SoftwareX 22 (2023) 101349

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

NESIM-RT: A real-time distributed spiking neural network simulator

Daniel J. Rosa-Gallardo [a], Juan Carlos de la Torre [a], Fernando M. Quintana [a], Juan P. Dominguez-Morales [b], Fernando Perez-Peña [a,*]

[a] School of Engineering, Universidad de Cádiz, Spain
[b] Robotics and Technology of Computers Lab. Universidad de Sevilla, Spain

ARTICLE INFO

Article history:
Received 9 December 2022
Received in revised form 16 February 2023
Accepted 28 February 2023

Keywords:
Neuromorphic engineering
Spiking neural networks
Simulation

ABSTRACT

The neuromorphic engineering field aims to mimic complex biological structures by means of mathematical models, implementing them in either analog or digital circuits. These models include the dynamic characteristics of biological neurons and synapses, which are interconnected creating spiking neural networks. These are first simulated using software frameworks in order to verify the expected behavior of the implemented model. In this work, a novel software, called NESIM-RT, for simulating spiking neural networks in real time is presented. A friendly user interface allows users to design and modify the network, together with visualising its output in real time. A novel implementation of the MDHCP protocol is proposed to support online changes in the network together with allowing the simulation in a many-cores distributed architecture. The performance of the proposed software is compared with other widely-used simulators in the neuromorphic engineering field, highlighting the advantages in terms of latencies and network scalability. NESIM-RT also allows exporting the SNN model directly to SpiNNaker, enabling an immediate transition between software simulation and hardware implementation.

Code metadata

Current code version	v1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-22-00398
Permanent link to Reproducible Capsule	https://github.com/ferper/nesimRT/releases/tag/v1.0
Legal Code License	GPL-3.0
Code versioning system used	git
Software code languages, tools, and services used	C++, Qt.
Compilation requirements, operating environments & dependencies	QtCreator, Qt5.15, Linux-based OS
If available Link to developer documentation/manual	https://github.com/ferper/nesimRT/tree/main/docs
Support email for questions	fernandoperez.pena@uca.es

1. Motivation and significance

The neuromorphic engineering community has evolved to a prospective where companies such as Intel are now involved with its Loihi chip [1]. The movement is towards a new approach which is focused on improving the computing capabilities of any device using the benefits of a neuromorphic approach. Since the core of this neuromorphic approach are the spiking neural networks, simulators of Spiking Neural Networks (SNNs) are playing a major role. Simulators offer the possibility of testing and tuning the performance of the architecture designed before allocating the network in a hardware platform.

In this manuscript, a new software simulator called NESIM-RT is presented. There are already many simulators available and all of them have pros and cons compared to the tool presented in this paper. The comparison with all of them is made considering only those that are open-source.

The users of these simulators can be early stage researchers that do not have a deep knowledge of the field. Thus, having a Graphical User Interface (GUI) could help users on the design of SNNs. Only Nengo [2] and Real-Time Analysis and Visualization

* Corresponding author.
E-mail address: fernandoperez.pena@uca.es (Fernando Perez-Peña).

https://doi.org/10.1016/j.softx.2023.101349

Table 1
Comparison between different SNN simulators.

	GUI	Live output	Programming skills	Open source
Nengo	Yes	Yes	Yes	Yes
NEST	No	No	Yes	Yes
Brian 2	No	No	Yes	Yes
RAVSim	Yes	Yes	No	Yes
NEVESIM	No	No	Yes	Yes
SPIKE	No	No	Yes	Yes
NESIM-RT	Yes	Yes	No	Yes

Simulator (RAVSim) [3] include a GUI that allows run-time interaction of the user. The former allows all kind of modifications and the latter is mostly thought for understanding how the network is affected when some parameters are modified.

Besides, the user might want to use a specific neuron model or designed a new one. Brian2 [4], Nengo, Neural EVEnt-based SIMulator (NEVESIM) [5], NEST [6] and SPIKE [7] allow a user extension of the neuron and/or synaptic models. However, in the case of SPIKE, the extension is complex since the user needs knowledge of C++ and CUDA to define the models. Also, in the case of NEVESIM, to define new models, some knowledge of the framework core (implemented in C++) is needed (not needed to include additional network elements). The rest of the tools allow Python scripting to extend the models.

