

Self-Programming Synaptic Resistor Circuit for Intelligent Systems

Christopher M. Shaffer, Atharva Deo, Andrew Tudor, Rahul Shenoy, Cameron D. Danesh, Dhruva Nathan, Lawren L. Gamble, Daniel J. Inman, and Yong Chen*

Unlike artificial intelligent systems based on computers which have to be programmed for specific tasks, the human brain “self-programs” in real time to create new tactics and adapt to arbitrary environments. Computers embedded in artificial intelligent systems execute arbitrary signal-processing algorithms to outperform humans at specific tasks, but without the real-time self-programming functionality, they are preprogrammed by humans, fail in unpredictable environments beyond their preprogrammed domains, and lack general intelligence in arbitrary environments. Herein, a synaptic resistor circuit that self-programs in arbitrary and unpredictable environments in real time is demonstrated. By integrating the synaptic signal processing, memory, and correlative learning functions in each synaptic resistor, the synaptic resistor circuit processes signals and self-programs the circuit concurrently in real time with an energy efficiency about six orders higher than those of computers. In comparison with humans and a preprogrammed computer, the self-programming synaptic resistor circuit dynamically modifies its algorithm to control a morphing wing in an unpredictable aerodynamic environment to improve its performance function with superior self-programming speeds and accuracy. The synaptic resistor circuits potentially circumvent the fundamental limitations of computers, leading to a new intelligent platform with real-time self-programming functionality for artificial general intelligence.

1. Introduction

The human brain has long served as the inspiration of artificial intelligent systems. A neural network (Figure 1a) processes voltage pulses at M presynaptic neurons inducing currents via synapses at N postsynaptic neurons by following the signal processing algorithm

$$I = wx \quad (1)$$

where $w = (w_{nm}) \in \mathbb{R}^{N \times M}$ denotes a matrix with w_{nm} as the weight (conductance) of a synapse connecting the m^{th} presynaptic and the n^{th} postsynaptic neuron, $x = (x_m) \in \mathbb{R}^M$ denotes a vector with x_m as the voltage pulses at the m^{th} presynaptic neuron, and $I = (I_n) \in \mathbb{R}^N$ denotes a vector with I_n as the current flowing into the n^{th} postsynaptic neuron, which triggers the voltage pulses $y = (y_n) \in \mathbb{R}^N$ with y_n as the pulses output from the n^{th} postsynaptic neuron. Concurrently, the synaptic weight matrix, w , is modified by following the learning algorithm^[1,2]


$$\dot{w} = \alpha z \otimes x \quad (2)$$

where α denotes the modification coefficient, $z = (z_n) \in \mathbb{R}^N$ denotes a function of $y = (y_n) \in \mathbb{R}^N$ with y_n as the voltage pulses at the n^{th} postsynaptic neuron (Equation S1, Supporting Information), and $z \otimes x$ represents the outer product between z and x . By integrating signal processing, memory, and correlative learning functions in each synapse, a neural network concurrently executes the signal-processing (Equation (1)) and learning (Equation (2)) algorithms in analog parallel mode to dynamically self-program w and create new functions in real time in unpredictable and arbitrary environments with general intelligence.^[1,3,4]

Computers embedded in artificial intelligent systems can execute arbitrary signal-processing algorithms^[5] to outperform humans at specific tasks such as pattern recognition^[6] and the Go game,^[7] but they have to be preprogrammed by humans and cannot adapt or develop new functions in unpredictable and arbitrary environments as humans do.^[4] The time and energy consumption to compute machine learning algorithms from a dataset with M -dimensional variables increase versus

Dr. C. M. Shaffer, A. Deo, Dr. A. Tudor, R. Shenoy, Dr. C. D. Danesh, D. Nathan, Prof. Y. Chen
Department of Mechanical and Aerospace Engineering
University of California
Los Angeles, CA 90095, USA
E-mail: yongchen@seas.ucla.edu

Dr. L. L. Gamble, Prof. D. J. Inman
Aerospace Engineering Department
University of Michigan
Ann Arbor, MI 48109, USA

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/aisy.202100016>.

© 2021 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.202100016

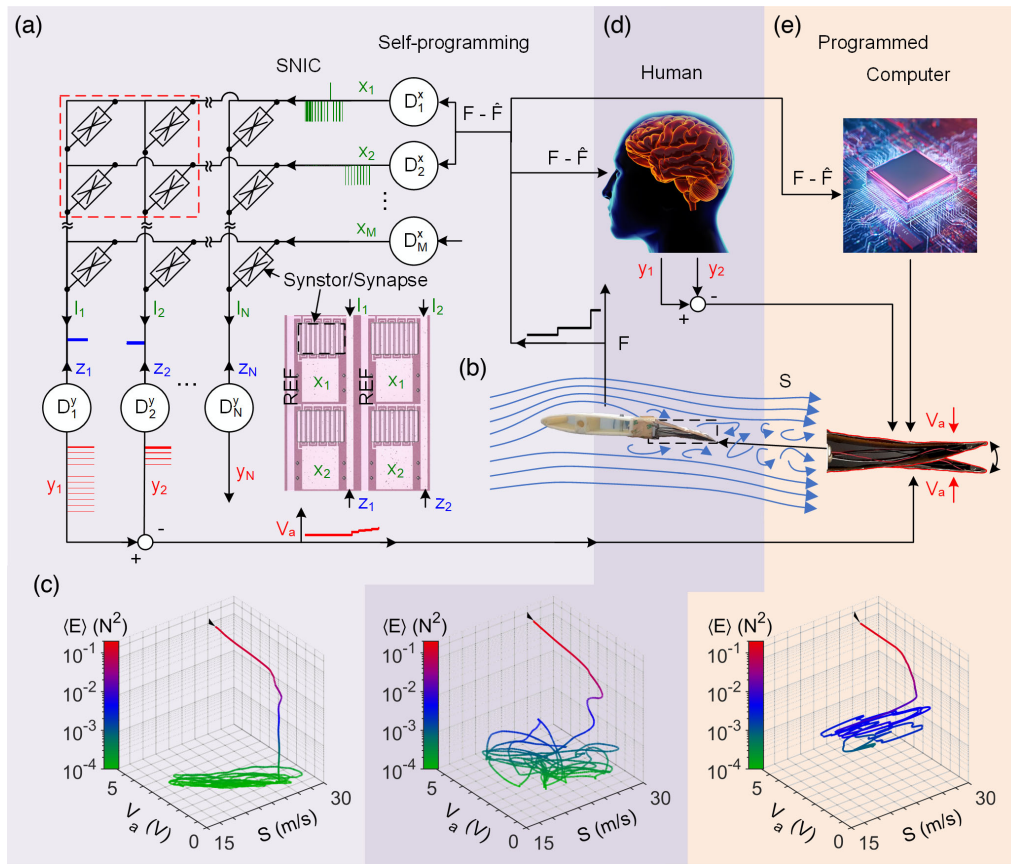


Figure 1. a) A schematic of an SNIC (a neural network) composed of $M \times N$ synstors (synapses) connected with M input (presynaptic) neurons, $D_1^i, D_2^i, \dots, D_M^i$, and N output (postsynaptic) neurons, $D_1^o, D_2^o, \dots, D_N^o$. A 2×2 crossbar synstor circuit is marked by a dashed line and shown in a microscope image in the inset. b) Left, an image of the morphing wing in wind (as illustrated by streamlines) with a randomly varied speed S to generate a lift-force F on the wing. Right, a photo shows that the trailing edge of the morphing wing is deflected upward by decreasing V_a and downward by increasing V_a to modify F toward its target value \hat{F} . c) The average objective function $\langle E \rangle = \frac{1}{2} \langle (F - \hat{F})^2 \rangle$ is plotted versus S and V_a in the typical morphing wing control processes by SNIC (left), human (middle), and computer (right). The black arrows indicate the evolving directions of the system. The illustration of d) a human and e) a computer receiving $F - \hat{F}$ signals and adjusting V_a to modify wing shapes.

