Instantaneous Feedback-Based Opportunistic Symbol Length Adaptation for Reliable Communication

Chin-Wei Hsu, Achilleas Anastasopoulos, *Senior Member, IEEE*, and Hun-Seok Kim, *Senior Member, IEEE*

***Abstract*— Although feedback cannot increase the channel capacity of memoryless channels, it can enhance the error rate performance and/or shorten the codeword length for the target performance. This work is based on an early work by Viterbi in 1965 that utilizes instantaneous feedback for reliable uncoded communications. We build on this work by incorporating convolutional codes as a new variable-symbol-length digital communication scheme using instantaneous feedback. In the proposed system, called Opportunistic Symbol Length Adaptation (OSLA), the symbol length opportunistically adapts to the noise realization observed within a sub-symbol interval to minimize the packet/codeword error rate. It is shown that the proposed OSLA scheme combined with tail-biting convolutional codes or turbo codes outperforms state-of-the-art non-feedback codes as well as a deep learning-based feedback scheme with up to 1.5 dB gain in noiseless and noisy feedback channels.**

***Index Terms*— Feedback communications, ultra reliable low latency communication (URLLC).**

I. Introduction

IT IS well known that feedback cannot increase the capacity of a memoryless additive white Gaussian noise (AWGN) channel or memoryless discrete channel [2, Chapter 7.12, 9.6]. It, however, can significantly increase the error exponent to improve the error rate[1] [5]. This benefit is important for short blocklength codes as capacity-achieving codes typically rely on long blocklengths. Therefore, feedback-based transmission is of high interest in the regime of short blocklength communications for emerging applications such as real-time control of autonomous vehicles to enable more reliable communications with an enhanced error rate exponent.

Since the study on feedback codes in Shannon's famous work [5], several communication schemes have been proposed that use feedback to pursue high reliability without relying on complicated code designs. The classic Schalkwijk-Kailath (SK) scheme can achieve a super exponential decaying rate of error probability w.r.t the blocklength (infinite error exponent) in AWGN channel with a simple yet elegant strategy when the feedback is noiseless [6]. However, the SK scheme is extremely sensitive to noise in the feedback channel, to the extend that even some small arithmetic imprecision prohibits it from attaining the claimed performance, thus it has been regarded as a practically infeasible scheme. Several works have proposed to solve the problem with modified algorithms [7], [8], [9], [10]. Although the assumptions in [7], [8], [9], and [10] for the feedback channel quality and computation precision are significantly relaxed compared to the original SK scheme, they are still unattainable in practical communication systems especially when the blocklength is relatively long. On the other end of the spectrum, deep learning-based schemes such as Deepcode [11] and generalized block attention feedback (GBAF) codes [12] were shown to have superior performance even with noisy feedback by exploiting the modeling power of deep neural networks. Deepcode and GBAF codes have been proven to be very effective for codeword-level feedback-based communications. However, they come with certain drawbacks such as limited scalability to longer blocklengths, and inflexible network models that need to be specifically trained for each code rate and length configuration.

Variable-blocklength codes have been well studied as a feedback-based reliable communication scheme. Burnashev [13], and Yamamoto and Itoh [14] proposed variable-blocklength codes that achieve the channel capacity with an optimal error exponent. Variable-blocklength codes can provide performance advantages over fixed-blocklength codes [3], [4], but they often come with the difficulty of maintaining state-synchronization between two communicating ends. To address that issue, a feedback coding-based synchronization scheme for noisy feedback channels was proposed in [15]. However, out-of-sync recovery in [15] still requires long delays, which is not consistent with applications requiring short blocklengths. Therefore, designing simple and robust feedback schemes for variable-blocklength codes is a challenge of great interest.

Surprisingly, very limited feedback information such as informing the encoder to terminate the transmission for variable blocklength can attain considerably faster convergence to capacity as analyzed in [16]. A similar concept of using

Manuscript received 8 July 2022; revised 13 December 2022 and 26 March 2023; accepted 4 April 2023. Date of publication 11 April 2023; date of current version 17 July 2023. This work was funded in part by NSF CAREER #1942806. An earlier version of this paper was presented in part at IEEE Global Communications Conference (GLOBECOM) [DOI: 10.1109/GLOBECOM46510.2021.9685257]. The associate editor coordinating the review of this article and approving it for publication was A. E. Pusane. *(Corresponding author: Chin-Wei Hsu.)*

The authors are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: chinweih@umich.edu; anastas@umich.edu; hunseok@umich.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2023.3266356.

Digital Object Identifier 10.1109/TCOMM.2023.3266356

[1]This is not always true for symmetric discrete memoryless channels where fixed-length codes can have the same error exponent without feedback [3], [4].

feedback to terminate the transmission can also be found in an early work by Viterbi [17] using sequential decision feedback. Unlike a variable-blocklength scheme that still has constant symbol length, the scheme proposed in [17] transmits the signal for a bit or a symbol (consisting of multiple bits) with variable duration and it is shown to provide up to 6dB SNR gain compared to a fixed symbol length (uncoded) transmission. However, the work did not gain much attention probably because of the impractical instantaneous feedback assumption in the era of analog communications as well as the fact that transmission was uncoded.

Thanks to modern digital integrated circuit (IC) technology for fast and low latency feedback decision computation, it is now practically possible to utilize instantaneous feedback (i.e., with much shorter delay than the symbol length) to improve communication reliability. Inspired by [17], we introduce Opportunistic Symbol Length Adaptation (OSLA), a feedback-based scheme that opportunistically adjusts the symbol length based on the noise realization observed at the receiver with sub-symbol granularity. OSLA is non-trivial generalization of [17] and it operates in discrete time for coded communication with feedback. Unlike Viterbi's work, where an M-ary signal is used to transmit a message of $\log_2 M$ bits to improve reliability, our system uses a multi-dimensional BPSK to transmit multiple coded bits without the constraint of using the same symbol length for $\log_2 M$ coded bits that are concurrently transmitted in an M-ary symbol. We propose a deliberate feedback scheme that prevents catastrophic feedback errors in noisy feedback channels, while using a tail-biting convolutional code (TBCC) or a turbo code in the forward channel.

OSLA is a *practical* instantaneous feedback-based scheme that can be applied to a communication system utilizing a convolutional code. Besides its superior reliability, OSLA possesses additional advantages of constant envelope signaling and low complexity transmitters unlike deep learning-based schemes such as Deepcode [11], which may be important for Internet-of-Things applications where low-cost and low-complexity transmitters are favored. Furthermore, OSLA utilizes binary decision feedback, which does not suffer from any arithmetic imprecision, and is robust to feedback noise. We therefore consider our proposed system as a strong candidate for ultra-reliable low-latency communications (URLLC) with short blocklengths in next generation wireless communication standards.

The contributions of this work are summarized as follows:

1) We propose OSLA, a feedback-based variable-symbol-length transmission scheme, to reliably transmit convolutional (including TBCC and turbo) coded messages.
2) We generalize [17] and propose a new feedback decision scheme that determines the length of each coded bit based on the real-time sub-symbol-granularity realization of state (and branch) metrics of the decoding algorithm executed at the receiver.
3) It is shown that OSLA combined with TBCC outperforms fixed-length state-of-the-art short codes as well as the recently proposed deep learning-based feedback scheme Deepcode [11].
4) Scalability of OSLA combined with turbo codes is shown with various blocklengths and signal-to-noise ratios (SNRs) to provide consistent gains over a conventional fixed-blocklength scheme that does not utilize feedback.
5) We propose and evaluate a novel feedback scheme for OSLA that uses a pulse-based feedback signal with a hidden Markov model to synchronize the transmitter and receiver in noisy feedback channels for robust communications.

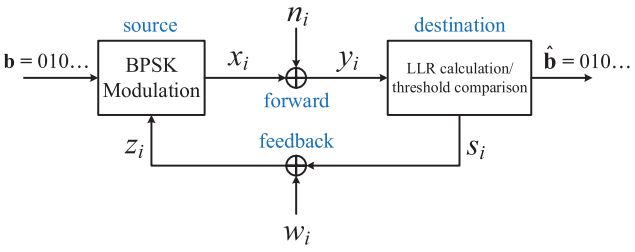


Fig. 1. OSLA-BPSK system model.

The remaining parts of this paper are organized as follows. Section II presents the proposed OSLA system for uncoded BPSK transmission. We then generalize the system to incorporate trellis-based coding including TBCC and turbo codes in section III. Our proposed feedback scheme for OSLA is discussed in section IV. Evaluation and comparison to other schemes are provided in Section V. Section VII discusses the limitations and future directions of this work. Finally, Section VI concludes the paper.

II. OSLA for Uncoded BPSK

A. OSLA-BPSK System Model

In this section, we introduce the discrete-time system model of sequential decision feedback for uncoded BPSK transmission. Although a similar continuous-time counterpart has been introduced in [17], we describe our discrete-time model for completeness of the paper. We use the term *OSLA-BPSK* for the uncoded system which will be generalized to a trellis coded scheme in a subsequent section.

The OSLA-BPSK system involves a source and destination with two channels connecting them: the forward and feedback channel which are discrete in time as illustrated in Figure 1. In this section, we assume noiseless feedback and thus the source receives perfect feedback from the destination.

