Received: 00 Month 0000 | Revised: 00 Month 0000 | Accepted: 00 Month 0000

DOI: xxx/xxxx

ARTICLE TYPE

# Frame Synchronization for Pulsed Jammed Satellite Telecommand Links

**Noels N.*, Moeneclaey M.**

Telecommunications and Information Processing Department, UGent, Ghent, Belgium

**Correspondence**
*Noels Nele, UGent, Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium. Email: nele.noels@ugent.be

## Summary

A new issue of the satellite telecommand synchronization and channel coding sublayer protocol [1] includes LDPC coded communication link transmission units (CLTU) that contain a 64-bit start sequence. The novel data structures allow operation at lower signal-to-noise ratios than before, and offer improved protection against jamming attacks. This paper considers the corresponding CLTU frame synchronization process. We derive practical algorithms to locate the start sequence in the presence of high noise levels and pulsed jamming. The different algorithms are compared in terms of implementation complexity and performance under various jamming conditions. It is shown that among the considered frame synchronizers, those involving a full search over the entire observation window provide the desired accuracy, i.e., they guarantee a frame synchronization error probability that is significantly smaller than the codeword error rate, for codeword error rates near a target value of $10^{-4}$. Among these synchronizers, the full-search hard-decision-directed correlation-based algorithm has the lowest complexity.

**KEYWORDS:**
Satellite Telecommand System, Frame synchronization, Pulsed jamming

## 1 | INTRODUCTION

With a view to increasing the robustness of satellite telecommand (TC) links against jamming, the planned next-generation TC systems adopt direct-sequence spread-spectrum (DSSS) modulation with a very long pseudo-noise (PN) spreading code repetition period and a high spreading factor, along with advanced channel coding [1]. In [2,3,4], the codeword error rate (CER) performances of the recommended coding schemes have been investigated under DSSS modulation in the presence of jamming, assuming perfect chip, carrier, symbol and frame synchronization.

The present study focuses on frame synchronization. Frame synchronization in additive white Gaussian noise (AWGN) channels is a well-investigated problem [5,6,7,8]. In contrast, frame synchronization in the presence of jamming has received only limited attention in the literature. Algorithms, specifically designed to operate under jamming conditions, are described and evaluated in [9,10,11]. The usual space TC packet transmission scenario, where a known start sequence (SS) is prefixed to each block of coded data is considered only in [10]; however, the corresponding analysis is limited to continuous jamming, and so does not include the important case of pulsed jamming. More recently, the performance of a simple frame synchronization procedure based on sequential hypothesis testing and hard symbol-decisions has been studied under pulsed jamming conditions in [12]. The considered frame synchronizer was designed with the objective to minimize the modifications to the legacy SS search algorithm described in the CCSDS standard [13,1] for missions using a (63,56) modified BCH code with a hard-limiting detector followed by a triple-error-detection or a single-error-correction decoder. A possible way to further reduce the synchronization error probability (SEP) under jamming is to adopt more involved frame synchronization algorithms, similar to the ones proposed in [14,8] for AWGN and fading channels, but generalized to include pulsed jamming.

The outline of this paper is as follows. Section 2 describes the pulsed-jammed TC communication system under investigation. Section 3 formulates the corresponding maximum-likelihood (ML) frame synchronization rule. The latter is theoretically optimal in the sense of minimum SEP,

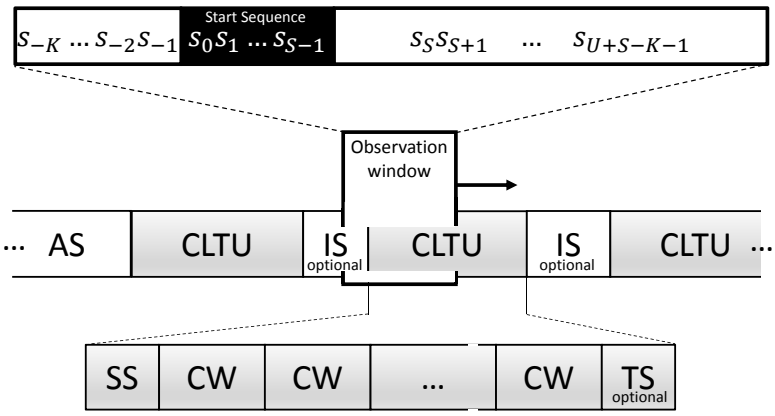


**FIGURE 1** Frame structure and illustration of observation window that contains a SS.

but difficult to implement in practice. Several practical ML-based frame synchronization algorithms are considered in Section 4. As opposed to the sequential frame synchronizer from [12], all considered algorithms are characterized by a full search over the complete a priori uncertainty region of the starting position of the SS. Among these are two decision-directed correlation-based algorithms, which do not require jammer state information (JSI). Additionally, we also consider more involved algorithms that require either average or instantaneous JSI. The sequential frame synchronizer previously considered in [12] is briefly revisited in Section 5. Section 6 compares the different algorithms in terms of implementation complexity, memory requirements and delay. Considering the specifics of next-generation TC systems [1,13], Section 7 presents numerical SEP results for the considered frame synchronization algorithms over pulsed-jamming channels. It is found that, in general, the largest threat comes from pulsed jammers with active periods that are long as compared to the size of the search window used to locate the SS. It is also shown that only the full-search frame synchronization algorithms yield a negligible contribution from frame synchronization errors to the overall system performance. Section 8 summarizes the main conclusions.

## 2 | SYSTEM DESCRIPTION

We consider a satellite TC communication system. The transmitting ground station uses forward error correction coding, Binary Phase Shift Keying (BPSK) and DSSS modulation to upload TCs of variable length over the physical channel to a receiving satellite. According to the CCSDS standard [13,1], during a communications session, a series of communication link transmission units (CLTUs) is generated and transmitted to the receiver (RX). This is illustrated in Fig. 1 . Each CLTU consists of a known SS that is immediately followed by a variable number of fixed-length codewords (CWs). Optionally, a tail sequence (TS) is transmitted after the last CW of a CLTU. The SS consists of S BPSK symbols, further denoted as $\mathbf{s}_0 = (s_0, s_1, ..., s_{S-1})$. For future missions, the different CLTUs of a communications session are delimited only by an optional idle sequence (IS), consisting of the repetition of a (1, -1) symbol pattern, and it is recommended that this IS is at least 8 symbols long. The first CLTU of a communications session is always preceded by an acquisition sequence (AS), also consisting of the repetition of a (1, -1) symbol pattern, to provide for initial symbol synchronization. Nominal systems typically include a feedback loop between the spacecraft and the ground station, whereby CLTU modulation at the ground station is started only after the confirmation that the spacecraft receiver has achieved symbol synchronization. However, such a feedback loop is not always available (a blind acquisition with no telemetry downlink is anyway necessary in off-nominal condition); therefore, a preferred minimum length of 128 AS symbols is recommended.

A block diagram of the TC communication system is depicted in Fig. 2 . The entire symbol sequence $\{s_k\}$, including the ASs, the SSs, the CWs, the TSs and the ISs, is converted into a sequence of non-return-to-zero pulses. This baseband signal is multiplied with a PN chip sequence to

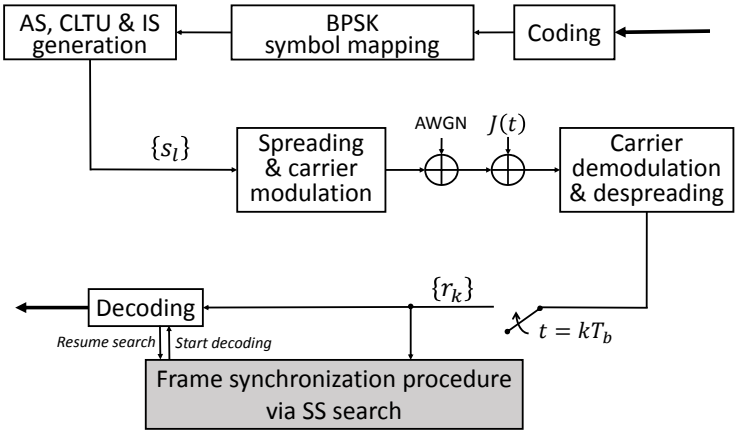


**FIGURE 2** Satellite TC communication system block diagram.

accomplish the spreading operation, and the resulting signal is modulated on a sinusoidal carrier $s(t)$. The bandwidth of the spread signal is about $\frac{T_b}{T_c}$ times as large as that of the original BPSK signal, with $T_b$ and $T_c$ denoting the bit interval and the chip interval, respectively[1].

The received DSSS signal is affected by AWGN with one-sided power spectral density (PSD) $N_0$ and by a pulsed jamming signal $J(t)$. The jammer is characterized by a repetition period equal to Y symbol intervals, which consists of an active period of D consecutive symbol intervals and an inactive period of $Y - D$ symbol intervals; the corresponding duty cycle of the jammer is given by $\rho = D/Y$. To simplify the analysis, the boundaries of the active and inactive periods are assumed to coincide with the symbol boundaries of the useful signal, so that a bit interval from the useful signal is either completely hit or not hit by the jammer. The $Y - D + 1$ possible starting positions of the active period within the corresponding repetition period are considered equally likely, and independent from one repetition period to the next. During the active period, the jammer power equals $P_{J,p}$, yielding a jammer energy per bit interval equal to $E_{J,p} = P_{J,p}T_b$ (the subscript 'p' refers to 'peak'); the long-term average jammer power is given by $P_{J,avg} = \rho P_{J,p}$.

The received signal is first despread (assuming perfect chip synchronization), then converted to baseband (assuming perfect carrier phase and frequency synchronization), and sampled at the symbol rate $1/T_b$ (assuming perfect symbol synchronization)[2]. The resulting samples $\{r_k\}$ are used for CLTU detection. The detection procedure involves:

1. Locating a SS to establish frame synchronization.
2. Decoding the subsequent CWs.

The search for the SS of the first CLTU of a communications session starts immediately after carrier, chip and symbol synchronization has been achieved. The search is interrupted during the decoding process and resumes when a decoding error is detected, or when a TS is found at the end a CW. In line with the CCSDS TC synchronization and channel coding standard[1], an incomplete decoder with an undetected CW error probability that is several orders of magnitude smaller than the detected CW error probability is assumed. As a result, frame synchronization errors almost surely cause an immediate decoding failure. If a complete decoder is employed (i.e., decoding never fails), the use of a TS is mandatory to force the SS search to resume at the end of a CLTU.

