Chapter 4

Spiking Central Pattern Generators through Reverse Engineering of Locomotion Patterns

Andrés Espinal, Marco Sotelo-Figueroa,
Héctor J. Estrada-García,
Manuel Ornelas-Rodríguez and
Horacio Rostro-Gonzalez

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.72348

Abstract

In robotics, there have been proposed methods for locomotion of nonwheeled robots based on artificial neural networks; those built with plausible neurons are called spiking central pattern generators (SCPGs). In this chapter, we present a generalization of reported deterministic and stochastic reverse engineering methods for automatically designing SCPG for legged robots locomotion systems; such methods create a spiking neural network capable of endogenously and periodically replicating one or several rhythmic signal sets, when a spiking neuron model and one or more locomotion gaits are given as inputs. Designed SCPGs have been implemented in different robotic controllers for a variety of robotic platforms. Finally, some aspects to improve and/or complement these SCPG-based locomotion systems are pointed out.

Keywords: central pattern generators, spiking neural networks, reverse engineering, metaheuristic, locomotion patterns

1. Introduction

Since its beginning, robotics has been a research field strongly influenced by nature. For robotic platforms where wheels to displace themselves are not used, researchers have taken inspiration not only in the physical form of living beings as archetypes of their designs (e.g., legged, finned or winged robotic platforms), but also in the mechanisms to allow their locomotion (e.g., walking, swimming or flying), mainly known as central pattern generators (CPGs) [1]. In biology, the basis of CPGs was settled down in Brown's studies about how the rhythmic movements of locomotion in living beings are created [2]. In [3], Brown experimentally discovered that neurons,

which inhibiting each other, generate periodically rhythmic activities that control bending and tension of muscles involved in locomotion. Moreover, GPGs have been linked to other unconscious activities besides locomotion such as swallowing, digestion, heart beating and breathing [4, 5].

Furthermore, CPGs have become a suitable alternative to nonbiologically inspired methods for locomotion systems of nonwheeled robotic platforms [6]; this is due to several and interesting features of CPGs such as adaptability, rhythmicity, stability and variety [7]. The CPG-based locomotion systems have been successfully designed and implemented at software and/or hardware levels for different nonwheeled robotic platforms [8] such as walking robots (biped [9], quadruped [10], hexapod [11] and octopod [12]), swimming robots [13], flying robots [14]), among others (i.e., snake robots [15] and salamander robots [16]). Although vast amount of works made and reported in the state of the art about CPG-based locomotion systems, there is not a general and standard methodology to build CPGs [6]; however, working with CPGs commonly involves the following three phases [7]: (1) choosing the processing unit model, the coupling type and the connectivity topology (modeling and analysis), (2) dealing with parameter tuning, usually solved by optimization methods and gait transition, to handle variation on gaits as type or frequency (modulation) and (3) executing the designed CPG at the software and/or hardware level (implementation).

In this chapter, we focus on locomotion systems for legged robots, which are based on spiking central pattern generators (SCPGs) and reverse engineering methods for automatically design them. The study and implementation of SCPGs as locomotion systems have been barely explored and compared with other CPGs, which are built with oscillators or low-plausible neuron models (see [6, 7, 17] as reference). The SCPGs are built with plausible neuron models known as spiking neurons, models that define the third generation of artificial neural networks [18]; these neuron models naturally receive and send spatio-temporal information as generating rhythmic patterns are required for CPGs. The SCPGs have been designed and implemented as locomotion systems for robotic platforms such as bipeds [19–21, 25], quadrupeds [23, 25] and hexapods [22, 24–27], where the design methodologies used in [19–21, 27] tend to follow the phases proposed in [7], while in [22–26] reverse engineering methods are used. Basically, a reverse engineering method to design SCPG-based locomotion systems for robotic platforms uses either deterministic or stochastic optimization methods, which, given an input set of discretized rhythmic signals and a fixed spiking neuron model, are capable of defining a spiking neural network (SNN), including both synaptic connections and weights, that endogenously and periodically replicates the input set of discretized rhythmic signals, which contribute to locomotion of a robotic platform. Herein, we present a generalization of reverse engineering methods to design SCPG-based locomotion systems for robotic platforms based on details of implementations of reviewed works.

2. Robotic platforms and controllers

Nowadays, there are a variety of robotic platforms, and each of them has particular technical specifications in design, displacement ways and so on. In reviewed works, for real implementations

of SCPG-based locomotion systems, three types of legged robotic platforms have been particularly used such as hexapod, quadruped and biped robots. Particularly, the used robotic platforms have 3 degrees of freedom (DOFs) or servomotors per leg, that is, the hexapod has 18 DOFs, the quadruped has 12 DOFs and the biped has 6 DOFs. Although for the hexapod and the quadruped, just two DOFs per leg were used; the two are closer to the body of robots and directly related to the movement of the robot. **Figure 1** shows the robotic platforms with a specific label for identifying the position of their respective servomotors.

Servomotors in the robotic platforms are handled by a processing unit, which in reviewed works, a SCPG is embedded into them to provide a locomotion mechanism to the robots. In **Figure 2**, we present the electronic boards, which have been used as processing units such as

