A bio-inspired hardware implementation of an analog spike-based hippocampus memory model

Daniel Casanueva-Morato, *Member, IEEE,* Alvaro Ayuso-Martinez, *Member, IEEE,* Giacomo Indiveri, *Senior Member, IEEE* J. P. Dominguez-Morales, *Member, IEEE,* and Gabriel Jimenez-Moreno, *Member, IEEE*

***Abstract*—The need for processing at the edge the increasing amount of data that is being produced by multitudes of sensors has led to the demand for mode power efficient computational systems, by exploring alternative computing paradigms and technologies. Neuromorphic engineering is a promising approach that can address this need by developing electronic systems that faithfully emulate the computational properties of animal brains. In particular, the hippocampus stands out as one of the most relevant brain region for implementing auto associative memories capable of learning large amounts of information quickly and recalling it efficiently. In this work, we present a computational spike-based memory model inspired by the hippocampus that takes advantage of the features of analog electronic circuits: energy efficiency, compactness, and real-time operation. This model can learn memories, recall them from a partial fragment and forget. It has been implemented as a Spiking Neural Networks directly on a mixed-signal neuromorphic chip. We describe the details of the hardware implementation and demonstrate its operation via a series of benchmark experiments, showing how this research prototype paves the way for the development of future robust and low-power mixed-signal neuromorphic processing systems.**

***Index Terms*—Hippocampus model, analog memory model, spiking neural network, neuromorphic engineering, DYNAP-SE**

I. Introduction

In recent years, the growing need to process the vasts amounts of data being generated has led to an increasing demand for computing systems with higher energy efficiency and in-memory computing abilities. Faced with these challenges, many research and development efforts have been devoted to finding solutions to these new needs [1]–[3]. As the conventional synchronous logic approach of digital computers leads to relatively high energy consumption requirements [2], [3], different fields have emerged with alternative approaches to the problem. Among all, neuromorphic engineering stands out as a promising field which focuses on implementing brain-inspired systems, with the ability to solve complex sensory-processing problems efficiently [4]–[6].

Here we focus on the original definition of neuromorphic systems [4], which aims to emulate the principle of neural computation using mixed-signal analog/digital electronic circuits. In particular, we investigate the computational properties of Spiking Neural Networks (SNNs) using full-custom neuromorphic processor chips [7]. These are networks of artificial spiking neurons interconnected by synapses. In these networks, information is transmitted among neurons in the form of asynchronous pulses (spikes), signals are represented as mean spike rates, calculated either over time (many spikes) or space (many neurons) [8]. Computation is carried out by creating networks with multiple layers and/or recurrent connections, and by implementing learning algorithms based on local synaptic plasticity mechanisms [9]. This approach has great advantages in terms of energy consumption [10], noise robustness [11]–[13] and real-time operation compared to conventional computing systems [14], [15].

While neurons produce all or none "digital" spikes, their dynamics and computational properties are carried out in the analog domain [16]. There are several advantages of analog computation in neuromorphic systems that arise from the exploitation of physical primitives of the computing substrate [8], [17], [18]. Previous work has shown that analog neuromorphic hardware can improve performance, energy efficiency, and scalability over its digital counterpart in multiple domains [11], [16], [19], [20].

However, analog neuromorphic circuits have the disadvantage that are heterogeneous and noisy, and systems built using this approach are difficult to configure and debug [11], [16]. To overcome these issues we take inspiration from the brain, which faces very similar challenges and solves them efficiently [11], [21].

In computational systems, memory storage and memory recall represent key operations. In the brain, there are different regions involved in learning, storing and processing information, such as external stimuli received by the animal. Among them, the hippocampus stands out. It is the region that works as an autoassociative short-term memory capable of learning and storing a large amount of information from different cortical regions. In addition, it is able to recall the complete information from a fragment of the original [22]. When information enters the hippocampus, it encounters the Dentate Gyrus (DG) brain structure. This region is responsible for increasing the dimensionality of the input data to facilitate its learning and subsequent storage. This reformatted and distributed information arrives at Cornu Ammonis 3 (CA3), a recurrent collateral network where, after a series of oscillations and learning processes, information is stored in the form of memories. Finally, before leaving the hippocampus, this information reaches Cornu Ammonis 1 (CA1), the region responsible for reformatting the information and reducing its dimensionality (close) to its original value. If the information that reaches the hippocampus is a fragment of a previously

D. Casanueva-Morato, A. Ayuso-Martinez, J. P. Dominguez-Morales, and G. Jimenez-Moreno are with the Robotics and Technology of Computers Lab., Universidad de Sevilla, Sevilla, España. G. Indiveri is with the Institute of Neuroinformatics, University of Zurich and ETH Zurich, Zurich, Switzerland. E-mail: dcasanueva@us.es

learned memory, when it reaches CA3, after a series of oscillations, this region is capable of recalling the complete memory and returning it to its output.

Models of hippocampal operation using SNNs have already been proposed in the past. In [23], a spiking hippocampal memory model capable of learning, recalling and forgetting memories was presented. Even though this was a software model, as its principles of operation were based on individual spikes, it was very sensitive to noise. Other spike-based bio-inspired hippocampal memory models, such as [24], based only on the CA3 region of the hippocampus, or [25], [26], with a sequential approach. However, these models have a low storage capacity and cannot work with non-orthogonal patterns (memories whose activity contains activations of neurons in common) [27], [28]. Alternative bio-inspired hippocampal memory models proposed in the literature are not ideally suited to neuromorphic hardware implementations, either because they are quite abstract with no direct spike-based equivalent building blocks [29]–[31], or only follow a hybrid rate-based/spiking approach [32], [33], or are only based on a direct conversion from Artificial Neural Network (ANN) to SNN [34]. In [35], the authors proposed an associative spike-based memory model that sacrifices dynamism when applying pruning techniques after a first learning phase. Finally, in [36], a memory system with an operator-based approach similar to digital gates is proposed.

In short, the characteristics of the hippocampus render it a promising solution to the problem of building efficient information processing and storage electronic systems. A mixed signal analog/digital approach adds more biological plausibility, and has the potential to improve the system's robustness, performance, and energy efficiency compared to a pure digital approach. Taking this into account, in this paper we propose an analog/digital spike-based memory model bio-inspired from the hippocampus able to learning memories, recall them from partial fragments and forget them, and also to work with both orthogonal and non-orthogonal patterns. We made the model robust to noise, overcoming some of the limitations of analogous pure digital models, by taking advantage of the intrinsic features of its analog components. We implemented the spike-based memory model on a full-custom analog hardware platform for emulating neural dynamics.

The rest of the paper is structured as follows: Section II briefly introduces the computational elements and electronic hardware used in this work. In Section III, we describe the proposed model is detailed. The experiments performed to evaluate the functionality and performance of the proposed model are explained in Section IV, along with the results obtained. Then, in Section V, the results of the experiments are discussed. Finally, the conclusions of the paper are presented in Section VI.

The source code used in this work is publicly available, together with the documentation including all the necessary details regarding the SNN architectures.

II. Materials

A. Spiking Neural Networks

Neuromorphic systems typically implement in hardware the third generation of neural networks, SNNs [37]. These networks consist of populations of neurons which process their incoming signals dynamically and produce action potentials (spikes) when their integrated inputs reach a threshold. They transmit their spikes to their target neurons instantaneously, via synapses. SNNs can be very efficient from a computational point of view, as they transmit spikes only when they occur, and they can be configured to carry out complex computations using very sparse activity both in space and time [38].

Each component of the SNN can be implemented using a variety of computational models that approximate the biological behaviour observed in nature. For neurons, the most widely used model is the Leaky Integrate-and-Fire (LIF) model [39], [40]. In this model the sum of the input currents, produced by the neurons synapses that have been stimulated by incoming spikes, drives the neuron's membrane potential. If the total input current is larger than the neuron's "leak" current, then the neuron's membrane potential increases until a threshold is reached (and otherwise the membrane potential leaks back to the neuron's resting state). Once the threshold is reached, the neuron produces a short pulse (a spike) and resets its membrane potential. After generating a spike, the neuron remains in the reset state for a set period of time (the neuron's refractory period), after which it starts integrating its input current again [41].

Synapses are modelled as connections with direction, delay and weight. The delay is the time it takes for the spike to get from the presynaptic neuron to the postsynaptic neuron, the weight denotes the amplitude of the change in the postsynaptic neuron's membrane potential, and the direction determines if the change is positive (for excitatory synapses) or negative (for inhibitory ones).

