength N for a CRC-ZTCC is given by

$$N = \left(K + m + (n-1) \left\lceil \frac{v}{n-1} \right\rceil \right) \frac{n}{n-1}, \quad (9)$$

giving

$$R = \frac{K}{N} = \frac{n-1}{n} \frac{K}{K + m + (n-1)\lceil \frac{v}{n-1} \rceil}. \quad (10)$$

We see from (9) that the $(n, n-1, v)$ convolutional encoder can accept any CRC degree m as long as $K + m$ is divisible by $(n-1)$.

TABLE II
OPTIMAL CRC POLYNOMIALS FOR PUNCTURED RATE-3/4 ZTCC AT BLOCKLENGTH $N = 128$ GENERATED BY $G = (23, 25)$ WITH $v = 4$, BY $G = (53, 75)$ WITH $v = 5$, AND BY $G = (133, 171)$ WITH $v = 6$

$K+v$	m	$v=4$			$v=5$			$v=6$		
		CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$
96	0	0x1	3	31	0x1	4	29	0x1	5	223
95	1	0x3	4	29	0x3	5	224	0x3	5	112
94	2	0x7	6	2173	0x7	5	83	0x7	6	427
93	3	0xF	6	597	0x9	6	379	0x9	6	135
92	4	0x11	6	323	0x1F	6	53	0x13	7	245
91	5	0x27	6	101	0x39	7	213	0x23	8	1206
90	6	0x71	7	286	0x79	7	46	0x65	8	590
89	7	0xC7	7	54	0x85	8	216	0xFD	8	122
88	8	0x199	8	407	0x153	8	22	0x163	8	17
87	9	0x20B	8	68	0x353	9	247	0x247	10	2158
86	10	0x439	9	400	0x7CD	10	1631	0x4E7	10	342

TABLE III
OPTIMAL CRC POLYNOMIALS FOR STANDARD RATE-3/4 TBCC AT BLOCKLENGTH $N = 128$ GENERATED BY $H = (33, 25, 37, 31)$ WITH $v = 4$, BY $H = (47, 73, 57, 75)$ WITH $v = 5$, AND BY $H = (107, 135, 133, 141)$ WITH $v = 6$

K	m	$v=4$			$v=5$			$v=6$		
		CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$
96	0	0x1	4	64	0x1	5	224	0x1	6	864
95	1	0x3	4	32	0x3	5	128	0x3	6	384
94	2	0x7	5	96	0x7	5	64	0x7	6	128
93	3	0x9	6	736	0x9	6	192	0xB	6	36
92	4	0x1B	6	320	0x15	6	64	0x1D	6	6
91	5	0x25	6	31	0x37	6	2	0x23	7	49
90	6	0x4D	6	1	0x4F	7	98	0x53	8	326
89	7	0xA3	7	70	0xD1	8	446	0xB1	8	76
88	8	0x10D	8	411	0x149	8	73	0x1D3	8	8
87	9	0x2ED	8	138	0x255	8	14	0x3F7	9	208
86	10	0x63B	8	23	0x70F	8	1	0x529	9	90
85	11	0xCA5	9	125	0xD57	9	17	0x9BD	10	387
84	12	0x1ED7	10	904	0x1B41	10	339	0x10AF	10	53

B. CRC Polynomials for Punctured Zero-Terminated Codes

In [23], the authors proposed an efficient list decoding sieve method to identify the distance-optimal CRC polynomial of a given degree. This approach takes a noiseless all-zeros codeword as the received signal and performs serial list Viterbi decoding to explore codewords in order of increasing Hamming weight. For each new codeword added to the list, we check if it passes any of the degree-m CRC polynomials. If a CRC polynomial can eliminate all codewords of a certain Hamming weight, the sieve approach keeps this CRC polynomial and codewords of the next greater weight are explored. The list decoding sieve continues until it reaches a codeword weight where all CRC polynomials check at least one codeword. This weight is the largest $d_{\min}$ that a degree-m CRC polynomial can achieve. The CRC polynomial that checks the least number of codewords at $d_{\min}$ is selected as the optimal CRC polynomial. This approach is computationally more efficient than the error event construction method of Yang [14] while producing the same results.

We follow the puncturing patterns provided [22] for rate-$3/4$ CCs with $v = 4, 5, 6$. These punctured CCs are obtained from puncturing 4 out of every 6 bits (3 symbols) for rate-$1/2$ convolutional codes. The blocklength of all CRC-CCs is $N = 128$ bits.

Table II shows the optimal CRC polynomials obtained by the list decoding sieve approach for punctured ZTCCs generated with $G = (23, 25)$, $G = (53, 75)$, $G = (133, 171)$, as well as the corresponding $d_{\min}$ and $A_{d_{\min}}$ of the CRC-ZTCCs. For a feedforward code, the ZT condition is satisfied by inputting v zero bits at the end of the information sequence. Thus, for a given information length K, the blocklength N of the punctured code is given by

$$N = (K + m + v)\,\frac{n}{n-1}. \tag{11}$$

The real rate of this code is given by

$$R = \frac{K}{N} = \frac{n-1}{n}\,\frac{K}{K+m+v}. \tag{12}$$

In general, the punctured CRC-ZTCCs have smaller $d_{\min}$ values and more nearest neighbors than the standard CRC-ZTCCs.