We consider large windows of $U + S$ observed samples (see Fig. 1 ); here, U is a design parameter smaller than the minimum length of a CLTU, to ensure that a window contains at most one complete SS. Based on the samples in the observation window, the frame synchronizer provides an estimate $\hat{K}$ of the delay (expressed in symbol intervals) of the start of a SS with respect to the start of that observation window. This estimate is then used to extract the CWs. Only if $\hat{K}$ corresponds to the actual start of the SS, the frame synchronization is correct. Observation windows that do not contain a complete SS will inevitably lead to a synchronization error.

[1]For TC applications, values of $\frac{T_b}{T_c}$ ranging from 10 to 1000 are typically considered[2], in combination with a PN sequence with a very long repetition period (the PN sequences reported in[15] have lengths (periods) between $2^{20}$ and $2^{26}$ ).

[2]At the start of a TC communication session, the transmitted ground station invokes a sequence of carrier modulation modes that are specifically designed to support the sequential acquisition of carrier, chip and symbol synchronization[1,13] prior to the reception of the first CLTU.

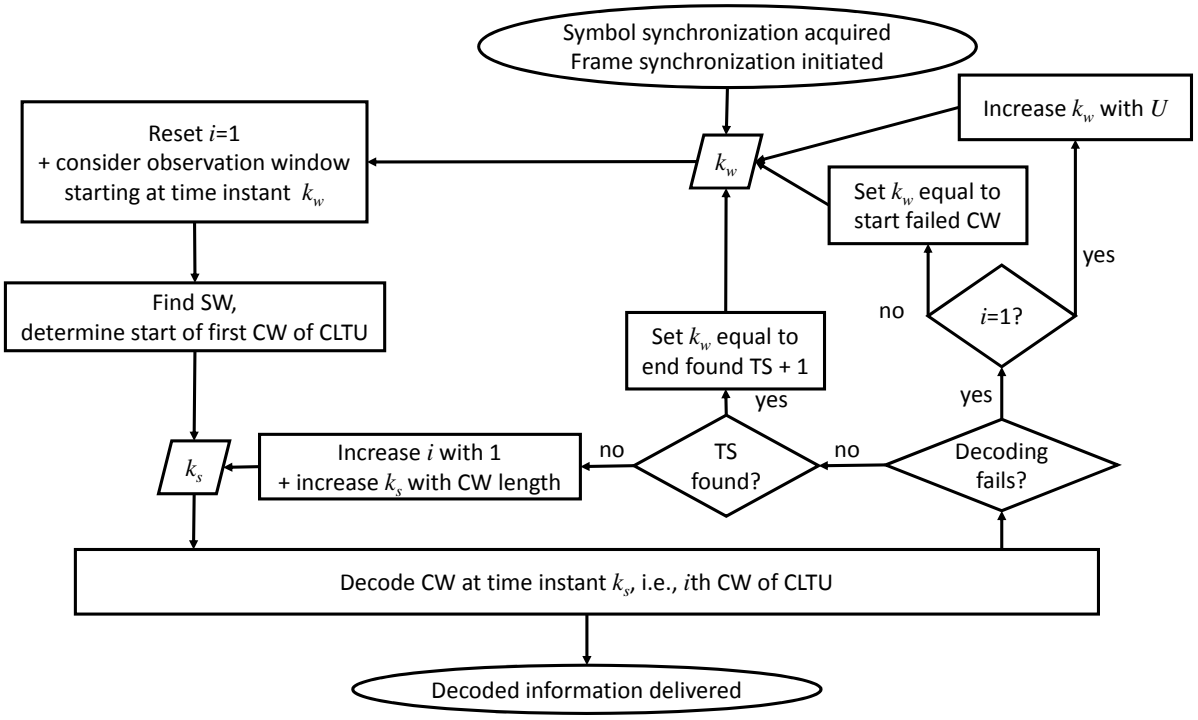


**FIGURE 3** Proposed CLTU detection procedure.

Let us consider the frame synchronization procedure that is illustrated by the flowchart in Fig. 3 , where $k_w$ denotes the starting position of the observation window, and the current CW is characterized by its starting position $k_s$ and its count i within the CLTU. If the first ($i = 1$) CW fails to decode, the SS search resumes in a window that overlaps with the last $S - 1$ samples of the previous search window (to avoid that a SS falls in between two windows)[8]. This way, the RX accounts for the fact the decoding failure might be caused by the absence of a complete SS in the considered observation window. On the other hand, if the first CW is properly decoded but one of the subsequent ($i > 1$) CWs fails to decode, the search for the SS resumes at the start of this failed CW. In this case, the RX assumes that the decoding failure results from the fact that the end of the CLTU has been over-run. If a TS is found, the search for the SS resumes after the TS. If the RX fails to detect the end of a CLTU, it will not start searching for a next SS; as a consequence one or more entire CLTUs may be missed. For incomplete decoders with a low undetected CW error rate, the probability that this happens is very small. For complete decoders, it is recommended that the RX performs a continuous TS search, rather than discrete TS checks after each CW. Indeed, if the RX checks for a TS after each CW only, there is a high probability that all the TSs that are received after a frame synchronization error are missed. This, in its turn, may result in the loss of all CLTUs received during a period equal to the maximum length of a CLTU. (It is safe to assume that the search for the SS will always resume after the maximum CLTU length, which is a known parameter.)

In the following we will focus on the case where the observation window contains a complete SS (see Fig. 1 ). Let $\mathbf{r} = (r_0, r_1, ..., r_{U+S-1})$ and $\mathbf{a} = (a_0, a_1, ..., a_{U+S-1})$ denote the the real-valued baseband samples in the observation window and the corresponding delayed symbols, respectively, with $a_k = s_{k-K}$, $s_k \in \{-1, 1\}$ and $\mathbf{s}_0 = (s_0, s_1, ..., s_{S-1})$ the known SS. Here, the discrete random variable K denotes the unknown delay of the start of the SS with respect to the start of the observation window. We have

$$r_k = \sqrt{E_s} s_{k-K} + n_k, \tag{1}$$

where the time index k refers to the kth symbol interval in the current observation window, $E_s$ is the received symbol energy, and the quantities $\{n_k\}$ represent the combined contribution from the AWGN and the pulsed jammer. The jamming contribution to $n_k$ is modeled by a zero-mean

Gaussian random variable; as the envisaged spreading factors $\frac{T_b}{T_c}$ are very large[2], this is an appropriate model for a variety of jammer waveforms. As a result, $\{n_k\}$ consists of independent real-valued zero-mean Gaussian random variables with time-dependent variance $\frac{1}{2}N_{0,eq}(k)$. When the jammer is active during the kth bit interval we have $N_{0,eq}(k) = N_0 + J_{0,p}$, with $J_{0,p} = E_{J,p}\frac{T_c}{T_b}$; when the jammer is inactive during the kth bit interval we have $N_{0,eq}(k) = N_0$. Hence, during its active periods the jammer has the same effect as AWGN, with one-sided PSD $J_{0,p}$, at the input of the receiver. The long-term average of $N_{0,eq}(k)$ is $N_0 + J_0$, with $J_0 = \rho J_{0,p}$.

## 3 | ML FRAME SYNCHRONIZATION

Considering in (1) a delay $K \in \mathscr{S}_U = \{0,1,...,U\}$ of the start of the SS compared to the start of the observation window, the samples $(r_K,...,r_{K+S-1})$ from the observation $\mathbf{r} = (r_0,r_1,...,r_{U+S-1})$ correspond to the known SS $\mathbf{s}_0 = (s_0,s_1,...,s_{S-1})$. The samples $(r_{K+S},...,r_{U+S-1})$ correspond to the $U-K$ symbols following the SS; besides CW symbols, these symbols can also contain optional TS symbols, optional IS symbols and SS symbols from the next CLTU. According to the standard, the number of CWs in a CLTU is unknown, in which case none of the symbols $(s_S,s_{S+1},...,s_{U+S-1-K})$ is known to the receiver. Finally, the samples $(r_0,...,r_{K-1})$ correspond to the K symbols $(s_{-K},s_{-K+1},...,s_{-1})$ preceding the SS. Taking into account that the frame synchronization starts when the symbol synchronization has been acquired, the symbols preceding the SS of the first CLTU belong to the AS, which is known to the RX. However, for a synchronization window containing the SS from one of the next CLTUs, the symbols preceding the SS can contain the optional IS of unknown length, the optional TS and a part of the last CW from the previous CLTU; hence, in this case not all symbols in front of the SS are known to the receiver. To avoid that the frame synchronizer operation depends on whether the SS of the first or a next CLTU is searched for, the frame synchronizer will be designed under the simplifying worst-case assumption that all U symbols surrounding the SS are unknown to the RX; these symbols will be modeled as independent equiprobable BPSK symbols. At the end of the section, we will briefly discuss how this design can be adapted if additional a priori information on the structure of the transmitted symbol sequence (fixed number of CWs per CLTU, minimum length IS per CLTU) is available.

The maximum likelihood (ML) frame synchronizer considers $k = \hat{K}$ as the start of the SS, where

$$\begin{aligned}\hat{K} &= \arg\max_{\tilde{K}\in\mathscr{S}_U} p\left(\mathbf{r}\middle|K=\tilde{K}\right),\\ &= \arg\max_{\tilde{K}\in\mathscr{S}_U} \ln p\left(\mathbf{r}\middle|K=\tilde{K}\right).\end{aligned} \tag{2}$$

In (2), $p\left(\mathbf{r}\middle|K=\tilde{K}\right)$ is the likelihood function of the delay K. When K is uniformly distributed over $\mathscr{S}_U$, the ML synchronizer minimizes the SEP. Assuming an AWGN channel, several practical implementations of this optimal frame synchronizer have been investigated in[5,7]. The extension to pulsed jamming channels basically involves taking into account the time-varying nature of the equivalent noise variance $N_{0,eq}(k)$.

