Neurocomputing 502 (2022) 57–70

Contents lists available at ScienceDirect

Neurocomputing

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

Neuromorphic adaptive spiking CPG towards bio-inspired locomotion

Pablo Lopez-Osorio [a], Alberto Patiño-Saucedo [b], Juan P. Dominguez-Morales [c], Horacio Rostro-Gonzalez [b,*], Fernando Perez-Peña [a,*]

[a] School of Engineering, Universidad de Cádiz, Spain
[b] Department of Electronics, DICIS-University of Guanajuato, Mexico
[c] Robotics and Technology of Computers Lab., Universidad de Sevilla, Spain

ARTICLE INFO

Article history:
Received 10 May 2021
Revised 22 June 2022
Accepted 27 June 2022
Available online 30 June 2022
Communicated by Zidong Wang

Keywords:
Neurorobotics
SpiNNaker
Central pattern generator
Spiking neural network
Neuromorphic hardware
Adaptive-learning

ABSTRACT

In recent years, locomotion mechanisms exhibited by vertebrate animals have been the inspiration for the improvement in the performance of robotic systems. These mechanisms include the adaptability of their locomotion to any change registered in the environment through their biological sensors. In this regard, we aim to replicate such kind of adaptability through a sCPG. This sCPG generates different locomotion (rhythmic) patterns which are driven by an external stimulus, that is, the output of a FSR sensor to provide feedback. The sCPG consists of a network of five populations of LIF neurons designed with a specific topology in such a way that the rhythmic patterns can be generated and driven by the aforementioned external stimulus. Therefore, eventually, the locomotion of an end robotic platform could be adapted to the terrain by using any sensor as input. The sCPG with adaptation has been numerically validated at software and hardware level, using the Brian 2 simulator and the SpiNNaker neuromorphic platform for the latest. In particular, our experiments clearly show an adaptation in the oscillation frequencies between the spikes produced in the populations of the sCPG while the input stimulus varies. To validate the robustness and adaptability of the sCPG, we have performed several tests by variating the output of the sensor. These experiments were carried out in Brian 2 and SpiNNaker; both implementations showed a similar behavior with a Pearson correlation coefficient of 0.905.

1. Introduction

It is well known that, in biology, rhythmic locomotion is produced by a neural structure called Central Pattern Generator (CPG) [1]. This structure is located at the spinal cord and it usually comprises of two neural populations which produce an alternating output of spikes. Eventually, these output spikes are used to activate the muscles fibers.

This approach of using CPGs can be borrowed to create locomotion in robotics. There are several possibilities to implement a CPG: using coupled-oscillators, using Artificial Neural Networks (ANNs) or using Spiking Neural Networks (SNNs). The closest biological implementation is to use an SNN. These networks are based on biological plausible neuron models and synaptic connections that implement biological features. The field of research called neuromorphic engineering aims to implement these networks on electronics, mimicking the way living beings have solved complex problems by using both analog and digital circuits. The neuromorphic robotics field puts together both the neuromorphic engineering and the roboticist communities [2]. The use of neural structures made of spiking neurons, coming from the neuromorphic engineering field, within robotics results in the need for less resources, less power consumption and a simplification of the algorithms [3] in comparison with traditional approaches based on ANN or coupled oscillators.

One of these neural structures is the CPG. This structure generates a rhythmic pattern at its output, which can be used within robotics to generate locomotion. Thus, a CPG creates gaits that are suitable to use within a robotic platform. These structures can generate a very stable pattern even without sensory information or brain activity [4].

As briefly shown, there is a growing community of researchers that are exploring the possibility of using spiking neurons within a CPG to create locomotion in robots [1]. Although there are some works that introduced local sensory feedback as [5,6], they used oscillators to model the behaviour of the CPG instead of spiking neuron models. Conversely, most of the previous works, spike-

* Corresponding authors.
E-mail addresses: pablo.osorio@uca.es (P. Lopez-Osorio), alpatisa@hotmail.com (A. Patiño-Saucedo), jpdominguez@atc.us.es (J.P. Dominguez-Morales), hrostrog@ugto.mx (H. Rostro-Gonzalez), fernandoperez.pena@uca.es (F. Perez-Peña).

https://doi.org/10.1016/j.neucom.2022.06.085
0925-2312/

based related, in the literature present an open-loop CPG which does not include any sensory information: in [7], the actuation of a lamprey-like robot is done by using an open-loop CPG and neuromorphic hardware. Then, in [8,9], the authors proposed a CPG implemented on an Field Programmable Gate Array (FPGA) and the SpiNNaker platform [10]; these three works do not offer the possibility of changing the originally produced pattern in real time. However, they showed that implementing CPGs using spiking neurons uses less power. There is a more recent paper which allows both real time functioning and pattern variation but without including any sensory information [11]. In [12], the open-loop CPG is implemented on Loihi [13] using an astrocytic network, producing two different gaits with 24 motor neurons. In, [14], authors propose the implementation of a CPG with the possibility of changing the amplitude, frequency and phase online without any sensory input required. The authors also pointed out that the architecture should include any sensory feedback to modify the behavior of the CPG. Regarding works that include sensory information, in [15], the CPG is built using coupled-oscillators instead of spiking neurons and the feedback to the CPG is included into the control loop of the equations of the CPG. In [16], 12 simulated neurons modulated by sensory feedback are used to build the CPG. Instead of what we propose, a SNN with a adaptation, they achieve different gaits by either moving the location or increasing the number of neural structures. Instead of what we propose, a SNN with adaptation, the authors achieve different gaits by either moving the location or increasing the number of neural structures. The neuron model used in that paper is Izhikevich [17] instead of the Leaky Integrate-and-Fire (LIF) model proposed in this work to reduce the computational resources used. The sensory information is used to adapt the time duration of each phase of the frequency switching to enable the actuators to reach the commanded position. Finally, in [18], a neuromorphic sensor has been used to select which predefined gait of the CPG should be activated. A very recent work by Strohmer et al. [19] suggests the use of a combination of non-spiking and spiking neurons to simulate a closed loop amplitude regulating network. The non-spiking neuron used in such work is in fact a spiking neuron (Leaky Integrate-and-Fire) with a threshold high enough to avoid firing. The spiking neuron model used is the adaptive exponential integrate-and-fire AdEx, a more complex model than the one used in this work and not easily implementable on neuromorphic hardware. The article focuses on investigating how combining spiking and non-spiking neurons can create a network of sensorimotor neurons capable of shaping the output of the network based on the analogue input. Although locomotion is very vaguely mentioned as an application of such a network, it is not an issue addressed in this paper. The authors also suggest the compatibility of their model with neuromorphic hardware. However, this compatibility is restricted only to the non-spiking neuron and to a cloud service called CloudBrain. In this regard, our proposal differs in different aspects, the first of them is the exclusive use of spiking neurons with a simpler model than the one presented in [19]. The second difference and contribution in this work is the use of information from a real sensor (Force Sensitive Resistor (FSR)) that feeds the network, which in the case of the locomotion of a legged robot will allow us to modify its dynamics depending on the terrain where the robot moves. The third and most important aspect is the validation of our model on neuromorphic hardware, specifically on the widely used SpiNNaker neuromorphic processor, using high level software tools that facilitate the reproducibility of this work by the scientific community. Finally, in this work we have carried out successful physical tests by injecting the force generated by the FSR sensor into different surfaces on one leg of the robot. This experiment can be replicated on the other legs of the robot and, together with a suitable interconnection of these, can generate the desired locomotion. In most of these works, the use of an external input to the CPG changes the performance of it. This performance can be defined as a neuromodulation. It has been shown that this modulation is essential to alter the behavior of a neural structure by modifying the synaptic connections [20].

Finally, there are works where the main focus is on the learning process of the robotic platform: in [21], the authors proposed a rewarding-learning process to teach a hexapod robot how to walk without any previous knowledge. A couple of sensors (a standard camera and a gyroscope) are used to provide the rewarding signal to the neural network based on a CPG. Although they used a digital version model neuron of the LIF, they do not use neuromorphic hardware to implemented the neural architecture; a Raspberry Pi is used instead. A similar approach based on reinforcement learning, but without using spiking neurons, was proposed to improve the locomotion of the NAO robot [22]. Another approach is used in [23], where the authors have two hexapod robots: an expert and a student. The student learns or imitates the gait of the expert by using a one-layer feedforward SNN and a Dynamic Vision Sensor (DVS) camera as input. However, the possibility for the robot to adapt to its environment is not implemented.

To summarize, the objective of this paper is to design and deploy a spiking architecture that makes possible the interaction of a spiking CPG with its environment. An external agent, i.e. a FSR, is introduced as the feedback stimulus to the network. This agent can modify the gait generated by the CPG. Therefore, the locomotion of a robotic platform (any legged robot) could be adapted to the terrain. Furthermore, the spiking network presented in this paper allows the introduction of the feedback sensory information on the loop of the CPG to provide adaptation. This adaptation SNN could be used with any sensor as input stimulus.

The rest of the paper is structured as follows: Section 2.1 introduces the materials used in this work, including the simulator and hardware used. The implemented methods are described in 2.2, together with the SNN model. Then, the results obtained are presented and discussed in Section 3. This section is divided into two different subsections: first, the experiments run using the software simulator and then, the same experiments run on the hardware platform. Finally, the conclusions are presented.

2. Materials and methods

2.1. Materials

This section describes both the software and hardware used to perform the experiments.

2.1.1. Brian 2

Brian 2 [24] is a neural simulator for SNNs written in Python programming language. Thus, it is a cross-platform which is available in different operating systems. In contrast to other SNN simulators such as NEURON [25] or PyNN [26], Brian 2 is highly flexible and it is easily adaptable with new non-standard neuron models and synapses. Brian 2 can be used to model and simulate complex problems faced by neuroscientists, as well as giving faster and more robust results before implementing the solution on a hardware platform.

2.1.2. SpiNNaker

SpiNNaker [27,10,28] is a massively parallel, multi-core computing system designed by the Advanced Processor Technologies (APT) Research Group from the University of Manchester. It was designed under the Human Brain Project (HBP) [29] for simulating parts of the brain by using SNNs. SpiNNaker machines consist of SpiNNaker chips, which have eighteen 200-Hz ARM968 processor

cores each [30]. This allows an asynchronous communication infrastructure for sending short packages (each of them representing a particular neuron firing) [31] identified using Address Event Representation (AER) [32]. Different SpiNNaker machines were built and commercialized, including SpiNN-3 and SpiNN-5, with 4 and 48 SpiNNaker chips each, respectively. They also include the spinnlinks [33], which allow real-time input/output interfacing with neuromorphic sensors and other neuromorphic platforms such as FPGAs [34–36]. A PyNN-based [26] software package called sPyNNaker [37] can be used to design and implement SNNs on these machines. The recently built million-core machine is at the School of Computer Science at the University of Manchester that can be used through the HBP portal. In this work, a SpiNN-5 machine was used to run the simulations proposed.

2.2. Methodology

We simulated our Spiking Central Pattern Generator (sCPG) model on a standard computer using the Brian 2 Simulator to characterize the network dynamics, to analyse the number of neurons per population needed and to adjust the network parameters. The simulation results guided the subsequent neuromorphic implementation using the SpiNNaker platform.

2.2.1. Neuron model

The LIF model is used to implement the neuron on both the software simulator and the SpiNNaker hardware platform [38]. The model is defined within the Eqs. (1) and (2).

$$\tau_m \frac{dV}{dt} = -(V - V_r) + RI(t) \tag{1}$$

$$if\ V(t) = V_{th} \quad then \quad \lim_{\delta \to 0; \delta > 0} V(t + \delta) = V_r \tag{2}$$

where V is the membrane potential of the neuron, R represents the resistance of the membrane, τ_m the time constant of the neuron, V_r the resting potential, V_{th} the threshold and $I(t)$ is the stimulus.

