Abstract—Decentralized slot synchronization protocols inspired from nature have recently gained considerable interest. By adjusting the internal time reference of a node in response to the detected timing of a received synchronization word, and by following simple rules, synchronization emerges from an initially completely uncoordinated situation. However these protocols typically assume that synchronization words are detected error free. In this paper the performance of biologically inspired slot synchronization is investigated when a realistic timing synchronization scheme is employed. We consider an ad hoc network where nodes communicate over an orthogonal frequency-division multiplexing (OFDM) air interface. The transmission is organized in frames, and each frame is preceded by a synchronization preamble with known repetitive parts. Specifically one common synchronization preamble is employed for all transmitting terminals, as, e.g., for the wireless LAN standard 802.11. Computer simulations verify that provided a sufficiently long synchronization word, reliable slot synchronization is maintained using a practical synchronization unit.

I. INTRODUCTION

Slot synchronization aims at aligning internal time references so that all nodes in the network agree on a common time slotted structure. By self-organized coordination among nodes, following simple rules, synchronization emerges from an initially random situation, so that packet transmissions are aligned in time.

Recent studies on decentralized time synchronization in ad hoc networks [1–3] are inspired by spontaneous synchronization of distributed biological systems, such as the synchronous flashing of fireflies observed in certain parts of the Southeast Asia. The underlying theory behind these phenomena is described by the synchronization of coupled oscillators [4], which interact upon detection of transmitted pulses from neighboring nodes. This presents the advantage in wireless networks that the exchange of timestamps is avoided, and that a synchronization word common to all nodes can be used, as there is no need to distinguish between transmitters. The only prerequisite is a node’s capability to accurately detect the timing of received pulses [1, 4], or synchronization words [2, 3], in order to properly adjust its time scale. This, however, is a critical assumption in practical applications, as there is no guarantee that a synchronization word or pulse can actually be detected error free.

Facing this challenge, we consider an ad hoc network where nodes communicate over an OFDM air interface, such as the Wireless LAN standard 802.11 [5]. In such systems each transmitted packet is typically composed of a preamble and of payload data. The synchronization preamble, which is common to all nodes, is needed to detect the start of a packet, adjust the automatic gain control and synchronize the receiver’s local oscillator in time and frequency to the transmitter. For the decentralized slot synchronization protocol in [3], the timing information of the preamble is also used for slot synchronization to update the internal clock.

Our contribution is to verify whether conventional OFDM timing synchronization schemes, devised for synchronization of an essentially isolated link, are applicable to decentralized slot synchronization protocol. To this end, we resort to the method proposed in [6] where a robust timing estimator is designed by searching for the correlation peak among the repetitive parts of the synchronization preamble. In spite of its effectiveness in a scenario where only one transmitter is active, this approach does not provide any guarantee to detect the arrival of a synchronization preamble in a multiuser asynchronous network. The main impairment is represented by the interference from other transmitting nodes, in the form of payload data and misaligned synchronization words. The interference tends to mask synchronization peaks and may also give rise to spurious peaks that do not correspond to any received preambles. This may disrupt the slot synchronization algorithm, as these spurious peaks may be misinterpreted as detected preambles from other nodes.

The remainder of this paper is structured as follows. In Section II biologically-inspired slot synchronization is introduced. Section III describes the link synchronization algorithm of [6] as well as the considered system model. Finally in Section IV the performance degradation on the link synchronization algorithm due to interference and its impact on the slot synchronization algorithm of [3] are evaluated through simulations.

II. BIOLOGICALLY-INSPIRED SLOT SYNCHRONIZATION

A. Synchronization of PCOs

The adopted distributed slot synchronization protocol is based on the theory of pulse-coupled oscillators (PCOs) [4]. Consider a network of $S$ nodes each acting as a PCO. The state of the $i$th PCO is described by its phase function $\phi_i(t)$
evolving as follows [4]
\[ \phi_i(t) = \frac{\phi_{th}}{T_i} \cdot t + \phi_i(0) \]  
(1)

where \( \phi_i(0) \) is the state at time \( t = 0 \), \( \phi_{th} \) is a threshold value and \( T_i \) is the cycle period of an uncoupled PCO [4].

