Pulse-Domain Signal Parsing and Neural Computation

Hans-Andrea Loeliger and Sarah Neff
ETH Zurich
Dept. of Information Technology & Electrical Engineering
{loeliger, neff}@isi.ee.ethz.ch

***Abstract*—We propose a new model of pulse-based computation based on inner-product filters with linear-system kernels. Each inner-product filter looks for some pulse pattern in its multichannel-input signal by projecting the input signal into a one-dimensional subspace; an output pulse is generated if this projection exceeds some threshold. A layered network of such filters can be used for self-synchronizing multiscale signal parsing. Such a network can be built with computational units that are biologically plausible neurons. The feasibility of the proposed approach is demonstrated with a network that understands Morse code.**

I. Introduction

Consider pattern detection in a (possibly multichannel) time signal by a network as shown in Figure 1. Each "filter" in Figure 1 is looking for some feature in its input signals and produces as output some sort of score signal, which may in turn be analyzed by subsequent "filters".

The investigation of such an architecture was begun in [1] (following a suggestion in [2]). As pointed out in [1], the format of the intermediate score signals is crucial, and a key insight in [1] is the observation that robust functionality can be achieved with pulse signals—signals consisting of unit pulses separated by some guard space—and not easily otherwise.

In this paper, we develop this approach further. We will focus on pulse-domain processing, i.e., processing on the right of the dashed line in Figure 1. Each feature detection filter will be looking for some multichannel pulse pattern with specific relative arrival times of the pulses in the different channels as illustrated in Figure 2.

We will not discuss the first layer in Figure 1, which extracts pulse-domain feature signals from the input signal(s) of the network. For these first-layer filters, we refer to the discussion of feature detection filters in [1] and [3].

We will show that the pulse domain allows us to use especially simple feature detection filters, which we call inner-product filters and which amount to projecting the signal to a one-dimensional subspace. We will also show how such filters can be designed to detect a given pulse pattern.

As pointed out in [1], networks of such feature detection filters are self-timed and tolerate small variations in the timing of the input pulses. In fact, such networks could be implemented by clockless continuous-time analog circuits.

In [1], the feasibility of the proposed approach was demonstrated with a network that "understands" Morse code. This network has now been redesigned to work with the simpler inner-product filters of this paper. The new filters allow much more design, and require less experimentation, than the filters used in [1].

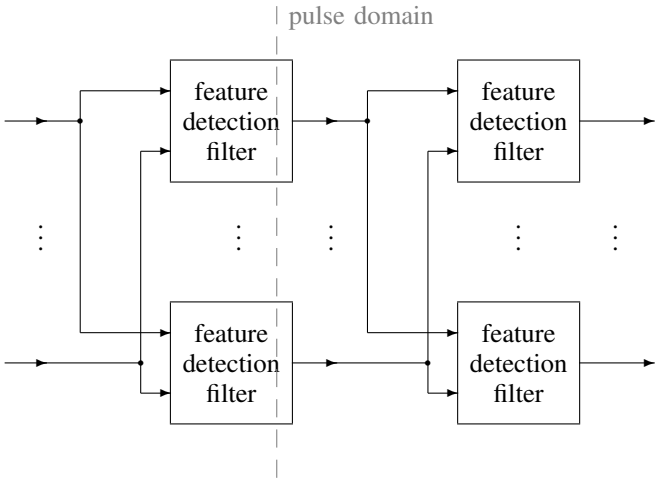

Fig. 1. Network of feature detection filters with pulse-domain processing beyond the first layer.

Finally, we observe that such inner-product filters can be implemented with biologically plausible neurons. We thus propose a new model of neural information processing where each neuron targets some particular temporal pulse pattern as in Figure 2.

For general background on neural networks with spiking neurons (both artificial and real), see [4]–[7].

This paper does not address learning such networks from data, which is an obvious topic of future research. Also, in this paper, we consider only hierarchical networks (as suggested by Figure 1) that have no loops.

