Chapter 15
Implementation of Associative Learning Using Cognitive-Inspired Robotic System

Tianze Liu, Noah Zins, Yan Zhang, and Hongyu An

15.1 Introduction

Deep neural networks (DNNs) have revolutionized artificial intelligence (AI), achieving remarkable success across various cognitive tasks through extensive training on large datasets [1]. These networks refine their predictions by minimizing errors via backpropagation, necessitating vast computational resources and substantial power consumption [2–6]. However, this dependence on large datasets and high power consumption presents significant challenges for autonomous operations, especially in environments with stringent Size, Weight, and Power (SWaP) constraints, such as planetary robotics [1, 2]. Planetary rovers, for example, must operate autonomously with limited energy and without human intervention [2].

To overcome these limitations, our research focuses on enhancing the autonomous capabilities of intelligent robots by emulating the associative learning processes observed in animals, mainly through the use of neuromorphic systems. However, applying neuromorphic systems to robotics presents its own set of challenges. These include the design of efficient learning algorithms, integrating neuromorphic hardware with robotic systems, and adapting animal learning behaviors to the robot's environment. Inspired by the brain's structure and function, Neuromorphic systems offer a more energy-efficient approach to artificial intelligence. Associative learning, a natural self-learning mechanism in animals, enables them to adapt to their environment by forming connections between

T. Liu · N. Zins · H. An (✉)
Department of Electrical and Computer Engineering, Michigan Technological University,
Houghton, MI, USA
e-mail: tianzel@mtu.edu; nwzins@mtu.edu; hongyua@mtu.edu

Y. Zhang (✉)
Department of Biological Sciences, Michigan Technological University, Houghton, MI, USA
e-mail: yzhang49@mtu.edu

A. Iranmanesh, H. Sayadi (eds.), *AI-Enabled Electronic Circuit and System Design*,
https://doi.org/10.1007/978-3-031-71436-8_15

concurrent events. This capability can be adapted for robots, allowing them to link information and experiences and adjust their behavior in dynamic environments like Mars.

Our study is structured into two main parts. The first part explores the theoretical aspects of associative learning, providing a comprehensive understanding of this natural self-learning mechanism in animals. The second part applies these principles to robotics, employing a large-scale associative learning system. This system, powered by Intel's Loihi neuromorphic chip, offers a more energy-efficient approach to artificial intelligence. It allows a mobile robot to learn associations between light and vibration signals as conditional and unconditional stimuli, mimicking the fear conditioning process observed in animals. We aim to demonstrate a significant advancement in autonomous robotics by implementing this system in a real-world scenario.

This research marks a significant milestone as the first real-world application of a cognitive-inspired associative learning paradigm using a mobile robot. By integrating the Loihi chip, we enhance signal processing speed and energy efficiency and herald a new era for autonomous robotics. In our experiment, the robot learns to associate light stimuli with vibration stimuli, mimicking fear conditioning processes observed in animals. The synaptic weights are adjusted using Hebbian learning principles, enabling the robot to adapt its responses based on the learned associations.

The key contributions of this chapter include the following:

1. Successfully demonstrate a novel associative learning paradigm in a real-world setting with a mobile robot.
2. Implementing a neural assembly designed for decision-making and applying it to the associative learning process within the robot.

Our findings demonstrate the vast potential of neuromorphic systems to replicate animal learning behaviors in robots and mark a significant leap forward in the field of autonomous robotics. This research paves the way for robots to adapt and learn from their environment, opening up new possibilities for their applications.

15.2 Background

By examining and mimicking the biological processes that facilitate associative learning in animals, we have developed an innovative self-learning paradigm for neuromorphic systems. This section explores recent advancements in neuromorphic computing and hardware and analyzes the mechanisms that support associative learning, from cellular functions to behavioral responses.

15.2.1 Neuromorphic Computing

Our research has pioneered a transformative advancement in neuromorphic systems, introducing a self-learning paradigm that emulates the biological processes underlying animal associative learning. This section delves into our significant strides in neuromorphic computing and hardware, dissecting the mechanisms that enable associative learning from cellular activities to behavioral responses. By doing so, we offer profound insights into the future potential of neuromorphic systems.

Neuromorphic systems, drawing inspiration from the astounding efficiency of the human brain, can potentially revolutionize artificial intelligence (AI) [7–12]. The human brain, a testament to nature's ingenuity, can execute complex tasks with a mere 20 watts of power [13]. This efficiency contrasts with artificial neural networks (ANNs), which demand extensive training and large datasets. The brain's adaptability, a product of its parallel processing capabilities, high connectivity, flexible network topology, integrated data memory and computation, and spike-based information representation, underscores the transformative potential of neuromorphic systems in AI and computing, instilling hope and inspiration for the future.

The human brain comprises billions of neurons and trillions of synapses, creating a highly interconnected three-dimensional network. Each neuron can communicate with over ten thousand other neurons simultaneously. Neurons function as signal processors, integrating incoming spiking signals and transmitting new spike sequences to other neurons through synapses. The strength of these signals is modulated by synaptic connection strength, which can be dynamically adjusted through synaptic plasticity [13–15]. When presynaptic and postsynaptic neurons fire together, their connection is strengthened, a principle known as Hebbian learning [16–19].

Neuromorphic systems, with their groundbreaking architecture, aim to overcome the inefficiencies of the von Neumann architecture, which separates computing and memory units, necessitating frequent data transfers between them. In contrast, neuromorphic systems use sparse, event-driven computation, activating only the required computing resources in response to specific events. By emulating brain architecture, including neurons, synapses, and learning methods, neuromorphic computing offers a more efficient future for computers and AI [20].

These systems harness specialized chips with artificial neurons to operate spiking neural networks (SNNs), encoding information through spike sequences akin to those in biological nervous systems. Intel's Loihi chips serve as a prime illustration of this technology [21, 22]. Unlike traditional GPUs and CPUs based on the von Neumann architecture, Loihi chips are purpose-built for neuromorphic computing and asynchronous SNNs. Unveiled in 2017, the first-generation Loihi-1 chips boast 130,000 electronic neurons and 130 million synapses across 128 neuromorphic cores, all housed within a compact 60 mm^2 chip size made possible by Intel's advanced 14 nm process [21, 22]. They employ digital leaky integrate-and-fire neurons with mesh-configured communication and configurable synapses that

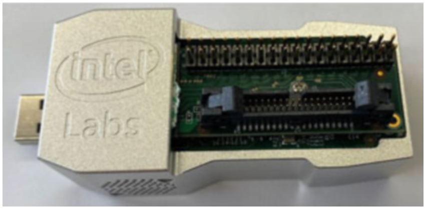


Fig. 15.1 Kapoho Bay with two onboard Loihi chips

Table 15.1 Introduction to Loihi and Loihi 2 chips

Feature	Loihi 1	Loihi 2
Technology	Intel 14 nm	Intel 4 (7 nm)
Die Area	60 mm^2	31 mm^2
Max # Neurons/Chip	128,000	1 million
Max # Synapses/Chip	128 million	120 million
Neuron Model	Generalized digital LIF	Fully programmable

support weight-sharing and compression. Synaptic plasticity in Loihi-1 chips can be fine-tuned using various biologically plausible learning rules, including Hebbian learning, spike-timing-dependent plasticity (STDP), and reward-modulated rules [21, 22]. Neurons fire when accumulated spikes reach a threshold within a specific time frame, transmitting signals to connected neurons.

Loihi-1 chips are seamlessly integrated with several neuromorphic platforms, which provide interfaces to combine the chip with other computer systems or Field-Programmable Gate Array (FPGA) devices. A shining example is the Kapoho Bay platform, which vividly demonstrates the versatility and collaborative potential of Loihi-1 chips, as depicted in Fig. 15.1. This cooperative aspect invites you, our esteemed audience, to be part of the transformative journey of neuromorphic systems.

Nahuku is a 32-chip Loihi board with a standard FPGA Mezzanine Card (FMC) connector, facilitating communication with the Arria FPGA development board. Pohoiki Spring is a large-scale Loihi system equipped with 100 million neurons and designed to function as a server for remote access. The second generation of Loihi chips, Loihi-2, was introduced in late 2021 [23]. Fabricated using Intel's four processes, previously called 7 nm technology, Loihi-2 has a reduced chip area of 31 mm^2, compared to the 60 mm^2 of the first-generation Loihi chips. Loihi-2 stands out with its unique feature of offering fully programmable neuron models, a significant improvement over the fixed neuron models in the earlier generation. These neuron behaviors can be customized with microcode instructions that support basic bitwise and mathematical operations. Designed for neuromorphic computing and edge devices, Loihi-2 provides enhanced computational and energy efficiency through parallel processing. A comparison between the two generations of Loihi chips is summarized in Table 15.1.