C. CRC Polynomials for Standard Tail-Biting Codes

The design of CRC polynomials for standard high-rate TBCCs follows the two-phase design algorithm shown in [14]. This algorithm is briefly explained below.

Consider a TB trellis $T = (V, E, \mathcal{A})$ of length N, where $\mathcal{A}$ denotes the set of output alphabet, V denotes the set of states, and E denotes the set of edges described in an ordered triple (s, a, s') with $s, s' \in V$ and $a \in \mathcal{A}$ [32]. Assume $|V| = 2^v$ and let $V_0 = \{0, 1, \ldots, 2^v - 1\}$. Define the set of IEEs at state $\sigma \in V$ as

$$\mathrm{IEE}(\sigma) \triangleq \bigcup_{l=1,2,\ldots,N} \overline{\mathrm{IEE}}(\sigma, l), \tag{13}$$

where

$$\begin{aligned}\overline{\mathrm{IEE}}(\sigma, l) \triangleq \{&(\boldsymbol{s}, \boldsymbol{a}) \in V_0^{l+1} \times \mathcal{A}^l : s_0 = s_l = \sigma; \\ &\forall j, 0 < j < l,\ s_j \notin \{0, 1, \ldots, \sigma\}\}.\end{aligned} \tag{14}$$

By concatenating elements in $\mathrm{IEE}(\sigma)$, one can build an arbitrarily long TB path that starts and ends at state σ. The first phase is called the collection phase, during which the algorithm collects $\mathrm{IEE}(\sigma)$ with output Hamming weight less than the threshold $\tilde{d}$ over a sufficiently long TB trellis. The second phase is called the search phase, during which the algorithm first reconstructs all TB paths of length N/n and output weight less than $\tilde{d}$ via concatenation of the IEEs and circular shifting of the resulting path. Then, using these TB paths, the algorithm searches for the degree-m optimal CRC polynomial by maximizing the minimum distance of the undetected TB path.

Table III presents the optimal CRC polynomials for TBCCs generated with $H = (33, 25, 37, 31)$, $H = (47, 73, 57, 75)$, and $H = (107, 135, 133, 141)$. [2] The design assumes a fixed blocklength $N = 128$. TB encoding avoids the rate loss caused by the overhead of zero termination. Specifically, for a given information length K, CRC degree m and an $(n, n-1, v)$ encoder, the blocklength N for a CRC-TBCC is given by

$$N = (K + m)\,\frac{n}{n-1}, \tag{15}$$

[2]This table is updated from [1], as an error was discovered in the previous CRC polynomial design procedure.

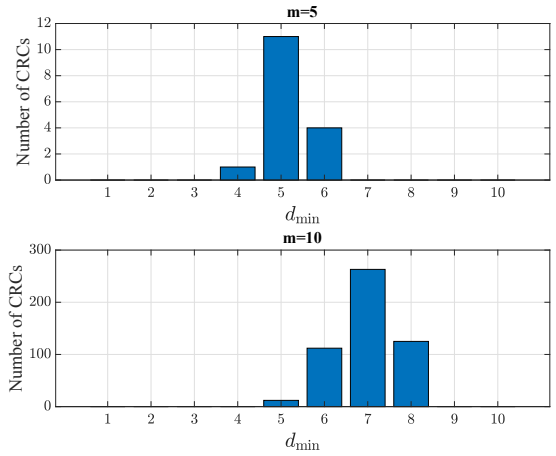

Fig. 4. Distribution of $d_{\min}$ of all CRC polynomials of degree 5 (top) and 10 (bottom) when concatenated with the $v = 4$ standard TBCC generated b $H = (33, 25, 37, 31)$.

giving

$$R = \frac{K}{N} = \frac{n-1}{n}\frac{K}{K+m}. \tag{16}$$

Fig. 4 shows the distribution of minimum distances for all degree-5 and 10 CRC polynomials for $v = 4$ CRC-TBCCs. While multiple CRC polynomials have the same maximized $d_{\min}$, their $A_{d_{\min}}$ differ. The optimal CRC polynomial designs have the minimal $A_{d_{\min}}$ value. This range of $d_{\min}$ values validates the effectiveness of our CRC polynomial design approach in the high-rate scenario.

D. CRC Polynomials for Punctured Tail-Biting Codes

The sieve method described in Sec. IV-B is extended to TBCCs efficiently with the application of the multi-trellis SLVD, since all codewords discovered by the multi-trellis satisfy the tail-biting condition. Table IV shows the optimal CRC polynomial designs for $v = 4, 5, 6$ punctured TBCCs generated with $G = (23, 25)$, $G = (53, 75)$, $G = (133, 171)$. Since there is no overhead of terminations, the punctured TBCCs have the same blocklength and rate at a given information length. Similar to the ZT case, a rate-$1/2$ code punctured to rate-$3/4$ has slightly worse $d_{\min}$ and $A_{d_{\min}}$ values compared to a standard rate-$3/4$ TBCC.