The paper is structured as follows. Section II introduces a new class of pulse-domain feature detection filters. The design of such a filter for a particular pulse pattern is addressed in Section III. An example of a complete network is outlined in Section IV. The implementation of such feature detection filters using neuron models from biology is addressed in Section V.

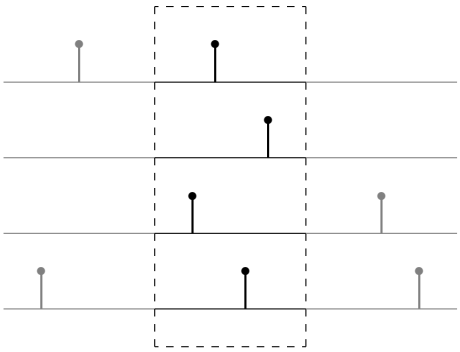

Fig. 2. A four-channel pulse pattern (inside the box), a possible target for an inner-product filter.

II. Pulse-Domain Inner-Product Filters

A. General Form

Let $y_1, y_2, \ldots \in \{0,1\}^L$ be the (pulse-domain) multichannel input signal of some feature detection filter. The main ingredient of feature detection filters as in [1] and [3] is a model signal $\tilde{y}_1, \tilde{y}_2, \ldots \in \mathbb{R}^L$ originating from an autonomous deterministic state space model

$$\tilde{y}_k = CA^{k-n}x_n \tag{1}$$

with $A \in \mathbb{R}^{m\times m}$, $C \in \mathbb{R}^{L\times m}$, and $x_n \in \mathbb{R}^m$. The matrix A is assumed to be regular, which means that the time-n state x_n completely determines $\tilde{y}_k$ for all k.

In this paper, we will exclusively use feature detection filters that compute the inner-product signal $\langle y, \tilde{y}\rangle_1, \langle y, \tilde{y}\rangle_2, \ldots \in \mathbb{R}$ defined as

$$\langle y, \tilde{y}\rangle_n \triangleq \sum_{k=1}^{n} y_k^{\mathsf{T}} \tilde{y}_k \tag{2}$$

for some fixed $x_n = s$. The model signal $\tilde{y}$ thus serves as a weighting kernel for the pulses in y. We will assume that all eigenvalues of A are strictly larger than 1, which implies that the sum (2) converges for $n \to \infty$, and we are primarily interested in the stationary case $n \gg 1$ where border effects can be neglected.

The feature detection filter produces a unit pulse at time n if $\langle y, \tilde{y}\rangle_n$ exceeds some threshold, whereupon any further pulses are suppressed for the duration of some guard interval.

The inner-product signal $\langle y, \tilde{y}\rangle_n$ is not actually computed using (2), but (more efficiently) by means of the quantity

$$\xi_n^{\mathsf{T}} \triangleq \sum_{k=1}^{n} y_k^{\mathsf{T}} CA^{k-n}. \tag{3}$$

We then have the recursion

$$\xi_n = (A^{-1})^{\mathsf{T}}\xi_{n-1} + C^{\mathsf{T}} y_n \tag{4}$$

and

$$\langle y, \tilde{y}\rangle_n = s^{\mathsf{T}}\xi_n. \tag{5}$$

Using (4) and (5), the inner-product signal is computed by a linear filter in state space form.

Such inner-product filters may be viewed as a (substantial) simplification of filters as in [1, eq. (20)]; this simplification is not possible for general signals, but it works fine in the pulse domain.

B. Two Examples

The following example is similar to, but improves upon, Example 3 of [1].