In terms of code generation, using Nengo's unique frontend, is an advantage since it allows different backends such as Graphics Processing Units (GPUs), Field Programmable Gate Arrays (FPGAs) or Loihi.

Related to the timing features of the simulators, the integration of Brian2 with GeNN [8] provides a performance boost in processing time thanks to the GPUs support without changing the programming language (Python). SPIKE also uses GPUs to run the simulations. On the contrary, NEVESIM does not take advantage from the GPUs since its main purpose is to be able to scale up the network by using a distributed architecture of CPUs.

Table 1 presents a brief comparison between different SNN software simulators.

The novelty of the tool presented is the combination of all of these features within the same software simulator: a run-time user interaction, a GUI that allows the user build a network with no programming skills at all, real-time performance, user defined equations for the neurons and synapses, code generation for Brian2 and SpiNNaker and the possibility of scale up the network by using a distributed architecture of PCs exploding the features of the User Datagram Protocol (UDP) protocol: fast and not guarantee develop. The former allows real-time and the latter emulates the stochastic biological firing pattern of a neural network.

NESIM-RT can be downloaded from: https://github.com/ferper/nesimRT.

2. Software description

2.1. Software architecture

NESIM-RT has been implemented by means of Qt as a cross-platform IDE. This framework, which is based on C++, consists of two main parts: the compiler, and the GUI (QtCreator). For developing NESIM-RT, QtCreator 4.7.1 has been used, together with the 5.11.2 64-bit version of the compiler. It has been developed in 64-bits Ubuntu 16.04.3 LTS with GNU GCC version 5.4.0.

The communication between the different objects in NESIM-RT has been done using the Signals & Slots mechanism, which is one of the main features of Qt. This allows communicating any objects of any type between them in a way that the information is processed as soon as it arrives. Mutex have also been used in order to control the access of the different NESIM-RT modules to the shared memory.

NESIM-RT uses multicast addressing and sockets to communicate its different elements. The biological behavior of neurons and synapses have been mimicked in the best way possible by means of multicast packets transferred using the UDP protocol. This protocol does not guarantee all the spikes generated to reach their destination, which is expected in the biological behavior. Moreover, UDP allows transmitting information without establishing a connection between neurons (no need to wait for an acknowledge response when sending a packet), which allows transmitting packets at a very high rate.

A novel protocol called Multicast DHCP (MDHCP) has been designed, which allows creating new synapses in real time without the need of stopping the simulation. Moreover, this protocol provides the execution of the simulation in a distributed architecture with several devices at the same time.

In order to establish the communication between all the neurons in the network, the concept of Mother Neuron (MN) is introduced. Each neuron in the network sends a request (REQ) in promiscuous mode, either if the neuron is local or remote. MN receives this request and posts an acknowledge that has the multicast address and the identifier of the neuron that it belongs to. Therefore, allowing the neuron to be discovered inside the network, which can now interact with it in real time. When a new synapse is created, neurons that take part in the projection instance the Synapse class, which is responsible for establishing the connection between the neurons and for dealing with the current value.

2.2. Software functionalities

NESIM-RT consists of an intuitive GUI that allows users to create, modify and simulate SNNs in real time. Therefore, the time that NESIM-RT takes to simulate the network is the same as the time that the user wants to simulate. This is specially useful, since most SNN simulators take much more time to simulate the network than the simulation time itself. This, together with allowing to add synapses, neurons and changing any parameter in real time, makes the proposed simulator avoid long waiting times in the simulation and to have more real-time functionalities than others.

NESIM-RT allows configuring the neuron model completely and adapting it to the user's needs. All of this can be done inside its GUI, which was developed with simplicity in mind. Users can create neurons and synapses from the GUI in a simple way.

The topology of the network can be scaled up, using a distributed network of different connected computers to simulate specific parts of the SNN, whose outputs are sent in real time to the central node, which receives the packets and processes them in order to plot the information accordingly. This functionality is one of the main advantages of NESIM-RT when compared to other alternatives.