2.2.2. Network model

The SNN model depicted in Fig. 1 was designed based on the neuron model presented in Section 2.2.1. The main objective of the proposed model is to generate a constant oscillation between the spikes produced in ensembles A and B, whose frequency varies depending on the value read from the FSR sensor. Thus, the proposed CPG is able to automatically adapt its behavior depending on the input stimulus. The SNN shown in Fig. 1 is the core of the sCPG. The output spikes of populations A and B will be eventually used to interface the actuation structure of the legs of the robotic platform.

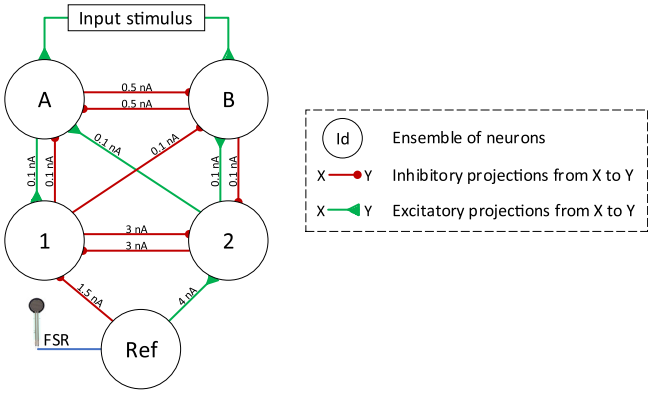

Fig. 1. Diagram of the proposed spiking neural network architecture. Populations A and B are self-excited and self-inhibited with a probability of 25% and 75% and weights of 4 nA and 1.5 nA, respectively. A and B are reciprocally inhibited with a probability of 75%, and are both injected with a constant external current in order to start generating the oscillation. Populations 1 and 2 are similar to A and B, and are connected to them in a way that greater spiking rates in the Ref population (higher values from the output of the FSR sensor) will produce higher oscillation frequencies in AB, whereas lower oscillation frequencies will be obtained for lower FSR values.

The SNN consists of different parts which are described next. It is important to note that each of the ensembles (also called populations) have the same number of neurons. The number of neurons in each population was set based on different experiments, which are presented in Section 3.

The main block of the architecture is the so-called CPG_{AB}, which consists of the populations A and B represented in Fig. 1. These populations are self-excited and self-inhibited with a probability of 25% and 75% and weights of 4 nA and 1.5 nA, respectively. Moreover, a 75% probability of having inhibitory synapses between neurons from the two aforementioned populations is also present. Thus, when one of the populations is producing spikes, the opposite is inhibited and, thus, generating the desired oscillation pattern. Populations A and B are injected with a constant external current in order to start generating the oscillation. Therefore, for these two populations Eq. 3 is used instead of Eq. 1.

$$\frac{dV}{dt} = \frac{V_r - V + R(I_{exc} - I_{inh} + I_{st})}{\tau_m} \tag{3}$$

Where I_{st} is the current injected to neurons in populations A and B. This value was set to 2.2 nA, which is sufficient for producing the desired rhythmic pattern.

Furthermore, populations 1 and 2 (CPG_{12}) both have the same number of neurons as those in CPG_{AB}, and are also interconnected in the same way. The projections between CPG_{12} and CPG_{AB} are depicted in Fig. 1, and follow the same aforementioned probabilities (25% for excitatory and 75% for inhibitory projections). The

Table 1
Neuron parameters for the proposed CPG in both the Brian 2 simulator and the SpiNNaker hardware platform.

Parameter	Value
u_{reset}	−55.0 mV
u_{rest}	−55.0 mV
u_{th}	15.0 mV
τ_m	6.0 ms
τ_{syn_e}	5.0 ms
τ_{syn_i}	8.75 ms
c_m	0.1875 nF
Δ_t	1.0 ms
I_{bias}	2.2 mA

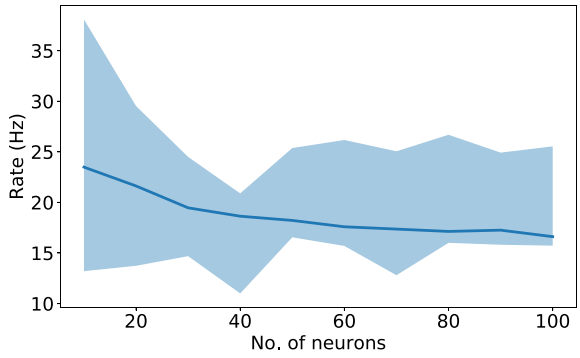

Fig. 2. Average, maximum and minimum rate values obtained for each number of neurons per population. The trace shows the mean, and, its shadow, the maximum and minimum rate of all the one thousand simulations performed.

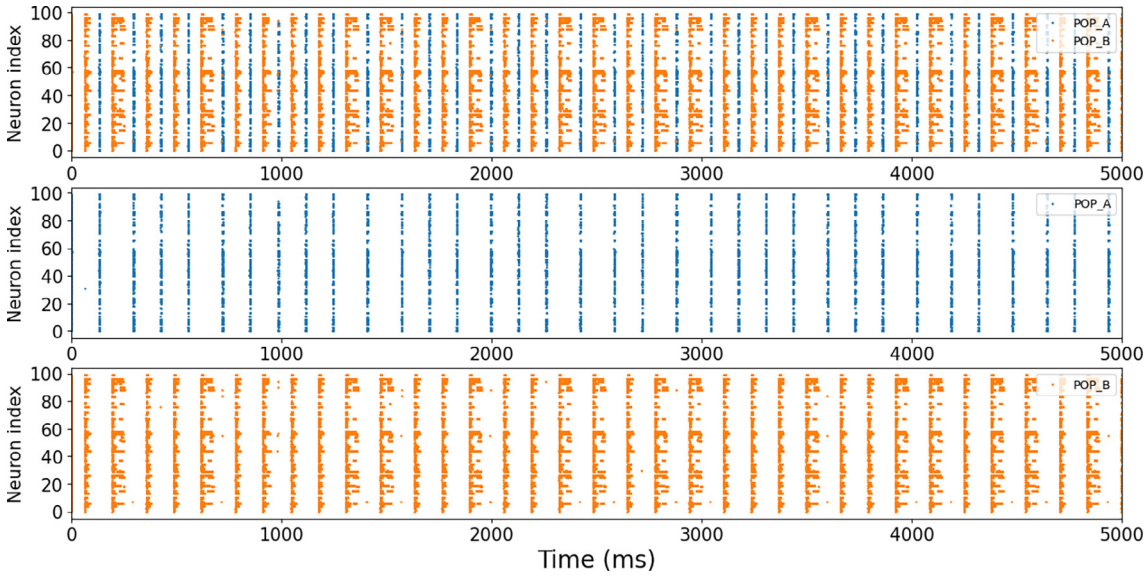

Fig. 3. Simulation of the CPG_{AB} with a I_{St} value of 10 nA. Increased oscillation frequency may be observed along with an increase in the amount of noise when compared to Fig. 4.

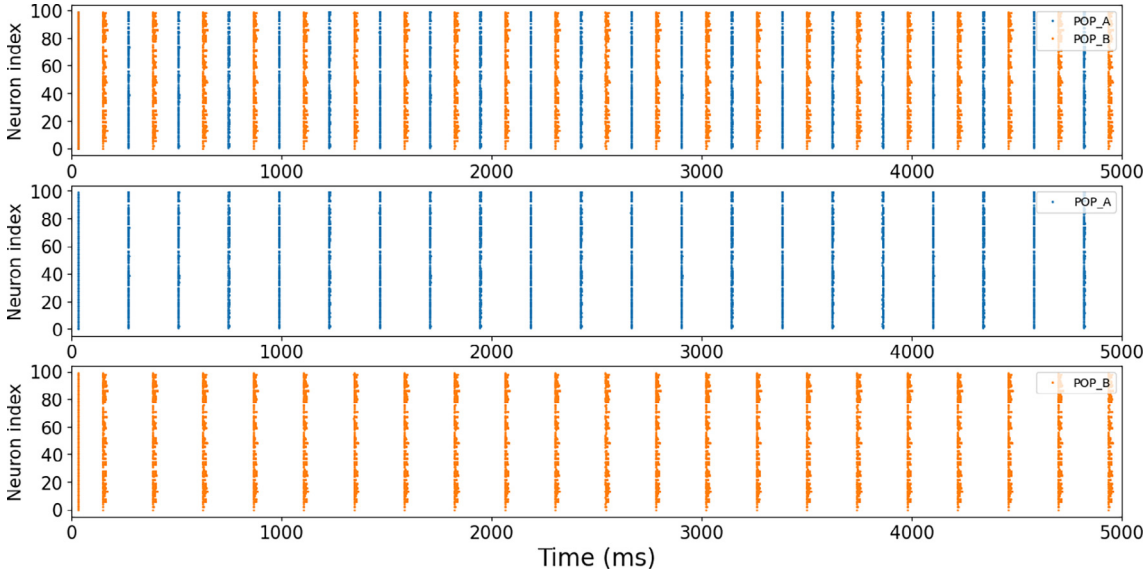

Fig. 4. Brian 2 simulation of the CPG_{AB} with a I_{St} value of 2.2 nA. A lower oscillation frequency and almost total noise removal can be observed when compared to Fig. 3.

weights between the different populations of the proposed model are specified in the same figure.

Finally, the reference population (*Ref*) was implemented as a Poisson distribution with variable frequency. In contrast to the populations from CPG_{AB} and CPG_{12}, the number of neurons in *Ref* was set to 50. These neurons are connected to populations 1 and 2 following the scheme presented in Fig. 1. This number of neurons was set to 50 since it was the optimum for producing biologically-plausible spiking rates in the population, with maximum and minimum frequencies close to the biological counterpart [39]. The spiking rate of the Poisson distribution depends on the values obtained from the FSR sensor used as input to population *Ref*. This sensor should be placed at the end of the leg of the robot. This sensor provides values between 0 and 5 V. Since the spinal cord ventral horn motor neuron alpha, which is the biological neuron taken as reference, has a spike rate between 10 and 171 Hz [39], a linear regression was established in order to match the frequency of the Poisson distribution with the voltage value read from the sensor.