The source modulates the kth bit b_k with a (variable length) BPSK symbol that consists of a series of chips $x_{k,i}, i = 1, \cdots, N_k$, where a chip refers to the smallest unit of transmission and is generally much shorter than a symbol in timescale. Here i is the chip index and N_k denotes the number of chips for b_k. As will be explained below, N_k is a random quantity that depends on the channel noise realization. Considering an AWGN model for the forward channel, the received chip is

modeled as

$$y_{k,i} = x_{k,i} + n_{k,i} \tag{1}$$

where $x_{k,i} = (-1)^{b_k}\sqrt{P}$ with P denoting the signal power and $n_{k,i} \sim \mathcal{N}(0, \sigma_n^2)$ is zero-mean Gaussian noise with variance σ_n^2. The log likelihood ratio (LLR) of each chip can be written as

$$\Delta L_{k,i} = \log \frac{Pr(y_{k,i}|b_k = 0)}{Pr(y_{k,i}|b_k = 1)} = \frac{2\sqrt{P}y_{k,i}}{\sigma_n^2}. \tag{2}$$

The LLR of b_k upon receiving all chips $y_{k,i}$ is the summation of the LLR of each chip (due to the independence of the noise)

$$\text{LLR}(b_k) = \sum_{i=1}^{N_k} \Delta L_{k,i}. \tag{3}$$

The LLR in (3) can be calculated recursively in time as

$$L_i(b_k) = L_{i-1}(b_k) + \Delta L_{k,i}, \quad i = 1, \ldots, N_k \tag{4}$$

with initial condition $L_0(b_k) = 0$.

In OSLA-BPSK, the destination calculates the cumulative LLR, $L_i(b_k)$, of each bit, and informs the source (via the feedback channel) when to stop sending additional chips related to bit b_k and start transmitting the next bit b_{k+1}. The decision of moving (or advancing) to the next bit is made when the destination has enough confidence on the current bit, i.e., $|L_i(b_k)| \geq L$ is satisfied for a predetermined LLR threshold L. Since we assume perfect feedback for now, the source perfectly receives the bit-advancing decision so it is always synchronized with the destination with one chip delay as shown in Figure 2. The design of a practical feedback scheme is discussed in the later section.

Under this model, the destination can guarantee that the LLR of each received bit is at least L and thus ensure a target bit error rate (BER) performance. Because the number of chips, N_k, for each bit is a random variable that depends on the noise realization, the length (and thus energy) of a symbol for each bit automatically adapts to the noise realization to ensure the target reliability.

Assuming a receiver front-end consisting of an anti-aliasing filter followed by chip based sampling at rate f_s, the noise variance is given by $\sigma_n^2 = \frac{N_0}{2} \cdot f_s$, where N_0 is the noise power spectral density level. Defining $\Delta t = 1/f_s$ as the time duration of one chip, the average length of one symbol is obtained by

$$\overline{T}_{\text{sym}} = \mathbb{E}\{N_k\}\Delta t = \overline{N}\Delta t. \tag{5}$$

The average energy per symbol E_s in OSLA is obtained by $E_s = P \cdot \overline{T}_{\text{sym}}$. And the LLR of each sample is expressed by

$$\begin{aligned}\Delta L_{k,i} &= \frac{2\sqrt{P}(\pm\sqrt{P} + n_{k,i})}{\frac{N_0}{2} f_s} \\ &= \pm\frac{4P\Delta t}{N_0} + \frac{4\sqrt{P}\Delta t \cdot n_{k,i}}{N_0} \qquad (6a) \\ &\sim \mathcal{N}(\pm\frac{4P\Delta t}{N_0}, \frac{8P\Delta t}{N_0}). \qquad (6b)\end{aligned}$$

Therefore, the bit error performance of OSLA-BPSK is fully determined by $\frac{P\Delta t}{N_0}$ and L.

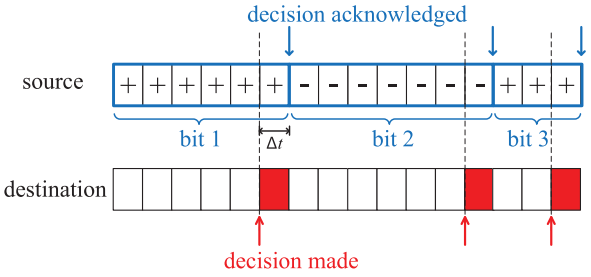


Fig. 2. Forward and feedback signal in OSLA-BPSK on a timeline.

An example of the OSLA-BPSK transmission is illustrated in Figure 2. Notice the Δt delay between the timing of bit-advancing decision from the destination and the acknowledgement at the source. It can be regarded as an *one chip delay*[2] *feedback system* whose delay is much shorter than the average *symbol* length as $\Delta t \ll \overline{T}_{\text{sym}}$ holds in general. A practical case to satisfy this one chip delay feedback assumption including the propagation delay and processing time will be discussed later.

B. Performance Analysis

The performance analysis of OSLA-BPSK is similar to the approach in [17]. A continuous time model in [17] allows analyzing the average stopping time through differential equations. We summarize the analysis result with our parameter definitions here.

As Δt approaches 0, the discrete-time Markov process described in (2) becomes a continuous-time Markov process and one can define a stopping time when the LLR of each bit becomes exactly $\pm L$ for the first time. Without loss of generality, assume $+L$ is observed and thus $\hat{b}_k = 0$ is estimated. Under this circumstance the probability of $b_k = 1$ equals to the error probability P_e, and the relation of P_e and L can be written as

$$\log \frac{1 - P_e}{P_e} = L \quad \text{or} \quad P_e = \frac{1}{e^L + 1}. \tag{7}$$

To analyze the average symbol time $\overline{T}_{\text{sym}}$, first note that the symbol transmission ends when the cumulative LLR reaches the threshold L. So the symbol time can be written as

$$T_{\text{sym}} = \inf\{t \geq 0 : L_t \notin (-L, L)\} \tag{8}$$

where L_t is now continuous in time and denotes the cumulative LLR at time t. Next, define the average symbol time given the initial cumulative LLR $L_0 = l$ as

$$\overline{T}(l) = \mathbb{E}\{T_{\text{sym}}|L_0 = l\}. \tag{9}$$

Following the derivation in [17], the result can be obtained through solving a differential equation given by

$$\overline{T}'(l) + \overline{T}''(l) + \frac{N_0}{4P} = 0. \tag{10}$$

[2]In this work, we consider this one chip delay 'instantaneous'.

With the boundary condition $\overline{T}(L) = \overline{T}(-L) = 0$, the solution of (10) is

$$\overline{T}(l) = -\frac{N_0}{4P}(L+l) + \frac{N_0 L}{2P}\frac{e^{L}-e^{-l}}{e^{L}-e^{-L}} \tag{11}$$

and therefore, the average symbol length with the initial $L_0 = 0$ is

$$\overline{T}_{\text{sym}} = \overline{T}(0) = \frac{N_0 L}{4P}\tanh(L/2). \tag{12}$$

Combining (7) and (12), and using the average energy per bit $E_b = P \cdot \overline{T}_{\text{sym}}$, we have

$$4\frac{E_b}{N_0} = (1-2P_e)\log\frac{1-P_e}{P_e} \xrightarrow{P_e \ll 1} P_e \approx e^{-4\frac{E_b}{N_0}}. \tag{13}$$

Compared to fixed-length BPSK, which has error rate $P_e = Q(\sqrt{2\frac{E_b}{N_0}}) \approx e^{-\frac{E_b}{N_0}}$ (at high SNR), OSLA-BPSK has 6 dB (factor of 4) gain w.r.t. E_b/N_0 as found in [17].

C. OSLA-BPSK Symbol Length and Signal Spectrum

The distribution of the OSLA symbol length T_{sym} can also be obtained analytically. Observe that transmission of one bit in OSLA-BPSK in a continuous time model is in fact a Wiener process with drift [18]. It has non-zero mean $\mu_s = \frac{4P}{N_0}$ (assuming $b_k = 0$ is transmitted without loss of generality) and variance $\sigma_s^2 = \frac{8P}{N_0}$ with boundaries at $+L$ and $-L$. The analysis on stopping time for Wiener process with drift has been shown in [18, p.223] for different boundary conditions. When the BER is small, the probability of stopping at the 'wrong' boundary (which is $-L$ for transmitting $b_k = 0$) is very small, so the probability density function (PDF) of T_{sym} can be approximated by only considering the 'correct' boundary,[3] and is given by

$$f_{T_{\text{sym}}}(t) \approx \frac{L}{\sqrt{2\pi\sigma_s^2 t^3}}\exp\big(-\frac{(L-\mu_s t)^2}{2\sigma_s^2 t}\big). \tag{14}$$

For a discrete-time model, the stopping time can be approximated by accounting for exceeding the exact boundary (due to the coarse time resolution) with a modified boundary value L' [19]. A good model is to use $L' = L + 0.586\sigma_s$ to replace L in (14) [19]. A probability mass function (PMF) to replace the PDF is obtained by using the chip duration Δt for integration.

Applying the symbol time distribution and the pulse shaping of a symbol, the autocorrelation function of the transmitted signal can be obtained following a standard methodology, leading to the analysis of the signal spectrum. The analysis is provided in [20] for OSLA-BPSK with continuous-time model. In this work, we will use empirical (simulation) results to show the OSLA signal spectrum of discrete-time model OSLA (both uncoded and coded cases) for comparison to conventional fixed-length schemes. It is worth pointing out that the spectrum of OSLA signal depends on the symbol length distribution, but not on the chip rate because the transmitted value does not change across chips within the same symbol.

[3]A more complicated exact formula of (14) with positive and negative boundaries is also provided in [18, p.233].