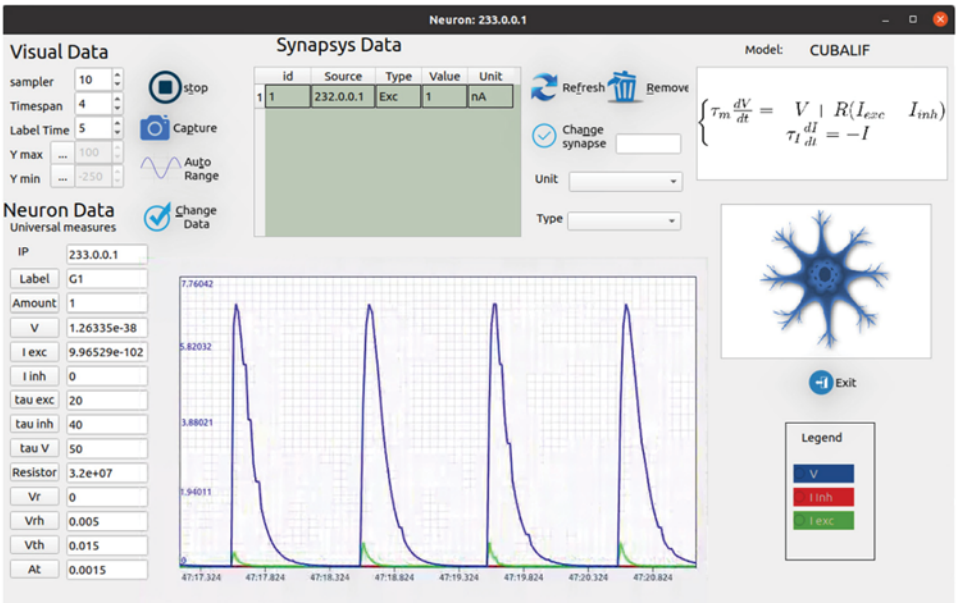

Fig. 1. Screenshot of the software simulator. In the central part, the traces of the membrane potential (in blue) and the excitatory current (in green) are shown. On the left hand side, the visualisation parameters and neuron data are shown. In the upper part, the synapses data can be seen. On the top right corner, the equation of the neuron is displayed. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

NESIM-RT also helps to reduce the learning curve in the use of the SpiNNaker hardware platform, since it offers the user the option of exporting the implemented neural network to a Python script that can be directly executed in this platform. To the best of the authors' knowledge, this is the first time that the use of SpiNNaker is open to researchers who lack sufficient programming skills to design the network.

3. Illustrative example

A test of NESIM-RT was recorded[1] to illustrate a real time example, where a network is created consisting of a generator that emits a spike every second and a neuron connected to the generator via an excitatory and self-inhibitory synapse. The video shows how both the network architecture and the neural parameters can be modified in real time, as well as its visualisation. A new neuron connected both to the generator and the other neuron with excitatory synapses is created and the membrane potentials of both neurons can be visualised on the general monitor. Furthermore, Fig. 1 shows a screenshot of the simulator where all parameters can be edited.

To validate the MDHCP protocol, 3 different computers with the same characteristics were used; HP Z820 Workstation, Intel Xeon E5-2620 2 × 2.10 GHz, 32 GB of RAM, NVIDIA Tesla K20c GPU and Ubuntu 16.04.3 LTS 64 bits. One of the machines runs NESIM-RT and neurons are progressively created on all of them. These machines, with 1 GB interfaces, are interconnected by an Extreme V300 8P-2T-W switch with CAT6 cables. Each experiment was performed 10 times for each machine. It consisted in creating neurons in a distributed way and taking average times both for the creation (running) and for their representation in the graphical part of NESIM-RT. Table 2 lists all the values used to carry out the experiments together with the results obtained. First, the number of neurons created is 1 per machine (a total of 3 neurons), then we increased the number of neurons up to 5 for a total of 15 neurons. We continued with 10 neurons, making a total of 30 neurons, and we ended up with 100 neurons per machine, for a total of 300 neurons. This number of neurons and hardware devices is just an example to check the performance of the simulator. NESIM-RT has no limit for the number of neurons and computers used. The number of neurons that a single computer would be able to simulate only depends on its available resources in terms of hardware–software.

