Nanoscale Memristor-based STDP Learning in a Radix-X Quantized Retinal Neural Network

Young Hwan Kim[1]*, Jason K. Eshraghian*[2]*, Yong Sook Goo*[3]*, and Kyoungrok Cho*[3]*

Young Hwan Kim
Pohang University of Science and Technology, South Korea
E-mail: youngk@postech.ac.kr

Jason K. Eshraghian
University of Michigan, Michigan, USA
University of Western Australia, WA, Australia
E-mail: jasonesh@umich.edu

Yong Sook Goo, Kyoungrok Cho
Chungbuk National University, South Korea
E-mail: ysgoo@chungbuk.ac.kr, krcho@cbnu.ac.kr

Keywords: retina, memristor, spiking neural model, spike-timing-dependent plasticity, STDP

The human retina sends visual signals to the brain's visual cortex from photoreceptors (rod and cone cells) through various synaptic pathways and performs crucial early vision processing before signals are passed to higher brain regions. This paper presents an artificial retina system implemented based on the leaky integrate-and-fire spiking neuron model. The architecture of the proposed retina system consists of a multilayer convolutional neural network (CNN), and the system uses spike-timing-dependent plasticity (STDP) as a feedforward learning rule. In addition, the system integrates a feedback plasticity learning rule to expedite learning convergence. The system weights are implemented using nanoscale memristor arrays, taking on a constrained (radix-X) range of conductance states. The proposed system produces an output image of 25×25 pixels, corresponding to the output retina ganglion cells that act as the interface between the retina and the visual cortex, using an input image of 100 ×100 pixels.

1. Introduction

In the field of artificial intelligence, neural networks are widely used as a means to apply learning techniques for solving many types of practical, and often high-dimensional, problems. The neural networks partially mimic several properties of biological neural networks, such as their hierarchical, cyclic nature. They have been successfully applied to various domains such as function approximation, image analysis, speech recognition, adaptive control, etc.[1-7] Biological neurons differ significantly from artificial neurons, with a simplified depiction shown in **Figure 1**. A stereotypically discrete signal is propagated through a synapse from one neuron to another, where the connections linking a pre-synaptic neuron's axon to the post-synaptic dendritic branch of a downstream neuron are the synapses. The signal is thought to propagate only in one direction, from the presynaptic neuron to the postsynaptic neuron, which can be represented in an acyclic manner. The levels of signal propagation intensity are modeled as network weights in the neural networks. During the learning process of neural networks, the values of interconnecting weights between neurons are iteratively adjusted.

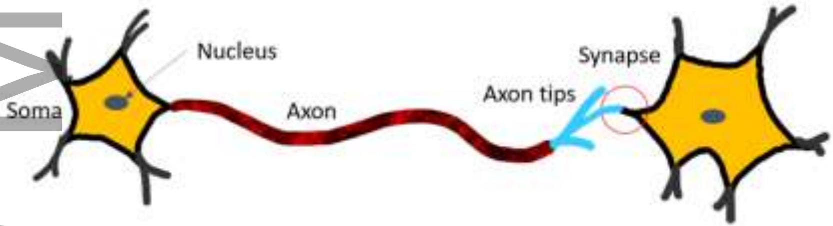


Figure 1. The neuron of the biological neural network

The neural networks consist of multi-layered structures of neurons and interconnections. Each connection between neurons of consecutive layers provides the output of one neuron (presynaptic neuron) to the input of the next neuron (postsynaptic neuron). One neuron may have more than one postsynaptic neuron, and in practice, may exceed 10s of 1000s of post-synaptic connections. Each neuron receives the weighted sum of the outputs of all presynaptic neurons as their input. An output is calculated by using an activation function. Some examples of the activation functions are the ReLU function, sigmoidal function, and tanh function. Although the neural networks are brain-inspired and have been remarkably successful in many applications, there are essential differences in the structure and computational methods when compared to the biological brain. The most fundamental differences is how information is encoded and propagated between nodes. With this observation, significant focus has shifted to investigating spiking neural networks (SNNs), referred to as the third generation neural networks. In SNNs, communication takes place via action potentials that are millivolt-scale fluctuations that last several milliseconds, and are referred to as spikes. Multiple spikes are often used to relay sensory input, referred to as spike trains, where it is thought that processing and computation take place using spike timing, which broadly encompasses latencies and spike rates. The use of spikes offers an engineering advantage by constraining inter-layer data communication to a single-bit event, and input-weight multiplication is simply traded for memory read-outs, which has been shown to significantly reduce both power consumption and latency.[8,9]

The retina is a light-sensitive layer of nerve cells lining the back wall inside the eye. It converts the light into neural signals and sends the signals to the brain to see objects. The biological structure of the retina is illustrated in **Figure 2**. As shown in the figure, the retina consists of photoreceptors, bipolar and ganglion cells, and amacrine and horizontal cells are connecting those cells. The cells in the retina can be represented as a neural network with synapses as shown in Figure 1.

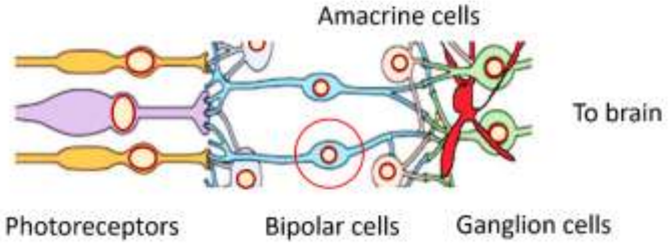


Figure 2. The biological structure of the retina

Although the retina is an essential component of the vision and central nervous systems, there have not been many studies to implement corresponding artificial systems, and the work in this area remains in the early stage.[10, 11] This paper presents the implementation of an artificial retina neural system. The proposed system is based on a nanoscale memristive retina neural network, and it uses spike-timing-dependent plasticity (STDP) learning with synaptic connections that are constrained to a Radix-X weight to assess performance in resource-constrained, quantized systems. STDP is a biological process that modulates the strength of connections between neurons, where potentiation and depression of the synaptic strength can be induced by precisely timed pairs of presynaptic and postsynaptic spikes.[12]

2. Memristor Devices as Synapses in a Spiking Neural Network

A memristor device is a non-volatile electronic memory device that has two terminals. The memristor was categorized as the fourth fundamental element following the resistor, the capacitor, and the inductor.[13] The original postulation of the memristor demonstrated how to program a resistance state via charge or flux, which enabled the resistance to be stored in a non-volatile manner. In practice, a variety of physical mechanisms are used to modulate resistance states, such as filamentary formation in a device, or mobile ion transport. Significant research has shown the capacity of a memristor to be used as the synapses of a spiking neural network, even with the spike-timing-dependent plasticity models.[14-18] Scientists have successfully shown that various materials such as carbon, chalcogenides, metal oxides, amorphous silicon, and polymer-nanoparticle composite materials exhibit memristive properties. In addition, it has already been shown that memristor synapses can support important functions such as STDP when integrated with metal-oxide-semiconductor neurons.[18]