M exponentially,^[8] referred to as the “curse of dimensionality.”^[9] Therefore learning algorithms are executed in off-site high-speed computers with high-power consumptions and bulky volumes.^[7,10,11] Despite improved parallelism and computational energy efficiencies, transistor-based computing circuits, such as the Summit supercomputer,^[12] graphics processing units (GPUs),^[7,11,13] tensor processing units (TPUs),^[14] field-programmable gate arrays (FPGAs),^[15] TrueNorth,^[16] and Tianjic^[17] neuromorphic circuits, are still based on the Turing computing model by executing algorithms with data transmissions between physically separated logic and memory transistors. Existing neuromorphic devices such as transistors,^[18,19] memristors,^[20,21] and phase-change memory resistors^[22] execute signal-processing algorithms (without conductance change) and learning algorithms (with conductance change) by applying voltage pulses with different amplitudes. To avoid change of conductance during signal processing, the voltage pulses for signal processing are decreased to smaller magnitudes than the voltage pulses for learning. When the signal-processing algorithm is executed in the circuits, the learning algorithm is interrupted and

vice versa.^[22–25] Therefore, unlike neurobiological networks, the existing neuromorphic circuits cannot execute signal-processing and learning algorithms concurrently and have to be trained or preprogrammed before executing signal-processing algorithms. Due to these limitations, the energy efficiencies for existing electronic circuits to compute learning algorithms are limited to the range of $\approx 10^7 - 10^{13}$ OPS/W (operations per second per watt),^[7,11–17,20,22,24] which are significantly lower than that of the human brain ($\approx 10^{15}$ OPS/W)^[26] and largely prevent artificial intelligent systems from self-programming on site in real time. Without real-time self-programming functionality, artificial intelligent systems fail in unpredictable environments beyond their preprogrammed domains^[27] and lack the brain-like general intelligence in arbitrary environments.^[4]

Recently we developed a synaptic resistor,^[28] abbreviated as synstor hereafter, to emulate a synapse. A synstor processes voltage pulses x by following $I = wx$, Equation (1), and learns from voltage pulses x and z by following $\dot{w} = az \otimes x$, Equation (2). Unlike existing electronic devices such as transistors, memristors, and phase-change memory resistors, the

synstors process and learn from the x and z voltage pulses with the same magnitudes, and the signal-processing ($I = \mathbf{w}x$, Equation (1)) and learning ($\dot{\mathbf{w}} = \alpha z \otimes x$, Equation (2)) algorithms can be executed concurrently in a synstor circuit without interrupting each other. A synstor circuit circumvents the energy consumption on data transmission and memory between logic and memory circuits for executing the signal-processing and learning algorithms separately in conventional computing circuits, and facilitates computations in analog parallel mode by integrating signal processing, memory, and correlative learning functions in each synstor. In this article, we demonstrate a synstor-based self-programming neuromorphic integrated circuit (abbreviated as SNIC hereafter) based on synstors (Figure 1a), which executes the signal-processing ($I = \mathbf{w}x$, Equation (1)) and correlative learning^{1,31} ($\dot{\mathbf{w}} = \alpha z \otimes x$, Equation (2)) algorithms concurrently in parallel analog mode to self-program the synstor conductance matrix, \mathbf{w} , toward its optimal values, $\hat{\mathbf{w}}$, and improve the performance function of a system spontaneously with an energy efficiency ($\approx 3.3 \times 10^{17}$ OPS/W) significantly higher than the energy efficiencies of computing circuits ($\approx 10^7 - 10^{13}$ OPS/W)^{7,11-16,20,22,24} and the human brain ($\approx 10^{15}$ OPS/W).²⁶

2. Experiment and Results

We fabricated a crossbar synstor circuit (Figure S1, Supporting Information), and each synstor²⁸ has a p-type semiconducting carbon nanotube (CNT) channel which forms Schottky contacts with Al input and output electrodes as a resistor. A HfO₂/TiO₂/HfO₂ charge trap heterojunction is sandwiched between the CNT channel and a grounded Al reference electrode as a capacitor (Figure S2, Supporting Information). As shown in Figure 1a, voltage pulses, x_m , on the input electrodes induce currents flowing through the CNT channels to the n^{th} output neuron circuit by following the signal-processing algorithm, $I_n = \sum_m \mathbf{w}_{nm} x_m$ (Equation (1)). When paired negative (positive) voltage pulses, x_m and z_n , are applied on the m^{th} input and n^{th} output electrodes of a synstor simultaneously, the pulses generate a potential difference between the CNT channel and the TiO₂ charge storage layer to drive electrons to hop through the HfO₂ dielectric layer, increasing the negative (positive) charge stored in the charge storage layer and in turn attracting (repelling) the holes in the p-type CNT channel to increase (decrease) the synstor conductance by following the learning algorithm, $\dot{\mathbf{w}}_{nm} = \alpha z_n x_m$ (Equation (2)), with $\alpha > 0$ ($\alpha < 0$). Otherwise, when $z_n = 0$ and/or $x_m = 0$, the x_m or z_n voltage pulse mainly decreases beyond the recessed TiO₂ charge storage layer, and the potential differences between the CNT channel and the TiO₂ charge storage layer are below the threshold values to modify the charge stored in the charge storage layer, so, $\dot{\mathbf{w}}_{nm} = \alpha z_n x_m = 0$ (Figure S3 and S4, Supporting Information). The synstor circuit executes the signal-processing ($I = \mathbf{w}x$, Equation (1)) and correlative learning ($\dot{\mathbf{w}} = \alpha z \otimes x$, Equation (2)) algorithms concurrently without interrupting each other (Experimental Section).