V. Complexity Analysis

In this section, we will discuss the decoding complexity for all SLVD methods presented in Sec. III. Section V-A covers the complexity analysis of rate-$(n-1)/n$ standard ZTCCs and TBCCs on a dual trellis. Section V-B provides the decoding complexity equations for punctured CCs. Finally, section V-C visualizes the performance-complexity trade-offs between the standard and punctured codes. The WAVA decoder is a low-complexity alternative for TBCCs, and we explore its complexity and performance for standard TBCCs.

TABLE IV
Optimal CRC Polynomials for Punctured rate-3/4 TBCC at Blocklength $N = 128$ Generated by $G = (23, 25)$ With $v = 4$, by $G = (53, 75)$ With $v = 5$, and by $G = (133, 171)$ With $v = 6$

K	m	$v=4$			$v=5$			$v=6$		
		CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$	CRC	$d_{\min}$	$A_{d_{\min}}$
96	0	0x1	3	32	0x1	4	32	0x1	5	256
95	1	0x3	4	32	0x3	5	256	0x3	5	128
94	2	0x7	6	2512	0x7	5	96	0x7	6	512
93	3	0xF	6	688	0x9	6	448	0x9	6	160
92	4	0x11	6	368	0x15	6	96	0x1B	6	64
91	5	0x33	6	176	0x25	6	7	0x3F	8	1637
90	6	0x71	6	7	0x55	7	224	0x77	8	767
89	7	0xD5	6	2	0xC3	8	1166	0xBD	8	365
88	8	0x1EB	7	20	0x129	8	281	0x101	8	27
87	9	0x343	8	211	0x367	8	79	0x2B7	8	4
86	10	0x677	8	69	0x41D	8	4	0x40D	8	1

A. Dual Trellis SLVD for Standard ZTCC and TBCC

In [14], the authors provided the complexity expression for SLVD of CRC-ZTCCs and CRC-TBCCs, where the convolutional encoder is of rate $1/n$. Observe that the dual trellis has no more than 2 outgoing branches per state, similar to the trellis of a rate-$1/n$ CC. Thus, we directly apply their complexity expression to SLVD over the dual trellis.

As noted in [14], the overall average complexity of SLVD can be decomposed into three components:

$$C_{\text{SLVD}} = C_{\text{SSV}} + C_{\text{trace}} + C_{\text{list}}, \tag{17}$$

where C_{SSV} denotes the complexity of a standard soft Viterbi (SSV), C_{trace} denotes the complexity of the *additional* traceback operations required by SLVD, and C_{list} denotes the average complexity of inserting new elements to maintain an ordered list of path metric differences.

C_{SSV} is the complexity of ACS operations and the initial traceback operation. For CRC-ZTCCs,

$$C_{\text{SSV}} = (2^{v+1}-2) + 1.5(2^{v+1}-2) + 1.5(K+m-v)2^{v+1} + c_1[2(K+m+v) + 1.5(K+m)]. \tag{18}$$

For CRC-TBCCs decoded using the single-trellis, this quantity is given by

$$C_{\text{SSV}} = 1.5(K+m)2^{v+1} + 2^v + 3.5c_1(K+m). \tag{19}$$

For CRC-TBCCs with the multi-trellis approach,

$$C_{\text{SSV}} = 2^v[1.5(K+m)2^{v+1}] + 3.5c_1(K+m). \tag{20}$$

The second component C_{trace} for CRC-ZTCC is given by

$$C_{\text{trace}} = c_1(\mathsf{E}[L]-1)[2(K+m+v) + 1.5(K+m)]. \tag{21}$$

For CRC-TBCCs, C_{trace} is given by

$$C_{\text{trace}} = 3.5c_1(\mathsf{E}[L]-1)(K+m), \tag{22}$$

for both single-trellis and multi-trellis approaches.

The third component, which is identical for ZT and TB, is

$$C_{\text{list}} = c_2\mathsf{E}[I]\log(\mathsf{E}[I]). \tag{23}$$

For CRC-ZTCCs,

$$\mathsf{E}[I] \le (K+m)\mathsf{E}[L], \tag{24}$$

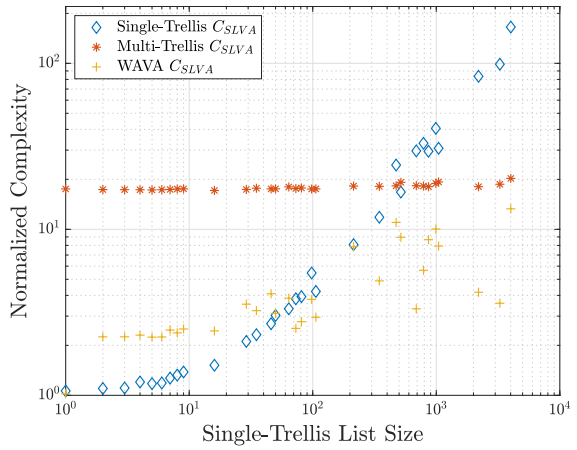

Fig. 5. The overall complexity comparison of the single-trellis, multi-trellis, and WAVA decoders for the TBCC generated with the $(4, 3, 4)$ encoder $H = (33, 25, 37)$, with blocklength of 128. The CRC polynomial of degree 3 is 0x9. All complexity values are normalized with respect to the single-trellis C_{SSV} at different list sizes.