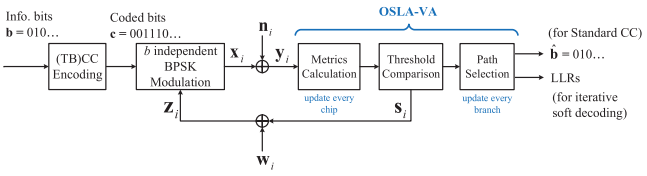


Fig. 3. OSLA for convolutional codes (OSLA-CC).

III. OSLA for Convolutional Codes

A. OSLA With Viterbi Algorithm

We now further enhance the performance of OSLA-BPSK by applying channel coding. Note that directly concatenating an outer channel encoder and decoder in an OSLA-BPSK scheme can only provide the uncoded performance gain because OLSA (when it operates independently from channel coding) would ignore that the likelihood of the current coded bit depends on the likelihood of earlier received coded bits. On the other hand, to fully exploit the available feedback, an entirely new feedback-driven coding scheme needs to be devised. Such a coding scheme specifically designed for OSLA is currently unavailable. Hence, in order to strike a balance between performance and complexity/compatibility, we generalize the OSLA-BPSK to include existing convolutional codes (CC) with a modified Viterbi algorithm (VA) named OSLA-VA, resulting in a (new) coding scheme that we call OSLA-CC.

Figure 3 shows the OSLA-CC system with three main blocks: encoding, modulation, and OSLA-VA. At the encoding stage, information bits $\mathbf{b}$ are first encoded by a (tail-biting) CC into the coded bits $\mathbf{c}$, which is done without utilizing any feedback. On the other hand, the modulation of the coded bits $\mathbf{c}$ takes advantage of feedback and it is intertwined with OSLA-VA to improve reliability. OSLA-VA serves as the OSLA demodulator as well as the Viterbi decoder. For each coded bit transmission, OSLA-VA evaluates path metrics in the trellis as a Viterbi decoder and provides OSLA feedback so that the length of each coded bit is adjusted.

Consider a rate a/b convolutional code, where every a information bits are encoded into b coded bits. The b coded bits that correspond to the kth branch in the trellis are denoted as $\mathbf{c}_k$ or $\{c_{km}, m = 1, \cdots, b\}$ with index m denoting the mth bit in that branch. To transmit the b coded bits concurrently, OSLA uses b independent BPSK channels (each using orthogonal resources other than time such as orthogonal frequencies). The channel model is the same as (1) except for that subscript k is changed to km to denote mth coded bit in kth branch. Note that the length (or number of chips N_{km}) of each coded bit is not fixed but variable.

If N_{km} were deterministic or known, the decoding algorithm would be the same as a conventional VA [21]. The main difference of OSLA-VA is that the decoding process itself is responsible for determining the length of each coded bit N_{km} and ensuring that the decoder has sufficient confidence on the most probable codeword.

In VA, the confidence of a codeword can be represented by the state metric of the corresponding trellis path following

classic definitions in [22]. And the state metric $M_k(s)$ for the state s after the kth branch is defined as the log likelihood (LL), following the update rule with path selection given by

$$M_k(s) = \max_{(s'\to s)\in\mathcal{T}} \{M_{k-1}(s') + \mu_k(\mathbf{c}^{s'\to s})\}. \quad (15)$$

In (15), $\mathcal{T}$ is the set of all valid state transitions, $\mathbf{c}^{s'\to s}$ denotes the coded bits associated with state transition $s' \to s$, and $\mu_k(\mathbf{c}^{s'\to s})$ is the kth branch metric given by

$$\mu_k(\mathbf{c}^{s'\to s}) = \sum_{m=1}^{b} \sum_{i=1}^{N_{km}} \Delta\mathrm{LL}_{km,i}(c_m^{s'\to s}) \quad (16)$$

with $\Delta\mathrm{LL}_{km,i}(c_m) = \log Pr(y_{km,i}|c_m)$ denoting the chip LL of the coded bit c_m calculated by the received chip $y_{km,i}$.

To determine N_{km} for each coded bit *during* the decoding process, the decoder is designed to observe additional received chips (increasing N_{km}) for the same coded bit until a predetermined criterion is met, which indicates the reliability of the most probable codeword so far. It is important to note that although the length of a coded bit N_{km} only affects the reliability of the corresponding coded bit, the criterion should be a function of all coded bit candidates $\mathbf{c}^{(j)}, j = 1, \cdots, 2^b$ because coded bits are not independent.

To quantify the confidence of coded bits and ensure the reliability of the most probable codeword at the same time, the confidence of $\mathbf{c}^{(j)}$ is defined as the largest state metric for a candidate $\mathbf{c}^{(j)}$, written as

$$W_k(\mathbf{c}^{(j)}) = \max_{s,s':\mathbf{c}^{s'\to s}=\mathbf{c}^{(j)}} \{M_{k-1}(s') + \mu_k(\mathbf{c}^{s'\to s})\}. \quad (17)$$

Notice that (17) involves the length of each coded bits N_{km} via $\mu_k(\mathbf{c}^{s'\to s})$ that is increasing as the decoder receives additional chips for each coded bit. The decoder has enough confidence on the largest $W_k(\mathbf{c}^{(j)})$ (corresponding to the most probable coded bits) when the metric is sufficiently larger than the second largest. Thus, one possible (but naive) bit-advancing criterion to stop increasing N_{km} for the kth state transition is defined as

$$\max_j W_k(\mathbf{c}^{(j)}) - \max_j{}_2 W_k(\mathbf{c}^{(j)}) \geq L \quad (18)$$

where $\max_2(\cdot)$ denotes the second largest value.

The naive criterion (18) makes the bit-advancing decision based on the set of coded bits $\mathbf{c}^{(j)}$. Thus it forces the same length N_{km} for all bit indices m for the kth branch. When this method is adopted, $W_k(\mathbf{c}^{(j)})$ can be sequentially updated with time index i as in

$$W_{k,i}(\mathbf{c}^{(j)}) = W_{k,i-1}(\mathbf{c}^{(j)}) + \sum_{m=1}^{b} \Delta\mathrm{LL}_{km,i}(c_m^{(j)}) \quad (19)$$

with an initial condition

$$W_{k,0}(\mathbf{c}^{(j)}) = \max_{s':\mathbf{c}^{(j)}\in\mathcal{C}_{s'}} \{M_{k-1}(s')\} \quad (20)$$

where $\mathcal{C}_{s'}$ denotes the set of all possible coded bits with associated state transition starting from s'. The decoder evaluates (18) using $W_{k,i}(\mathbf{c}^{(j)})$ instead of $W_k(\mathbf{c}^{(j)})$ as time index i increases until the criterion is satisfied. At that point, N_{km} is set to the current time index i for all m.

Criterion (18) is highly inefficient because the coded bits for $\max_j W_k(\mathbf{c}^{(j)})$ and those for $\max_j{}_2 W_k(\mathbf{c}^{(j)})$ are often identical except for one. Determining the length N_{km} for all m's based on a single non-identical coded bit can result in unnecessarily longer lengths for other common coded bits without improving the reliability because the excessive length (i.e., energy) for those equally contributes to the $\max$ and $\max_2$ terms.

To resolve this issue, we propose an *asynchronous multi-channel bit-advancing scheme* where each coded bit sequence with a particular index m is transmitted using a dedicated orthogonal BPSK channel (e.g., orthogonal frequency carrier) to allow separate asynchronous bit-advancing decisions for each channel. For that, we set another individual and asynchronous bit-advancing criterion for each bit/channel in addition to the criterion (18). The individual criterion has a form very similar to (18) except that the inspected metrics only account for one bit. Following the same strategy, the metric (19) is separated for each coded bit in $\mathbf{c}^{(j)} = \{c_1^{(j)}, \ldots, c_b^{(j)}\}$ such as

$$W_{km,i}(c) = W_{km,i-1}(c) + \Delta\mathrm{LL}_{km,i}(c) \quad (21)$$

where the coded bit c is either 0 or 1, with an initial condition $W_{km,0}(c) = \max_{j:c_m^{(j)}=c} W_{k,0}(\mathbf{c}^{(j)})$.

The decoder with this asynchronous bit-advancing scheme evaluates the first criterion (18) using (19) as well as the individual criterion version using $W_{km,i}(c)$ for each channel. It determines the length N_{km} for a particular coded bit when *either* of the two criteria is satisfied and advances the channel to the next coded bit without waiting for the decision on the other channels/bits. When the mth bit is advanced with length N_{km}, the calculation of (19) stops at $i = N_{km}$ and (16) only sums up to $i = N_{km}$ for that particular bit.

When the length N_{km} is determined for all coded bits, the state metrics (15) based on branch metrics (16) are calculated for the next branch decoding. Some BPSK channels may advance to the next branch index $k+1$ before the completion of the state metric updates for the other coded bits for the kth branch. Those channels can start *pre-calculating* the second term $\Delta\mathrm{LL}_{k+1,m,i}(c_m^{(j)})$ in (21) for the $(k+1)$-th branch.

The algorithm pseudo-code of OSLA-VA is described in Algorithm 25.

B. OSLA With Iterative Decoding of TBCC and Turbo Codes

State-of-the-art codes often involve iterative decoding that uses LLR as the decoder (soft) input. To extend OSLA to such coding schemes, we propose to concatenate an iterative decoder to trellis coded OSLA. Specifically, we apply OSLA to TBCC (or turbo) codes, which turns out to be particularly effective for relatively short (or long) code lengths.