At time instant \( t = \tau_i \) for which \( \phi_i(\tau_i) = \phi_{th} \), the \( i \)th PCO is said to fire as it performs the two following operations: it emits a pulse and resets the state of its phase function to zero, \( \phi_i(\tau_i^+) = 0 \), where \( \tau_i^+ \) indicates an infinitesimal time instant after \( \tau_i \). If not coupled to any other oscillator, each node keeps naturally firing with a constant period equal to \( T_i \).

When coupled to others, the phase function of each oscillator is perturbed by the reception of pulses transmitted by its neighbors. In [4] it is assumed that pulses are infinitesimally short and coupling between nodes is instantaneous. Consequently, a node cannot receive a pulse when firing. Under these circumstances, when oscillator \( j \) fires at \( t = \tau_j \), all receiving nodes in the neighborhood adjust their phase functions as follows:
\[ \phi_i(\tau_j^+) = \phi_i(\tau_j) + \Delta \phi(\phi_i(\tau_j)) . \]  
(2)
The phase increment \( \Delta \phi \) depends on the phase at the instant of receiving a pulse. A simple but yet effective response leading to synchrony is determined by the linear phase function
\[ \phi_i + \Delta \phi(\phi_i) = \min (\alpha \cdot \phi_i + \beta, \phi_{th}) \]  
(3)
where \( \alpha \) and \( \beta \) are coupling parameters, which are common to all. Under these conditions, all nodes synchronize, i.e., they agree on a common firing instant, independent of the number of active PCOs and for any set of initial states \{\( \phi_i(0) \}\}, provided that \( \alpha > 1 \) and \( 0 < \beta < \phi_{th} \) [4].

B. Meshed Emergent Firefly Synchronization

In the ensuing discussion, the PCO model described in the previous section can be modified to be employed in a wireless communication network.

In [1] it was demonstrated that propagation delays can make a system of PCOs unstable. Instability can be avoided by introducing a refractory period \( T_r \), a time interval added after the firing instant in which a node cannot increment its phase function upon reception of a pulse. Denoting \( \theta^{(i,j)} \) the propagation delay between nodes \( i \) and \( j \), it can be shown that for convergence, \( T_i \) must be larger than the maximum two-way propagation delay, that is
\[ T_i > 2 \cdot \max_{i,j} \left\{ \theta^{(i,j)} \right\} . \]  
(4)

Multicarrier signals are typically composed of transmission blocks with non-negligible duration. These blocks delay the coupling between nodes, which is considered instantaneous in [4]. To combat transmission delays a delay tolerant approach was devised in [2]. In [3] the synchronization protocol was further modified to fit a frame structure, which is composed of a synchronization preamble denoted \( \text{sync} \) of duration \( T_{w} \), followed by payload data denoted \( \text{data} \) of duration \( T_{d} \), with \( T_{w} + T_{d} = T_{t} \), referred to as meshed emergent firefly synchronization (MEmFiS). The considered transmit slot is shown in Fig. 1(a).

![Fig. 1. (a) Transmit slot and (b) Listen slot in MEmFiS.](image)

Fig. 1(b) presents the receiving period of a node. A receiving slot in MEmFiS is composed of a refractory period of length \( T_r \) and a listening period of duration \( T_{l} \), with \( T_r + T_{l} = T_{t} \). During the listening period the phase function (1) grows linearly and is adjusted upon detection of synchronization preambles transmitted by other nodes.

With MEmFiS a node enters a transmit slot when a data packet is ready for transmission, and follows the rules of the listen slot when no data is present. Thus, nodes synchronize as data is exchanged, and unslotted packet transmissions seamlessly turn to slotted transmissions. However, whilst nodes are not synchronized, multiple synchronization preambles are not aligned in time may be received, which affects their detection. The remainder of this paper examines the impact of misaligned reception of simultaneously received common synchronization preambles.

III. DETECTION OF THE SYNCHRONIZATION WORD IN A MULTICARRIER AD HOC NETWORK

In this section, OFDM synchronization algorithms are reviewed. The synchronization unit scans the received signal stream in order to detect synchronization preambles that mark the start of a new packet. To facilitate network timing synchronization by means of the biological-inspired protocol described in the previous section, the synchronization unit cannot rely on any a priori information, such as the timing of previously received packets.