and for CRC-TBCCs with either single-trellis or multi-trellis approach,

$$\mathsf{E}[I] \leq (K+m)\mathsf{E}[L] + 2^v - 1. \tag{25}$$

In the above expressions, c_1 and c_2 are two computer-specific constants that characterize implementation-specific differences in the implemented complexity of traceback and list insertion (respectively) as compared to the ACS operations of Viterbi decoding. In this paper, we assume that $c_1 = c_2 = 1$ and use (24) and (25) to estimate $\mathsf{E}[I]$ for CRC-ZTCCs and CRC-TBCCs.

Note that $\mathsf{E}[I]$ and $\mathsf{E}[L]$ values vary depending on whether the single-trellis or multi-trellis approach is used. Using the multi-trellis approach significantly reduces C_{trace} and C_{list} because only TB paths are included. On the other hand, as seen from (19) and (20), the multi-trellis approach amplifies the first component C_{SSV} by nearly 2^v. The overall trade-off is depicted in Fig. 5, which shows the complexity comparison of the three proposed SLVD methods for a $v = 4, m = 3$ standard CRC-TBCC decoded using the dual trellis. Random codewords with blocklength $N = 128$ are generated and their single-trellis list sizes are measured by passing through a single-trellis SLVD. The runtime of each complexity component is normalized with respect to the value of single-trellis C_{SSV}. When the single-trellis list size is 1, the multi-trellis SLVD has an overall runtime that is over 10 times greater than that of the single-trellis SLVD. At low noise levels, the list size of a single trellis is almost always 1, resulting in a substantially lower runtime compared to that of a multi-trellis. As SNR decreases, there is an exponential growth in the complexity terms C_{trace} and C_{list} for the single-trellis decoder. The list size grows much more slowly for the mutli-trellis decoder because it does not include non-TB codewords in the list. As a result, trellis construction is the main contributor to the complexity of multi-trellis. Thus the multi-trellis decoder has similar complexity across all SNR levels. At a single-trellis list size of around 5×10^2, the overall runtime C_{SLVD} of both approaches becomes the same. This indicates that at high SNRs, single-trellis is the optimal approach. But when the noise level is high, the multi-trellis approach has a more favorable runtime since it guarantees to satisfy the TB condition.

Upon applying WAVA, the overall average complexity for CRC-TBCC is incremented by ACS operations during the additional forward pass, if needed. Let the probability that the optimal path of the initial traceback does not satisfy either TB or CRC condition be P_{WAVA}. The list rank of the decoder is 1 with a probability of $1 - P_{\text{WAVA}}$. Thus we have the updated complexity:

$$C_{\text{SLVD}} = C_{\text{SSV}} + P_{\text{WAVA}}(C_{\text{WAVA}} + C_{\text{trace}} + C_{\text{list}}), \tag{26}$$

where

$$C_{\text{WAVA}} = 1.5(K+m)2^{v+1} + 2^v. \tag{27}$$

The yellow data points in Fig. 5 represent the overall complexity of the WAVA decoder normalized with respect to the single-trellis C_{SSV}. The complexity for initializing the WAVA decoder is about 2 times of that for the single-trellis decoder, giving it a disadvantage at low SNRs. When list size is 1, the WAVA decoder matches the complexity of the single-trellis decoder since one iteration is sufficient. At a list size of around 50, the overall complexity of the WAVA decoder reaches the same level as the single-trellis decoder. The WAVA decoder always operates at a complexity lower than the multi-trellis decoder.

B. Primal Trellis SLVD for Punctured ZTCC and TBCC

A punctured convolutional code of rate $(n-1)/n$ is obtained from puncturing the outputs of a rate-$1/2$ code. Therefore, the complexity of the SLVD for the punctured and original codes are the same, which is presented in [14]. To keep this section self-contained, we will show the rate-$1/n$ complexity analysis here.

The overall complexity of the punctured SLVD consists of the same three components as that of the standard SLVD in 17.

For CRC-ZTCCs,

$$\begin{aligned} C_{\text{SSV}} &= 5(2^v - 1) + 3(K + m - v)2^v \\ &\quad + c_1[2(K + m + v) + 1.5(K + m)]. \end{aligned} \tag{28}$$

For CRC-TBCCs,

$$C_{\text{SSV}} = (3K + 3m + 1)2^v + 3.5c_1(K + m). \tag{29}$$

The other two components C_{trace} and C_{list}, as well as the expected number of insertions $\mathsf{E}[I]$, remain the same for punctured SLVD as the dual-trellis SLVD (Eq. 21 - 25).

