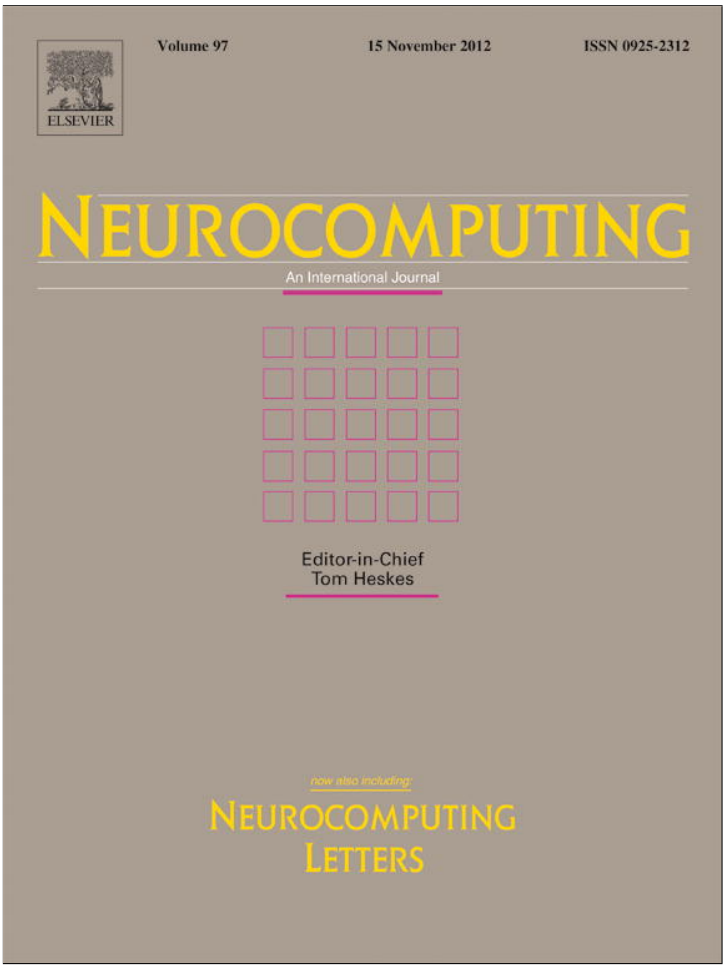

Neurocomputing 97 (2012) 115–124

Contents lists available at SciVerse ScienceDirect

Neurocomputing

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

Phase oscillator neural network as artificial central pattern generator for robots

Pablo Kaluza [a,*], Teodor Cioacă [b]

[a] Abteilung Physikalische Chemie, Fritz-Haber-Institut der Max-Planck-Gesellschaft, Faradayweg 4-6, 14195 Berlin, Germany
[b] Essensys Software Solutions, Copilului Str, Nr. 16, 012178 Bucharest, Romania

ARTICLE INFO

Article history:
Received 28 December 2011
Received in revised form
24 April 2012
Accepted 18 May 2012
Communicated by R. Tadeusiewicz
Available online 6 July 2012

Keywords:
Central pattern generator
Phase oscillators
Neural network
Robot controlling

ABSTRACT

We design an artificial central pattern generator and we exemplify its integration using different prototypic virtual robot models. The artificial central pattern generator is modeled through a network of phase oscillators. Virtual robots are controlled by this central pattern generator using an interface that interpolates stored angular states of their target poses. Finally, we consider control signals coming from the robot's environment to the central pattern generator and to the interface between the central pattern generator and the robot. We show that the control signals can change the dynamics of the system in order to modify the robot's movement patterns. We study three cases: changing the frequency of the pattern sequences, switching between gaits of locomotion and the reorientation of a robotic entity.

1. Introduction

Central pattern generators (CPGs) of biological systems are autonomous networks of neurons that produce endogenously periodic sequences of patterns without needing external rhythmic input [1]. They have a key importance for animals, being responsible for many functions such as locomotion, breathing, and in general automatic movements of the animal body and organs. Models of central pattern generators focused on biological aspects have been proposed mainly for controlling gaits of animals [1–4]. Central pattern generators in the context of artificial neural networks have been developed in terms of recurrent networks with time-dependent recurrent back-propagation for learning, but also by using closed chains of nodes with synchronously updated and excitatory connections [5].

On the other hand, artificial central pattern generators are nowadays particularly considered for robot control [6,7]. In such applications, different models of central pattern generators are proposed. In general, similar to the case of biological systems, these models consist of coupled oscillators that map the states to the body's articulations. Thus, different gaits can be achieved by properly tuning the coupling between these oscillators. In these models there exists a quite important correlation between the size of the neural network and the anatomical structure of the robot. Other models of artificial circuits are used providing a higher level of integration in order to obtain self-organized adaptation in robot behavior [8]. Differently from these models, we present in this work a CPG with characteristics independent from the robot anatomy (robot shape). Additionally, our model is designed in blocks allowing a very flexible adaptation to different robot configurations.

Our aim is to engineer a toy system modeling an artificial neural network of oscillators inspired by biological central pattern generators. Further, we use this network to control the movement of virtual prototypic robots in different environmental scenarios. In particular, we want to describe the system using a system of differential equations. As we will see, the internal continuous dynamics of the CPG are very useful for controlling the robot's movement.

We build a model consisting of three blocks. The first one, the central pattern generator (CPG), is a network of phase oscillators with the ability to encode binary information (e.g. patterns) as phase-locked motion. This approximation is well known in models of associative memory which are the extensions of the Hopfield model [9]. They replace the spins by phase oscillators, falling in a class of Kuramoto models [10–18]. This network of phase oscillators is driven by a pacemaker which activates the different patterns of the sequences as a function of time.

The second part of the model consists of an interface between the central pattern generator and the virtual robot (interface CPG–robot). The function of this block is to map the patterns of

* Corresponding author. Tel.: +49 30 84135104; fax: +49 30 84135106.
E-mail addresses: kaluza@fhi-berlin.mpg.de (P. Kaluza), teodor.cioaca@essensys.ro (T. Cioacă).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2012.05.019

the CPG to the angular states that describe the robot's spatial position. We have found that the dynamical properties of our phase oscillator network are quite useful to interpolate the angular states of the robot. Additionally, the dynamics produces a realistic evolution of the robot movement. Finally, the third part of the model describes the virtual prototypic robots. In this work we consider four different robots in order to implement several kinds of external controlling signals.

Given the characteristic of this model, the integration of external control signals from the robot's environment can be easily implemented. Signals arriving to the central pattern generator or to the CPG–robot interface are considered. We study different possibilities to use these external signals to modify the speed of the sequences, gait switching and change in the direction of the robot's displacement.

This work is organized as follows. In the first section we present the model with its different parts: the central pattern generator (CPG), the CPG–robot interface and the robot operation. Additionally, we show a detailed example for the CPG operation. In the second section, we show the main results. We present key examples of CPG–robot integration and interaction scenarios with the external control signals from the environment. Finally, we present the main conclusion and discussions in the last section.

2. Model

The model consists of three main blocks: the central pattern generator (CPG), the interface between the CPG and the robot body, and the actual virtual robot model. Fig. 1 shows a schematic representation of the model.

The central pattern generator (CPG) has a pacemaker ψ, P stored patterns ${}^{k}\xi$ with phase instants (delays) τ_k, and a retrieval network of N phase oscillators. The retrieval network encodes information as phase differences between the oscillators. The actual state $\xi(t)$ of this network evolves between the set of patterns (ξ) as function of the phase of ψ. That is, when ψ activates a stored pattern k, the state $\xi(t)$ of the retrieval network evolves towards ${}^{k}\xi$ with $k=1,..,P$ (we use a pre-superscript to indicate the index of store pattern).

The CPG–robot interface acts as a transduction device between the actual state $\xi(t)$ of the CPG and the angular state of the robot $\vec{u}(t)$. Each state ${}^{k}\xi$ of the CPG is mapped to an angular state $\vec{u}^{k}$ of the robot body (${}^{k}\xi \rightarrow \vec{u}^{k}$). Thus, the temporal evolution of the robot movements $\vec{u}(t)$ is slaved to the temporal evolution of the CPG $\xi(t)$. The robot is a prototypic virtual device that allows us to study the results of the CPG. The robot states are defined by a vector of angles $\vec{u}(t)$, angles that indicate the actual spatial position of the different members of the robot body. The number of elements of the vector $\vec{u}(t)$ is equal to the number of degrees of freedom A of the robot.