The OSLA-VA scheme discussed in Section II-A is designed to identify the most probable trellis path with the largest metric as the decoded bits after the last path selection at the end of the trellis. As the length N_{km} of each coded bit is determined during the OSLA-VA operation, the LLR for each coded bit c_{km} can be obtained as a byproduct using the

Algorithm 1 OSLA-VA

input : Trellis $\mathcal{T}$

// Initialization

$M_0(s) = 0$ for every state s in $\mathcal{T}$

for *each branch k* **do**

  // Reset metrics

  for *each coded bits sequence j* **do**

    $W_k(\mathbf{c}^{(j)}) = \max_{s,s':\mathbf{c}^{s'\to s}=\mathbf{c}^{(j)}} \{M_{k-1}(s') + \mu_k(\mathbf{c}^{s'\to s})\}$

  for *each dimension m* **do**

    $W_{km}(c) = \max_{j:c_m^{(j)}=c} W_k(\mathbf{c}^{(j)})$

  // Keep receiving chips

  advanceFlag $= [0, 0, \cdots, 0]$

  while *not all advanceFlag* **do**

    for *each dimension m* **do**

      if *advanceFlag[m]* **then**

        store received chip in a buffer for future use

      else

        calculate $\Delta\mathrm{LL}_{km}(0)$ and $\Delta\mathrm{LL}_{km}(1)$

    // Check branch

    for *each coded bits sequence j* **do**

      $\Delta\mathrm{LL}_k(\mathbf{c}^{(j)}) = \sum_{m=1}^{b} \Delta\mathrm{LL}_{km}(c_m^{(j)})$

      $\mu_k(\mathbf{c}^{(j)}) \leftarrow \mu_k(\mathbf{c}^{(j)}) + \Delta\mathrm{LL}_k(\mathbf{c}^{(j)})$

      $W_k(\mathbf{c}^{(j)}) \leftarrow W_k(\mathbf{c}^{(j)}) + \Delta\mathrm{LL}_k(\mathbf{c}^{(j)})$

    if $\max W_k(\mathbf{c}^{(j)}) - \max_2 W_k(\mathbf{c}^{(j)}) \geq L$ **then**

      advanceFlag $= [1, 1, \cdots, 1]$

    // Check each dimension

    for *each dimension m* **do**

      $W_{km}(0/1) \leftarrow W_{km}(0/1) + \Delta\mathrm{LL}_{km}(0/1)$

      if $|W_{km}(0) - W_{km}(1)| \geq L$ **then**

        advanceFlag$[m] = 1$

  // Update branch

  for *each state s* **do**

    $M_k(s) = \max_{(s'\to s)\in\mathcal{T}} \{M_{k-1}(s') + \mu_k(\mathbf{c}^{s'\to s})\}$

following equation:

$$\mathrm{LLR}(c_{km}) = \sum_{i=1}^{N_{km}} \frac{2\sqrt{P}y_{km,i}}{\sigma_n^2}. \tag{22}$$

These LLRs for all coded bits can be fed into a conventional soft-input decoder for TBCC or turbo codes to decode the original information bits. The TBCC encoder and decoder are the same as in a fixed length scheme as they operate independently from the underlying (OSLA) modulation/demodulation process.

TBCC is one of the state-of-the-art short codes [23]. It is widely used for short message communications including the LTE control channel [24]. Unlike a standard convolutional code, the trellis of a TBCC starts with the state determined by the tail bits as the name indicates. Since the starting state is not pre-determined, decoding needs to start with equal metrics for all states treating them as a potential valid state. In our proposed scheme, LLRs (22) obtained from OSLA-VA are fed into a wrap-around Viterbi algorithm (WAVA) decoder [25] for additional iterative TBCC decoding. In WAVA, a standard VA is performed (using LLRs from OSLA-VA as the soft-input) for each iteration, which is repeated until the stopping criterion is met for the metrics of all tail-biting paths. WAVA gaurantees that the output tail-biting path is the optimal solution if the stopping criterion is met before the maximum iteration number is reached. In this scheme, OSLA-VA can be regarded as the first iteration of WAVA (or part of the demodulation process). The proposed combination of OSLA and TBCC is termed OSLA-TBCC.

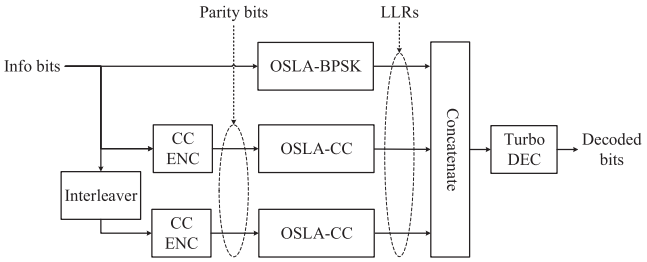


Fig. 4. OSLA with turbo codes.

Turbo codes, on the other hand, are well known for their excellent performance for long blocklengths. A widely used turbo code adopted in the LTE standard [24] is a rate $1/3$ code that uses two recursive systematic convolutional (RSC) codes with an interleaver. It consists of four parts: a payload sub-block, two parity sub-blocks, and 12 tail bits. As the payload sub-block is identical to the (uncoded) information bits, it can be regarded as a systematic code. The second and third parity sub-blocks are the output bits of the two RSC codes whose constraint length is 4. As shown in Figure 4, the second RSC encoder takes the interleaved information bits as the input. The 12 tail bits are used to terminate the RSC trellis paths with zero states.

Application of OSLA to turbo codes is straight-forward. Since the payload sub-block is identical to the uncoded information bits, OSLA-BPSK can be directly applied. The second and third parity sub-blocks are the outputs of RSC codes. Therefore OSLA-VA is used for the transmission of these. Finally, the tail bits are transmitted with OSLA-BPSK. These four parts can be transmitted either sequentially or simultaneously with orthogonal resources (e.g., orthogonal sub-carriers) as independent messages. After completing OSLA transmissions of all four parts, the LLRs are concatenated together and sent to a conventional turbo decoder for iterative decoding. Figure 4 shows the structure of combining the proposed OSLA scheme with turbo codes (tail bits are omitted for simplicity). This scheme is termed OSLA-turbo.

C. Complexity of OSLA

Instantaneous feedback implies that the latency to compute and send the feedback decision to the source/transmitter is negligible compared to the average symbol length. The proposed OSLA assumes a chip-based feedback scheme where the decision must be made for each chip in real-time. Complexity of OSLA feedback decision, therefore, is a main concern for applying OSLA to practical systems. Note that the complexity of OSLA-BPSK is significantly lower than that of OSLA-VA. The additional iterative decoding step such as WAVA involved in OSLA-TBCC or OSLA-turbo is irrelevant to real-time feedback decision computation as it can be performed *after*

receiving all coded bits via OSLA-VA or OSLA-BPSK. Therefore, in this section, we focus on analyzing the complexity of OSLA-VA to assess the feasibility of OSLA in real-time systems.

Computation of OSLA-VA can be categorized into two parts: *chip update* and *branch update*. After every chip update, the destination/receiver updates the metric and decides whether to advance to the next bit/symbol or not. When every coded bit of the k-th branch has been advanced, branch metrics are updated by (15), followed by add-compare-select (ACS) operations for path selection.

Chip update involves computing the metrics W_k in (19) and W_{km} in (21), and also checking if any advancing criterion is satisfied. For every chip, we first obtain $\Delta\text{LL}_{km,i}(c)$ for all m and $c \in \{0, 1\}$. For BPSK signaling $x = \pm\sqrt{P}$ in the AWGN channel, $\Delta\text{LL}_{km,i}(c) = \frac{y_{km,i}x}{\sigma_n^2} + c_0$ for a constant c_0 holds. Calculating its scaled version (by a factor of a given constant σ_n^2) can be further simplified without explicit computations by ignoring the constant term and postponing the sign operation in $y_{km,i}x$ to a later add operation. These values are then used for updating (19) and (21), which involve 1 and b additions, respectively. A total of $2b+b2^b$ additions are needed to evaluate all possible (scaled versions of) W_k and W_{km}. The remaining steps involve finding the two largest metrics and comparing their difference to a fixed (and scaled) threshold value as in (18) and the criteria for individual coded bits. These steps further require at most $2^{b+1} + b + 1$ comparisons and $b + 1$ additions. Therefore, the total number of operations required in *chip update* is $(b + 2)2^b + 4b + 2$.

Branch update is performed after all branch metrics are available, i.e., N_{km} has been determined for all m's of the k-th branch. Given N_{km}, branch update is identical to that of the conventional Viterbi algorithm, where an ACS is executed for every state. The branch metric $\mu_k(\mathbf{c})$ in (15) can be obtained by the difference between $W_{k,0}(\mathbf{c})$ and $W_{k,N}(\mathbf{c})$, which requires b additions. An ACS requires 2 additions and 1 compare-select (a combined single operation), therefore there are a total of $2^{K-1} \cdot 3 + b$ operations for *branch update*, where K denotes the constraint length of the code. The big-O complexity representation for OSLA-VA is $O(k(\overline{N}b2^b + 2^{K-1}))$, where the first term is for *chip update* and the second term is from the ACS branch update in a Viterbi decoder.

Since *chip update* happens at a much higher rate than *branch update*, the chip update complexity is more critical for real-time OSLA-VA transmission. Notice that the number of operations involved in *chip update* does not scale with the number of states of the trellis but it only relates to the number of output bits per branch b, which is usually small. It is possible to store the pre-calculated $\Delta\text{LL}_{km,i}$ in a memory before they are needed to update the metrics once the initial conditions of (20) are ready upon the completion of previous branch update. Although the number of operations involved in *branch update* exponentially grows with K, fully parallel ACS computing to update all branch metrics simultaneously with low latency using 2^{K-1} parallel hardware instances in modern digital ICs is certainly feasible [26], [27] for a practical K (e.g., $K = 12$).