Since the observation $\mathbf{r}$ depends not only on the delay K to be estimated but also on the symbol vector $\mathbf{a}$, the likelihood function of the delay is obtained as

$$p\left(\mathbf{r}\middle|K=\tilde{K}\right) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a})\, p\left(\mathbf{a}\middle|K=\tilde{K}\right), \tag{3}$$

where $p(\mathbf{r}|\mathbf{a})$ is given by $p(\mathbf{r}|\mathbf{a}) = \prod_{k=0}^{U+S-1} p(r_k|a_k)$, with (within a normalization factor not depending on $(r_l,a_l)$)

$$p(r_k|a_k) = \exp\left(-\frac{(r_k-\sqrt{E_s}a_k)^2}{N_{0,eq}(k)}\right), \tag{4}$$

and $p(\mathbf{a}|K)$ denotes the probability mass function of $\mathbf{a}$ conditioned on K. Hence, the ML estimate (2) reduces to

$$\hat{K} = \arg\max_{\tilde{K}\in\mathscr{S}_U} T\left(\tilde{K}\right), \tag{5}$$

where $T\left(\tilde{K}\right) = C\left(\tilde{K}\right) + R\left(\tilde{K}\right)$, with

$$C\left(\tilde{K}\right) = \sum_{k\in I_{\tilde{K}}} \dot{r}_k s_{k-\tilde{K}}, \tag{6}$$

$$R\left(\tilde{K}\right) = \sum_{k\in J_{\tilde{K}}} \ln\left(\cosh\left(\dot{r}_k\right)\right), \tag{7}$$

$$I_{\tilde{K}} = \left\{\tilde{K},\tilde{K}+1,...,\tilde{K}+S-1\right\}, \tag{8}$$

$$J_{\tilde{K}} = \left\{0,1,...,\tilde{K}-1\right\} \cup \left\{\tilde{K}+S,\tilde{K}+S+1,...,U+S-1\right\}, \tag{9}$$

and

$$\dot{r}_k = \frac{2\sqrt{E_s}}{N_{0,eq}(k)} r_k. \tag{10}$$

The frame synchronizer operating according to (5) needs perfect instantaneous JSI in order to obtain the quantities $N_{0,eq}(k)$ in (10) for $k \in \{0,...,S+U-1\}$; this synchronizer will be referred to as JSI_perf. Although perfect JSI is not available in practice, the corresponding performance serves as a useful benchmark for the performance of practical frame synchronizers. JSI_perf differs from the frame synchronizer presented in [5] in the time-variant scaling of the observed samples in (10). Taking into account that $N_{0,eq}(k)$ can fluctuate significantly, further simplification of the metric $T\left(\tilde{K}\right)$ through approximating the function $\ln\cosh(\cdot)$ as in [5,7] is difficult. The computational complexity associated with (5) can be reduced significantly by computing $R\left(\tilde{K}\right)$ from (7) recursively as:

$$R\left(\tilde{K}\right) = R\left(\tilde{K}-1\right) - \ln\left(\cosh\left(\dot{r}_{\tilde{K}+S-1}\right)\right) + \ln\left(\cosh\left(\dot{r}_{\tilde{K}-1}\right)\right), \tag{11}$$

for $\tilde{K} = 1,...,U$, with $R(0)$ as in (7). Note that a similar recursive computation of $C\left(\tilde{K}\right)$ is not possible.

In the following cases, the RX may have additional information about the symbols surrounding the SS:

1. A mission operates with a guaranteed minimum IS length X. If this is the case, the RX knows that the first X symbols prior to the SS are alternating {-1,+1} symbols (from AS or IS).

2. A mission operates with a fixed CLTU size L and a guaranteed minimum IS length X. In this case, the RX knows that the CLTU ends exactly L symbol periods after the start of the SS, so the TS will typically be omitted to save unnecessary overhead. However, the RX additionally knows that the first X symbols following the end of a CLTU are known IS symbols.

The knowledge in cases 1 and 2 can be exploited by replacing, in (6)-(7), $I_{\tilde{K}}$ and $J_{\tilde{K}}$ by $I_{\tilde{K}}^{(1)} = \left\{\max\left(0,\tilde{K}-X\right),...,\tilde{K}+S-1\right\}$ and $J_{\tilde{K}}^{(1)} = \{0,1,...,U+S-1\} \setminus I_{\tilde{K}}^{(1)}$ for case 1 and by $I_{\tilde{K}}^{(2)} = \left\{\max\left(0,\tilde{K}-X\right),...,\tilde{K}+S-1\right\} \cup \left\{\tilde{K}+S+L,\tilde{K}+S+L+1,...,\min\left(\tilde{K}+S+L+X-1,U+S-1\right)\right\}$ and $J_{\tilde{K}}^{(2)} = \{0,1,...,U+S-1\} \setminus I_{\tilde{K}}^{(2)}$ for case 2. Alternatively, the RX could be designed to search for the known symbol combination $SS^{(1)}$ in case 1 or $SS^{(2)}$ in case 2, with

$$SS^{(1)} = \left(\underbrace{\text{IS or AS}}_{X}\ \underbrace{\text{SS}}_{S}\right) \tag{12}$$

and

$$SS^{(2)} = \left(\underbrace{\text{IS}}_{X}\ \underbrace{\text{SS}}_{S}\ \underbrace{??\cdots?}_{L-S}\ \underbrace{\text{IS}}_{X}\right), \tag{13}$$

rather than for the SS only. It should be noted, however, that it can not be guaranteed that the first 1 to X symbols of $SS^{(i)}$, $i = 1,2$, always fall inside the first observation window (starting as soon as symbol synchronization is acquired). Nevertheless, redefining the observation window size as $W^{(1)} = S+X+U$ for case 1 and $W^{(2)} = L+2X+U$ for case 2, with $U-X$ smaller than the minimum length of a CLTU, the frame synchronization logic from Fig. 3 still applies. Moreover, the ML estimation rule (5)-(7) with (10) remains valid, provided that the vector of observed samples is denoted as $\mathbf{r} = \left(r_0, r_1, ..., r_{W^{(i)}-1}\right)$ for case i, the search space $\mathcal{S}_U = \{0,1,...,U+X\}$ is employed for the first observation window and $\mathcal{S}_U = \{X, X+1,...,U+X\}$ otherwise, and the index sets $I_K$ and $J_K$ are replaced by $I_K^{(i)}$ and $J_K^{(i)}$ for case i, such that, for a given delay K, the samples $r_k \in I_K^{(i)}$ from $\mathbf{r}$ correspond to $SS^{(i)}$ and $J_K^{(i)} = \left\{0,1,...,W^{(i)}-1\right\} \setminus I_K^{(i)}$. In all cases, as compared to the ML frame synchronizer that assumes that all U symbols surrounding the SS are unknown to the RX, a smaller SEP can be expected for given $\left\{N_{0,eq}(k)\right\}$ because a larger amount of prior knowledge is exploited. However, the impact on the SEP will be limited because alternating symbol sequences have bad auto-correlation properties and because X is typically much smaller than S. Hence, the above mission-specific variations are not further investigated.

## 4 | PRACTICAL FULL-SEARCH FRAME SYNCHRONIZERS

In this section we describe several frame synchronizers that are suited for practical implementation, and which can be considered as approximations of the JSI_perf synchronizer from (5). These synchronizers either use an estimate of the JSI, or avoid the need for JSI altogether. Among all these algorithms, JSI_est is the only one that performs a time-variant scaling of the received samples. The JSI_avg, JSI_no and HD algorithms have been studied previously in [5,6,7] for use under AWGN conditions.

### 4.1 | Estimated Instantaneous JSI

From (5), a more practical frame synchronization procedure results by replacing the unknown one-sided PSD $N_{0,eq}(k)$, of the equivalent noise at the input of the receiver during the kth symbol interval, by an estimate $\hat{N}_{0,eq}(k)$ obtained from the observed samples $\{r_k\}$.

Let us consider the following simple hard-decision-directed ML-based sliding window estimate of $N_{0,eq}(k)$, which tacitly assumes that the jammer state does not change for $k \in \{l-W, \ldots, l+W\}$:

$$\hat{N}_{0,eq}(k) = \sum_{k'=k-W}^{k+W} t_{k'}, \tag{14}$$

with

$$t_k = \frac{2}{2W+1}\left(r_k - \sqrt{E_s}\hat{a}_k\right)^2. \tag{15}$$

Here, $\hat{a}_k = \mathrm{sgn}(r_k)$ represents the hard decision on the delayed symbol $a_k = s_{k-K}$, with $\mathrm{sgn}(x)$ denoting the sign of $x$, and $W$ is a design parameter determining the size of the sliding window (i.e., $2W+1$ samples). For large $E_s/N_{0,eq}(k)$, with high probability we have $\hat{a}_k = s_{k-K}$ and, hence, $r_k - \sqrt{E_s}\hat{a}_k = w_k$. Note that $\hat{N}_{0,eq}(k)$ can be computed recursively as

$$\hat{N}_{0,eq}(k+1) = \hat{N}_{0,eq}(k) - t_{k-W} + t_{k+1+W}, \tag{16}$$

for $k = 0, \ldots, U+S-2$, with $\hat{N}_{0,eq}(0)$ as in (14).

The delay estimate $\hat{K}$ follows from (5), with $N_{0,eq}(k)$ in (10) replaced by $\hat{N}_{0,eq}(k)$; the resulting frame synchronizer will be denoted JSI_est(W). Noting that $\hat{N}_{0,eq}(k)$ must be computed for $k = 0, \ldots, S+U-1$, it follows from (14) that, to obtain all required $\hat{N}_{0,eq}(k)$, we need the observations $\{r_{-W}, \ldots, r_{W+U+S-1}\}$, which contains more samples than the vector $\mathbf{r}$ used for frame synchronization itself.