In the following paragraphs we study in detail each of these blocks.

2.1. Central pattern generator

The central pattern generator (CPG) produces periodically sequences of states without needing external rhythm units. External input, however, can be used for control, to triggering different actions or modifying some parameters of the CPG, as we shall describe later on in this work. We design a CPG with two blocks as Fig. 1 shows. It has a pacemaker ψ, a retrieval network with N phase oscillators, and P memorized patterns ${}^{i}\xi$ with phase instants τ_i ($i \in 1,\ldots,P$). The main idea behind this architecture is to control the retrieval network dynamics by using the pacemaker ψ through the couplings given by the stored patterns and phase

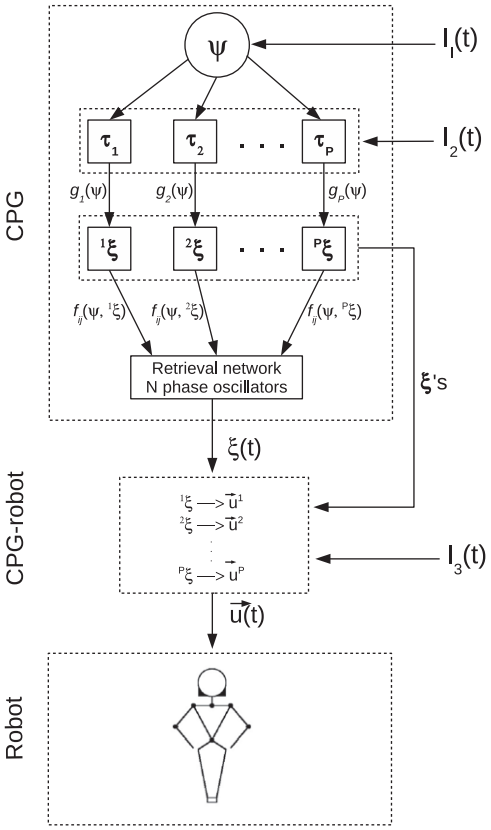

Fig. 1. Schematic representation of the model. The model has three main blocks: the central pattern generator CPG, the CPG–robot interface and the virtual robot. The CPG has a pacemaker ψ, a number of P stored patterns ${}^{k}\xi$ with phase instants τ_k, where $k=1,\ldots,P$, and a retrieval network with N phase oscillators. We consider three external inputs from the environment to the system: $I_1(t)$ and $I_2(t)$ to the CPG, and, $I_3(t)$ to the interface CPG–robot.

instants. This architecture with blocks, having a pacemaker in one of them, has been also proposed for some other biologically inspired central pattern generators [1,4].

The pacemaker ψ has the function of activating the patterns ξ's, one by one as a function of its phase, at phase instants τ_i. Consequently, the pacemaker works as a clock that controls the timing of the system. The retrieval network consists of N globally coupled phase oscillators. Its function is to encode the stored patterns ξ as phase differences between its N oscillators. The retrieval network is a special case of the associative memory proposed by Nishikawa et al.[14]. This network is characterized by an energy function L, which is a function of $N-1$ phase differences. The minima of this function can be located in states where the phase differences are only zero or π (phase-locked dynamics). Thus, the retrieval network has the capacity to encode 2^{N-1} patterns as phase differences. We take as reference the last oscillator N for encoding. Thus, a pattern ξ is encoded as a vector of $N-1$ phase differences $\xi=(\Delta\phi_{1N},\ldots,\Delta\phi_{N-1,N})$ with $\Delta\phi_{ij}=\phi_j-\phi_i$. In the following we label these states by their decimal representation, i.e., for example with $N=3$ oscillators we have $\xi^1=(\pi,0)$ and $\xi^2=(0,\pi)$ (for the decimal label we use post-superscripts).

By construction, when a pattern $\xi^{(s)}$ is activated by the pacemaker ψ at phase instant $\tau_{(s)}$, the function L has only one minimum corresponding to the state $\xi^{(s)}$. Given the gradient dynamics of the retrieval network, the oscillators converge to a phase-locked state with the required phase differences. In this way, the network can retrieve the stored patterns. The system can be seen as an associative memory driven by a pacemaker.

In the following we present the different blocks of the CPG and their dynamical operations.

2.1.1. Retrieval network

The following model of a dynamical system has been partially studied in the context of cluster partitions of phase oscillators networks by the author [19]. However, we present now a more detailed analysis in terms of artificial neural networks of phase oscillators. We use a type of associative memory of N phase oscillators ϕ_i with $i=1,\ldots,N$ and $\phi_i \in [0:2\pi)$. Its energy function L is defined according to

$$L=-\frac{K}{4N}\sum_{i=1}^{N}\sum_{j=1}^{N}(\cos(\phi_j-\phi_i)+f_{ij})^2. \tag{1}$$

In this expression, the constant K indicates the strength of the coupling between the oscillators. Here, f_{ij} are the couplings between the oscillators. The potential L depends only on the phase differences between the oscillators, thus there are only $N-1$ independent variables. Since we use the N oscillator as reference, our independent variables are $\Delta\phi_{iN}=\phi_N-\phi_i$. In the case that all the couplings $f_{ij}=0$, the energy function L has 2^{N-1} minima with equal depth. They are located in the vertices of the hypercube of size π with a vertex in the origin, in the space of phase differences $\{\Delta\phi_{ij}\}$. This case corresponds to a Kuramoto model with double periodicity.

When a pattern $\xi^{(s)}$ has been selected by the pacemaker for retrieval, L must have the only minimum on the corresponding hypercube vertex. To ensure that, the couplings f_{ij} must be properly chosen. The Hebbian rule for storing only a pattern in the network can be written, according to our notation, as

$$f_{ij}(\alpha,\xi^{(s)})=\alpha\left(1-\frac{2}{\pi}\left|\xi_i^{(s)}-\xi_j^{(s)}\right|\right). \tag{2}$$

In this equation, α is a parameter that controls the stability of the fixed points located on the hypercube vertices. For $\alpha>1$, the retrieval network is a special case of the Nishikawa model [14,15,19] where only a pattern is stored and the network has zero-error retrieval for it. In this case, the hypercube vertex corresponding to $\xi^{(s)}$ is the only minimum, the other vertices are at least saddle in one direction in the space of phase differences.

The dynamics of the retrieval network comes from the negative gradient of Eq. (1) as

$$\dot{\phi}_i=\frac{K}{N}\sum_{j=1}^{N}\left(f_{ij}\sin(\phi_j-\phi_i)+\frac{1}{2}\sin 2(\phi_j-\phi_i)\right). \tag{3}$$

When a pattern $\xi^{(s)}$ is selected for retrieval through Eq. (2) in Eq. (3), the actual state of the system $\xi(t)$ evolves to the corresponding vertex of the hypercube, independent from the initial conditions, since there exists only one stable fixed point of the dynamics.

2.1.2. Pacemaker ψ

We have shown that it is possible to tune the couplings of the retrieval network in order to have only one minimum in the potential L. So thus, the system can retrieve the desired selected pattern $\xi^{(s)}$ starting from any initial condition. Our aim is to make the coupling time-dependent, so that, we can select different patterns as a function of time.

The pacemaker controls the time dependence of the coupling f_{ij}. It is modeled by a phase oscillator ψ ($\psi\in[0:2\pi)$) with a natural frequency ω. Its dynamics are given by

$$\dot{\psi}=\omega. \tag{4}$$

The states ξ's are retrieved as a function of the phase of ψ. When a pattern $\xi^{(s)}$ must be activated at phase instant $\tau_{(s)}$, during a phase fraction B, the corresponding couplings $f_{ij}(\psi)$, according to the selection rule 2, are expressed as

$$f_{ij}(\psi,\xi^{(s)})=\alpha\left(1-\frac{2}{\pi}\left|\xi_i^{(s)}-\xi_j^{(s)}\right|\right)g_{(s)}(\psi). \tag{5}$$

In this equation

$$g_{(s)}(\psi)=\Theta(\psi-\tau_{(s)})-\Theta(\psi-B-\tau_{(s)}). \tag{6}$$

Here, $\Theta(x)$ is a step function ($\Theta(x)=0$ if $x<0$, and $\Theta(x)=1$ if $x>0$). Then, the function $g_{(s)}(\psi)$ is a unit pulse which is different from zero in the interval between $\psi=\tau_{(s)}$ and $\psi=\tau_{(s)}+B$.