Loihi 1 utilizes a Leaky Integrate and Fire (LIF) model to implement neurons. The LIF model is favored because it captures neural dynamics' essential information processing functions while computationally simple enough for efficient evaluation. The behavior of LIF neurons is described by the following equations [24]:

$$C_m \frac{dV_m}{dt} = G_L\,(E_L - V_m) + A * I_{\text{app}}, \text{ if } V_m > V_{\text{th}} \text{ then } V_m = V_{\text{reset}}, \qquad (15.1)$$

$$\tau_{RC} = C_m / G_L, \qquad (15.2)$$

where C_m is the membrane capacitance, G_L is the leak conductance, E_L is the leak potential, V_m is the membrane potential, A is the input signal gain, I_{app} is the input current, and τ_{RC} is the membrane RC time constant. The Integrate and Fire (IF) model is a simplified version of the Leaky Integrate and Fire (LIF) model. Suppose neurons function similarly to LIF neurons but do not have a decaying membrane potential. This simplification is achieved by setting the equation's membrane resistance-capacitance (RC) time constant to infinity.

15.2.2 Associative Learning

Associative learning in rodents, particularly rats, has been extensively studied using the T-maze paradigm, a classical experimental setup designed to investigate the cognitive processes underlying spatial navigation and memory. In this maze, rats are typically exposed to a simple T-shaped structure with distinct arms and a choice point. The arms of the maze present different spatial cues or stimuli, and the rodent learns to associate specific outcomes or rewards with particular arm choices.

During the initial phases of T-maze experiments, rodents often display exploratory behaviors, navigating through the maze without a clear preference for either arm. However, as the learning trials progress, a remarkable shift occurs. Through repeated exposure to the maze, rats begin to form associations between specific stimuli or cues in the arms and the outcomes associated with those choices. This associative learning process is notably demonstrated when the rats consistently choose one arm over the other, guided by anticipating a positive result, such as a food reward.

Furthermore, T-maze studies have revealed that rats are adaptable to modifying their choices based on changes in the spatial cues or the introduction of new stimuli. For instance, if the location of the reward is switched from one arm to the other, rats exhibit a learning curve as they adjust their navigational preferences to align with the updated reward contingencies. This flexibility in adapting to changing conditions underscores the dynamic nature of associative learning in T-maze experiments.

The neural mechanisms underlying T-maze associative learning involve intricate interactions within the rat's brain, particularly in regions associated with memory, spatial navigation, and reward processing. The T-maze paradigm continues to serve as a valuable tool for unraveling the complexities of associative learning in rodents, shedding light on the fundamental principles governing cognitive processes and behavior in mammalian brains.

15.3 Fear Conditioning with Cognitive-Inspired Robots

This section outlines the methodology and design process for simulating fear conditioning in rats using mobile robotics. Instead of the traditional electric shock as the unconditional stimulus (US) in fear conditioning experiments, we used a vibration platform to deliver vibrations. Similarly, a mounted light source replaced the buzzer tone typically used as the conditional stimulus (CS). To ensure reliable results, the Leaky Integrate and Fire (LIF) neurons were efficiently implemented in Nengo, a neuromorphic simulator developed by Applied Brain Research [25]. For several experiments, Intel's Loihi chip served as the backend for Nengo, enhancing data processing efficiency. The Robot Operating System (ROS) framework managed the cognitive-inspired robot, miming certain aspects of human cognition, ensuring smooth data transfer to and from the Nengo program [26].

15.3.1 Simulation and Preliminary Testing

The central focus of our experiment is to modify signal pathways using Hebbian learning, a crucial aspect of our research. To demonstrate essential associative learning using Nengo, we constructed a primary network. This network includes two programmatically controlled input nodes representing the CS and US. As illustrated in Fig. 15.2, each input node is connected to a specific LIF neuron that activates in response to the stimuli. These neurons are labeled as CS neurons and US neurons. A third LIF neuron, the response neuron, is also established to indicate the network's response to the stimuli.

The values used for the LIF parameters from Eqs. (15.1 and 15.2) are given for the three LIF neurons in Table 15.2.

For all LIF neurons in Nengo, the firing threshold is fixed at 1 V, and input gain is modified instead. The initial value for τ_{RC} Nengo's standard implementation of

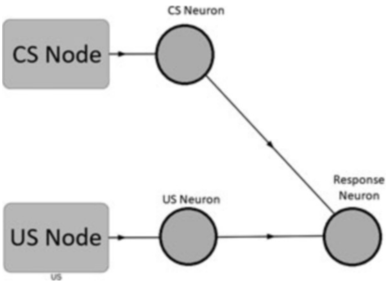


Fig. 15.2 Neural network for demonstrating Hebbian learning

Table 15.2 LIF neuron parameters for simple associative learning

Neuron types	τ_{RC}	A	V_{reset} (V)	V_{th} (V)
US neuron	0.02	1.0	0.01	1.0
CS neuron	0.02	1.0	0.01	1.0
Response neuron	0.02	1.0	0.01	1.0

spiking LIF neurons is 0.02, which was not modified as the neurons exhibited the desired behavior. V_{reset} was set at 0.01 V so the neurons fire when they receive the default programmable input stimulus without modifying the default gain of 1.0. The US neuron's output spikes are relayed to the response neuron via an unmodifiable synaptic connection modeled by the lowpass filter with the impulse response:

$$h(t) = \frac{1}{\tau} e^{\frac{-t}{\tau}} \tag{15.3}$$

The synaptic time constant, τ, is measured in seconds. We used a τ value of 0.005 seconds for all synapses, the default model in Nengo, which effectively serves our purpose. Synaptic connections have "weights," scalar values that influence the output signal, acting as a gain for the synapse output current. Inhibitory connections, which decrease the postsynaptic neuron's potential, are created by assigning a negative value to the synapse weight. The Hebbian learning rule, a fundamental principle in neuroscience, states that synapses between simultaneously active neurons will be strengthened. This principle, detailed in Eq. (15.4) of our neural network model, guides the synapses to strengthen when neurons fire together.

$$\Delta w = \eta r_i r_j, \tag{15.4}$$

where w is the synaptic weight, η is the learning rate constant, r_i is the firing rate of the presynaptic neuron, and r_j is the firing rate of the postsynaptic neuron. According to this equation, synaptic strength increases in proportion to the firing rates of both neurons when they fire simultaneously. Initially, the synaptic weight is set to 0.0001, allowing the US input signal to activate the response neuron, while the CS input signal alone does not elicit a response. The learning rate, empirically set at 2×10^{-5}, balances learning speed and network stability. A higher learning rate could cause instability, while a lower rate might lead to slow convergence. Each training cycle involves presenting the US and CS for two seconds each, with a one-second overlap, followed by one second without any stimulus, as shown in Fig. 15.3. This timing ensures sufficient exposure to the stimuli without overwhelming the network.

Our research initially demonstrates the network's response to both stimuli, followed by training cycles designed to increase the synaptic weight. After sufficient repetitions, the synapse strength rises to the point where the presentation of the CS alone triggers the response neuron, driven solely by the output of the CS neuron. This result highlights the potential of our approach, successfully demonstrating associative learning in the network.

After successfully validating the associative learning proof of concept in Nengo, we embarked on a pivotal phase of the mobile robotics experiment. Implementing a vibration platform as the unconditional stimulus (US) was significant. This was due to the inherent challenges of rapid collisions causing instability within the Gazebo simulation environment. We meticulously recorded precise vibration data to overcome this and seamlessly integrated it into the simulation. The vibration table was calibrated to generate 15 Hz vibrations with an amplitude of 1.2 mm, the

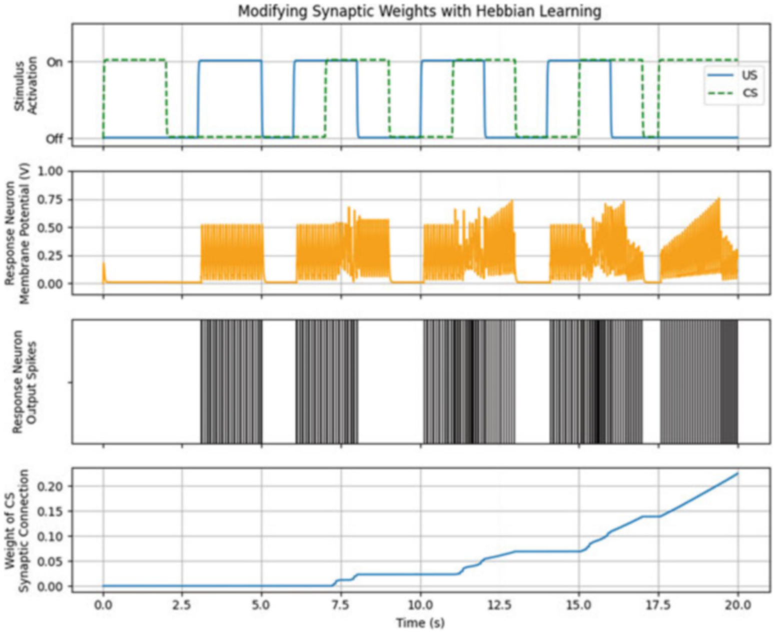


Fig. 15.3 Hebbian learning causing synaptic weight modification

lowest available setting, to ensure minimal impact on the cognitive-inspired robot. The robot's onboard IMU was instrumental in measuring these vibrations.

With Clearpath's ROS software operating on the robot, we received the IMU data and published the robot's acceleration to an IMU topic, making this data accessible to other ROS nodes. This setup allowed us to record and replay real-time vibration data and simulate real-world conditions. Nengo was executed within a ROS node that subscribed to the IMU topic, establishing a direct communication channel with Nengo. The raw acceleration data during vibration, as shown in Fig. 15.4, underwent minimal preprocessing before being integrated into the network.