An important aspect of neural networks is given by the learning rules that govern the process of learning and storing information. While Spike-Timing-Dependent Plasticity (STDP) learning rules have been investigated to a great extent in SNNs [42]–[44], more recent spike-based synaptic plasticity mechanisms that take into account additional factors (such as the neuron's membrane potential or it's recent firing activity) have been shown to be more powerful (see [9] for an overview of rules that are also compatible with neuromorphic hardware).

One of the extensions of the plain STDP rule that can reproduce more accurately experimental data from real synapses, is the triplet STDP rule [45]. While the basic STDP rule only takes into account pairs of presynaptic and postsynaptic spikes to calculate the synaptic weight variation (increasing the weight if the post-synaptic spike is produced after the pre-synaptic one arrives, and decreasing it in the opposite case), the STDP triplet rule considers, in addition, the case of a presynaptic spike followed by a postsynaptic spike and another presynaptic spike, and the case of a postsynaptic spike followed by a presynaptic spike and another postsynaptic spike. The former case results in a decrease in synaptic weight, while the latter case results in an increase in synaptic weight.

This learning rule can reproduce the fine spike-timing behavior of the basic STDP rule for low frequency of input/output spikes, and can also explain and reproduce the rate-based (or correlation-based) Hebbian type rules [46] for high frequency regimes.

B. The DYNAP-SE chip

While there is a wide range of dedicated *digital* neuromorphic processors that can implement SNNs [47]–[49], mixed-signal analog/digital implementations are still an active area of research and can only be found as proof-of-concept prototypes. In this work we use one of such prototypes denoted as "Dynamic neuromorphic asynchronous processor - scalable" (DYNAP-SE) [7].

DYNAP-SE is a hardware platform featuring a scalable multicore architecture with heterogeneous memory structures for dynamic asynchronous event-based processing. It is as a hybrid platform that comprises analog circuits which implement the synaptic and neural dynamics, and asynchronous digital circuits to program the network connectivity and route the spikes among firing neurons. Each DYNAP-SE chip has four cores, and each core has 256 neurons. Each neuron has a fan-in of 64 synapses and a fan-out of 4000 synapses. The chips were manufactured with 0.18 μm 1P6M CMOS technology and comprise hierarchical asynchronous routers as well as integrated full-custom asynchronous SRAM and CAM memory cells distributed among the cores. In this project we used a board containing 4 DYNAP-SE chips. All parameters of the circuits inside each core, such as the leak of the neuron or its refractory period, are shared, therefore they have the same nominal value. However, due to device mismatch, the actual value of each circuit parameter is different. A typical coefficient of variation for these circuit parameters in the DYNAP-SE is approximately 20% [11].

The neuron circuits in the DYNAP-SE implement a model equivalent to the Adaptive-Exponential Integrate and Fire (AdExp-I&F) [5], [50], whose parameters can be configured to behave like LIF neurons. Synapses and biophysically realistic synapse dynamics are implemented using the current-mode Differential Pair Integrator (DPI) log-domain filter [51], which can be configured to give rise to 4 possible synapse types: AMPA (fast, excitatory), NMDA (slow, excitatory), GABA B (subtractive inhibitory) and GABA A (shunting inhibitory).

III. Analog hippocampus computational memory model

A. Architecture

The architecture of the bio-inspired hippocampus memory model proposed in this work is presented in Fig. 1. The block-level design is based on the digital memory model previously proposed in [23]. However, in this work, we propose an analog design, not only at the level of internal implementation of each of the blocks, but also at the level of functionality.

The model does not work with individual spikes, but with spike trains within a time window. Specifically, windows of between 5 and 10 milliseconds (ms) were considered, within which 5 to 20 spikes are expected. This gives a higher tolerance to random noise, which may occur both at the input and within the network. At the same time, this feature makes the network more robust to the small variations that will be encountered as a consequence of the hardware platform used and the analog approach.

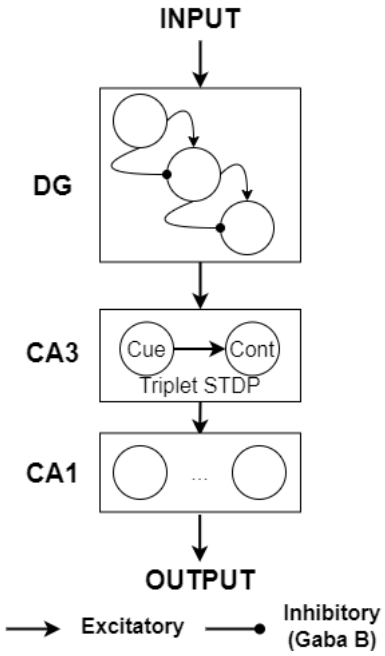


Fig. 1: Architecture of the analog hippocampal computational memory model proposed. The model divides its architecture into 3 blocks: DG, CA3 and CA1. The word Triplet STDP marks those synapses that exhibit the triplet implementation of the STDP learning mechanism.

The design and implementation of each of the components of the proposed model are detailed below: DG, CA3 and CA1.

A.1 Dentate gyrus

DG receives an input memory and is responsible for increasing the dispersion of the information in the memory to facilitate its subsequent learning and storage. In the proposed model, DG acts as a decoder by partially dispersing the memory. The part of the memory that is dispersed is called memory cue, while the rest of the memory content that remains unchanged is called memory content. Specifically, maximum sparsity is applied to the cue, i.e., one-hot encoding or sparse encoding is achieved.

To attain this sparse partial encoding over the memory, the structure presented in Fig. 2 is used. The content of the memory will be passed unchanged directly to the next layer of the model. However, the memory cue will pass through a structure similar to a cascading filter acting as a Winner Take All (WTA) network. In this way, for every possible combination of input neuron activity (minus the absence of activity of all neurons), an output neuron will be activated, generating a spike train and inhibiting the rest. Thanks to this structure, the activity of a set of N input neurons will be mapped onto $2^N - 1$ output neurons. Given N input neurons with binary states (generate spike or not), there are

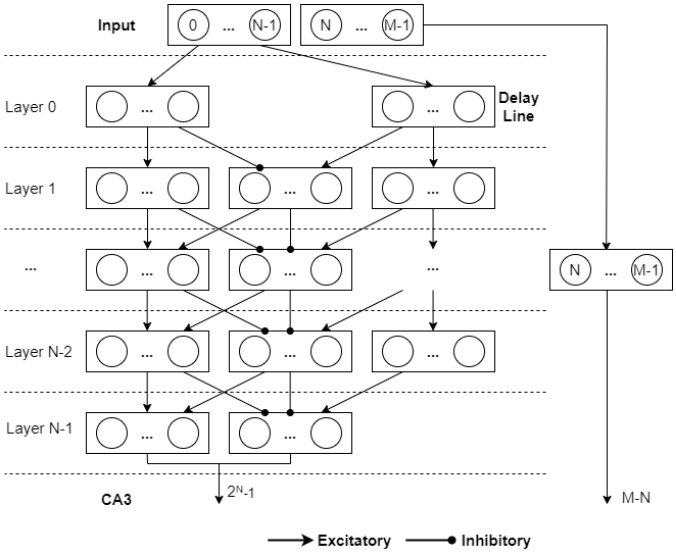


Fig. 2: Architecture of the DG model proposed based on a cascading Winner-Take-All filter network.

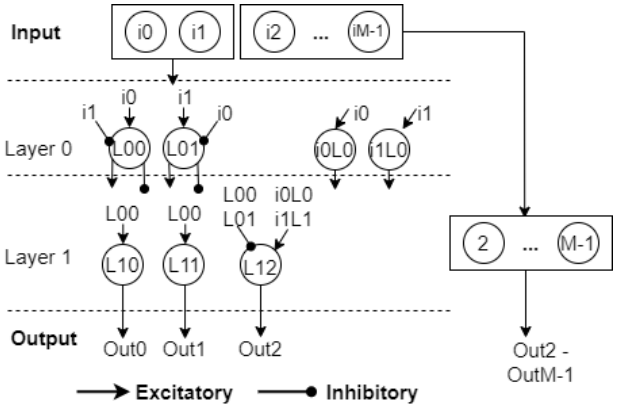


Fig. 3: Example of the network corresponding to a simple DG block receiving an input of M neurons, where only the first 2 neurons (i0 and i1) will be disperse and the remaining ones (i2-iM-1) will pass through unchanged.