The value of W determines the size of the sliding window used to estimate $\hat{N}_{0,eq}(k)$, and should be selected taking the following trade-off into account: (i) when the JSI is constant over the sliding window, the accuracy of the estimate $\hat{N}_{0,eq}(k)$ at high $E_s/N_{0,eq}(k)$ improves with increasing W; (ii) the smaller the value of W compared to $\min(D, Y-D)$, the larger the probability that the JSI is constant over the sliding window. In addition, it should be taken into account that a larger W increases the computational complexity of (14).

### 4.2 | Estimated Average JSI

A further simplification of (5) results from ignoring the time-variability of $N_{0,eq}(k)$ over the observation window, in which case we end up with the ML synchronizer for an AWGN channel from [5,7]. In (5) the quantity $N_{0,eq}(k)$ is replaced by an estimate $\hat{N}_{0,avg}$ of its average $N_{0,avg}$ over the observation window; this average is given by

$$N_{0,avg} = \frac{1}{U+S}\sum_{k=0}^{U+S-1} N_{0,eq}(k). \tag{17}$$

We use for $\hat{N}_{0,avg}$ an estimate similar to (14), but with the sum taken over the entire observation window, i.e.,

$$\hat{N}_{0,avg} = \frac{2}{U+S}\sum_{k=0}^{U+S-1}\left(r_k - \sqrt{E_s}\hat{a}_k\right)^2. \tag{18}$$

The frame synchronizer that computes $\hat{K}$ according to (5) with $N_{0,eq}(k)$ in (10) replaced by $\hat{N}_{0,avg}$ will be further referred to as JSI_avg.

### 4.3 | No JSI

A simple ad hoc frame synchronizer which does not require any (estimated) JSI simply maximizes over the symbol index $\tilde{K}$ the correlation of $(r_{\tilde{K}}, \ldots, r_{\tilde{K}+S-1})$ with the SS (see also [7]). In this case, $\hat{K}$ is given by (5) with $T\left(\tilde{K}\right)$ given by

$$T\left(\tilde{K}\right) = \sum_{k=\tilde{K}}^{\tilde{K}+S-1} r_k s_{k-\tilde{K}}. \tag{19}$$

The corresponding frame synchronizer will be referred to as JSI_no. This synchronizer can be viewed as resulting from keeping in the right-hand side of (2) only terms involving the samples $(r_{\tilde{K}}, \ldots, r_{\tilde{K}+S-1})$ that correspond to the trial location of the SS, and assuming that the JSI does not change over the observation interval. In (19), scaling of the received samples as in (10) is no longer performed. It is easily understood that such scaling would even be detrimental to the SEP performance:

- The quantity $T\left(\tilde{K}\right)$ in (19) is a Gaussian random variable with mean $\mu\left(K-\tilde{K}\right)=\sqrt{E_s}\sum_{k=0}^{S-1}s_{k+\tilde{K}-K}s_k$ and variance $\upsilon\left(\tilde{K}\right)=\sum_{k=\tilde{K}}^{\tilde{K}+S-1}\frac{N_{0,eq}(k)}{2}$, where $\mu(0)=S\sqrt{E_s}$ and $\upsilon\left(\tilde{K}\right)\leq S\frac{N_0+J_{0,p}}{2}$ for all $K\in\mathscr{S}_U$. Let $\varepsilon$ denote the maximum value of the correlation $\left|\sum_{k=0}^{S-1}s_{k+\tilde{K}-K}s_k\right|$ for $\tilde{K}\in\mathscr{S}_U\setminus\{K\}$; then, a sufficient condition for low SEP is $S\gg\max\left(\varepsilon,\frac{N_0+J_{0,p}}{2E_s}\right)$.

- Now, let us consider the quantity $\dot{T}\left(\tilde{K}\right)=\sum_{l=\tilde{K}}^{\tilde{K}+S-1}\dot{r}_l s_{l-\tilde{K}}$ with scaled received samples $\dot{r}_l$ rather than $r_l$. Like $T\left(\tilde{K}\right)$, $\dot{T}\left(\tilde{K}\right)$ is a Gaussian random variable. The mean of $\dot{T}\left(\tilde{K}\right)$ is $\dot{\mu}\left(K-\tilde{K}\right)=2E_s\sum_{k=0}^{S-1}\frac{s_{k+\tilde{K}-K}s_k}{N_{0,eq(k)}}$ and its variance is $\dot{\upsilon}\left(\tilde{K}\right)=\sum_{k=\tilde{K}}^{\tilde{K}+S-1}\frac{2E_s}{N_{0,eq}(k)}$. In this case, it is easily verified that $S\frac{2E_s}{N_0+J_{0,p}}\leq\mu(0)$, $|\mu(k)|\leq\varepsilon\frac{2E_s}{N_0}$ and $\upsilon\left(\tilde{K}\right)\leq S\frac{2E_s}{N_0}$. Hence, the SEP from (5) with $\dot{T}\left(\tilde{K}\right)$ is guaranteed to be small only if $S\gg\left(\frac{N_0+J_{0,p}}{N_0}\right)\max\left(\varepsilon,\frac{N_0+J_{0,p}}{2E_s}\right)$, which a significantly more stringent constraint than $S\gg\max\left(\varepsilon,\frac{N_0+J_{0,p}}{2E_s}\right)$ for (5) with $T\left(\tilde{K}\right)$.

### 4.4 | Hard Decisions

A hard-decision-based variant of the JSI_no frame synchronizer from Section 4.3 (see also [7]) consists of replacing in (19) $r_k$ by the hard decision $\hat{a}_k=\mathrm{sgn}(r_k)$, yielding

$$\hat{K}=\arg\min_{\tilde{K}\in\mathscr{S}_U}d_H\left(\hat{\mathbf{a}}_{\tilde{K}},\mathbf{s}_0\right), \tag{20}$$

where $\hat{\mathbf{a}}_{\tilde{K}}=\left(\hat{a}_{\tilde{K}},\hat{a}_{\tilde{K}+1},\ldots,\hat{a}_{\tilde{K}+S-1}\right)$ and $d_H\left(\hat{\mathbf{a}}_{\tilde{K}},\mathbf{s}_0\right)$ is the Hamming distance between $\hat{\mathbf{a}}_{\tilde{K}}$ and $\mathbf{s}_0$. The frame synchronizer operating according to (20) will be termed HD.

## 5 | SEQUENTIAL-SEARCH FRAME SYNCHRONIZER

In order to avoid the full search in (20) over all $\tilde{K}$ in $\mathscr{S}_U$, a further simplification consists in selecting a threshold value $t$ and defining $\hat{K}$ as the smallest value of $\tilde{K}$ for which $d_H\left(\hat{\mathbf{a}}_{\tilde{K}},\mathbf{s}_0\right)$ is smaller than or equal to $t$ (see also [6]). We will further refer to this frame synchronizer as HD_thr(t), with $t$ denoting the threshold value. The HD_thr(t) algorithm is only a minor extension of the legacy frame synchronizer that is specified in the CCSDS telecommand standard [1], which was recommended for BCH coding and a SS of 16 symbols, and which declared frame synchronization when at the output of a hard symbol-detector a 16-symbols sequence was found that differed from this SS in at most 0 (when the BCH code was used for triple-error detection) or 1 (when the BCH code was used for single-error correction) symbols.

While for the full-search algorithms JSI_perf, JSI_est(W), JSI_avg, JSI_no and HD, the SEP in windows that contain a complete SS simply is the probability that $\hat{K}$ differs from K, the situation is somewhat different for HD_thr(t). For HD_thr(t), the SEP in windows that contain a complete SS is the probability of the union of two events. The first event is a false alarm, which corresponds to the case where $d_H\left(\hat{\mathbf{a}}_{\tilde{K}},\mathbf{s}_0\right)\leq t$ for at least one value of $\tilde{K}$ in $\{1,2,\ldots,K-1\}$, yielding $\hat{K}<K$. The second event is missed detection, in which case we have $d_H\left(\hat{\mathbf{a}}_K,\mathbf{s}_0\right)>t$. The selection of the threshold value $t$ is a trade-off between the false-alarm probability (increasing function of $t$) and the missed-detection probability (decreasing function of $t$).

## 6 | IMPLEMENTATION CONSIDERATIONS

As the considered frame synchronization algorithms are intended for implementation on an on-board telemetry tracking and command (TTC) transponder, a comparison in terms of computational complexity and memory requirements is of great importance.

First, we consider JSI_perf. A possible implementation is outlined in Algorithm 1, where the average number of elementary operations (eops), including elementary memory access operations, associated with each step is indicated between parentheses. It follows that a total of $4US+14U+11S+2\alpha$ eops is required, where $\alpha\in[0,U]$ represents the average number of values $\tilde{K}$ in $\{1,2,\ldots,U\}$ for which $T\left(\tilde{K}\right)$ in (5) is strictly larger than any element from $\{T(0),\ldots,T\left(\tilde{K}-1\right)\}$.

Next, we consider the JSI_est(W) and JSI_avg algorithms that use estimated rather than true JSI. Both these algorithms comprise all the processing steps of JSI_perf, supplemented with prior steps to estimate the JSI. In the case of JSI_est(W), $U+S$ PSD values $\hat{N}_{0,eq}(k)$, $k=0,1,\ldots,U+S-1$, have to be computed. Assuming that the quantities $t_k$ from (15), for $k=-W,-W+1,\ldots,U+S+W-1$, are computed and stored prior to running the recursion (16), evaluating these PSD values requires $13U+13S+14W$ elementary operations. This brings the average total number of eops for JSI_est(W) to $4US+27U+24S+14W+2\alpha$. In addition, as compared to JSI_perf, extra memory is required to store the $U+S+2W$ values of $t_k$, $k=-W,-W+1,\ldots,U+S+W-1$. On the other hand, in the case of JSI_avg, the computation of $\hat{N}_{0,av}$ (18) from $\mathbf{r}$ requires only $6U+5S$ additional elementary operations and no additional memory as compared to JSI_perf. The average total number of eops for JSI_avg amounts to $4US+20U+16S+2\alpha$.