The notion of memristors was first postulated by Leon Chua in 1971, based on the possible symmetry in the relationship between charge and flux. In May 2008, the HP research team of Stanley Williams published the experimental fabrication of the memristor.[19] Since then, researchers have been actively investigating resistive memories and exploring their possible applications, including memristive synapses in neuromorphic computing architectures.[20, 21] The device structure of a memristor is illustrated in **Figure 3**(a). The semiconductor thin-film has a region with a high concentration of dopants (positive ions), and this corresponds to a low resistance R_{on}. The remaining region has a low dopant concentration, and this corresponds to a much higher resistance R_{off}. We model the resistance of the device using the two variable resistors that are connected in series. We fabricated a memristor whose structure is the TiO_{2-x} /TiO_2 heterojunction layer, illustrated in Figure 3(b). The bottom electrode is ITO, and the top electrode is Ag. The upper TiO_{2-x} layer has a 5% oxygen deficiency. The atomic layer deposition (ALD) was used to deposit the TiO_{2-x} layer using titanium tetraisopropoxide and O_2 as a precursor and an oxygen source, respectively. The upper TiO_{2-x} layer is 20nm thick, and the lower TiO_2 layer is 2nm thick. Figure 3(b) and 3(c) illustrate the chip micrograph, cross-sectional view, and the fabricated memristor's I–V characteristic curve, respectively.

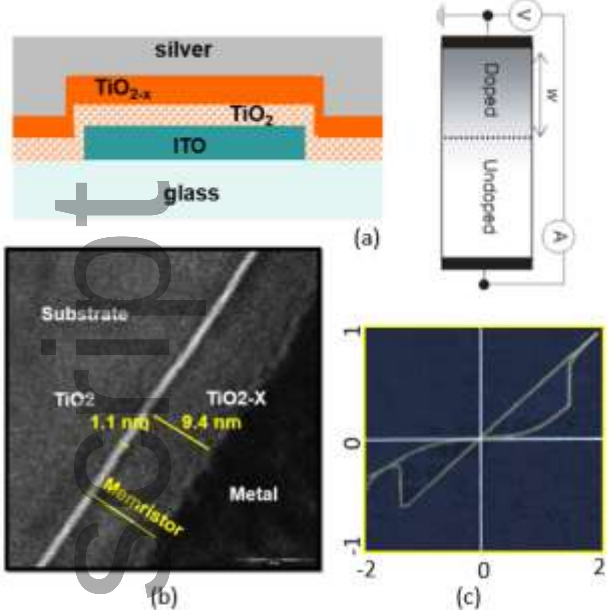


Figure 3. Fabricated memristor (a) device structure (b) cross-sectional view (the memristor thickness is 10.5 nm) (c) I–V characteristic curve

Figure 4 shows a typical memristor crossbar used as synapses with CMOS neurons in a neural network. In the figure, the cross points of the two layers are synapses, made of memristors, and their conductance values represent the connection weights between presynaptic and postsynaptic neurons.

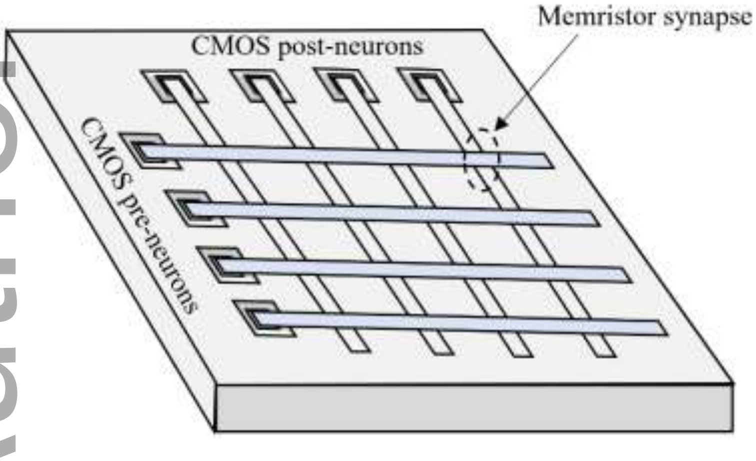


Figure 4. A memristor crossbar is used as synapses in a spiking neural network

3. Artificial Retina CNN Architecture

3.1 Proposed CNN Architecture

The biological retina can be abstracted into three layers from photoreceptors to ganglion cells, and the number of cells are reduced by a factor of approximately 100:1 on the path to the visual cortex. The proposed retina system is implemented using a convolutional neural network (CNN) architecture and memristor arrays. The architecture has two CNN layers and one fully connected layer, encoding the input image resolution from 100x100 to 25x25 which is a shrunk- down structure corresponding to the biological retina. The benefit of this step is the reduction of computational complexity in the following processing stages. **Figure 5** illustrates the proposed architecture.

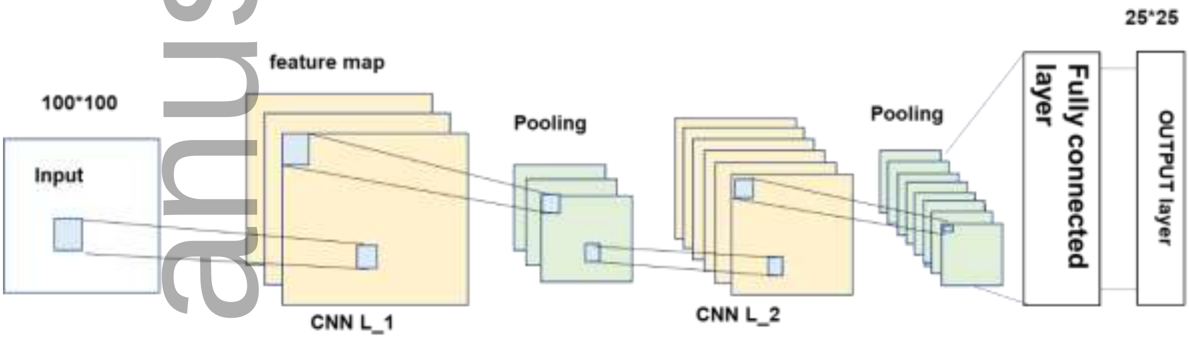


Figure 5. The proposed system architecture

The architecture uses three convolutional filters in the first layer, generating three feature maps from an input image. Then, a pooling layer downsamples the feature maps, summarizing the presence of features. For this, average pooling is performed. In the second layer, another convolution is performed using three 2×2 convolution filters. Finally, a fully connected layer is used for the output spike read-out. We verified the pattern recognition performance of the proposed architecture on the Tensorflow platform with character patterns generated similarly to MNIST. Figure 6(a) shows the examples of the 4×4 sized 360 test input patterns used in the experiments. They were made from a set of alphabetic characters. Figure 6.2(b) illustrates the recognition accuracy of three number systems, real number, radix-11 number, and binary number systems. As shown in the figure, the proposed Radix-11 CNN achieved the classification accuracy of 100 %, although it converged more slowly when compared to the real-number weight-based CNN. In the experiment, the binary number system did not get convergence to 100% accuracy.