Based on these three blocks (CPG_{AB}, CPG_{12} and *Ref*) and the connections among them, the proposed behavior explained at the beginning of this section was achieved. Therefore, following Fig. 1, different scenarios can be analyzed. In the case where the spike rate of population *Ref* is greater than the oscillation frequency of CPG_{AB}, population 1 will be inhibited and population 2 will be excited. Since population 1 is excited from A, but the number of spikes is lower than the ones that are inhibiting the same population from *Ref*, population 1 will have very low activity. The opposite occurs in population 2, which will be inhibited from B but excited from *Ref* in a stronger way and, thus, having considerable more activity than population 1. The activity from population 2 will excite CPG_{AB}, increasing its oscillation frequency. Conversely, the opposite happens when the spike rate of *Ref* is

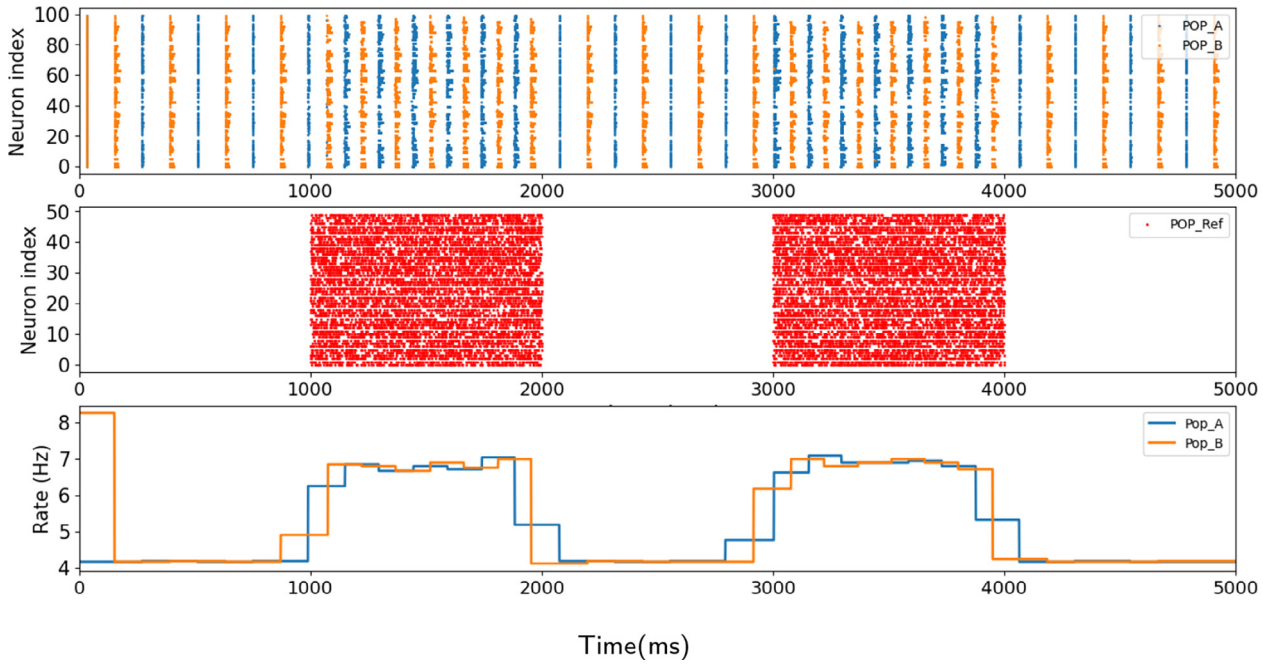

Fig. 5. Brian 2 simulation of the proposed SNN model when having a 5 V input from the FSR sensor for 1000 ms between $t = 1000$ ms and $t = 2000$ ms and between $t = 3000$ ms and $t = 4000$ ms (generating a Poisson distribution of 171 Hz in *Ref*). For the rest of the simulation time, the output of the sensor was set to 0 V. The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

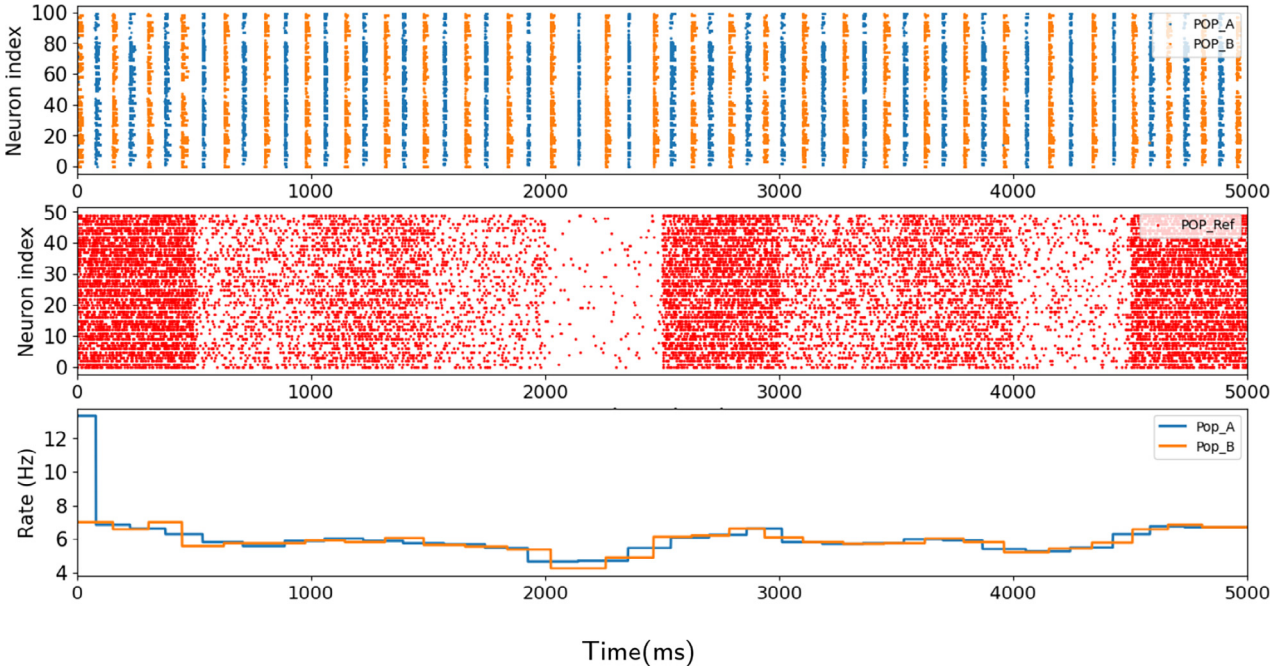

Fig. 6. Brian 2 results obtained when simulating random voltage readings from the FSR sensor (random values change every 500 ms). The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

lower than the oscillation frequency of CPG_{AB}, where population 1 will have more activity than 2 and, therefore, inhibit CPG_{AB} in order to reduce its frequency. As a summary, CPG_{AB} is the main one that will eventually drive the motors. Then, CPG_{12} is used to interface the feedback sensor. The latter would adapt its frequency to the feedback sensor output and then, this frequency would be transmitted to the former. Therefore, the dynamical properties of the feedback sensor are not directly in contact with the main locomotion generator. As a result, the proposed network is able to adapt the frequency of the CPG based on an input stimulus. This model was simulated in Brian 2 and emulated using SpiNNaker, and the results are shown in Section 3.

3. Results and discussion

3.1. Brian 2 simulations

The first experiment was performed to determine the number of neurons per population of the CPG that was needed to achieve a stable rate value along the simulations. The neuron parameters

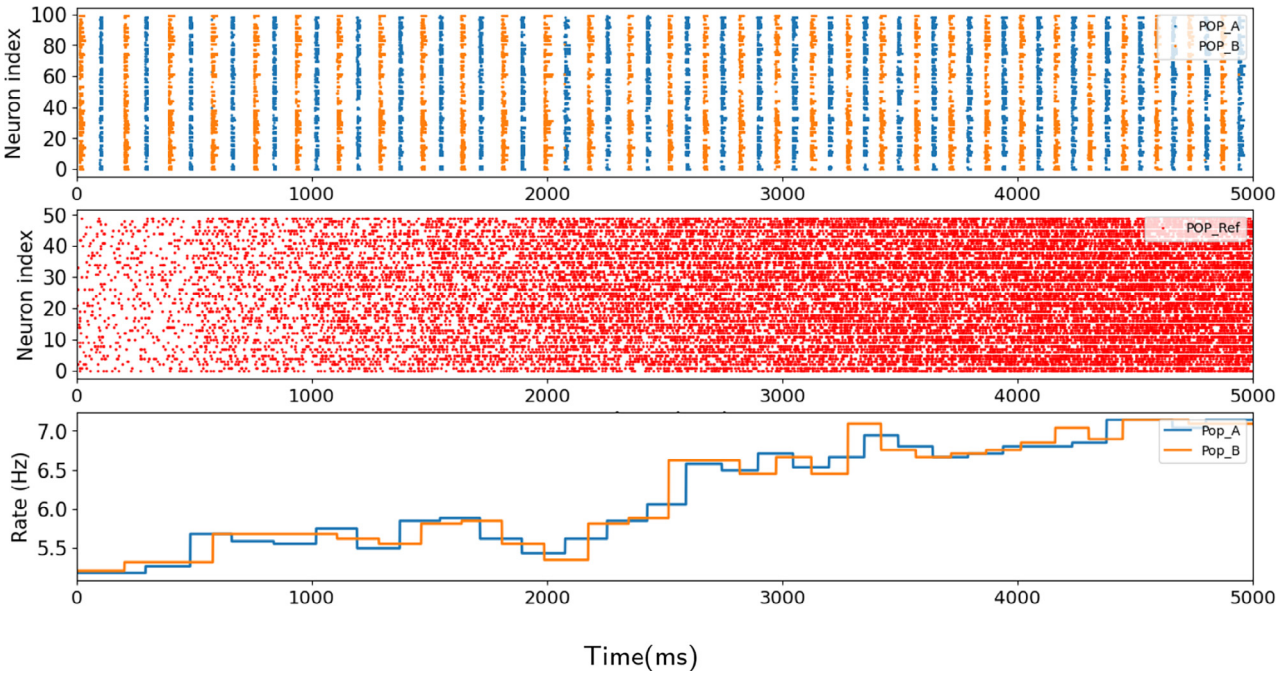

Fig. 7. Brian 2 results obtained when simulating a continuous increase in the frequency of *Ref* (increments of 20 Hz per 500 ms). The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

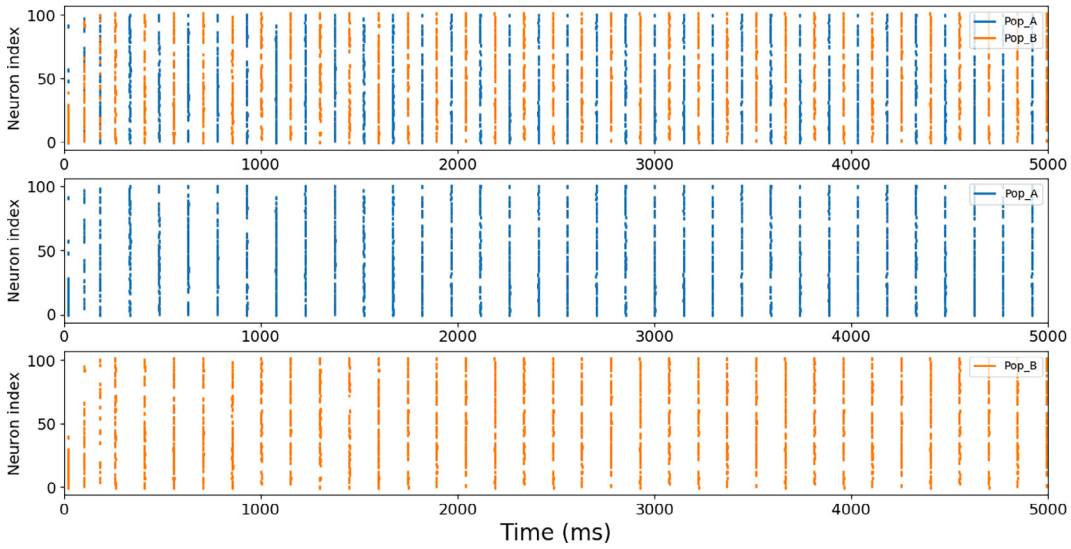

Fig. 8. SpiNNaker implementation of the CPG_{AB} with a constant oscillation frequency.

used for all the experiments are shown in Table 1. These parameters were optimized and found by means of a grid search algorithm [40]. Fig. 2 shows the maximum, minimum and mean values obtained for the rate of the simulations performed. A thousand simulations per number of neurons were run. As the number of neurons increased, the rate achieved is more stable and the standard deviation becomes lower. Starting from 40 neurons per population in the CPG_{AB}, the standard deviation is less than 1.5 and the behaviour is more stable. A hundred neurons per population shows the most stable output rate with the minimum deviation.