This configuration was pivotal in enabling a realistic simulation of the robot's response to the unconditional stimulus. It ensured that the data utilized in Nengo accurately mirrored the conditions the robot would encounter in the real world. This meticulous approach to data handling and simulation setup was not just important but crucial for the success of the associative learning experiments. It provided a robust foundation for testing and validation, enhancing the credibility of our results.

The z-axis had an average magnitude of 9.81 m/s^2 due to Earth's gravity, which was subtracted to bring the resting acceleration for all three axes to zero. Initially, the resultant acceleration, as shown in Eq. (15.5), was used to evaluate the vibration state of the robot. However, the z-axis acceleration exhibited significant deviations from its resting value compared to the other axes, suggesting that z-axis acceleration alone was sufficient to indicate vibration. In a meticulous process, the preprocessing equation was then simplified as follows:

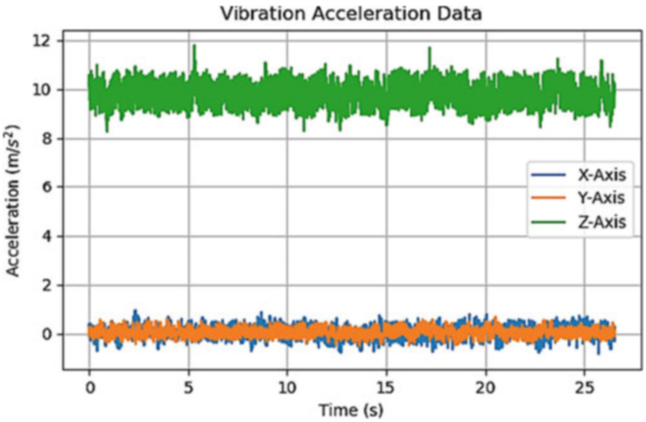


Fig. 15.4 Acceleration data from vibration table at 15 Hz

$$a_{res} = \sqrt{{a_x}^2 + {a_y}^2 + (a_z - 9.8)^2} \qquad (15.5)$$

However, as shown in Fig. 15.4, the z-axis acceleration deviates significantly more from its resting value than the other two axes. This observation indicates that measuring the z-axis acceleration alone is sufficient to detect vibrations. Consequently, the preprocessing equation was simplified to the following:

$$a_r = \mid a_z - 9.8 \mid \qquad (15.6)$$

This new equation is ideal as preprocessing should be minimized to take full advantage of the neuromorphic system's potential for efficiency. A comparison of a_{res} and a_r during vibration is shown in Fig. 15.5.

The rectified z-axis acceleration, which showed trends and magnitudes similar to the resultant acceleration, was suitable for vibration preprocessing. Therefore, the US input node was adjusted to receive IMU data and use the rectified z-axis acceleration as its output. This node functioned as a spike generator with a firing rate proportional to the input value, albeit with some modifications from the original design. To minimize preprocessing, raw values from the node remained unscaled and unnormalized. Instead, we carefully tuned the US neuron's input synapse and LIF parameters, now designated as the vibration detection neuron, to achieve the desired response to vibration signals. The parameters chosen for the simulation experiment, outlined in Table 15.3, were meticulously calibrated, reflecting the precision of our research methodology.

The default τ_{RC} constant of 0.02 s is maintained, as it is sufficient for the desired functionality. The other two parameters, gain (A) and bias (V_{reset}) are empirically calculated and optimized based on the experimental setup to ensure they produce the desired responses. These parameters were derived for the vibration detection neuron

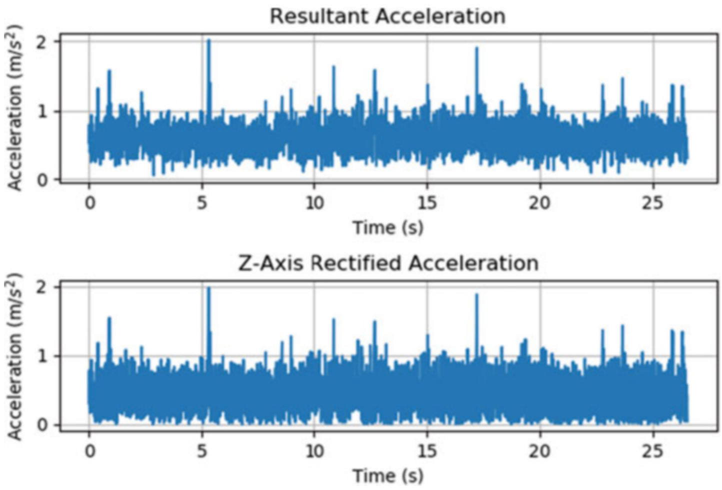


Fig. 15.5 Comparison of resultant acceleration, a_{res}, and rectified z-axis acceleration, a_r

Table 15.3 LIF neuron parameters for associative learning in mobile robotics simulation

Neuron types	τ_{RC}	A	V_{reset} (V)	V_{th} (V)
(US) Vibration neuron	0.02	1.3	0.6	1.0
(CS) Brightness neuron	0.02	0.9	0.15	1.0
(Response) Movement neuron	0.02	1.0	0.01	1.0

to ensure the neuron continuously sends output spikes only when the vibration platform is enabled, as shown in Fig. 15.6.

Moreover, the filter time constant of the synapse connecting the input to the vibration neuron is increased to 0.2 s. This modification helps to ensure that the vibration neuron does not fire in response to minor, abrupt movements of the cognitive-inspired robot, such as halting.

Our experiment involved using a simulated stereo camera to measure the brightness of a light, a departure from the traditional use of a buzzer tone in rat experiments. While Gazebo does not directly support the ZED 2 stereo camera on our cognitive-inspired robot, it does provide functionality for the Bumblebee2 camera, which serves our needs. We focused on the right camera of the Bumblebee2, as a single camera was sufficient for our experiment and met our requirements. The Gazebo environment, as shown in Fig. 15.7, was meticulously designed to closely mimic our real-world experimental setup, complete with a light panel and a placeholder for the vibration platform.

The light panel has a circular light centered in the right camera frame when the cognitive-inspired robot is on the vibration platform, as shown in Fig. 15.8.

Integrating image data with Nengo was a complex task, surpassing the challenges of processing acceleration data. It necessitated the creation of a custom “Nengo process” and additional support structures. These Nengo processes, versatile in their

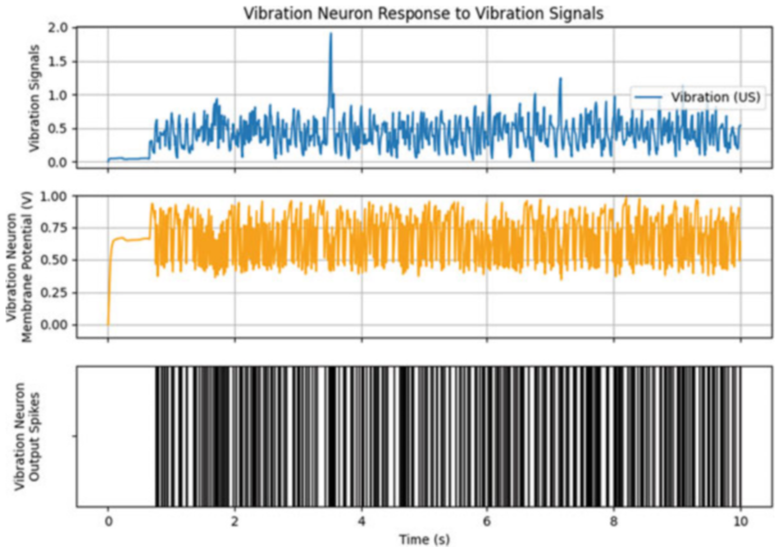
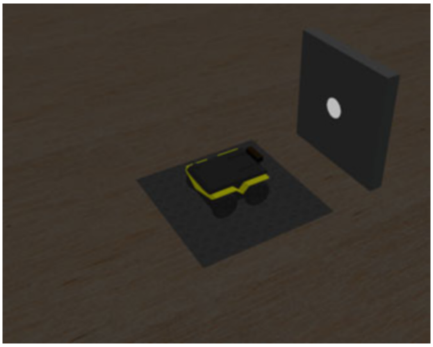


Fig. 15.6 Vibration neuron response to acceleration

Fig. 15.7 Gazebo environment setup for associative learning experiment



functions, were utilized to generate node outputs and simulate dynamical systems for neuron groups. Specifically, we developed a Nengo process, dubbed the "camera process," to manage image data for our experiment. This process retrieves images from a dedicated ROS service, reduces their resolution, and converts pixel values from 0 to 255 to a continuous range of -1 to 1, rendering them suitable as stimulus inputs. The ROS service, subscribing to the image data topic, stores the most recent frame, acting as a single-sample buffer to prevent data overload.