A. System Model

Without loss of generality, we consider one node that receives signals from \( M \) neighboring nodes. Each frame is composed of \( N_B \) data blocks. We adopt a baseband-equivalent discrete-time signal model with sampling period \( T_s = T/N \), where \( N \) is the number of available subcarriers and \( 1/T \) the subcarrier spacing. The time-domain samples of the \( m \)th user, \( 1 \leq m \leq M \), during the \( i \)th OFDM block are expressed by
\[ s_{m,i}(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} c_{m,i}(n) \cdot e^{j2\pi nk/N}, \quad -N_k \leq k \leq N-1 \]  
(5)
where \( N_k \) is the length of the cyclic prefix (CP) expressed in sampling periods while \( c_{m,i}(n) \) is the modulated symbol over
the $n$th subcarrier. The signal transmitted by the $m$th terminal is the concatenation of $N_{B}$ adjacent blocks and is given by

$$s_m(k) = \sum_{i=0}^{N_{B}-1} s_{m,i}(k-i\cdot N_{T})$$

(6)

where $N_{T} = N + N_{g}$ is the block length including the CP. Each stream $s_m(k)$ propagates through a multipath channel characterized by an impulse response $h_m = [h_m(0), h_m(1), \ldots, h_m(L-1)]^{T}$ of length $L$ (in sampling periods), and arrives at the considered terminal with some synchronization mismatch relative to the local time and frequency scales. We denote $\theta_m$ the timing offset, i.e., the timing misalignment of the frame start to the $m$th user, expressed in sampling periods, while $\varepsilon_m$ is the frequency offset normalized to the subcarrier spacing. After baseband conversion and sampling, the signal stream at the receiving terminal is in the form

$$r(k) = \sum_{m=1}^{M} r_m(k-\theta_m) + w(k)$$

(7)

where $w(k)$ represents complex-valued additive white Gaussian noise (AWGN) with variance $\sigma^2_w$ while $r_m(k)$ is the signal from the $m$th user, which reads

$$r_m(k) = e^{j2\pi \varepsilon_m \cdot k/N} \sum_{\ell=0}^{L-1} h_{m}(\ell) \cdot s_{m}(k-\ell).$$

(8)

A desirable property of a timing estimation scheme is its robustness to frequency offsets. If the training block exhibits a repetitive structure in the time-domain, then a robust timing estimator can be designed by searching for the correlation peak among the repetitive parts. Solutions in this sense have been proposed in [6, 7], where the synchronization preamble is composed of several identical parts with possible sign inversions. In this work we assume that the training block is the same for each user, and is composed of $Q$ repeated segments $s_q$, $0 \leq q \leq Q-1$, in the time-domain. Each segment has $P$ samples and is expressed by

$$s_q = b_q \cdot s$$

(9)

where $s$ is a $P$-dimensional vector independent of $q$, while $b_q \in \{\pm 1\}$ are bipolar symbols which determine the training pattern $b = [b_0, b_1, \ldots, b_{Q-1}]^{T}$. The training block is preceded by a CP of length $N_{g}$ and has a total of $N_{1} = N_{g} + P \cdot Q$ samples. Hence, the number of samples in each frame is found to be $N_{F} = N_{1} + N_{B} \cdot N_{T}$, and the duration of the frame is equal to $T_{f} = N_{F} \cdot T$.

B. Timing Metric

The algorithm of [6] is preferred to other OFDM synchronization algorithms, because its timing metric exhibits a high and narrow peak when a synchronization preamble is present. This way, when two preambles are relatively close, which is typical for a multiuser asynchronous scenario, the metric should exhibit two peaks. This section describes how the timing metric is generated.

Let $k = 0$ denote the start of a frame in the time scale of the considered terminal, which observes the received signal $r(k)$ expressed by (7) and (8).