Table 2
Results of the MDHCP tests. Each row of the table shows the number of neurons running in each device and the average time taken for the system to have all the neurons active (t_act), as well as the average time to incorporate their behavior into the visualiser (t_vis). The average times correspond to 10 runs of each test.

PC_1	PC_2	PC_3	t_act (ms)	t_vis (ms)
1	1	1	200	250
5	5	5	350	403
10	10	10	510	572
100	100	100	7308	7418

4. Impact

We expect that NESIM-RT will be widely used in the neuromorphic field, both for academic and scientific purposes, since it allows the design and simulation of custom SNN models in real time by means of a guided user-friendly GUI that does not require any prior programming knowledge.

The neuromorphic community can benefit from this software tool for many different applications: from the need for tuning specific neuron parameters, up to performing real-time distributed computing simulations where large SNN models are considered.

NESIM-RT also facilitates researchers the use of Brian2 and sPyNNaker, since it includes a mechanism to export the networks designed to the Python code used in those libraries. Therefore, the users can eventually implement their networks in hardware platforms such as SpiNNaker. NESIS-RT is shipped completely open-source using GPLv3 license, together with a descriptive user manual containing all the different tools that the software consists of.

[1] https://youtu.be/joPfjm98TAs

5. Conclusions

In this work, a novel real-time distributed spiking neural network simulator has been presented, called NESIM-RT. The developed tool overcomes particular limitations of other state-of-the-art simulators and provides a user-friendly, open-source and real-time solution that emulates the stochastic biological firing pattern of a neural network. A novel protocol, called Multicast DHCP, was designed ad-hoc in order to allow the dynamic generation of synapses during simulation time. Moreover the possibility of a distributed execution allows scaling up the network topology. NESIM-RT's user-friendly GUI facilitates the complete configuration of the neuron model and allows users with limited programming skills to use SpiNNaker and Brian2, since it allows directly exporting the models designed from the simulator to these platforms.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Link to the code shared on the paper

Acknowledgments

Fernando M. Quintana and Juan Carlos de la Torre are funded by the Spanish *Ministerio de Ciencia, Innovación y Universidades* under the FPU grants FPU18/04321 and FPU17/00563. This work is part of the project Nemovision (PID2019-109465RB-I00, MICIU/AEI /10.13039/5011000110), and the support from the European Regional development Fund, OPTIMALE (FEDER-UCA18-108393) and MIND-ROB (PID2019-105556GB-C33), together with the Andalusian Regional Project PAIDI2020 GENIUS (P18-FR-2399). Besides, this publication is part of the project TED2021-131880B-I00 supported by the Spanish Ministry (with support from the European Union "NextGenerationEU"). All authors want to thank Juan A. Herrera Rodriguez for his support to the manuscript.

References

[1] Davies M, Srinivasa N, Lin T-H, Chinya G, Cao Y, Choday SH, et al. Loihi: A neuromorphic manycore processor with on-chip learning. IEEE Micro 2018;38(1):82–99.
[2] Bekolay T, Bergstra J, Hunsberger E, DeWolf T, Stewart TC, Rasmussen D, et al. Nengo: A Python tool for building large-scale functional brain models. Front Neuroinform 2014;7:48.
[3] Koravuna S, Rückert U, Jungeblut T, et al. SNNs model analyzing and visualizing experimentation using RAVSim. In: International conference on engineering applications of neural networks. Springer; 2022, p. 40–51.
[4] Stimberg M, Brette R, Goodman DF. Brian 2, an intuitive and efficient neural simulator. In: Skinner FK, editor. ELife 2019;8:e47314.
[5] Pecevski D, Kappel D, Jonke Z. NEVESIM: Event-driven neural simulation framework with a Python interface. Front Neuroinform 2014;8:70.
[6] Gewaltig M-O, Diesmann M. Nest (neural simulation tool). Scholarpedia 2007;2(4):1430.
[7] Ahmad N, Isbister JB, Smithe TSC, Stringer SM. Spike: A GPU optimised spiking neural network simulator. 2018, p. 461160, BioRxiv.
[8] Stimberg M, Goodman DF, Nowotny T. Brian2GeNN: Accelerating spiking neural network simulations with graphics hardware. Sci Rep 2020;10(1):1–12.