2^N combinations of possible input states or $2^N - 1$ input combinations if the one formed by the inactivity of all input neurons is discarded. The latter case is omitted since it is not possible to distinguish whether it is due to an absence of input activity or to input activity defined by the non-activation of all neurons.

At each layer of the cascade filter, three main structures can be distinguished. In the architecture of Fig. 2, there are 3 columns of neurons, starting from right to left:

- On the one hand, there is the Delay Line, a set of populations of neurons that are responsible for transmitting the input activity throughout the network. To this end, in each layer there is a population of the same size as the memory cue, with a 1-to-1 excitatory connection with the same population in the next layer. In the last layer, this population is omitted, since there is no subsequent layer that will need the original input cue information.
- On the other hand, there are populations that compute for each layer a subset of possible combinations of input activities to calculate the output neuron to be activated. This population receives direct information from the input, i.e., it presents excitatory connections from the input in the first layer or from the Delay Line in the remaining layers.
  In layer i, the input activity combinations consisting exclusively of i+1 neurons are determined. Thus, in layer 0, all possible combinations of activation of 1 input neuron are checked, in layer 1 all possible combinations of activation of 2 input neurons are checked, and so on and so forth. Consequently, in layer i, the neurons will have only i+1 excitatory synapses that specify which input combination they will be activated with, and the filter will have as many layers as there are inputs to disperse.
- Finally, it finds the populations that transmit the activity of the output neurons to the next layers and, finally, to the output. It propagates the activity of the output neurons already computed in previous layers through these same populations and, at the same time, receives the result of the new output neurons of the immediately preceding layer. This means that this population increases in number of neurons for each layer it passes through, since for each layer, the subset of output neurons that it must propagate is larger.
  At the synapse level, this operation requires 1-to-1 excitatory synapses from both the output propagation population and the output activity computation population of the immediately preceding layer. However, for everything to work as a WTA network, it is necessary that at the very moment that an output neuron is activated, this activity must inhibit all other output neurons in the network. That is, both the output computation population activity and the output propagation activity of one layer will present inhibitory all-to-all synapses with the output computation neuron populations in the next layer.

Fig. 3 shows the structure that DG would have in a simple case where the memory is made up of M neurons and only the first 2 are used as cue. In this example, DG would take the activity of those two input cue neurons and disperse it into the activity of 3 output neurons, while the remaining M-2 neurons would pass unchanged to the next layer of the model.

A.2 CA3

CA3 is where the learning, storage and recall of the memories that reach the model takes place. To this end, it receives the activity corresponding to the dispersed memory via 1-to-1 excitatory synapses from DG. The set of neurons that receive the information from the cue is called CA3cue, and the set of neurons that receive the information from the content is called CA3cont. The neurons of CA3cue present excitatory synapses with an all-to-all ratio connection with CA3cont.

To achieve memory learning, the STDP triplet learning mechanism is used in the CA3cue-CA3cont synapses. Thanks

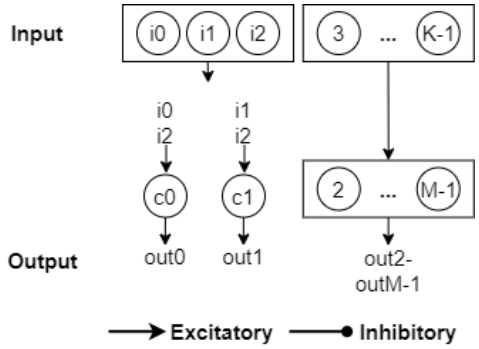


Fig. 4: Example of the network corresponding to a simple CA1 block, which receives an input from 3 neurons and decodes it into the activation of 2 neurons in the output. The rest of the neurons do not perform any operation and, thus, their activity is unchanged.

to this, when the spike trains from the DG reach both populations, a variation of the synaptic weight will be produced in those synapses whose pre- and/or post-synaptic neurons are activated. This variation will determine learning and subsequent recall or forgetting of the input memory.

A.3 CA1

CA1 will act as an encoder to perform the reverse operation to DG and recover the original format of the input information. As only the cue of the memory is affected by the DG information dispersal operation, CA1 will only act on this part of the memory, and the content will pass unchanged through this component.

To achieve this functionality, CA1 will have a structure similar to the DG cascade filter but consisting of a single layer. All those DG neurons that were activated by input neuron i will now participate in the activation of output neuron i of CA1. Thus, given the activation of a cue neuron at the CA1 input, all those neurons that, in combination, activated that neuron in DG will be activated. This combination of dispersed cue neurons activating CA1 output neurons results in excitatory synapses.

Fig. 4 shows the structure that CA1 would have in a simple case where the memory is made up of M neurons, only the first 2 are used as cue and, therefore, CA1 receives the cue dispersed in the activity of 3 neurons. CA1 will decode the activity of these 3 neurons in only 2 of them, to recover the original format of the cue. As the activation of the original cue neuron 0 participated in the activation of the dispersed cue neuron 0 and 2 in DG, this CA1 neuron receives excitatory synapses from the dispersed neurons 0 and 2 from CA3. The same happens for output neuron 1 with the dispersed cue neurons 1 and 2.

A.4 Full network

Given all details about the architecture of the proposed model, Fig. 5 shows an example of a network for a hippocampal memory with a maximum capacity of 3 memories

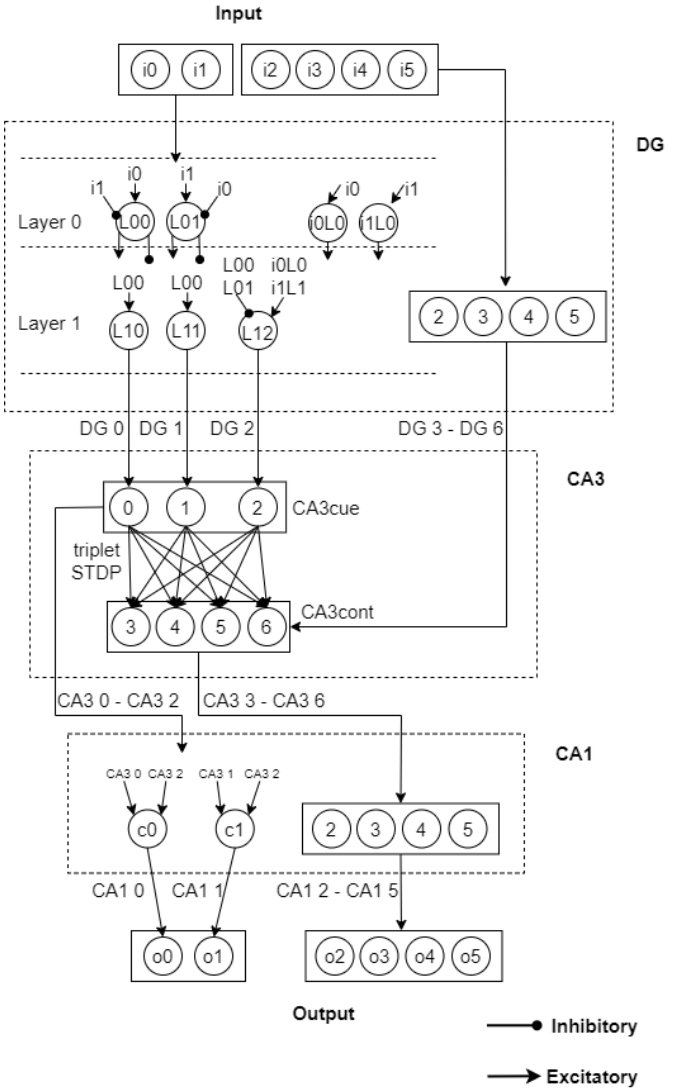


Fig. 5: Example of the network corresponding to a simple hippocampal memory with storage capacity for a maximum of 3 memories at the same time with 6 neurons of activity for each memory. The memory is formed by the activity of 6 neurons: neurons 0 and 1 act as cue and neurons 2, 3, 4 and 5 as content.

at the same time formed by the activity of 6 neurons. Of the 6 neurons forming the memory, the first 2 act as the cue and the last 4 as the content. In DG, the activity of the 2 neurons acting as the cue will be dispersed in 3 neurons, while the activity of the remaining 4 neurons acting as the content will remain unchanged. The output neurons of DG are mapped 1 to 1 onto CA3 neurons in two layers, those receiving the cue and those receiving the content. The first layer is fully connected to the second one and it is, at these synapses, where the STDP triplet learning mechanism is located. Finally, in CA1, the activity of the 3 neurons encoding the cue in CA3 will be recoded in 2 neurons, recovering their original format, while the remaining 4 neurons encoding the content will pass through unchanged.