A timing metric is computed for each frame and collected into a vector of dimension $N_{F}$. More precisely, we denote $\Lambda_j = [\Lambda_j(0), \Lambda_j(1), \ldots, \Lambda_j(N_{F} - 1)]^{T}$ the metric obtained during the $j$th frame ($j = 0, 1, 2, \ldots$) with entries [6]

$$\Lambda_j(k) = \frac{Q}{Q-1} \cdot \frac{|\lambda_j(k)|}{E_j(k)}$$

for $0 \leq k \leq N_{F} - 1$

(10)

where $\lambda_j(k)$ and $E_j(k)$ are defined as

$$\lambda_j(k) = \sum_{q=0}^{Q-2} b_q \cdot b_{q+1} \cdot q [k + q \cdot P + j \cdot N_{F}]$$

(11)

with

$$q[k] = \sum_{p=0}^{P-1} r[k + P + p].r^{*}[k + p]$$

and

$$E_j(k) = \sum_{q=0}^{Q-1} \sum_{p=0}^{P-1} |r[k + q \cdot P + p + j \cdot N_{F}]|^{2}$$

(12)

where the superscript * denotes the complex conjugate and $|\cdot|$ denotes the absolute value of the enclosed quantity. Fig. 2 illustrates a realization of $\Lambda_j(k)$ in the case of two and four active nodes with delays $\theta_m = 160 + 100 \cdot m$ samples, for $m = 0, 1, 2, 3$. The training block is composed of 16 segments with the following pattern

$$b = [1, -1, -1, 1, 1, -1, -1, 1, 1, -1, -1, 1, 1, -1, -1, 1, 1]^{T}$$

(13)

Each segment is made of 16 samples while the CP has length $N_{g} = 32$. This corresponds to a training block with overall duration $N_{1} = 288$. The number of subcarriers per OFDM symbol is set to $N = 128$. The channel responses $h_m$ are modeled as statistically independent Gaussian random vectors with zero-mean and an exponentially decaying power delay profile. The channels have length $L = 8$ and the average received energy is the same for each user. For simplicity, the frequency offsets are set to zero and thermal noise is neglected.

As expected, in Fig. 2 the timing metric exhibits a peak whenever a synchronization preamble of a given user arrives at the receiving terminal. However, the presence of data blocks from the other nodes interferes with the synchronization preamble, which tends to mask the synchronization peaks and also gives rise to spurious peaks. This is especially visible for $M = 4$ where the spurious peak around $k = 600$ is as high as the synchronization peak at $k = 160$.

In the following we investigate whether the timing metric given in (10)–(12) can be employed to provide a real-time decision about the presence of a synchronization preamble in the received signal. The timing metric $\Lambda_j(k)$ is computed at each sampling instant $k \cdot T_{c}$, using the last $P \cdot Q$ received samples $r_k = [r(k), r(k-1), \ldots, r(k-P \cdot Q-1)]^{T}$ as indicated in (10)–(12). The arrival of a synchronization preamble is then declared as soon as $\Lambda_j(k)$ overcomes a suitable threshold

$$\Lambda_j(k) \geq \gamma.$$

(14)
The choice of an appropriate threshold $\gamma$ is important, as $\gamma$ makes hard decisions whether correlation peaks of $\Lambda_j(k)$ in (10) are declared as a valid synchronization preamble or not.

IV. SIMULATION RESULTS

A. False Alarm and Missed Detection Probabilities

The detection of a synchronization preamble as outlined in the previous section, is evaluated through false alarm and detection probabilities. The synchronization word detector tests two hypotheses:

- $\mathcal{H}_0$: no signal or data was transmitted.
- $\mathcal{H}_1$: a synchronization word was transmitted.

The false alarm probability is defined as the probability of choosing $\mathcal{H}_1$ when $\mathcal{H}_0$ is true, which can be written as

$$P_{fa} = p(\Lambda_j(k) \geq \gamma | \mathcal{H}_0) \quad (15)$$

A false alarm corresponds to an unwanted peak that is caused by correlation of noise, data or a partial synchronization word with the synchronization word.