Once the number of neurons per population was fixed, to verify if the architecture presented in Section 2.2 behaved as expected, the operation of the CPG_{AB} in isolation was analyzed. This experiment ensured that it was able to produce a constant oscillation. Then, the operation of the same CPG was examined once interconnected with the CPG_{12}, performing tests with different stimuli to analyze the results obtained. Finally, the entire architecture was connected and analyzed in different scenarios based on the external input from the FSR sensor.

3.1.1. CPG_{AB} analysis

The topology of the CPG_{AB} can be seen in Fig. 1, where green arrows denote excitatory connections and red arrows denote inhibitory connections. As mentioned in Section 2.2.2, I_{St} is a constant current injected to all neurons in populations A and B, with a fixed value of 2.2 nA.

This current is the minimum value required to produce the oscillatory pattern in the CPG. While the proposed topology can work with higher values of I_{St}, this generates a higher frequency of oscillation and a noticeable increase in the noise introduced in

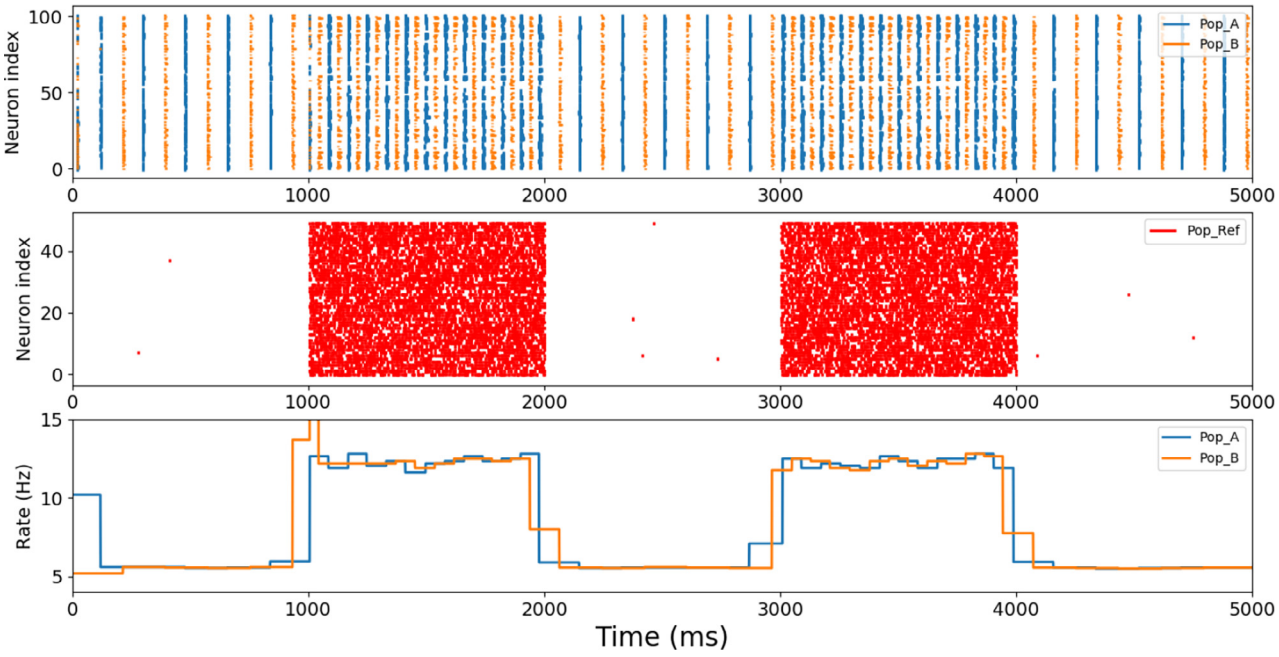

Fig. 9. SpiNNaker implementation of the CPG for extreme values of the input stimulus (0.1 Hz during most of the time, and 171 Hz between $t = 2000$ ms and $t = 3000$ ms, and between $t = 4000$ ms and $t = 5000$ ms). The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

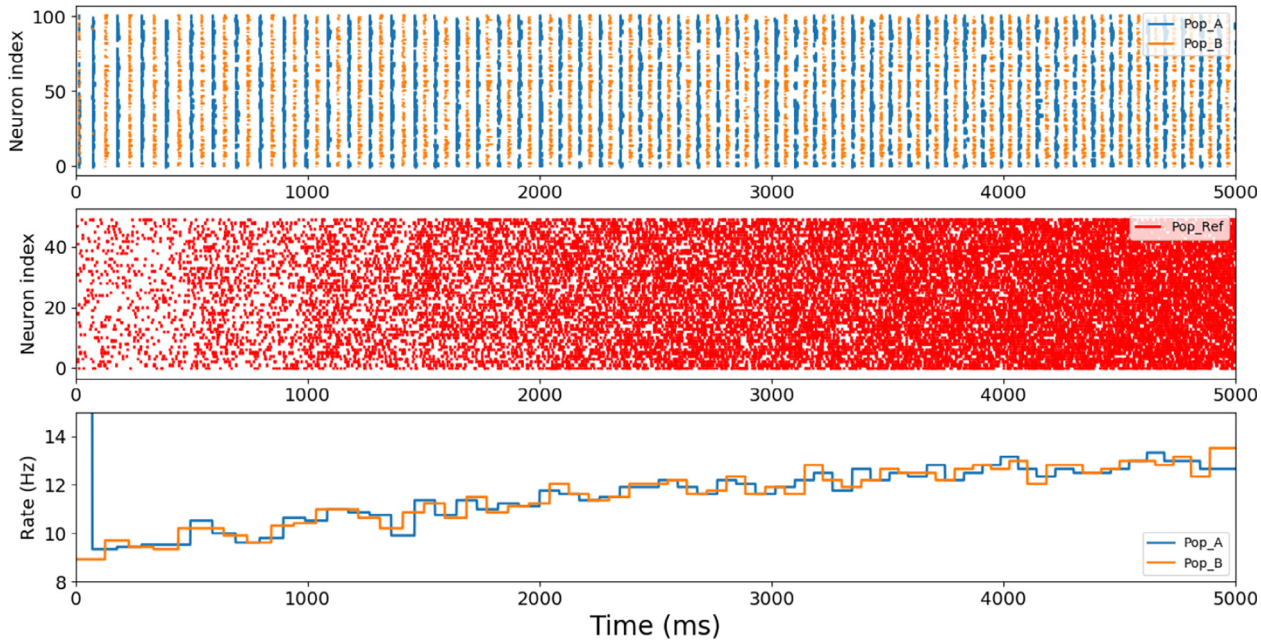

Fig. 10. SpiNNaker implementation of the CPG for ten increasing values of the stimulus rate (increments of 20 Hz per 500 ms). The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

the simulation (see Figs. 3 and 4). Thus, this value was set to 2.2 nA in order to be able to more easily appreciate the effect of feedback on the CPG. The results of the simulation performed are shown in Fig. 4, where a frequency of, approximately, 5.8 Hz was obtained for each population, while the total frequency of the CPG was 11.6 Hz.

3.1.2. Analysis of the effect of the sensor when used as input to the SNN

In order to study the robustness of the network against sudden changes in the oscillation frequency, an experiment where the values read from the sensor were alternating between maximum and minimum voltage peaks was performed. Initially, a 5 V input was simulated in 1 s and 2 s, both with a duration of 1 s. This made neurons in population *Ref* to fire at a frequency of, approximately, 171 Hz during this period. Before 1 s, between 2 s and 3 s, and after 4 s the sensor readings corresponded to 0 V. Fig. 5 shows the results of this simulation. It can be observed that, at time zero, since no information was being received from the sensor, *Ref* had no activity. Therefore, population 1 was excited by CPG_{AB} and population 2 was inhibited. In turn, population 1 slightly inhibited

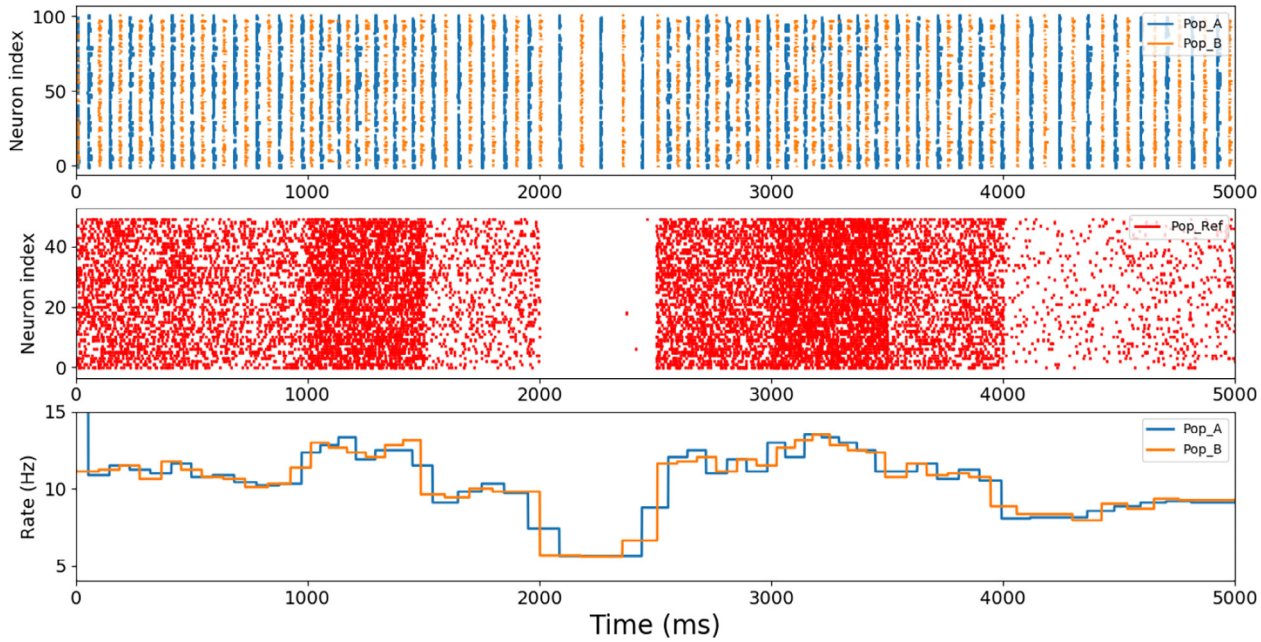

Fig. 11. SpiNNaker implementation of the CPG for ten random values of the stimulus rate (between 0 Hz and 200 Hz). The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

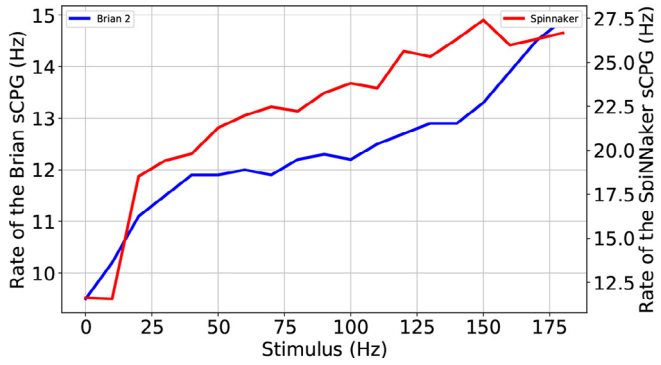

Fig. 12. Comparison of the results obtained in both Brian 2 simulation and SpiNNaker implementation. The plot shows the rate generated by the sCPG when the input stimulus (population *Ref*) is changed from 0 Hz to 180 Hz. The blue trace shows Brian 2 results (left y-axis) and the red trace the SpiNNaker results (right y-axis).