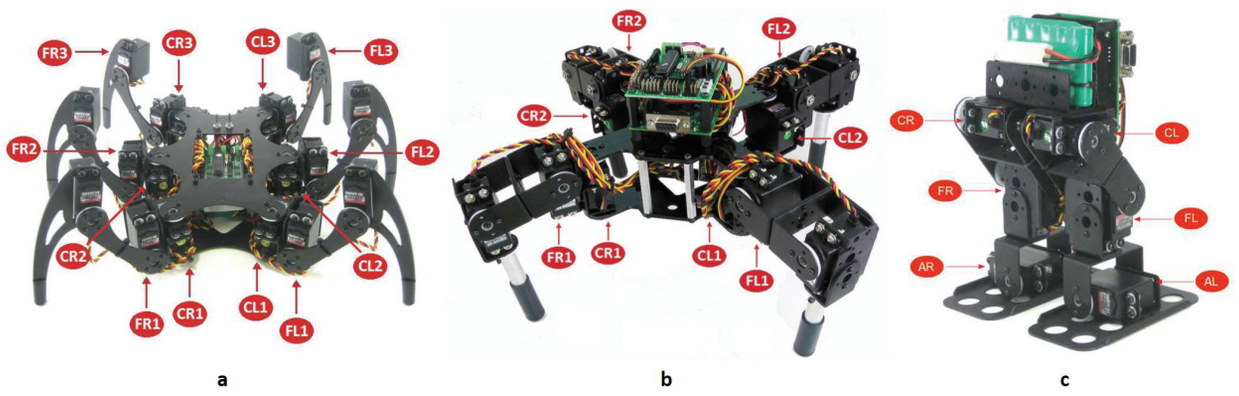

Figure 1. Robotic platforms from Lynxmotion company. (a) Phoenix hexapod robot model, (b) symmetric quadruped robot model and (c) Brat biped robot model. In ovals are marked and labeled the servomotors used in their locomotion where letters C, F and a stand for Coxa, femur and ankle, respectively, letters L and R represent left and right sides, and numbers mark the number according to their position (taken from [25]).

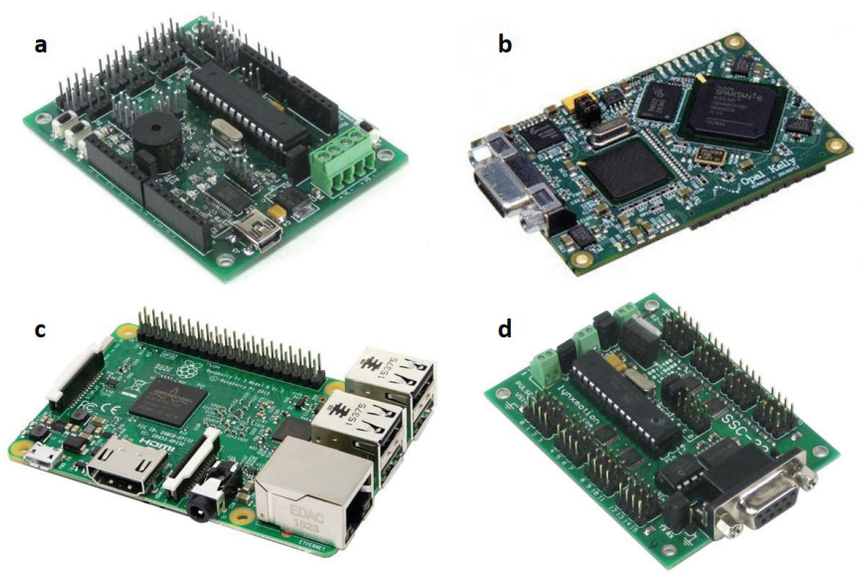

Figure 2. Boards for robot control. Processing units: (a) Arduino board, BotBoarduino for Lynxmotion robots, (b) FPGA Spartan 6 XEM6310-LX45 board from OpalKelly, and (c) Raspberry Pi 3 Model B board. Servo controller: (d) SSC-32 board.

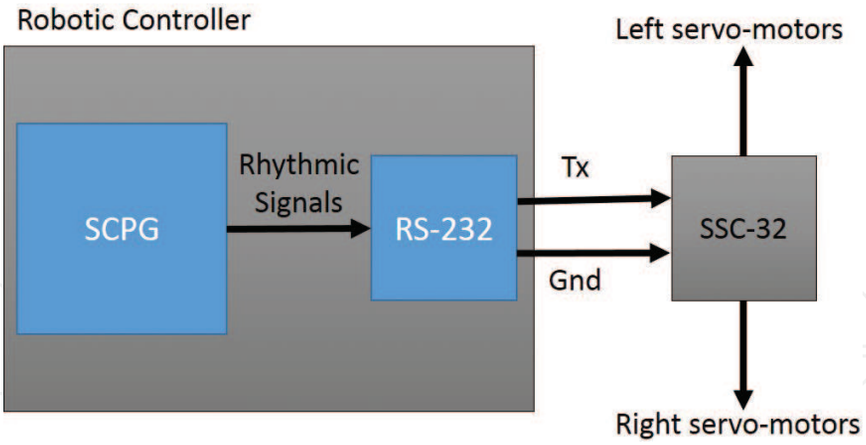

Figure 3. System block diagram of robotic controller coupled with servomotors of robotic platforms through a servo controller.

Arduino (**Figure 2a**), Field Programmable Gate Array (FPGA) (**Figure 2b**) and a Raspberry Pi 3 Model B (**Figure 2c**) boards. Also, a servo controller board (**Figure 2d**) is required to send the output of the processing units to the servomotors of the robotic platforms.

Basically, the integration of the processing boards and platforms works as follows: an embedded SCPG into a processing board generates rhythmic signals, which are sent to the legs through a servo controller. This converts the spiking activity generated by the SCPGs into a control signal (voltage). The transmission process is carried out by using the RS-232 communication protocol. **Figure 3** shows a block diagram of this integration.

Thus, an important aspect to achieve locomotion in robotic platforms is to design a SCPG according to the capabilities of the processing board, which, for reviewed works, must exactly replicate and periodically generate specific rhythmic patterns. In Section 3, we describe in detail the functionality of SCPGs and methods used for designing them.

3. SCPG-based locomotion system

The SCPGs, reviewed in this chapter, can generate endogenously discrete rhythmic signals; in other words, each periodical signal of a locomotion gait is represented by means of a spike train with spike times occurring periodically. This idea was firstly presented by Rostro-Gonzalez et al. in [22], where SCPGs built with discrete spiking neurons (Section 3.2.1) were automatically designed by using a deterministic reverse engineering method (Section 4.1) to imitate walking forms of a stick insect; based on steps sketches of walking forms of stick insect reported in [28], Rostro proposed three sets of discrete rhythmic signals as locomotion gaits (Section 3.1) to achieve hexapod robot locomotion by means of designing one SCPG for each of them.