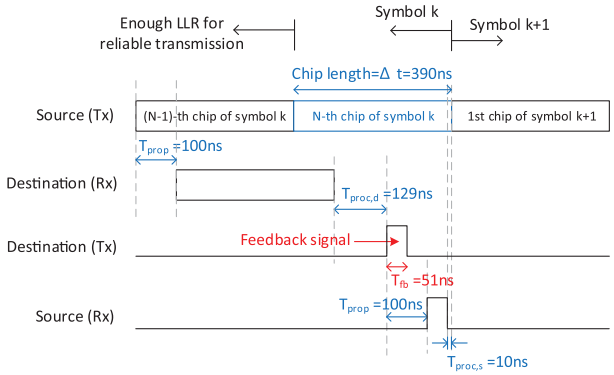


Fig. 5. A practical OSLA system example timeline to attain $\leq$1-chip feedback delay including propagation and processing delays.

D. Feasibility of Feedback Within One-Chip Delay

For a reasonable example configuration (that is used to evaluate OSLA-TBCC's performance in the later section) with $b = 2$, $K = 11$, and $\overline{N} = 20$, the average number of operations per chip is 128.5 according to the analysis in the previous subsection. We argue that this real-time computation complexity is practically feasible in modern digital ICs where a large number of parallel computation units are instantiated (e.g., Xilinx UltraScale XCVU440 FPGA [28] has 2,880 DSP units). An example URLLC application with 1 ms latency and 32-byte packet (after encoding) using the aforementioned configuration has the chip length of $\Delta t = 0.39\mu$s given $\overline{N} = 20$. For a digital IC that can run at $\geq$1 GHz, this chip duration corresponds to $\geq$390 cycles, which is sufficient to perform (on average) 128.5 operations which are mostly computed in parallel using dedicated computing hardware instances.

Besides the computation at the destination, a practical system also needs to consider propagation delay and processing time at the source. An example of a practical feedback mechanism with propagation delay and processing time is illustrated in Figure 5. Assume the distance between the source and the destination is 30m, this introduces an one-way propagation delay of 100ns. Also assume a worst-case serialized implementation with a 1GHz processor for the processing at the destination thus 129ns processing time, and assume the processing time at the source (estimating the binary decision) takes 10 cycles, or 10ns. Note that the additional delay is the combination of two-way propagation delay ($2T_{\text{prop}}$) and the processing time at both sides ($T_{\text{proc,d}}$ and $T_{\text{proc,s}}$), which is 339 ns and is less than one chip. By using a shorter feedback signal (T_{fb}, details introduced in the later section), the delay due to feedback is kept as one chip. With this example, we argue that a practical OLSA system with $\leq$ 1-chip feedback latency (Fig. 2) is feasible as assumed throughout this work.

IV. Feedback Signaling in OSLA

A. Pulse Feedback Signal

OSLA uses a feedback channel to inform the source when to advance to the next symbol/bit. Synchronizing the source and destination on the chip and symbol indices is the main goal

of the feedback signaling. The only information conveyed in the feedback channel is the *timing* of the symbol-advancing decision.

We propose a pulse position/timing based feedback scheme to transmit a pulse when the destination makes a symbol-advancing decision. The feedback channel remains idle (i.e., no transmission) when the destination expects more chips/samples for the same symbol. Thus the feedback signal s_i for chip index i is given by

$$s_i = \begin{cases} \sqrt{P_{\text{fb}}} & \text{if symbol-advancing decision made at } i-1 \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

where P_{fb} is the power of the feedback pulse.[4] Note that each forward channel using an orthogonal resource requires a dedicated feedback channel as shown in Figure 1 and 3.

Under this design, the probability of feedback detection error is

$$P_d = Q\left(\sqrt{\frac{P_{\text{fb}}}{4\sigma_w^2}}\right) \tag{24}$$

where σ_w^2 is the noise variance in the feedback channel of each chip/sample. Note that a single detection error can destroy the synchronization between the source and destination, causing a catastrophic failure of the transmission. With this pulse based signaling, the feedback error rate will be $1-\mathbb{E}\{(1-P_d)^{N_{\text{total}}}\}$, where N_{total} is the random number of total chips on the feedback channel for the entire codeword transmission.

Note that this feedback scheme allocates the transmit power only for the time slot where symbol advancing occurs. The downside of this scheme is the relatively wide bandwidth usage which is inversely proportional to the chip duration Δt, not the average symbol duration.

B. Enhanced Synchronization With HMM

The source uses the received feedback signal to estimate the symbol-advancing decision made by the destination at every chip. However, the aforementioned naive scheme does not consider the fact that each chip has a different probability to advance to the next symbol. For example, the first chip of a symbol is very unlikely to be the last chip advancing to the next symbol. This chip-dependent probability can be taken into consideration to improve the reliability of the feedback estimation.

OSLA transmission can be viewed as a state transition process where each state corresponds to a unique symbol. The system either stays in the same state (symbol) or advances to the next state, whereas the transition probability changes with the chip index (or the elapsed time in a state).

We propose a 2-D state structure for state estimation, where one dimension is the symbol index and the other dimension is the number of chips spent on a symbol (i.e., chip index of a symbol) as illustrated in Figure 6. A state denoted as $S_{i,j}$ indicates that the system is transmitting the j-th chip of the i-th symbol. For the next chip, the state $S_{i,j}$ can transition to one of only two possible next states $S_{i+1,1}$ or $S_{i,j+1}$ depending on symbol advancing or not, respectively. We assume the system is a hidden Markov model (HMM), and the transition probability only depends on the current state, not on the past history. This assumption holds for uncoded OSLA-BPSK, but it is not necessarily true for trellis coded OSLA. However, we make this simplifying assumption as it greatly reduces the complexity of the state estimation process at the source, without affecting significantly the practical performance of the proposed scheme.

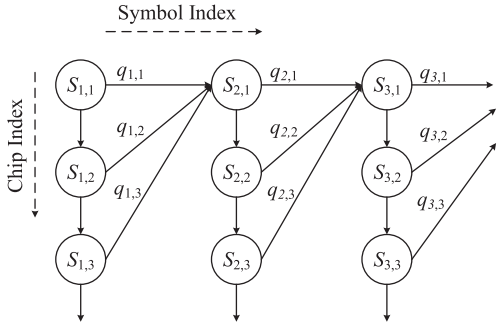


Fig. 6. 2-D state transition model for enhanced synchronization with HMM.

The source uses a classic forward algorithm [29] for HMM to estimate the likelihood of each state given the received feedback signal z_i and the knowledge of symbol advancing probability $q_{i,j} = Pr(S(t+1) = S_{i+1,1} | S(t) = S_{i,j})$, where t is the transmission time index in chip units. For each chip, the forward algorithm uses current beliefs of all states at time t denoted as $\alpha_t(S)$ as well as the received signal z_{t+1} to update the belief of each state based on the following three steps:

1) Calculate the sum of probabilities of transitions into state $S_{i,j}$ by

$$F(S_{i,j}) = \begin{cases} \sum_{k=1}^{\infty} q_{i-1,k} \cdot \alpha_t(S_{i-1,k}) & j = 1 \\ (1 - q_{i,j-1}) \cdot \alpha_t(S_{i,j-1}) & j \neq 1. \end{cases} \tag{25}$$

2) Calculate the emission probability of state $S_{i,j}$ by

$$Pr(z_{t+1}|S_{i,j}) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma_w^2}} \exp(-\frac{(z_{t+1} - \sqrt{P_{\text{fb}}})^2}{2\sigma_w^2}) & j = 1 \\ \frac{1}{\sqrt{2\pi\sigma_w^2}} \exp(-\frac{z_{t+1}^2}{2\sigma_w^2}) & j \neq 1. \end{cases} \tag{26}$$

3) Update the belief for each state by

$$\alpha_{t+1}(S_{i,j}) = F(S_{i,j}) \cdot Pr(z_{t+1}|S_{i,j}). \tag{27}$$

The initial beliefs are set as $\alpha_1(S_{1,1}) = 1$ and $\alpha_1(S_{i,j}) = 0$ for all other states. For each chip, the source updates the belief of each state, and selects the maximum one as the

[4]Here we assume the feedback pulse length is equal to the chip length. A shorter pulse ($T_{\text{fb}} < \Delta t$) with higher power (fix the energy) can be used as suggested in the previous section to compensate for delays from propagation and processing.

most likely state to transmit the corresponding chip/symbol indicated by that state. Note that synchronization between the source and destination is maintained without performance degradation as long as the symbol index i is correct even if the estimated state $S_{i,j}$ is not (i.e., chip index j is incorrect). Therefore, this scheme is robust to temporary chip index mismatches between the source and destination.

The number of states with non-zero beliefs in the HMM forward algorithm grows with the time index t to $1+\frac{t(t-1)}{2}$. This can potentially cause computation complexity issues for a relatively long packet. However, it is observed that the number of states with non-negligible probability (e.g., > 0.001) stays very small (e.g., < 5) for a feedback channel that has sufficiently high SNR (e.g., $>$ 2dB). Thus, we can prune the vast majority of states to limit the number of states to evaluate.