Initial tests indicated that the framerate of the image data was inadequate, necessitating a shift to a compressed image stream. This shift entailed the development of additional software to interface with and decode the images, resulting in specific modifications to Nengo and the ROS service. These modifications were crucial in ensuring the successful implementation of the camera process and establishing a steady stream of images. The CS node was subsequently altered to utilize the

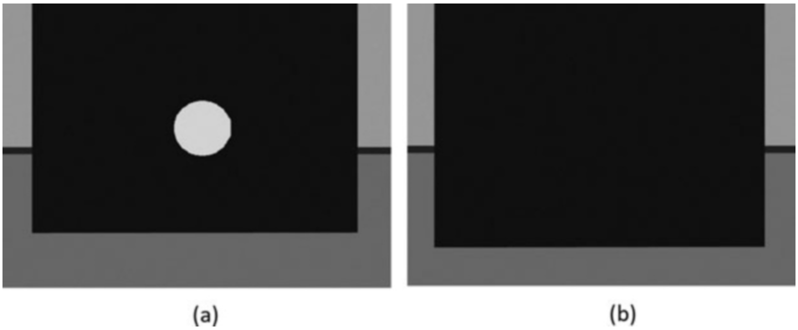


Fig. 15.8 Images from the simulated camera light panel in Gazebo (on and off)

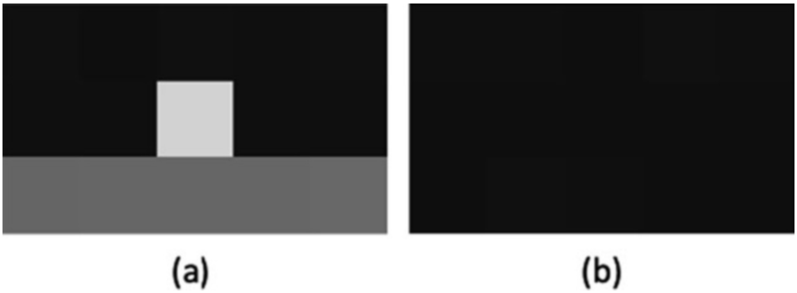


Fig. 15.9 Resized images for brightness detection in simulation (on and off)

camera process, enabling the output of rescaled pixel-intensity data to the network. This marked a significant milestone in our experiment, showcasing the successful integration of image data with Nengo. As shown in Fig. 15.9, the image is rescaled to 5 × 3 pixels. Hence, one center pixel contains the approximate average value of the entire light, and the camera process' extraneous computations are minimized.

The output of the CS node is derived from the intensity of the central pixel, which is then connected to the brightness neuron. Other outputs from the CS node remain unused. The Leaky Integrate and Fire (LIF) parameters for the brightness neuron, illustrated in Fig. 15.10, were meticulously and empirically optimized to ensure the desired functionality, much like the adjustments made for the vibration neuron. These parameters enable the brightness neuron to fire continuously when the light panel is active and to cease firing immediately once the light is turned off, as shown in Fig. 15.10.

Finally, the response neuron was adapted into a movement neuron by creating a "movement output node" that issues movement commands to the cognitive-inspired robot, mimicking the fear response seen in rats. To enhance stability during Nengo's execution and avoid issues such as division by zero, a slight bias was added to V_{reset} the movement neuron, ensuring smooth integration with the movement node.

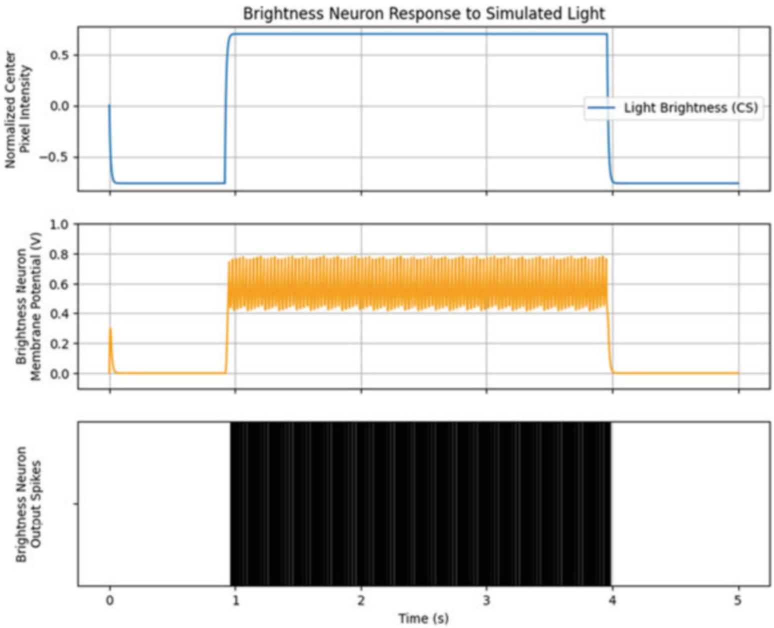


Fig. 15.10 Brightness neuron activating with light stimulus in simulation

The movement node is crucial to the system, facilitating communication with the cognitive-inspired robot through a ROS movement topic. Nengo publishes the movement data, which is then received and processed by the Jackal software. Clearpath's software on the robot interprets these commands, controls the motor drivers, and uses wheel encoders and other sensors to calculate odometry. When the movement neuron fires, the movement node directs the robot backward at a speed of 0.3 m/s. This movement command is maintained for approximately one second after the last spike, after which the robot returns to a neutral position, as depicted in Fig. 15.11.

Figure 15.12 shows that the associative learning network and three new functioning subsystems are ready to be tested in the simulation experiment.

Implementing real-time modifications in Nengo is a significant milestone in our research. This innovation underscores the crucial role of multithreaded programming in aligning Nengo's simulation time steps with real-time operations and successfully bridges the gap between Nengo and ROS/Gazebo. Traditionally, Nengo runs time steps as quickly as possible, while ROS and Gazebo aim for real-time execution, often resulting in Nengo receiving time-stretched, 'slow-motion' data. Our approach has successfully overcome this challenge, allowing Nengo networks to operate with a cumulative difference of less than one timestep from perfect real-time. This achievement significantly boosts performance and eliminates the need for further modifications.

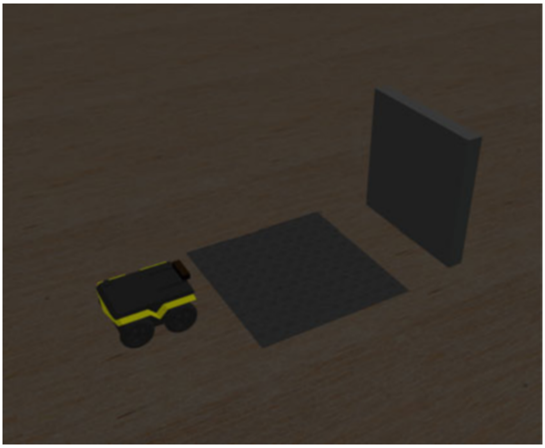


Fig. 15.11 The robot moved to a neutral position after movement response

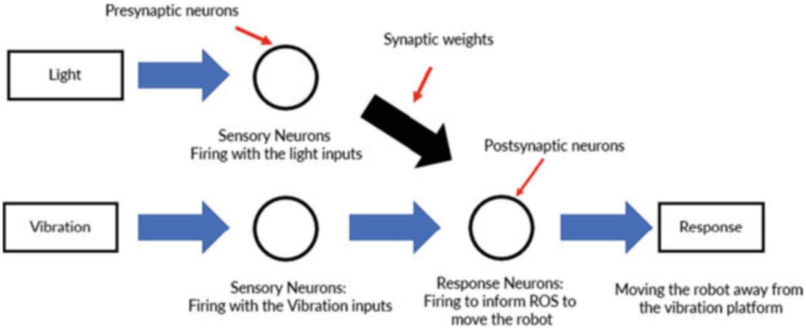


Fig. 15.12 System for associative learning with mobile robotics experiment

This advancement, made possible by our real-time modifications, has been instrumental in transitioning from simulation to real-world experiments and scaling up the network size. The impact of this transition is significant, as it opens up new possibilities for our research. Although NengoGUI offers real-time simulation execution, our custom techniques have proven far more accurate and consistent in timing. This accuracy is due to our custom real-time Nengo, which synchronizes the average time step execution with real-time rather than focusing on each step. NengoGUI's higher resource consumption and computational limitations make it less suitable for larger, more complex networks. Our custom real-time solution addresses these limitations, highlighting its critical value in the field and paving the way for exciting future research.

The real-time modifications enabled the successful simulation of the experiment within the Gazebo environment. The experiment followed a systematic process: the light was initially activated to show the absence of response to the CS, then the vibration table was 'activated' by replaying vibration data, demonstrating the US triggering the movement response. We then conducted training cycles with overlapping periods of the two stimuli. Finally, the light was reactivated to trigger the movement response without the vibration stimulus, showcasing successful associative learning through the Hebbian modification of synaptic weight. Throughout the process, the learning rate was consistently maintained at 2×10^{-5}, demonstrating the stability and reliability of our real-time modifications.