In **Figure 4**, Rostro-Gonzalez's approach to achieve locomotion for six-legged robots through discrete events over time by means of SNNs is schematized. In **Figure 4a**, a walking form of stick insect reported in [28] is presented; black rectangles represent a leg on ground, while white ones represent a leg in the air. The L1 row is marked with dotted rectangle to exemplify

http://dx.doi.org/10.5772/intechopen.72348

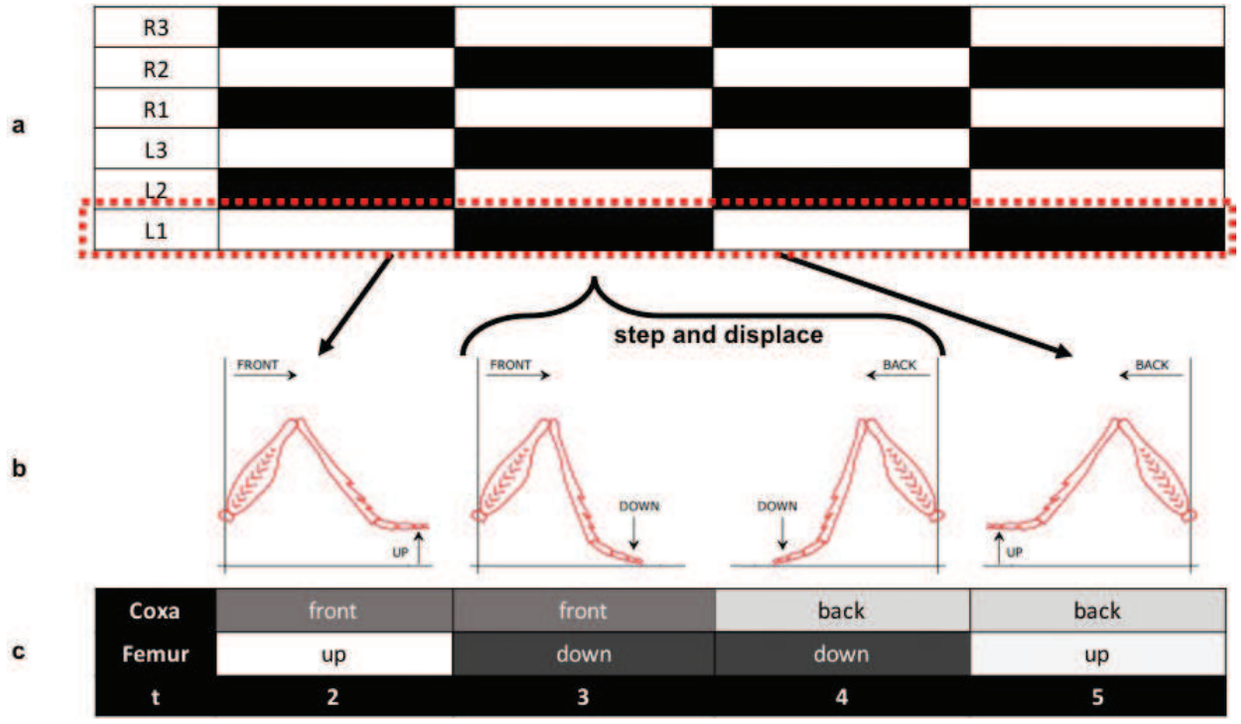

Figure 4. Representation of extrapolation of steps observed of hexapod insect into discrete rhythmic signals for SCPG-based locomotion design. (a) Walking form of stick insect reported in [28], (b) extrapolation of a step into position of a leg over time and (c) proposal of a step (femur row) with additional information (coxa row) represented as spike trains, in each row, darker rectangles represent a spike and lighter ones indicate the absence of spike.

how a step of real hexapod insect is interpreted and extended to a step of robot hexapod. Notice in **Figure 4b** that leg sketch coincides with a black rectangle in **Figure 4a** as leg is on the ground, and at this point the leg displacement that contributes to whole walking action occurs. **Figure 4c** shows the rhythmic signals over time to move a leg according to locomotion gait in **Figure 4a**; the coxa moves to front with spike events and to back without spike events, while the femur moves to down with spike events and to up in the absence of spike events. Finally, the combination of such movements according to the presence and absence of spikes of all legs provokes locomotion of legged robot.

Later, Espinal et al. generated SCPG-based locomotion systems for quadruped and hexapod robots based on Rostro-Gonzalez's idea by using the same discrete spiking neuron model (Section 3.2.1) and a stochastic reverse engineering (Section 4.2) for designing them. There were designed and implemented SCPG-based locomotion systems for quadruped robots in [23] and hexapod robots in [24]; the difference with the Rostro's work is that more compact SNN topologies were achieved, and it was achieved to design a single SCPG capable of generating the three original locomotion gaits for hexapod robots and was extended for quadruped ones as well.

Lately, SCPG-based locomotion systems for hexapods, quadrupeds and bipeds were designed by Guerra-Hernandez et al. in [25]. In his work, there was proposed a locomotion gait for biped robots following the Rostro's idea, and SNNs were designed by using the discrete spiking neuron model and a variant of stochastic reverse engineering (Section 4.2) proposed by Espinal et al.

Lately, in [26], Perez-Trujillo et al. designed SCPG-based locomotion systems for hexapod robots based on Rostro-Gonzalez's, Espinal's and Guerra-Hernandez's works. The contribution of Perez-Trujillo's work was to create SCPG-based locomotion systems built with a nondiscrete spiking neuron model (Section 3.2.2), and the reverse engineering method was a variant stochastic one (Section 4.2).

The reviewed works are summarized in **Table 1**, including robotic platforms and processing boards as well as reverse engineering method and spiking neuron model used.

Following subsections complement the description of SCPG-based locomotion systems. In Section 3.1, the different rhythmic signal sets reported for each locomotion gait to robotic platforms are shown. Section 3.2 describes the spiking neuron models that have been used to define SNN as SCPGs on robot locomotion.