B. Operating principle

The spiking activity of the model is spatially encoded in time. This encoding is based on the fact that all neural activity within a sub-population in one time window refers to the same memory, while neural activity in the same sub-population but in different time windows refers to different memories.

The proposed model is capable of learning, recalling from a fragment and forgetting memories. These operations are performed on the fly automatically based on the input information to the network, without the need to make changes to its architecture to switch between one operation or another.

B.1 Learning

The learning operation starts with the input of the complete memory to the network via spike trains. This memory will be partially dispersed in DG and reach CA3. In CA3, thanks to the use of the STDP triplet learning mechanism at the synapses connecting CA3cue with CA3cont, learning takes place.

Those synapses that connect cue and content neurons that are activated as part of the memory will have their weight increased. The weight increase of the learning mechanism was configured, as well as the dynamics of the neurons, to work only with spike trains. To achieve this, the variations produced at the spike level are small, but at the spike train level, these small changes are sufficient to learn the input memory. Specifically, the input of 3 separate spike trains is necessary to ensure the correct learning of the complete memory. After passing through CA3, the complete memory will recover its original format when it reaches CA1 and leave the network.

Between each input spike train, a time separation of approximately 100 ms is necessary to prevent the learning rule from mixing the synaptic weight variations of one train with those of the next. If smaller time separations are taken, unwanted synaptic weight decrements may occur and it cannot be ensured that these synaptic weight variations are sufficient to correctly learn the memory. Therefore, the network will take 300 ms to perform a learning operation.

B.2 Recall

The recall operation begins with the input of a previously learned memory fragment (cue) to the network. Upon arrival at CA3, the spike train will activate the corresponding CA3cue neuron and this, in turn, will be transmitted across the different synapses connecting to CA3cont. Only those content neurons whose synapses with the input cue neuron present a sufficiently high synaptic weight will be activated. In other words, only those content neurons that belong to the memory that is characterised by the input cue neuron will be activated. Finally, the activity represented in spike trains of these neurons will pass through CA1 to recover its original format and the complete memory will be obtained at the output of the network.

The time it takes for the network to recall the entire memory since its cue is entered is 25 ms. This will be the time it will take for the spike train to navigate the entire network until it reaches its output after CA1.

B.3 Forget

The forgetting operation does not occur explicitly in the network, but occurs indirectly by attempting to learn a memory whose cue is common to another previously learned memory. When this happens, at the same time that the neural activity of the new memory reaches CA3, the previously learned memory, with which it shares the cue, is recalled. In this case, the network must learn the new memory and forget the previously learned memory.

On the one hand, as DG is constructed, the content activity will arrive to CA3 before the dispersed cue activity. This is because the cue must pass through several layers; for each layer it passes through, the delay of moving from one layer to another is accumulated. On the other hand, the neurons that receive the information about the content of the memory in CA3 are configured in such a way that the increase of potential in the arrival of spikes is lower. This causes the spike train to decrease in frequency and increase its dispersion in CA3cont.

By combining both effects, what is achieved is that, within the working time window of the learning mechanism, the activation of the content neurons of the new memory occurs in the first half and those of the old memory in the second half, both in a distributed manner throughout this time window. These properties exploit the advantages of the STDP triplet implementation by making the synaptic weight variations for synapses involved in the new memory positive and those involved in the old memory negative. In short, there is a decrease in the synapses that store the old memory, causing it to be forgotten, and an increase in the synapses that store the new memory, causing it to be learned.

This operation requires the same execution time as a normal learning operation, i.e., 300 ms.

IV. Experimentation and results

The proposed bio-inspired hippocampal memory model was implemented on the DYNAP-SE hardware platform. This hardware implementation of the model presents a capacity to learn and store up to 7 different memories at the same time, where each memory is defined by the activity of 11 neurons (also called memory size). The model with this capacity presents a network consisting of a total of 56 neurons (30 from DG, 15 from CA3 and 11 from CA1). We used a total of 94 static excitatory and inhibitory synapses (20 from IN-DG, 33 from DG-DG, 15 from DG-CA3, 15 from CA3-CA1 and 11 from CA1-OUT) and 56 dynamic synapses with the STDP triplet learning mechanism (from CA3cue-CA3cont). All inhibitory synapses used are GABA B type.

Although this particular implementation was used, the model was also tested for other network sizes of smaller and larger capacity in terms of both number of memories and memory size. In a generic way, if we have a memory of size M in a memory with a capacity of N memories, the first $\lceil \log_2(N+1) \rceil$ neurons would correspond to the cue ($cueSize$) and the remaining $M - cueSize$ would correspond to the content ($contSize$). Taking these variables into account, the model would have a consumption of $3 * M + 2 * N + cueSize^2 * (cueSize - 2)$ neurons, $4 * M + 2 * N + cueSize *$

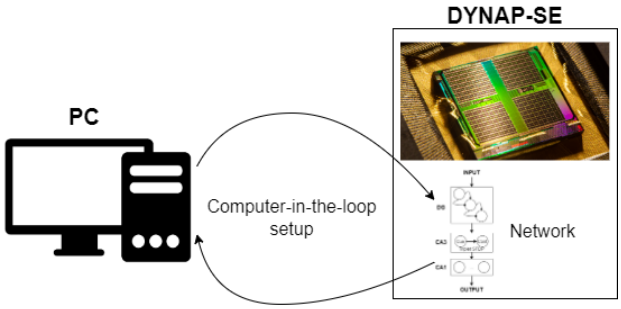


Fig. 6: Computer-in-the-loop hardware setup used in the experiments.

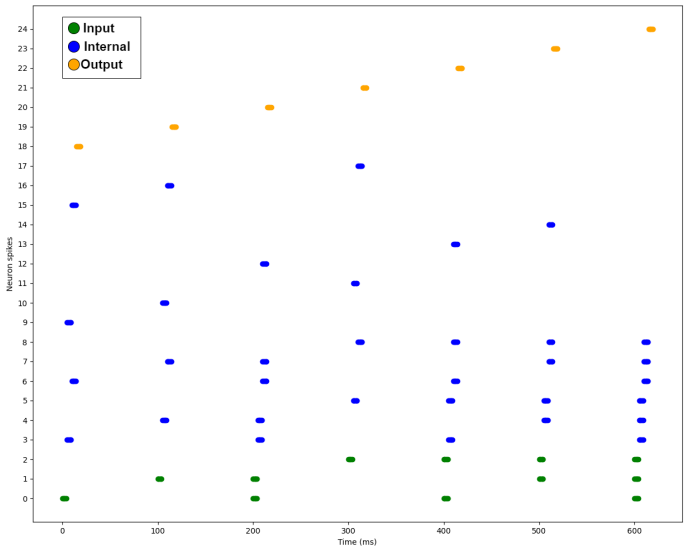
$(cueSize^3 - 2 * cueSize^2 + 3 * cueSize - 6)$ static synapses and $N * contSize$ dynamic synapses with the triplet STDP learning mechanism. Due to the impossibility of regulating the delay at the synapses within the hardware platform used, it is necessary to add additional neurons and synapses in the circuit. Specifically, these exponential terms derive from the propagation of the output sparse activity in DG whose number of neurons, layers and synapses depends on the number of input neurons to be sparsed. If the model is implemented on a different platform that allows controlling the delays of synapses, this resource consumption would be lower, changing from exponential terms to linear ones.

On the hardware implementation of the model, a set of experiments were developed to verify its correct functioning. These experiments consist in stimulating the model by connecting its input to a neural activity generator to observe its behaviour in response to the input of information in the form of memories. For each experiment, a rasterplot is included, which summarises the spiking activity of the network during the experiment from the generated input activity to its output from the network. The X-axis represents the temporal evolution of the experiment in ms and the Y-axis represents each neuron of the network identified by the population that it belongs to and the internal ID within the population. Each point represents a spike fired by the neuron marked by the Y-axis at the time instant marked by the X-axis.