Example 1 Suppose we want to detect the particular pulse pattern shown in Figure 3, where three pulses arrive at different times in three different channels. The weighting signal $\tilde{y}$ of a suitable inner-product filter is shown in Figure 4. This weighting signal may be obtained by

$$A = \lambda \begin{pmatrix} \cos(\Omega) & -\sin(\Omega) \\ \sin(\Omega) & \cos(\Omega) \end{pmatrix} \tag{6}$$

with $\Omega = 2\pi/400$, $\lambda = 1.0025$, $s = (1,0)^{\mathsf{T}}$, and with a matrix C as described in Section III-A. The corresponding inner-product signal is shown in Figure 5 together with a suitable threshold for triggering the output pulse. □

The next example shows that the matrix A need not have complex eigenvalues.

Example 2 Suppose we want to detect the same pulse pattern (Figure 3) as in Example 1, but now with the matrix

$$A = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \tag{7}$$

with eigenvalues $\lambda_1 = 1.005$, $\lambda_2 = 1.012$, and $\lambda_3 = 1.015$. The matrix C has the form

$$C = \begin{pmatrix} c_{1,1} & c_{1,2} & 0 \\ 0 & c_{2,2} & c_{2,3} \\ 0 & 0 & c_{3,3} \end{pmatrix} \tag{8}$$

with nonzero entries as described in Section III-B. The resulting weighting signal $\tilde{y}$ is shown in Figure 6 and the corresponding inner-product signal is shown in Figure 7. □

C. Remarks

1) The proper functioning of such filters requires the pulses in each channel to be sufficiently separated.
2) We chose to describe inner-product filters in discrete time, but the translation to continuous time is obvious. This applies, in particular, to both examples in Section II-B. Moreover, such filters can, in principle, be implemented by analog circuits.
3) It is obvious from Figures 4 and 6 that such filters can tolerate some variation of the pulse positions around their nominal positions.
4) Simple filters as in Examples 1 and 2 can easily cope with high-dimensional input signals, i.e., $L \gg 1$.
5) For $L \gg 1$, due to the linearity of (4) and (5), such a filter can generically cope both with erasures (i.e., missing pulses) and with some random extra pulses in its input signals.

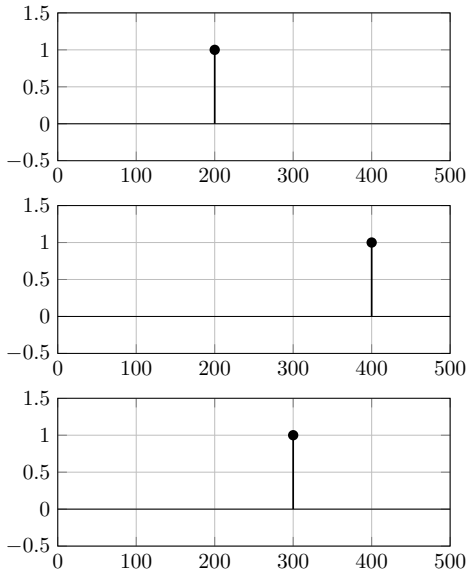

Fig. 3. Three-channel pulse pattern of Examples 1 and 2.

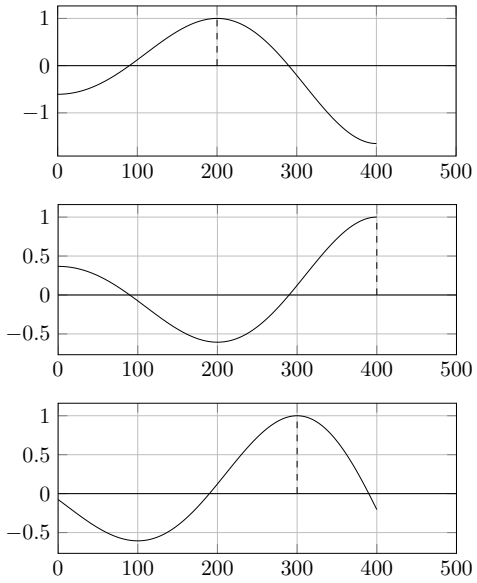

Fig. 4. Example 1: the weighting signal $\tilde{y}$ (for $n = 400$) in all three channels.