3.1. Locomotion patterns

The locomotion patterns contain the discrete rhythmic signals for each servomotor of a robotic platform. Each of them serves to define a specific locomotion gait that a SCPG must replicate endogenously and periodically. Besides, they are used for engineering reverse methods (Section 4) to design SCPGs. In **Figures 5–7**, show the different designed discrete rhythmic signals for hexapod, quadruped and biped robots, respectively; for each row corresponds a servomotor with the same label according to the robotic platform.

3.2. Spiking neuron models

3.2.1. Beslon-Mazet-Soula neuron model

The Belson-Mazet-Soula (BMS) neuron [29] is a discrete-time model derived from a well-known spiking neuron, that is, the integrate-and-fire [30] model. The BMS neuron model is defined by Eqs. (1) and (2), and they describe the behavior of the i-th neuron over time k; former equation models its membrane potential V_i, and the last equation defines its firing state Z_i.

Author/Work	Robotic platform	Robotic controller	Spiking neuron model	Reverse engineering method
Rostro et al. [22]	Hexapod	FPGA	BMS Neuron	Deterministic
Espinal et al. [23]	Quadruped	Arduino	BMS Neuron	Stochastic
	Hexapod	FPGA		
Espinal et al. [24]	Hexapod	FPGA	BMS Neuron	Stochastic
Guerra et al. [25]	Hexapod	FPGA	BMS Neuron	Variant stochastic
	Quadruped	FPGA		
	Biped	FPGA		
Perez et al. [26]	Hexapod	Raspberry Pi 3	LIF Neuron	Variant stochastic

Table 1. Summary of Legged Robot Locomotion System Configurations and reverse engineering methods.

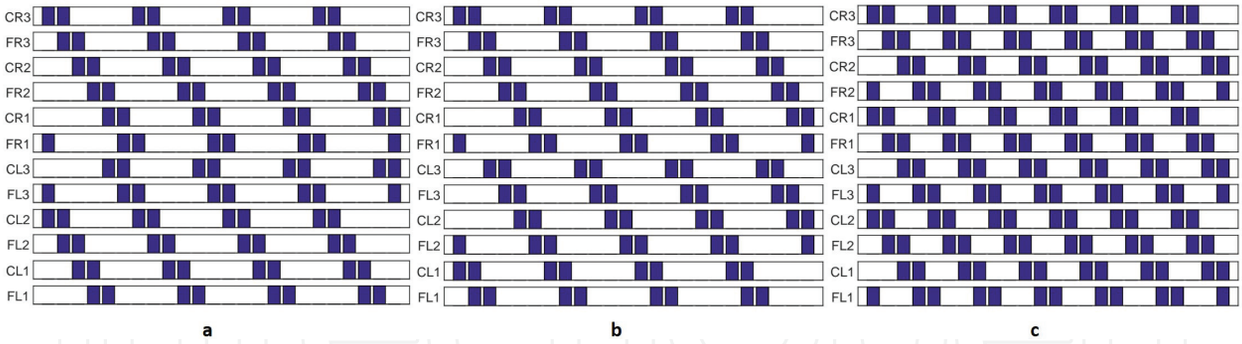

Figure 5. Hexapod robot locomotion gaits proposed in [22] (figures taken from [26]). (a) Walk gait, (b) jog gait and (c) run gait.

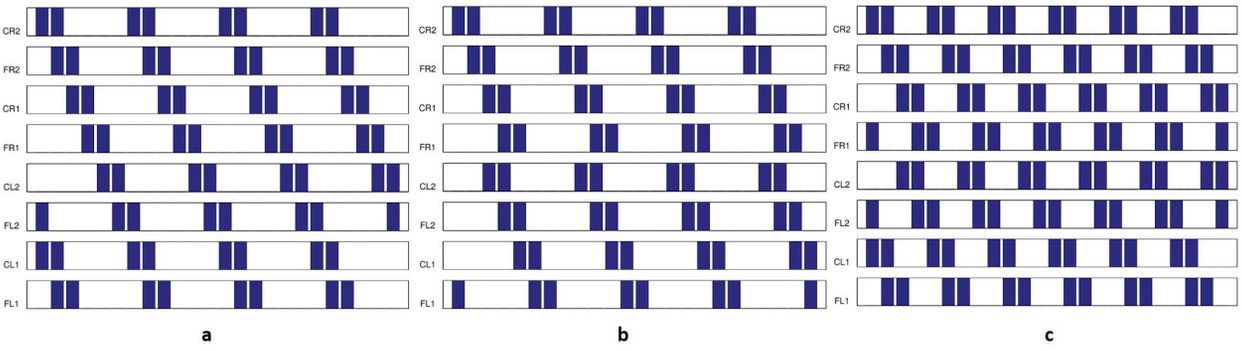

Figure 6. Quadruped robot locomotion gaits proposed in [23] (figures taken from [26]). (a) Walk gait, (b) jog gait and (c) run gait.

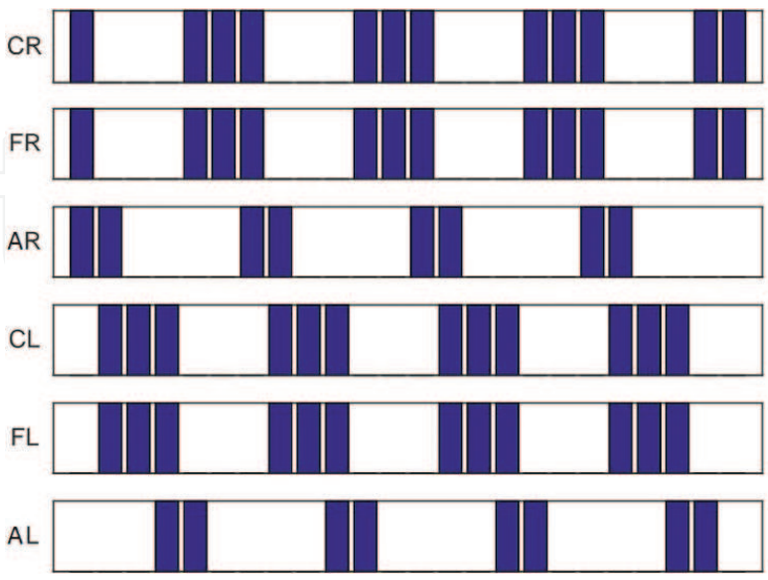

Figure 7. Biped walking locomotion gait proposed in [26].