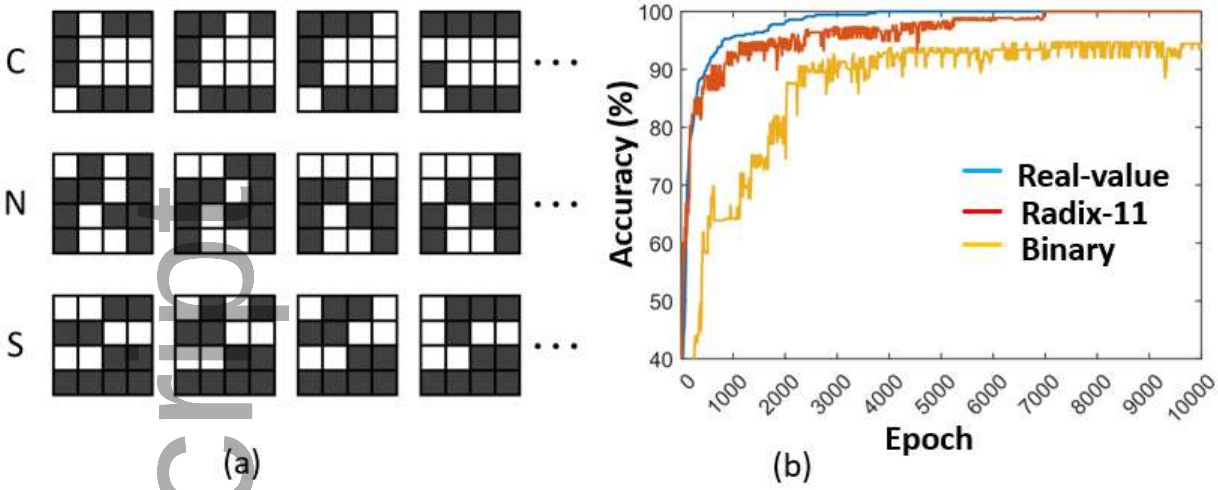


Figure 6. Performance evaluation of the proposed Radix-11 system (a) Examples of the 4×4 test input patterns (b) Inference accuracy for 3 number systems

3.2 Input PWM Circuit for SNN

The proposed system implements a spike neural model and employs unsupervised learning with STDP. In the neural network adapting the spiking neural model, the input activity level is controlled by adjusting the input current, i.e., the current magnitude and supply time. For example, rate coding or pulse width modulation (PWM) coding can be used with fixed current magnitude. In the case of an input driving a memristor, a voltage source in series with memristive devices can be used to adjust the input current magnitude in addition to the supplying time. In the proposed system, the PWM coding is used to represent the gray (intermediary) levels of monochromatic input images. **Figure 7** illustrates the rate coding and PWM coding representations of data. For other layers, the proposed system uses the memristor scheme to represent the input level.

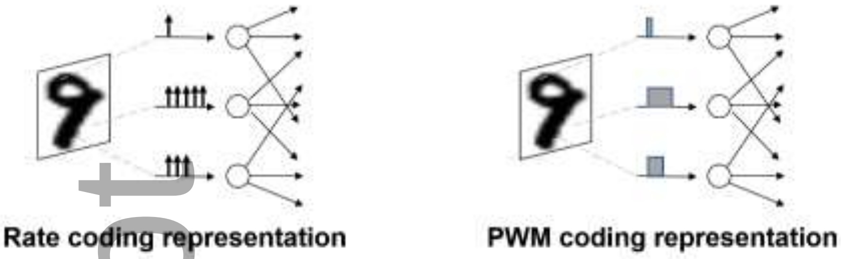


Figure 7. Rate coding and PWM coding techniques

The circuit for PWM coding can be implemented using logic gates and a MOSFET switch, as illustrated in **Figure 8**. We allocate a dedicated time slot for each input bit, where the length of the time slot depends on the weight of each bit. While the proposed system uses 8-bit input signals, a PWM coding circuit for the 4-bit input is shown in the figure. The circuit is for the input of 1001. In the circuit, the total time length to turn the current switch is 9⊿T.

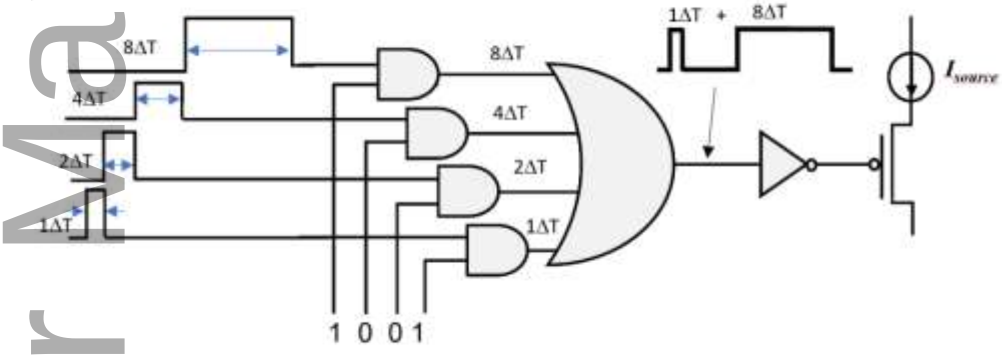


Figure 8. The circuit for PWM coding (Gray level for 1001)

3.3 The Leaky Integrate-and-Fire (LIF) Circuit

Amongst the most common neuron models to model spiking dynamics include the conductance-based Hodgkin-Huxley model family, although the integrate-and-fire class of models is more broadly adopted in neural network training due to their simpler dynamics, without compromising on spike-based encodings. Our proposed system uses the leaky integrate-and-fire (LIF) model, which contains a leak term in the membrane potential.[22] In this model, the membrane voltage decays exponentially with time. Thus, its behavior is usually represented by an RC circuit. The functional circuit of the LIF model is illustrated in **Figure 9**. In the circuit, resistor R_M represents the leak term. An input current I_{input} is controlled by adjusting its magnitude or supplied period time. If the membrane voltage V_m

reaches the given threshold voltage V_{TH}, the circuit generates a spike δ_i at the output. This spike is propagated to the next neural element, and it resets the membrane voltage to the initial state by turning the membrane switch on. The membrane voltage V_m is represented as follows in the form of the first-order ordinary differential equation.

$$I_{input}(t) - \frac{V_m(t)}{R_m} = C_m \frac{dV_m(t)}{dt} \qquad (1)$$

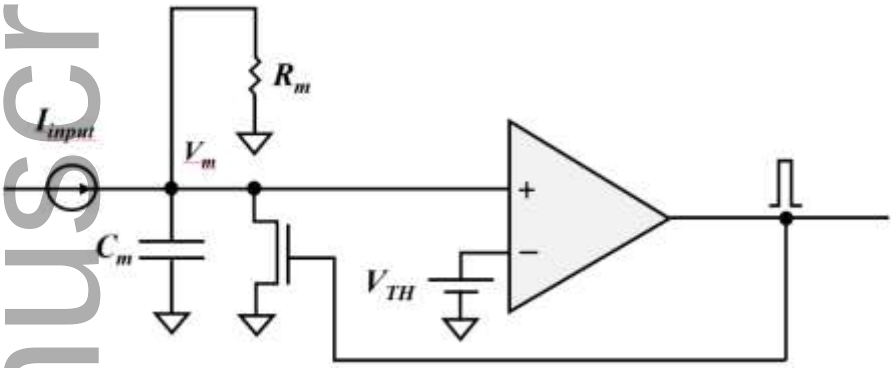


Figure 9. The functional circuit of the LIF model

The spike neural network using the LIF model is illustrated in **Figure 10**. As shown in the figure, a neuron accumulates the membrane potential using the presynaptic spikes. With every presynaptic spike, the membrane potential V_m increases and then decreases with time. Once V_m reaches V_{TH}, a neuron generates a spike at the output, and it resets V_m to the ground level.