**Algorithm 1** Procedure to find $\hat{K}$ (5) from $\mathbf{s}_0$, $\mathbf{r}$ and $\left\{\frac{2\sqrt{E_s}}{N_{0,eq}(0)},...,\frac{2\sqrt{E_s}}{N_{0,eq}(U+S-1)}\right\}$. Proposed JSI_perf frame synchronizer implementation.

Input: $\mathbf{s}_0$, $\mathbf{r}$

Memory space to store:

- $\overrightarrow{\mathbf{r}}$ and $\overrightarrow{\mathbf{v}}$: vectors of $U+S$ components from $\mathbb{R}$,
- $\overrightarrow{\mathbf{s}}$: vector of S components from $\{-1,1\}$,
- $\overrightarrow{M}$: scalar from $\mathbb{R}$,
- $\overrightarrow{K}$: scalar from $S_U$.

Initialization: write $\mathbf{s}_0$ to $\overrightarrow{\mathbf{s}}$, $\mathbf{r}$ to $\overrightarrow{\mathbf{r}}$ and $\left(\frac{2\sqrt{E_s}}{N_{0,eq}(0)},...,\frac{2\sqrt{E_s}}{N_{0,eq}(U+S-1)}\right)$ to $\overrightarrow{\mathbf{v}}$.

Processing steps:

1. Read $\overrightarrow{\mathbf{r}}$ and $\overrightarrow{\mathbf{v}}$, compute $\{\dot{r}_0,...,\dot{r}_{U+S-1}\}$ as in (10), and write result to $\overrightarrow{\mathbf{r}}$. $(4(U+S)$ eops)
2. Read $\overrightarrow{\mathbf{r}}$, look-up $\{\ln(\cosh(\dot{r}_0)),...,\ln(\cosh(\dot{r}_{U+S-1}))\}$, and write result to $\overrightarrow{\mathbf{v}}$. $(3(U+S)$ eops)
3. Read $\overrightarrow{\mathbf{s}}$, read the S first entries of $\overrightarrow{\mathbf{r}}$ and read the U last entries of $\overrightarrow{\mathbf{v}}$. Compute $T(0)=C(0)+R(0)$ using (6) and (7). Write result to $\overrightarrow{M}$. $(2U+4S$ eops)
4. For $\tilde{K}=1,2,...,U$, repeat the following. Read $\overrightarrow{\mathbf{s}}$ and read the entries $\tilde{K}+1$ till $\tilde{K}+S$ of $\overrightarrow{\mathbf{r}}$. Use (6) and (11) to compute $T\left(\tilde{K}\right)$ from $T\left(\tilde{K}-1\right)$. Compare result to the value stored in $\overrightarrow{M}$. If $T\left(\tilde{K}\right)$ is strictly larger than that value, write $T\left(\tilde{K}\right)$ to $\overrightarrow{M}$ and write $\tilde{K}$ to $\overrightarrow{K}$. $((4S+4)U+2\alpha$ eops)

Output: $\hat{K}=\overrightarrow{K}$

Finally, we consider the HD, the JSI_no and the HD_thr algorithms. As opposed to the JSI_est and JSI_avg algorithms, these algorithms only require the evaluation and maximization of a size-S correlation and thereby avoid the additional table look-ups for $\ln(\cosh(\cdot))$ and do not require the computation, processing (as a scaling factor in (10)) and storage of noise PSD estimates. For JSI_no, the number of eops, including elementary memory access operations, reduces to $4US+U+2\alpha$. Moreover, storage space for $U+S$ variables in $\mathbb{R}$ (estimated PSD values) is saved with respect to JSI_perf. As compared to JSI_no, HD requires the computation and storage of hard symbol-decisions but involves computing a integer-valued Hamming distance rather than a real-valued correlation. HD_thr(t) is the only frame synchronizer that does not introduce an additional delay; this is because it performs a sequential rather than a full-search algorithm. As a result HD_thr(t) involves on average only half as many computations as HD.

An overview of the average number of eops per observation window for the various algorithms is provided in Table 1 . For given U and S, HD_thr has the lowest implementation complexity of all considered algorithms. Further, the HD and JSI_no algorithms yield a significantly lower complexity than JSI_est and JSI_avg that take into account the whole structure of the observed symbol sequence.The approximate expressions in the first column only hold for large values of U and S. The numerical values listed in the next two columns correspond to the values of U, S and W that will be employed in the numerical results section (Section 7), namely U = 575, S=64 and $W=32$. We observe that, with these parameter values, the complexity of the sequential search algorithm HD_thr is about half as large as the complexity of the full search algorithms. The various full search algorithms only show rather small variations in average number of eops. Roughly speaking, the algorithms that do not use JSI (JSI_no and HD) require 6% less eops than JSI_avg and about 9% less eops than JSI_est.

Since different types of eops may come with a different implementation complexity, the composition of the expressions provided above are further detailed in Table 2 , which separately indicates the average number of elementary comparisons (COMP), additions (ADD), multiplications (MUL), table look-ups (LUT), exclusive or operations (XOR) and memory reading or writing operations (R/W) per observation window and per synchronizer.

## 7 | NUMERICAL PERFORMANCE EVALUATION

We present numerical results for $\mathrm{SEP}_*$, i.e., the SEP in windows that contain a complete SS. We consider pulsed jamming that is characterized by the parameters $\left(D,\rho,\frac{E_s}{J_{0,p}}\right)$; the corresponding $\frac{E_s}{J_0}$ is obtained as $\frac{E_s}{J_0}=\frac{E_s}{\rho J_{0,p}}$ . The focus lies on next-generation TC systems, with $S=64$ (8 bytes), a

| | Expression | Lower bound ($\alpha = 0$) for $(U, S, W)$ as in Section 7 | Upper bound ($\alpha = U$) for $(U, S, W)$ as in Section 7 |
|---|---|---|---|
| JSI_perf | $4US + 14U + 11S + 2\alpha \approx 4US$ | $15.6 \cdot 10^4$ | $15.7 \cdot 10^4$ |
| JSI_est | $4US + 27U + 24S + 14W + 2\alpha \approx 4US$ | $16{,}5 \cdot 10^4$ | $16{,}6 \cdot 10^4$ |
| JSI_avg | $4US + 20U + 16S + 2\alpha \approx 4US$ | $16{,}0 \cdot 10^4$ | $16{,}1 \cdot 10^4$ |
| JSI_no | $4US + U + 2\alpha \approx 4US$ | $14{,}8 \cdot 10^4$ | $14{,}9 \cdot 10^4$ |
| HD | $4US + 2U + S + 2\alpha \approx 4US$ | $14{,}9 \cdot 10^4$ | $15.0 \cdot 10^4$ |
| HD_thr | $2US + 1.5U + 2S \approx 2US$ | $7.5 \cdot 10^4$ | $7.5 \cdot 10^4$ |

**TABLE 1** Average number of elementary operations per observation window for the various algorithms; expressions in terms of the size S of the SS, the width $U + S$ of the observation window and the width $2W + 1$ of the PSD estimation window, as well as numerical lower and upper bound values for $U = 575$, $S = 64$ and $W = 32$ as in Section 7.

| | # COMP | # ADD | # MUL | # LUT | # XOR | # R/W |
|---|---|---|---|---|---|---|
| JSI_perf | U | $US + 5U + S$ | $US + U + 2S$ | $U + S$ | 0 | $2US + 6U + 7S + 2\alpha$ |
| JSI_est(W) | $2U + S + 2W$ | $US + 8U + 4S + 4W$ | $US + 4U + 5S + 2W$ | $2(U + S)$ | 0 | $2US + 11U + 12S + 6W + 2\alpha$ |
| JSI_avg | $2U + S$ | $US + 6U + S$ | $US + 3U + 4S$ | $2(U + S)$ | 0 | $2US + 7U + 8S + 2\alpha$ |
| JSI_no | U | US | US | 0 | 0 | $2US + 2\alpha$ |
| HD | $2U + S$ | US | 0 | 0 | US | $2US + U + S + 2\alpha$ |
| HD_thr(t) | $U + S$ | 0.5US | 0 | 0 | 0.5US | $US + 0.5U + S$ |

**TABLE 2** Average number of elementary comparisons (COMP), additions (ADD), multiplications (MUL), table look-ups (LUT), exclusive or operations (XOR) and memory reading or writing operations (R/W) per observation window for the various algorithms, expressed in terms of the size S of the SS, the width $U + S$ of the observation window and the width $2W + 1$ of the PSD estimation window.

SS given by

$$\mathbf{s}_0 = (034776C7272895B0)_{16}\,, \tag{21}$$

in hexadecimal notation, and rate 1/2 LDPC(512,256) channel coding [1]. As in [2,4,12], we assume a nominal operating SNR of $E_s/N_0$ = 7 dB. The parameter U specifying the observation window size $(U + S)$ is fixed to $U = 575$, which is one less than the minimum size of a CLTU for the recommended code with block length 512[3]. Monte Carlo simulations are performed. For each realization of the delay K, uniformly distributed in $\{0, 1, ..., U + S\}$, $U + S$ BPSK symbols $(s_{-K}, s_{-K+1}, ..., s_{U+S-K-1})$ are generated, where $(s_0, s_1, ...s_{S-1})$ is the known SS $\mathbf{s}_0$ and the U remaining entries are independent equiprobable BPSK symbols. This is a simplifying assumption. In reality, the symbols preceding $\mathbf{s}_0$ are the last $(K - X')$ bits of a CW, followed by $X'$ (with $X' \geq 0$) -1/+1 bits of an IS or AS. The bits succeeding $\mathbf{s}_0$ are the first $(U - K)$ bits of another CW. Strictly speaking, the coding involved in the observed symbols might have an impact on the synchronizers ' performances because it effects the statistics of the correlation terms in (6), (19) and (20). We argue that it is nevertheless safe to replace the coded symbols by independent symbols in the simulations because the considered LDPC codes display a high level of randomness and because the number of bits involved in the correlation is much smaller than the code word length. Furthermore, by design $\mathbf{s}_0$ has a low correlation with sequences of alternating -1/+1 symbols [13] such that replacing in the simulations IS and AS symbols by random symbols can be expected to yield overestimated $\mathrm{SEP}_*$ results. Through simulations (not shown here)

[3] By choosing U as large as possible, we minimize the overhead that results from processing observation windows that do not contain a SS. As specified in [1], no TS is used in the case of LDPC (512, 256) coding.