When there are P stored patterns in the central pattern generator, the general form for the couplings becomes

$$f_{ij}(\psi)=\alpha\sum_{k=1}^{P}\left(1-\frac{2}{\pi}\left|{}^{k}\xi_i-{}^{k}\xi_j\right|\right)g_k(\psi). \tag{7}$$

In this way, the pacemaker establishes, for a period B of its cycle, a specific pattern that is to be reached by the dynamics of the retrieval network. The time-dependent coupling can be seen as a set of high order synapses. The synapses between the oscillators of the retrieval network are modulated by a third neuron, the pacemaker. Models of high order synapses have been already proposed for artificial neural networks [20,21].

2.1.3. CPG operation

In order to show the operation of this model, we construct a system with $N=6$ oscillators in the retrieval network and we store $P=4$ patterns. The stored patterns have been randomly chosen among the $2^5=32$ possible patterns that can be encoded by the retrieval network. They are the patterns ${}^{1}\xi^{21}$, ${}^{2}\xi^{10}$, ${}^{3}\xi^{5}$ and ${}^{4}\xi^{18}$ and are used in our system to play the sequence ${}^{1}\xi^{21}\rightarrow{}^{2}\xi^{10}\rightarrow{}^{3}\xi_5\rightarrow{}^{4}\xi^{18}$. We select, for simplicity, the phase instants for these patterns according to $\tau_k=2\pi(k-1)/4$ (with $k=1,\ldots,P$). Each pattern is activated during a phase period $B=\pi/4$. The pacemaker has $\omega=1$.

The system evolves according to the dynamics equation Eq. (4) for the pacemaker, and the couplings from Eq. (7). However, the dynamics of the retrieval network (Eq. (3)) need to have a new term with white noise $\eta(t)$ of intensity T as follows:

$$\dot{\phi}_i=\frac{K}{N}\sum_{j=1}^{N}\left(f_{ij}\sin(\phi_j-\phi_i)+\frac{1}{2}\sin 2(\phi_j-\phi_i)\right)+T\eta_i(t). \tag{8}$$

The retrieval network must operate with noise in order to allow the dynamics to escape from the unstable fixed points, otherwise the system stalls in such points and it is not possible to perform sequences of patterns.

A detailed study of the effect of the noise in systems of phase oscillators with gradient dynamics can be found in Ref. [22]. With noise, the state of the system $\xi(t)$ does not evolve to the stable fixed point $\xi^{(s)}$ and the phase-locked synchronization breaks down. However, $\xi(t)$ is moving around $\xi^{(s)}$ if the noise intensity is small and the oscillators form a well-localized *cloud* around their average position $\xi^{(s)}$. The width of this cloud is determined by the intensity of the noise T. Consequently, in the case of a low noise intensity, the phase differences have well defined values and the correct identification of the selected pattern is achieved. On the other hand, with large noise intensity it does not exist a well defined cloud around the equilibrium $\xi^{(s)}$ and it is not possible to recognize the pattern anymore. Therefore, a small intensity of noise does not change the stability properties of the system. Characteristic to living system, such a behavior is similar to the tremor exhibited in the limbs of animals. Filtering out unwanted tremor components is not the focus of this work, but progress in the field has dealt with this issue [23,24]. Finally, we have chosen a small noise intensity in order to perform a proper

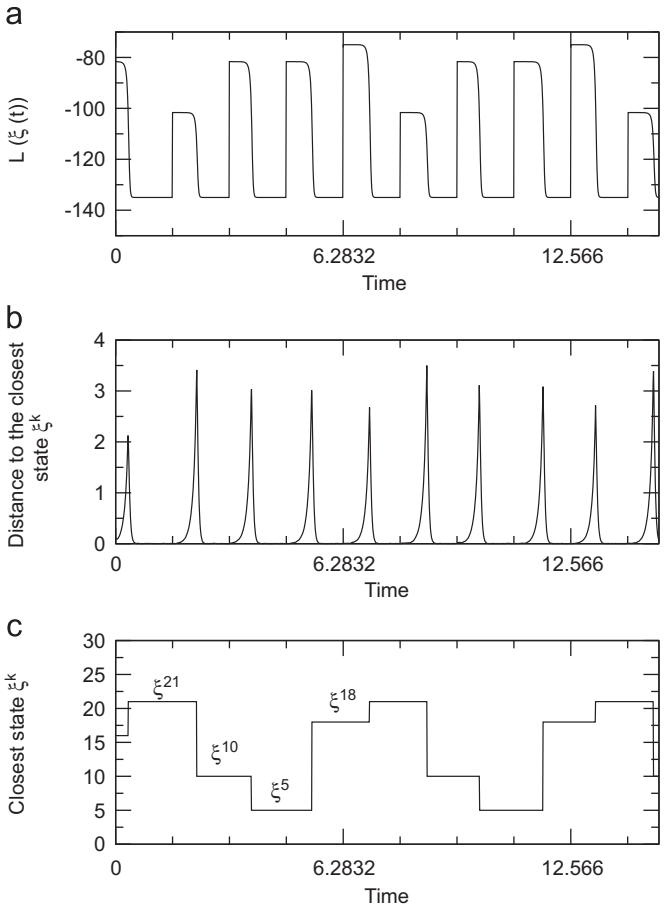

Fig. 2. Sequence ${}^1\xi^{21} \rightarrow {}^2\xi^{10} \rightarrow {}^3\xi^{5} \rightarrow {}^4\xi^{18}$. (a) Energy function L evaluated on the actual state $\xi(t)$ as a function of time. (b) Euclidean distance in the phase difference space between the actual state $\xi(t)$ and the closest corner of the hypercube as a function of time. (c) Closest state (smaller Euclidian distance) as a function of time.

pattern recognition, and allowing the system to simultaneously scape from the unstable fixed points.

The model has been integrated using a stochastic second order Runge Kutta algorithm [25] with $\Delta t = 0.01$ and white noise with intensity $T=0.01$. For this example we use $K=5$ and $\alpha = 2$.

Fig. 2 shows the temporal evolution of the system. Fig. 2(a) shows the actual energy $L(\xi(t))$ of the system as a function of time. We observe that when a state ${}^k\xi$ is activated at phase instant τ_k, $L(\xi(t))$ increases its value since the actual state $\xi(t)$ is no longer the minimum. After that, the system evolves slowly since it must escape from the unstable fixed point. When it moves relatively far from the fixed point, the dynamics is much faster and the system finds the global minimum corresponding to the selected pattern ${}^k\xi$. Then, the dynamics is again slow since the system is reaching the stable fixed point of the dynamics. This process is repeated by each pattern of the sequence.

Fig. 2(b) shows the euclidean distance in the space of phase differences between the actual state of the system $\xi(t)$ and the closest binary state (on the vertices of the hypercube). We observe that when a state is activated at time τ_k, the distance increases slowly while the system escapes from the unstable fixed point. The distance increases even more until the new selected pattern ${}^k\xi$ becomes the nearest one. Thus, the distance decreases again indicating that the system has reached the selected pattern ${}^k\xi$.

Finally, Fig. 2(c) presents the closest state (smaller euclidean distance) of the system as a function of time. Note that a selected state ${}^k\xi$ becomes the nearest to the actual state $\xi(t)$ after a delay from the phase instant τ_k, since the system must evolve towards it. Additionally, it remains as the closest one when a new state has been selected. This indicates, obviously, that the internal dynamics of the associative memory block given by K and α must be faster than the dynamics of the pacemaker ω. Otherwise, the system does not have enough time to find the request state before a new state is selected. In the same way, the phase period B can be chosen shorter than the phase difference between two consecutive states, with the only condition that it must be longer than the internal dynamics of the retrieval network.

The fact that the internal dynamics of the retrieval network must be faster than the pacemaker indicates the existence of a kind of threshold frequency. High frequencies of the pacemaker ω cannot be followed by the retrieval network and the sequences cannot be perfectly performed. The threshold frequency must be studied as a probabilistic quantity due to the stochastic dynamics of the system. In effect, there is a probability $P(\omega,K,\alpha,T)$ for a sequence to be perfectly performed for a given system (fixed ξ's and τ's). Large values for K and α accelerate the dynamics of the retrieval network. In the same way, high noise intensity T helps the system to escape from the unstable fixed point and accelerate the retrieval process. In this work we do not study the statistical properties of such a function and we have considered in the examples values for these parameters that make this function $P(\omega,K,\alpha,T)$ approximately equals to the unit.