$$V_i[k] = \gamma V_i[k-1](1 - Z_i[k-1]) + \sum_{j=1}^{N} W_{ij} Z_j[k-1] + I_i^{ext} \tag{1}$$

In Eq. (1), $\gamma \in [0, 1]$ represents the leaky factor. The number of spiking neurons into the neural network is given by N. The synaptic strength weights are given by W_{ij}. The I_i^{ext} is either a varying or constant external stimuli; due to that, CPGs endogenously produce periodic patterns $I_i^{ext} = 0$.

$$Z_i[k] = \begin{cases} 1 & if\ V_i[k] \geq \theta \\ 0 & otherwise \end{cases} \tag{2}$$

For Eq. (2), the fixed firing threshold is given by θ. Eq. (2) is used in Eq. (1) for tracking spike occurrence ($Z_i[k]$) and resetting the membrane potential of i-th neuron $(1 - Z_i[k])$.

3.2.2. Integrate-and-fire neuron model

The integrate-and-fire (I&F) neuron [30], basically, models the evolution of its membrane potential's state over time as a resistor-capacitor (RC) circuit. In particular, the current-based leaky integrate-and-fire (LIF) model, or "if_curr_exp" model in the PyNN library [31], is a LIF neuron with a fixed firing threshold and exponentially decaying postsynaptic conductance given in Eq. (3); besides, the model requires of *tau_refrac* to define the refractory value and *v_thresh* to set the fixed firing threshold.

$$\frac{dv}{dt} = \frac{ie + ii + i\_offset + i\_inj}{c\_m} + \frac{v\_rest - v}{tau\_m} \tag{3}$$

In Eq. (3), the membrane potential is represented with v. The excitatory and inhibitory current injections *ie* and *ii* are modelled by differential equations in Eqs. (4) and (5), respectively. The *i_offset* stands for a base input current, and *i_inj* is an external current injection; both added each timestep, but $i\_inj = 0$ due to the nature of CPGs.

$$\frac{die}{dt} = -\frac{ie}{tau\_syn\_E} \tag{4}$$

$$\frac{dii}{dt} = -\frac{ii}{tau\_syn\_I} \tag{5}$$

Finally, tau_syn_E and tau_syn_I, in Eqs. (5) and (6), are excitatory and inhibitory input current decay time constant.

4. Reverse engineering methods for designing SCPGs

In this section, we describe reverse engineering methods for automatically design SNNs by defining both their topology and synaptic weights. The reviewed methods can be generalized in diagram shown in **Figure 8**.

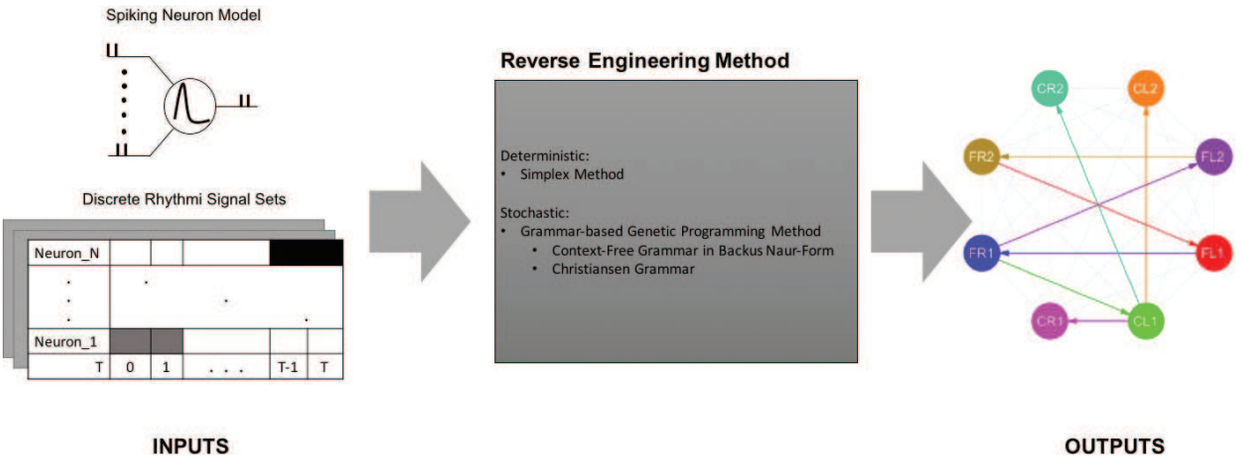

Figure 8. Diagram of reverse engineering method for designing SCPG.

Basically, the generalization defines an input-process-output system where its inputs are a fixed spiking neuron model (Section 3.2) and one or more sets of discrete rhythmic signals (Section 3.1), and the process is defined according to an optimization method, which can be deterministic (Section 4.1) or stochastic (Section 4.2). Finally, the output is generally a partially connected, directed and weighted graph that defines all aspects of a SNN to behave as a SCPG.

Next, the optimization is briefly described, and their strengths and weaknesses are pointed out.

4.1. Deterministic method: Simplex method

This method, originally proposed by Rostro in [34], was created to build SNNs to replicate recorded biological neural dynamics. It was developed to work with the BMS spiking neuron model (Section 3.2.1). It is described in [22] as follows:

First, Eq. (1) must be rewritten in form expressed in Eq. (6).

$$V_i[k] = \sum_{j=1}^{N} W_{ij} \sum_{\tau=0}^{\tau_{ik}} \gamma^{\tau} Z_j[k-\tau-1] + I_{ik\tau} \tag{6}$$

where $I_{ik\tau} = \sum_{\tau=0}^{\tau_{ik}} \gamma^{\tau} I_i^{(\text{ext})}[k-\tau]$ with: $\tau_{ik} = k - \arg\min_{l>0} \{Z_i[l-1] = 1\}$; see [34] for derivation details. With Eqs. (1) and (2), a linear programming system can be formulated to determine the synaptic connections and weights of SNNs, which replicates locomotion gaits represented as rhythmic spiking dynamics through the evolution of all $V_i[k]$ (the membrane potential of each spiking neuron into the network), which are not known. Now, we define the expression: $Z_i[k] = 0 \Leftrightarrow V_i[k] < \theta$ and $Z_i[k] = 1 \Leftrightarrow V_i[k] \geq \theta$; where $\theta = 1$ for simplification. Last expression can be written as next inequality: $Z_i[k] = 0 \Leftrightarrow V_i[k] < \theta$.