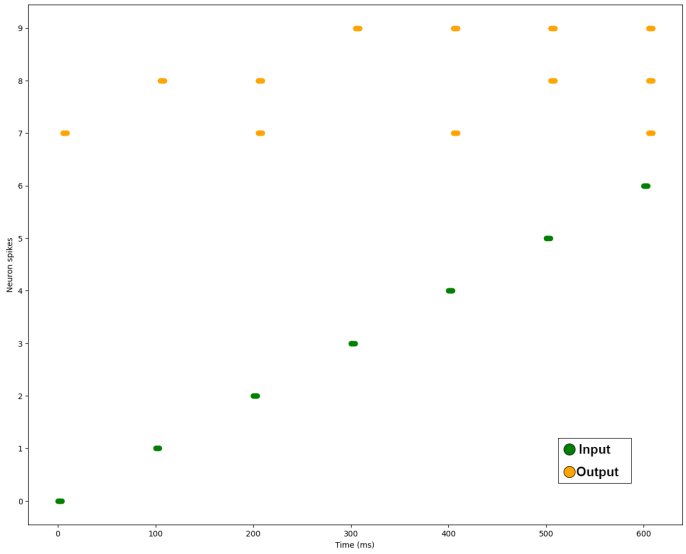
Fig. 6 shows the hardware setup used to carry out the experiments. The DYNAP-SE hardware platform contains the desired network implementation. In addition, a computer-in-the-loop setup is necessary to implement the triplet STDP learning algorithm in DYNAP-SE [52]. DYNAP-SE returns the information from the network in real time to the PC and, based on this information, it calculates the changes of weights in those synapses with the triplet STDP mechanism. This information is also communicated in real time to the network within DYNAP-SE to modify its weights.

A. DG (decoder) and CA1 (encoder)

The first experiments aim to demonstrate the sparse coding achieved in DG together with the decoding and recovery of the original format achieved in CA1. To attain this, both components are stimulated with a sweep of all possible combinations of inputs they can receive. The neural activity resulting from



(a)



(b)

Fig. 7: Raster plot of spiking activity of the (a) DG layer and (b) CA1 layer during a sweep of all possible inputs.

this experiment for DG is shown in Fig. 7a and for CA1 in Fig. 7b. In both cases, only the neural activity of the part of the model that works with the cue is considered; the content part is not of interest for these experiments, as it would simply pass the input activity to the output unchanged.

Starting with the experiment applied to DG, on the one hand, neurons with id from 0 to 2 are in charge of generating the input activity to the network. On the other hand, neurons with id from 3 to 8 are the neurons of the DG Delay Line subpopulations in charge of transmitting the input to the different layers of the model, neurons with id from 9 to 17 are in charge of computing the sparse output and transmitting it to the last layer, and neurons with id from 18 to 24 represent the output activity of the network, that is, the neuronal activity of the sparse coding after propagating to the last layer.

For a decoder with 3 input neurons, a total of 7 possible

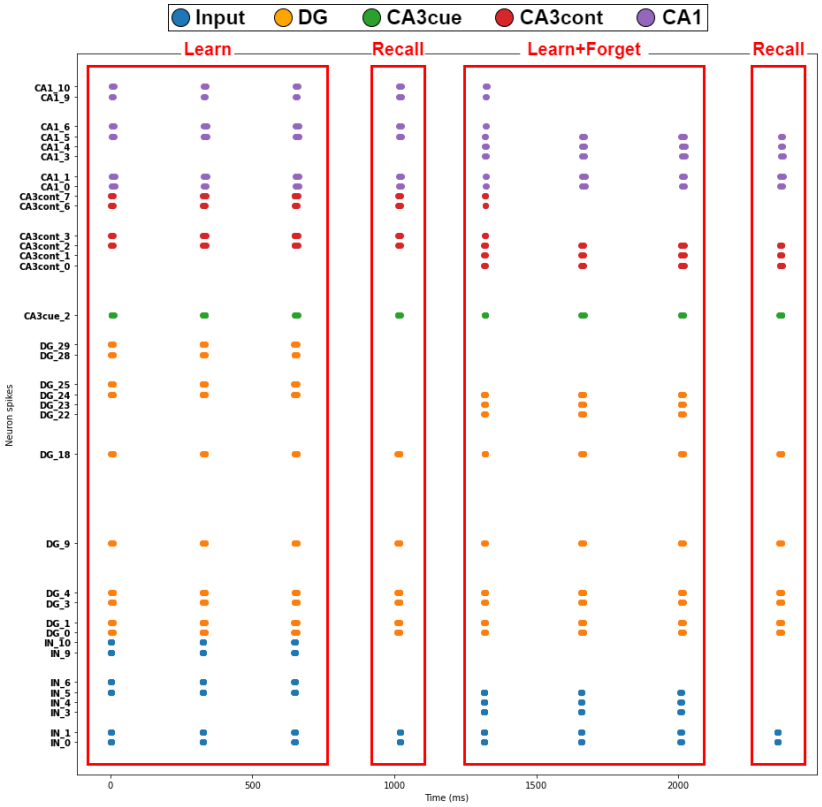


Fig. 8: Raster plot of spiking activity of the network during the operation test consisting of learning, recalling and forgetting operations.

combinations of activity are available, which occur at ms 0, 100, 200, 300, 400, 500 and 600. Each time an input is received, this input pattern is repeated in layer 0 and 1 to be transmitted to layers 1 and 2, respectively. Then, depending on the input combination, the neurons will compute which output neuron should be activated and propagated to the last layer, and the other output neurons will be inhibited, reaching the functionality of a WTA network. For each possible input combination, the activation of only one different output neuron is obtained in each case.

For the experiment applied to CA1, neurons with id from 0 to 6 are in charge of generating the input activity to the network and neurons with id from 7 to 9 are in charge of computing the inputs to generate the output activity. For an encoder with 7 input neurons, there are a total of 7 possible input combinations (the activation of each input separately). These combinations of inputs occur at ms 0, 100, 200, 300, 400, 500 and 600. For each possible input, a different combination of output neurons is obtained, namely, the combination of neuronal activity that originated that sparse activity in DG.

B. Learn, recall and forget

This experiment attempts to demonstrate how the different operations of the model work. To this end, the input neurons to the network are configured in a way that their activity carries out the following sequence of operations: learning, recalling, learning with forgetting and recalling. The resulting network activity for this experiment is shown in Fig 8.

The experiment begins with the learning of a memory, characterised in this case by the activation of neurons IN_0, IN_1, IN_5, IN_6, IN_9 and IN_{10} at ms 0, 350 and 700. Neurons with id IN_0 and IN_1 represent the memory cue and the remaining neurons (IN_5, IN_6, IN_9 and IN_{10}) represent the memory content. The content part will pass unchanged through DG (DG_{24}, DG_{25}, DG_{28} and DG_{29}) and reach the CA3cont subpopulation in CA3 ($CA3cont_2$, $CA3cont_3$, $CA3cont_6$, $CA3cont_7$). The part corresponding to the cue will be dispersed in DG; the result of which propagates to the CA3cue subpopulation of CA3, activating the neuron with id $CA3cue_2$.

At that instant, the spike trains that characterise the cue and the content of the memory will be in CA3, triggering its learning. Finally, this activity reaches CA1, where the content remains unchanged ($CA1_5$, $CA1_6$, $CA1_9$, $CA1_{10}$) and the cue recovers its original format ($CA1_0$ and $CA1_1$).

After finishing the learning operation, it is necessary to verify that the memory has been correctly learned, while also verifying the recall operation. At ms 1050, the recall operation begins with the input of the recall fragment corresponding to the cue (IN_0 and IN_1). After passing through DG, the dispersed cue arrives at CA3cue, activating the neuron with id $CA3cue_2$ and propagating to the CA3cont subpopulation. In CA3cont, the content neurons associated with that cue in the previous learning ($CA3cont_2$, $CA3cont_3$, $CA3cont_6$, $CA3cont_7$) are activated. All this activity will reach CA1, where the complete memory is observed after the recoding of the cue.