CPG_{AB}. At 1 s, *Ref* started firing at a frequency of approximately 171 Hz, exciting population 2 and inhibiting population 1. During this period, the former excited CPG_{AB}, increasing its oscillation frequency. After 2 s, population 1 started dominating population 2 again, slightly inhibiting CPG_{AB} again. Exactly the same behavior can be seen again starting at 3 s. In this figure it can be observed how the frequency of CPG_{AB} increased considerably between 1 s and 2 s and between 3 s and 4 s, obtaining minimum frequencies of 8 Hz and maximum frequencies of 15 Hz.

On the other hand, different simulations of more realistic cases were performed. Initially, 10 random voltage values were used in order to simulate different readings from the FSR sensor. These values were updated every 500 ms. Specifically, the frequency values for Poisson distribution in *Ref* were (171, 40, 80, 30, 5, 130, 50, 76, 20, 150) Hz. In Fig. 6 it can be seen how CPG_{AB} adapts its oscillation frequency based on the inputs stimuli, obtaining maximum and minimum frequency peaks of 15 Hz and 8 Hz, respectively.

After this, a constant increase in the values of the readings from the sensor was simulated. In particular, increases in steps of 20 Hz per 500 ms were introduced in the frequency of *Ref*. To check the performance limits of the CPG_{AB}, the last injected frequency value exceeded up to 17% the maximum theoretical value of 171 Hz. Fig. 7 shows the results of this experiment.

As can be seen in the figure, although there is a significant increase in the amount of noise, the oscillation frequency of CPG_{AB} does not increase, achieving a maximum frequency of 15 Hz.

3.2. Running the model on SpiNNaker

The proposed sCPG model was tested in the SpiNNaker neuromorphic hardware. The neuronal model is defined in sPyNNaker [37], a PyNN-based software interface that allows a quick prototyping and implementation of spiking neural networks in the SpiNNaker platform.

The spiking neuron model is the LIF with fixed threshold and decaying-exponential post-synaptic current, whose parameters are given in Table 1. These parameters were chosen so as to emulate the neuron dynamics of the simulations in Brian 2.

In order to show that the model achieves a good performance in SpiNNaker, we performed three tests: first, by verifying that the constant oscillations of the CPG_{AB} were observed and matched the rates presented in Brian 2. Then, by implementing the whole sCPG with increasing rates of the input sensor represented by the reference population. Finally, by testing the sCPG under random stimuli.

3.2.1. CPG_{AB} implementation

For the implementation of the CPG in Spinnaker (see Fig. 8) 100 neurons were used, each with a constant current I_{St} equal to 2.2 nA, like in the equivalent Brian 2 experiment. The measured rate of the oscillatory pattern of populations A and B was 11.62 Hz, which matches the rate measured in the Brian 2 simulation. Although some spikes were lost in the raster plot compared to Brian 2, which can be attributed to limited support of floating-point calculations in SpiNNaker, the pattern appears consistent and with little noise.

3.2.2. CPG under different stimuli

The SpiNNaker implementation of the full CPG, including the feedback network was tested under different conditions of the

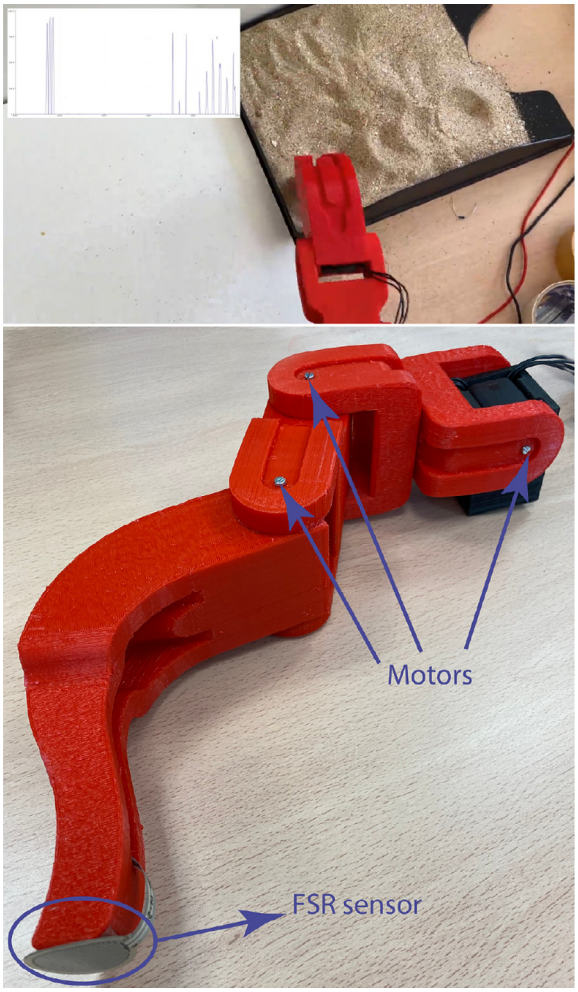

Fig. 13. Top: snapshot of the setup used to perform the experiments with the FSR sensor for two different terrains is shown. A video of the experiments can be seen in: https://youtu.be/3s89p3qYnCU. In the video, the leg was moving by hand. Bottom: picture of one of the legs to show where the FSR sensor and mot.ors are located.

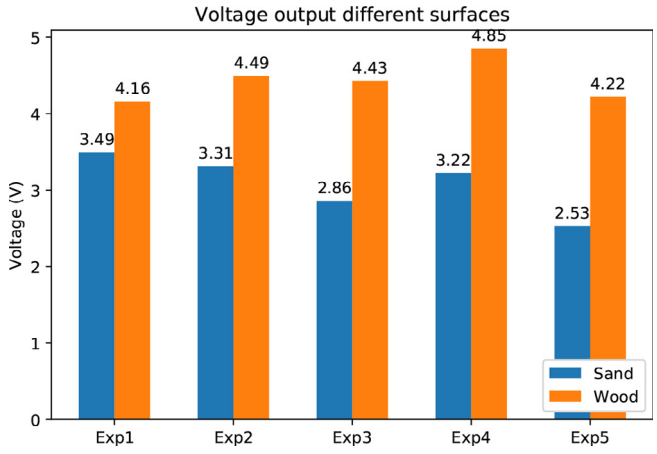

Fig. 14. The mean output voltage of the FSR sensor for two different terrains is shown. A single leg was moved up and down in two different surfaces five times. The blue bars correspond to the experiments performed in a sand surface, while the orange bars correspond to the experiments performed in a woody terrain. At the top of the bars, the mean value is shown. The standard deviation is 0.34 for the sand surface and 0.24 for the wood surface. As it can be seen, the values obtained in a softer terrain (sand) are lower than in a stiffer (wood) terrain.

stimulus. First, we simulated a sudden change of the value of the sensor, represented by the rate of the Poisson generator in the *Ref* population (see Fig. 9). This rate was set to 0.1 Hz for the first 2 s of the simulation and oscillated between 171 Hz and 0.1 Hz for the next four seconds, in order to appreciate both regimes of the CPG. It can be seen how, with a low *Pref* frequency, the first feedback population dominates and the rate of the output oscillatory is low, at around 12.5 Hz. With a high *Pref* frequency, it is the second feedback population which dominates and the measured oscillatory pattern displays a peak frequency of 27.5 Hz.

Figs. 10 and 11 show the spiking response of the SpiNNker CPG to increasing rates of *Ref* and to random values of *Ref*, respectively. The two regimes can be clearly observed in the spiking response of populations 1 and 2 and in the measured oscillatory rates of Populations A and B, proving that the feedback mechanism works correctly.

3.3. Comparison between the results obtained in Brian 2 and SpiNNaker

Fig. 12 shows the comparison made between both approaches: the simulations on Brian 2 and the implementation on the SpiNNaker platform. In this experiment, the same input stimulus from the sensor was used for both approaches, which went from 0 Hz to 180 Hz. The rates generated by both sCPGs were very similar. In the case of the Brian 2, the operation frequency of the sCPG ranged from 9.5 Hz to 14.9 Hz, and from 11.62 Hz to 26.66 Hz in the case of the SpiNNaker platform. The calculated Pearson correlation coefficient between both is 0.905.

3.4. Testing the sCPG models with real stimuli from the FSR sensor.

The values read from the FSR are used as feedback, and they determine the kind of surface on which the robot is stepping on. As shown in Fig. 14, the feedback values determine if the surface is softer or stiffer. Since the frequency of the sCPG is correlated with the output value from the FSR, the speed of the robot will be also directly correlated with the kind of surface that the robot is stepping onto. The amplitude of the leg during the swing movement will remain the same while the speed of the swing will be decreased if the surface is soft and increased if the surface is stiff. This behaviour can be checked in Fig. 19.

Tests have been performed with real stimuli received from an FSR sensor implanted in one of the robot's legs. Specifically, wood and sand terrains have been used to verify that the hypotheses proposed in this paper are correct. Fig. 13 shows the setup of the experiments performed. The mean values obtained can be seen in Fig. 14. Five experiments were performed to compare a firmer ground (wood) and a more irregular one (sand). The bars on the plot represent the mean voltage whenever the leg is completely touching the surface. As it can be seen, the values obtained on the wooden surface exceed 4 volts, while in sand they are below 3.5 volts.

Thanks to the relationship established between the value of the voltage obtained by the FSR and the *Pref* population firing rate, it is possible to stimulate to a greater or lesser extent the oscillation frequency of the CPG_{AB}, which directly translates into a substantial variation of the walking speed of a robot. These experiments have been simulated in Brian 2 and implemented in SpiNNNaker in order to verify the performance of the whole system. The results from these experiments are presented in Figs. 15 and 17 for the Brian 2 simulations, and in Figs. 16 and 18 for the SpiNNaker deployments.

As shown in Fig. 19, the difference between the frequency generated by *Pref* (which maps the FSR output) in sand and wood is, approximately, 40%, while the results show that the CPG_{AB} oscilla-

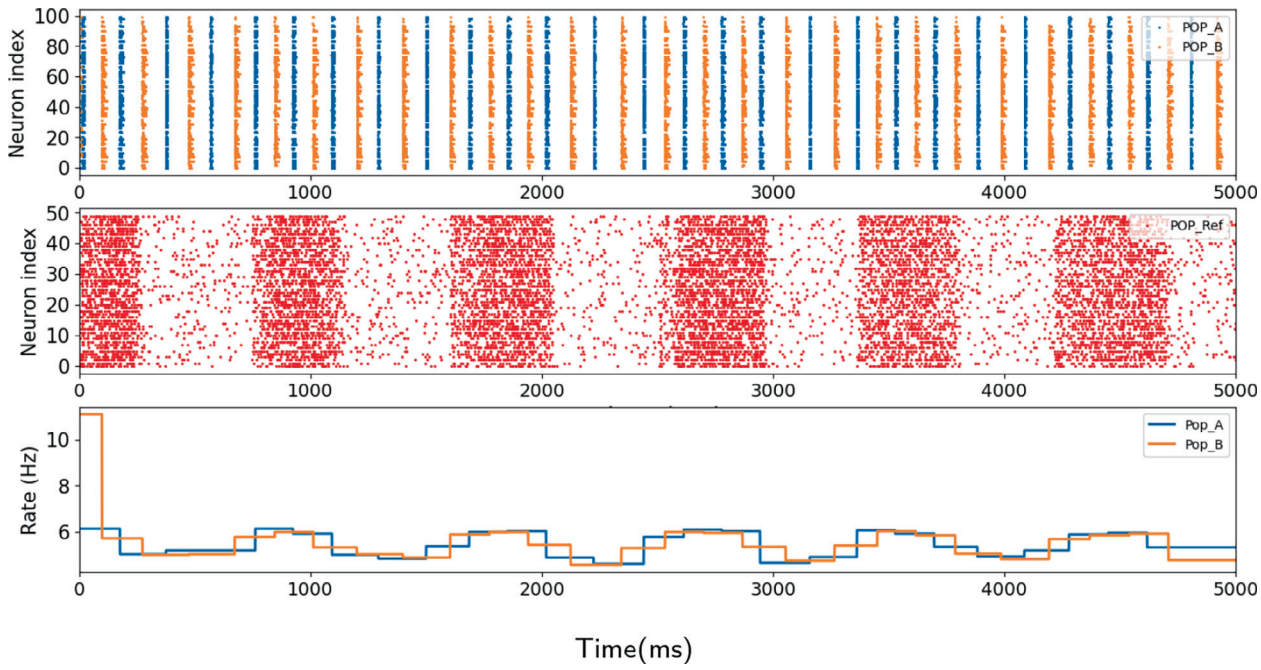

Fig. 15. Brian 2 simulation of the CPG for values of the FSR on one leg when walking on a sandy terrain. The first row shows the output spikes of the CPG_{AB}, the middle row shows the spikes fired by the population mapping the FSR sensor output and the third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