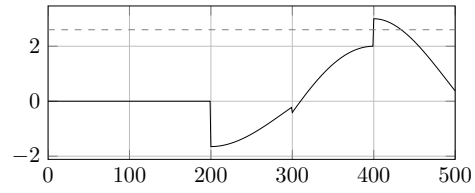

Fig. 5. Example 1: the inner-product signal (2) (for $n = 1, \ldots, 500$) for the filter of Figure 4 when fed with the clean signal of Figure 3. The dashed line is a suitable threshold for triggering the output pulse.

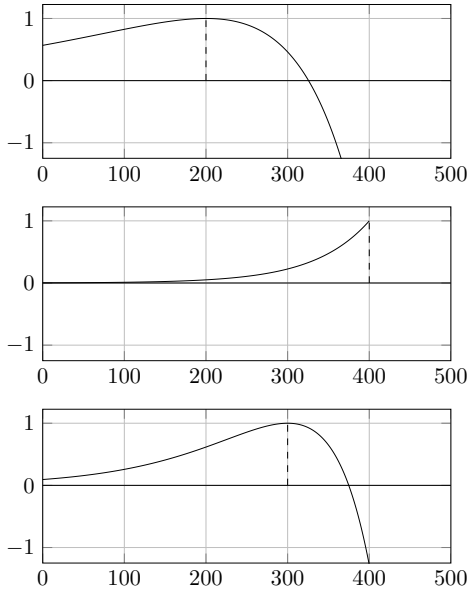

Fig. 6. Example 2: the weighting signal $\tilde{y}$ (for $n = 400$) in all three channels.

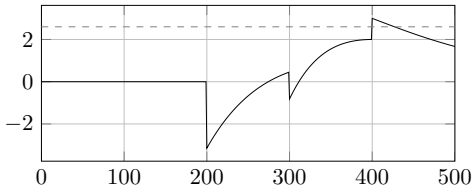

Fig. 7. Example 2: the inner-product signal (2) (for $n = 1, \ldots, 500$) for the filter of Figure 6 when fed with the clean signal of Figure 3. The dashed line is a suitable threshold for triggering the output pulse.

III. Filters for Monomial Pulse Patterns

A multichannel signal will be called *monomial* if it is zero everywhere except for (at most) one unit pulse in every channel. For example, the pulse pattern in Figure 3 is monomial.

The *preferred monomial pattern* of a given inner-product filter is the monomial signal y that maximizes $\max_n \langle y, \tilde{y} \rangle_n$ (the peak amplitude of the inner-product signal) among all monomial patterns. For example, it is obvious from Figures 4 and 6 that the preferred monomial pattern for the inner-product filters of Examples 1 and 2 is Figure 3.

Note that inner-product filters (as in Examples 1 and 2) that are designed to recognize some monomial pattern work perfectly well also with non-monomial signals provided that the pulses in each channel are sufficiently separated.

We now address the design of inner-product filters that prefer a given monomial pulse pattern $y = y_1, y_2, \ldots \in \mathbb{R}^L$. We will assume that there is a pulse in every channel, that the pulse in channel $\ell \in \{1, \ldots, L\}$ occurs at time $k_\ell \leq n$, and that we wish to recognize the pulse pattern at time n.

We will address the design of the corresponding filters both for generalizations of Examples 1 and for generalizations of Example 2. In both cases, we will assume that the matrix

$A \in \mathbb{R}^{m\times m}$ is given; the task is to construct $C \in \mathbb{R}^{L\times m}$ and $s \in \mathbb{R}^m$ such that the resulting filter prefers (if possible) the given monomial pulse pattern.