The forward algorithm assumes that the transition probabilities for each state pair, $q_{i,j}$, are known. While it is possible to obtain them analytically for uncoded OSLA-BPSK, the analysis is difficult for trellis coded OSLA. Thus, in this work, we empirically obtain these probabilities via Monte Carlo simulations. Note that these probabilities depend only on the forward channel SNR but not the feedback channel SNR.

The main advantage of using HMM to track the probability of each state is that one single miss detection of the feedback signal does not necessarily destroy the synchronization for the entire packet because the tracking may correct the error for the subsequent symbols in the packet.

C. Trade-off Between Asynchronous and Synchronous Advancing Schemes

In Section III-A, we introduced an asynchronous symbol/bit-advancing scheme where b coded bits that determine one state transition in the Viterbi trellis are transmitted using b dedicated orthogonal channels so that each can be advanced to the next symbol asynchronously. Such an asynchronous scheme outperforms the synchronous version (where b-bits advance at the same time to be synchronized with Viterbi trellis state transition) when the perfect feedback reliability is assumed. For a realistic feedback channel with a finite SNR, however, there is a potential advantage to use a synchronous symbol-advancing scheme for more reliable feedback.

Since all the channels advance symbols at the same time in the synchronous scheme, only one advancing schedule (and thus one HMM) needs to be maintained. Therefore, feedback signaling power can be concentrated into a single feedback channel (as opposed to splitting the power into b separate feedback channels for asynchronous feedback for each), improving the SNR and reliability of the feedback signal. Moreover, the bandwidth of the feedback signal is also reduced thanks to a fewer number of feedback channels. The synchronous advancing scheme reduces the feedback transmit power and bandwidth by a factor of b times in the feedback channel for the same feedback error rate.

This implies that there exists a trade-off between the asynchronous and synchronous symbol-advancing schemes. The former has better forward channel reliability, whereas the latter enables more reliable feedback given the same feedback SNR. As evaluated in Section V, a better feedback scheme can be chosen based on the SNR realization of the forward and feedback channel.

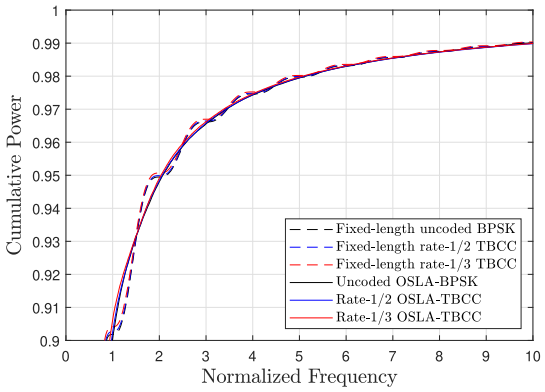


Fig. 7. Occupied bandwidth.

V. Evaluation

We evaluate the performance of OSLA with Monte Carlo simulations. In all simulations, the average symbol length of OSLA is controlled by setting a proper threshold L for symbol advancing criteria evaluation to match the symbol length of a fixed-length scheme. For a fair error rate comparison, all schemes are evaluated with the same E_b/N_0 and spectral efficiency.

Due to the variable symbol length, analyzing the spectral efficiency of OSLA is non-trivial, thus we only show results based on numerical analysis of the occupied bandwidth of OSLA compared to that of a fixed-length scheme. The occupied bandwidth is defined as the minimum frequency range that contains a certain percentage (e.g., 95%) of the total power. Figure 7 shows the cumulative power over the normalized frequency (with respect to the (coded) bit rate) of fixed-length and OSLA schemes. Rectangular pulse-shaping is assumed for both schemes. Chip duration Δt in OSLA is set to be $1/(10b)$ of the average symbol length for a rate $1/b$ coding ($b = 1$ for the uncoded case). Compared to a fixed-length scheme, it is observed in Fig. 7 that OSLA occupies similar cumulative power profile and bandwidth. Note that in practical systems, non-rectangular (e.g., root-raised-cosine) pulse shaping can be applied to a fixed-length scheme for reducing the occupied bandwidth. Similarly, head/tail ramping up/down can be applied to OSLA, but it is non-trivial and is left as a future work.

A. OSLA-BPSK

Figure 8 shows the BER performance of uncoded OSLA-BPSK for various $\Delta t/\overline{T}_{\text{sym}}$ settings. The system approaches the continuous model when $\Delta t/\overline{T}_{\text{sym}}$ is smaller, and the simulation result is well aligned with the analysis (13). Performance degradation occurs when $\Delta t/\overline{T}_{\text{sym}}$ is larger with longer feedback delay of Δt that results in wasted transmit energy or symbol length. The performance gap between OSLA-BPSK

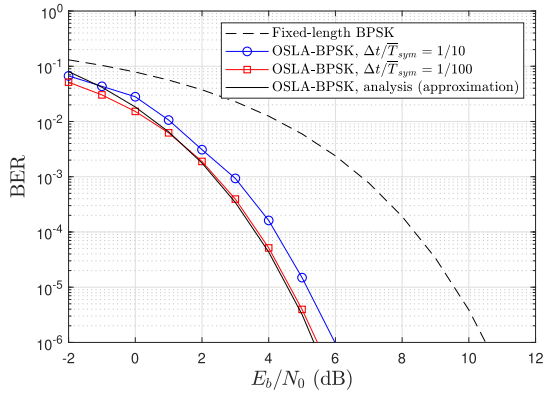


Fig. 8. OSLA-BPSK BER performance and analysis.

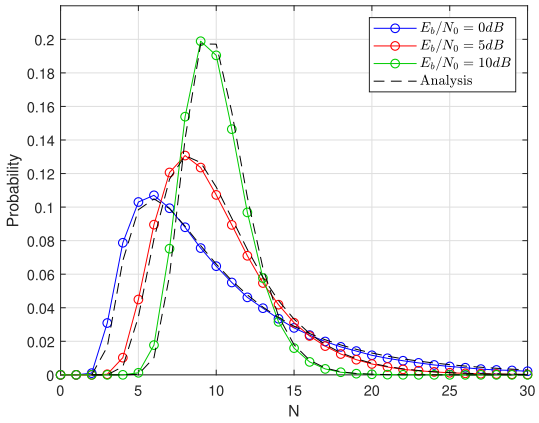


Fig. 9. Distribution of $N = T_{\text{sym}}/\Delta t$ in OSLA-BPSK.

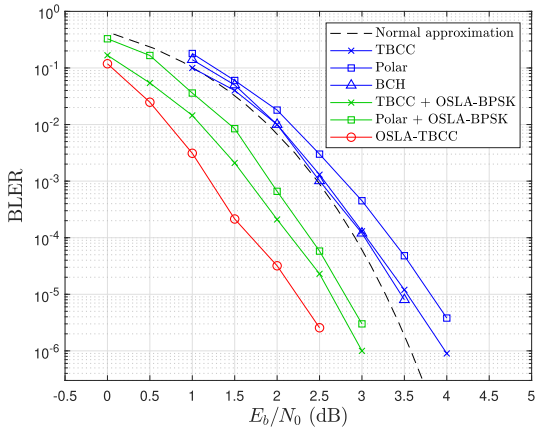


Fig. 10. OSLA-TBCC BLER performance comparison with non-feedback schemes.

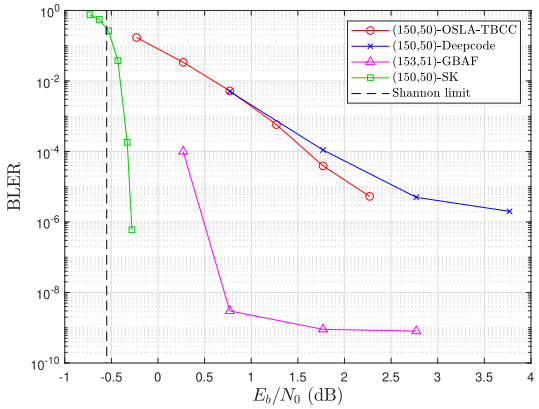


Fig. 11. OSLA-TBCC BLER performance comparison with feedback-based schemes.

and fixed-length BPSK (both uncoded) is approximately 6 dB at high SNR as expected from the analysis (13).

Figure 9 plots the distribution of $N = T_{\text{sym}}/\Delta t$, the number of chips per symbol, of OSLA-BPSK for different SNR scenarios. The expected number of chips per symbol $\overline{N} = \mathbb{E}\{N\}$ is set (by controlling L) to 10 in the simulation. As the figure shows, the analysis (14) matches the simulation results very well. Notice the distribution of N is dependent on the SNR for a given $\overline{N}$ (= 10 in Fig. 9). For a higher SNR, the variance of N is smaller, and its distribution can be approximated by a Gaussian distribution.

B. Trellis Coded OSLA

Figure 10 shows the block error rate (BLER) performance evaluation of a rate-$1/2$ (128,64) OSLA-TBCC compared to state-of-the-art non-feedback short codes. BLER performance a rate-$1/3$ (150,50) OSLA-TBCC compared to other feedback-based schemes is shown in Figure 11. All schemes that have the identical coding rate also have the same spectral efficiency as shown in Figure 7. The constraint length of TBCC is set to 11 and Δt in OSLA is set to $1/(10b)$ of the average symbol length for the coding rate of $1/b$. Smaller Δt is avoided for faster simulation although it would enhance the error performance. Noiseless feedback channel is assumed in these plots.

It is observed that (128,64)-OSLA-TBCC significantly outperforms state-of-the-art non-feedback polar, TBCC and BCH codes [30] by about 1.5dB. The normal approximation [31] of an (128,64) non-feedback code in binary input AWGN channel is also shown. OSLA-TBCC can surpass the normal approximation curve thanks to the utilization of feedback.