2.2. Robot operation

The configuration of the robot body is encoded as a vector of angles $\vec{u}$ with A elements. Each angle u_i corresponds to a particular articulation that gives the orientation of a body member in space (degree of freedom). A robot gait consists of a periodic temporal sequence of P configurations $\vec{u}^k$ with $k = 1,\ldots,P$.

In order to control a robot, we need to specify a continuous evolution between the angular states $\vec{u}^k$. A naive way for describing the state transition is to use linear interpolation between two consecutive states. Not only the angular path between the states is needed, but also the dynamics $\vec{u}(t)$ to evolve on this path. Robot movements cannot be instantaneous (infinite acceleration). Starting from a state (zero speed), the body moves slowly initially and accelerates moving to the next state. In the same way, it cannot stop suddenly; it must decelerate and reach the next state while achieving the halt speed. As we shall show, the proposed CPG is able to reproduce this interpolation of angles with dynamics that resemble a biological systems.

2.3. Interface CPG–Robot

In order to control the robot movements with the CPG an interface between them is required. Our idea is to generate a map between the actual state of the CPG, $\xi(t)$, and the angular vector $\vec{u}(t)$ corresponding to different robot poses. More specifically, when the actual state of the CPG $\xi(t)$ is one of the stored patterns ${}^k\xi$ (fixed point), the position state $\vec{u}^k$ must be a set of joint angles from the robot's configuration set. In consequence, the numbers of patterns ${}^k\xi$ that the CPG must store is equal to the number of robot states $\vec{u}^k$ played in the gaits sequence.

When the robot moves between two consecutive angular states $\vec{u}^k$ and $\vec{u}^{k+1}$, the actual state of the robot $\vec{u}(t)$ must be a blend of these two states. An usual way to compute this mixing is using a linear interpolation between these angular states and provide the transit dynamics for this path. In the interface CPG–robot the interpolation and the dynamics for robot state $\vec{u}(t)$ are provided by the internal dynamics of the CPG as follows:

$$\vec{u}(t) = \frac{1}{\pi}\sum_{k=1}^{P}({}^k\xi \cdot \xi(t))\vec{u}^k. \tag{9}$$

This equation indicates that when the actual state $\xi(t)$ of the CPG coincides with the pattern ${}^{k}\xi$, their dot product is one, and the angular state $\vec{u}^{k}$ is the actual robot state $\vec{u}(t)$. On the other hand, when $\xi(t)$ is moving in the space of phase differences between the fixed points, its projections on the different states are not null, and the actual state $\vec{u}(t)$ is a combination of the $\vec{u}^{k}$ states. Note that in order to perform the dot product, the CPG–robot interface must have access to the stored patterns ξ's as described in Fig. 1.

However, one question arises: what is the CPG interpolation characteristic of the angular states by using Eq. (9)? Is it linear? In order to respond these questions we have studied numerically how the actual state $\xi(t)$ of the CPG evolves in the phase difference space when it moves between orthogonal patterns ${}^{k}\xi$. In particular, we have considered, for this purpose solely, states with only one element equal to π. They are states of the form ξ^{2^n} with n an integer number.

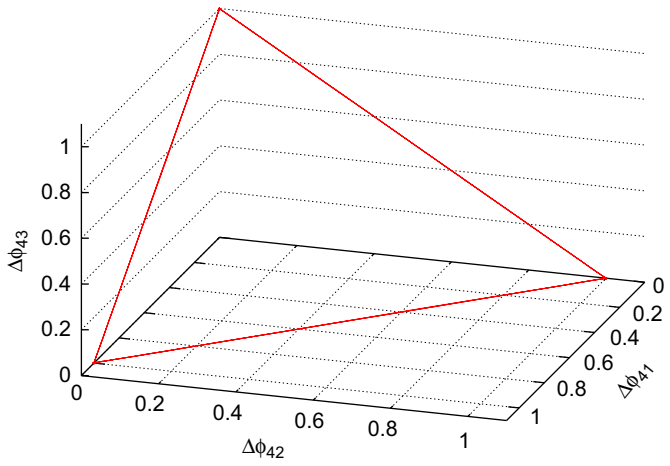

Fig. 3. Evolution of $\xi(t)/\pi$ in the phase space difference for a CPG with $N=4$ oscillators and $P=3$ patterns ${}^{1}\xi^{1}=(\pi,0,0)$, ${}^{2}\xi^{2}=(0,\pi,0)$ and ${}^{3}\xi^{4}=(0,0,\pi)$.

Fig. 3 shows an evolution for a CPG with $N=4$ oscillators in the retrieval network. We have stored three orthogonal patterns ${}^{1}\xi^{1}=(\pi,0,0)$, ${}^{2}\xi^{2}=(0,\pi,0)$ and ${}^{3}\xi^{4}=(0,0,\pi)$. The CPG evolves following the sequence ${}^{1}\xi\rightarrow{}^{2}\xi\rightarrow{}^{3}\xi$ periodically. As Fig. 3 shows we observe that the evolution corresponds to straight paths linking the fixed points (hypercube vertices). This indicates that when the system evolves between two states ${}^{k}\xi$ and ${}^{k+1}\xi$, the actual state $\xi(t)$ is obtained by a linear interpolation between such two states. Note that the simulation is performed with noise, however, its intensity is quite small to note its effects in the trajectory.

We have found that the same behavior is found in systems with higher dimensionality (larger number of oscillators) if we store orthogonal patterns of the form ξ^{2^n} in the CPG. In general

$$R(t)=\frac{1}{\pi}\sum_{i=1}^{N-1}\xi_i(t)=1. \tag{10}$$

$R(t)$ is an order parameter for this system. Eq. (10) indicates that the evolution of $\xi(t)$ occurs on a hyperplane of dimension $N-2$ that intersects the axes at π in the space of phase differences.

The dynamics of the CPG does not only induce a linear interpolation between states, it also produces a particular trajectory for the robot. While the CPG starts to escape from an unstable fixed point because of the noise (see Fig. 2), the dynamics is very slow (as discussed previously). When the actual state $\xi(t)$ is far from the initial unstable fixed point the system moves faster and accelerates (faster movements of the robot). Finally, when $\xi(t)$ is reaching the next stable fixed point, the dynamics is again slow, and the robot stops its movement.

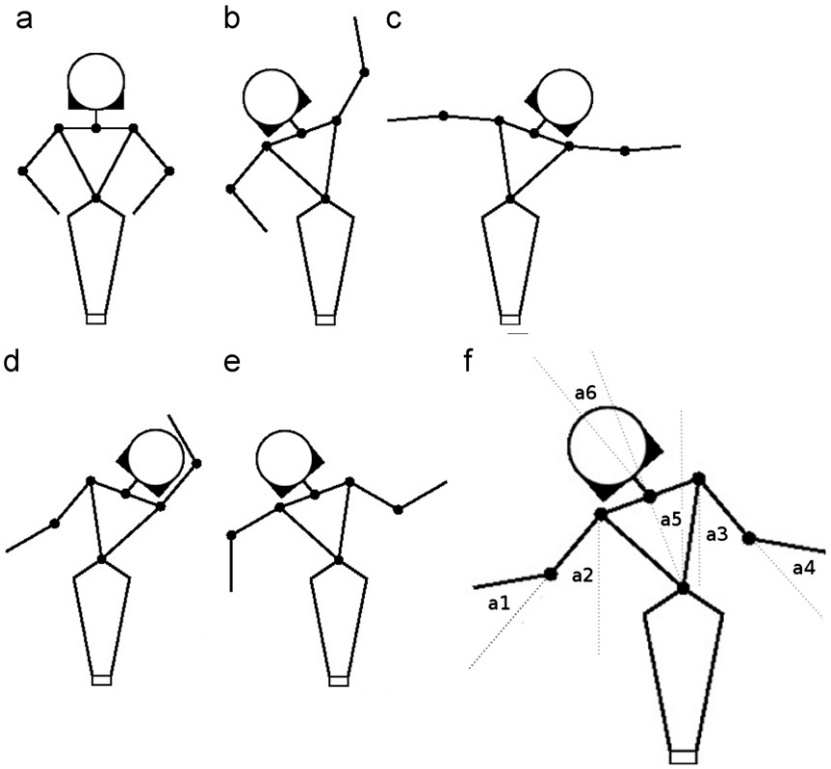

Fig. 4. Dancing robot. (a) Pattern ${}^{1}\xi^{1}\rightarrow\vec{u}^{1}=(-80,40,40,-80,0,0)$. (b) Pattern ${}^{2}\xi^{2}\rightarrow\vec{u}^{2}=(-80,40,150,40,20,20)$. (c) Pattern ${}^{3}\xi^{4}\rightarrow\vec{u}^{3}=(-10,95,85,10,-20,-20)$. (d) Pattern ${}^{4}\xi^{8}\rightarrow\vec{u}^{4}=(20,40,140,70,-20,-20)$. (e) Pattern ${}^{5}\xi^{16}\rightarrow\vec{u}^{5}=(-60,60,60,60,20,20)$. (f) Generic angle vector $\vec{u}^{k}=(a1,a2,a3,a4,a5,a6)$. Body angles are expressed in degrees.