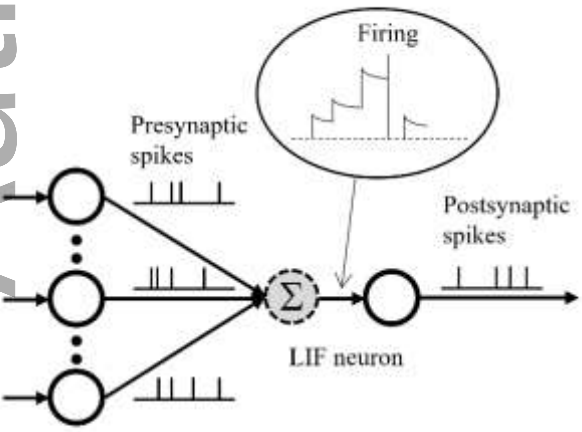


Figure 10. The spike neural model to build the proposed retina system

3.4 Neural Network with Weighted-Memristor Array

A memristor crossbar can parallelize many multiplication and addition operations, corresponding to matrix dot product computation, which is fundamental for mapping to a neural network. The mapping concept is shown in **Figure 11**. The input spike of the neural network corresponds to the input voltage of the crossbar, and the weight corresponds to the conductance of the memristor. In the memristor crossbar, the current generated by the input voltage and conductance is summed in one column (or bit-line), which functionally corresponds to how input and synaptic conductance are weighted and summer in an artificial neuron model. **Equation (2)** expresses a column operation on a crossbar network.

$$i_{total} = \sum_{i=0}^{n} v_i * g_i \qquad (2)$$

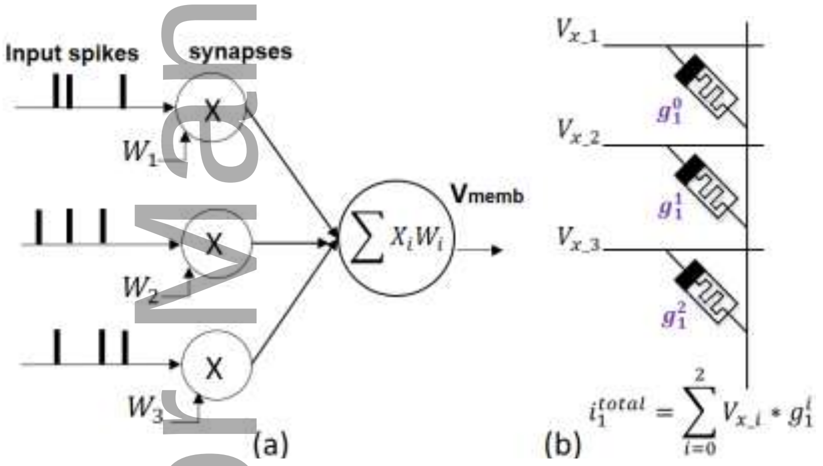


Figure 11. Mapping of a memristor array (a) a neural network (b) a corresponding memristor array

The ide I-V characteristics of a memristor are shown in **Figure 12**. The slope of the ‘butterfly wings’ represents the memristor conductance, which in the ideal case, depends on the energy applied to the memristor. We used an ideal memristor curve instead of the one shown in Figure 3 to simulate the proposed Radix-11 number system as highly advanced manufacturing technology is required to use the fabricated memristors for system implementation. In this paper, we apply three types of fixed energy values that result in three different resistances, as shown in **Equation 3**. While the analog characteristics of memristors and current-mode operation increase susceptibility to variations of conductance states amongst memristors, spiking dynamics enable a significant degree of noise to be absorbed into the subthreshold dynamics of spiking neuron models[21,23].

$$\begin{cases} G_0 = 2^{(V_t+V_a)} = 2^{v_t} * 2^0 & = 1\ G_m \\ G_1 = 2^{(V_t+V_b)} = 2^{v_t} * 2^1 & = 2\ G_m \\ G_2 = 2^{(V_t+V_c)} = 2^{v_t} * 2^2 & = 4\ G_m \end{cases} \text{, } where\ V_a = 0\ V, V_b = 1\ V, V_c = 2\ V \qquad (3)$$

The voltage V_t is the threshold voltage, which turns on the memristor, and V_a, V_b, and V_c are the additional voltages that are added to V_t to obtain the conductance. For example, an applied voltage of $V_b = 1$ V sets a resistance of 2 G_m, where G_m is the ON conductance of the memristor.

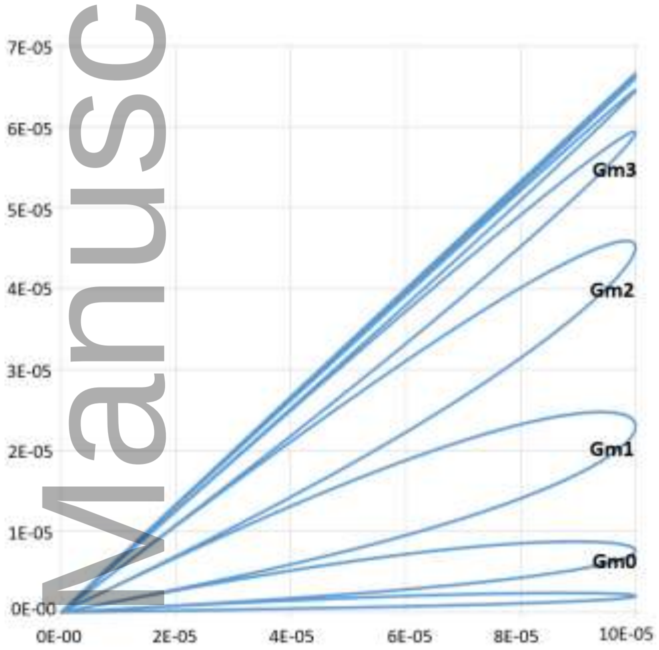


Figure 12. The I-V characteristic of the memristor

In this paper, we propose a weighted memristance that combines three resistances at a cross-point. An example is given in **Figure 13**. The equivalent resistance is obtained between the vertical line and the horizontal line. Combining the three weighted conductance of 1 R_m, 0.5 R_m, and 0.25 R_m, which are in parallel, provides 11 different conductance values at a cross-point, which is mapped to Radix 11 numbers. For example, the combination of 0.5 R_m and 0.25 R_m is equivalent to 0.75 R_m. This represents a weight value in the proposed Radix-11 approach.