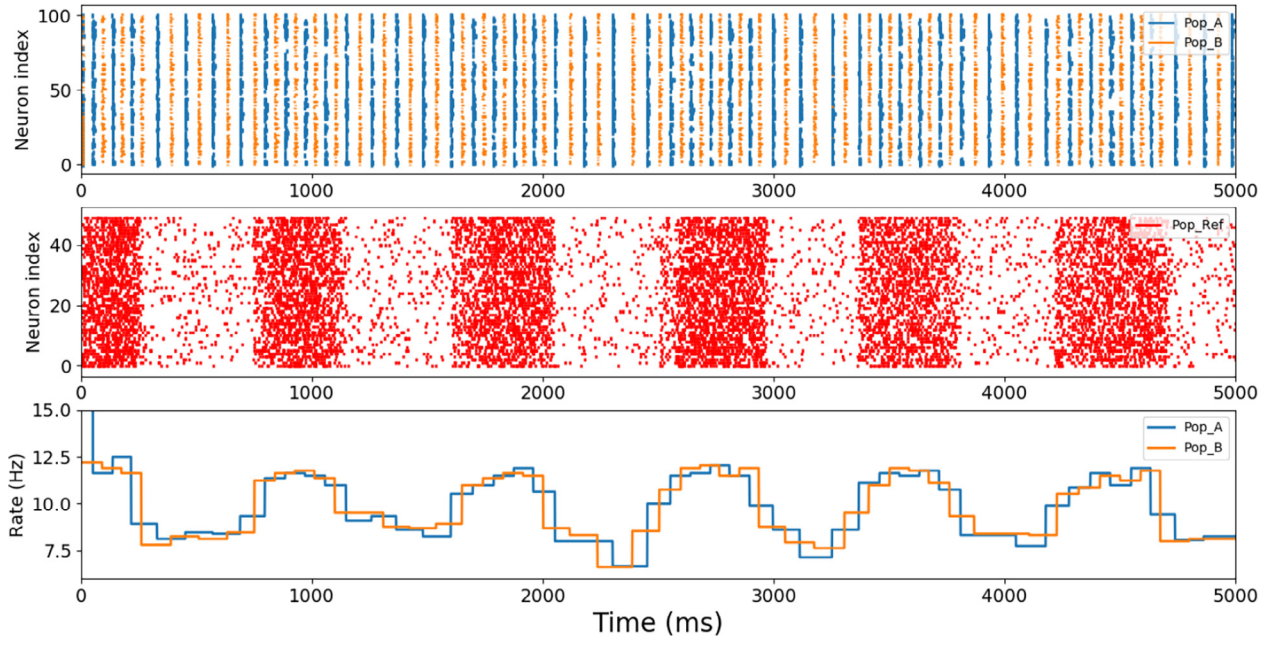

Fig. 16. CPG implemented on SpiNNaker when the leg is touching a sandy terrain. The first row shows the output spikes of the CPG_{AB}, the middle row shows the spikes fired by the population mapping the FSR sensor output and the third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

tion is 17% lower when the leg is in the sandy terrain than when it is in wooden terrain. The frequency difference shows that it is possible to adapt the sCPG frequency to the external conditions given by the FSR output.

A new experiment involving a transition between sandy and wooden terrains was performed to demonstrate the variability of the frequency of the spiking CPG. For this purpose, only the maximum values obtained from the *Pref* population were taken, which correspond to the values obtained when the leg was fully supported on the ground, either in sand or wood. The same experiment was performed in Brian 2 (Fig. 20) and SpiNNaker (Fig. 21): the first 2500 ms correspond to the robot walking on sand, while the last 2500 ms correspond to the robot walking on wood.

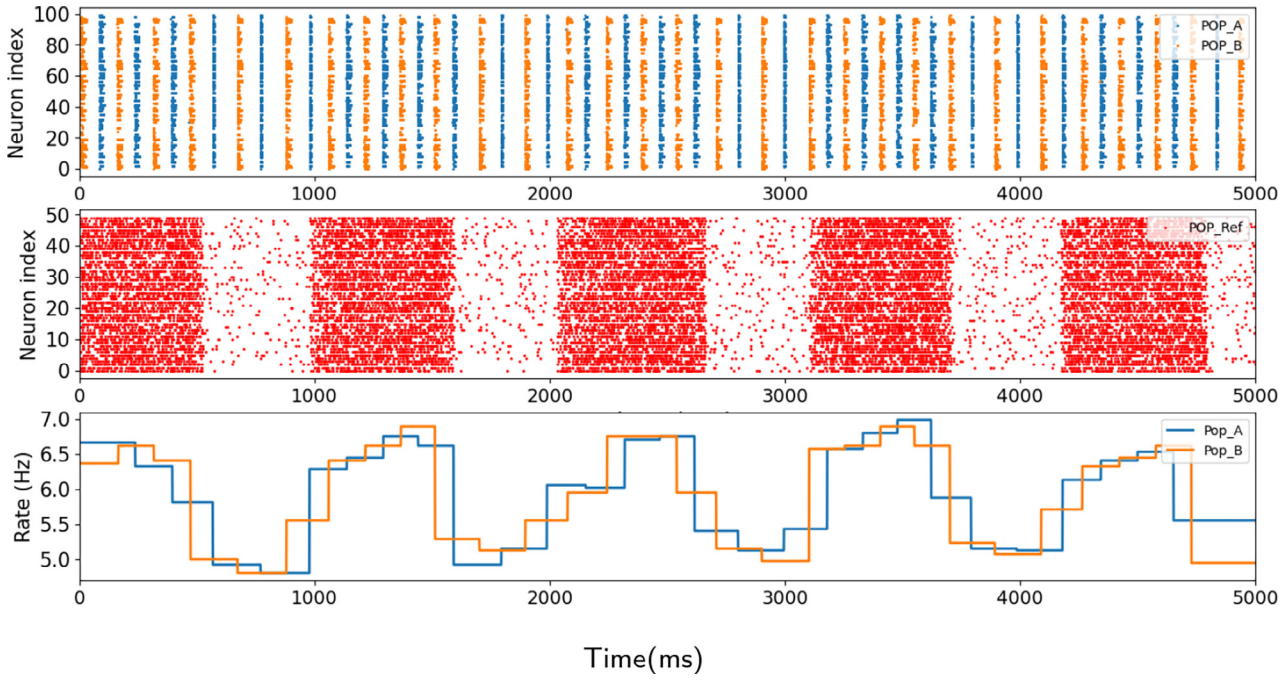

Fig. 17. Brian 2 simulation of the CPG for values of the FSR on one leg when walking on a wooden terrain. The first row shows the output spikes of the CPG_{AB}, the middle row shows the spikes fired by the population mapping the FSR sensor output and the third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

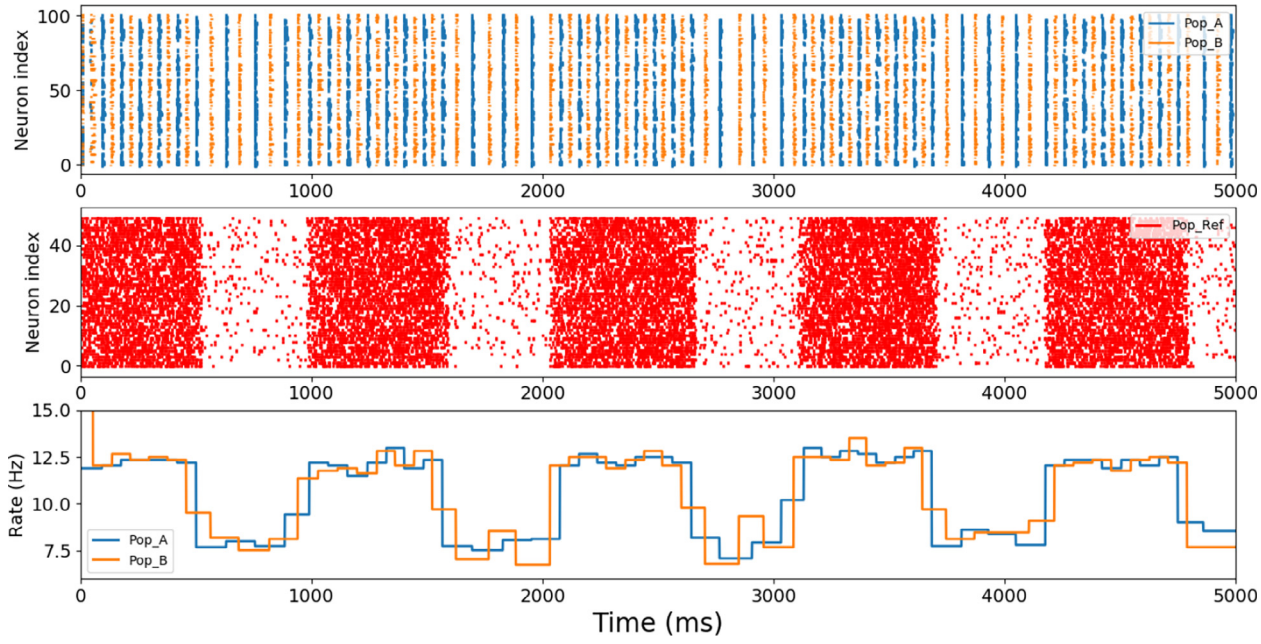

Fig. 18. CPG implemented on SpiNNaker when the leg is touching a wooden terrain. The first row shows the output spikes of the CPG_{AB}, the middle row shows the spikes fired by the population mapping the FSR sensor output and the third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

As can be seen in Fig. 20, the frequency of the CPG in Brian 2 for the sandy terrain oscillates around 6 Hz, while for the wooden terrain it increases up to 7 Hz.

On the other hand, in SpiNNNaker (Fig. 21), the frequency of the CPG obtained when the robot was walking on a sandy terrain oscillates around 12 Hz, increasing up to 13 Hz for the wooden terrain.

4. Conclusions

In this paper, we have presented what, to the best of our knowledge is, the first sCPG that incorporates a feedback in the loop to modify the locomotion frequencies when applied to any legged robot through an adaptation mechanism. The feedback was pro-

vided by a FSR (but any sensor might be used) to simulate the force exerted on each limb of a future robot. With the use of this sensor, we injected different stimuli to the sCPG. Firstly, we performed some tests to find the optimal (minimum) input value, where the feedback effect could be easily observed. Then, some tests were performed to observe the effect in the oscillation frequencies under different stimuli. Finally, we carried out some experiments where the values read from the sensor were alternating between maximum and minimum voltage peaks in order to study the robustness of the sCPG against sudden changes in the oscillation frequency. From these experiments, we conclude that our sCPG presents a robust behavior in the adaptability of the oscillation frequencies. These frequencies can be used further in the generation of locomotion gaits for legged robots with the advantage that they can be modified depending on the terrain conditions.