To test SNIC in a practical challenging environment, an SNIC composed of a 2×2 crossbar synstor circuit and two input and two output integrate-and-fire neuron circuits was connected to a morphing wing^{29,30} in a wind tunnel (Figure 1a,b, Experimental

Section). The synstor conductance matrix, \mathbf{w} , had random values before a self-programming process, and the goal was to set \mathbf{w} in the real-time self-programming process to tune the lift-force on the wing, F , toward the target value, $\hat{F} = 0.3$ N, and minimize an objective function $E = \frac{1}{2}(F - \hat{F})^2$. The wind speed, S , changed randomly in the wind tunnel in the range of 17 – 29 m/s to emulate an unpredictable aerodynamic environment which caused the lift-force on the wing, F , to vary randomly in the range of 0 – 1 N. F was also influenced by the shape of the wing, which was controlled by a voltage, V_a , applied on a piezoactuator in the wing (Figure 1b, Experimental Section). F was detected by a sensor in the wind tunnel, and the sensory signals were processed by input neurons to trigger 10 ns-wide input voltage pulses, x , with an amplitude of 1.5 V or –1.75 V (Figure 2a). When $F > \hat{F}$, the pulses were triggered from the first input neuron only; when $F < \hat{F}$, the pulses were triggered from the second input neuron only. The firing rates of the x input pulses were a nonlinear monotonically increasing sigmoid function of $|F - \hat{F}|$ (Experimental Section, Figure S5, Supporting Information). x induced currents I via the synstor circuit by following Equation (1), $I = \mathbf{w}x$, and I flowed into integrate-and-fire output neuron circuits to generate output pulses, y , and feedback pulses, z , at the output electrodes of the circuit. The firing rates of the y and z pulses were nonlinear monotonically increasing functions of $|I|$ (Experimental Section, Figure S6, Supporting Information). The actuation voltage, V_a , was modified by y following $\dot{V}_a = \rho(r_{y1} - r_{y2})$ with r_{y1} and r_{y2} as the firing rates of output pulses from the first and second output neurons, respectively, and $\rho = 8$ mV. V_a was applied on a piezoactuator to modify the wing shape, lift-force F , and objective function E (Figure 1c). A wave of 10 ns-wide 1.5 V (–1.75 V) z pulses was triggered at the first (second) output electrodes at ≈ 575 ms before a wave of y pulses were triggered, and a train of 10 ns-wide –1.75 V (1.5 V) z pulses was triggered at the first (second) output electrodes at ≈ 575 ms after the train of y pulses were triggered. (Figure 2a, Experimental Section, Equation S1, Supporting Information). The time shifts between y and z pulses were mainly set to accommodate the system time delay between the wing and SNIC. The synstor conductance matrix \mathbf{w} was modified by the z and x voltage pulses by following the learning algorithm, $\dot{\mathbf{w}} = \alpha z \otimes x$ (Equation (2)) in the real-time self-programming process to change V_a and minimize the objective function E under the wind conditions with randomly varied speed S (Figure 1c).

To compare the self-programming processes between SNIC and the human brain, 14 human participants without any pre-knowledge about the morphing wing and its control system received $F - \hat{F}$ signals visually and were instructed to minimize the difference between F and \hat{F} and the objective function $E = \frac{1}{2}(F - \hat{F})^2$ by sending output signals y by pressing two keys preset randomly in a keyboard to increase or decrease the actuation voltage V_a on the wing (Figures 1d and 2b, Experimental Section). In the real-time self-programming processes, E was reduced by dynamically modifying V_a under wind with the same randomly varied speed S as that in the SNIC trials (Figure 1c). To compare the self-programming control processes by SNIC and human brains with control processes by a preprogrammed computer, a proportional-integral-derivative (PID) controller implemented on a computer received $F - \hat{F}$ signals and output

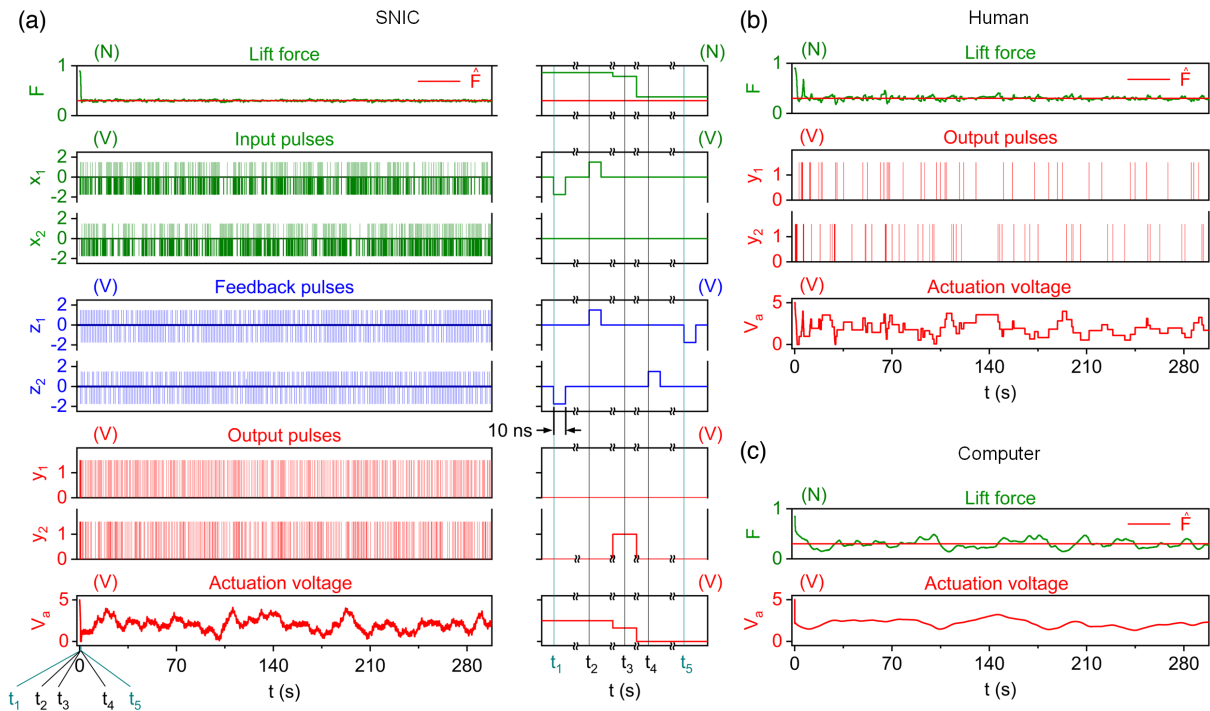


Figure 2. a) The lift-force on the wing F , its target value \hat{F} , input voltage pulses x_n , feedback voltage pulses z_n , output voltage pulses y_n , and V_a in a typical trial by SNIC are plotted versus time t (left) and at moments t_1, t_2, t_3, t_4 , and t_5 equal to 0.498, 0.516, 0.549, 1.080, and 1.099s, respectively (right). F, \hat{F}, y_n , and V_a are plotted versus t in typical trials by b) the human and c) a computer.

V_a signals to modify the wing shape and F (Figure 1e). The PID controller with various gains was tested to control V_a and the shape of the wing experiencing wind with a static speed $S = 28.7 \text{ m/s}$, and the optimal PID gains leading to the minimal average E were identified (Figure S7, Supporting Information). After the PID controller was preprogrammed to the optimal gains, the PID controller modified the shape of the wing while experiencing wind with the same randomly varied speed S as that in the SNIC and human trials, emulating an unpredictable aerodynamic environment beyond the preprogrammed condition (Figure 1e and 2c).