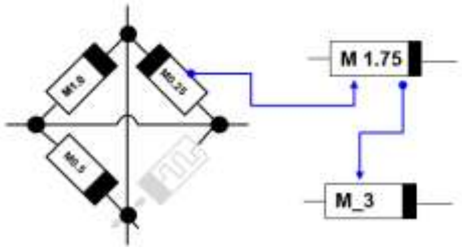
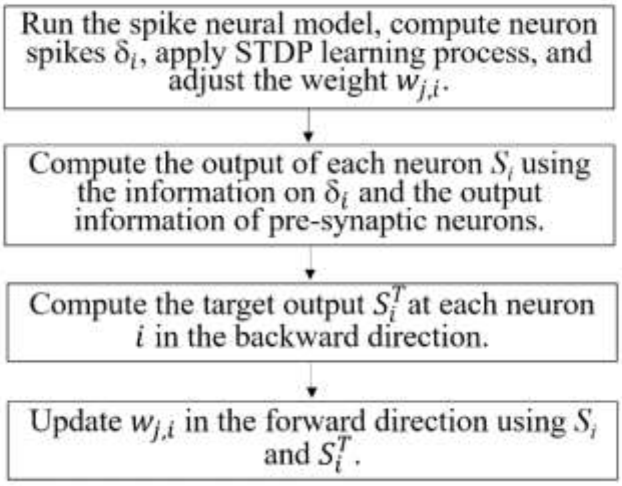


Figure 13. Three weighted memristors on a cross-point, equivalent resistance, and radix_11 symbol mapped to it

4. The proposed STDP-Based Learning Method

The learning process of the proposed system consists of two steps, the feedforward STDP learning followed by the feedback plasticity learning process. For the feedback plasticity learning process, it was shown that the output of the LIF neuron, to be denoted as S_i, can be used to regulate the network weights in addition to spikes.[24] In the proposed system, we use the same idea for the feedback learning process. The learning process of the proposed system is illustrated in **Figure 14**. The learning process is performed during each learning epoch.



Run the spike neural model, compute neuron spikes δ_i, apply STDP learning process, and adjust the weight $w_{j,i}$.

Compute the output of each neuron S_i using the information on δ_i and the output information of pre-synaptic neurons.

Compute the target output S_i^T at each neuron i in the backward direction.

Update $w_{j,i}$ in the forward direction using S_i and S_i^T.

Figure 14. Overall learning process during each learning epoch

In the proposed learning algorithm, the output of the LIF neuron i, $S_i(\mathrm{t}+1)$, is given as follows:

$$S_i(\mathrm{t}+1) = \sum_{j}^{N} w_{j,i} S_j(t+1) + \tau\delta_i(t+1) \qquad (4)$$

where $w_{j,i}$ is the weight between neuron j and i, δ_i is the spike at neuron i, and τ is the coefficient to control the magnitude of the output. $S_i{}'$s are computed from the first layer to the last layer in the forward direction, and $S_i^T{}'$s are computed backward using the prediction error of S_i from the last layer.

4.1 Feedforward Spike-Timing-Dependent Plasticity (STDP) Learning

4.1.1 Input Spike Train

We consider a feedforward network with N input $x_i(t)$ as Dirac delta spike trains connected to a single output neuron through the weights $W_i(t)$ and giving rise to the postsynaptic spike train $y(t)$.[25] The input spike trains exhibit average firing rates. We considered a feedforward network. Let $x(t)=[x_1(t), ..., x_N(t)]^T$ denote the vector of N inputs where $x(t)$ is given in **Equation (5)** that is the Dirac delta spike train of neuron i at time t, where t_i is the spike timing.

$$X_i(t) = \sum_{t_i} \delta(t - t_i) \qquad (5)$$

4.1.2 Spike-Timing-Dependent Plasticity (STDP)

SNNs communicate via neurons that emit binary spikes. STDP is a learning rule that modulates the strength of a synaptic connection based on the temporal correlation of spikes between connected neurons in the SNN. This section provides an overview of STDP learning. Although there are symmetric and asymmetric STDP rules for changing the strength of synapses, only the asymmetric STDP rule is considered in this work.[26]

Pair-based STDP

In general, STDP requires a lot of computation and delays in the updating process of synaptic weights because the learning function has to be calculated for every pair of presynaptic and postsynaptic spikes.[27] Here we assume that neurons A and B are synaptically connected, and the spike signal is transmitted from neuron A to neuron B. When neuron A fires and then neuron B fires within a particular time window, the synaptic connection is strengthened in the long-term potentiation (LTP) mode.[28] Conversely, in spike signaling, if B fires and then neuron A fires, long-term depression (LTD) is induced, resulting in decreased synaptic connections between them. We define the firing time of neuron A as t_{pre} and the firing time of neuron B as t_{post}. Then, the timing difference between the presynaptic and postsynaptic spikes can be expressed as $\Delta t = t_{pre}-t_{post}$. The amount of weight modification Δw between two synapses can be written as follows.

$$\Delta w = \begin{cases} A^- \ e^{-\Delta t_1/\tau_+} & \Delta t_1 < 0 \\ A^+ \ e^{-\Delta t_1/\tau_+} & \Delta t_1 > 0 \end{cases} \qquad (6)$$

where A^+ and A^- are positive and negative constants, respectively, which determine the maximum amount of synaptic change, and τ_+ is the potentiation time constant, and τ_- is the depression time constant.

Triplet-based STDP

It is known that triplet-based STDP is more biologically plausible than pair-based STDP.[21] Triplet-based STDP is a method of adjusting synaptic weights when the triplet spikes occur in the sequence of presynaptic-postsynaptic-presynaptic spikes or postsynaptic-presynaptic-postsynaptic spikes. The synaptic coupling strength is depressed when the activation of the postsynaptic neuron is low and potentiated when the activation is high. This is a key property of the Bienenstock–Cooper–Munro (BCM) synaptic learning rule. The BCM synaptic rule has been shown to maximize the selectivity of postsynaptic neurons, and it provides a possible explanation for experience-dependent cortical plasticity, such as directional selectivity.[30]

In the triplet model of spike-timing–dependent plasticity (STDP), learning depends on the interactions of three precisely timed spikes. It has been shown that this learning model describes plasticity experiments that the classical pair-based STDP rule has failed to capture.[26, 29] **Figure 15**(a) shows the triplet spikes associated with LTD and LTP. LTD traces the temporal relationship of two presynaptic spikes and one postsynaptic spike, as shown in the figure. If the postsynaptic spike occurs first and the presynaptic spike occurs, the presynaptic spike does not affect the next neuron. Conversely, in the case of LTP, if the presynaptic spike stimulates the postsynaptic neuron, the presynaptic neuron potentiates the membrane potential of the postsynaptic neuron. Figure 15(b) shows the synaptic weight variation by the temporal difference of these spikes.