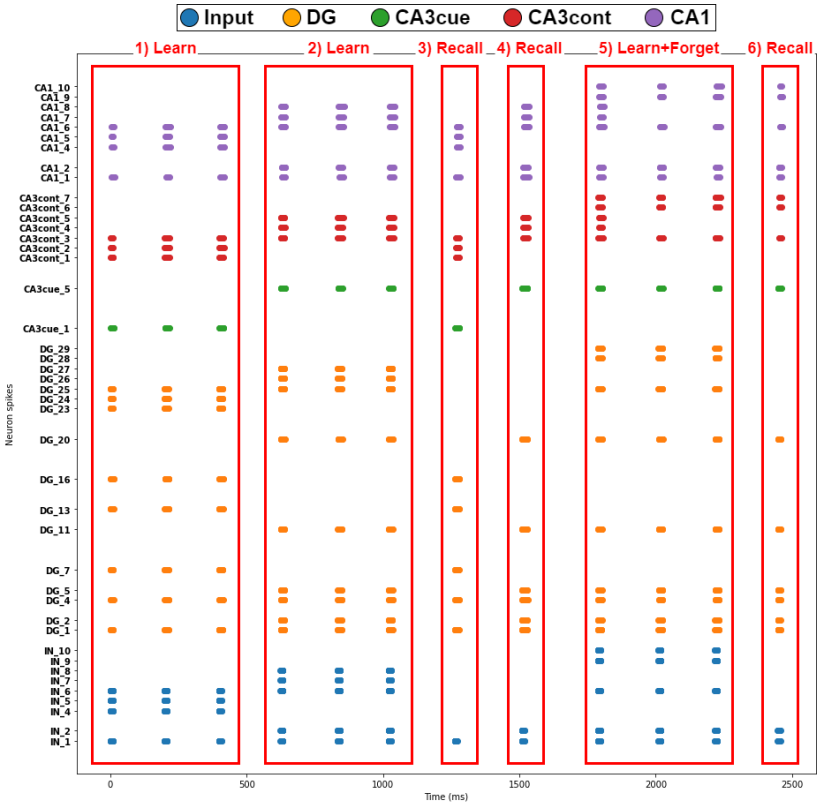


Fig. 9: Raster plot of the spiking activity of the network during the operation test consisting of combinations of several operations.

The following step corresponds to the learning operation of the memory formed by the activation of neurons with id IN_0, IN_1, IN_3, IN_4 and IN_5 at ms 1300, 1650 and 2000. This memory presents the same cue as the previous memory; therefore, upon reaching CA3, all the neurons present in both memories are activated. Due to the sequentiality and distribution of these activations, the first memory will be forgotten in the first input of the new memory and the second memory will be learnt.

To check whether the forgetting had occurred correctly, a final recall operation is carried out. At ms 2350, the cue that is common to both memories (IN_0 and IN_1) is reintroduced to the network and after the operation is completed, only the activation of neurons corresponding to the second memory can be seen in CA1.

C. Combined operations sequence

The last experiment extends the previous one by carrying out a set of 6 operations: 2 learning, 1 learning with forgetting and 3 recallings. The resulting network activity for this experiment is presented in Fig 9. The aim is to demonstrate the robustness of the information learned and verify that, after several operations, regardless of whether or not this information is involved, it is still stored and can be recalled.

Initially, the learning of two memories takes place: the first one characterised by the activation of neurons with id IN_1, IN_4, IN_5 and IN_6, and the second one by neurons with id IN_1, IN_2, IN_6, IN_7 and IN_8. The learning of the first memory occurs at ms 0, 250 and 450, and the learning of the second memory at ms 600, 800 and 1050. Then, at ms 1250 and 1500, the recall of the first and second memory is given, respectively. For this, the memory fragment corresponding to the cue, IN_1 for the first one and IN_1 and IN_2 for the second one, is introduced. In both cases, it can be observed that the complete memory is recalled without problems.

Subsequently, a learning operation of a third memory (IN_1, IN_2, IN_6, IN_9, IN_{10}) is carried out, whose cue is common to that of the second memory. Therefore, at the same time as the third memory is learnt, the second one is forgotten, as can be observed in the result of the recall operation initiated at ms 2450, when the cue common to both (IN_1 and IN_2) is introduced.

V. Discussion

The results of the experiments described in Section IV demonstrate the correct functioning of the analog bio-inspired hippocampus memory model proposed in this work. The first experiment verified the ability of DG to encode the input information to facilitate learning and improve the subsequent storage capacity of memories, as well as the ability of CA1 to decode and recover the original format of this information at its output. The second experiment demonstrated the network's ability to learn, recall and forget memories. Finally, the third experiment verifies the integrity and robustness of the information learned by the model, which is not forgotten or corrupted after several intermediate operations from learning

to recall, or after the recall itself. Furthermore, this set of experiments demonstrated the network's ability to work with both orthogonal and non-orthogonal memories.

The development of the model has been carried out on the DYNAP-SE hardware platform rather than being simulated in software. On the one hand, the deterministic behavior of the digital machines used to run simulators makes it challenging to simulate certain random characteristics of analog systems. These limitations can lead to a deviation of the behavior obtained from that expected in neuromorphic systems. Meanwhile, the hardware platform makes use of the physical analog primitives with which they are built to achieve these analog principles. Therefore, it allows the construction and operation of neuromorphic systems in conditions more similar to those found in the brain. On the other hand, a hardware implementation of the memory model facilitates input/output interfacing with other systems. This enhances its integration and usability in different proposed neuromorphic applications. In addition, a hardware implementation of the model is more efficient than a simulation of it in terms of performance, at the cost of being more complex and time-consuming to develop. Thanks to this, it is possible to achieve real-time operation, which is not possible on a simulator.

The decision to use spike trains instead of individual spikes as the operating principle of the network makes the model more noise tolerant and robust to variations from the expected behaviour. In case of receiving individual unexpected spikes (noise) from different sources, such as mismatch of board parameter values, noise in the input data or externally induced noise within the network, it will be ignored or will have a minimal impact within an activity flow characterised by spike trains.

At the same time, for the learning operation, several short spike trains in small time windows were used over a single spike train with a large time window. If a single spike train was used, the learning mechanism would have to be configured to perform large and fast changes. This would make it unstable and susceptible to individual spike noise. The use of multiple spike trains, on the other hand, allows the learning mechanism to be configured to make this operation more progressive over time. Each spike train will result in small increments in synaptic weights, which in sum will result in a complete learning of the memory. In addition, this ensures that small, unexpected changes in activity and/or noise are not learned by requiring a persistent and long-lasting adaptation over time.

There is biological evidence to support these characteristics. Mammals learn by repetition, with each repetition usually taking small periods of time. In addition, long-term memory formation occurs through the use of constant repetition of the memory over time, while memories that are rarely used are quickly forgotten to make room for new memories [22]. However, these features come with the disadvantage of needing more time to perform the operations correctly. This is more accentuated in the learning and forgetting operations by needing a temporary window of rest between each spike train to avoid mixing operations with each other.

The learning operation in the model has a dual purpose: the ability to learn new memories and to forget old memories. To achieve this functionality, the triplet STDP mechanism was used, which is not only closer to biology than the standard STDP, but is also the most appropriate when working with hippocampal information, as it is able to better extract the dynamics of this type of network [45].

TABLE I: Comparison between the digital ([23]) and analog (proposed in this paper) design of spike-based bio-inspired hippocampal memory models.

	[23]	This work
Paradigm	Digital	Analog
Implementation	Software simulation on hardware platform	Hardware implementation on hardware platform
Resource usage	Linear as a function of capacity	Linear as a function of capacity *
Noise effect	Highly sensitive to noise	Noise resistant and robust to punctual variations
Learning rule	STDP	Triplet STDP
Temporal performance (learn and recall)	7-6 ms	300-25 ms
Storage capability	High	High
Type of patterns	Orthogonal and non-orthogonal	Orthogonal and non-orthogonal
Functionality	Learn, recall and forget	Learn, recall and forget
Content persistence	Leaks unused content after some operations	Content without leakage
Neural network	SNN	SNN
Bio-inspiration	Medium	High

* Ignoring the restrictions of the hardware platform used (more information in Section IV).

Another highlight is the design of DG as a multi-layer structure that functions as a cascaded filter. If a single-layer structure were used to compute all outputs at the same time, as a consequence of the use of spike trains, the output would be noisy. This would be due to the time it takes for the inhibitory synapses that connect the neurons to each other, thus, once an output is reached, it inhibits the others, resulting in a WTA selection. In the case of a single layer, this inhibition would be late and would result in the activation of neurons of unwanted outputs. At the same time, the use of inhibitory GABA B synapses was necessary in the implementation. Thanks to their temporal behaviour and intensity, this type of inhibitory synapse was better suited to the network's input activity flow and to the proposed architecture itself.

Comparing the model described in this paper with the proposals of other authors, we are going to focus on those articles mentioned in Section I. The proposed model is able to work with both orthogonal and non-orthogonal patterns and presents a great capacity for learning and configurable storage, unlike [24]–[28]. [32] and [33] propose hybrid models between SNN and ANN, [34] perform a direct conversion from ANN to SNN, and [30] and [31] are not directly spike-based, while the proposed model is purely spike-based, making use of SNNs and taking all the advantages that this type of network offers.