Figure 11 shows that (150,50)-OSLA-TBCC can outperform Deepcode [11], a deep learning-based feedback scheme that has the same spectral efficiency, especially in the high SNR region. Another state-of-the-art deep learning-based scheme called generalized block attention (GBAF) code [12] improves the performance and outperforms OSLA-TBCC in the noiseless feedback case when the message length is set to 50 (or 51) bits. The BLER of Schalkwijk-Kailath (SK) scheme [32] is also shown in the same figure. Although SK scheme is better than OSLA-TBCC / GBAF and closer to Shannon limit in a noiseless feedback channel, it is practically infeasible because of its noiseless feedback assumption and extreme sensitivity to the numerical precision. On the contrary, OSLA-TBCC and Deepcode, and GBAF are more practical schemes as they can

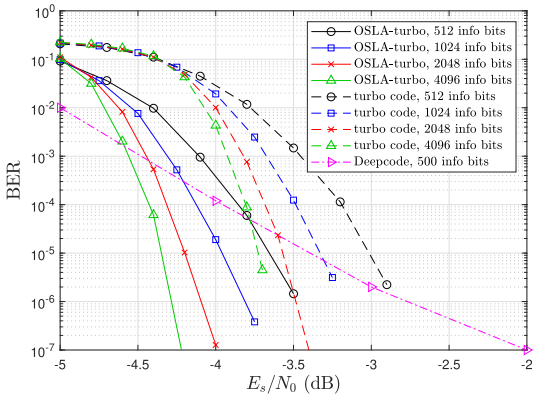


Fig. 12. OSLA-turbo BER performance comparison.

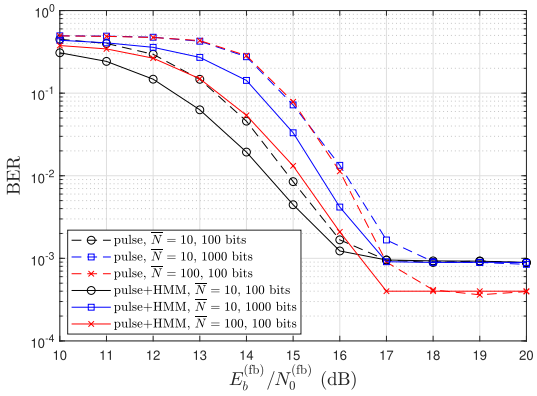


Fig. 13. OSLA-BPSK with noisy feedback.

tolerate noisy feedback and do not suffer from the arithmetic imprecision issue.

Figure 12 shows the BER performance of OSLA-turbo with different codeword lengths under noiseless feedback. The turbo code settings in the simulation follow the LTE standard [24], and the number of decoder iteration cycles is set to 5. OSLA-turbo outperforms fixed-length non-feedback turbo codes with 0.5∼0.7 dB. The gap between OSLA and fixed-length turbo codes slightly decreases with longer codeword lengths, showing that the OSLA feedback scheme is more advantageous for shorter codeword lengths. This behavior is expected as turbo coding is asymptotically capacity achieving. OSLA-turbo outperforms Deepcode for BER $< 10^{-4}$ for 500 information bits (other longer codeword settings are not available in [11]). Another deep learning based scheme GBAF [12] is excluded from the comparison in Figure 12 because it does not provide the codeword length scalability beyond the demonstrated short information length of 51 bits. One main drawback of Deepcode and GBAF is its limited scalability. Unlike OSLA, Deepcode's BER slope remains almost the same regardless of the codeword length. Moreover, they need a different neural network model for each particular codeword length and rate setting. To achieve satisfactory performance for long codewords, the authors of [11] propose to use an outer turbo code with an inner Deepcode. However, it inevitably lowers the coding rate ($1/9$ in [11]). On the contrary, OSLA can easily scale the codeword length without changing any code structure.

C. OSLA With Noisy Feedback

Figure 13 shows the BER performance of OSLA-BPSK with noisy feedback in different $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$ settings, where $E_b^{(\mathrm{fb})}$ is the feedback energy per forward-channel information bit and $N_0^{(\mathrm{fb})}$ is the noise power spectral density of the feedback channel. The forward channel E_b/N_0 is set to 3dB. With the proposed HMM-based synchronization, the required $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$ for feedback is relaxed by about 1dB compared to a naive scheme without an HMM for the same (forward channel) BER performance. The figure also shows that both $\overline{N}$ (expected number of chips per symbol) and packet length affect the required $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$. For the naive feedback scheme without an HMM, the total number of chips $\overline{N}_{\mathrm{total}} = \overline{N} \cdot$ (number of information bits) of a packet governs the feedback robustness, whereas in the HMM-based feedback scheme, the number of information bits plays a more important role than $\overline{N}$. Note that BER loss is negligible when the feedback channel SNR is sufficiently high ($E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})} \geq 17$ dB) as the synchronization error probability is substantially lower than the forward channel BER. It is also observed that the number of states with non-negligible probabilities (< 0.0001) in the HMM is always less than 3 at 17 dB SNR. In the high feedback SNR regime, the attainable forward channel BER is lower for a larger $\overline{N}$ (shorter Δt) as shown in Fig. 8.

The BLER performance with error correction coding in noisy feedback channels is shown in Figure 14 with respect to $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$ for various forward channel E_b/N_0 settings (1, 1.77 and 3dB). Note that $E_b^{(\mathrm{fb})} = (b/a) \cdot P_{\mathrm{fb}} \Delta t$ holds for the coding rate of a/b. The same $(150, 50)$ setting as in the previous simulation is used for comparison. First, observe that the synchronous advancing scheme for OSLA-TBCC has about 4.7 dB feedback-SNR gain over the asynchronous advancing scheme thanks to the power saving from using only one feedback channel instead of three ($= b$). However, the synchronous scheme has a worse/higher BLER floor compared to the asynchronous advancing scheme for sufficiently high $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$ conditions, exhibiting the trade-off between the forward communication and feedback reliability. OSLA-TBCC, regardless of synchronous and asynchronous advancing schemes, shows more reliable feedback in terms of $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$ compared to Deepcode and Modulo-SK scheme [7], which is a variant of the SK scheme. Although Modulo-SK can achieve lower BLER for sufficiently high $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$, it is not scalable to significantly longer codewords because the required numerical precision for the forward channel grows with the length of the codeword (the length of >50 bits is impractical and difficult to simulate). It also requires a specific setting for each combination of forward and feedback SNRs. On the other hand, OSLA-TBCC does not suffer from the same forward channel numerical precision issue for longer codewords (as shown in Fig. 12) and it does not rely on the knowledge of feedback SNR. Note that the state-of-the-art deep learning-based scheme GBAF [12] is excluded from the comparison

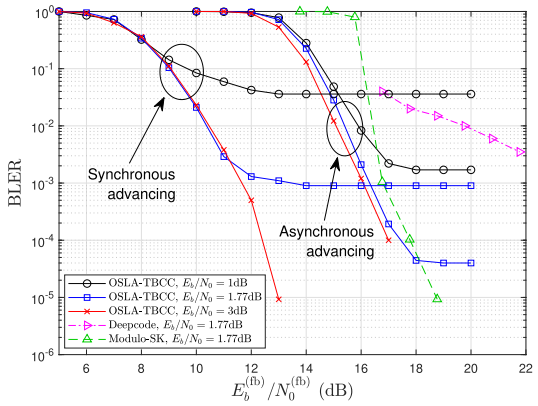


Fig. 14. OSLA-TBCC with noisy feedback.

TABLE I
COMPARISON BETWEEN OLSA AND OTHER FEEDBACK-BASED SCHEMES

Scheme	Error Slope	Feedback Robustness	Blocklength Scalability	Complexity Real-time Feasibility	Arithmetic Imprecision Tolerance	Flexibility*
Non-Feedback	Medium	N/A	Excellent	Good	Good	Excellent
(Modulo-)SK	Excellent	Good**	Poor	Excellent	Poor	Excellent
Burnashev	Excellent	Poor	Poor	Good	Medium	Good
DeepCode	Good	Good	Medium	Poor	Good	Poor
GBAF	Excellent	Good	Medium	Poor	Good	Poor
OSLA	**Good**	**Excellent**	**Excellent**	**Good**	**Good**	**Excellent**

* To adjust code rate or adapt to different SNR conditions.
** When Modulo-SK is used.

in Figure 14 because it reports the performance only when $E_b^{(\mathrm{fb})}/N_0^{(\mathrm{fb})}$ is 21.8 dB, which is substantially higher than the requirement of OSLA-TBCC.

VI. DISCUSSION AND FUTURE DIRECTIONS

TABLE I summarizes the comparison of different feedback-based schemes in various aspects. Note that OSLA possesses distinctive advantages in blocklength scalability and code rate flexibility. Considering other relative strengths such as relaxed computation precision requirements and robustness of the feedback, our proposed OSLA has a great potential as a practical communication scheme. However, it has certain limitations listed as follows inviting future works:

1) OSLA in Non-AWGN Channel Models: In this work, we only evaluated the performance in memoryless AWGN channels. The concept of OSLA can easily be applied to memoryless discrete channels or erasure channels with straightforward modifications to metric calculation in (2) or (16). On the other hand, threshold setting for frequency-selective fading channels and extension to channels with memory require further investigation.