By substituting Eq. (6) in the last expression, we can get a linear programming system [35], given the expression in Eq. (7).

$$e_i = A_i w_i + b_i \geq 0 \tag{7}$$

where

- $A_i = \begin{pmatrix} \cdots & \cdots & \cdots \\ \cdots & (2Z_i[k]-1)\sum_{\tau=0}^{\tau_{ik}} \gamma^{\tau} Z_j[k-\tau-1] & \cdots \\ \cdots & \cdots & \cdots \end{pmatrix}$
- $b_i = (\cdots \ (2Z_i[k]-1)(I_{ik\tau}-1) \ \cdots)^t$
- $w_i = \begin{pmatrix} \cdots & W_{ij} & \cdots \end{pmatrix}^t$
- $e_i = (\cdots \quad (2Z_i[k]-1)(V_i[k]-1) \quad \cdots)^t$

Solving the aforementioned formulated linear programming problem in Eq. (7), by using a simple method (or any existing linear programming solver), we obtain the synaptic weights of all neurons and, indirectly, a SNN topology.

Next, the features of this method are listed as follows:

- Strengths:
  - The definition of whole SNN is made by executing the method once.
  - It has been successfully used as a reverse engineering method for designing SNNs, which replicate recorded biological neural dynamics.
- Weaknesses:
  - It can design SNN by using only BMS spiking neuron models.
  - It generates one SNN for replicating just one neural dynamic pattern.

4.2. Stochastic method: Grammar-based genetic programming

For stochastic reverse engineering methods, evolutionary algorithms have been used; particularly, a variant of well-known genetic programming [36] called grammatical evolution (GE) [37]. Practically, GE is an optimization tool that searches approximated optimal solutions by representing them indirectly for a given problem; thus, working with GE to solve a specific problem requires four components: problem instance(s), representation of solutions, a fitness function to evaluate solution's quality and a search engine.

For the SCPG design problem, two types of representations have been proposed: one as a Context-Free Grammar (CFG) in Backus-Naur Form (BNF) and another as a Christiansen Grammar (CG) to use GE (in [25, 26]) and a variant called Christiansen Grammar Evolution (CGE) [38] (in [23, 24]), respectively. In general, both representations describe languages that define the presynaptic connectivity (including weights) of a specific spiking neuron; the common structure

of expected words of two grammars is as follows: $\overbrace{id_1, weight_1}^{1st\ configured\ synapse} |\cdots| \overbrace{id_n, weight_n}^{n-th\ configured\ synapse}$. The connectivity defined by a word has at least one connection and a maximum of connections according to number of neurons into the SNNs. The main difference between both representations is that words of CG representation are syntactically and semantically correct, this means that any generated word is valid and there are not repeated indexes, while words of CFG BNF are just syntactically correct or any generated word is just valid.

The fitness function is usually defined by the problem; to solve this, three fitness functions based on SPIKE [39] (used in [23, 24, 26]) and Victor-Purpura distances [40] (used in [25]) to compare similarity between generated spike train and target spike train to guide the search process have been proposed. Basically, first and second distance-based fitness functions are for generating one SCPG per locomotion gait; the difference is that the second one looks for minimal presynaptic connectivity and the first one does not care about number of presynaptic connections. The third distance-based fitness function allows to generate a single SCPG, which can replicate different locomotion gaits.

The search engine in GE is usually a metaheuristic algorithm, which tries to improve the quality of solutions. For reviewed works, three different algorithms have been used: Univariate Marginal Distribution Algorithm [40] (used in 23), (1 + 1)-Evolution Strategy [41] (used in [24]) and Differential Evolution [42] (used in [25, 26]).

For implementation details, see [23–25]. Next, the features of this method are listed:

- Strengths:
  - It can design SNNs which use either BMS spiking neuron model or LIF neuron model.
  - It can handle design criteria to design compact SNN topologies or SNN, which can replicate different neural dynamics or locomotion gaits.
- Weaknesses:
  - The process must be executed several times to build a single SNN; due to that, it defines synaptic connection and weights one neuron at time.
  - It has not been tested on other design problems than design SCPGs.

5. Discussion and conclusion

Nowadays, autonomous robot locomotion is still a valid problem that has been partially solved in robotics. Particularly, locomotion of nonwheeled robotic platforms is a problem highly susceptible for trying to be solved by means of bioinspired algorithms known as CPGs. However, sometimes working with CPGs may represent a problem itself since its design; this is due to the different choices that must be made before implement the CPG according to [7]. In

this chapter, we have explored researches made on SCPG field, a particular type of CPGs, which have been barely explored to date. The SCPGs are built with spiking neurons, a plausible neuron model, which handle similar information as such observed in biological neural systems. We specifically focus on SCPG designed by approaches that allow to dispense with human experts for explicitly define each CPG design phase. These kinds of works use reverse engineering methods to solve de SCPG design problem as an optimization one. By means of these methods, there are generated weighted and directed graphs as SNNs, which endogenously generate rhythmic discrete signals to allow locomotion of legged robots.

Biological CPGs do not work in isolation; they depend on the information interaction with other parts of the central nervous system [32]; even, external afferent inputs are used to shape their outputs [33]. Based on the aforementioned, the next step of SCPG-based locomotion systems could be their integration in navigation systems to endow them with sensors and can build more robust and plausible bioinspired algorithms.