15.3.1.1 Experimental Validation

One of the most exciting achievements of our research was successfully transitioning the associative learning experiment from a simulated environment to a real-world application. This milestone opens up numerous possibilities for practical applications. This success was made possible through a series of modifications. Significantly, the vibration input node and vibration neuron had already been optimized for real-world vibration signals, eliminating the need for further adjustments. Moreover, the existing movement response neuron and output node proved adequate for the real-world setup. The new experimental configuration, shown in Fig. 15.13, primarily differed in its approach to brightness perception, showcasing the adaptability and scalability of our method. This setup was effectively tailored to accommodate specific real-world conditions, demonstrating its robustness and practicality. For example, adapting to [specific real-world conditions] revealed [explain the implications of these adaptations], underscoring the flexibility of our approach. This ability to seamlessly adjust the experimental setup to real-world conditions highlights the strength of our research methodology and its potential for practical applications in dynamic environments.

The cognitive-inspired robot was positioned on a testing platform constructed from nine wooden panels, each measuring 23 inches by 23 inches and standing eight inches tall. The center panel, highlighted in red in Fig. 15.13, served as a vibration platform with a vibration table underneath, delivering 15 Hz vibration signals that the IMU monitored. Adding a new background, light panel, and environmental lighting resulted in a completely different set of images for the updated camera, as depicted in Fig. 15.14.

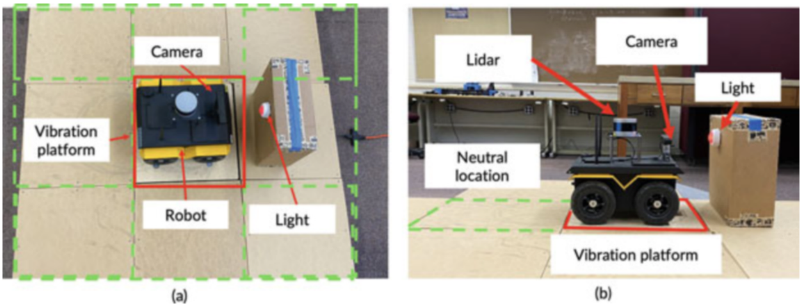


Fig. 15.13 Experimental setup for real-world associative learning experiment

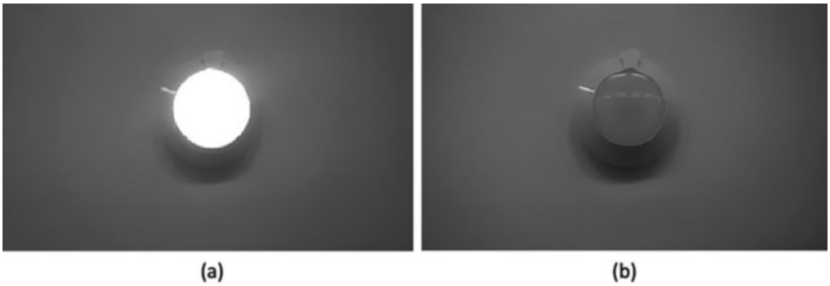


Fig. 15.14 Images from the ZED 2 camera showing the light panel on and off

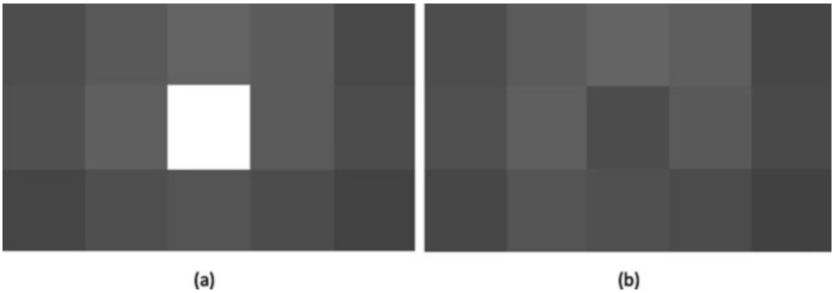


Fig. 15.15 Example images of the light panel from the real-world experiment

Table 15.4 LIF neuron parameters for associative learning in real-world robotics experiment

Neuron types	τ_{RC}	A	V_{reset} (V)	V_{th} (V)
(US) Vibration neuron	0.02	1.3	0.6	1.0
(CS) Brightness neuron	0.02	0.3	−1.0	1.0
(Response) Movement neuron	0.02	1.0	0.01	1.0

The new camera requires a revised Nengo camera process and ROS service to input image data into the network, but the 5 × 3 image center pixel output setup remains the same (Fig. 15.15).

Because of the new environment, the LIF parameters must again be empirically derived, yielding the values listed in Table 15.4.

The updated brightness neuron parameters reproduce the desired brightness detection function of the neuron, as shown in Fig. 15.16.

Now that all subsystems are functional in the real-world experimental setup, the same process from the simulation is used to train the network, as shown in Figs. 15.17 and 15.18.

The exact value 2×10^{-5} is used as the learning rate η for the Hebbian learning process. As demonstrated in Fig. 15.19, the strength of the synaptic connection in the conditional pathway increases as the stimuli are presented simultaneously.

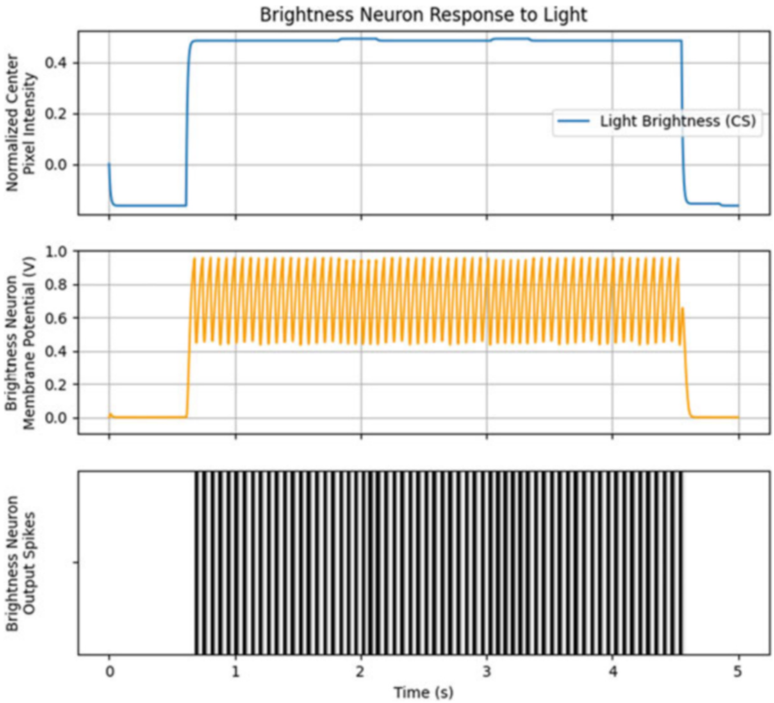


Fig. 15.16 Brightness neuron firing in response to the activated light panel

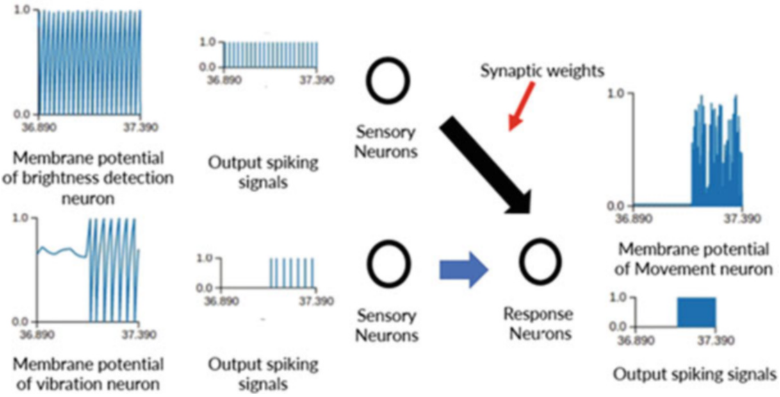


Fig. 15.17 Membrane potentials and spiking outputs of brightness detection, vibration, and movement neurons

The initial findings, illustrated in Fig. 15.19, demonstrated effective associative learning in a mobile robotics adaptation of the fear conditioning experiment. Although the system leverages biological principles to achieve associative learning,

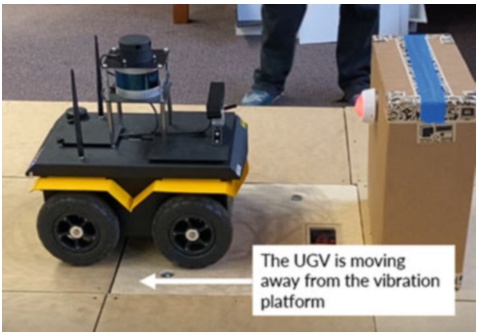


Fig. 15.18 Cognitive-inspired robot moving away from vibration platform

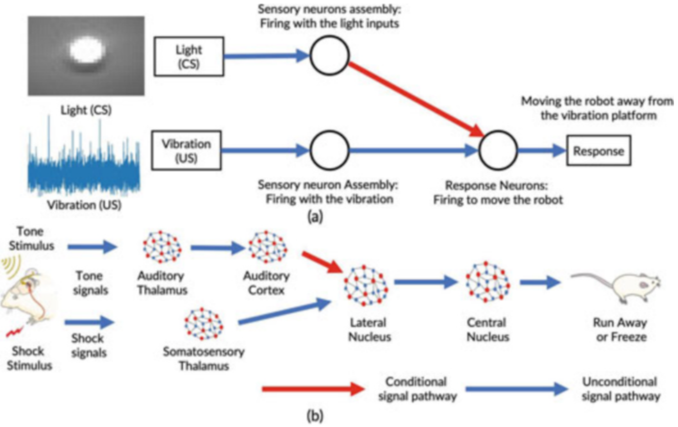

Sensory neurons assembly: Firing with the light inputs
Light (CS)
Light (CS)
Moving the robot away from the vibration platform
Vibration (US)
Response
Vibration (US)
Sensory neuron Assembly: Firing with the vibration
Response Neurons: Firing to move the robot
(a)
Tone Stimulus
Tone signals
Auditory Thalamus
Auditory Cortex
Lateral Nucleus
Central Nucleus
Run Away or Freeze
Shock Stimulus
Shock signals
Somatosensory Thalamus
Conditional signal pathway
Unconditional signal pathway
(b)

Fig. 15.19 Synaptic weight change from Hebbian learning in a real-world experiment

the current neural assemblies must be simplified. They may be more suitable for handling more intricate information-processing tasks.