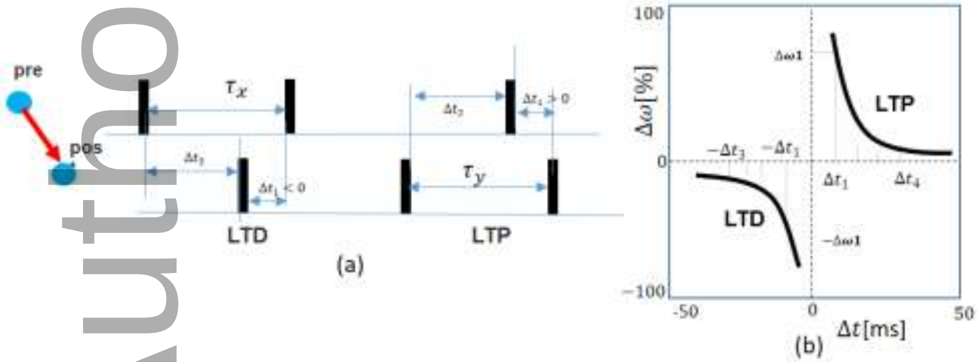


Figure 15. STDP learning rule (a) Triplet spike train for LTD and LTP (b) synaptic depression and potentiation are induced by $\Delta t_1 = t_{post} - t_{pre}$ using the triplets of spikes

The weight variation is given as **Equation (7)**

$$\begin{aligned} \Delta W_{-} &= n^{-}(1 + A^{+}\, e^{-\Delta t_1/\tau_{-}})e^{-\Delta t_3/\tau_x} \text{ if } \Delta t_1 < 0 \\ \Delta W_{+} &= n^{+}(1 + A^{-}\, e^{-\Delta t_1/\tau_{+}})e^{-\Delta t_2/\tau_y} \text{ if } \Delta t_1 > 0 \end{aligned} \tag{7}$$

where the synaptic weight is potentiated at times when a postsynaptic spike occurs or depressed at the time when a presynaptic spike occurs. The depression and potentiation amplitude parameters are A^{-}, n^{-}, A^{+}, and n^{+}, while $\Delta t_1 = T_{\text{post}} - T_{\text{pre}}$, $\Delta t_3 = T_{post} - T'_{pre}$, and , $\Delta t_2 = T_{pre} - T'_{pos}$, are the time differences between pre(t) and pos(t), and post(t) and pre(t-1), and pos(t) and pos(t-1), respectively. Here, τ_{-} and τ_{x} and τ_{+}, τ_{y} time constants that determine the windows in which depression and potentiation take place.[27, 31, 32] Besides, it is needed to distinguish between the current spike and the previous spike of the same type. The weight change is described with the STDP learning rate η as shown in **Equation (8).** We can train the system by applying this unsupervised learning rule.

$$W_t = W_{t-1} + \eta \Delta W \qquad (8)$$

4.2 Feedback Plasticity Learning

STDP is a model of the biological process. Although STDP works well for adjusting the connection strengths of the spike neural network, it takes a long time to transfer the information to the deep network layers. As this makes learning convergence challenging, several alternative approaches have been adopted to use any available information other than the spike information to expedite the convergence of the network.[23, 33, 34, 35] The second step of the proposed learning algorithm is to update synaptic weight backward from the last synaptic weights using the output of neuron S_i.[22] Let S_i^T denote the target S at layer i. Then, the prediction error e_i is defined as follows.

$$e_i = \sum_{i=1}^{n} |S_i - S_i^T| \qquad (9)$$

where n is the number of sample nodes at layer i. Let L denote the number of total layers. With supervised learning, S_L^T is given, and, thus, e_L is given. The target S of the penultimate layer, S_{L-1}^T, is given as follows.[22]

$$S_{L-1}^T = S_{L-1} - \eta_t \Delta S = S_{L-1} - \eta_t W_{L-1}^T \cdot e \qquad (10)$$

where η_t represents the learning rate of the target. For $i = 1,2,\ldots,L-2$, the target output S_i^T can be computed as follows [10].

$$S_i^T = S_i - G_i(e) = S_i - B_i * e + b_i \qquad (11)$$

In the above equation, B_i denotes the random feedback weight of the i^{th} layer, and b_i represents the random feedback bias. After computing $S_i{}'$s and $S_i^T{}'$s, the proposed algorithm makes the final weight adjustment Δw in the forward direction, where Δw and W are given as follows:[22]

$$\begin{aligned} \Delta w &\propto S_j\,(S_i - S_i^T) \\ W &= W - \eta_w \Delta w \end{aligned} \qquad (12)$$

where S_j and S_i indicate the presynaptic and postsynaptic output, and η_w is the learning rate of the weight.

5. CNN Implementation and Simulation

In this work, we develop a training methodology for the convolutional layers that consists of an input layer followed by intermediate hidden layers and a final classification layer. The hidden layers consist of multiple convolutional and pooling layers, which are often arranged in an alternating manner. These convolutional and pooling layers represent the intermediate stages of the feature extractor. The spikes from the feature extractor are combined to generate a one-dimensional vector input for the fully-connected layers to produce the final classification. The convolutional and fully-connected layers contain trainable parameters (i.e., synaptic weights).

Figure 16 illustrates the convolution process for making down the resolution of retina output image at ganglion cell, which uses three of the 2×2 2-D filters. Figure 16(d) is the output of the convolution process. The implementation of the convolution circuit used for the input is illustrated in **Figure 17**. The image signal is modulated into a PWM signal and input to the LIF circuit, as shown in Figure 17(a). The LIF circuit accumulates the signal energy and fires when the membrane voltage goes over the threshold voltage. It, then, returns to the initial state, as shown in Figure 17(b). In Figure 17(c), the pulses obtained from the LIF stimulate the convolution circuit as the input spikes. The convolution circuit uses a weighted memristor array, and it incorporates a compensation column. The compensation is carried out with OP amplifiers at the output stage that allows the use of the negative value of the weight. In Figure 16, kernel_1, kernel_2, and kernel_3 implement the first, the second, and the third 2×2 2-D filter of Figure 16(c). The three sets of four input spike trains $\begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$, $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$, and $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ came from the first two rows of the input spikes of Figure 16(b). Their expected outputs are illustrated in the first row of each 3×3 table of Figure 16(d). The circuit of Figure 17 was simulated by using Hspice.[34] The simulation results are shown in **Figure 18**. In Figure 18, V(23), V(24), and V(25) represent these output values. These outputs are the membrane potential of a neuron and become the input signal to the next neuron. Therefore, it is formalized into spikes through the LIF stage and becomes the spike train of the next stage. We find that the magnitudes of the three pulses of V(23) are in good agreement with the first row of the output spikes of Figure 16(e), which is highlighted with a white box.

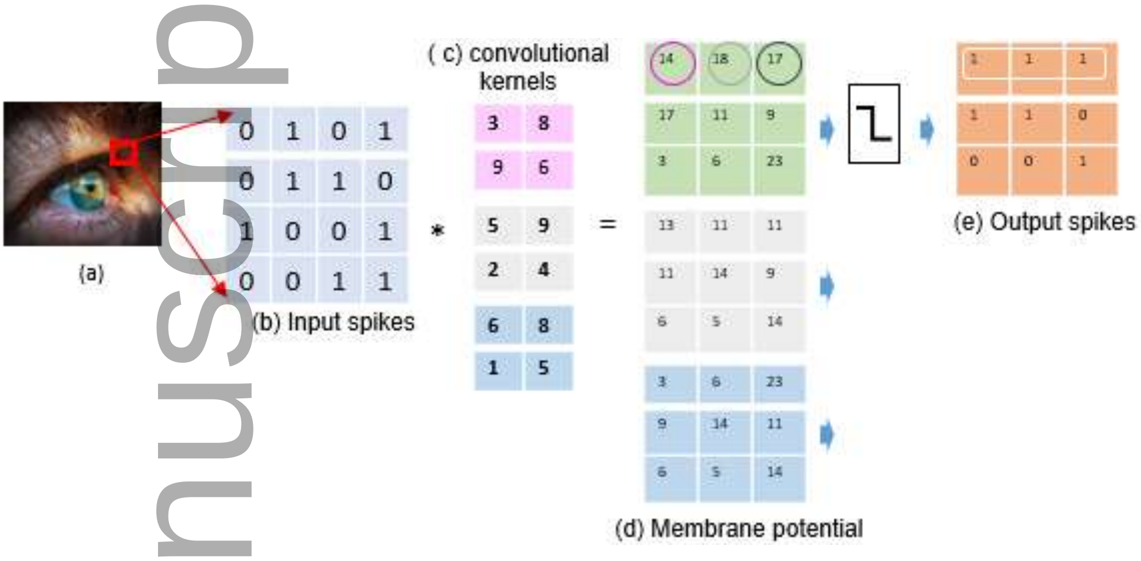


Figure 16. The signal in a CNN (a) Image (b) input spike train (c) three 2×2 convolution filters (d) convolution results (e) output spikes