Finally, there are other reasons to keep studying and implementing SCPGs, which go far beyond the locomotion of nonwheeled robots, their possible application in other areas, like medicine in developing prosthetic robotic devices for patients with spinal damage or amputated limbs [20].

Acknowledgements

The authors wish to thank the Consejo Nacional de Ciencia y Tecnología (CONACyT) for the support through the Computational Neuroscience: Theory of Neuromorphic Systems Development project, N. 1961, the University of Guanajuato and National Technology of Mexico. Horacio Rostro-Gonzalez wishes to thank the University of Guanajuato for the support provided through a sabbatical year.

Author details

Andrés Espinal[1]*, Marco Sotelo-Figueroa[1], Héctor J. Estrada-García[2], Manuel Ornelas-Rodríguez[3] and Horacio Rostro-Gonzalez[4,5]

*Address all correspondence to: aespinal@ugto.mx

1 Department of Organizational Studies, DCEA-University of Guanajuato, Guanajuato, Mexico

2 Department of Electrical Engineering, DICIS-University of Guanajuato, Salamanca, Mexico

3 Division of Postgraduate Studies and Research, ITL-National Technology of Mexico, Leon, Mexico

4 Department of Electronic Engineering, DICIS-University of Guanajuato, Salamanca, Mexico

5 Neuroscientific System Theory, Department of Electrical and Computer Engineering, Technical University of Munich, Munich, Germany

References

[1] Floreano D, Ijspeert AJ, Schaal S. Robotics and neuroscience. Current Biology. 2014; **24**(18):R910-R920. DOI: 10.1016/j.cub.2014.07.058

[2] Brown TG. The intrinsic factors in the act of progression in the mammal. Proceedings of the Royal Society of London. Series B, containing papers of a biological character. 1911; **84**(572):308-319. DOI: 10.1098/rspb.1911.0077

[3] Brown TG. On the nature of the fundamental activity of the nervous centres; together with an analysis of the conditioning of rhythmic activity in progression, and a theory of the evolution of function in the nervous system. The Journal of Physiology. 1914;**48**(1):18-46. DOI: 10.1113/jphysiol.1914.sp001646

[4] Marder E, Bucher D. Central pattern generators and the control of rhythmic movements. Current Biology. 2001;**11**(23):R986-R996. DOI: 10.1016/S0960-9822(01)00581-4

[5] Patel LN. Central pattern generators: Optimisation and application. In: Chiong R, editor. Nature-Inspired Algorithms for Optimisation. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. pp. 235-260. DOI: 10.1007/978-3-642-00267-0_8

[6] Ijspeert AJ. Central pattern generators for locomotion control in animals and robots: A review. Neural Networks. 2008;**21**(6):642-653. DOI: 10.1016/j.neunet.2008.03.014

[7] Yu J, Tan M, Chen J, Zhang J. A survey on CPG-inspired control models and system implementation. IEEE Transactions on Neural Networks and Learning Systems. 2014; **25**(3):441-456. DOI: 10.1109/TNNLS.2013.2280596

[8] Barron-Zambrano JH, Torres-Huitzil C. CPG implementations for robot locomotion: Analysis and design. In: Dutta A, editor. Robotic Systems-Applications, Control and Programming. InTech; 2012. pp. 161-182. DOI: 10.5772/25827

[9] Endo G, Morimoto J, Matsubara T, Nakanishi J, Cheng G. Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot. The International Journal of Robotics Research. 2008;**27**(2):213-228. DOI: 10.1177/0278364907084980

[10] Liu C, Chen Q, Wang D. CPG-inspired workspace trajectory generation and adaptive locomotion control for quadruped robots. IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics. 2011;**41**(3):867-880. DOI: 10.1109/TSMCB.2010.2097589

[11] Barron-Zambrano JH, Torres-Huitzil C. FPGA implementation of a configurable neuromorphic CPG-based locomotion controller. Neural Networks. 2013;**45**:50-61. DOI: 10.1016/j.neunet.2013.04.005

[12] Inagaki S, Yuasa H, Suzuki T, Arai T. Wave CPG model for autonomous decentralized multi-legged robot: Gait generation and walking speed control. Robotics and Autonomous Systems. 2006;**54**(2):118-126. DOI: 10.1016/j.robot.2005.09.021

[13] Jia X, Chen Z, Petrosino JM, Hamel WR, Zhang M. Biological undulation inspired swimming robot. In: Okamura A, editor. Robotics and Automation (ICRA), 2017 IEEE International

Conference on 29 May-3 June 2017. Singapore: IEEE; 2017. pp. 4795-4800. DOI: 10.1109/ICRA.2017.7989558

[14] Chung SJ, Dorothy M. Neurobiologically inspired control of engineered flapping fligh. Journal of Guidance, Control, and Dynamics. 2010;**33**(2):440-453. DOI: 10.2514/1.45311

[15] Wang Z, Gao Q, Zhao H. CPG-inspired locomotion control for a snake robot basing on nonlinear oscillators. Journal of Intelligent and Robotic Systems. 2017;**85**(2):209-227. DOI: s10846–016–0373-9

[16] Ding R, Yu J, Yang Q, Tan M. Dynamic modelling of a CPG-controlled amphibious biomimetic swimming robot. International Journal of Advanced Robotic Systems. 2013;**10**(4):11. DOI: 10.5772/56059

[17] Wu Q, Liu C, Zhang J, Chen Q. Survey of locomotion control of legged robots inspired by biological concept. Science in China Series F: Information Sciences. 2009;**52**(10):1715-1729. DOI: 10.1007/s11432-009-0169-7

[18] Maass W. Networks of spiking neurons: The third generation of neural network models. Neural Networks. 1997;**10**(9):1659-1671. DOI: 10.1016/S0893-6080(97)00011-7

[19] Lewis MA, Tenore F, Etienne-Cummings R. CPG design using inhibitory networks. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation. ICRA 2005; 18–22 April 2005; Barcelona, Spain. IEEE; 2005. pp. 3682-3687. DOI: 10.1109/ROBOT.2005.1570681