The proposed model provides fully dynamic and real-time operation, whereas [35] sacrifices these characteristics by using offline pruning techniques. This work presents an analog implementation based on spike trains, and thus more robust to noise and anomalous variations in the information or the network, while works such as [23] and [36] present implementations in digital simulation platforms of SNNs and with an operation based on individual spikes, i.e., sensitive to any type of noise. Specifically, this model presents a functional analog alternative to the proposal given in [23]. A more detailed comparison is shown in Table I.

VI. Conclusions

In this work, a fully functional analog spike-based memory model bio-inspired by the hippocampus was proposed. This model was not only simulated in software, but also implemented with SNNs on the DYNAP-SE hardware platform. Through a set of experiments, the ability of the memory model to learn memories, recall them from a fragment of themselves and forget them was demonstrated. All these operations are performed automatically, depending on the input information, in a single simulation for each experiment, without interruptions or changes in the network between different operations. Although a concrete implementation of the model was shown, its parameterised design enables the implementation of memories with greater or lesser capacity.

The result of the experiments led to the discussion of their similarities with its biological counterpart. The design of the model based on DG, CA3 and CA1, the use of the triplet STDP learning mechanism in CA3 for learning, the cascade filter structure of DG and the functioning as an autoassociative memory, among other features, makes the proposed model resemble the biological model. Furthermore, a comparison of the proposed model with other computational models found in the literature was carried out. This work presents the first hardware implementation on a special-purpose hardware platform for SNNs of a fully functional spike-based bio-inspired hippocampal analog memory model, paving the road for the development of future more complex neuromorphic systems.

Nevertheless, the memory model still has room for improvement and extensions that remain for future work. Currently, the dispersion achieved in DG is the same as that of a deterministic WTA system. A possible extension would be to model DG as another type of information dispersion structure present in the literature and compare both designs in terms of time, resource consumption and robustness. This model could be applied within neuromorphic robotic navigation and control systems as a memory system capable of learning and recalling sequences of movements, the navigation environment, trajectories toward a goal position, etc. Furthermore, it would be interesting to carry out a detailed study of the tolerance and robustness of the model to constant and even periodic random noise and compare it to its spike-based but digital counterparts under the same conditions.

The source code of the implemented model and the experiments and simulations performed is available on an open-source GitHub repository[1].

Acknowledgments

This work is part of the project SANEVEC TED2021-130825B-I00, funded by the Ministerio de Ciencia e Innovación (MCIN), Agencia Estatal de Investigación (AEI) of Spain, MCIN/AEI/10.13039/501100011033, and by the European Union NextGenerationEU/PRTR, and was partially supported by project PID2019-105556GB-C33 funded by MCIN/AEI /10.13039/501100011033. D. C.-M. was supported by a "Formación de Profesorado Universitario" Scholarship and by "Ayudas complementarias de movilidad" from the Spanish Ministry of Education, Culture and Sport.

D. Casanueva-Morato would like to thank Giacomo Indiveri and his group for hosting him during a three-month internship between 1st June 2023 and 31th August 2023, during which the idea of this paper was originated and most of the results presented in this work were obtained.

[1]https://github.com/dancasmor/A-bio-inspired-hardware-implementation-of-a-analog-spike-based-hippocampus-memory-model

References

[1] A. Vanarse, A. Osseiran, and A. Rassau, "Neuromorphic engineering—a paradigm shift for future im technologies," *IEEE Instrumentation & Measurement Magazine*, vol. 22, no. 2, pp. 4–9, 2019.

[2] S. Soman and M. Suri, "Recent trends in neuromorphic engineering," *Big Data Analytics*, vol. 1, pp. 1–19, 2016.

[3] F. Zenke, S. M. Bohté, C. Clopath, I. M. Comşa, J. Göltz, W. Maass, T. Masquelier, R. Naud, E. O. Neftci, M. A. Petrovici *et al.*, "Visualizing a joint future of neuroscience and neuromorphic engineering," *Neuron*, vol. 109, no. 4, pp. 571–575, 2021.

[4] C. Mead, "Neuromorphic engineering: In memory of misha mahowald," *Neural Computation*, vol. 35, pp. 343–383, 2023.

[5] E. Chicca, F. Stefanini, C. Bartolozzi, and G. Indiveri, "Neuromorphic electronic circuits for building autonomous cognitive systems," *Proceedings of the IEEE*, vol. 102, no. 9, pp. 1367–1388, Sep. 2014.

[6] Z. Sun, V. Cutsuridis *et al.*, "Brain simulation and spiking neural networks," *Cognitive Computation*, pp. 1–3, 2023.

[7] S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri, "A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (dynaps)," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 1, pp. 106–122, 2018.

[8] G. Indiveri and Y. Sandamirskaya, "The importance of space and time for signal processing in neuromorphic agents," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 16–28, 2019.

[9] L. Khacef, P. Klein, M. Cartiglia, A. Rubino, G. Indiveri, and E. Chicca, "Spike-based local synaptic plasticity: a survey of computational models and neuromorphic circuits," *Neuromorphic Computing and Engineering*, vol. 3, no. 4, p. 042001, Nov. 2023. [Online]. Available: http://dx.doi.org/10.1088/2634-4386/ad05da

[10] S. Kim, S. Park *et al.*, "Spiking-yolo: spiking neural network for energy-efficient object detection," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 270–11 277.

[11] D. Zendrikov, S. Solinas, and G. Indiveri, "Brain-inspired methods for achieving robust computation in heterogeneous mixed-signal neuromorphic processing systems," *Neuromorphic Computing and Engineering*, vol. 3, no. 3, p. 034002, 2023.

[12] S. Kundu, M. Pedram, and P. A. Beerel, "Hire-snn: Harnessing the inherent robustness of energy-efficient deep spiking neural networks by training with crafted input noise," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5209–5218.

[13] W. Guo, M. E. Fouda *et al.*, "Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems," *Frontiers in Neuroscience*, vol. 15, p. 638474, 2021.

[14] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[15] J. Zhu, T. Zhang, Y. Yang, and R. Huang, "A comprehensive review on emerging artificial neuromorphic devices," *Applied Physics Reviews*, vol. 7, no. 1, p. 011312, 2020.

[16] D. Ivanov, A. Chezhegov, M. Kiselev, A. Grunin, and D. Larionov, "Neuromorphic artificial intelligence systems," *Frontiers in Neuroscience*, vol. 16, p. 1513, 2022.

[17] R. Sarpeshkar, "Analog versus digital: extrapolating from electronics to neurobiology," *Neural computation*, vol. 10, no. 7, pp. 1601–1638, 1998.

[18] G. Indiveri, B. Linares-Barranco, T. J. Hamilton *et al.*, "Neuromorphic silicon neuron circuits," *Frontiers in neuroscience*, vol. 5, p. 73, 2011.

[19] I. Kataeva, S. Ohtsuka, H. Nili, H. Kim, Y. Isobe, K. Yako, and D. Strukov, "Towards the development of analog neuromorphic chip prototype with 2.4 m integrated memristors," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2019, pp. 1–5.

[20] B. Cramer, S. Billaudelle, S. Kanya, A. Leibfried, A. Grübl, V. Karasenko, C. Pehle, K. Schreiber, Y. Stradmann, J. Weis *et al.*, "Surrogate gradients for analog neuromorphic computing," *Proceedings of the National Academy of Sciences*, vol. 119, no. 4, p. e2109194119, 2022.