The implementation of the sCPG was firstly done by using the Brian 2 simulator and then, considering the same parameters, it was migrated to SpiNNaker. The results in both cases are highly correlated, as it is shown in Fig. 12; this fact demonstrates the reproducibility of our architecture on any platform.

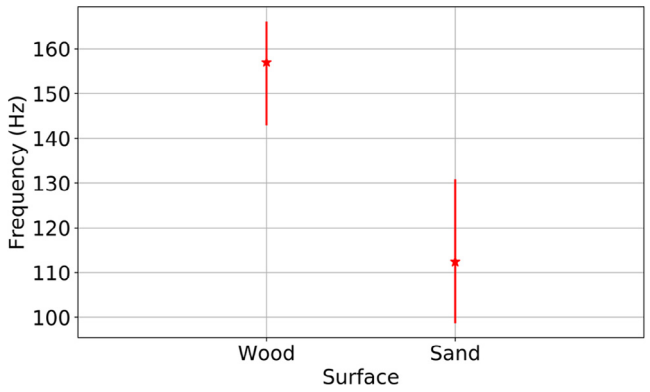

Fig. 19. Frequency generated by the *Pref* for two surfaces: wood and sand. The bars shows the mean, maximum and minimum frequency. The mean frequency for the wood surface is 157 Hz and for the sand terrain is 112.4 Hz.

Compared to the biological locomotion mechanism, we considered that our approach is highly plausible in two senses. The first one is that the proposed network model is based on spiking neurons, which are considered the neuron models that most closely mimic the behavior of biological neurons. The second aspect is that the implementation performed in SpiNNaker allowed us to improve it in terms of the power consumption, and the hardware by itself attempts to be an artificial representation of the brain.

Different experiments with real values from the output of the FSR sensors after attaching it to a leg have been performed. The proposed sCPG was stimulated with values obtained from the FSR when walking on two different terrains: sand and wood. The results confirm that the sCPG is able to adapt its oscillation rate depending on the feedback obtained from the surface that the robot is stepping onto.

As shown in Fig. 22, this network can be applied to any configuration of joints per leg, where one joint represents the vertical movement of the leg, and the other joint represents the horizontal movement. although a third joint is necessary in certain robotic models (e.g. hexapod models), it only acts as a support for the robot, so it would only be necessary to energize the motor to keep the joint in a fixed position.

As a future work, we propose to embed the SpiNNaker system into different legged robots (e.g. biped, quadruped and hexapod) to validate our approach on a real robotic platform by using the architecture shown in Fig. 22. In this architecture, the output of the Spiking Central Pattern Generator CPG_{AB} would drive four populations to execute the movement of each leg. Populations Forward Internal Motor (FIMX) and Reverse Internal Motor (RIMX) would control the clockwise and counterclockwise movement of the internal joint of the leg, respectively, while the populations FEMX and REMX would produce the clockwise and counterclockwise movement of the external joint. Furthermore, our proposal could

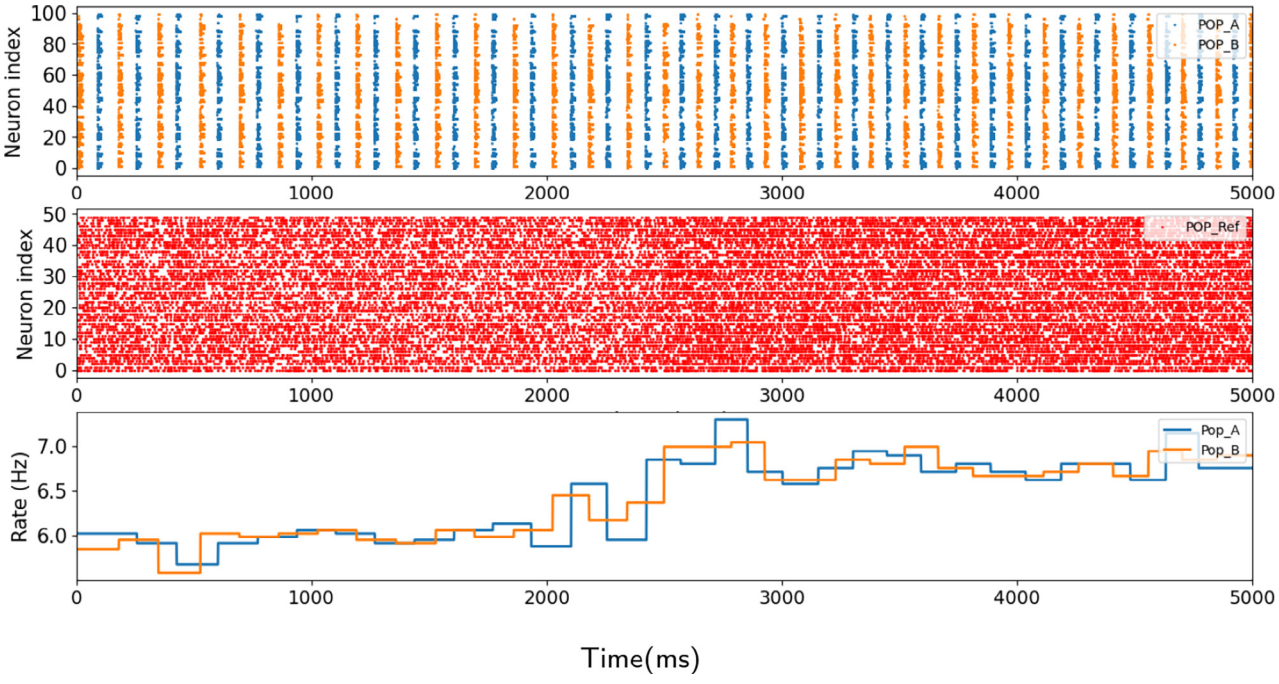

Fig. 20. Performance of the CPG simulated with Brian 2 along a move from a sandy terrain to a wooden one. The first 2500 ms corresponds to the robot walking on a sandy terrain, where the average frequency is 6 Hz, while in the last 2500 ms the frequency increases up to 7 Hz in a wooden terrain. The third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

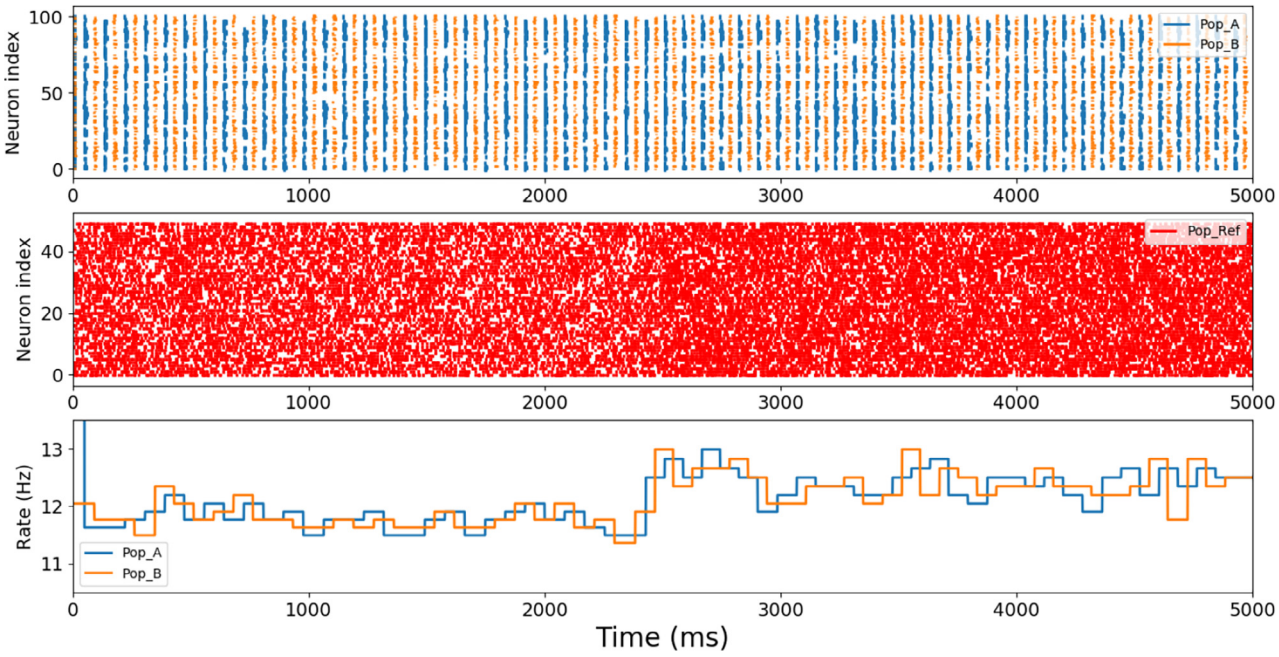

Fig. 21. Performance of the CPG implemented on SpiNNaker along a move from a sandy terrain to a wooden one. The first 2500 ms corresponds to the robot walking on a sandy terrain, where the average frequency is 12 Hz, while in the last 2500 ms the frequency increases up to 13 Hz in a wooden terrain. and the third row shows the firing rate of each population of the CPG in Hz. Thus, the global output rate of the CPG will be the combination of both.

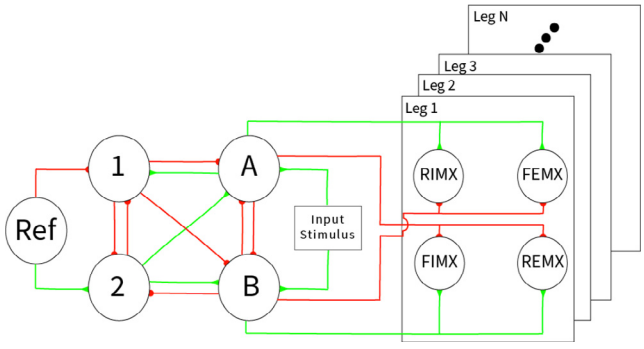

Fig. 22. Diagram of the proposed implementation architecture to drive a robotic platform with N legs. The red connections denote inhibitory connections and the green ones excitatory ones. The structure of populations that has to be replicated for each leg is composed of: FIMX, the RIMX, the FEMX and the REMX.

directly process spatio-temporal patterns with the same sCPG by incorporating a neuromorphic sensor.

CRediT authorship contribution statement

Pablo Lopez-Osorio: Methodology, Validation, Software, Conceptualization, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Alberto Patino-Saucedo:** Methodology, Validation, Software, Data curation, Writing – original draft, Writing – review & editing. **Juan P. Dominguez-Morales:** Project administration, Writing – original draft, Writing – review & editing, Resources, Conceptualization, Supervision. **Horacio Rostro-Gonzalez:** Funding acquisition, Project administration, Writing – original draft, Writing – review & editing, Resources, Supervision. **Fernando Perez-Peña:** Funding acquisition, Project administration, Writing – original draft, Writing – review & editing, Resources, Conceptualization, Supervision.

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.

Acknowledgements

This work was partially supported by the Interreg Atlantic Area Programme through the European Regional Development Fund (TIDE - Atlantic network for developing historical maritime tourism, EAPA_630/2018), by the Spanish grant (with support from the European Regional Development Fund) MIND-ROB (PID2019-105556 GB-C33) and by the EU H2020 project CHIST-ERA SMALL (PCI2019-111841–2).