A. Sinusoidal Filters

For $m = 2M$, let

$$A = \begin{pmatrix} J_1 & 0 & 0 & \dots & 0 \\ 0 & J_2 & 0 & \dots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \dots & 0 & J_M \end{pmatrix}, \tag{9}$$

where $J_j \triangleq \lambda_j \operatorname{rotm}(\Omega_j)$ with $\lambda_j \geq 1$, $j = 1, \dots, M$, and

$$\operatorname{rotm}(\Omega) \triangleq \begin{pmatrix} \cos(\Omega) & -\sin(\Omega) \\ \sin(\Omega) & \cos(\Omega) \end{pmatrix}. \tag{10}$$

The matrix C with rows $c_1, \dots, c_L$ is then given by

$$c_\ell \triangleq s^{\mathsf{T}} \begin{pmatrix} R_1 & 0 & 0 & \dots & 0 \\ 0 & R_2 & 0 & \dots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \dots & 0 & R_{m/2} \end{pmatrix} A^{n-k_\ell}, \tag{11}$$

where $s \triangleq (1, 0, 1, 0, \dots, 1, 0)^{\mathsf{T}} \in \mathbb{R}^m$ and where $R_j \triangleq \frac{\operatorname{rotm}(\phi_j)}{\cos(\phi_j)}$ with $\phi_j \triangleq \arctan\left(\frac{\ln(\lambda_j)}{\Omega_j}\right)$.

Note that the target pulse pattern enters the filter design only by the term A^{n-k_ℓ} in (11).

Theorem 1. If

$$\max_{\ell\in\{1,\dots,L\}} (n - k_\ell) < \min_{\Omega\in\{\Omega_1,\dots,\Omega_M\}} \frac{3\pi/2}{\Omega} \tag{12}$$

then this inner-product filter prefers the given pulse pattern and the corresponding maximum value of the inner-product signal is $Lm/2$. □

The proof amounts to verifying that (11) results from maximizing (2) separately for each channel and for each frequency (as in Figure 4). The details are omitted due to space constraints. The condition (12) is conservative, but some such condition is necessary.

B. Filters with Simple Real Poles

Let A be a real diagonal matrix with different diagonal elements $\lambda_j > 1$, $j = 1, \dots, m$. Let $s \triangleq (1, 1, \dots, 1)^{\mathsf{T}} \in \mathbb{R}^m$ and let C be a matrix with rows $c_1, \dots, c_L \in \mathbb{R}^m$ as follows. If $k_\ell = n$, then

$$c_\ell \triangleq (0, \dots, 0, 1, 0, \dots, 0)^{\mathsf{T}}, \tag{13}$$

where the position of the single nonzero entry is arbitrary. If $k_\ell < n$, then c_ℓ has exactly two nonzero entries at arbitrary positions i and j with coefficients

$$c_{\ell,i} \triangleq \frac{\log \lambda_j}{\log \lambda_j - \log \lambda_i} \lambda_i^{n-k_\ell} \tag{14}$$

and vice versa with i and j exchanged. The choice of the positions of the nonzero elements of C does affect the shape of the weighting signal $\tilde{y}$ (as in Figure 6), but it is immaterial for the validity of the following theorem.

Theorem 2. Such a filter prefers the given pulse pattern, and the corresponding maximum value of the inner-product signal is L. □

The proof is omitted due to space constraints.

IV. A Network for Parsing Morse Code

As an example of a complete network, a four-layer network that parses Morse code was described in [1]. In the meantime, this network has been redesigned to work with the inner-product filters of this paper (which are simpler than the filters used in [1]). The network has been made to work both with filters as in Example 1 and with filters as in Example 2. The new filters of this paper also simplify the proper setting of the firing thresholds, as is obvious from Figures 5 and 7. In addition, we no longer use thresholded signals as in [1, Sec. VI], but we do everything in the pulse domain.

The operation of the network is illustrated by Figure 8. The top row in Figure 8 shows some clean Morse code represented as on-off signaling. This signal is modulated onto a 400-Hz tone, which is transmitted acoustically. The second row in Figure 8 shows the signal out of the receiver's microphone. This signal is then processed by a simple tone detector, which produces the pulse-domain signal labeled "A1" in Figure 8; a dot produces a single pulse and a dash produces three pulses. This tone detector is the whole first layer in Figure 1.