C. Complexity Comparison

Fig. 6 and 7 display the trade-off between the SNR gap to the RCU bound and the average decoding complexity at the target FER of 10^{-4} for CRC-ZTCCs designed in Table I and CRC-TBCCs designed in Table III. In addition, these figures directly compare the proposed dual trellis decoding scheme

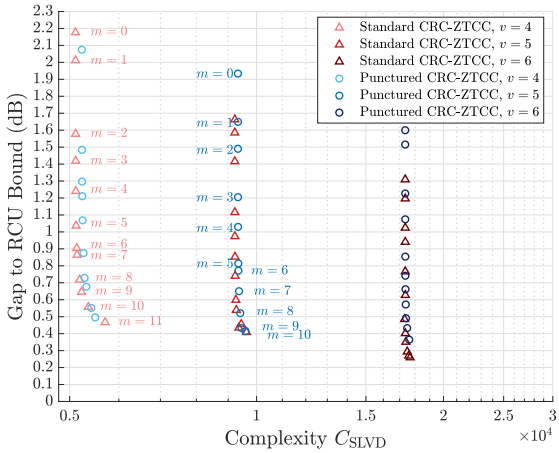

Fig. 6. The SNR gap to the RCU bound vs. the average complexity of SLVD of standard CRC-ZTCC codes in Table I and punctured CRC-ZTCC codes in Table II for target FER of 10^{-4}. Markers from top to bottom with the same color correspond to CRC polynomials with m = 0, . . . , 11 for standard CRC-ZTCCs, and m = 0, . . . , 10 for punctured CRC-ZTCCs. For punctured ZTCC with $v = 4, m = 0$, the gap to RCU bound is substantially high at 2.8262 dB.

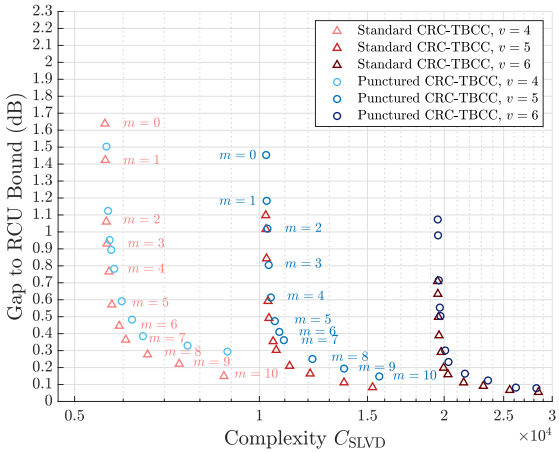

Fig. 7. The SNR gap to the RCU bound vs. the average complexity of SLVD of standard CRC-TBCC codes in Table III and punctured CRC-TBCC codes in Table IV for target FER of 10^{-4}. Markers from top to bottom with the same color correspond to CRC polynomials with m = 0, . . . , 10. For punctured TBCC with $v = 4, m = 0$, the gap to RCU bound is substantially high at 2.4257 dB.

with the punctured scheme. The average decoding complexity of SLVD is evaluated according to the expressions in Sec. V-A and V-B. We see that for a fixed v (ZT or TB, standard or punctured), increasing the CRC degree m significantly reduces the gap to the RCU bound, at the cost of a small increase in complexity. CRC-TBCCs generally have greater complexity than CRC-ZTCCs because the list decoder goes through many non-TB codewords. The minimum gap of 0.25 dB is achieved by the standard CRC-ZTCC with $v = 6$ and $m = 10$, and the minimum gap of 0.05 dB is achieved by the CRC-TBCC with $v = 6$ and $m = 10$. For CRC-TBCCs, the gap to RCU bound continues to decrease when CRCs of higher degrees are applied, but the complexity grows substantially. For a more legible figure, we only show CRC polynomials of degrees up to 10 in Fig. 7.

For the same CRC degree m, increasing the overall constraint length v dramatically increases the complexity, while achieving a minimal reduction in the SNR gap to the RCU bound. On the other hand, the performance of CRC-ZTCC can be improved drastically by applying CRC polynomials of higher degrees. Both Fig. 6 and 7 demonstrate that for all three cases of constraint lengths v, one additional bit in the CRC benefits the decoding performance by moving closer to the RCU gap with a minimal cost in complexity.

Additionally, for the same CRC degree m and constraint length v, the standard high-rate codes generally perform better than the punctured codes while maintaining a similar decoding complexity. As the CRC degree increases, the performance and complexity of these two coding schemes draw nearer. Note that for CRC-ZTCCs, the rates for standard and punctured codes are different for $v = 4$ and $v = 5$, where the punctured codes have a higher rate due to fewer termination overhead bits.

Fig. 8 shows the trade-off of complexity and performance for decoding CRC-TBCCs with a single trellis SLVD and a WAVA-based SLVD. The CRC-TBCCs used are of rate-3/4