[20] Russell A, Orchard G, Etienne-Cummings R. Configuring of spiking central pattern generator networks for bipedal walking using genetic algorithms. In: IEEE International Symposium on Circuits and Systems. ISCAS 2007; 27–30 May 2007; New Orleans, LA, USA. IEEE; 2007. pp. 1525-1528. DOI: 10.1109/ISCAS.2007.378701

[21] Russell A, Orchard G, Dong Y, Mihalas S, Niebur E, Tapson J, et al. Optimization methods for spiking neurons and networks. IEEE Transactions on Neural Networks. 2010;**21**(12): 1950-1962. DOI: 10.1109/TNN.2010.2083685

[22] Rostro-Gonzalez H, Cerna-Garcia PA, Trejo-Caballero G, Garcia-Capulin CH, Ibarra-Manzano MA. Avina-Cervantes, et al. a CPG system based on spiking neurons for hexapod robot locomotion. Neurocomputing. 2015;**170**:47-54. DOI: 10.1016/j.neucom.2015.03.090

[23] Espinal A, Rostro-Gonzalez H, Carpio M, Guerra-Hernandez EI, Ornelas-Rodriguez M, Puga-Soberanes HJ, et al. Quadrupedal robot locomotion: A biologically inspired approach and its hardware implementation. Computational Intelligence and Neuroscience. 2016; **2016**:14. DOI: 10.1155/2016/5615618

[24] Espinal A, Rostro-Gonzalez H, Carpio M, Guerra-Hernandez EI, Ornelas-Rodriguez M, Sotelo-Figueroa M. Design of spiking central pattern generators for multiple locomotion gaits in hexapod robots by christiansen grammar evolution. Frontiers in Neurorobotics. 2016;**10**:13. DOI: 10.3389/fnbot.2016.00006

[25] Guerra-Hernandez EI, Espinal A, Batres-Mendoza P, Garcia-Capulin C, Romero-Troncoso R, Rostro-Gonzalez H. A FPGA-based neuromorphic locomotion system for multi-legged robots. IEEE Access. 2017;**5**:8301-8312. DOI: 10.1109/ACCESS.2017.2696985

[26] Newton AJH, Seidenstein AH, McDougal RA, Pérez-Cervera A, Huguet G, Tere M, et al. 26th annual computational neuroscience meeting (CNS*2017): Part 3. BMC Neuroscience. 2017;**18**(Suppl 1):96-176. DOI: 10.1186/s12868-017-0372-1

[27] Cuevas-Arteaga B, Dominguez-Morales JP, Rostro-Gonzalez H, Espinal A, Jimenez-Fernandez AF, Gomez-Rodriguez F, et al. A SpiNNaker application: Design, implementation and validation of SCPGs. In: Rojas I, Joya G, Catala A, editors. 14th International Work-Conference on Artificial Neural Networks (IWANN 2017); June 14–16; Cadiz, Spain. Cham: Springer; 2017. pp. 548-559. DOI: 10.1007/978-3-319-59153-7_47

[28] Grabowska M, Godlewska E, Schmidt J, Daun-Gruhn S. Quadrupedal gaits in hexapod animals–inter-leg coordination in free-walking adult stick insects. Journal of Experimental Biology. 2012;**215**(24):4255-4266. DOI: 10.1242/jeb.073643

[29] Soula H, Beslon G, Mazet O. Spontaneous dynamics of asymmetric random recurrent spiking neural networks. Neural Computation. 2006;**18**(1):60-79. DOI: 10.1162/089976606774841567

[30] Gerstner W, Kistler WM. Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press; 2002. 480 p

[31] Davison A, Brüderle D, Eppler JM, Kremkow J, Muller E, Pecevski D, et al. PyNN: A common interface for neuronal network simulators. Frontiers in Neuroinformatics. 2009;**2**:11. DOI: 10.3389/neuro.11.011.2008

[32] Arena P. The central pattern generator: A paradigm for artificial locomotion. Soft Computing-A Fusion of Foundations, Methodologies and Applications. 2000;**4**(4):251-266. DOI: 10.1007/s005000000051

[33] MacKay-Lyons M. Central pattern generation of locomotion: A review of the evidence. Physical Therapy. 2002;**82**(1):96-83. DOI: 10.1093/ptj/82.1.69

[34] Rostro-Gonzalez H, Cessac B, Viéville T. Parameter estimation in spiking neural networks: A reverse-engineering approach. Journal of Neural Engineering. 2012;**9**(2):026024. DOI: 10.1088/1741-2560/9/2/026024

[35] Bixby RE. Implementing the simplex method: The initial basis. ORSA Journal on Computing. 1992;**4**(3):267-284. DOI: 10.1287/ijoc.4.3.267

[36] Koza JR. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press; 1992. 819 p

[37] Ryan C, Collins JJ, O'Neill M. Grammatical evolution: Evolving programs for an arbitrary language. In: Banzhaf W, Poli R, Schoenauer M, Fogarty TC, editors. Proceedings Genetic Programming: First European Workshop, EuroGP'98 Paris, France, April 14–15, 1998; Berlin, Heidelberg: Springer Berlin Heidelberg; 1998. pp. 83-96. DOI: 10.1007/BFb0055930

[38] Ortega A, De La Cruz M, Alfonseca M. Christiansen grammar evolution: Grammatical evolution with semantics. IEEE Transactions on Evolutionary Computation. 2007;**11**(1): 77-90. DOI: 10.1109/TEVC.2006.880327

[39] Kreuz T, Chicharro D, Houghton C, Andrzejak RG, Mormann F. Monitoring spike train synchrony. Journal of Neurophysiology. 2013;**109**(5):1457-1472. DOI: 10.1152/jn.00873.2012

[40] Simon D. Evolutionary Optimization Algorithms. John Wiley & Sons; 2013. 772 p

[41] Engelbrecht AP. Computational Intelligence: An Introduction. 2nd ed. Weley: Chichester; 2007. 628 p

[42] Feoktistov V. Differential Evolution: In Search of Solutions. New York: Springer; 2006. 195 p

IntechOpen