3. Self-Programming Process

In an SNIC or human self-programming process, when $F = \hat{F}$, $E = 0$, $x = 0$, and $\dot{w} = \alpha z \otimes x = 0$ (Equation (2)), w reaches an equilibrium state $\hat{w} = \text{argmin}_w E$. Although w and \hat{w} were not experimentally measured, the relative deviation of effective w from \hat{w} , $\Delta w(t) = [w(t) - \hat{w}] / |w(0) - \hat{w}|$, was extrapolated (Equation S2, Supporting Information). The average E over a fixed moving time window, $\langle E \rangle$, is shown versus t in Figure 3a, versus Δw and the wind speed S in Figure 3b, and versus $|\Delta w|$ in Figure 3c. Although w for SNIC and humans was not preprogrammed, and had random positive or negative initial deviations from \hat{w} , w was modified to \hat{w} , decreasing $\langle E \rangle$ toward equilibrium values, E_{eq} , within $\approx 5.1 \text{ s}$ for SNIC and $\approx 10 \text{ s}$ for humans in their self-programming processes. When the wind speed S changed chaotically, leading to increases in $|\Delta w|$

and E , w was spontaneously modified toward \hat{w} under the varied S , decreasing $\langle E \rangle$ monotonically versus t in the self-programming processes (Figure 3). The dynamic change of E in the self-programming process can be expressed as (Supporting Information, Theorem 1)

$$\langle \dot{E} \rangle = -\beta \langle E \rangle + \delta E \quad (3)$$

where $\beta \geq 0$, and δE is related to the environmental influence and nonlinear term of E . In the self-programming processes for SNIC and humans, $\delta E < \beta \langle E \rangle$ and $\langle \dot{E} \rangle = -\beta \langle E \rangle + \delta E < 0$. Thus $\langle E \rangle$ represented a Lyapunov function and was asymptotically decreased toward its dynamic equilibrium value E_{eq} , leading $\langle w \rangle$ to be modified toward $\langle \hat{w} \rangle$ in the self-programming process; when $\delta E = \beta \langle E \rangle$, $\langle \dot{E} \rangle = 0$, and E reached its dynamic equilibrium value $E_{\text{eq}} = \delta E / \beta$ under $\langle w \rangle = \langle \hat{w} \rangle$. The solution of Equation (3) gives $\langle E \rangle = E_{\text{eq}} + (\langle E \rangle - E_{\text{eq}}) e^{-\beta t} + \delta E * e^{-\beta t}$, where $\delta E * e^{-\beta t}$ represents the convolution between δE and $e^{-\beta t}$. When $\beta t \gg 1$, $\langle E \rangle \approx E_{\text{eq}}$; thus, β represents the self-programming speed to modify $\langle E \rangle$ toward E_{eq} and w toward \hat{w} (Equation (S5), Supporting Information). With its gains preprogrammed to their optimal values under a static wind speed, the PID controller decreased E initially, but the gains were not modified toward their optimal values dynamically under the varied S , leading to E significantly larger than those of the SNIC and humans (Figure 1c).

In a self-programming process, β in Equation (3) represents the speed to reduce $\langle E \rangle$ toward E_{eq} and modify w toward \hat{w} (Equation S5, Supporting Information). As shown in

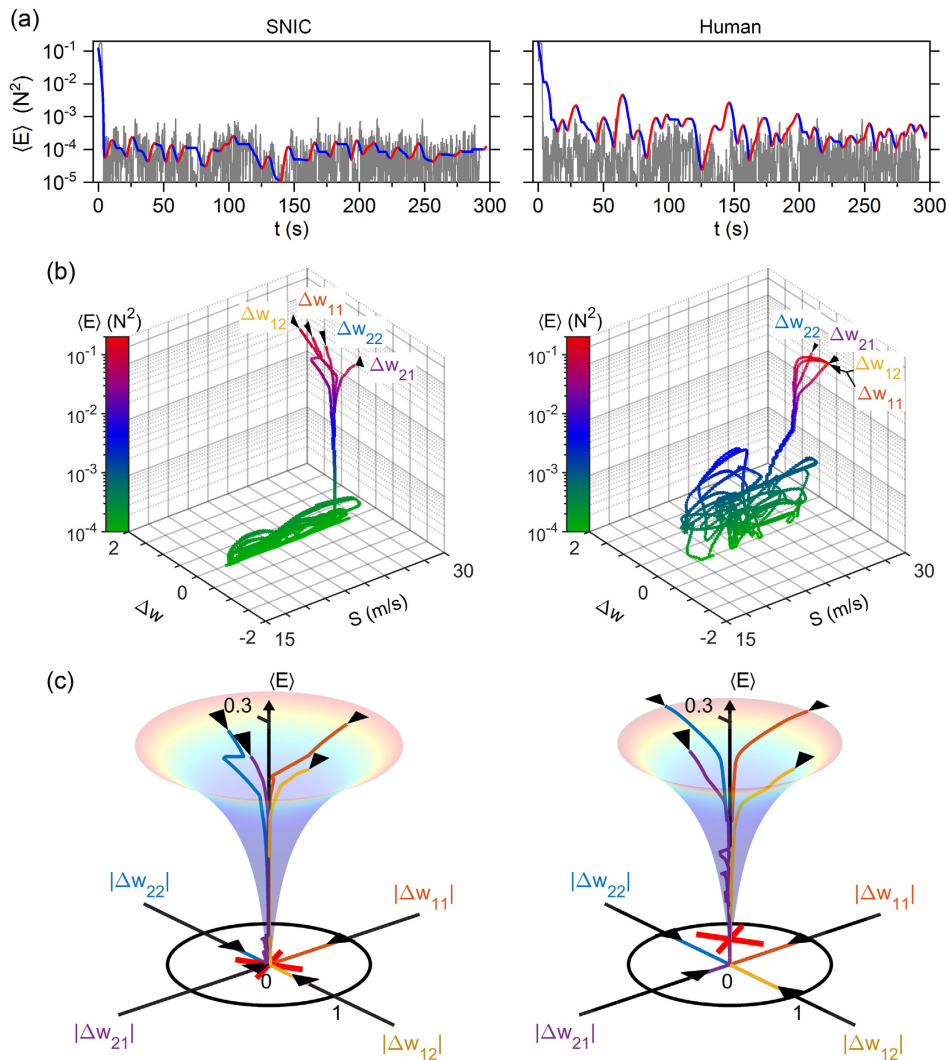


Figure 3. In the real-time self-programming processes of a SNIC (left) and human (right), a) objective functions E are shown versus t in gray color, and average objective functions, $\langle E \rangle$, are displayed versus time t in blue when $d\langle E \rangle/dt < 0$ and in red when $d\langle E \rangle/dt > 0$. b) $\langle E \rangle$ is plotted versus the relative deviations of the device conductances from the optimal conductances, Δw_{11} , Δw_{12} , Δw_{21} , and Δw_{22} , and wind speed, S . c) E is plotted at logarithmic scale in 3D plots versus $|\Delta w_{11}|$, $|\Delta w_{12}|$, $|\Delta w_{21}|$, and $|\Delta w_{22}|$ at different azimuthal angles in linear scale. The evolving directions of Δw are indicated by arrows at the base planes of the 3D plots. $\langle E \rangle$ versus $\langle \Delta w \rangle$ is best fitted by $\langle E \rangle = \frac{1}{2}g^{E/w}\langle \Delta w \rangle^2 + E_{eq}$ and shown as 3D surfaces, with E_{eq} marked by red crosshairs.