Note that according to the interface operation, the robot angular state $\vec{u}(t)$ is coupled to the dynamics of the CPG $\xi(t)$. For this reason, the noise coming from the stochastic dynamics is also translated to the robot movements. A more realistic model of interface could filter this noise.

3. Results

In this section we present four examples of virtual robots where we study different aspects of this model and the possibility of controlling them by integration of external signals from the robot's environment.

3.1. Dancing robot

The first example of a prototypic robot is a "dancing robot" that performs a sequence of $P=5$ key poses (Fig. 4(a)–(e)). It has six degree of freedom $A=6$ as Fig. 4(f) shows. Since we have five configurations, the CPG has $N=6$ oscillators in the retrieval network. The other parameters of the CPG are: $\omega=1$, $\alpha=6$, $K=1$, $B=2\pi/5$ and phase instants $\tau_k=2\pi(k-1)/5$ with $k=1,..,P$.

Fig. 5 shows the temporal evolution of the system. Supplementary material video1 shows the temporal evolution of the robot as a short animation clip. We observe in Fig. 5(a) the energy function $L(\xi(t))$ as a function of time. When a pattern is selected to be the minimum of the system, L increases its value since the actual state of the CPG is no longer a suitable candidate. Following this situation, $\xi(t)$ moves to the selected minimum and L decreases to the minimum value. This process is repeated for the different states as long the system is running.

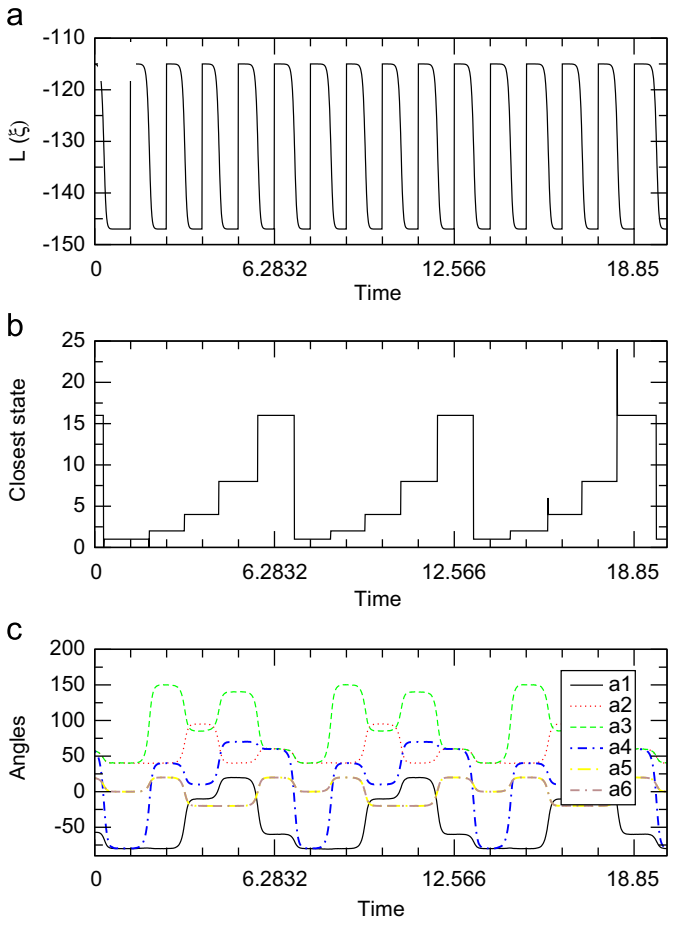

Fig. 5. Temporal evolution of the dancing robot. (a) Energy function L as a function of time evaluated in the actual state $\xi(t)$ of the CPG. (b) Closest state ${}^{k}\xi$ to the actual state $\xi(t)$ as a function of time. (c) Angular state $\vec{u}(t)$ as a function of time.

Fig. 5(b) shows the closest state of the CPG with respect to the corners of the hypercube as a function of time. The state of the CPG $\xi(t)$ evolves among the stored orthogonal states ${}^{1}\xi^{1}$, ${}^{2}\xi^{2}$, ${}^{3}\xi^{4}$, ${}^{4}\xi^{8}$ and ${}^{5}\xi^{16}$ periodically as a function of time. Fig. 5(c) presents the elements of the vector position $\vec{u}(t)$ of the robot as a function of time. We observe the slow acceleration and deceleration at instants when the patterns are changed by the pacemaker. Between these points, the movements are faster, as we require for a realistic dynamics.

The dynamics of this system is stochastic, thus, the time needed by the system to escape from an unstable fixed point is different for each newly retrieved pattern. As a consequence, each cycle of the system is different from the others, and the robot, obviously, follows a stochastic trajectory.

In order to generate any different sequence from these patterns, we can mention two ways. The first one consists in creating a new different map between the stored patterns ${}^{k}\xi$ of the CPG and the angular states $\vec{u}^{k}$. However, this possibility cannot be studied in this work since we do not expand and adapt Eq. (9), in this purpose, by an artificial network. The second possibility consists in the assignation of new τ values for the stored patterns. This second way is particularly simple to implement in our CPG model, since we do not need to change the network architecture in any way. In the next example of a virtual robot, we show the implementation of this second possibility.

Finally, we can mention the possibility to integrate the first input control signal $I_1(t)$. Consider the case when we need to change the frequency ω of the CPG. In such a case, we can make the natural frequency of the pacemaker a function of $I_1(t)$, i.e., $\omega=f(I_1(t))$. (see Fig. 1). This simple mechanism of control allows us to change the speed of the robot's movement. A more developed example of this technique is shown later in our last example of a virtual robot.

3.2. Quadruped robot. Gait switching

In this example we study the possibility to switch between two different gaits using an external signal $I_2(t)$ connected to the phase instants τ's of the CPG (see Fig. 1). Consider the quadruped robot from Fig. 6. It has four degrees of freedom corresponding to its legs (Fig. 6(e)). Different gaits can be achieved depending on the patterns taken for a sequence. The system has been constructed storing $P=4$ patterns (Fig. 6(a)–(d)), such that the retrieval network has $N=5$ phase oscillators. The other parameters of the system are: $K=1$, $\alpha=2$ and $\omega=1$.

We want to switch between two gaits, each of them being a sequence of two patterns. The first sequence $S_1=(\xi^1,\xi^2)$ corresponds to the walking gait. The second one $S_2=(\xi^4,\xi^8)$ corresponds to the running gait. The external input $I_2(t)$ must switch the system from S_1 to S_2 at $t=10$. We chose the external input as $I_2(t)=0$ for $t<10$, and $I_2(t)=1$ for $t>10$.

The external signal controls the phase instants of the stored patterns as follows: $\tau_1=0-DI_2(t)$ and $\tau_2=\pi-DI_2(t)$, and, $\tau_3=0-D(1-I_2(t))$ and $\tau_4=\pi-D(1-I_2(t))$. We use the constant $D=100$ in this example. With this selection of interaction, only two patterns are plugged simultaneously in Eq. (7). For $t<10$, $\tau_1=0$ and $\tau_2=\pi$, and, $\tau_3=-100$ and $\tau_4=\pi-100$. For $t>10$, $\tau_1=-100$ and $\tau_2=\pi-100$, and, $\tau_3=0$ and $\tau_4=\pi$.