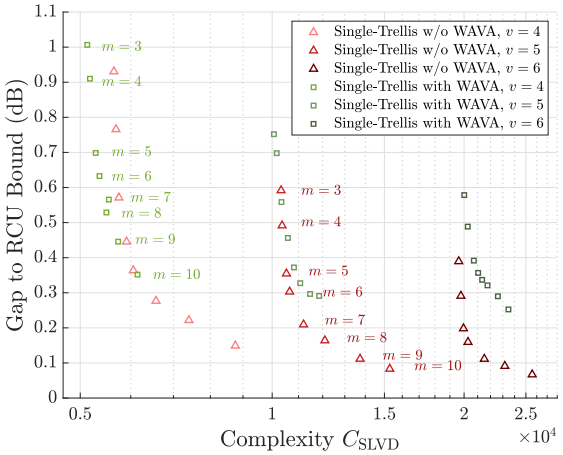

Fig. 8. The SNR gap to the RCU bound vs. the average complexity of SLVD of CRC-TBCC codes in Table III for target FER of 10^{-4}. The results for both single-trellis decoding and WAVA decoding are demonstrated. Each color represents a specific CRC-aided CC shown in the table. Markers from top to bottom with the same color correspond to CRC polynomials with m = 3, . . . , 10, where $m = 0$ represents the convolutional codes without CRC.

for $m = 3, \ldots 10$ in III. The WAVA decoder has a larger gap to RCU bound than the single-trellis decoder due to the extra ACS operations during the first traceback. However, the complexity of the WAVA decoder is smaller than that of the single-trellis decoder, and the difference increases as the CRC degree increases. For all constraint lengths v, the WAVA decoder at $m = 10$ has a similar complexity as the single-trellis decoder at $m = 6$.

VI. Results and Discussion

In this section, we will report and discuss the FER vs SNR performances of the CRC-CCs. Section VI-A compares the standard CRC-ZTCC with the punctured CRC-ZTCC, as well as shows the performance difference between using a single

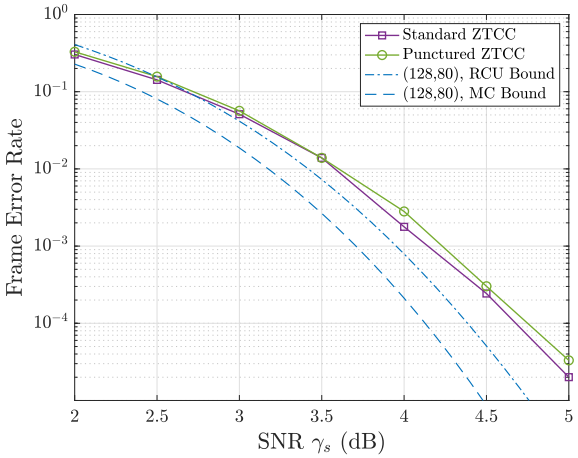

Fig. 9. FER vs. SNR for $v = 6, m = 10$ standard and punctured CRC-ZTCCs. The standard ZTCC is generated with the $(4, 3, 6)$ encoder $H = (107, 135, 133, 141)$ and the punctured ZTCC is generated by $G = (133, 171)$. The optimal CRC polynomials of degree 10 are 0x59F and 0x4E7, respectively. For the RCU and MC bounds, values in parenthesis denote blocklength N and information length K, respectively.

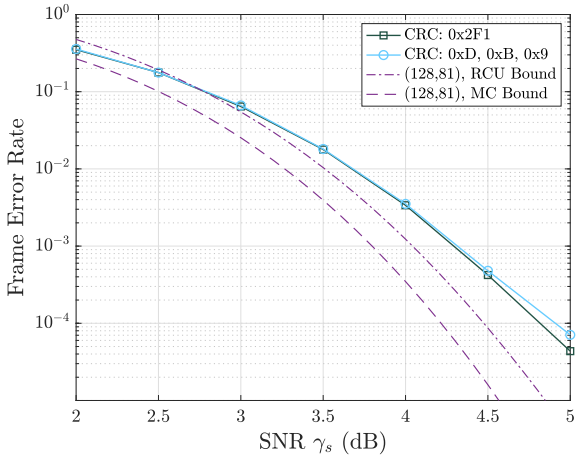

Fig. 10. FER vs. SNR for $v = 6$, $m = 9$ CRC-ZTCCs designed under Karimzadeh *et al.*'s scheme [24] and our scheme. Both CRC-ZTCCs have information length $K = 81$ and blocklength $N = 128$.

longer CRC and multiple shorter CRCs. Section VI-B covers the performance of standard and punctured CRC-TBCCs.

A. CRC-ZTCC Results

Fig. 9 shows the performance comparison of standard and punctured $v = 6$ CRC-ZTCCs with degree-10 CRC polynomials. For both CRC-ZTCCs, the blocklength is 128 bits and the information length is 80 bits, yielding a code rate of 0.625. The standard CRC-ZTCC has slightly better FER performance than the punctured code. At the target FER of 10^{-4}, the gap between the two schemes is around 0.08 dB.

In [24], Karimzadeh *et al.* considered designing optimal CRC polynomials for each input rail of a multi-input CC. In their setup, an information sequence for an $(n, n-1, v)$ encoder needs to be split into $(n-1)$ subsequences before CRC encoding. In contrast, the entire information sequence in our framework is encoded with a single CRC polynomial. Then the resulting sequence is evenly divided into $(n-1)$ subsequences, one for each rail. To compare the performance between these two schemes, we design three degree-3 optimal CRC polynomials, one for each rail, for ZTCC with $H = (107, 135, 133, 141)$. The three CRC polynomials jointly maximize the minimum distance of the CRC-ZTCC. For the single-CRC design, we use the single degree-9 optimal CRC polynomial for the same encoder from Table I. Both CRC-ZTCCs have an information length $K = 81$ and blocklength $N = 128$. Fig. 10 shows the performance comparison between these two codes, showing that at high SNRs, a single degree-9 optimal CRC polynomial outperforms three degree-3 optimal CRC polynomials, one for each rail. This suggests that a single optimal CRC polynomial may suffice to provide superior protection for each input rail. The decoding complexity is similar regardless of the CRC scheme.