Fig. 3 plots the false alarm probability as a function of the detection threshold $\gamma$ for $M=2$ and $M=8$ simultaneously transmitting nodes, which are equally spaced by 200 samples. The training block is composed of $Q$ identical segments $s_q$, while each segment is composed of $P=32$ samples. The total frame length $N_F$ is set such that the preamble corresponds to 10% of a frame. From Fig. 3, for a low detection threshold $\gamma$, the false alarm probability $P_{fa}$ increases when the number of simultaneous transmitters is increased from $M=2$ to $M=8$. However increasing the detection threshold $\gamma$ lowers the gap between $M=8$ and $M=2$, and therefore effectively reduces the false alarm probability. Indeed, as the threshold increases, only very high peaks are acknowledged. From Fig. 3 these are less likely to occur when $M=8$, because of the high interference level coming from the superimposed data from these nodes. However setting the threshold to a high value discards some synchronization peaks, and increases the probability of missing synchronization words.

The missed detection probability is defined as the probability of choosing $\mathcal{H}_0$ when $\mathcal{H}_1$ is true

$$P_{md} = p(\Lambda_j(k) \leq \gamma | \mathcal{H}_1) \quad (16)$$

Fig. 4 plots the missed detection probability as a function of $\gamma$ under the same conditions as in Fig. 3. From Fig. 4 the missed detection rate significantly grows, when the number of simultaneous transmitters is increased from $M=2$ to $M=8$. This can be attributed to the fact that the missed detection probability is more sensitive to interference from data as $M$ increases, as spurious synchronization peaks are more likely to occur. Thus for a threshold set to $\gamma=0.4$, $P_{md}$ approaches 20% when $M=8$, whereas $P_{md}<10^{-4}$ when $M=2$.

B. Performance Evaluation of MEmFiS

The synchronization word detector is designed according to the Neyman-Pearson criterion [8]. First the detection threshold $\gamma$ in (14) is fixed for a given false alarm rate $P_{fa}$, and then the detection rate is evaluated $(1-P_{md})$. In particular we are interested in deriving the detection probability for a constant false alarm probability per slot. Given $N_F$ detection instants
the time to synchrony. Fig. 5 plots the cumulative distribution function in (3). The synchronization preamble composed of \( Q \) repetitions of \( P \) for a Poisson distribution with mean arrival rate of \( \lambda \) for \( Q \) threshold is set assuming \( M \) the following, a cautious approach is taken, and the detection activity varies from \( P \). Consequently, the missed detection probability varies from \( P_{\text{md}}=0.2 \) when eight nodes are transmitting simultaneously to below \( 10^{-4} \) for two nodes.

Simulations of MEmFiS are conducted for 5,000 sets of initial conditions \( \{\phi_i(0)\} \) in (1). The arrival of packets follows a Poisson distribution with mean arrival rate of \( \lambda=5 \) packets per slot, and the coupling settings are set to \( \alpha=1.2 \) and \( \beta=0.01 \) in (3). The synchronization preamble composed of \( Q=32 \) repetitions of \( P=32 \) samples each, and represents 10% of a transmit slot. Fig. 5 plots the cumulative distribution function (cdf) of the time to synchrony \( T_{\text{sync}} \) normalized to the slot duration \( T_i \) for a fully-connected network of 16 nodes, i.e., nodes can communicate directly with all others.

Choosing the detection threshold \( \gamma \) in (14) is important. In the following, a cautious approach is taken, and the detection threshold is set assuming \( M=8 \) in Fig. 3. So, for example for \( Q=16 \), if \( P_{fa} \) is fixed to \( 10^{-3} \), the threshold would be set to \( \gamma=0.36 \). Consequently, the missed detection probability is \( 10^{-4} \) for two nodes.

Simulations of MEmFiS are conducted for 5,000 sets of initial conditions \( \{\phi_i(0)\} \) in (1). The arrival of packets follows a Poisson distribution with mean arrival rate of \( \lambda=5 \) packets per slot, and the coupling settings are set to \( \alpha=1.2 \) and \( \beta=0.01 \) in (3). The synchronization preamble composed of \( Q=32 \) repetitions of \( P=32 \) samples each, and represents 10% of a transmit slot. Fig. 5 plots the cumulative distribution function (cdf) of the time to synchrony \( T_{\text{sync}} \) normalized to the slot duration \( T_i \) for a fully-connected network of 16 nodes, i.e., nodes can communicate directly with all others.