Fig. 7 shows an example of evolution for the gait switching. Supplementary material video2 shows a schematic animation of the simulation. Fig. 7(a) shows the temporal evolution of the energy function evaluated in the actual state $\xi(t)$ of the CPG. We observe that at $t=10$ when the external signal $I_2(t)$ changes the system evolves to an intermediate energy value. After that, the

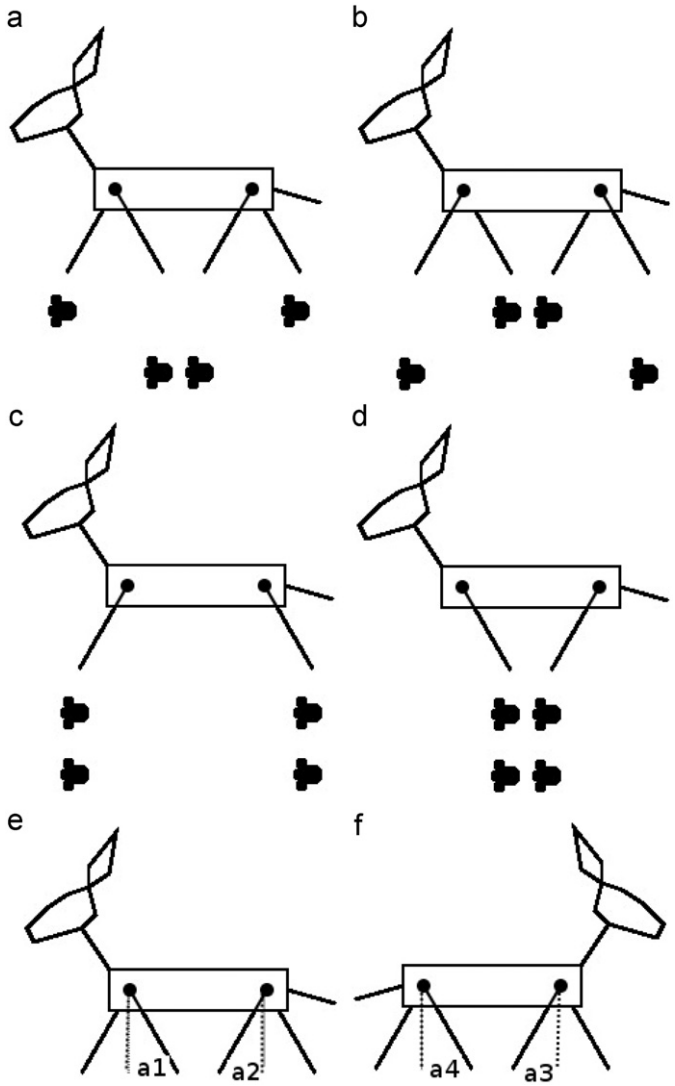

Fig. 6. Quadruped robot configurations. (a) Pattern ${}^{1}\xi^{1} \rightarrow \vec{u}^{1} = (30,-30,-30,30)$. (b) Pattern ${}^{2}\xi^{2} \rightarrow \vec{u}^{2} = (-30,30,30,-30)$. (c) Pattern ${}^{3}\xi^{4} \rightarrow \vec{u}^{3} = (30,-30,30,-30)$. (d) Pattern ${}^{4}\xi^{8} \rightarrow \vec{u}^{4} = (-30,30,-30,30)$. (e) Generic angle vector $\vec{u}^{k} = (a1,a2,a3,a4)$ in degrees. The black marks show the tracks from above of the quadruped robot.

system continues evolving between the same maximum and minimum values.

Fig. 7 shows the order parameter $R(t)$ as a function of time. In the transition between gaits $S_1 \rightarrow S_2$ at $t=10$, it holds that $R \neq 1$. That indicates that, during the transition, the CPG produces an interpolation between all stored patterns ξ's. This process is particularly useful for us since it allows a continuous transition between the angles until the new sequence (gaits) is well established.

Fig. 7(c) shows the closest state of the system ${}^{k}\xi$ (hypercube vertex) to the actual one $\xi(t)$ as a function of time. Here we can see clearly how the system evolves. For $t<10$ the system evolves between ξ^{1} and ξ^{2}, while for $t>10$ the system evolves between ξ^{4} and ξ^{8}.

Finally, Fig. 7(d) presents the angles of the robot's configuration. For $t<10$, legs one and four ($a1$ and $a4$), and two and three ($a2$ and $a3$) are in phase respectively. These two pairs are in opposite phases. This configuration corresponds to the waking gaits. After the application of the external signal, $t>10$, legs one and three ($a1$ and $a3$), and two and four ($a2$ and $a4$) are in phase respectively. Such a configuration corresponds to the running gaits. That shows that the gaits switching works properly.

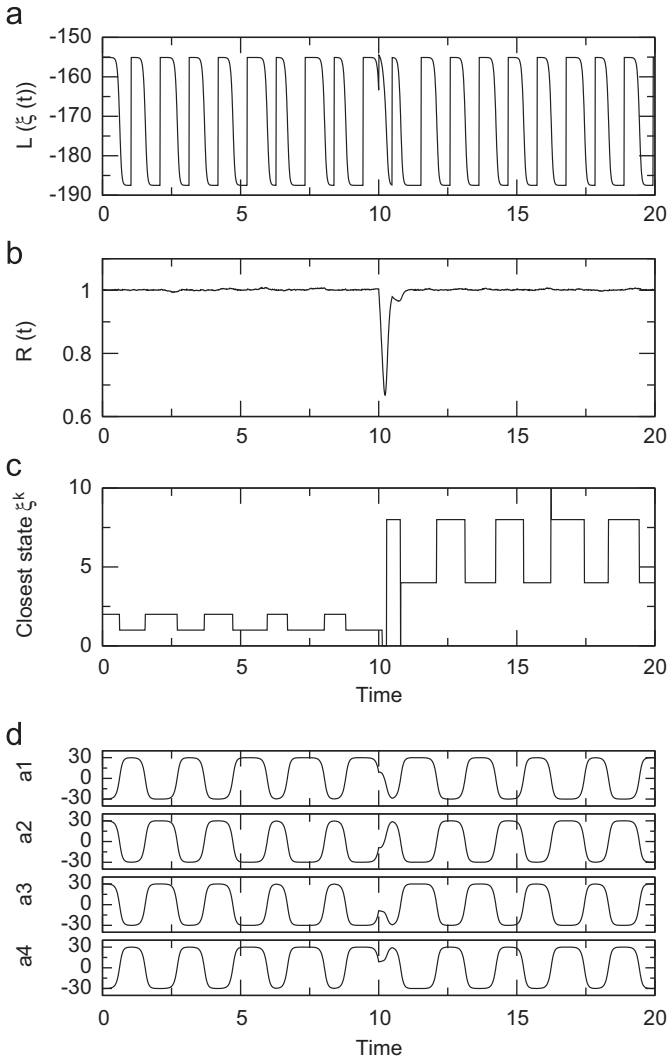

Fig. 7. Quadruped robot evolution. (a) Energy function $L(\xi(t))$ as a function of time evaluated in the actual state $\xi(t)$. (b) Order parameter $R(t)$ as a function of time. (c) Closest state ${}^{k}\xi$ to the actual state $\xi(t)$ as a function of time. (d) Angles as a function of time. The external signal is $I_2(t)=0$ for $t<10$ and $I_2(t)=1$ for $t>10$.

Note that the running gait in a real animal must be faster than the walking gait. We can implement this speed difference by using the input $I_1(t)$, increasing the frequency of the pacemaker as we have pointed out for the dancing robot.

3.3. Hexapod. Signaling to the interface

We consider in this example a hexapod virtual robot (ant-like). The hexapod has $A=6$ degree of freedom, one for each leg. The walking gaits of this robot consist of $P=2$ patterns shown in Fig. 8(a), (b). Fig. 8(c) indicates the angles we consider. The CPG in this case has $N=3$ oscillators in order to store these two patterns. The other parameters of the system are: $K=0.75$, $\alpha=8$ and $\omega=3$. The phase delays are for this example $\tau_1=0$ and $\tau_2=\pi$.