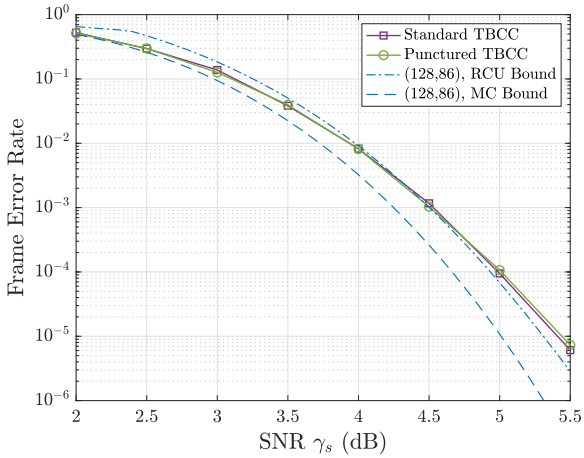

Fig. 11. FER vs. SNR for $v = 6, m = 10$ standard and punctured CRC-TBCCs. The standard TBCC is generated with the $(4, 3, 6)$ encoder $H = (107, 135, 133, 141)$ and the punctured TBCC is generated by $G = (133, 171)$. The optimal CRC polynomials of degree 10 are 0x529 and 0x40D, respectively. A single-trellis list decoder is used.

B. CRC-TBCC Results

The performance of CRC-aided list decoding of ZTCCs relative to the RCU bound is constrained by the termination bits appended to the end of the original message, which are required to bring the trellis back to the all-zero state. TBCCs avoid this overhead by replacing the zero termination condition with the TB condition, which states that the final state of the trellis is the same as the initial state of the trellis [33].

Fig. 11 shows the FER vs. SNR for standard and punctured $v = 6$ CRC-TBCCs with degree-10 CRC polynomials at a fixed blocklength of 128 and information length of 86. Both codes are able to closely approach the RCU bound. At the target FER of 10^{-4}, the gap between the two schemes is within 0.05 dB. Compared to the CRC-TBCCs with the same v and m, the CRC-ZTCCs have a rate loss of around 0.04 dB because of the termination overhead.

VII. Conclusion

This paper shows that both standard and punctured high-rate CRC-aided CCs are able to approach the RCU bound for the BI-AWGN channel. The best CRC-TBCCs with the single-trellis ML decoder approach the RCU bound within 0.1 dB for a target FER of 10^{-4} at a blocklength of $N = 128$ bits. Concatenated with optimal CRC polynomials, the performance and complexity of the standard and punctured high-rate convolutional codes are similar. In addition, adding one bit to the CRC can improve the FER more than adding an additional memory element to the CC does for both standard and punctured CRC-CC schemes.

For rate-$(n-1)/n$ TBCCs concatenated with optimal CRC polynomials, this paper considers three list decoding algorithms: a multi-trellis approach, a single-trellis approach, and a modified single trellis approach with pre-processing using the Wrap Around Viterbi Algorithm (WAVA). For the cases of standard codes, on which our simulations focused, all three algorithms use the dual trellis to reduce complexity. The multi-trellis approach achieves the smallest expected list rank, but it suffers from a significantly larger overall complexity than the single-trellis approach. For the single trellis approach, we consider both an ML decoder and a non-ML decoder that uses WAVA pre-processing. WAVA pre-processing achieves a significantly smaller expected list size at the price of a worse FER performance.

References

[1] W. Sui, H. Yang, B. Towell, A. Asmani, and R. D. Wesel, "High-rate convolutional codes with CRC-aided list decoding for short blocklengths," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 98–103.

[2] M. Rice, "Comparative analysis of two realizations for hybrid-ARQ error control," in *1994 IEEE Global Commun. Conf.*, 1994, pp. 115–119.

[3] "Universal mobile telecommunications system (UMTS); multiplexing and channel coding (FDD); 3GPP TS 25.212 version 7.0.0 release 7," European Telecommunications Standards Institute, Tech. Rep., 2006.

[4] "LTE; evolved universal terrestrial radio access (E-UTRA); multiplexing and channel coding; 3GPP TS 36.212 version 15.2.1 release 15," European Telecommunications Standards Institute, Tech. Rep., 2018.

[5] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[6] H. Ji, S. Park, J. Yeo, Y. Kim, J. Lee, and B. Shim, "Ultra-reliable and low-latency communications in 5G downlink: Physical layer aspects," *IEEE Wireless Commun. Mag.*, vol. 25, no. 3, pp. 124–130, 2018.

[7] L. Gaudio, T. Ninacs, T. Jerkovits, and G. Liva, "On the performance of short tail-biting convolutional codes for ultra-reliable communications," in *SCC 2017; 11th Int. ITG Conf. Syst., Commun., and Coding*, Feb. 2017, pp. 1–6.