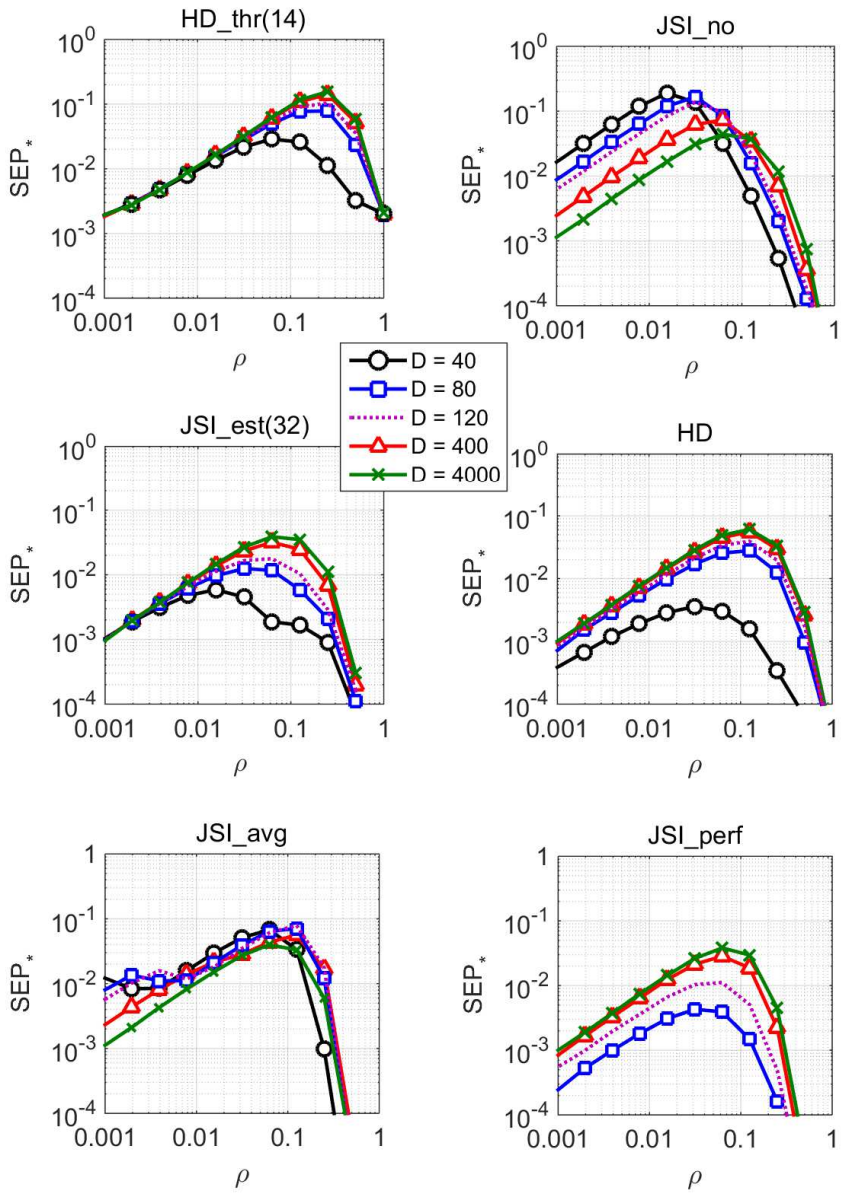


**FIGURE 4** $\mathrm{SEP}_*$ as a function of $\rho$ at $E_s/N_0$ = 7 dB, for S = 64, $E_s/J_0$ = 0 dB and U = 575.

we have verified that said overestimation is smaller than a factor of 2. Hence, the obtained $\mathrm{SEP}_*$ values can be considered as a meaningful upper bound on the actual $\mathrm{SEP}_*$ during a TC communication session.

The probability $\mathrm{SEP}_*$ is displayed as a function of the duty cycle $\rho$ in Fig. 4 for $E_s/J_0$ = 0 dB and in Fig. 5 for $E_s/J_0$ = 5 dB; the considered algorithms are JSI_perf, JSI_est(W) with W = 32, JSI_avg, JSI_no, HD and HD_thr(t), with t = 14 as proposed in [12]. Note that $\mathrm{SEP}_*$ for JSI_perf and D = 40 is smaller than $10^{-5}$ for all $\rho$ in $[10^{-3}, 1]$ and therefore falls outside the range of the bottom-right diagram in Figs. 4 and 5 . Comparing Fig. 4 to Fig. 5 , we observe that, for given $\rho$ and D, $\mathrm{SEP}_*$ decreases with increasing $E_s/J_0$. This was to be expected since, for given $\rho$, $J_0$ is proportional to the peak jammer power. We now study the behavior of $\mathrm{SEP}_*$ as a function of the duty cycle $\rho$ and the active period D of the jammer, for given $E_s/J_0$.

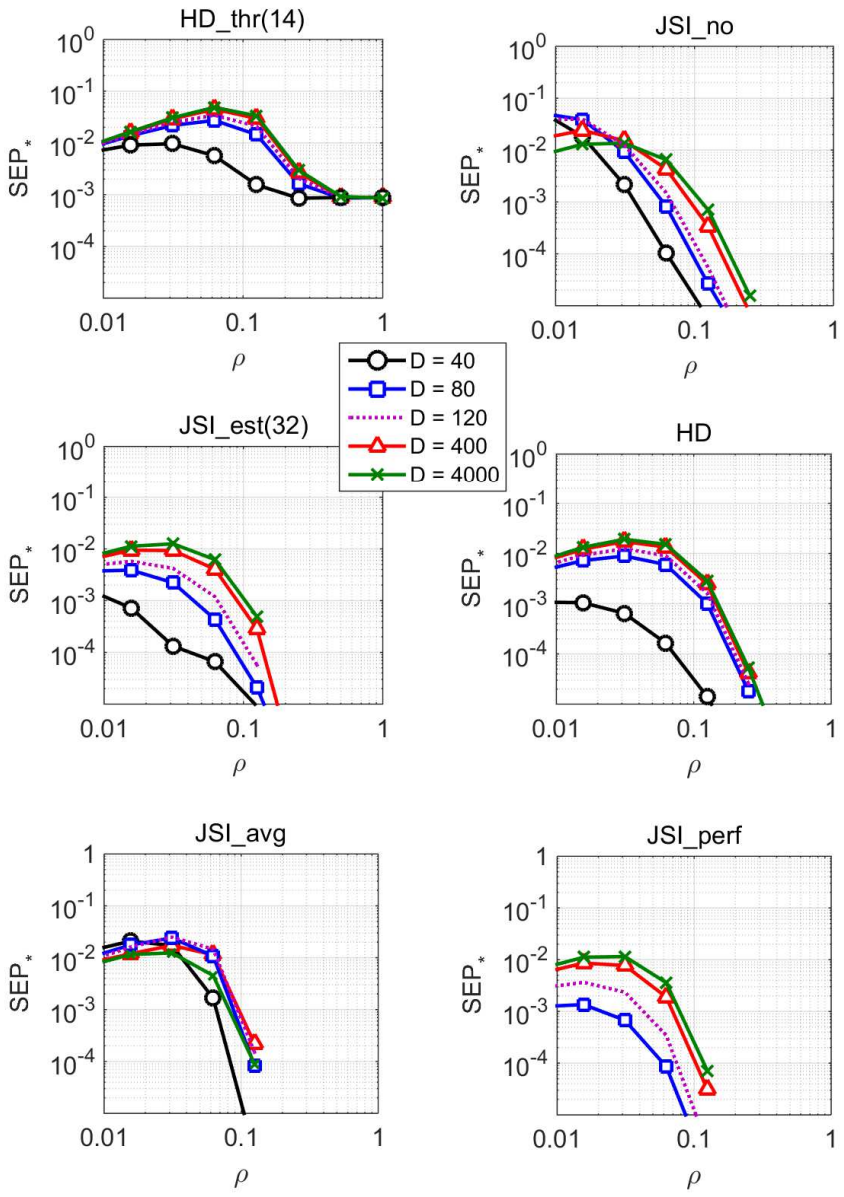


**FIGURE 5** $\mathrm{SEP}_*$ as a function of $\rho$ at $E_s/N_0$ = 7 dB, for $S = 64$, $E_s/J_0$ = 5 dB and $U = 575$.

1. For fixed D, there is a (worst-case) value of $\rho$ that maximizes $\mathrm{SEP}_*$. Let $N_J$ denote the average amount of symbols (within an observation window) that are hit by jamming, and let $\sigma_J^2 = \left(\frac{N_0}{E_s} + \frac{J_0}{\rho E_s}\right)$ denote the equivalent noise variance (normalized to the symbol energy) that is experienced by these jammed symbols. It is intuitively clear that $\mathrm{SEP}_*$ is an increasing function of both $N_J$ and $\sigma_J^2$. For given $E_s/N_0$, $E_s/J_0$ and D, a change in $\rho$ has opposite effects on $N_J$ and $\sigma_J^2$; more specifically, with increasing $\rho$, $N_J$ increases while $\sigma_J^2$ decreases. The former and latter effect are dominant for small and large $\rho$, respectively, which explains the existence of a worst-case value for $\rho$. This worst-case $\rho$ is seen to decrease when (i) $E_s/J_0$ increases (for fixed D) or (ii) D decreases (for fixed $E_s/J_0$).