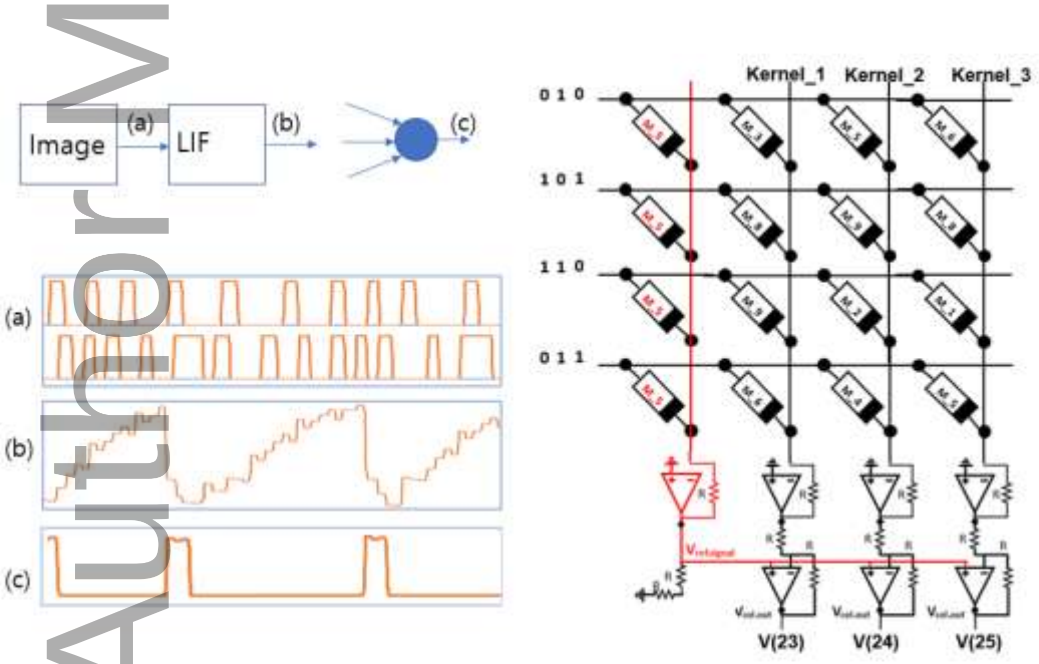


Figure 17. The convolution circuit for the three 2×2 convolution filters of Figure 16 (The final output is connected to the input of the next stage in the form of a pulse.)

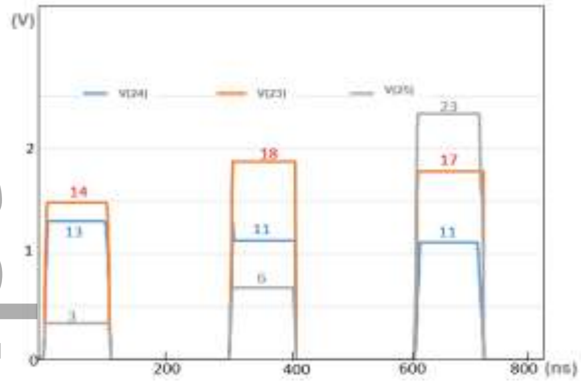


Figure 18. Simulation results of Figure 17 (Three colored outputs are mapped to the first row of Figure 16(d).)

6. Conclusions

Artificial intelligence seeks to draw inspiration from the capabilities of human intelligence processes in solving problems or making decisions with computer systems. Over the past few decades, there has been tremendous progress in the field of artificial intelligence. The areas that have made such great strides include speech recognition and computer vision. This paper proposes an artificial retina network implemented in a multilayer convolutional neural network, popularly used in speech recognition and computer vision. The retina is a light-perceptive organ, and an essential component of the central nervous system as well. This means that the retina can process visual information, and this local processing reduces the cost of transmitting the visual signal from photoreceptors to the visual cortex.

The implemented artificial retina network consists of an input layer, an output layer, one fully-connected layer, and two hidden layers. For each hidden layer, three of 2×2 convolution filters are used. The proposed system, based on the spike-timing-dependent plasticity, is trained using the feedforward plasticity learning and the feedback plasticity learning algorithms. For realizing the synapse weights, the proposed system uses nanoscale memristor arrays with Radix-X constrained weights. The proposed system approximates the given input image of 100 ×100 pixels and produces an output image of 25×25 pixels, corresponding to the output of retina ganglion cells connecting to the visual cortex, from an input image of 100 ×100 pixels mimicking the image processing process of the biological retina system. The behavior of the convolutional circuit in the system was simulated using a circuit simulator, HSpice.[36] The simulation results are in good agreement with the high-level convolution results. The simulation results suggest the proposed artificial retina system has a high probability of working if implemented as a microchip.

Acknowledgements

This research was supported by the Basic Science Research Program (NRF-2020R1F1A1069381 and NRF-2022R1A2C2004793) through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (MSIP).

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] S. Elfwing, E. Uchibe, and K. Doya, *Neural Networks*, **2018**, 107, 3–11.

[2] F. Scarselli and A. C. Tsoi, *Neural networks*, **1998**, 11, 15–37.

[3] M.Egmont-Petersen, D.de Ridder, and H.Handels, *Pattern Recognition*, **2002**, 35, 10, 2279-2301.

[4] T. Mikołajczyk, K. Nowicki, A. Bustillo, and D. Yu Pimenov, *Mechanical Systems and Signal Processing*, **2018**, 104, 503-513.

[5] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh and K. Shaalan, *IEEE Access*, **2019**, 7, 19143-19165.

[6] K. Kim, K. Eshraghian, H. Kang, K. Cho, *J. of Nano. and Nanotech*. 2021, 18910.

[7] A. J. Ijspeert, *Neural Networks*, **2008**, 21, 642–653.

[8] J. K. Eshraghian, M. Ward, E. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, W. D. Lu, *arXiv preprint arXiv:2109.12894*, **2021.**

[9] M. Azghadi, C. Lammie, J. K. Eshraghian, M. Payvand, E. Donati, B. Linares-Barranco, G. Indiveri, *IEEE Transactions on Biomedical Circuits and Systems*, **2020**, 14, 1138-1159.

[10] F. Sadikoglu, S. Uzelaltinbulat, *Procedia Com. Sci.*, **2016**, 2016.09.365.

[11] L. Bao, J. Kang, Y. Fang, Z. Yu, Z. Wang, Y. Yang, Y. Cai, R. Huang, *Sci Rep.* **2018**, 31958-6.