[8] M. C. Coşkun, G. Durisi, T. Jerkovits, G. Liva, W. Ryan, B. Stein, and F. Steiner, "Efficient error-correcting codes in the short blocklength regime," *Physical Commun.*, vol. 34, pp. 66 – 79, 2019.

[9] C. Yue, M. Shirvanimoghaddam, B. Vucetic, and Y. Li, "A revisit to ordered statistics decoding: Distance distribution and decoding rules," *IEEE Trans. Inf. Theory*, vol. 67, no. 7, pp. 4288–4337, 2021.

[10] L. Dolecek, D. Divsalar, Y. Sun, and B. Amiri, "Non-binary protograph-based LDPC codes: Enumerators, analysis, and designs," *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 3913–3941, 2014.

[11] G. Liva, E. Paolini, B. Matuz, S. Scalise, and M. Chiani, "Short turbo codes over high order fields," *IEEE Trans. Commun.*, vol. 61, no. 6, pp. 2201–2211, 2013.

[12] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[13] E. Arıkan, "From sequential decoding to channel polarization and back again." [Online]. Available: http://arxiv.org/abs/1908.09594

[14] H. Yang, E. Liang, M. Pan, and R. D. Wesel, "CRC-aided list decoding of convolutional codes in the short blocklength regime," *IEEE Trans. Inf. Theory*, vol. 68, no. 6, pp. 3744–3766, 2022.

[15] C. Y. Lou, B. Daneshrad, and R. D. Wesel, "Convolutional-code-specific CRC code design," *IEEE Trans. Commun.*, vol. 63, no. 10, pp. 3459–3470, Oct. 2015.

[16] N. Seshadri and C. E. W. Sundberg, "List Viterbi decoding algorithms with applications," *IEEE Trans. Commun.*, vol. 42, no. 234, pp. 313–323, Feb. 1994.

[17] R. Schiavone, "Channel coding for massive IoT satellite systems," Master's thesis, Politechnic University of Turin (Polito), 2021.

[18] R. Schiavone, R. Garello, and G. Liva, "Performance improvement of space missions using convolutional codes by CRC-aided list Viterbi algorithms," *IEEE Access*, vol. 11, pp. 55 925–55 937, 2023.

[19] S. Lin and D. J. Costello, *Error Control Coding: fundamentals and applications*, 2nd ed. New Jersey, USA: Pearson Prentice Hall, 2004.

[20] J. B. Cain, G. C. Clark, and J. M. Geist, "Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding (corresp.)," *IEEE Trans. Inf. Theory*, vol. 25, pp. 97–100, 1979.

[21] T. Yamada, H. Harashima, and H. Miyakawa, "A new maximum likelihood decoding of high rate convolutional codes using a trellis," *Elec. and Commun. in Japan Part I-commun.*, vol. 66, pp. 11–16, 1983.

[22] D. Haccoun and G. Begin, "High-rate punctured convolutional codes for Viterbi and sequential decoding," *IEEE Transactions on Communications*, vol. 37, no. 11, pp. 1113–1125, 1989.

[23] R. Wesel, A. Antonini, L. Wang, W. Sui, B. Towell, and H. Grissett, "ELF codes: Concatenated codes with an expurgating linear function as the outer code," in *2023 13th Int. Sym. Topics in Coding (submitted) arXiv:2306.07467*, 2023.

[24] M. Karimzadeh and M. Vu, "Optimal CRC design and serial list Viterbi decoding for multi-input convolutional codes," in *2020 IEEE Global Commun. Conf.*, 2020, pp. 1–6.

[25] R. Shao, S. Lin, and M. Fossorier, "Two decoding algorithms for tailbiting codes," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1658–1665, 2003.

[26] C. Douillard, M. Jézéquel, C. Berrou, N. Brengarth, J. Tousch, and N.-N. Pham, "The turbo code standard for DVB-RCS," 2004.

[27] F. Soong and E.-F. Huang, "A tree-trellis based fast search for finding the n-best sentence hypotheses in continuous speech recognition," in *ICASSP 91: 1991 Int. Conf. Acoustics, Speech, and Signal Processing*, 1991, pp. 705–708 vol.1.

[28] A. Hasham, "A new class of priority queue organizations," Master's thesis, 1986, aAI0662089.

[29] M. Roder and R. Hamzaoui, "Fast tree-trellis list Viterbi decoding," *IEEE Trans. Commun.*, vol. 54, no. 3, pp. 453–461, Mar. 2006.

[30] R. Hinze, "Constructing red-black trees," 10 1999.

[31] J. King, W. Ryan, C. Hulse, and R. D. Wesel, "Efficient maximum-likelihood decoding for TBCC and CRC-TBCC codes via parallel list Viterbi," 2023.

[32] R. Koetter and A. Vardy, "The structure of tail-biting trellises: minimality and basic principles," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2081–2105, Sep. 2003.

[33] H. Ma and J. Wolf, "On tail biting convolutional codes," *IEEE Trans. Commun.*, vol. 34, no. 2, pp. 104–111, Feb. 1986.