Figure 4a, β increases with increasing average change rate of w at the initial stage of the self-programming process, $|\dot{w}|^e$, which can be increased by increasing the firing rates of x , y , and z pulses and decreasing the capacitances and leakage currents in the input and output neurons in SNIC (Figure S5 and S6, Supporting Information). The equilibrium objective function E_{eq} defined in Equation (3) represents the accuracy to modify w toward \hat{w} (Equation S3, Supporting Information). As shown in Figure 4b, E_{eq} reached its minimal values (data points 2 and 5) when the average change rate of w was near the equilibrium stage of the self-programming process, $\langle |\dot{w}| \rangle^e = 0.62/s$ for SNIC and $\langle |\dot{w}| \rangle^e = 0.32/s$ for humans. When $\langle |\dot{w}| \rangle^e$ is decreased from its optimal values (data points 1 and 4), the β value is decreased, and $E_{eq} = \delta E/\beta$, leading to the increase in E_{eq} as $E_{eq} = \delta E/\beta$.

When β is decreased to zero in a control experiment without z pulses or self-programming, $\langle E \rangle = \delta E > 0$ (Equation (3)) and E_{eq} reaches the maximal value in Figure 4b. When $\langle |\dot{w}| \rangle^e$ is increased from its optimal values (data points 3 and 6), w is modified at a high rate, and w overshoots with respect to \hat{w} , leading to the fluctuation of $|\Delta w|$ and E and the increase in E_{eq} (Figure 4c.). In the self-programming processes, when w is close to \hat{w} , the pulse firing rates are decreased by the leakage current in the integrate-and-fire neuron circuits to avoid the overshoot of w with respect to \hat{w} and reduce E_{eq} ; when w deviates from \hat{w} , the pulse firing rates are increased as a nonlinear function of input signals to the neuron circuits to increase β and decrease E at high speed (Figure S5 and S6, Supporting Information). By optimizing the neuron circuits in SNIC, the average self-programming

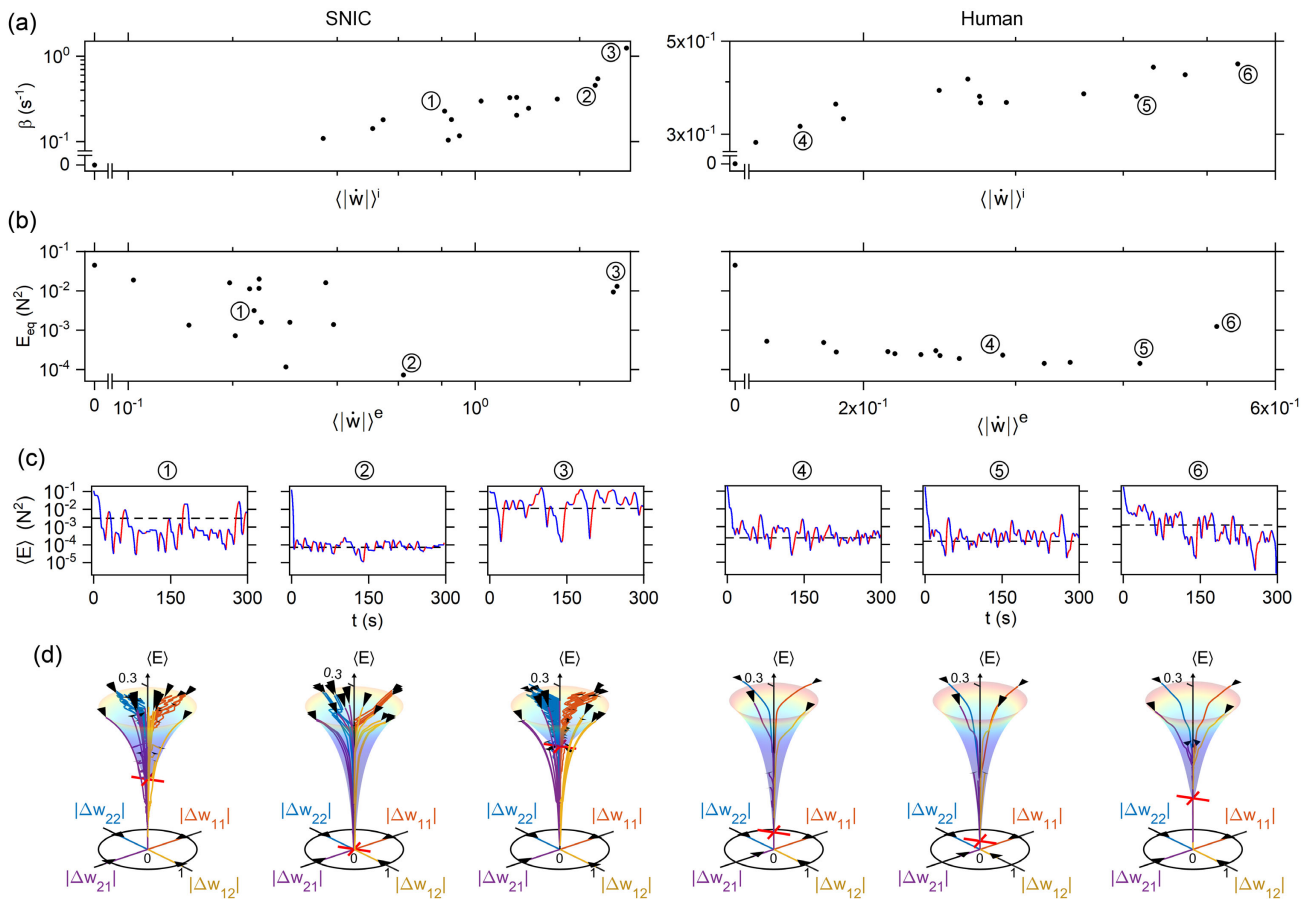


Figure 4. a) Learning speeds β and b) equilibrium objective functions E_{eq} are plotted versus the modification rates of their relative conductance matrix at the initial stages, $\langle |\dot{w}| \rangle^i$, and equilibrium stages, $\langle |\dot{w}| \rangle^e$, of the self-programming processes of (left) SNICs with different neuron circuits and (right) different humans. In the self-programming processes of (left) SNICs and (right) humans with different $\langle |\dot{w}| \rangle^i$ and $\langle |\dot{w}| \rangle^e$ (as marked by 1, 2, 3, 4, 5, and 6), c) average objective functions $\langle E \rangle$ are plotted versus time t in blue when $d\langle E \rangle/dt < 0$, and in red when $d\langle E \rangle/dt > 0$, and d) $\langle E \rangle$ is plotted at logarithmic scale in 3D plots versus $|\Delta w_{11}|$, $|\Delta w_{12}|$, $|\Delta w_{21}|$, and $|\Delta w_{22}|$, which are shown versus time at the base planes of the 3D plots, at different azimuthal angles by blue, red–orange, yellow, and violet lines, respectively. The arrows at the starting points of the lines indicate the evolving directions. $\langle E \rangle$ is best fitted as a function Δw by $\langle E \rangle = \frac{1}{2} g^{E/w} \langle \Delta w \rangle^2 + E_{eq}$ and shown in 3D surfaces, with $\langle E_{eq} \rangle$ marked by red crosshairs in (d) and dashed lines in (c).

speed β ($0.46s^{-1}$) and E_{eq} ($7.2 \times 10^{-5}N^2$) in the self-programming process of SNIC are superior to β ($0.37s^{-1}$) and E_{eq} ($3.4 \times 10^{-4}N^2$) of the humans (Figure 3 and 4).