[12] M. C. W. van Rossum, G. Q. Bi, G. G. Turrigiano, *J. of Neurosci.*, **2000**, 20-23-08812.2000.

[13] L. Chua, *IEEE Trans. on Circuit Theory*, 1971, 1083337.

[14] S. Jo, T. Chang, I. Ebong, B. Bhavitavya, P. Bhadviya, Mazumder, W. Lu, *Nano Letters*, **2010,** *10* , 1297-1301.

[15] S. Kim, C. Du, P. Sheridan, W. Ma, S. Choi, W. Lu, *N*ano Lett., **2015,** *15*, 2203-2211.[16] A. Adamatzky, L. Chua, Memristor Networks, *Springer Science & Business Media*, NY USA, **2013,** 2013957702.

[17] S. Kang, D. Choi, J. K. Eshraghian, P. Zhou, J. Kim, B. S. Kong, X. Zhu, A. S. Demirkol, A. Ascoli, R. Tetzlaff, W. D. Lu, L. O. Chua, *IEEE Transactions on Circuits and Systems I: Regular Papers*, **2021.**

[18] M. Rahimi Azghadi, Y. C. Chen, J. K. Eshraghian, J. Chen, C. Y. Lin, A. Amirsoleimani, A. Mehonic, A. J. Kenyon, B. Fowler, J. C. Lee, Y. F. Chang, *Advanced Intelligent Systems*, **2020**, 1900189.

[19] D. Strukov, G. Snider, D. Stewart, and R. Williams, *Nature*, **2008**, 06932.

[20] Z. Wang, S. Joshi, S. Savel'ev, H. Jiang, R. Midya, P. Lin, M. Hu, N. Ge, J. P. Strachan, Z. Li, Q. Wu, M. Barnell, G. Li, H. L. Xin, R. S. Williams, Q. Xia, and J. J. Yang, *Nature Mater*, **2017,** 4756.

[21] J. K. Eshraghian, X. Wang, W. D. Lu, *IEEE Nanotechnology Magazine*, **2022**.

[22] A. N. Burkitt, *Biol Cybern*, **2006**, 95, 97–112.

[23] J. K. Eshraghian, W. D. Lu, *arXiv preprint arXiv:2201.11915*, **2022.**

[24] D. Zhao, Y. Zeng, T. Zhang, M. Shi, F. Zhao, *Frontiers in Computational Neuroscience*, **2020**, 576841.

[25] T. Amirhossein and M. Anthony, *Neurocom.*. **2018**, 11.014.

[26] T. Voegtlin, *BMC Neurosci*, **2009**, 1471-2202-10-S1-P139.

[27] J. Gjorgjieva, C. Clopath, J. Audet, J. Pfister, *PNAS* , **2011** , 1105933108.

[28] K. Furuya, J. Ohkubo, *Journal of the Physical Society of Japan*, **2021**, 90, 074802.

[29] R. Kempter, W. Gerstner, van H. Leo, Hebbian learning and spiking neurons. *Phys. Rev.*, **1999**, 59.4498.

[30] M. Rahimi Azghadi, N. Iannella, S. F. Al-Sarawi, G. Indiveri and D. Abbott, *Proceedings of the IEEE*, **2014**, 2314454.

[31] J.P. Pfister, W. Gerstner, *J. Neurosci.* **2006**, 10.1523.

[32] R. Froemke, Y. Dan, *Nature*, **2002**, 416433a.

[33] P. U. Diehl, M. Cook, *Front. Comput. Neurosci.*, **2015**, 00099.

[34] T. Zhang, Y. Zeng, D. Zhao, M. Shi, *in Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, **2018**.

[35] J. H. Lee, T. Delbruck, M. Pfeiffer, *Front. Neurosci.*, **2016**, 00508.

[36] Synopsys,. HSPICE User Guide: Simulaton aqnd Analysis. .Version B-20088.09, Synopsys, **2008**.

Author biographies

Young Hwan Kim received the Ph.D. degree in electrical engineering from the University of California, Berkeley, USA, in 1988. From 1988 to 2020, he was a Professor with the Electrical Engineering, POSTECH, South Korea, where he is currently a Professor Emeritus. His research interests include display systems, design technology for deep-submicron VLSI systems, computer vision, and speech recognition. He has served as a journal editor, IEIE of Korea, and as the General Chair and a Committee Member for various Korean domestic and international technical conferences, including International SoC Design Conference, IEEE ISCAS 2012, and IEEE APCCAS 2016.

Jason K. Eshraghian received a Ph.D. degree from The University of Western Australia in 2020. He is currently a Post-Doctoral Researcher and Fulbright Scholar with the University of Michigan. He was the Track Chair of the IEEE ISCAS in 2019 and 2020. Jason was awarded the best paper awards for the 2019 IEEE VLSI Systems and the 2019 IEEE Artificial Intelligence Circuits and Systems Conference, and the 2020 Best Live Demonstration Award at the IEEE International Conference on Circuits, Electronics and Systems. His research includes neuromorphic computing, spiking neural networks, and in-memory computing using memristors.

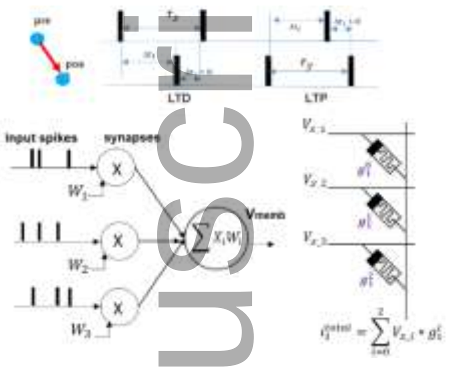
Yong Sook Goo received the M.D. degree and the Ph.D. degree in physiology from Seoul National University, Seoul, South Korea, in 1988 and 1993, respectively. She completed Post-Doctoral Training from the Department of Physiology, Tulane University Medical School, in 1998. She is currently a Professor of Physiology with the School of Medicine, Chungbuk National University Cheongju, South Korea. Her current research interest is on retinal network analysis. Dr. Goo has directed several projects funded by the KOSEF, KRF, and Ministry of Health and Welfare on retinal network study for retinal prosthesis since 2001.

Kyoung-Rok Cho received a Ph.D. degree in Electrical Engineering from the University of Tokyo, Japan, in 1992. From 1979 to 1986, he was with LG Electronics in Korea. He is currently a Full Professor at the College of Electrical and Computer Eng. of Chungbuk National University, Korea. His research interests include high-speed and low-power circuit design, SoC and platform design, CMOS image sensors, and memristive systems. He was awarded the IEEE Very Large Scale Integration Systems Best Paper Award and IEIE award in 2019.

Table of Contents

Title ((no stars)) Nanoscale Memristor-based STDP Learning in a Radix-X Quantized Retinal Neural Network

Young Hwan Kim, Jason K. Eshraghian, Yong Sook Goo, and Kyoungrok Cho*



An artificial retina system is implemented based on the leaky integrate-and-fire spiking neural model. The architecture of the proposed retina system consists of a multilayer convolutional neural network (CNN), and the system uses spike-timing-dependent plasticity (STDP) as a feedforward learning rule. The system weights are implemented using nanoscale memristor arrays, taking the constrained (radix-X) range of conductance states.