The remaining three rows in Figure 8 show examples of pulse-domain signals out of the second and third layer of the network: B5 detects a dot, B4 detects a dash at the end of a letter, and C5 detects a dot followed by a dash at the end of a letter. In total, the network comprises 41 inner-product filters, 26 of which detect the actual letters. For more details, we refer to [1].

This example demonstrates, in particular, that such a network is self-timed and does not require any external synchronization.

V. Neurons

The mode of computation proposed in this paper has some obvious similarities with biological neural networks. This raises the question whether an inner-product filter can be built with a biological plausible neuron model.

There is no trace of real neurons doing anything like Example 1. The case seems more hopeful for Example 2: indeed, the classical integrate-and-fire neuron model [4] amounts to an inner-product filter with $m = 1$ and $A = (\lambda_1)$. But this is inadequate for our purpose: the detection of temporal patterns as in Example 2 requires at least two different time constants λ_1 and λ_2.

However, modern neuron models as in [8]–[10] can indeed

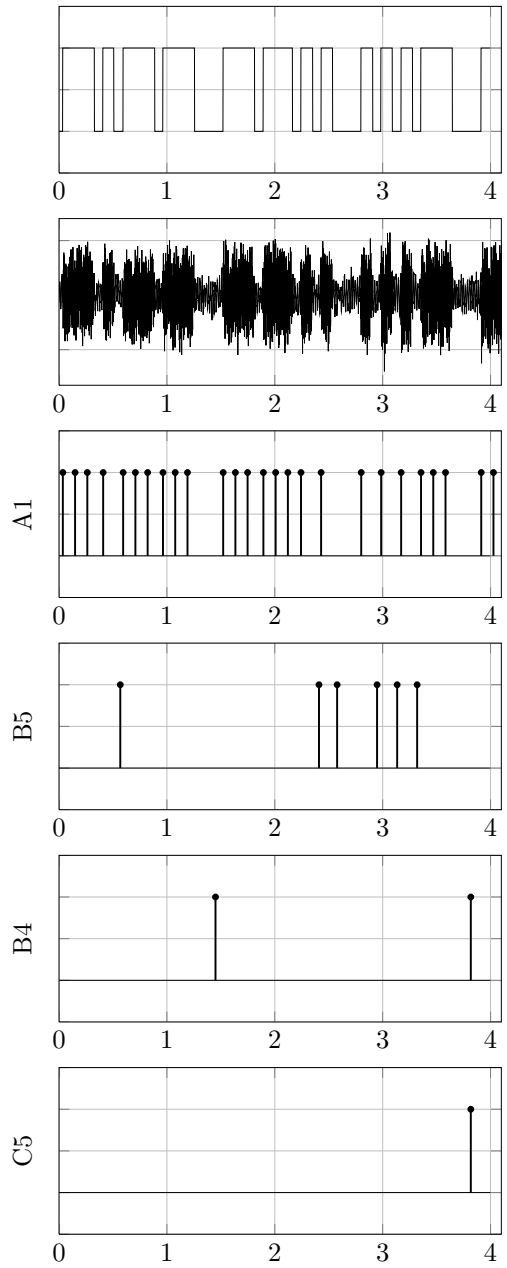

Fig. 8. Morse code network. Top: clean morse code signal. Second: raw data from microphone. Third: pulse-domain output of tone detector. Subsequent: pulse-domain feature signals B5, B4, and C5 as in [1].

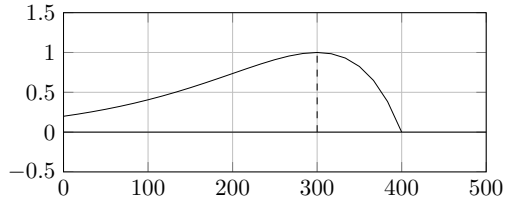

Fig. 9. Weighting signal (time-reversed impulse response) of neuron model.

be paraphrased by (4) and (5) with A of the form

$$(A^{-1})^{\mathsf{T}} = \begin{pmatrix} \lambda_0 & \alpha_1 & \dots & \alpha_M \\ 0 & \lambda_1 & & 0 \\ \vdots & & & \vdots \\ 0 & \dots & 0 & \lambda_M \end{pmatrix}, \tag{15}$$

with $s = (1, 0, \dots, 0)$, and with C of the form

$$C^{\mathsf{T}} = \begin{pmatrix} 0 & \dots & & & & & & & 0 \\ 1 & \dots & 1 & 0 & \dots & & & & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & \dots & & & & 0 & 1 & \dots & 1 \end{pmatrix}; \tag{16}$$

the first component of the state vector ξ_n is the potential in the soma (the main cell body) while the remaining M components represent the potential in separate dentritic trees.

The weighting signals (= time-reversed impulse responses from the synapses, disregarding firing) then look like in Figure 9, where the time constant (and the position of the peak) differs between different dentritic trees. Any such neuron thus prefers some well-defined monomial pattern (as in Section III), and thus makes a useful inner-product filter. For example, the redesign of the Morse code network of Section IV for such neurons is straightforward.

VI. Conclusion

We have proposed a new mode of signal processing using linear filters and well-separated unit pulses. Such networks are self-timed and require no extra synchronization, and they can be implemented with biologically plausible neurons.

References

[1] H.-A. Loeliger, S. Neff, and Ch. Reller, "Self-synchronizing signal parsing with spiking feature-detection filters," *Proc. 52nd Annual Allerton Conference on Communication, Control, and Computing,* Monticello, Illinois, USA, Oct. 1–3, 2014, pp. 123–128.

[2] Ch. Reller, M. V. R. S. Devarakonda, and H.-A. Loeliger, "Glue factors, likelihood computation, and filtering in state space models," *Proc. 50th Annual Allerton Conference on Communication, Control, and Computing,* Monticello, Illinois, USA, Oct. 1–5, 2012, pp. 686–689.

[3] L. Bruderer, H.-A. Loeliger, and N. Zalmai, "Local statistical models from deterministic state space models, likelihood filtering, and local typicality," *Proc. 2014 IEEE Int. Symp. on Information Theory (ISIT)*, Honolulu, Hawaii, June 29 – July 4, 2014, pp. 1106–1110.

[4] W. Maass and C. Bishop (eds.), *Pulsed Neural Networks.* Cambridge, MA: MIT Press, 1999.

[5] E. N. Brown, R. E. Kass, and P. P. Mitra, "Multiple neural spike train data analysis: state-of-the-art and future challenges," *Nature Neuroscience,* vol. 7, no. 8, May 2004, pp. 456–461.

[6] H. Paugam-Moisy and S. Bohte, "Computing with Spiking Neuron Networks," *Handbook of Natural Computing,* (G. Rozenberg, T. Bäck, and J. N. Kok, eds.) Chapt. 10, Springer Verlag, 2011.

[7] T. Sejnowski and T. Delbruck, "The language of the brain," *Scientific American,* vol. 307, no. 4, pp. 54–59, 2012.

[8] P. Poirazi, T. Brannon, B. W. Mel, "Pyramidal neuron as two-layer neural network," *Neuron*, vol. 37, pp. 989–999, March 2003.

[9] T. Branco, M. Häusser, "The single dendritic branch as a fundamental functional unit in the nervous system," *Current Opinion in Neurobiology*, vol. 20, pp. 494–502, August 2010.

[10] M. P. Jadi, B. F. Behabadi, A. Poleg-Polsky, J. Schiller, B. W. Mel, "An augmented two-layer model captures nonlinear analog spatial integration effects in pyramidal neuron dendrites," *Proceedings of the IEEE*, vol. 102, No. 5, May 2014.