In case that the absolute values (amplitudes) of the angles of the legs have the same value, the robot has, obviously, no deviation from the robot's axis of symmetry while moving. Turning left (right) can be achieved maintaining the previous gaits, but making the amplitude values of the angles of the right (left) side a_R (a_L) larger than the values of the angles in left (right) side.

Now the aim is to reorient the robot displacement towards a target (in this example, a sugar cube, see Fig. 8(c)). Consider the external signal $I_3(t)$ that has as information the angle x between

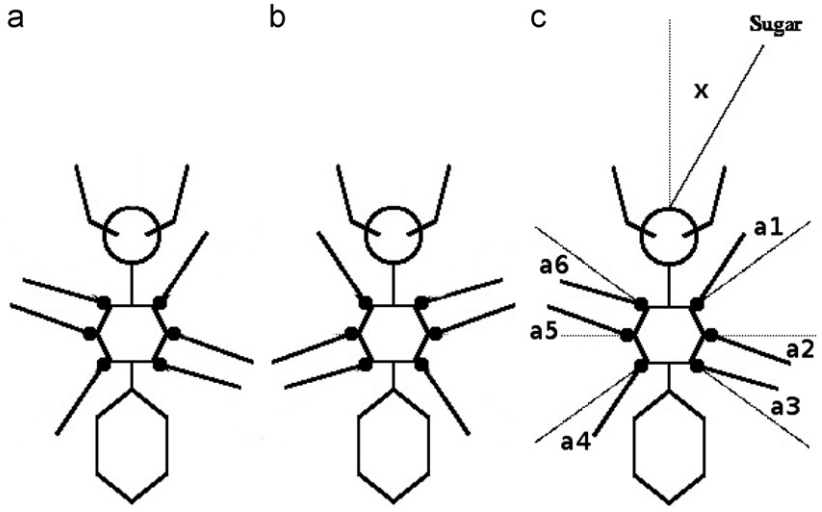

Fig. 8. Hexapod robot. (a) Pattern ${}^1\xi^1 \rightarrow \vec{u}^1 = (20,-20,20,-20,20,-20)$. (b) Pattern ${}^2\xi^2 \rightarrow \vec{u}^2 = (-20,20,-20,20,-20,20)$. (c) Angle vector $\vec{u}^k = (a1,a2,a3,a4,a5,a6)$. Angle x between the robot body symmetry axis and the target (sugar).

the symmetry axis of the robot body (direction of displacement) and the direction to the target, hence $I_3(t) = x$. We want to use this information in our model in order to change the direction of the advancing robot. Achieving this is possible by connecting the external signal $I_3(t)$ to the CPG–robot interface (see Fig. 1).

This problem can be easily solved in the interface CPG–robot. We plug in the amplitude of the angles: the right side $a_R = (a-0.2I_3(t))$, for the angles of the left side $a_L = (a+0.2I_3(t))$. Here $a=20$ is the reference angle. The position vectors are now: $\vec{u}^1 = (a_R,-a_R,a_R,-a_L,a_L,-a_L)$ and $\vec{u}^2 = (-a_R,a_R,-a_R,a_L,-a_L,a_L)$. Hence, the angular amplitude is now a function of the target position with respect to the robot.

Fig. 9 shows the angles as a function of time. The supplementary material video3 shows a capture of this simulation. The angle x varies linearly from left ($x<0$) to right ($x>0$) as a function of time Fig. 9(a). Fig. 9(b–g) shows the angles in degrees for the leg orientations.

Initially, for $t=0$, the angle $x=-50$, so the target is on the left side of the robot. In this situation, $a_R > a_L$, and the robot should be able to turn left. At $t=15$ the target is in front of the robot, and, $x=0$ and $a_R=a_L$. Thus, the robot has a straight trajectory curve towards the target. Finally, for $t>15$, the target is in the right side and $a_R < a_L$. With the new angles the robot can turn right. This simple interaction allows us to change the direction of displacement without affecting the operation of the CPG.

3.4. Penguin robot. Signaling the pacemaker

In this last example we present another possibility in order to change the robot's displacement direction. Consider the penguin robot from Fig. 10. Each wing is controlled by its own CPG, we call them CPG_L and CPG_R, for the left and right side respectively. The left CPG_L has a natural frequency ω_L, and the right CPG_R has a natural frequency ω_R. Each of these CPGs has $N=3$ oscillators and $P=2$ stored patterns each one (both have the same patterns). Each wing has only one degree of freedom $A=1$. The angle a_R for the right wing, and the angle a_L for the left wing are shown in Fig. 10(a) and (b). The other parameters of the system are $K=0.75$, $\alpha=9$ and $\omega=3$. The phase instants are set as $\tau_1=0$ and $\tau_2=\pi$.

In order to swim straightforward, the penguin must move both wings simultaneously and in phase with the same speed. If it wants to turn left (right), the right (left) wing must move faster than the left (right) wing. The different speeds of the wings can be induced by varying the natural frequencies of the pacemakers of their CPGs as implemented in our example.

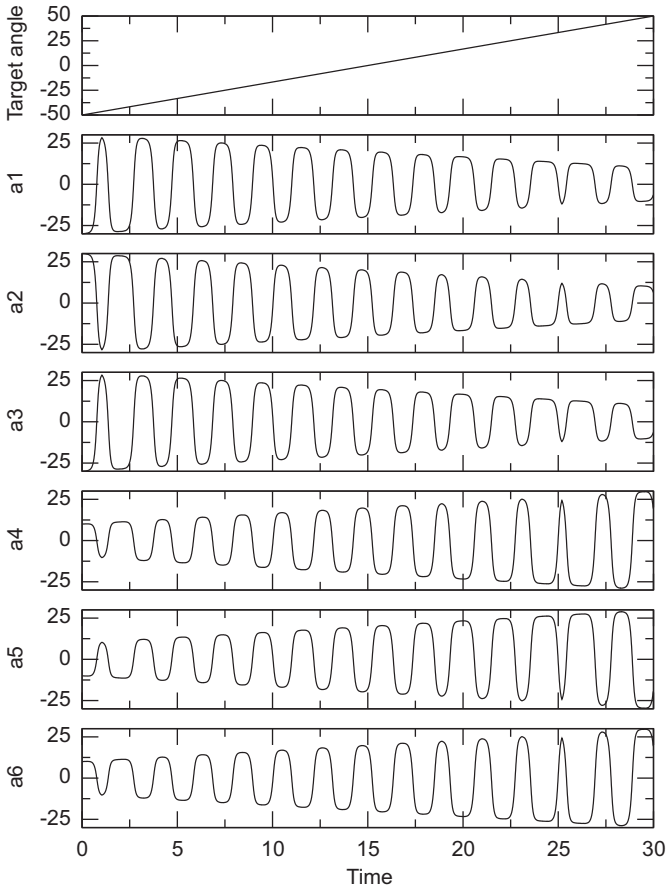

Fig. 9. Angles as function of time. (a) Angle x between the robot symmetry axis and the direction of the target. (b–g) Angles of the robot legs (see Fig. 8).

The particularity now in this example is that the pacemakers of these two CPGs are coupled by a Kuramoto interaction κ. Thus, in case that both pacemakers have the same natural frequency $\omega_R = \omega_L$, they synchronize in phase, and both wings have the same speed and phase. If the natural frequencies are different $\omega_R \neq \omega_L$, the pacemakers are no longer synchronized and they can have different speeds as we need in order to turn. The critical κ value needed to have phase synchronization is function of the

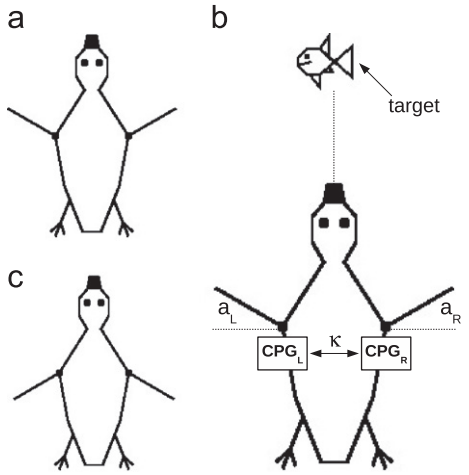

Fig. 10. Penguin robot. (a) Pattern ${}^1\xi^1 \rightarrow \vec{u}^1 = (30)$. (b) Pattern ${}^2\xi^2 \rightarrow \vec{u}^2 = (-30)$. (c) Angle vector Pattern $\vec{u}^k = (a)$ with $a = a_L$ and $a = a_R$ for CPG_L and CPG_R respectively. There are two central pattern generators, CPG_L and CPG_R, they are coupled by a Kuramoto interaction of strength $\kappa = 0.5$.

frequency difference between ω_L and ω_R as it is know from Kuramoto theory of phase oscillators [10].