15.3.1.2 Locally Competitive Algorithm

In our quest to improve the associative learning network, we developed a novel Locally Competitive Algorithm (LCA) network inspired by sparse coding for the brightness detection neural assembly. As illustrated in Fig. 15.20, this network used the same camera process to connect with the image stream. However, the images were resized to 24 × 48 pixels to simulate a more realistic image processing scenario. This successful implementation of the LCA network highlighted our research's innovative approach and effectiveness, paving the way for handling more complex information-processing tasks.

To minimize redundant neural computations, we focused on using only the center region of the image containing the light as input to the LCA network. Usually, in convolutional LCA, input patches overlap, leading to connections between feature

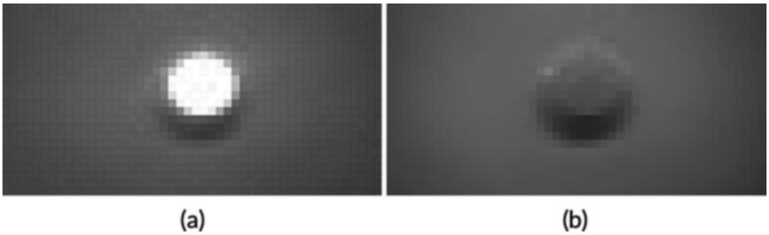


Fig. 15.20 Higher resolution images of the light panel on (**a**) and off (**b**)

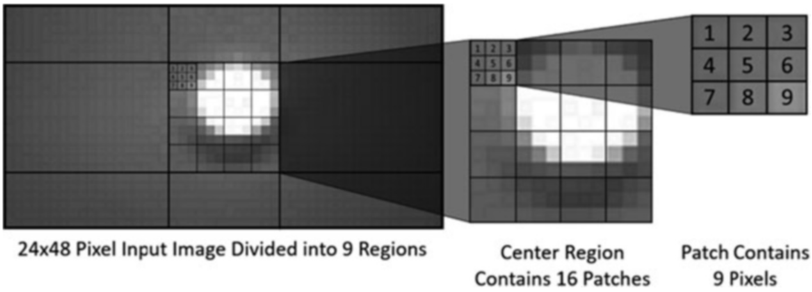

1 2 3
4 5 6
7 8 9
24x48 Pixel Input Image Divided into 9 Regions
Center Region Contains 16 Patches
Patch Contains 9 Pixels

Fig. 15.21 Image division layout showing regions and patch structure

neurons of these patches. We selected a convolutional stride of three to prevent this overlap and isolate each LCA network. This strategy aimed to preserve the necessary functionality while simplifying the implementation and reducing computational requirements. Each of the 16 patches in the center region was fed into individual LCA optimization networks, as depicted in Fig. 15.21.

To efficiently manage neural activity computations, we focused solely on the center region containing the light for input to the LCA network. Unlike traditional convolutional LCA, which uses overlapping input patches and creates connections between feature neurons, we used a convolutional stride of three to avoid overlaps and isolate each LCA network. This method simplifies the implementation of the LCA network and reduces computational resource requirements while maintaining the desired functionality. As illustrated in Fig. 15.21, each of the 16 center region patches is processed by individual LCA optimization networks (Fig. 15.22).

This configuration is termed single layer because it comprises one layer of "feature neurons" that solve the LCA optimization function in conjunction with the synapses connecting these neurons to the inputs.

$$\dot{u} = \frac{1}{\tau}\left(\Phi^T x - u - \left(\Phi^T \Phi - I\right), a = T_\lambda(u)\right. \tag{15.7}$$

$$T_\lambda(u) = 0 \text{ if } u \leq \lambda, \quad \text{else } \; T_\lambda(u) = u - \lambda \tag{15.8}$$

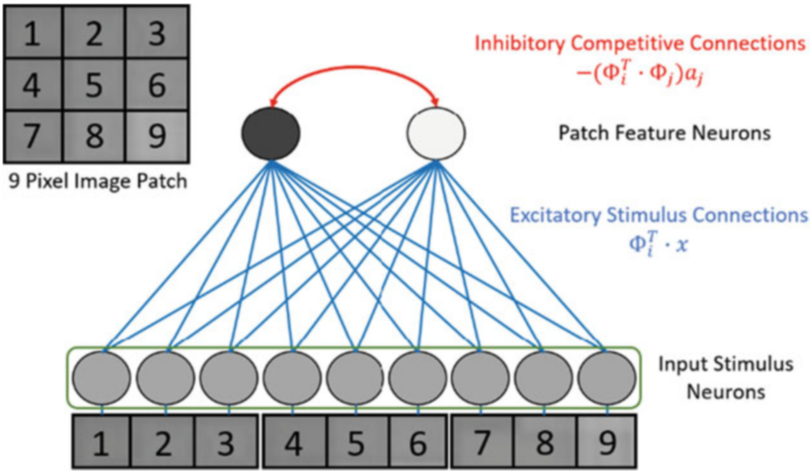


Fig. 15.22 Single-layer LCA network for brightness detection in one image patch

The input x is transmitted from the output of the camera process to the LCA layer, which then solves for the "sparse code." This layer adjusts the firing rates of each feature neuron to converge to the coefficients a_i. Each feature neuron receives input associated with a dictionary atom or feature Φ_i, as the weights of the synapses connecting it to the input are elements of the vector Φ_i.

For this experiment, a dark feature was manually created for the dictionary by selecting vector elements that are all negative, reflecting pixel intensities ranging from -1 to 1. Similarly, a light feature was created using positive vector elements. All feature vectors Φ_i must have a unit norm to ensure the magnitude of the feature vectors does not impact the sparsity penalty in Eq. (15.3). Ideally, the dictionary Φ should be overcomplete, meaning the dimension should be more significant than x's. However, this experiment uses only two features—light and dark. Hence, the neural assembly for brightness detection can only be described as "inspired by" sparse coding. While the under-complete dictionary does not fully solve the sparse coding problem and may not converge to a solution, the underlying competitive mechanisms are considered sufficient for achieving the desired functionality.

It could be overcomplete to enhance the dictionary by increasing the number of features and, consequently, the number of feature neurons, exceeding the input size. Training with spiking neurons could also generate learned dictionary features using a modified LCA implementation [27]. These additional feature neurons could remain solely positive and negative, representing variations of light and dark features with minimal theoretical changes. Alternatively, more complex features could be introduced to reduce unnecessary neural activity and improve the network's accuracy and efficiency.

Alternatively, developing features like gradients or other essential image elements could minimize the unwanted activity of the light feature neurons by offering better representations of the less-defined patches of the image. Additionally, expanding the input to accept RGB pixel values instead of scalars would enable more selective dictionary features, theoretically reducing unwanted activity in feature neurons when presented with dissimilar inputs. The brightness perception network incorporates a second layer following the LCA layer, which integrates the output spikes from the corresponding patch feature neurons in each region, as illustrated in Fig. 15.23.

A proper LCA implementation necessitates that the LCA layer (feature neurons) use the Integrate and Fire neuron model. Conveniently, the existing brightness neuron from Table 15.4 fulfills the required function of the Layer 2 neuron in Fig. 15.23 without any modifications. Table 15.5 summarizes the parameters for the neurons in the network.