Choosing the detection threshold \( \gamma \) in (14) is important. In the following, a cautious approach is taken, and the detection threshold is set assuming \( M=8 \) in Fig. 3. So, for example for \( Q=16 \), if \( P_{fa} \) is fixed to \( 10^{-3} \), the threshold would be set to \( \gamma=0.36 \). Consequently, the missed detection probability varies from \( P_{\text{md}}=0.2 \) when eight nodes are transmitting simultaneously to below \( 10^{-4} \) for two nodes.

Simulations of MEmFiS are conducted for 5,000 sets of initial conditions \( \{\phi_i(0)\} \) in (1). The arrival of packets follows a Poisson distribution with mean arrival rate of \( \lambda=5 \) packets per slot, and the coupling settings are set to \( \alpha=1.2 \) and \( \beta=0.01 \) in (3). The synchronization preamble composed of \( Q=32 \) repetitions of \( P=32 \) samples each, and represents 10% of a transmit slot. Fig. 5 plots the cumulative distribution function (cdf) of the time to synchrony \( T_{\text{sync}} \) normalized to the slot duration \( T_i \) for a fully-connected network of 16 nodes, i.e., nodes can communicate directly with all others.

Choosing the detection threshold \( \gamma \) in (14) is important. In the following, a cautious approach is taken, and the detection threshold is set assuming \( M=8 \) in Fig. 3. So, for example for \( Q=16 \), if \( P_{fa} \) is fixed to \( 10^{-3} \), the threshold would be set to \( \gamma=0.36 \). Consequently, the missed detection probability varies from \( P_{\text{md}}=0.2 \) when eight nodes are transmitting simultaneously to below \( 10^{-4} \) for two nodes.

Simulations of MEmFiS are conducted for 5,000 sets of initial conditions \( \{\phi_i(0)\} \) in (1). The arrival of packets follows a Poisson distribution with mean arrival rate of \( \lambda=5 \) packets per slot, and the coupling settings are set to \( \alpha=1.2 \) and \( \beta=0.01 \) in (3). The synchronization preamble composed of \( Q=32 \) repetitions of \( P=32 \) samples each, and represents 10% of a transmit slot. Fig. 5 plots the cumulative distribution function (cdf) of the time to synchrony \( T_{\text{sync}} \) normalized to the slot duration \( T_i \) for a fully-connected network of 16 nodes, i.e., nodes can communicate directly with all others.

Choosing the detection threshold \( \gamma \) in (14) is important. In the following, a cautious approach is taken, and the detection threshold is set assuming \( M=8 \) in Fig. 3. So, for example for \( Q=16 \), if \( P_{fa} \) is fixed to \( 10^{-3} \), the threshold would be set to \( \gamma=0.36 \). Consequently, the missed detection probability varies from \( P_{\text{md}}=0.2 \) when eight nodes are transmitting simultaneously to below \( 10^{-4} \) for two nodes.

Simulations of MEmFiS are conducted for 5,000 sets of initial conditions \( \{\phi_i(0)\} \) in (1). The arrival of packets follows a Poisson distribution with mean arrival rate of \( \lambda=5 \) packets per slot, and the coupling settings are set to \( \alpha=1.2 \) and \( \beta=0.01 \) in (3). The synchronization preamble composed of \( Q=32 \) repetitions of \( P=32 \) samples each, and represents 10% of a transmit slot. Fig. 5 plots the cumulative distribution function (cdf) of the time to synchrony \( T_{\text{sync}} \) normalized to the slot duration \( T_i \) for a fully-connected network of 16 nodes, i.e., nodes can communicate directly with all others.

Choosing the detection threshold \( \gamma \) in (14) is important. In the following, a cautious approach is taken, and the detection threshold is set assuming \( M=8 \) in Fig. 3. So, for example for \( Q=16 \), if \( P_{fa} \) is fixed to \( 10^{-3} \), the threshold would be set to \( \gamma=0.36 \). Consequently, the missed detection probability varies from \( P_{\text{md}}=0.2 \) when eight nodes are transmitting simultaneously to below \( 10^{-4} \) for two nodes.