2. For a fixed $\rho$ larger than about $0.1\frac{J_0}{E_s}$ [4], which yields $\rho > 0.1$ and $\rho > 0.032$ in Fig. 4 and Fig. 5, respectively, the maximum values of $\mathrm{SEP}_*$ for JSI_perf, JSI_est(32), JSI_no, HD and HD_thr(14) are obtained for active jammer periods D much larger than the observation window length U + S (e.g., D = 4000, while U + S = 640). For such large values of D, the observation window can be considered as either completely hit (with probability $\rho$) or not hit (with probability $1-\rho$) by the jammer. As a result, $\mathrm{SEP}_*$ is well approximated by $(1-\rho)\,\mathrm{SEP}_{*,\rho=0} + \rho\,\mathrm{SEP}_{*,\rho=1}$, where $\mathrm{SEP}_{*,\rho=0}$ denotes the value of $\mathrm{SEP}_*$ in the absence of jamming, and $\mathrm{SEP}_{*,\rho=1}$ is the value of $\mathrm{SEP}_*$ in the presence of continuous jamming with a power equal to the peak power of the actual jammer. Moreover, in most practical situations, $(1-\rho)\,\mathrm{SEP}_{*,\rho=0}$ is negligibly small as compared to $\rho\mathrm{SEP}_{*,\rho=1}$, in which case $\mathrm{SEP}_*$ can be approximated as $\rho\mathrm{SEP}_{*,\rho=1}$. Taking into account that U + S is limited by the codeword length, a jammer can for given $\rho$ and $J_{0,p}$ select without penalty an active period D much larger than U + S. Under the condition $D \gg U+S$, it is sufficient to compare JSI_perf, JSI_est(32), JSI_no, HD and HD_thr(t) based on their $\mathrm{SEP}_{*,\rho=1}$ performance. This worst-case analysis essentially comes down to an AWGN channel analysis with the proper one-sided noise PSD, equal to $N_0 + J_{0,p}$.

3. For fixed $\rho$ larger than about $0.1\frac{J_0}{E_s}$, the largest $\mathrm{SEP}_*$ values for JSI_avg are not achieved for D = 4000 but rather for D = 400 or D = 120. This deviating behavior of JSI_avg compared to the other algorithms can be explained as follows. For D = 4000, JSI_avg and JSI_perf perform essentially the same; this is because $N_{0,\mathrm{av}}$ from (17) almost surely equals $N_{0,\mathrm{eq}}(k)$, for $k = 0,1,...,U+S-1$ when $D \gg U+S$, and $\hat{N}_{0,\mathrm{av}}$ from (18) is an accurate estimate of $N_{0,\mathrm{av}}$ when $U+S \gg 1$. For given $\rho$, the probability that the observation window is only partially hit, increases as D decreases, causing $N_{0,\mathrm{av}}$ to differ from $N_{0,\mathrm{eq}}(k)$, for $k = 0,1,...,U+S-1$, so that JSI_avg deteriorates compared to JSI_perf for $D \in \{400, 120, 80, 40\}$. On the other hand, the performance of JSI_perf itself significantly improves with diminishing D. The combined effect results in the existence of a worst-case value of D for which JSI_avg achieves a maximum value of $\mathrm{SEP}_*$. This worst-case D decreases with decreasing $\rho$. For $\rho$ larger than about $\left(10\frac{E_s}{J_0}\right)^{-1}$, close-to-maximum SEP is obtained for D = 400, which value is relatively independent of $E_s/J_0$.

In the following, we investigate the worst-case performances of the different synchronizers considered. Restricting our attention to the realistic constraint $E_s/J_{0,p} > 0.1$, the maximum SEP for JSI_avg occurs at $D \approx 400$, whereas for the other synchronizers the condition $D \gg U+S$ yields the largest SEP. We will present curves of $\mathrm{SEP}_*/\rho$ (instead of $\mathrm{SEP}_*$) versus $E_s/J_{0,p}$ for different $\rho$. For a synchronizer operating under the condition $D \gg U+S$, we have $\mathrm{SEP}_*/\rho \approx \mathrm{SEP}_{*,\rho=1}$, which indicates that $\mathrm{SEP}_*/\rho$ is essentially independent of $\rho$; in this case, we display only the curve for $\rho = 1$, i.e., $\mathrm{SEP}_{*,\rho=1}$, in order not to overload the figure. For JSI_avg with $D \approx 400$, distinct curves of $\mathrm{SEP}_*/\rho$ will be shown for different values of $\rho$.

Among the practical frame synchronization algorithms considered, the HD_thr(t) algorithm has the lowest complexity and the lowest delay. Under the condition $D \gg U+S$, Fig. 6 shows $\mathrm{SEP}_*/\rho$ versus $\frac{E_s}{J_{0,p}}$, for JSI_perf and for HD_thr(t) with $t = 12, 14, 16, 18$. We observe that the performance of HD_thr(t) exhibits an error floor for large $\frac{E_s}{J_{0,p}}$; this floor corresponds to the false alarm probability in the absence of jamming, and increases with increasing t. For given $\frac{E_s}{J_{0,p}}$, the SEP can be minimized by optimizing over t; the optimum t decreases with increasing $\frac{E_s}{J_{0,p}}$. However, since accurate JSI, required to select the optimum t, is assumed unavailable for this synchronizer, we are left to conclude that HD_thr(t) exhibits a rather limited robustness to pulsed jamming. The HD_thr(t) algorithm is significantly outperformed by JSI_perf; this indicates that more involved practical frame synchronization algorithms are required to get closer to the JSI_perf performance.

Fig. 7 displays $\mathrm{SEP}_*/\rho$ versus $\frac{E_s}{J_{0,p}}$, related to the JSI_perf, JSI_est(32), JSI_no and HD synchronizers for $D \gg U+S$, and to JSI_avg (with $\rho$ = 0.1, 0.25, 0.5, 0.8, 1) for $D = 400$. The curves for JSI_avg with $\rho = 1$ and for JSI_perf essentially coincide (only the former is shown), because for $\rho = 1$ (i.e., continuous jamming) $N_{0,\mathrm{eq}}(k)$ equals $N_{0,\mathrm{av}}$ for the entire observation window, and the estimate $\hat{N}_{0,\mathrm{av}}$ is very close to $N_{0,\mathrm{av}}$. With decreasing $\rho$, $\mathrm{SEP}_*/\rho$ for JSI_avg slightly increases compared to JSI_perf, because of the growing probability that the observation window is only partially hit by jamming, yielding $N_{0,\mathrm{av}}$ (and $\hat{N}_{0,\mathrm{av}}$) to be different from $N_{0,\mathrm{eq}}(k)$, for $k = 0,1,...,U+S-1$; the curves for JSI_avg with $\rho$ = 0.1, 0.25, 0.5 essentially coincide (only the former is shown). Comparing to Fig. 6, we observe that the practical full-search algorithms outperform the HD_thr(t) synchronizer, at the expense of additional complexity and higher delay. The synchronizers using estimated JSI (JSI_est(32) and JSI_avg) perform better than those not using JSI (JSI_no, HD). Among all practical synchronizers considered, JSI_avg provides the best performance, which for increasing $\rho$ gets very close to the JSI_perf performance. In spite of its somewhat higher complexity, JSI_est(32) is outperformed by JSI_avg. Among the full-search synchronizers not requiring JSI, JSI_no performs better than HD.

Finally, we relate the frame synchronizer performance to the overall decoder performance. We denote by $\mathrm{CER}_*$ the CW error rate (CER) in the absence of frame synchronization errors; $\mathrm{CER}_*$ for the LDPC(512,256) code is reported in Fig. 23.3 of [4], under pulsed jamming with $E_s/N_0$ = 7 dB, $Y = 5120$, $\rho \in \{1, 0.5, 0.05\}$ and various $E_s/J_{0,p}$, assuming a conventional belief propagation decoder performing a maximum of 100 iterations, whereby the unknown $E_s/N_{0,\mathrm{eq}}(k)$ were replaced by $E_s/(N_0 + J_0)$, with $N_0 + J_0$ denoting the long-term average of $N_{0,\mathrm{eq}}(k)$. When at a given operating point the condition $\mathrm{SEP}_* \ll \mathrm{CER}_*$ holds, the overall CER (considering decoding errors caused not only by noise and jamming but also by erroneous frame synchronization) will be very close to $\mathrm{CER}_*$: the overall CER is determined mainly by the error-correcting capability of the

[4] This condition is equivalent to $E_s/J_{0,p} > 0.1$. Considering the relation $\frac{E_s}{J_{0,p}} = \frac{E_s}{E_{J,p}} \cdot \frac{T_b}{T_c}$ and the large spreading factors ($\frac{T_b}{T_c}$ ranging from 10 to 1000) envisaged for future TC systems, values of $E_s/J_{0,p}$ lower than -10 dB correspond to unrealistically high jammer peak power levels.

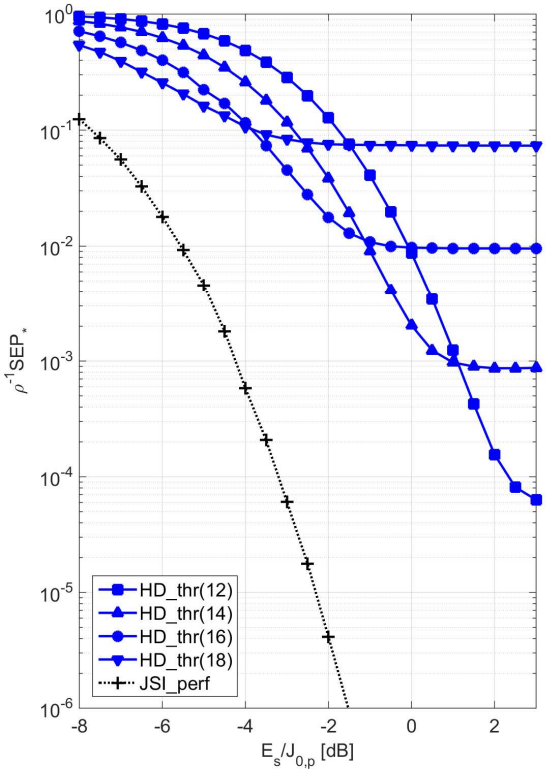


**FIGURE 6** $\rho^{-1}\mathrm{SEP}_*$ for HD_thr(t) and JSI_perf as a function of $\frac{E_s}{J_{0,p}}$, for $\frac{E_s}{N_0}$ = 7 dB, S = 64, U = 575.