References

[1] A.J. Ijspeert, Central pattern generators for locomotion control in animals and robots: a review, Neural networks 21 (2008) 642–653.
[2] J.L. Krichmar, H. Wagatsuma, Neuromorphic and brain-based robots, Cambridge University Press, 2011.
[3] G. Indiveri, B. Linares-Barranco, T.J. Hamilton, A. Van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud, et al., Neuromorphic silicon neuron circuits, Front. Neurosci. 5 (2011) 73.
[4] R.J. Vogelstein, F.V. Tenore, L. Guevremont, R. Etienne-Cummings, V.K. Mushahwar, A silicon central pattern generator controls locomotion in vivo, IEEE Trans. Biomed. Circuits Syst. 2 (2008) 212–222.
[5] T. Kano, K. Sakai, K. Yasui, D. Owaki, A. Ishiguro, Decentralized control mechanism underlying interlimb coordination of millipedes, Bioinspiration Biomimetics 12 (2017) 036007.
[6] D. Owaki, T. Kano, K. Nagasawa, A. Tero, A. Ishiguro, Simple robot suggests physical interlimb communication is essential for quadruped walking, J. R. Soc. Interface 10 (2013) 20120669.
[7] E. Donati, F. Corradi, C. Stefanini, G. Indiveri, A spiking implementation of the lamprey's Central Pattern Generator in neuromorphic VLSI, in: IEEE 2014 Biomedical Circuits and Systems Conference, BioCAS 2014 – Proceedings, pp. 512–515..

[8] H. Rostro-Gonzalez, P.A. Cerna-Garcia, G. Trejo-Caballero, C.H. Garcia-Capulin, M.A. Ibarra-Manzano, J.G. Avina-Cervantes, C. Torres-Huitzil, A CPG system based on spiking neurons for hexapod robot locomotion, Neurocomputing 170 (2015) 47–54.
[9] B. Cuevas-Arteaga, J.P. Dominguez-Morales, H. Rostro-Gonzalez, A. Espinal, A.F. Jimenez-Fernandez, F. Gomez-Rodriguez, A. Linares-Barranco, A SpiNNaker application: design, implementation and validation of SCPGs, in: International Work-Conference on Artificial Neural Networks, Springer, pp. 548–559..
[10] S.B. Furber, F. Galluppi, S. Temple, L.A. Plana, The spinnaker project, Proc. IEEE 102 (2014) 652–665.
[11] D. Gutierrez-Galan, J.P. Dominguez-Morales, F. Perez-Peña, A. Jimenez-Fernandez, A. Linares-Barranco, NeuroPod: a real-time neuromorphic spiking CPG applied to robotics, Neurocomputing 381 (2020) 10–19.
[12] I. Polykretis, G. Tang, K.P. Michmizos, An astrocyte-modulated neuromorphic central pattern generator for hexapod robot locomotion on intel's loihi, International Conference on Neuromorphic Systems (2020) 1–9.
[13] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S.H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, et al., Loihi: A neuromorphic manycore processor with on-chip learning, IEEE Micro 38 (2018) 82–99.
[14] B. Strohmer, P. Manoonpong, L.B. Larsen, Flexible spiking cpgs for online manipulation during hexapod walking, Front. Neurorobot. 14 (2020) 41.
[15] G. Sartoretti, S. Shaw, K. Lam, N. Fan, M. Travers, H. Choset, Central pattern generator with inertial feedback for stable locomotion and climbing in unstructured terrain, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 1–5..
[16] A. Spaeth, M. Tebyani, D. Haussler, M. Teodorescu, Neuromorphic closed-loop control of a flexible modular robot by a simulated spiking central pattern generator, in: 2020 3rd IEEE International Conference on Soft Robotics (RoboSoft), IEEE, pp. 46–51..
[17] E.M. Izhikevich, redWhich model to use for cortical spiking neurons?, IEEE Trans Neural Networks 15 (2004) 1063–1070.
[18] D. Gutierrez-Galan, J.P. Dominguez-Morales, F. Perez-Pena, A. Jimenez-Fernandez, A. Linares-Barranco, Live Demonstration: Neuromorphic Robotics, from Audio to Locomotion Through Spiking CPG on SpiNNaker, in: 2019 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, pp. 1–1..
[19] B. Strohmer, R.K. Stagsted, P. Manoonpong, L.B. Larsen, Integrating non-spiking interneurons in spiking neural networks, Front. Neurosci. 15 (2021) 184.
[20] R.M. Harris-Warrick, Neuromodulation and flexibility in central pattern generator networks, Curr. Opin. Neurobiol. 21 (2011) 685–692.
[21] A.S. Lele, Y. Fang, J. Ting, A. Raychowdhury, Learning to walk: Spike based reinforcement learning for hexapod robot central pattern generation, in: 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), IEEE, pp. 208–212..
[22] C. Li, R. Lowe, T. Ziemke, Humanoids learning to walk: a natural CPG-actor-critic architecture, Front. Neurorobot. 7 (2013) 5.
[23] J. Ting, Y. Fang, A.S. Lele, A. Raychowdhury, Bio-inspired gait imitation of hexapod robot using event-based vision sensor and spiking neural network, arXiv preprint arXiv:2004.05450 (2020)..
[24] M. Stimberg, R. Brette, D.F. Goodman, Brian 2, an intuitive and efficient neural simulator, Elife 8 (2019) e47314.
[25] N.T. Carnevale, M.L. Hines, The NEURON book, Cambridge University Press, 2006.
[26] A.P. Davison, D. Brüderle, J.M. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, P. Yger, PyNN: a common interface for neuronal network simulators, Front. Neuroinformatics 2 (2009) 11.
[27] S.B. Furber, D.R. Lester, L.A. Plana, J.D. Garside, E. Painkras, S. Temple, A.D. Brown, Overview of the spinnaker system architecture, IEEE Trans. Comput. 62 (2012) 2454–2467.
[28] S. Furber, P. Bogdan, SpiNNaker: A Spiking Neural Network Architecture, Boston-Delft: now publishers (2020).
[29] H. Markram, The human brain project, Sci. Am. 306 (2012) 50–55.
[30] E. Painkras, L.A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D.R. Lester, A.D. Brown, S.B. Furber, SpiNNaker: A 1-W 18-core system-on-chip for massively-parallel neural network simulation, IEEE J. Solid-State Circuits 48 (2013) 1943–1953.
[31] L.A. Plana, S.B. Furber, S. Temple, M. Khan, Y. Shi, J. Wu, S. Yang, A GALS infrastructure for a massively parallel multiprocessor, IEEE Design Test Comput. 24 (2007).
[32] M. Mahowald, VLSI analogs of neuronal visual processing: a synthesis of form and function, Ph.D. thesis, California Institute of Technology Pasadena, 1992..
[33] L.A. Plana, J. Garside, J. Heathcote, J. Pepper, S. Temple, S. Davidson, M. Luján, S. Furber, spiNNlink: FPGA-Based Interconnect for the Million-Core SpiNNaker System, IEEE Access 8 (2020) 84918–84928.
[34] A. Yousefzadeh, M. Jabłoński, T. Iakymchuk, A. Linares-Barranco, A. Rosado, L.A. Plana, S. Temple, T. Serrano-Gotarredona, S.B. Furber, B. Linares-Barranco, On multiple AER handshaking channels over high-speed bit-serial bidirectional LVDS links with flow-control and clock-correction on commercial FPGAs for scalable neuromorphic systems, IEEE Trans. Biomed. Circuits Syst. 11 (2017) 1133–1147.
[35] J.P. Dominguez-Morales, A. Jimenez-Fernandez, A. Rios-Navarro, E. Cerezuela-Escudero, D. Gutierrez-Galan, M.J. Dominguez-Morales, G. Jimenez-Moreno, Multilayer spiking neural network for audio samples classification using spinnaker, in: International conference on artificial neural networks, Springer, pp. 45–53..
[36] T. Schoepe, D. Gutierrez-Galan, J.P. Dominguez-Morales, A. Jimenez-Fernandez, A. Linares-Barranco, E. Chicca, Neuromorphic sensory integration for combining sound source localization and collision avoidance, in: 2019 IEEE Biomedical Circuits and Systems Conference (BioCAS), IEEE, pp. 1–4..
[37] O. Rhodes, P.A. Bogdan, C. Brenninkmeijer, S. Davidson, D. Fellows, A. Gait, D.R. Lester, M. Mikaitis, L.A. Plana, A.G. Rowley, et al., sPyNNaker: a software package for running PyNN simulations on SpiNNaker, Front. Neurosci. 12 (2018) 816.
[38] W. Gerstner, W. Kistler, Spiking Neuron Models. Single Neurons, Populations, Plasticity, Cambridge University Press, 2002.
[39] S.J. Tripathy, J. Savitskaya, S.D. Burton, N.N. Urban, R.C. Gerkin, NeuroElectro: a window to the world's neuron electrophysiology data, Front. Neuroinformatics 8 (2014) 40.
[40] F.J. Pontes, G. Amorim, P.P. Balestrassi, A. Paiva, J.R. Ferreira, redDesign of experiments and focused grid search for neural network parameter optimization, Neurocomputing 186 (2016) 22–34.

Pablo Lopez-Osorio is a PhD student in Bioengineering, Automation and Robotics. He received the Engineering degree in Electronics from the University of Cádiz (Spain) and his M.S. degree (Research in Systems and Computation Engineering) from the University of Cadiz (Spain) in 2019 and 2020 respectively. He is an Associate researcher in the University of Cadiz since 2020. His research interests include neuromorphic engineering, spiking neural networks, motor control, telecomunications, neurorobotics and neuromorphic sensors.

Alberto Patiño-Saucedo is a PhD student in Electrical Engineering at the University of Guanajuato (Mexico). He received the Engineering degree in Electronics from the Industrial University of Santander (Colombia) and the M.E. degree in Electrical Engineering from the University of Guanajuato in 2015 and 2017 respectively. His research interests include neuromorphic engineering, spiking neural networks, computational neuroscience and machine learning.

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 (Sevilla, Spain), in 2014, 2015 and 2018, respectively.

From October 2015 to December 2018, he was a PhD student in the Architecture and Technology of Computers Department of the University of Seville with a research grant from the Spanish Ministry of Education and Science. Since January 2019, he has been working as Assistant Professor in the same department. His research interests include neuromorphic engineering, spiking neural networks, neuromorphic sensors, audio processing, medical image analysis, convolutional neural networks and computer-aided diagnosis systems.

Horacio Rostro-Gonzalez received his B.Eng. in Electronics from the ITC in 2003 and a M.E. degree with honors in Electrical Engineering from the University of Guanajuato in 2006, both in Mexico and the D.Sc. in Computational Neuroscience from the University of Nice-Sophia Antipolis, France in 2011. From 2011 to 2012, he also was a research fellow at the University of Cyprus, where he mainly worked on the FP7 EU SCANDLE Project. Between 2017 and 2018 he was visiting professor within the NST research group at the Technical University of Munich. Since May 2012, he is a full professor within the Department of Electronics at the University of Guanajuato in Mexico. He has authored around 40 journals and conferences proceeding papers. His research interests include reconfigurable electronics (FPGAs), neuromorphic engineering, computational neuroscience, bio-inspired algorithms, robotics, parallel computing and signal processing.

Fernando Perez-Peña received the Engineering degree in Telecommunications from the University of Seville (Spain) and his Ph.D. degree (specialized in neuromorphic motor control) from the University of Cadiz (Cadiz, Spain) in 2009 and 2014 respectively. In 2015 he was a postdoc at CITEC (Bielefeld University, Germany). He is an Assistant Professor in the Architecture and Technology of Computers Department of the University of Cadiz since 2014. His research interests include neuromorphic engineering, CPG, motor control and neurorobotics.