4. Conclusion

In summary, we demonstrated an SNIC based on synstors to emulate a neurobiological network based on synapses to execute the signal-processing ($I = wx$, Equation (1)) and correlative learning ($\dot{w} = \alpha x \otimes x$, Equation 2) algorithms concurrently in parallel analog mode. Unlike a programmable computer, the synstor conductance matrix w does not have to be preprogrammed and can be spontaneously modified toward the optimal matrix \hat{w} , minimizing the objective function E in a self-programming process in complex and unpredictable environments. An SNIC controlled a morphing wing, modified its lift force F toward a targeted value \hat{F} , and minimized the objective function $E = \frac{1}{2}(F - \hat{F})^2$ toward its equilibrium value E_{eq} in a wind with

randomly varied speeds. The correlative learning algorithm executed in the synstor circuit can be extended to various learning algorithms including supervised, unsupervised, and reinforcement learning algorithms, leading to the optimization of predefined or self-organized objective functions in intelligent systems.^[3,31] Unlike artificial intelligent systems based on computers which have to be preprogrammed for specific tasks, SNIC does not have to be preprogrammed and can “self-program” heuristically by executing the correlative learning algorithm in real time in arbitrary environments for general intelligence. In comparison with humans and a preprogrammed computer, an SNIC demonstrated self-programming speeds and E_{eq} superior to those of the humans and computer. SNIC circumvents the energy consumptions on data transmissions in conventional computing circuits, facilitating a computing energy efficiency of $\approx 3.3 \times 10^{17}$ OPS/W (Experimental Section, Equation (6), Figure 5) significantly higher than the energy efficiencies of computing circuits ($\approx 10^7 - 10^{13}$ OPS/W)^[7,11–16,20,22,24] and the human brain ($\approx 10^{15}$ OPS/W).^[26] The speed to compute parallel

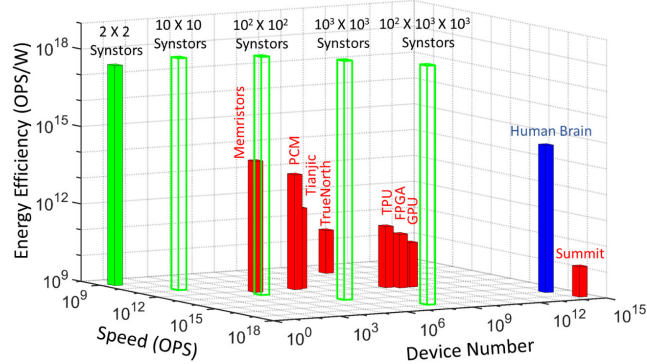


Figure 5. A 3D plot displays the computing energy efficiencies, speeds, and device numbers in a 2×2 synstor circuit in this work (green), projected 10×10 , $10^2 \times 10^2$, $10^3 \times 10^3$, and $10^2 \times 10^3 \times 10^3$ synstor circuits (green), the human brain (blue), Summit supercomputer, Volta V100 GPUs from Nvidia, Stratrix 10 FPGA from Intel, TPUs from Google, TrueNorth neuromorphic circuit from IBM, Tianjic neuromorphic circuits from Tsinghua University, phase-change memory circuit from IBM (signal processing only, learning excluded), and memristor circuits from UMass/HP (signal processing only, learning excluded).

signal-processing and learning algorithms in a SNIC increases linearly with increasing circuit scale (Experimental Section, Equation (4), Figure 5), the power consumption of an SNIC increases with increasing circuit scale (Experimental Section, Equation (5)), and the computing energy efficiency of an SNIC approximately does not change with increasing circuit scale (Experimental Section, Equation (6), Figure 5). A circuit of 10^6 synstors will have a speed (6×10^{14} OPS) comparable with the speeds ($\approx 10^{12} - 10^{14}$ OPS) of TPU, GPU, and FPGA circuits with $\sim 10^9 - 10^{11}$ transistors and consume much less power ($\approx 30 \mu\text{W}$) than those of the transistor-based circuits ($\approx 40 \text{W}$).^[13–15] A circuit of 10^9 synstors will have a speed (6×10^{16} OPS) comparable with the speeds of the human brain ($\approx 10^{16}$ OPS) and the Summit supercomputer ($\approx 10^{17}$ OPS) and consumes a power ($\approx 40 \text{mW}$) much less than those of the human brain ($\approx 30 \text{W}$ with $\approx 10^{14}$ synapses)^[26] and Summit supercomputer ($\approx 10^7 \text{W}$ with $\approx 10^{14}$ transistors).^[12] There is “plenty of room at the bottom” to scale up synstor circuits with high speed, low power consumption, high energy efficiency, and small circuit scale/volume for a new computing platform that can self-program in real time in arbitrary and unpredictable environments for artificial general intelligence.

5. Experimental Section

Learning Algorithm in a Synstor Circuit: In the self-programming process of a synstor, the feedback pulses, \mathbf{z} , follow $\mathbf{z} = \mathbf{y} * \bar{\theta}$ (Equation S1, Supporting Information), where

$$\bar{\theta}(t) = \begin{cases} \theta(t) & \text{when } -\tau_- < t < 0 \\ -\theta(t) & \text{when } \tau_+ > t > 0 \\ 0 & \text{when } t = 0 \text{ or } t \geq \tau_+ \text{ or } t \leq -\tau_- \end{cases}, \text{ the time constants } \tau_{\pm} > 0$$

and $\tau_- > 0$, the function $\theta(t) > 0$. For the average $\bar{\theta}$ over learning period T , $\langle \bar{\theta} \rangle = 0$, and the average \mathbf{z} over learning period T , $\langle \mathbf{z} \rangle = 0$, and $\bar{\mathbf{z}} = \mathbf{z} - \bar{\mathbf{z}} = \mathbf{z}$. To generate feedback pulses with $\mathbf{z}_n = \mathbf{y}_n * \bar{\theta}$, a train of positive (negative) feedback pulses with a pulse firing rate proportional

to $\theta(t - t_n)$ within the time window $t_n - \tau_- < t < t_n$ at the n^{th} (complementary) output electrode and a train of negative (positive) feedback pulses with a pulse firing rate proportional to $\theta(t - t_n)$ within the time window $t_n < t < t_n + \tau_+$ were triggered at the n^{th} output electrode.