Layer 1's V_{reset} parameter was empirically fine-tuned to achieve the desired functionality. The synapse model was adjusted by eliminating the filter and multiplying the weights by the spikes to implement spiking LCA. The synaptic weights connecting to the feature neurons were set to match the values of the correspond-

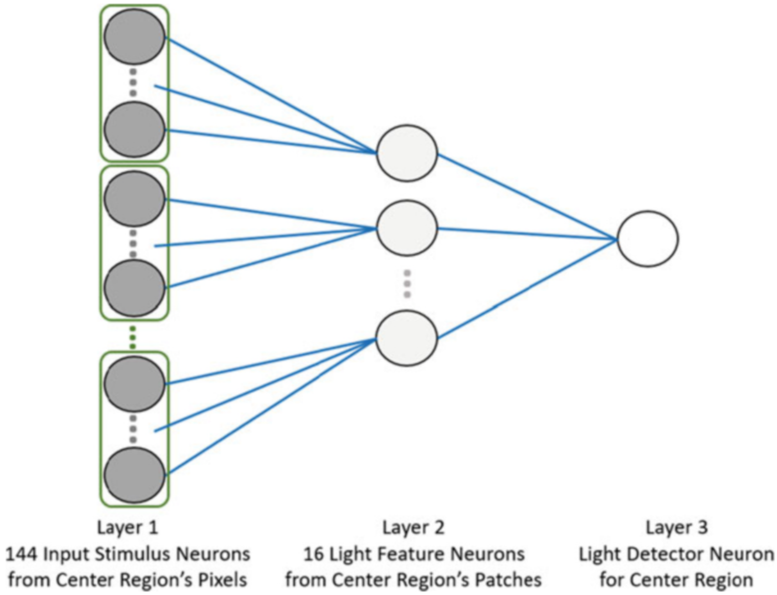


Fig. 15.23 Partition of brightness perception network used for the images' center region (note that only the neurons associated with the light feature are depicted)

Table 15.5 LCA neuron parameters for light detection

Neuron types	τ_{RC}	A	V_{reset} (V)	V_{th} (V)
LCA neuron (Layer 1)	0.02	1.0	$-\lambda = 0.85$	1.0
Light detector neuron (Layer 2)	0.02	0.3	-1.0	1.0

Table 15.6 Synaptic weights for LCA neurons

Presynaptic connection	Postsynaptic connection	Weight
Input stimulus	Light feature neuron	0.111
Input stimulus	Dark feature neuron	−0.111
Dark feature neuron	Light feature neuron	−1.0
Light feature neuron	Dark feature neuron	−1.0

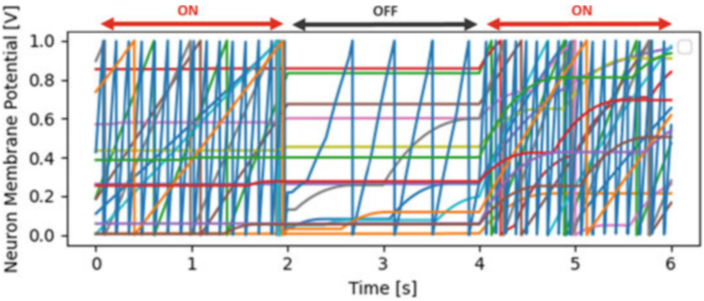


Fig. 15.24 Membrane potentials of light feature neurons of the center region

ing dictionary atom. Feature neurons have inhibitory connections with weights determined by the matrix $-\left(\Phi_i^T \bullet \Phi_j\right) a_j$. The synaptic weights are detailed in Table 15.6.

The synapses linking the inputs to the light and dark feature neurons are precisely designed to generate rapid responses. These synapses produce excitatory signals when the corresponding input (light for light neurons and dark for dark neurons) is detected, and inhibitory signals are produced when the opposite input is present. This quick reaction, a key trait of neuronal behavior, underscores the system's efficiency and is a direct outcome of the synaptic design. However, it is fascinating to note that Loihi 1 neurons, unlike most biological neurons, cannot be excitatory and inhibitory simultaneously, which adds a unique dimension to the functioning of neural networks.

When the network is exposed to an image with the light turned on, the activity level of the light feature neurons in the center region undergoes a significant surge. In contrast, when the image switches to one with the light turned off, the activity of these neurons promptly plummets. This alternating pattern, with each image displayed for two seconds, creates a unique sequence of neuronal activity. As depicted in Fig. 15.24, this sequence vividly showcases the network's dynamic response to changing stimuli, a phenomenon that never fails to inspire awe, highlighting its efficiency and adaptability.

The output spikes from the light feature neurons drive the activity of the second-layer brightness detection neuron, which is shown in Fig. 15.25. The neuron's output reliably spikes when the light is on and not off.

Integrating the brightness detection LCA network with the associative learning network was smooth, as both were implemented using Nengo. The existing network was modified by replacing the brightness neuron with the LCA network, which

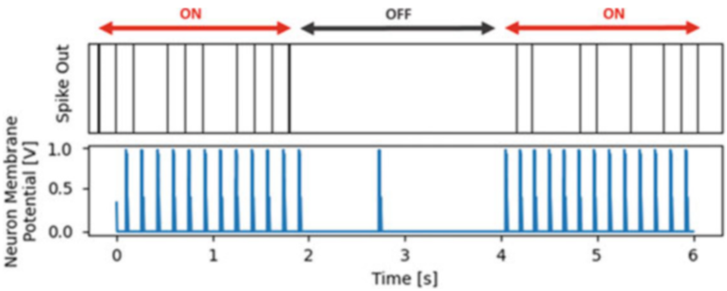


Fig. 15.25 Membrane potential and spike output of layer two brightness detection neuron corresponding to the center region

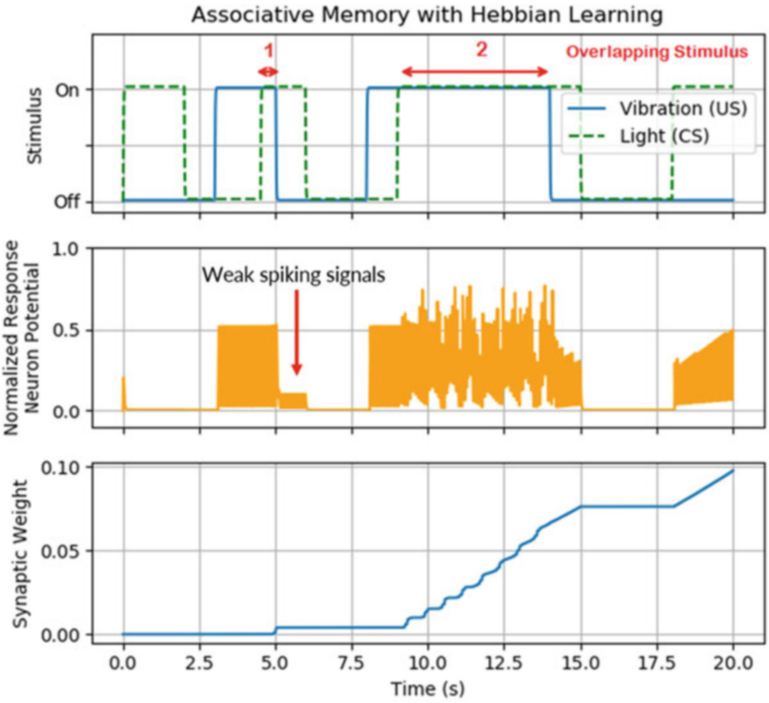


Fig. 15.26 Weight change from Hebbian learning in an experiment with LCA

now serves as the output from the camera process. No further adjustments were necessary because the layer two neuron, which connects to the learning synapse or conditional pathway, remains the same as the previous brightness neuron. With this updated configuration, the associative learning experiment was repeated, yielding significant results, as shown in Fig. 15.26.

The next stage of our experiment involves deploying it onto Intel's Loihi neuromorphic hardware, a step that enhances real-world applicability and leverages

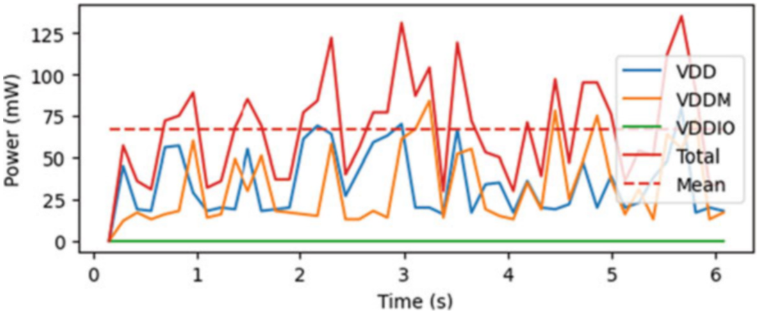


Fig. 15.27 Power consumption of LCA network running on Loihi

neuromorphic computing's performance and energy efficiency. The Nengo Loihi program, a truly innovative creation by ABR, utilizes Intel's NxSDK platform to run Nengo network models on Loihi, ensuring seamless integration.

This extension allows Nengo models to be executed on Loihi without modifying the network neurons and modules, a testament to our system's adaptability. However, the extension needs adaptation to incorporate Hebbian learning because only the STDP learning rule is natively supported. Consequently, the majority of the LCA network runs on Loihi. Intel's NxSDK software, optimized for running LCA on Loihi, facilitates the creation of the same LCA network used in previous experiment stages.

A significant limitation of Loihi 1 is its inability to produce inhibitory output spikes necessary for standard spiking LCA, a constraint not present in Loihi 2. While NxSDK developers have proposed a workaround for Loihi 1, which involves [detailed description of the workaround], its effectiveness is still being evaluated. Additional architectural adjustments required for Loihi include using a 4-bit resolution for synaptic weights, as opposed to the 24-bit floating-point resolution used by Nengo, and reducing the simulation time step from 1 ms to 20 ms when running on Loihi.

Power consumption is meticulously monitored during the execution of the LCA network on Loihi to evaluate energy efficiency. Power measurements are taken for compute logic (VDD), SRAM memory units (VDDM), and the IO interface (VDDIO), with the results shown in Fig. 15.27.