code, and the effect of frame synchronization errors is negligible. On the other hand, when $\mathrm{SEP}_* > \mathrm{CER}_*$, the overall CER is dominated by the frame synchronization errors rather than the code. Typically, an overall CER equal to about $10^{-4}$ is considered as a target value for TC applications. For several $(\rho, \mathrm{D})$, Table 3 shows the operating point (denoted $\left(\frac{E_s}{J_{0,p}}\right)_{\mathrm{ref}}$), for which the LDPC(512,256) code achieves $\mathrm{CER}_* = 10^{-4}$, and, for the different synchronizers, the value of $\mathrm{SEP}_*$ (denoted $\mathrm{SEP}_{\mathrm{ref}}$) corresponding to this operating point; also displayed is the value of $\frac{E_s}{J_{0,p}}$ (denoted $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$) for which the considered synchronizers achieve $\mathrm{SEP}_* = 10^{-4}$. For the HD_thr synchronizer, we have selected the threshold value t that is optimum for the corresponding $\left(\frac{E_s}{J_{0,p}}\right)_{\mathrm{ref}}$. The entries related to the synchronizers are slightly pessimistic, as they are derived from Figs. 6 and 7 , which assume the worst-case value of D. We observe that all full-search synchronizers yield $\mathrm{SEP}_{\mathrm{ref}} \ll 10^{-4}$ and $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}} < \left(\frac{E_s}{J_{0,p}}\right)_{\mathrm{ref}}$, which indicates that operation at $\left(\frac{E_s}{J_{0,p}}\right)_{\mathrm{ref}}$ will result in an overall CER very close to $10^{-4}$, the CER corresponding to perfect frame synchronization; among these full-search synchronizers, the HD algorithm (using hard decisions and not requiring JSI) has the smallest complexity. On the other hand, the sequential hard-decision-based HD_thr synchronizer gives rise to $\mathrm{SEP}_{\mathrm{ref}} > 10^{-4}$ and $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}} > \left(\frac{E_s}{J_{0,p}}\right)_{\mathrm{ref}}$, which implies that operation at $\left(\frac{E_s}{J_{0,p}}\right)_{\mathrm{ref}}$ will be considerably affected by synchronization errors, yielding an overall CER exceeding $10^{-4}$.

## 8 | CONCLUSIONS

In view of the recent update of the CCSDS standard for satellite TC systems [1,13], we have considered the problem of detecting a known start sequence (SS) in an unknown symbol stream, in the presence of pulsed jamming. A first candidate algorithm is the sequential-search, hypothesis-testing frame synchronizer HD_thr previously considered in [12], which is demonstrated to exhibit only a limited robustness to pulsed jamming. To improve the synchronizer performance, five other frame synchronizers are considered. All of them are ML-based, full-search, peak detection algorithms, i.e., a metric is computed at all possible starting positions of the SS, and the maximum metric value is found. These full-search algorithms

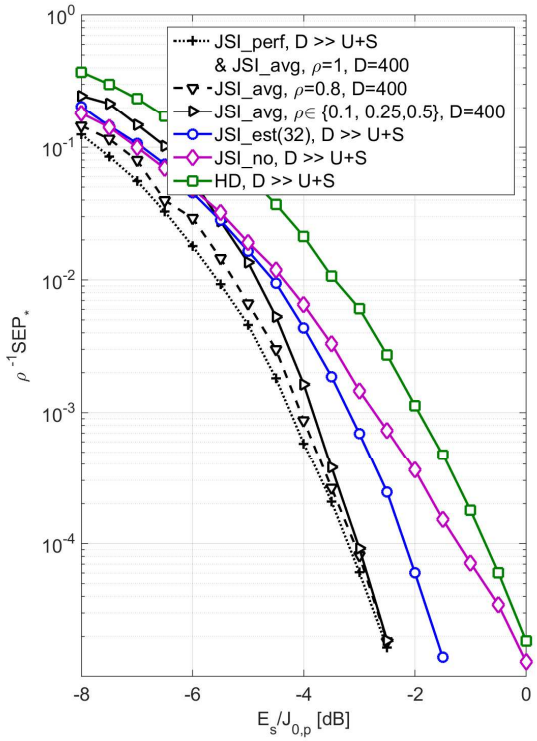


**FIGURE 7** $\rho^{-1}\mathrm{SEP}_*$ as a function of $\frac{E_s}{J_{0,p}}$, for $\frac{E_s}{N_0}$ = 7 dB, S = 64, and U = 575.

differ regarding the type of the jammer state information (JSI) required for their operation: known instantaneous JSI (not achievable in practice, but provides performance benchmark), estimated instantaneous JSI, estimated average JSI, and no JSI.

Numerical results for the specific 64-bit SS specified in [1] show that the effect of a realistic pulsed jammer, with a given duty cycle and a given average power, on the performance of most of the considered algorithms is worst for jammer pulses that are long as compared to the SS Moreover, the improvements offered by the proposed full-search algorithms over the sequential-search algorithm HD_thr are found to be considerable. It is shown that the relatively simple HD full-search frame synchronizer, which uses hard symbol-decisions and does not require JSI, guarantees a synchronization error probability that is small as compared to the code word error probability (for CERs near a target value of $10^{-4}$) in the presence of pulsed jamming with a duty cycle of 100%, 50% or 5%. The other full-search synchronizers outperform the HD synchronizer, at the expense of a higher complexity.

## ACKNOWLEDGMENT

Part of this work has been supported by the European Space Agency funded activity SatNEx IV COO1-PART 2 WI 2 ”Physical layer security”. The views of the authors of this paper do not reflect the views of ESA.

## References

1. CCSDS 231.0-B-3 TC synchronization and channel coding. Blue Book. Issue 3. September 2017.

2. Baldi M, Chiaraluce F, Garello R, Maturo N, Aguilar Sanchez I, Cioni S. Analysis and performance evaluation of new coding options for space telecommand links – part II: jamming channels. *International Journal on Satellite Communication and Networking.* 2015;33(6):527–542.

| | $(\rho, \mathrm{D})$ | (1,5120) | (0.5,2560) | (0.05,256) |
|---|---|---|---|---|
| LDPC(512,256) | $\left(\frac{E_s}{J_{0,p}}\right)_{\text{ref}}$ | 1 dB | 1.5 dB | -0.5 dB |
| JSI_perf | $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ | -3.25 dB | -3.5 dB | -4.25 dB |
| | $\mathrm{SEP}_{\text{ref}}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ |
| JSI_est(32) | $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ | -2.25 dB | -2.5 dB | -3.5 dB |
| | $\mathrm{SEP}_{\text{ref}}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ |
| JSI_avg | $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ | -3.25 dB | -3.25 dB | -4 dB |
| | $\mathrm{SEP}_{\text{ref}}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ |
| JSI_no | $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ | -1.25 dB | -1.75 dB | -3.25 dB |
| | $\mathrm{SEP}_{\text{ref}}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ |
| HD | $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ | -0.75 dB | -1 dB | -1.75 dB |
| | $\mathrm{SEP}_{\text{ref}}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ | $\ll 10^{-4}$ |
| HD_thr | $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ | 2.25 dB | 1.75 dB | 0.75 dB |
| | $\mathrm{SEP}_{\text{ref}}$ | $10^{-3}$ | $2 \cdot 10^{-4}$ | $2 \cdot 10^{-4}$ |

**TABLE 3** Comparison of the main factors affecting the TC loss rate: frame synchronization performance and decoding performance. Here, $\left(\frac{E_s}{J_{0,p}}\right)_{\text{ref}}$ denotes the operating point for which the LDPC(512,256) code achieves $\mathrm{CER}_* = 10^{-4}$ and $\mathrm{SEP}_{\text{ref}}$ denotes the value of $\mathrm{SEP}_*$ corresponding to this operating point. Further, $\left(\frac{E_s}{J_{0,p}}\right)_{10^{-4}}$ denotes the value of $\frac{E_s}{J_{0,p}}$ for which the considered synchronizers achieve $\mathrm{SEP}_* = 10^{-4}$.


3. Martinelli P, Cianca E, Simone L. Comparison of channel codes in presence of pulsed jammers in TT&C links. In: 7th Advanced Satellite Multimedia Systems Conference and 13th Signal Processing for Space Communincation Workshop (ASMS/SPSC):170–173; September 2014.
4. Garello R, Chiaraluce F. *Advanced coding schemes for direct sequence spread spectrum telecommand links*. Final Report: ESTEC; September 2013.
5. Massey J. Optimum frame synchronization. *IEEE Transactions on Communications*. April 1972;20(2):115–119.
6. Chiani M., Martini MG. On sequential frame synchronization in AWGN channels. *IEEE Transactions on Communications*. 2006;54(2):339–348.
7. Chiani M, Martini MG. Analysis of optimum frame synchronization based on periodically embedded sync words. *IEEE transactions on communications*. 2007;55(11):2056–2060.
8. Pfletschinger S, Navarro M, Closas P. Frame synchronization for next generation uplink coding in deep space communications. In: 2015 IEEE Global Commununications Conference (GLOBECOM); December 2015.
9. Currie RJ, Rushforth CK. Frame synchronization in the presence of burst jamming. In: IEEE Military Communications Conference (MILCOM):315–319; 1985.
10. Driessen PF. Performance of frame synchronization in packet transmission using bit erasure information. *IEEE Transactions on Communications*. 1991;39(4):567–573.
11. Rushforth CK, Currie RJ. Frame synchronization in the presence of errors and erasures. *IEEE Transactions on Aerospace and Electronic Systems*. July 1983;AES-19(4):498–505.
12. Noels N, Moeneclaey M. Performance of advanced telecommand frame synchronizer under pulsed jamming conditions. In: 2017 IEEE International Conference on Communications (ICC); May 2017.
13. CCSDS 230.1-G-2 TC synchronization and channel coding - summary of concept and rationale; Green Book. Issue 2. November 2012.
14. Liang Y, Rajan D, Eliezer OE. Sequential frame synchronization based on hypothesis testing with unknown channel state information. *IEEE Transactions on Communications*. 2015;63(8):2972–2984.
15. Simone L, Fittipaldi G, Aguilar Sanchez I. Fast acquisition techniques for very long PN codes for on-board secure TTC transponders. In: IEEE Military Communincation Conference (MILCOM):1748–1753; 2011.