Synstor Circuit Fabrication: The synstor circuit was fabricated by the process reported previously.^[28] Si wafers with a 100 nm-thick SiO_2 layer were diced into $3 \text{ cm} \times 3 \text{ cm}$ square chips. A $10 \mu\text{m}$ -long and 50 nm-thick Al reference electrode (Figure S1a, Supporting Information) was deposited by electron beam (e-beam) evaporation (CHA Industries, CHA Mark 40) and patterned by photolithography and wet chemical etching with tetramethylammonium hydroxide (TMAH)-based photoresist developer (AZ 300 MIF Developer). A 22 nm-thick HfO_2 barrier layer and a 2.5 nm-thick TiO_2 charge storage layer (Figure S1b, Supporting Information) were deposited by atomic layer deposition (Cambridge NanoTech, Fiji Thermal and Plasma Atomic Layer Deposition (ALD)). The TiO_2 film was patterned (Figure S1c, Supporting Information) by photolithography and CF_4/O_2 (5:1 pressure) reactive ion etching (Technics RIE) to form a $10 \mu\text{m}$ -long pattern aligned to the Al reference electrode. A 6.5 nm-thick HfO_2 barrier layer (Figure S1d, Supporting Information) was deposited by ALD, encapsulating the patterned TiO_2 charge storage layer. The chip surface was coated by an adhesion monolayer of poly (L-lysine) (PLL). A randomly oriented network of semiconducting single-walled CNTs was deposited by immersion coating (Figure S1e, Supporting Information) in an aqueous 99.9% pure semiconducting single-walled CNT aqueous solution (Nanointegris, IsoNanotubes-S99.9%). Residual surfactant was removed from the surface by immersion in isopropanol (IPA) for 1 h, rinsed with IPA, and dried by nitrogen blow dry. CNTs were doped to p-type by adsorbing O_2 acceptors from atmosphere. A 50 nm Al film (Figure S1f, Supporting Information) was deposited by e-beam evaporation and patterned by the same process used for the Al reference electrode to form input and output electrodes. The CNTs were capped by a Parylene-C (PLC) polymer passivation layer deposited (Figure S1g, Supporting Information) by thermal evaporation (Specialty Coating Systems, 2010 Parylene Vacuum Deposition System). The CNT network and PLC layer were patterned by photolithography with SU-8 photoresist and O_2 RIE to form a $20 \mu\text{m}$ -long CNT channel (Figure S1g, Supporting Information). The SU-8 was an etch mask for the CNTs and PLC during O_2 RIE and encapsulated the CNTs, and PLC prevented ambient doping of CNTs by atmosphere.

SNIC Testing: Voltage pulses, x and z , were generated by function generators (Agilent 33250 A) with voltage amplitudes ranging between -2V and 2V , a duration ranging between 10 ns and 5 ms, and a frequency ranging between 50 MHz and 100 Hz and were applied to the input and output electrodes of synstor circuits. The x and z voltage pulses were gated by switches (Maxim, MAX383), which were activated by a digital voltage module (National Instruments, 9403 E Digital Input/Output). The synchronized input and output pulses from a single generator prevented phase shifts between the pulses. Output voltage pulses, y , were also read by the digital voltage module. The input, output, feedback, actuation, and reference voltage signals were measured by an analog module (National Instruments, 9205 Analog Input). To extrapolate the synstor conductance w , input pulses $x = -1.75 \text{V}$ were applied to a synstor, and the output current from the synstor was measured by an operational amplifier (Microchip Technologies, MPC6022).

Morphing Wing: The morphing wing was a wing section with a 12-inch chord and a morphing trailing edge, which used a macrofiber composite (MFC) piezoelectric actuator and a flexure box mechanism to modify the camber of the trailing edge.^[30,32] The actuator had a 3D-printed elastomeric honeycomb skin for tailored stiffness, and the piezoelectric mechanism allowed for fast response time. The morphing wing had applications in stall recovery during wind gusts, optimizing the lift distribution to increase aerodynamic efficiency and reducing turbulence. The design was scalable to multiple piezoelectric actuators along the spanwise edge (spanwise morphing trail edge) to achieve continuous wing shape change, but the hysteresis of the piezoelectric actuator increased the difficulty of controlling the wing, which was adapted by the real-time self-programming functionality of the synstor circuit. The output voltage signals from the synstor circuit, y , triggered an analog actuation voltage,

V_a , from an analog voltage module (National Instruments, NI-9264). V_a was amplified by a high-voltage driver (Avid LLC, AVID-EHV-MFC.B2) to a range from -0.5 to 1.5 kV to control the piezoelectric actuators and modify the wing shape.

Wind Tunnel: The morphing wing was tested in a laminar flow and an open-circuit wind tunnel (Aerolab) with a 24×24 in. test section. A fan in the wind tunnel was driven by a high-voltage driver (ABB, ACS550-01-046 A-2 AC) to randomly change wind speed, S , in the range between 17.3 and 28.7 m s $^{-1}$. S was measured by a pitot tube and air velocity transducer (TSI, 8455). The lift-force on the wing, F , was measured using a force–torque multiaxis load cell (JR3, 30E12A-4-I40-EF 40N3.1S) with a measurement range of ± 80 N and resolution of 0.01 N, attached to a morphing wing mounting shaft. The sensory signals of S and F were read by a voltage I/O device (National Instruments, PCIe 6353 Multifunction I/O Device).

Human Controllers: The $F - \hat{F}$ values were dynamically displayed to the participants, who pressed two keys on a keyboard to increase or decrease the actuation voltage, V_a . The change rate of V_a was determined by the key-strokes, $\dot{V}_a = \rho_h(a_{k1} - a_{k2})$, where $a_{k1} = \begin{cases} 1 & \text{when key 1 is pressed} \\ 0 & \text{when key 1 is not pressed} \end{cases}$ and $a_{k2} = \begin{cases} 1 & \text{when key 2 is pressed} \\ 0 & \text{when key 2 is not pressed} \end{cases}$, and ρ_h was randomly set to 31 mV/ms or -31 mV/ms before each experiment started. V_a was modified by the humans to minimize the objective function $E = \frac{1}{2}(F - \hat{F})^2$ under the randomly changed wind speed S .

PID Controller: The lift-force error, $e_F = F - \hat{F}$, was sent to the PID controller to induce the actuation voltage V_a following the PID control algorithm, $\dot{V}_a = K_p e_F + K_i \int_0^t e_F(t') dt' + K_d \dot{e}_F$, where K_p denotes the proportional gain, K_i denotes the integral gain, and K_d denotes the derivative gain. After the gains of the PID controller were set to various combinations of $K_p = 10^{-4}, 10^{-3},$ or 10^{-2} V/N \cdot s; $K_i = 10^{-3}, 10^{-2},$ or 10^{-1} V/N \cdot s 2 ; and $K_d = 10^{-2}, 10^{-1},$ or 1 V/N, the PID controller modified V_a and the shape of the wing while experiencing wind with static speed $S = 28.7$ m/s. The average objective function $\langle E \rangle = \frac{1}{2}(F - \hat{F})^2$ during the control processes is shown as a function of $K_p, K_i,$ and K_d in Figure S7, Supporting Information, and $\langle E \rangle$ approaches its minimal value under the optimal gains with $K_p = 10^{-5}$ V/N \cdot s, $K_i = 10^{-4}$ V/N \cdot s 2 , and $K_d = 10^{-3}$ V/N. After the PID controller was preprogrammed to the optimal gains, the PID controller modified V_a and the shape of the wing experienced the randomly changed wind speeds S , emulating an unpredictable aerodynamic environment beyond the static wind speed.