2) Hard Latency Constraint: As a variable length scheme, the blocklength of OSLA is not fixed but a random variable (given a fixed average length). Many delay-sensitive applications have a hard deadline where the packet is consider lost when not delivered in time. The proposed OSLA would often fail to satisfy such a hard latency requirement due to the random nature of its blocklength. However, one can formulate a modified OSLA problem to control the symbol length under a strict blocklength deadline constraint. Designing a new policy for OSLA in such cases is a potential future direction.

3) Delayed Feedback: This work is based on the assumption of instantaneous feedback where the delay is only a single chip. This assumption may not hold due to propagation delay and processing time in some practical systems. A system that experiences longer (>1 chip) delay does not immediately fail and the performance gracefully degrades as the longer delay causes excessive energy on the current symbol making it more reliable than necessary, reducing the available energy for following symbols. Performance degradation due to this longer delay is benign when it is still substantially shorter than the average symbol length. Relaxing the delay assumption and devising a modified strategy for the minimized performance degradation is a potential future work.

4) OSLA with Other Coding Schemes: This work only investigates trellis codes (CC, TBCC and Turbo) for coded message transmission. Extension of OSLA to other coding schemes is of great interest but not straightforward. Combination with linear block codes or deep learning-based codes is a promising future direction. Different symbol-advancing criteria need to be investigated when other coding schemes are adopted.

VII. CONCLUSION

In this paper, we propose OSLA, an instantaneous feedback-based transmission scheme that automatically adapts the symbol length based on the noise realization at the receiver via instantaneous feedback to guarantee the target reliability for communication. OSLA can be combined with trellis codes such as turbo and TBCC to boost the performance, providing lower BLER than state-of-the-art short codes including a deep learning-based feedback scheme. Moreover, OSLA can easily scale to longer codeword lengths with consistent gain over fixed-length feedback-less schemes. Using pulse-based feedback signaling and HMM-based state synchronization, OSLA operates reliably in noisy feedback channels.

REFERENCES

[1] C.-W. Hsu, A. Anastasopoulos, and H.-S. Kim, "Instantaneous feedback-based opportunistic symbol length adaptation for reliable communication," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6.
[2] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Hoboken, NJ, USA: Wiley, 2006.
[3] R. L. Dobrushin, "An asymptotic bound for the probability error of information transmission through a channel without memory using the feedback," *Probl. Peredachi Inf.*, vol. 8, pp. 160–161, Jan. 1962.
[4] E. A. Haroutunian, "Lower bound for error probability in channels with feedback," *Problems Inf. Transmiss.*, vol. 13, no. 2, pp. 36–44, 1977.
[5] C. E. Shannon, "The zero error capacity of a noisy channel," *IRE Trans. Inf. Theory*, vol. 2, no. 3, pp. 8–19, Sep. 1956.
[6] J. Schalkwijk and T. Kailath, "A coding scheme for additive noise channels with feedback–I: No bandwidth constraint," *IEEE Trans. Inf. Theory*, vol. IT-12, no. 2, pp. 172–182, Apr. 1966.
[7] A. Ben-Yishai and O. Shayevitz, "Interactive schemes for the AWGN channel with noisy feedback," *IEEE Trans. Inf. Theory*, vol. 63, no. 4, pp. 2409–2427, Apr. 2017.
[8] Y. Urman and D. Burshtein, "Feedback channel communication with low precision arithmetic," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2020, pp. 2067–2072.

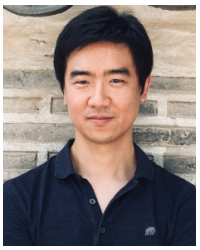
[9] N. C. Martins and T. Weissman, "Coding for additive white noise channels with feedback corrupted by quantization or bounded noise," *IEEE Trans. Inf. Theory*, vol. 54, no. 9, pp. 4274–4282, Sep. 2008.
[10] Z. Chance and D. J. Love, "Concatenated coding for the AWGN channel with noisy feedback," *IEEE Trans. Inf. Theory*, vol. 57, no. 10, pp. 6633–6649, Oct. 2011.
[11] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath, "Deepcode: Feedback codes via deep learning," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 1, pp. 194–206, May 2020.
[12] E. Ozfatura, Y. Shao, A. G. Perotti, B. M. Popović, and D. Gündüz, "All you need is feedback: Communication with block attention feedback codes," *IEEE J. Sel. Areas Inf. Theory*, vol. 3, no. 3, pp. 587–602, Sep. 2022.
[13] M. V. Burnashev, "Data transmission over a discrete channel with feedback. Random transmission time," *Problemy Inf. Inform.*, vol. 12, no. 4, pp. 10–30, Dec. 1976.
[14] H. Yamamoto and K. Itoh, "Asymptotic performance of a modified Schalkwijk–Barron scheme for channels with noiseless feedback (Corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-25, no. 6, pp. 729–733, Nov. 1979.
[15] S. C. Draper and A. Sahai, "Variable-length channel coding with noisy feedback," *Eur. Trans. Telecommun.*, vol. 19, no. 4, pp. 355–370, Jun. 2008.
[16] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Feedback in the non-asymptotic regime," *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 4903–4925, Aug. 2011.
[17] A. J. Viterbi, "The effect of sequential decision feedback on communication over the Gaussian channel," *Inf. Control*, vol. 8, no. 1, pp. 80–92, Feb. 1965.
[18] A. N. Borodin and P. Salminen, *Handbook of Brownian Motion Facts and Formulae*. Basel, Switzerland: Birkhauser, 1996.
[19] P. L. Smith, "A note on the distribution of response times for a random walk with Gaussian increments," *J. Math. Psychol.*, vol. 34, no. 4, pp. 445–459, Dec. 1990. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022249690900233
[20] C.-W. Hsu, H.-S. Kim, and A. Anastasopoulos, "Autocorrelation and spectrum analysis for variable symbol length communications with feedback," 2022, *arXiv:2211.11879*.
[21] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 2, pp. 260–269, Apr. 1967.
[22] J. G. Proakis and M. Salehi, *Digital Communications*, 5th ed. New York, NY, USA: McGraw-Hill, 2007.
[23] H. Ma and J. Wolf, "On tail biting convolutional codes," *IEEE Trans. Commun.*, vol. COM-34, no. 2, pp. 104–111, Feb. 1986.
[24] *Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and Channel Coding*, Standard 3GPP TS 36.212 2018.
[25] R. Y. Shao, S. Lin, and M. P. C. Fossorier, "Two decoding algorithms for tailbiting codes," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1658–1665, Oct. 2003.
[26] G. Fettweis and H. Meyr, "High-rate Viterbi processor: A systolic array solution," *IEEE J. Sel. Areas Commun.*, vol. 8, no. 8, pp. 1520–1534, Oct. 1990.
[27] G. Fettweis and H. Meyr, "High-speed parallel Viterbi decoding: Algorithm and VLSI-architecture," *IEEE Commun. Mag.*, vol. 29, no. 5, pp. 46–55, May 1991.
[28] *Xilinx UltraScale XCVU440 FPGA*. Accessed: Mar. 26, 2023. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/1-66ql3z.html
[29] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
[30] G. Liva, L. Gaudio, T. Ninacs, and T. Jerkovits, "Code design for short blocks: A survey," 2016, *arXiv:1610.00873*.
[31] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, Apr. 2010.
[32] A. Ben-Yishai and O. Shayevitz. *SK and Modulo-SK MATLAB Code*. Accessed: Dec. 13, 2022. [Online]. Available: https://github.com/assafbster/Modulo-SK

Chin-Wei Hsu received the B.S. degree in electrical engineering and the M.S. degree in communication engineering from the National Taiwan University, Taipei, Taiwan, in 2015 and 2017, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Michigan, Ann Arbor, MI, USA, in 2023. His research interests include novel modulation, coding schemes, and low power design for wireless communications.

Achilleas Anastasopoulos (Senior Member, IEEE) was born in Athens, Greece, in 1971. He received the Diploma degree in electrical engineering from the National Technical University of Athens, Greece, in 1993, and the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California, in 1994 and 1999, respectively.

He is currently an Associate Professor with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA. He is the coauthor of the book titled *Iterative Detection: Adaptivity, Complexity Reduction, and Applications* (Reading, MA: Kluwer Academic, 2001). His research interests include communication and information theory, with an emphasis on channel coding and multi-user channels, control theory, with an emphasis on decentralized stochastic control and its connections to communications and information theoretic problems, analysis of dynamic games, and mechanism design for resource allocation on networked systems.

Dr. Anastasopoulos was a recipient of the "Myronis Fellowship" from the Graduate School, University of Southern California, in 1996 and the NSF CAREER Award in 2004; and the coauthor for the paper that received the Best Student Paper Award in ISIT 2009. He served as a TPC member for ICC 2003 and 2015–2018; Globecom 2004 and 2012; VTC 2007, 2014 and 2015; ISIT 2015; and SPAWC 2018; a TPC Co-Chair for the Communication Theory Symposium, ICC'21. He served as an Associate Editor for IEEE TRANSACTIONS ON COMMUNICATIONS from 2003 to 2008. He has been serving as an Associate Editor for IEEE TRANSACTIONS ON INFORMATION THEORY, since 2020.



Hun-Seok Kim (Senior Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 2001, and the Ph.D. degree in electrical engineering from the University of California at Los Angeles, Los Angeles, CA, USA, in 2010. He is currently an Associate Professor with the University of Michigan, Ann Arbor, MI, USA. His research focuses on system analysis, novel algorithms, and VLSI architectures for low-power/high-performance wireless communications, signal processing, computer vision, and machine learning systems. He was a recipient of the DARPA Young Faculty Award in 2018 and the NSF CAREER Award in 2019. He is an Associate Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING.