Realizing Video Time Decoding Machines with Recurrent Neural Networks

Aurel A. Lazar and Yiyin Zhou

Abstract— **Video Time Decoding Machines faithfully reconstruct bandlimited stimuli encoded with Video Time Encoding Machines. The key step in recovery calls for the pseudo-inversion of a typically poorly conditioned large scale matrix. We investigate the realization of time decoders employing only neural components. We show that Video Time Decoding Machines can be realized with recurrent neural networks, describe their architecture and evaluate their performance. We provide the first demonstration of recovery of natural and synthetic video scenes encoded in the spike domain with decoders realized with only neural components. The performance in recovery using the latter decoder is not distinguishable from the one based on the pseudo-inversion matrix method.**

I. Introduction

Time Encoding Machines (TEMs) model the representation (encoding) of stimuli by sensory systems with neural circuits that communicate via spikes (action potentials). TEMs asynchronously encode time-varying analog stimuli into a multidimensional time sequence [1]. Given Nyquist-type rate conditions, a bandlimited signal can be recovered with arbitrary accuracy by Time Decoding Machines (TDMs). Video Time Encoding Machines (vTEMs) encode space-time-varying signals including visual stimuli (movies, animation) into a multidimensional time sequence [2]. Different models of neural encoding circuits have been investigated including circuits with random parameters [3].

Hardware implementations of TEMs are also available. For example Asynchronous Sigma-Delta Modulators (ASDMs), that have been shown to be an instance of TEMs, can be robustly implemented in low power analog VLSI [4]. With the ever decreasing voltage and increasing clock rate, amplitude domain high precision quantizers are more and more difficult to implement. In the nanoworld, it is more cost effective to measure "time" as opposed to measuring "space" (signal amplitude). The next generation silicon encoders are expected to operate in the time domain [5]. Representing information in time domain follows the miniaturization trends of nanotechnology.

Although the encoding mechanism can be efficiently implemented in neural circuits, the reconstruction algorithms call for the pseudo-inversion of a large scale matrix. Several real-time reconstruction algorithms have been demonstrated in the past [5], [6]. In this paper, we seek a solution to the reconstruction of time encoded signals using neural hardware components [7]. Clearly, a decoding circuit built using neural components has to minimize the same cost function that leads to a solution via a matrix pseudo-inversion.

Neural network methods for solving optimization problems have received considerable attention in the past 20 years [8]. These networks have attractive properties. First, the key components of the neural network mimic the properties of biological neurons. Second, the structure of the neural network can be efficiently implemented in analog VLSI.

We propose in this paper a reconstruction method for signals encoded with TEMs based on recurrent neural networks. We further extend the method to the recovery of space-time stimuli encoded with vTEMs. Simulation results show that the proposed method provides high quality reconstructions that are comparable to the ones obtained by applying the matrix pseudo-inverse method. The recurrent neural network decoding method has two main advantages: (i) it is intrinsically parallel and thereby *scalable* for real-time decoding, and (ii) it can be implemented using simple neural hardware components.

This paper is organized as follows. In Section II, a faithful reconstruction algorithm is provided for time-varying bandlimited stimuli encoded with a single neuron TEM. In Section III, two recurrent neural networks are used to reconstruct the encoded signal, and their performance is examined by simulations. In Section IV, space-time stimuli are encoded with video TEMs and their faithful reconstruction is obtained using a class of video TDMs. In Section V, a recurrent neural network realization is given for the recovery of stimuli encoded by video TDMs. Section VI briefly concludes the paper.

II. Time Encoding Machines

In order to better understand the encoding properties of the more complex vTEM architecture, we first consider a single-input single-output TEM that encodes time-varying signals and provide the stimulus reconstruction algorithm.

In what follows we shall assume that the signal $u = u(t), t \in \mathbb{R}$, is mapped into a time sequence $(t_k), k \in \mathbb{Z}$. Given the complexity of the encoding and decoding process, it is advantageous to assume that the signal u lives in a finite dimensional space.

A. Modeling the Stimuli

The time-varying stimuli (signals) assumed here are elements of a Hilbert space. The Hilbert space $\mathcal{H}_t$ we consider is the space of trigonometric polynomials, where every element

Aurel A. Lazar and Yiyin Zhou are with the Department of Electrical Engineering, Columbia University, New York, NY 10027, USA (email: {aurel, yiyin}@ee.columbia.edu). Proceedings of the 2011 International Joint Conference on Neural Networks, San Jose, CA, July 31 - August 5, 2011, to appear.

$u = u(t), t \in \mathbb{R}$, is of the form

$$u(t) = \sum_{m_t=-M_t}^{M_t} c_{m_t} e_{m_t}, \tag{1}$$

with

$$e_{m_t} = \exp\left(jm_t \frac{\Omega_t}{M_t} t\right)$$

an element of the basis spanning the space $\mathcal{H}_t$; Ω_t and M_t are the bandwidth and the order of the space of trigonometric polynomials, respectively. Note that every element in this Hilbert space is periodic with period

$$S_t = \frac{2\pi M_t}{\Omega_t}.$$

Assuming that all signals are real, $c_{-m_t} = \overline{c_{m_t}}$, where $\overline{(\cdot)}$ denotes the complex conjugate.

The inner product in $\mathcal{H}_t$ is defined in the usual way: $\forall u, v \in \mathcal{H}_t$,

$$\langle u, v \rangle = \frac{1}{S_t} \int_0^{S_t} u(t)\overline{v(t)} dt. \tag{2}$$

Note that, the space of trigonometric polynomials is a finite dimensional Hilbert space, and therefore, a Reproducing Kernel Hilbert Space (RKHS), with reproducing kernel

$$K(t, s) = \sum_{m_t=-M_t}^{M_t} e_{m_t}(t - s), \tag{3}$$

with $t, s \in \mathbb{R}$.

Modeling the set of stimuli in a Hilbert space enables us to use the geometry of the space to reduce stimulus encoding to projections on a set of functions.

B. Encoding

The encoding of time-varying signal consists of two cascaded modules, as shown in Fig. 1. The signal is passed through a temporal receptive field $D_T(t)$ before being fed into a neural circuit. The processing of the temporal receptive field is modeled as filtering. More formally, we define an operator ${}^rL : \mathcal{H}_t \to \mathcal{H}_t$ such that

$$v(t) = {}^rLu = \int_{\mathbb{R}} D_T(t - s)u(s)ds = (D_T * u)(t).$$

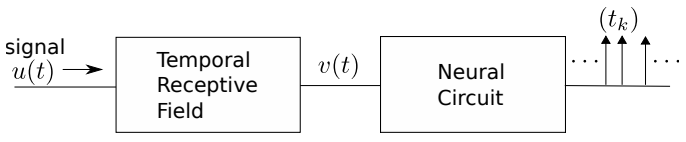

Fig. 1. Encoding diagram of a TEM with a single neuron.

The neural circuit encodes the output of the receptive field. The neural circuit can be realized with different neural building blocks such as Integrate-And-Fire neurons (IAF) and Hodgkin-Huxley neurons as well as Asynchronous Sigma-Delta Modulators (ASDM). In what follows we denote the spike times of the spike train at the output of the neural circuit as $(t_k), k = 0, 1, 2, \cdots, n$.

The operation of the neural circuit can be described by a bounded linear functional ${}^{\mathbb{T}}L_k : \mathcal{H}_t \to \mathbb{R}$. The explicit formula of this functional is determined by the t-transform [1] of the neuron and it is given by

$${}^{\mathbb{T}}L_k u = q_k, \text{for } u \in \mathcal{H}_t.$$

where ${}^{\mathbb{T}}L_k$ and q_k usually depend on $(t_k), k = 0, 1, 2, \cdots, n$. For example, for an ideal IAF neuron with the t-transform given by

$$\int_{t_k}^{t_{k+1}} u(s)ds = \kappa\delta - b(t_{k+1} - t_k),$$

we have

$${}^{\mathbb{T}}L_k u = \int_{t_k}^{t_{k+1}} u(s)ds,$$

$$q_k = \kappa\delta - b(t_{k+1} - t_k),$$

where κ, δ and b are, respectively, the integration constant, the threshold and the bias of the IAF neuron.

Combining the two cascaded building blocks together, we define the bounded linear functionals $L_k : \mathcal{H}_t \to \mathbb{R}$ as

$$L_k = {}^{\mathbb{T}}L_k {}^rL$$

and therefore

$$L_k u = {}^{\mathbb{T}}L_k {}^rLu = q_k.$$

By the Riesz representation theorem, the above functionals can be expressed in inner product form as

$$L_k u = \langle u, \phi_k \rangle, \text{for all } u \in \mathcal{H}_t,$$

where, by the reproducing property,

$$\phi_k(t) = \langle \phi_k, K_t \rangle = L_k \overline{K_t},$$

with $K_t(s) = K(t, s)$.

Formulation of time encoding of stimuli in inner product form provides a simple but very powerful insight into the encoding process itself. Since the inner products are merely projections of the time-varying stimulus onto the axes defined by the ϕ_k's, encoding is interpreted as generalized sampling, and the q_k's are the measurements given by sampling the signal. Note however that unlike in traditional sampling, the sampling functionals in time encoding are signal dependent.

C. Reconstruction

Reconstruction of a time-varying signal u is formulated here as the variational problem

$$\hat{u} = \underset{u \in \mathcal{H}_t}{\operatorname{argmin}} \{ \sum_{k=1}^{n} \left(\langle u, \phi_k \rangle - q_k \right)^2 + n\lambda \|u\|^2_{\mathcal{H}_t} \}. \tag{4}$$

By the Representer Theorem, the solution to problem (4) is of the form

$$\hat{u} = \sum_{k=1}^{n} c_k \phi_k. \tag{5}$$

Substituting the solution into equation (4), the coefficients c_k can be obtained by solving the unconstrained optimization problem

$$\text{minimize } \|\mathbf{G}\mathbf{c}-\mathbf{q}\|_{l_2}^2+n\lambda\mathbf{c}^T\mathbf{G}\mathbf{c}, \tag{6}$$

where $\mathbf{c}=[c_1,c_2,\cdots,c_n]^T$, $\mathbf{q}=[q_1,q_2,\cdots,q_n]^T$ and $\mathbf{G}$ is a symmetric matrix with entries

$$[\mathbf{G}]_{k,l}=\langle\phi_k,\phi_l\rangle=\sum_{m_t=-M_t}^{M_t}\left(\int_{t_k}^{t_{k+1}}(D_T*e_{m_t})(s)ds\cdot\int_{t_l}^{t_{l+1}}(D_T*e_{-m_t})(s)ds\right).$$

The minimization problem in (6) has an explicit analytical solution with $\mathbf{c}$ the solution of the system of linear equations

$$\mathbf{G}^T(\mathbf{G}+n\lambda\mathbf{I})\mathbf{c}=\mathbf{G}^T\mathbf{q}, \tag{7}$$

where $\mathbf{I}$ is the $n\times n$ identity matrix. Therefore, the stimulus reconstruction depends on solving a system of linear equations. Given that the matrix $\mathbf{G}$ is usually singular, this is often achieved by employing the (computationally demanding) Moore-Penrose pseudo-inverse.

III. Using Recurrent Neural Networks for TDMs

Based on [8] and [9], we present here two recurrent neural networks, respectively, for reconstructing the encoded signal from spike times.

A. Recurrent Neural Network I

Using a general gradient approach for solving problem (6), we consider the system of differential equations

$$\frac{d\mathbf{c}}{dt}=-\boldsymbol{\mu}\nabla E(\mathbf{c}), \tag{8}$$

with initial condition $\mathbf{c}(0)=\mathbf{0}$, where

$$\mathbf{E}(\mathbf{c})=\frac{1}{2}\left(\|\mathbf{G}\mathbf{c}-\mathbf{q}\|_{l_2}^2+n\lambda\mathbf{c}^T\mathbf{G}\mathbf{c}\right), \tag{9}$$

and $\boldsymbol{\mu}(\mathbf{c},t)$ is an $n\times n$ symmetric positive definite matrix that determines the speed of convergence and whose entries are usually dependent on the variables $\mathbf{c}(t)$ and time t. It follows that

$$\nabla E(\mathbf{c})=\mathbf{G}^T\left((\mathbf{G}+n\lambda\mathbf{I})\mathbf{c}-\mathbf{q}\right). \tag{10}$$

Since $E(\mathbf{c})$ is clearly convex in $\mathbf{c}$, the system of differential equations (8) asymptotically approaches the unique solution of the regularized optimization problem (6).

Consequently, we have

$$\frac{d\mathbf{c}}{dt}=-\boldsymbol{\mu}\mathbf{G}^T\left((\mathbf{G}+n\lambda\mathbf{I})\mathbf{c}-\mathbf{q}\right).$$

The above set of differential equations can be mapped into a recurrent neural network as shown in Figure 2. This is a three layer neural network. In the first layer, consisting of n multiply/add units as shown in the left most column, the vector $(\mathbf{G}+n\lambda\mathbf{I})\,\mathbf{c}-\mathbf{q}$ is computed. The multiplication factors are the entries of the matrix $\mathbf{G}+n\lambda\mathbf{I}$ and the vector $\mathbf{q}$. In the second layer, $\nabla E(\mathbf{c})$ is evaluated. This layer also consists of n multiply/add units, with multiplication factors provided by the entries of the matrix $\mathbf{G}$. Note that $\mathbf{G}$ is a symmetric matrix. The gradient is weighted by the learning rate $\boldsymbol{\mu}$ in the third layer, that also consists of n multiply/add units. The outputs of the third layer provide the time derivative of the vector $\mathbf{c}(t)$. The time derivatives are then integrated and the outputs are fed back to the first layer.

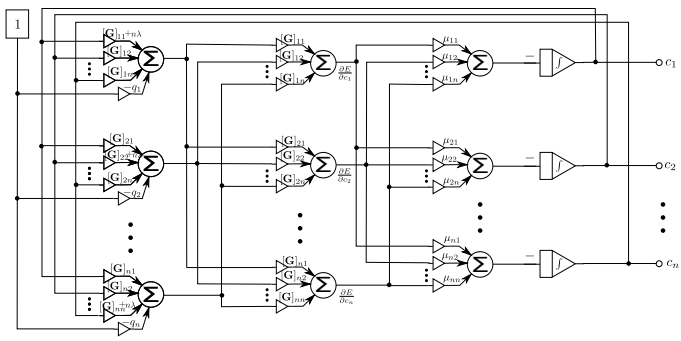

Fig. 2. Block diagram of the recurrent neural network I.

B. Recurrent Neural Network II

Alternatively, we can formulate the reconstruction of the encoded stimulus as the spline interpolation problem

$$\hat{u}=\underset{u\in\mathcal{H}_t,\{L_ku=q_k\}_{k=1}^n}{\operatorname{argmin}}\{\|u\|_{\mathcal{H}_t}^2\}, \tag{11}$$

that seeks to minimize the norm as well as satisfy all the t-transform equations. In addition, the solution is of the form (5). Substituting the solution into equation (11), the vector of coefficients $\mathbf{c}$ are the solution of the optimization problem

$$\begin{array}{ll}\text{minimize} & \frac{1}{2}\mathbf{c}^T\mathbf{G}\mathbf{c}\\ \text{subject to} & \mathbf{G}\mathbf{c}=\mathbf{q},\end{array} \tag{12}$$

where $\mathbf{G},\mathbf{c},\mathbf{q}$ are as defined in equation (6). We notice that due to the RKHS property, $\mathbf{G}$ is a positive semidefinite matrix. Therefore, the above optimization problem is a convex quadratic programming problem with equality constraints.

Problem (12) can be reformulated as a standard quadratic programming problem. By setting $\mathbf{x}=\left[\begin{array}{cc}\mathbf{x}_+{}^T & \mathbf{x}_-{}^T\end{array}\right]^T$ with $\mathbf{x}_+\geq 0,\mathbf{x}_-\geq 0$ such that $\mathbf{c}=\mathbf{x}_+-\mathbf{x}_-$, we obtain the following convex programming problem

$$\begin{array}{ll}\text{minimize} & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x}\\ \text{subject to} & \mathbf{A}\mathbf{x}=\mathbf{q},\mathbf{x}\geq 0,\end{array} \tag{13}$$

where

$$\mathbf{Q}=\left[\begin{array}{cc}\mathbf{G} & -\mathbf{G}\\ -\mathbf{G} & \mathbf{G}\end{array}\right],$$

and

$$\mathbf{A}=\left[\begin{array}{cc}\mathbf{G} & -\mathbf{G}\end{array}\right].$$

The neural network that solves problem (13) is then given by [9]

$$\frac{d}{dt}\left(\begin{array}{c}\mathbf{x}\\ \mathbf{y}\end{array}\right)=\beta\left(\begin{array}{c}\left(\mathbf{x}-\alpha\mathbf{Q}\mathbf{x}+\alpha\mathbf{A}^T\mathbf{y}\right)^+-\mathbf{x}\\ \alpha\left(-\mathbf{A}\mathbf{x}+\mathbf{q}\right)\end{array}\right), \tag{14}$$

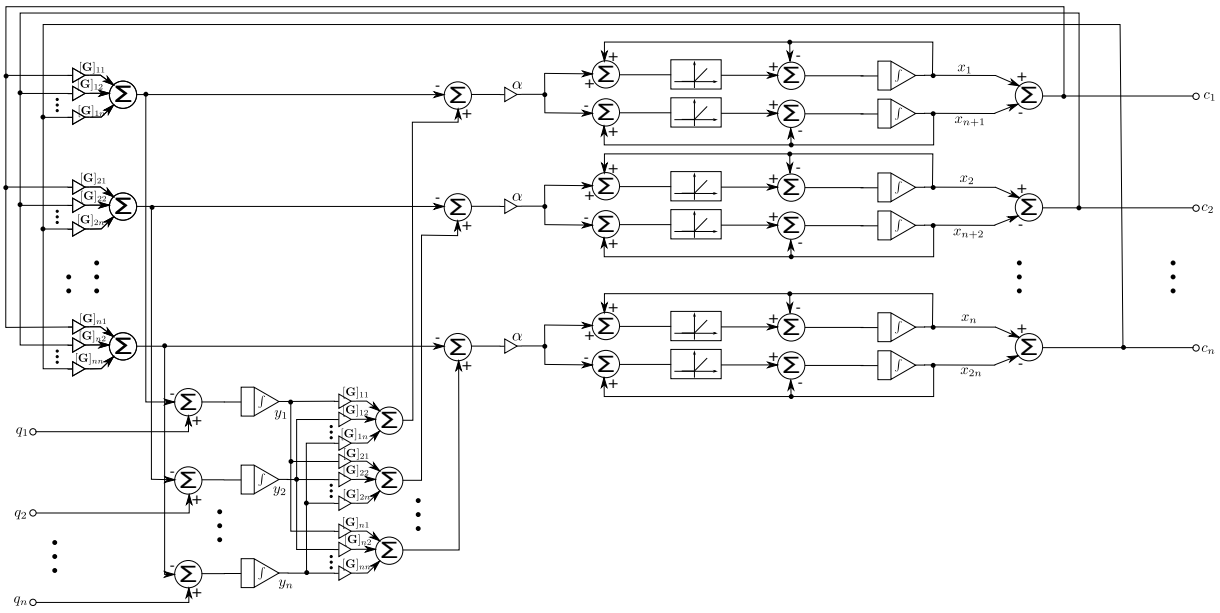

Fig. 3. Block diagram of the recurrent neural network II with $\beta = 1$.

where $(\mathbf{x})^+ = [(x_1)^+, \cdots, (x_n)^+]^T$ and $(x_i)^+ = \max\{0, x_i\}$, α is a positive constant and $\beta > 0$ is the scaling constant. The recurrent neural network implementing (14) can be realized by a simplified diagram as shown in Fig. 3, where $\beta = 1$. Note that we simplified the circuit diagram by observing that $\mathbf{Qx} = \begin{bmatrix} (\mathbf{Gc})^T & -(\mathbf{Gc})^T \end{bmatrix}^T$, $\mathbf{A}^T\mathbf{y} = \begin{bmatrix} (\mathbf{Gy})^T & -(\mathbf{Gy})^T \end{bmatrix}^T$ and $\mathbf{Ax} = \mathbf{Gc}$.

The two recurrent neural networks described above are realized with adders, integrators, multipliers and piecewise linear activation functions and, equally importantly, are highly parallel. For solving large scale problems in real-time these can be implemented in analog VLSI.

C. Results

In what follows, we show some illustrative examples of the neural networks discussed above being used in the reconstruction of signals encoded by TEMs.

We consider a bandlimited signal u with bandwidth $\Omega_t = 2\pi \cdot 10\ rad/s$ of the form

$$u(t) = \sum_{n=1}^{20} u(kT)\text{sinc}\left(\Omega_t(t - kT)\right),$$

where $T = \frac{\pi}{\Omega_t}$ and the signal is defined in the time interval $[0, 1]$. Note that the signal can be well approximated by (1) using appropriate parameters. Here, we consider $M_t = 20$. The temporal receptive field we employed is a low-pass filter with impulse response given by

$$D_T(t) = 100e^{-0.01t}\left(\frac{(0.01t)^6}{6!} + \frac{(0.01t)^7}{7!}\right).$$

An ideal IAF neuron is used with parameters $\kappa = 1, b = 1.5, \delta = 0.05$. 35 spikes were generated in the simulation.

Recurrent neural network I was first used in reconstruction, $\boldsymbol{\mu}$ set to a diagonal matrix with each of the diagonal entries 10^7 and $\lambda = 10^{-11}$. After running the neural network for 5 seconds, the reconstruction is shown in Fig. 4(a) with a resulting signal-to-noise-ratio (SNR) of 48.0 [dB]. As a comparison, by solving equation (7), the SNR of the reconstruction is 65.91 [dB]. We note from the trajectories of $\mathbf{c}(t)$ shown in Fig. 4(b) that the convergence is very fast in the beginning, but as the gradient decreases, the convergence rate slows down substantially. Reconstruction quality reaches 52.0 [dB] after the network ran for about 40 seconds. The objective function as a function of time is shown in Fig. 4(c). The SNR of the stimulus reconstruction is shown in Fig. 4(d).

When recurrent neural network II was used in the reconstruction, $\beta = 10^4$ and $\alpha = 200$. The SNR of the reconstruction is 65.65 [dB], a performance that is very close to the SNR achieved by the pseudo-matrix inversion method. The trajectory of the output $\mathbf{c}(t) = \mathbf{x}^+ - \mathbf{x}^-$ is shown in Fig. 5(a). We notice that after 5 seconds the system has not yet settled down to an equilibrium point. The objective function $\mathbf{c}^T\mathbf{Gc}$ depicted in Fig. 5(b) has reached its stability point after about 0.9 seconds. This may be due to the fact that the equality constraint is hard or even impossible to satisfy. Even if it is satisfied, $\mathbf{c}$ may be very large because $\mathbf{G}$ is ill-conditioned. As shown in Fig. 5(d), the l_2 norm of the error $\mathbf{Gc} - \mathbf{q}$ is still decreasing at an extremely slow rate after $t = 1$ but may take a very long time to vanish. Nevertheless, as shown in Fig. 5(c), the SNR of the reconstructed signal stays very high even when using $\mathbf{c}$ at $t = 0.7$s. Consequently, a stopping rule needs to be introduced for determining whether the objective function has remained essentially unchanged.

As seen in the above examples, both neural networks provide a high quality reconstruction for time-varying signals.

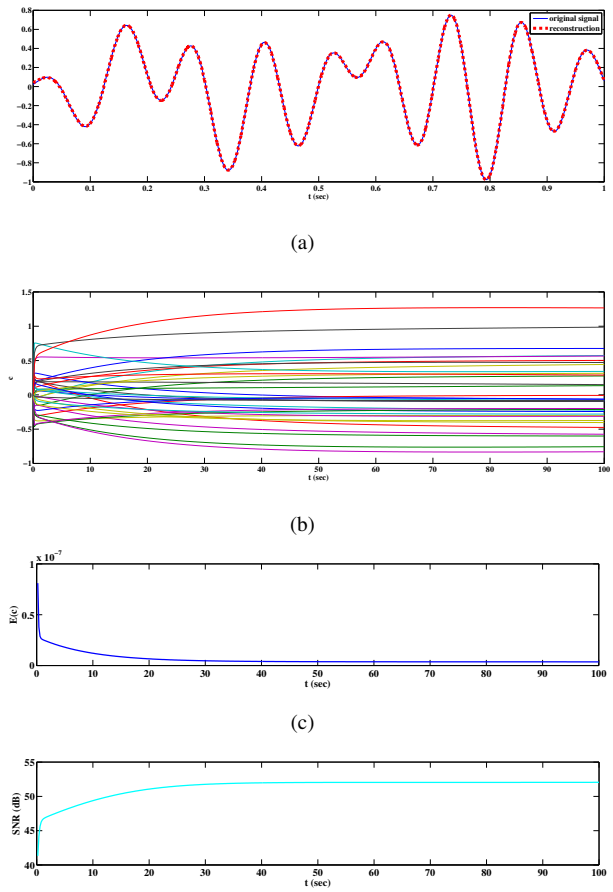

Fig. 4. Characterization of the performance of stimulus recovery using the recurrent neural network I. (a) Original (blue) and the reconstructed stimulus (red). (b) Trajectories of the vector $\mathbf{c}$. (c) Value of the objective function $\mathbf{c}^T\mathbf{G}\mathbf{c}$ as a function of time. (d) SNR of the reconstructed signal.

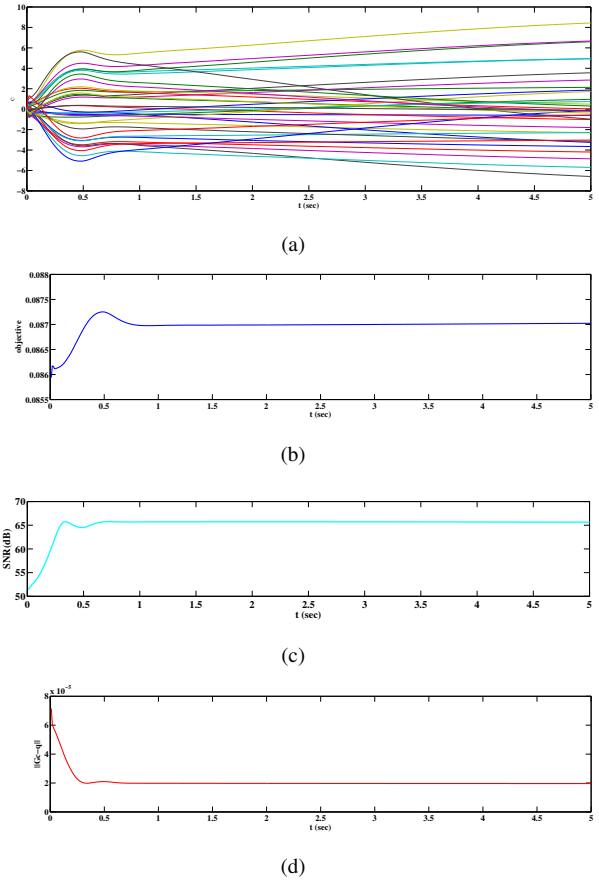

Fig. 5. Characterization of stimulus recovery using the recurrent neural network II. (a) Trajectories of the vector $\mathbf{c}$. (b) Value of the objective function $\mathbf{c}^T\mathbf{G}\mathbf{c}$ as a function of time. (c) SNR of the reconstructed signal. (d) Norm of the error $\mathbf{G}\mathbf{c}-\mathbf{q}$ of the equality constraint.

IV. Video Time Encoding Machines

As we already mentioned, video TEMs represent space-time stimuli in the spike domain. The video TEM architecture consists of receptive fields in cascade with a population of spiking neurons. Along with the parameters describing the encoding circuit, the multidimensional output spike train is used to reconstruct the original spatio-temporal stimulus. Following an approach similar to the one in the previous sections, we first describe the process of encoding and then provide a reconstruction algorithm based on spike times.

A. Modeling of video signal

The space of trigonometric polynomials we considered in the previous sections can easily be extended to the tri-variable trigonometric space of polynomials $\mathcal{H}$. In $\mathcal{H}$ every element is expressed as

$$I(x,y,t)=\sum_{m_x=-M_x}^{M_x}\sum_{m_y=-M_y}^{M_y}\sum_{m_t=-M_t}^{M_t}c_{m_x,m_y,m_t}e_{m_x,m_y,m_t} \tag{15}$$

where

$$e_{m_x,m_y,m_t}(x,y,t)=\exp\left(jm_x\frac{\Omega_x}{M_x}x+jm_y\frac{\Omega_y}{M_y}y+jm_t\frac{\Omega_t}{M_t}t\right)$$

is an element of the basis of the space $\mathcal{H}$, $(\Omega_x,\Omega_y,\Omega_t)$ and (M_x,M_y,M_t) are, respectively, the bandwidth and the order of the trigonometric polynomial in each variable. An element $I\in\mathcal{H}$ is, respectively, periodic in each variable with period

$$S_x=\frac{2\pi M_x}{\Omega_x},S_y=\frac{2\pi M_y}{\Omega_y},S_t=\frac{2\pi M_t}{\Omega_t}.$$

The inner product is defined $\forall I_1,I_2\in\mathcal{H}$ as,

$$\langle I_1,I_2\rangle=\frac{1}{S_xS_yS_t}\int_0^{S_x}\int_0^{S_y}\int_0^{S_t}I_1(x,y,t)\overline{I_2(x,y,t)}dxdydt. \tag{16}$$

$\mathcal{H}$ is also a RKHS with reproducing kernel

$$K(x,y,t;x',y',t')=\sum_{m_x=-M_x}^{M_x}\sum_{m_y=-M_y}^{M_y}\sum_{m_t=-M_t}^{M_t}e_{m_x,m_y,m_t}(x-x',y-y',t-t').$$

The effectiveness of using the above Hilbert space as a model of visual stimuli is justified in [3].

B. Encoding

As shown in Fig. 6, the architecture of the vTEM consists of two cascaded building blocks, similar in structure to the TEMs described in the previous sections.

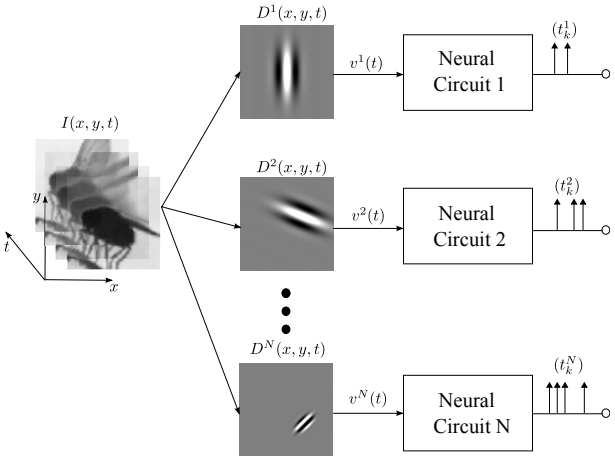

Fig. 6. Architecture of the Video Time Encoding Machine.

First, the video signal is passed through a set of visual receptive fields. The operation of the jth visual receptive field $D^j(x,y,t)$ is given by the operator ${}^{\mathbb{S}}L^j : \mathcal{H} \to \mathcal{H}_t$ by

$$ {}^{\mathbb{S}}L^j I = \int_{\mathbb{R}} \left(\int_{\mathbb{R}^2} D^j(x,y,s) I(x,y,t-s) dxdy \right) ds = v^j(t). \tag{17}$$

Here, $\mathcal{H}_t$ denotes the univariable trigonometric polynomial space with bandwidth Ω_t and order M_t. Note that the above operator maps a 3-D space into a 1-D space.

A simplified case when the visual receptive field is spatio-temporal separable is usually considered. In this case, the j'th receptive field can be separated into the spatial receptive field $D_S^j(x,y)$ and the temporal receptive field $D_T^j(t)$ such that

$$D^j(x,y,t) = D_S^j(x,y) D_T^j(t).$$

The type of spatial receptive fields usually considered are Gabor frames and Difference of Gaussian frames. They resemble the spatial receptive fields of simple cells in the Primary Visual Cortex (V1) and Retinal Ganglion Cells (RGCs) in the retina, respectively.

Each spatial receptive field is derived from a mother wavelet. Given the mother wavelet $\gamma(x,y)$, the set of all receptive fields can be obtained by performing the following three operations or their combinations:

- Dilation $D_\alpha, \alpha \in \mathbb{R}\backslash\{0\}$: $D_\alpha \gamma(x,y) = |\alpha|^{-1} \gamma(\frac{x}{\alpha}, \frac{y}{\alpha})$,
- Rotation $R_\theta, \theta \in [0, 2\pi)$: $R_\theta \gamma(x,y) = \gamma(x\cos\theta + y\sin\theta, -x\sin\theta + y\cos\theta)$,
- Translation $\tau_{x_0,y_0}, (x_0,y_0) \in \mathbb{R}^2$: $\tau_{x_0,y_0}\gamma(x,y) = \gamma(x - x_0, y - y_0)$.

In order to reconstruct the video signal, we require the set of the receptive field to form a frame and cover the entire spatial field.

Then, each output of the visual receptive fields is fed into a neural circuit. Let us denote the output of the jth neural circuit as $(t_k^j), k = 1, 2, \cdots, n_j$. The operation of the neural circuit can be described by a bounded linear functional ${}^{\mathbb{T}}L_k^j : \mathcal{H}_t \to \mathbb{R}$, where

$$ {}^{\mathbb{T}}L_k^j v^j = q_k^j, \text{for } v^j \in \mathcal{H}_t.$$

Combining the two cascaded modules together and assuming that there is a total N visual receptive fields, the jth of which is connected with the jth neural circuit that generates a single spike train $(t_k^j), k = 1, 2, \cdots, n_j, j = 1, 2, \cdots, N$, we define bounded linear functionals $L_k^j : \mathcal{H} \to \mathbb{R}$ as

$$L_k^j = {}^{\mathbb{T}}L_k^{j\,\mathbb{S}}L^j$$

and therefore

$$L_k^j I = {}^{\mathbb{T}}L_k^{j\,\mathbb{S}}L^j I = \langle I, \phi_k^j \rangle = q_k^j,$$

where

$$\phi_k^j(x,y,t) = \langle \phi_k^j, K_{x,y,t} \rangle = L_k^j \overline{K_{x,y,t}},$$

with $K_{x,y,t}(x',y',t') = K(x,y,t;x',y',t')$. Thus the operation of the vTEMs can again be reduced to generalized sampling for functions in $\mathcal{H}$.

C. Video Time Decoding

As before, the output spike trains of vTEMs are used for the reconstruction of the video stimulus. Adapting the previous notation, reconstruction is again formulated as a variational problem of the form

$$\hat{I} = \underset{I \in \mathcal{H}}{\operatorname{argmin}} \{ \sum_{j=1}^{N} \sum_{k=1}^{n_j} \left(\langle I, \phi_k^j \rangle - q_k^j \right)^2 + n\lambda \|I\|_{\mathcal{H}}^2 \}, \tag{18}$$

where λ is the smoothing parameter and $n = \sum_{j=1}^{N} n_j$ is the total number of spikes. Its solution is of the form

$$\hat{I} = \sum_{j=1}^{N} \sum_{k=1}^{n_j} c_k^j \phi_k^j, \tag{19}$$

where $\mathbf{c} = \left[c_1^1, c_2^1, \cdots, c_{n_1}^1, c_1^2, c_2^2, \cdots, c_{n_2}^2, \cdots, c_{n_N}^N \right]^T$ satisfies the system of linear equations

$$\mathbf{G}^T(\mathbf{G} + n\lambda \mathbf{I})\mathbf{c} = \mathbf{G}^T \mathbf{q} \tag{20}$$

with $\mathbf{q} = \left[q_1^1, q_2^1, \cdots, q_{n_1}^1, q_1^2, q_2^2, \cdots, q_{n_2}^2, \cdots, q_{n_N}^N \right]^T$, $\mathbf{I}$ the $n \times n$ identity matrix and $\mathbf{G}$ the block matrix

$$\mathbf{G} = \begin{bmatrix} \mathbf{G^{11}} & \mathbf{G^{12}} & \cdots & \mathbf{G^{1N}} \\ \mathbf{G^{21}} & \mathbf{G^{22}} & \cdots & \mathbf{G^{2N}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{G^{N1}} & \mathbf{G^{N2}} & \cdots & \mathbf{G^{NN}} \end{bmatrix},$$

with entries of each block given by

$$\left[\mathbf{G}^{ij} \right]_{kl} = \langle \phi_k^i, \phi_l^j \rangle.$$

Again, the video reconstruction problem reduces to solving a system of linear equations [3].

Similar to the single neuron case, we can formulate the reconstruction as the spline interpolation problem

$$\hat{I} = \underset{I\in\mathcal{H},\{L_k^j I=q_k^j\}_{(k,j)=(1,1)}^{(n,N)}}{\operatorname{argmin}} \{\|I\|_{\mathcal{H}}^2\}. \tag{21}$$

The solution is given by equation (19) and the vector $\mathbf{c}$ is the solution to the optimization problem

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\mathbf{c}^T\mathbf{G}\mathbf{c} \\ \text{subject to} & \mathbf{G}\mathbf{c} = \mathbf{q} \end{array} \tag{22}$$

with $\mathbf{G}, \mathbf{c}, \mathbf{q}$ defined as in equation (20).

V. Using Recurrent Neural Networks for Video TDMs

A. Recurrent Neural Networks

The two recurrent neural networks proposed in section III can be used, with appropriate modifications, for realizing video TDMs as well.

For realizing recurrent neural network I differential equation (8) with $\mathbf{c}, \mathbf{G}$ and $\mathbf{q}$ defined as in equation (20) can be used. For realizing recurrent neural network II, the differential equation (14) can again be used with $\mathbf{c}, \mathbf{G}$ and $\mathbf{q}$ defined by equation (20).

As we have already seen in the results of encoding of stimuli with a single neuron TEM the decoding may perform well for small problems that are well-conditioned. If the problem is ill-conditioned, however, the recurrent neural network is not able to find the exact solution in a short period of time. As the encoding circuit grows larger, the situation could get worse, since the number of spikes for reconstructing a short video is usually larger than $\sim 10,000$. A settling time τ can be chosen whenever the cost function $E(\mathbf{c})$ is within a neighborhood of radius ε of the minimum. In this case $\mathbf{c}(t)$ can be viewed as an approximate solution to the unconstrained optimization problem. In addition, an adaptive learning rate [8] can be used to obtain faster convergence.

B. Results

In this section we show an illustrative example of encoding and decoding a natural video sequence. Shot by a high speed camera, the video describes the flight initiation of a *fruit fly* [3]. It consists of 100 frames with a screen size of 96×96 pixels. For simplicity, the video is considered to have 100 frames per second with a spatial resolution of $1/16$ (degree/pixel). $I(x, y, t)$ is defined on the domain $[-3, 3] \times [-3, 3] \times [0, 1]$. To encode the video, the vTEM consisted of 3,408 spatial visual receptive fields, i.e.,

$$D^j(x, y, t) = D^j(x, y)\delta(t), j = 1, 2, \cdots, 3408,$$

where $\delta(t)$ is the Dirac delta function. All the spatial receptive fields are derived from a Gabor mother wavelet. The output of each receptive field is encoded with an ideal IAF neuron. The temporal resolution of each spike time is 10^{-6} s. A total of 47,242 spikes were fired by the $3,408$ neurons in the 1 second duration of the video. To reconstruct the video, we chose the space of trigonometric polynomials with parameters $\Omega_x = \Omega_y = 2\pi \cdot 4\ rad/degree, \Omega_t = 2\pi \cdot 4\ rad/s, M_x = M_y = 40, M_t = 8$.

The simulations were performed on a computational platform of Tesla S2050 GPU's. The use of GPU's is very natural for our problem setting since the decoding circuit is highly parallelized. Due to memory constraints, 7 overlapped segments of the video were reconstructed each time: $[0, 200]$ ms, $[150, 350]$ ms, $[300, 500]$ ms, $[450, 650]$ms, $[600, 800]$ ms, $[750, 950]$ ms and $[900, 1000]$ ms. Two consecutive segments have an overlap of 50 ms. For the reconstruction of each time segment, the spikes fired during the time interval of the segment were used. In order to reduce error at the boundary of each segment, for each neuron, the last spike fired before the time segment starts and the first spike fired after the segment ends were used in the stimulus reconstruction on each segment. This process of segmented recovery results in a total of about 12,000 spikes in each segment. After 7 segments were reconstructed separately, the whole video was stitched together. The stitching of 50 ms overlaps between two consecutive segments follows the simple rule: $I_1(x, y, t)(1 - \theta(t)) + I_2(x, y, t)\theta(t)$, where $\theta(t) = \sin^2\left(\frac{\pi}{2} \cdot \frac{t}{0.05}\right)$ and I_1, I_2 are the 50 ms overlaps of the first and the second segment, respectively [5].

We first show the decoding results achieved with a video TDM realized using the recurrent neural network I. In the simulations, $\boldsymbol{\mu}$ was set to a diagonal matrix with equal diagonal entries. The initial value of the diagonal entries was set to 5 and exponentially increased with a rate 2.5^{-10} until it reached about 300 and the values kept constant thereafter. (Further increase of the rate may result in numerical instability.) After running the neural network for 200 ms, the overall SNR of the video was 25.4 [dB] and the SSIM index was 0.949.

In the first row of Fig. 7, we show the original video signal at multiple time instances. From left to right, the motion video at $0.1s, 0.3s, 0.5s, 0.7s$ and $0.9s$ is depicted. In the second row of Fig. 7, the video reconstruction using recurrent neural network I is shown. The reconstruction showed high perceptual quality, although the time for the neural network to reach the results shown is quite large. A better convergence may be achieved by adapting the learning rate separately for each gradient component.

Recall that in the single neuron case the stimulus recovery using the recurrent neural network II showed a better performance than when using the recurrent neural network I. Furthermore decoding with RNN II showed a decoding performance comparable with the recovery method based on the pseudo-matrix inversion. The reconstruction of video stimuli encoded in the spike domain with a TDM realized with the recurrent neural network II after $20\ ms$ is shown in Fig. 7, bottom row. The SNR is 27.8 dB and the SSIM is 0.967. Note that the decoding performance is similar to that obtained using the pseudo-inverse method of reconstruction.

With appropriate parameters, the recurrent neural network II is able to provide a high quality solution to the decoding problem in very short amount of time. The result above was

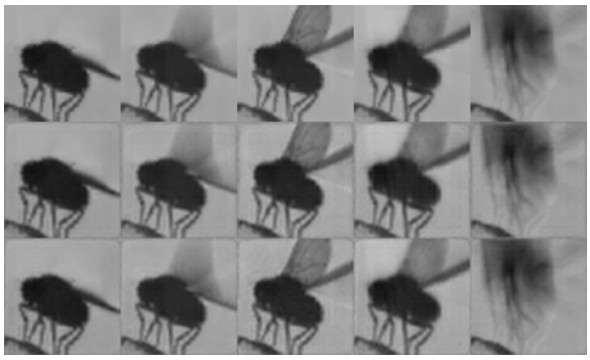

Fig. 7. Original Video (top row), reconstruction using the recurrent neural network I (middle row) and the recurrent neural network II (bottom row) at $t = 0.1, 0.3, 0.5, 0.7$ and 0.9 seconds (from left to right).

obtained after simulating the neural network for only 20 msec. However, even for a shorter amount of time, the RNN II-based recovery method is able to provide visually acceptable reconstructions. In Fig. 8 we show the reconstruction of the same video frame after $0.1, 0.5, 1, 5, 10, 15, 20$ msec. Perceptually, they all show high reconstruction quality. The SNR and SSIM of the reconstruction are already over 25 [dB] and 0.94, respectively, at $t = 1$ ms.

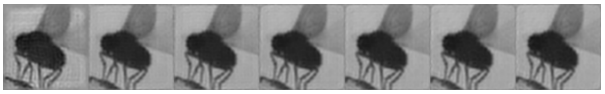

Fig. 8. The same video frame reconstructed using the recurrent neural network II and shown at iteration time t = 0.1, 0.5, 1, 5, 10, 15 and 20 msec (from left to right).

It is also interesting to note that even in simulations, the amount of time it takes to obtain a high quality reconstruction is about the same as when employing the pseudo-inverse matrix method. A high performance single precision pseudo-inverse of a square matrix of row order $\sim 12,000$ on a Tesla C2050 GPU takes about 300 seconds, and the SNR of the resulting reconstruction is about 27.7 [dB]. With 300 seconds, the neural network for the same problem can be simulated for about 10 milliseconds. As we have shown, the perceptual quality as well as other metrics, such as the SSIM index and the SNR, are comparable to using the pseudo-inverse matrix method of stimulus recovery. Thus we were able to provide an alternative to the pseudo-inverse method of realization of the video TDM. Moreover, the simulation of the proposed neural network only requires matrix-vector multiplications that can be readily parallelized for solving substantially larger problems.

VI. Discussion

The overall method proposed here shall be extended to more complex situations. As long as the reconstruction is formulated as an optimization problem, a large variety of recurrent neural networks can be utilized. These problems include nonlinear optimization with equality or inequality constraints [9], [10]. Additional constraints can be imposed on the reconstruction problem. For example, a sparse solution can be obtained by minimizing the l_1 norm, and the problem can be formulated as a linear program. Recently, a time domain linear programming circuit has been proposed [11]. The connections between each layer of the neural network in the proposed circuit are represented in time domain. Such a circuit can operate at a very high pulse rate and thus supports a real-time solution of the optimization problem.

As we have seen, the size of the recurrent neural network equals the number of spikes to be decoded, rather than the number of neurons that generate these spikes. For a more realistic visual stimulus, a much larger neural network is needed for stimulus reconstruction. Therefore, a massive number of neurons is required to process the information encoded by a relatively small number of neurons. This observation may explain why there is an explosively larger number of spiking neurons in V1 than in the retina.

How to model the computation of the entries of the matrix $\mathbf{G}$ and the vector $\mathbf{q}$ with processes native to dendritic trees will be described elsewhere.

VII. Acknowledgments

The research reported here was supported by the AFOSR under grant # FA9550-09-1-0350 and, in part, by a grant of computer time from the City University of New York High Performance Computing Center under NSF Grants CNS-0855217 and CNS-0958379.

References

[1] A. Lazar and L. Toth, "Perfect Recovery and Sensitivity Analysis of time Encoded Bandlimited Signals," *IEEE Transactions on Circuits and Systems-I: Regular Papers*, vol. 51, no. 10, pp. 2060–2073, Oct 2004.

[2] A. A. Lazar and E. A. Pnevmatikakis, "Video time encoding machines," *IEEE Transactions on Neural Networks*, vol. 22, no. 3, pp. 461–473, March 2011.

[3] A. A. Lazar, E. A. Pnevmatikakis, and Y. Zhou, "Encoding natural scenes with neural circuits with random thresholds," *Vision Research*, vol. 50, no. 22, pp. 2200–2212, October 2010, special Issue on Mathematical Models of Visual Coding.

[4] P. R. Kinget, A. A. Lazar, and L. T. Toth, "On the robustness of the vlsi implementation of a time encoding machine," in *IEEE International Symposium on Circuits and Systems*, May 2005.

[5] A. A. Lazar, E. K. Simonyi, and L. T. Toth, "An overcomplete stitching algorithm for time decoding machines," *IEEE Transactions on Circuits and Systems-I: Regular Papers*, vol. 55, no. 9, pp. 2619–2630, October 2008.

[6] J. Harris, J. Xu, M. Rastogi, A. Singh-Alvarado, V. Garg, J. Principe, and K. Vuppamandla, "Real Time Signal Reconstruction from Spikes on a Digital Signal Processor," in *IEEE International Symposium on Circuits and Systems*, no. 18-21, 2008, pp. 1060–1063.

[7] A. A. Lazar, "A Simple Model of Spike Processing," *Neurocomputing*, vol. 69, pp. 1081–1085, June 2006.

[8] A. Cichochi and R. Unbehauen, *Neural Networks for Optimization and Signal Processing*. John Wiley & Sons, 1993.

[9] Y. Xia and J. Wang, "A Recurrent Neural Network for Solving Nonlinear Convex Programs Subject to Linear Constraints," *IEEE Transactions on Neural Networks*, vol. 16, no. 2, pp. 379–386, Mar 2005.

[10] Y. Xia, G. Feng, and J. Wang, "A Recurrent Neural Network for Solving Nonlinear Optimization Problems with Inequality Constraints," *IEEE Transactions on Neural Networks*, vol. 19, no. 8, pp. 1340–1353, 2008.

[11] J. Cruz-Albrecht and P. Petre, "Pulse Domain Linear Programming Circuit," US Patent 7724168, May 2010.