Power measurements are made throughout the experiment, with values averaged and reported every eight steps. These measurements are crucial as they provide insights into the system's energy efficiency, a key consideration for real-world applications. Figure 15.27 shows that VDDIO's power consumption is negligible compared to the other two measurements. VDD and VDDM have approximately equal average power consumptions of 30 and 29 mW, respectively. This indicates that the system operates within acceptable power consumption limits, making it a viable option for energy-efficient neuromorphic computing applications.

Table 15.7 Comparison of scale and association capability with other state-of-the-art works

	Neuron	Synapse	Dataset	Learning scheme	Biology scenarios
[28]	6	3	N/A	Simulation	N/A
[29]	3	1	N/A	Simulation	N/A
[30]	5	6	N/A	Simulation	N/A
[31]	3	1	N/A	Simulation	N/A
[32]	3	1	N/A	Simulation	N/A
[33]	3	2	N/A	Simulation	N/A
[34]	3	2	N/A	Simulation	Cellular Association in Aplysia
[35]	20	100	Pretrained with datasets	Simulation	N/A
This work	1419	1420	No dataset for pretraining	Experiment	Fear conditioning of rats

In the experiments, the synaptic weights between the brightness detection neuron and the movement neuron are modified during the training process. As a result, after associative memory learning, the mobile robot will move away from the vibration platform under the stimulus of light, even with no vibration signal presented, demonstrating successful online learning in real time. Compared to other state-of-the-art associative memory works listed in Table 15.7, we reproduce the classic fear conditioning experiments of rats using a mobile robot and the Loihi chip rather than simply simulation. In addition, the scales of our neural networks outperform other works.

15.4 Conclusion

This study introduces a novel self-learning paradigm based on associative learning, specifically fear conditioning, using a mobile robot equipped with a neuromorphic system (Loihi chip) for online learning. The robot learns to respond to light signals through repeated simultaneous exposure to light (conditional stimulus) and vibration (unconditional stimulus), mimicking the fear conditioning process observed in rats. The simplicity and computational efficiency of Leaky Integrate and Fire (LIF) neurons are utilized for detecting lights and vibrations, with Hebbian learning facilitating signal pathway modifications.

Future research avenues include improving the Locally Competitive Algorithm (LCA) implementation for broader applicability, applying sparse coding and LCA to natural language processing, and streamlining vibration detection. Enhancing the system's performance could involve incorporating Hebbian learning into Nengo Loihi and choosing the appropriate development platform (Intel's Lava) and camera (DVS). These considerations are crucial for optimizing outcomes in neuromorphic computing, robotics, and AI research.

Acknowledgment This work was supported by the Robust Intelligence program in Directorate for Computer and Information Science and Engineering (CISE) of National Science Foundation under Award Number 2245712.

References

1. Goodfellow, I., Yoshua, B., & Aaron, C. (2016). Deep learning. *Deep Learning*, 785. https://doi.org/10.1016/B978-0-12-391420-0.09987-X
2. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv*, 1810.04805.
3. Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning*. MIT Press.
4. Sun, C., Shrivastava, A., Singh, S., & Gupta, A. (2017, 2017). Revisiting unreasonable effectiveness of data in deep learning era. *Proceedings of the IEEE International Conference on Computer Vision*, 843–852. https://doi.org/10.1109/ICCV.2017.97
5. Sengupta, S., et al. (2020). A review of deep learning with special emphasis on architectures, applications and recent trends. *Knowledge-Based Systems, 194*, 105596.
6. An, H., Al-Mamun, M. S., Orlowski, M. K., Liu, L., & Yi, Y. (2020). Robust deep reservoir computing through reliable Memristor with improved heat dissipation capability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.
7. Zins, N., Zhang, Y., Yu, C., & An, H. (2023). Neuromorphic computing: A path to artificial intelligence through emulating human brains. In *Frontiers of Quality Electronic Design (QED)* (pp. 259–296). Springer.
8. Roy, K., Jaiswal, A., & Panda, P. (2019). Towards spike-based machine intelligence with neuromorphic computing. *Nature, 575*(7784), 607–617.
9. Mead, C. (1990). Neuromorphic electronic systems. *Proceedings of the IEEE, 78*(10), 1629–1636.
10. An, H. (2020). *Powering next-generation artificial intelligence by designing three-dimensional high-performance neuromorphic computing system with memristors*. Virginia Tech.
11. Bai, K., & Yi, Y. (2019). Opening the "Black Box" of silicon chip design in neuromorphic computing. In *Bio-inspired technology*. IntechOpen.
12. Bai, K., & Yi, Y. (2018). DFR: An energy-efficient analog delay feedback reservoir computing system for brain-inspired computing. *ACM Journal on Emerging Technologies in Computing Systems (JETC), 14*(4), 45.
13. Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S. A., & Hudspeth, A. (2000). *Principles of neural science*. McGraw-Hill.
14. Baird, E., Srinivasan, M. V., Zhang, S., & Cowling, A. (2005). Visual control of flight speed in honeybees. *Journal of Experimental Biology, 208*(20), 3895–3905.
15. Kern, R., Boeddeker, N., Dittmar, L., & Egelhaaf, M. (2012). Blowfly flight characteristics are shaped by environmental features and controlled by optic flow information. *Journal of Experimental Biology, 215*(14), 2501–2514.
16. Kempter, R., Gerstner, W., & Van Hemmen, J. L. (1999). Hebbian learning and spiking neurons. *Physical Review E, 59*, 4498–4514. https://doi.org/10.1103/PhysRevE.59.4498
17. Levy, N., Horn, D., Meilijson, I., & Ruppin, E. (2000). Distributed synchrony of spiking neurons in a Hebbian cell assembly. *Advances in Neural Information Processing Systems 12, 14*, 129–135.
18. Van Rossum, M. C., Bi, G. Q., & Turrigiano, G. G. (2000). Stable Hebbian learning from spike timing-dependent plasticity. *Journal of Neuroscience, 20*(23), 8812–8821.
19. Caporale, N., & Dan, Y. (2008). Spike timing–dependent plasticity: A Hebbian learning rule. *Annual Review of Neuroscience, 31*, 25–46.
20. Bai, K., An, Q., & Yi, Y. (2019). Deep-DFR: A memristive deep delayed feedback reservoir computing system with hybrid neural network topology. In *Proceedings of the 56th annual design automation conference 2019* (p. 54). ACM.

21. Davies, M., et al. (2021). Advancing neuromorphic computing with Loihi: A survey of results and outlook. *Proceedings of the IEEE*.
22. Davies, M., et al. (2018). Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro, 38*(1), 82–99.
23. Orchard, G., et al. (2021). Efficient neuromorphic signal processing with Loihi 2. In *2021 IEEE workshop on signal processing systems (SiPS)* (pp. 254–259). IEEE.
24. Miller, P. (2018). *An introductory course in computational neuroscience*. MIT Press.
25. Bekolay, T., et al. (2014). Nengo: A Python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics, 7*, 48.
26. DiLuoffo, V., Michalson, W. R., & Sunar, B. (2018). Robot Operating System 2: The need for a holistic security approach to robotic architectures. *International Journal of Advanced Robotic Systems, 15*(3), 1729881418770011.
27. Parpart, G. G., et al. (2022). Dictionary learning with accumulator neurons. In *ICONS*.
28. Yang, J., Wang, L., Wang, Y., & Guo, T. (2017). A novel memristive Hopfield neural network with application in associative memory. *Neurocomputing, 227*, 142–148. https://doi.org/10.1016/j.neucom.2016.07.065
29. Liu, X., Zeng, Z., & Wen, S. (2016). Implementation of memristive neural network with full-function Pavlov associative memory. *IEEE Transactions on Circuits and Systems I: Regular Papers, 63*(9), 1454–1463.
30. Hu, X., Duan, S., Chen, G., & Chen, L. (2017). Modeling affections with memristor-based associative memory neural networks. *Neurocomputing, 223*, 129–137. https://doi.org/10.1016/j.neucom.2016.10.028
31. Moon, K., et al. (2014). Hardware implementation of associative memory characteristics with analogue-type resistive-switching device. *Nanotechnology, 25*(49), 495204. https://doi.org/10.1088/0957-4484/25/49/495204
32. Ziegler, M., et al. (2012). An electronic version of Pavlov's dog. *Advanced Functional Materials, 22*(13), 2744–2749. https://doi.org/10.1002/adfm.201200244
33. Pershin, Y. V., & Di Ventra, M. (2010). Experimental demonstration of associative memory with memristive neural networks. *Neural Networks, 23*(7), 881–886.
34. An, H., Zhou, Z., & Yi, Y. (2017). Memristor-based 3D neuromorphic computing system and its application to associative memory learning. *In 2017 IEEE 17th International Conference on Nanotechnology, NANO 2017* (pp. 555–560). New York: IEEE. https://doi.org/10.1109/NANO.2017.8117459
35. An, H., An, Q., & Yi, Y. (2019). Realizing behavior level associative memory learning through three-dimensional memristor-based neuromorphic circuits. In *IEEE Transactions on Emerging Topics in Computational Intelligence*. New York: IEEE.