[21] A. Mehonic and A. J. Kenyon, "Brain-inspired computing needs a master plan," *Nature*, vol. 604, no. 7905, pp. 255–260, 2022.
[22] E. T. Rolls, *Brain computations: what and how*. Oxford University Press, USA, 2021.
[23] D. Casanueva-Morato, A. Ayuso-Martinez, J. P. Dominguez-Morales, A. Jimenez-Fernandez, and G. Jimenez-Moreno, "A bio-inspired implementation of a sparse-learning spike-based hippocampus memory model," *arXiv preprint arXiv:2206.04924*, 2022.
[24] ——, "Spike-based computational models of bio-inspired memories in the hippocampal ca3 region on spinnaker," in *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2022, pp. 1–9.
[25] C. H. Tan, E. Y. Cheu *et al.*, "Associative memory model of hippocampus ca3 using spike response neurons," in *International conference on neural information processing*. Springer, 2011, pp. 493–500.
[26] C. H. Tan, H. Tang, K. C. Tan, and M. Yuan, "A hippocampus ca3 spiking neural network model for storage and retrieval of sequential memory," in *2013 IEEE Conference on Cybernetics and Intelligent Systems (CIS)*. IEEE, 2013, pp. 134–139.
[27] R. Mueller and A. V. Herz, "Content-addressable memory with spiking neurons," *Physical Review E*, vol. 59, no. 3, p. 3330, 1999.
[28] M. Matsugu and A. L. Yuille, "Spatiotemporal information storage in a content addressable memory using realistic neurons," *Neural Networks*, vol. 7, no. 3, pp. 419–439, 1994.
[29] A. S. Shiva and A. Hussain, "Continuous time recurrent neural network model of recurrent collaterals in the hippocampus ca3 region," in *International Conference on Brain Inspired Cognitive Systems*. Springer, 2016, pp. 343–354.
[30] M. K. Benna and S. Fusi, "Place cells may simply be memory cells: Memory compression leads to spatial tuning and history dependence," *Proceedings of the National Academy of Sciences*, vol. 118, no. 51, p. e2018422118, 2021.
[31] G. Ma, R. Jiang, L. Wang, and H. Tang, "Dual memory model for experience-once task-incremental lifelong learning," *Neural Networks*, vol. 166, pp. 174–187, 2023.
[32] T. Zhang, Y. Zeng *et al.*, "Hmsnn: hippocampus inspired memory spiking neural network," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2016, pp. 2301–2306.
[33] Y. Zhang, Y. Chen, J. Zhang, X. Luo, M. Zhang, H. Qu, and Z. Yi, "Minicolumn-based episodic memory model with spiking neurons, dendrites and delays," *IEEE transactions on neural networks and learning systems*, no. 99, pp. 1–15, 2022.
[34] Y. Yue, M. Baltes, N. Abujahar, T. Sun, C. D. Smith, T. Bihl, and J. Liu, "Hybrid spiking neural network fine-tuning for hippocampus segmentation," *arXiv preprint arXiv:2302.07328*, 2023.
[35] H. He, Y. Shang, X. Yang *et al.*, "Constructing an associative memory system using spiking neural network," *Frontiers in neuroscience*, vol. 13, p. 650, 2019.
[36] A. Ayuso-Martinez, D. Casanueva-Morato, J. P. Dominguez-Morales, A. Jimenez-Fernandez, and G. Jimenez-Moreno, "Construction of a spike-based memory using neural-like logic gates based on spiking neural networks on spinnaker," *IEEE Transactions on Emerging Topics in Computing*, 2023.
[37] W. Maass and C. Bishop, *Pulsed Neural Networks*. MIT Press, 1998.
[38] J. L. Lobo, J. Del Ser, A. Bifet, and N. Kasabov, "Spiking neural networks and online learning: An overview and perspectives," *Neural Networks*, vol. 121, pp. 88–100, 2020.
[39] L. Abbott, "Lapicque's introduction of the integrate-and-fire model neuron (1907)," *Brain Research Bulletin*, vol. 50, pp. 303–304, 1999.
[40] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh *et al.*, "Deep learning in spiking neural networks," *Neural networks*, vol. 111, pp. 47–63, 2019.
[41] R. B. Stein, "A theoretical analysis of neuronal variability," *Biophysical Journal*, vol. 5, no. 2, pp. 173–194, 1965.
[42] H. Markram, W. Gerstner, and P. Sjöström, "Spike-timing-dependent plasticity: a comprehensive overview," *Frontiers in Synaptic Neuroscience*, vol. 4, no. 2, pp. 1–3, 2012.
[43] L. Abbott and S. Nelson, "Synaptic plasticity: taming the beast," *Nature Neuroscience*, vol. 3, pp. 1178–1183, Nov. 2000.
[44] J. Sjöström, W. Gerstner *et al.*, "Spike-timing dependent plasticity," *Spike-timing dependent plasticity*, vol. 35, no. 0, pp. 0–0, 2010.
[45] J.-P. Pfister and W. Gerstner, "Triplets of spikes in a model of spike timing-dependent plasticity," *Journal of Neuroscience*, vol. 26, no. 38, pp. 9673–9682, 2006. [Online]. Available: https://www.jneurosci.org/content/26/38/9673
[46] R. Kempter, W. Gerstner, and J. L. Van Hemmen, "Hebbian learning and spiking neurons," *Physical Review E*, vol. 59, no. 4, pp. 4498–4514, 1999.
[47] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The SpiNNaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.
[48] M. Davies, N. Srinivasa, T.-H. Lin *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *Ieee Micro*, vol. 38, no. 1, pp. 82–99, 2018.
[49] P. A. Merolla, J. V. Arthur, Alvarez-Icaza *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.
[50] R. Brette and W. Gerstner, "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity," *Journal of neurophysiology*, vol. 94, no. 5, pp. 3637–3642, 2005.
[51] C. Bartolozzi and G. Indiveri, "Synaptic dynamics in analog vlsi," *Neural computation*, vol. 19, no. 10, pp. 2581–2603, 2007.
[52] J. Zhao, M. Monforte, G. Indiveri, C. Bartolozzi, and E. Donati, "Learning inverse kinematics using neural computational primitives on neuromorphic hardware," *npj Robotics*, vol. 1, no. 1, p. 1, 2023.

Daniel Casanueva-Morato received the B.S. degree in computer engineering from University of Seville, Spain, in 2020 and the M.S. degree in computer engineering and networks from University of Granada, Spain, in 2021. Currently, he is a Ph.D. student in the Departmenf of Computer Architecture and Technology at the University of Seville, supported by a "Formación de Personal Universitario" Scholarship from the Spanish Ministry of Education, Culture and Sport. His research focuses on computational neuromorphic architectures for modeling brain regions. His research interests include neuromorphic engineering, spiking neural networks, brain-computer interfaces and embedded systems.

Alvaro Ayuso-Martinez received the B.S. degree in computer engineering from University of Seville, Spain, in 2020 and the M.S. degree in computer engineering and networks from University of Granada, Spain, in 2021. Currently, he is a PhD student in the Robotics and Technology of Computers Lab. in the Department of Computer Architecture and Technology at Universidad de Sevilla. His research focuses on neuromorphic engineering, more specifically on spiking neural network architectures, being part of the MIND-ROB research project. His research interests include neuromorphic engineering, spiking neural networks, real-time audio processing and embedded systems.

Juan P. Dominguez-Morales received the B.S. degree in computer engineering, the M.S. degree in computer engineering and networks, and the Ph.D. degree in computer engineering (specializing in neuromorphic audio processing and spiking neural networks) from the University of Seville, in 2014, 2015 and 2018, respectively. His Ph.D. was granted with a research grant from the Spanish Ministry of Education and Science. Since February 2023, he has been working as Associate Professor in the same university. His research interests include neuromorphic engineering, spiking neural networks, audio processing and deep learning.

Gabriel Jimenez-Moreno received the M.S. Degree in Physics (electronics) and the Ph.D. Degree from the University of Seville (Seville, Spain), in 1987 and 1992, respectively. He was granted a Fellowship from the Spanish Science and Technology Commission (CICYT). Currently, he is a Full Professor at the University of Seville. From 1996 until 1998, he was Vice-Dean of the E.T.S. Ingenieria Informatica, University of Seville, where he participated in the creation of the Department of Computer Architecture. He is the author of several papers on robotics, neuromorphic engineering, and computer architecture, and has also directed many research projects on neuromorphic systems. His research interests include neural networks, vision processing systems, embedded systems, computer interfaces, and computer architectures.

Giacomo Indiveri obtained an M.Sc. degree in electrical engineering in 1992 and a Ph.D. degree in computer science in 2004 from the University of Genoa, Italy. Engineer by training, Indiveri has also expertise in neuroscience, computer science, and machine learning. He has been combining these disciplines by studying natural and artificial intelligence in neural processing systems and in neuromorphic cognitive agents. His group uses these neuromorphic circuits to validate brain inspired computational paradigms in real-world scenarios, and to develop a new generation of fault-tolerant event-based neuromorphic computing technologies. Indiveri is senior member of the IEEE society, and a recipient of the 2021 IEEE Biomedical Circuits and Systems Best Paper Award. He is also an ERC fellow, recipient of three European Research Council grants.