SNIC Computing Speed and Energy Efficiency: In comparison with computers, the equivalent computing operations in an $M \times N$ synstor circuit were approximately equal to $3MN$ to implement the signal-processing algorithm ($I = \mathbf{w}\mathbf{x}$, Equation (1)), $2MN$ for multiplications between \mathbf{w} and \mathbf{x} (MN for accumulations), and $3MN$ to implement the learning algorithm ($\dot{\mathbf{w}} = \alpha \mathbf{z} \otimes \mathbf{x}$, Equation (2)), $2MN$ outer products between $\alpha, \mathbf{x},$ and \mathbf{z} , MN for \mathbf{w} modifications. The speed for the synstor circuit to implement $6MN$ equivalent computing operations for parallel signal processing and learning is

$$V_c = 6MNf_c \quad (4)$$

where f_c denotes the operation frequency of the circuit. When voltage pulses are applied on its input or output electrode of an $M \times N$ synstor circuit connected with N output integrate-and-fire neuron circuits, the average power consumption in an $M \times N$ synstor circuit is

$$P_c \approx MN\langle \mathbf{w} \rangle V_a^2 D_p + NE_p \langle r_y \rangle \quad (5)$$

where $\langle \mathbf{w} \rangle$ denotes the average conductance of the synstors, V_a denotes the magnitude of pulses, D_p denotes the average duty-cycle of the pulses, E_p denotes the average energy consumption to trigger a pulse from integrate-and-fire neuron circuits, and $\langle r_y \rangle$ denotes the average firing rates of pulses from output neuron circuits. The input neurons are part of the circuit of last layer, so their energy consumption is not included in the SNIC energy consumption of the current layer. The computing energy efficiency

of a synstor circuit is equal to its computing speed V_c divided by its power consumption P_c

$$C_E = 6f_c / (\langle \mathbf{w} \rangle V_a^2 D_p + E_p \langle r_y \rangle / M) \quad (6)$$

The computing energy efficiency of SNIC operated with operation frequency $f_c = 100$ MHz, average synstor conductances $\langle \mathbf{w} \rangle = 10$ nS, $V_a = 1.75$ V, $D_p = 0.06$, $E_p = 10$ pJ, $r_{in} = 300$ kHz, and $\langle r_y \rangle = 160$ Hz and was approximately equal to 3.3×10^{17} OPS/W.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The authors acknowledge the support of this work by the Air Force Office of Scientific Research (AFOSR) under the programs “Avian-Inspired Multifunctional Morphing Vehicle (FA9550-16-1-0087)” and “Brain Inspired Networks for Multifunctional Intelligent Systems in Aerial Vehicles (FA9550-19-0213).”

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

artificial general intelligence, neuromorphic circuits, self-programming, synaptic resistors

Received: January 27, 2021

Revised: March 9, 2021

Published online: May 18, 2021

- [1] D. O. Hebb, *The organization of behavior: a neuropsychological theory*, J. Wiley/Chapman & Hall, New York, USA 1949.
- [2] Y. Dan, M.-M. Poo, *Neuron* **2004**, *44*, 23.
- [3] Z. Chen, S. Haykin, J. J. Eggermont, S. Becker, *Correlative Learning: A basis for Brain and Adaptive Systems*, Vol. 49, John Wiley & Sons, New York 2008.
- [4] T.-J. Huang, *Int. J. Autom. Comput.* **2017**, *14*, 520.
- [5] A. Turing, *Mind* **1950**, *59*, 433.
- [6] J. Schmidhuber, *Neural Networks* **2015**, *61*, 85.
- [7] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, *Nature* **2016**, *529*, 484.
- [8] R. P. Feynman, *Feynman Lectures on Computation*, CRC Press, Boca Raton, FL, USA 2018.
- [9] R. Bellman, *Science (Washington, DC, U.S.)* **1966**, *153*, 34.
- [10] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap,

- K. Simonyan, D. Hassabis, *Science (Washington, DC, U.S.)* **2018**, 362, 1140.
- [11] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, J. Chen, J. Chen, Z. Chen, M. Chrzanowski, A. Coates, G. Diamos, K. Ding, N. Du, E. Elsen, J. Engel, W. Fang, L. Fan, C. Fougner, L. Gao, C. Gong, A. Hannun, T. Han, L. Johannes, B. Jiang, C. Ju, et al., *presented at International conference on machine learning*, New York, NY, USA, **2016**.
- [12] Oak Ridge National Laboratory, *America's newest and smartest super-computer*, <https://www.olcf.ornl.gov/summit/> (accessed: May 2018).
- [13] Nvidia, Nvidia Tesla V100 GPU architecture, <http://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf> (accessed: August 2017).
- [14] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P.-L. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, et al., *presented at Proc. of the 44th Annual International Symposium on Computer Architecture*, New York, USA, **2017**.
- [15] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. T. Liew, K. Srivatsan, D. Moss, S. Subhaschandra, *presented at Proc. of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, **2017**.
- [16] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, D. S. Modha, *Science (Washington, DC, U.S.)* **2014**, 345, 668.
- [17] J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He, *Nature* **2019**, 572, 106.
- [18] K. Kim, C.-L. Chen, Q. Truong, A. M. Shen, Y. Chen, *Adv. Mater* **2013**, 25, 1693.
- [19] C. Diorio, P. Hasler, A. Minch, C. A. Mead, *IEEE Trans. Electron Devices* **1996**, 43, 1972.
- [20] C. Li, M. Hu, Y. Li, H. Jiang, N. Ge, E. Montgomery, J. Zhang, W. Song, N. Dávila, C. E. Graves, Z. Li, J. P. Strachan, P. Lin, Z. Wang, M. Barnell, Q. Wu, R. S. Williams, J. J. Yang, Q. Xia, *Nat. Electron.* **2018**, 1, 52.
- [21] T. Berzina, A. Smerieri, M. Bernabò, A. Pucci, G. Ruggeri, V. Erokhin, M. P. Fontana, *J. Appl. Phys* **2009**, 105, 124515.
- [22] S. Ambrogio, P. Narayanan, H. Tsai, R. M. Shelby, I. Boybat, C. Di Nolfo, S. Sidler, M. Giordano, M. Bodini, N. C. P. Farinha, B. Killeen, C. Cheng, Y. Jaoudi, G. W. Burr, *Nature* **2018**, 558, 60.
- [23] Q. Lai, L. Zhang, Z. Li, W. F. Stickle, R. S. Williams, Y. Chen, *Adv. Mater.* **2010**, 22, 2448.
- [24] M. Prezioso, F. Merrikkh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, D. B. Strukov, *Nature* **2015**, 521, 61.
- [25] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, V. Srikumar, *ACM SIGARCH Comput Architect News* **2016**, 44, 14.
- [26] R. Kurzweil, *The Singularity Is Near: When Humans Transcend Biology*, Penguin, Viking, New York **2005**.
- [27] H. Surden, M.-A. Williams, *Cardozo L. Rev.* **2016**, 38, 121.
- [28] C. D. Danesh, C. M. Shaffer, D. Nathan, R. Shenoy, A. Tudor, M. Tadayon, Y. Lin, Y. Chen, *Adv. Mater.* **2019**, 31, 1808032.
- [29] S. Barbarino, O. Bilgen, R. M. Ajaj, M. I. Friswell, D. J. Inman, *J. Intell. Mater. Syst. Struct* **2011**, 22, 823.
- [30] L. L. Gamble, A. M. Pankonien, D. J. Inman, *AIAA J.* **2017**, 1.
- [31] Y. Chen, *Advanced Intelligent Systems* **2020**, 2000219.
- [32] A. Pankonien, D. J. Inman, in *Proc. SPIE 8688, Active and Passive Smart Structures and Integrated Systems* **2013**, 868815.