The penguin must follow the target (fish) (see Fig. 10(c)). The angle x between its direction of displacement and the target is taken as input $I_1(t)$. If $x > 0$ (target on the right side), $I_1(t) = 0.75$, if $x < 0$ (target on the left side), $I_1(t) = -0.75$, and for $x = 0$ (target in front), $I_1(t) = 0$. The external signal $I_1(t)$ changes the natural frequency of the pacemakers according to

$$\omega_L = \omega + I_1(t), \tag{11}$$

$$\omega_R = \omega - I_1(t). \tag{12}$$

Thus, the frequency of the CPG increases or decreases depending on the direction in which the penguin must swim.

The dynamics of the pacemakers are given by:

$$\dot{\psi}_L = \omega_L + \kappa \sin(\psi_R - \psi_L), \tag{13}$$

$$\dot{\psi}_R = \omega_R + \kappa \sin(\psi_L - \psi_R). \tag{14}$$

Fig. 11 shows an example of evolution for this system. Supplementary material video4 shows an animation of the robot during this simulation. In this example, the position of the target changes as a function of time, for $0 < t < 20$ the target is in front of the penguin ($x = 0$). For $20 < t < 50$ the target is on the right side ($x > 0$), and for $50 < t < 80$, the target is on the left side ($x < 0$). Finally, for $t > 80$ the target is again in front of the penguin (see video4).

Fig. 11(a) and (b) shows the evolution of the energy functions of the CPG_L and CPG_R respectively. We observe that their evolutions are in phase during the periods when $\omega_L = \omega_R$ ($t < 20$ and $t > 80$). During these time intervals, the wings exhibit also in phase oscillations (see Fig. 11(c) and (d)). There are some temporal differences due to the stochastic nature of the system. So, along these time intervals, the robot moves straightforward.

During the time interval when $\omega_R \neq \omega_L$ ($20 < t < 80$), both CPGs are essentially disconnected and they can work with different frequencies as we see in Fig. 11a,b,c,d. In this situation the control input signal $I_1(t)$ allows hypothetically changing the direction of displacement of the robot that is following the target.

Note that the Kuramoto interaction between the pacemakers allows the system to evolve gradually between the different states of synchronization of the pacemakers (from phase synchronization to desynchronization). As a consequence, there are not abrupt changes in the dynamics of the system (movements) even when the external control signal $I_1(t)$ has discontinuities or fast changes.

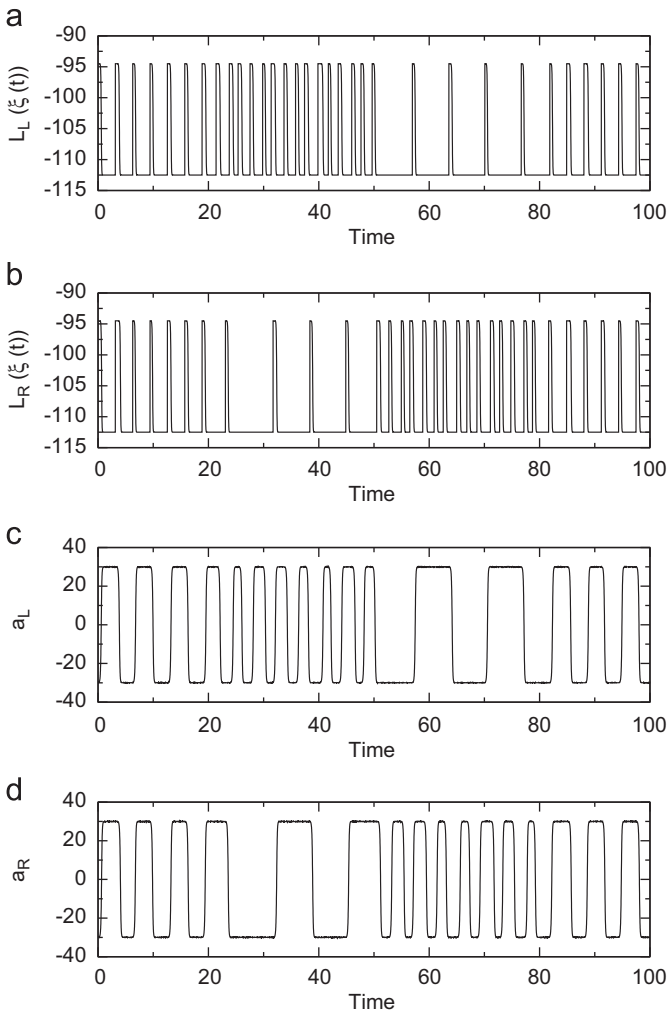

Fig. 11. Penguin evolution. (a) Energy function $L_L(\xi(t))$ of the CPG_L evaluated in the actual state ${}^L\xi(t)$ as a function of time. (b) Energy function $L_R(\xi(t))$ of the CPG_R evaluated in the actual state ${}^R\xi(t)$ as a function of time. (c) Angle of the left wing a_L as a function of time. (d) Angle of the right wing a_R as a function of time.

4. Discussions and conclusions

In this work we have presented a model of central pattern generator, a model of interface between the CPG and the virtual robots, and the integration of different control signals coming from the robot environment.

The central pattern generator constructed with phase oscillators can be seen as an associative memory driven by a pacemaker. This architecture in blocks agrees with many proposed models of central pattern generators [1]. In particular our model is very robust because its retrieval mechanism; the system has a gradient dynamics in a potential with only one minimum.

Several previous models of CPG for gaits generation use for each degree of freedom of the robot or animal an oscillator [2,3,6], so thus, the number of oscillators is a function of the shape of the robot or animal. In contrast, in our model of CPG, the size of the system (number of oscillators of the retrieval network) is a function only of the number of patterns P that are stored for the sequences.

Differently from the model of Steingrube et al. [8], where the different sequences of the CPG are generated by stabilizing particular periodic orbits as a function of the external signals, in our model, several sequences can be generated by varying the phase instants τ's of the stored patterns. Obviously, all the

involved patterns must be stored in the network in the couplings between the pacemaker and the retrieval network.

We want to emphasize the necessity of a stochastic dynamics in the retrieval network of the CPG for the robot to operate naturally and properly. Without noise, the system cannot escape from the unstable fixed points and the CPG cannot generate the sequences. The utility of noise in neural system is well known, being the stochastic dynamics proposed as a principle of brain functionality [26].

The function of the CPG is to generate binary patterns and interpolate between them. That function is independent from the concrete application of the patterns and depends only on the interface which makes use of the CPG. This suggests that our CPG can be used as an independent unit of a bigger system. Different movements can be driven by the same CPG if the proper interfaces are connected to it.

The interface between the robot and the CPG makes use of the rich internal dynamics of the retrieval network. We have found that using orthogonal patterns, the CPG generates linear interpolation between the consecutive states of the sequences. Additionally, dynamical characteristics of acceleration and deceleration close to the fixed points produce realistic movements for the robots.

We have shown how external controlling signals can be implemented in order to change the robot dynamics. These signals have been connected in different parts of the system. We have presented in four examples: several possibilities of controlling, changing the frequency of the CPG, generation and switching of gaits, and controlling the robot orientation while tracking targets.

Finally, we want to note the possibility of hardware implementation of this model in real physical devices. In effect, devices of electronic oscillators with synchronization properties have been recently proposed for fast sensing and labeling of image objects [27]. The modularity characteristics of the presented model can be very helpful since the same CPG device can be used in different implementations, independently from the physical characteristic of the robots as it has been previously shown.

Acknowledgments

We thank Prof. Alexander S. Mikhailov very much for valuable discussions and support for this project.

Appendix A. Supplementary material

Supplementary data associated with this article can be found in the online version of http://dx.doi.org.10.1016/j.neucom.2012.05.019.

References

[1] S.L. Hooper, Central pattern generators, Curr. Biol. 10 (5) (2000) R176–R177.
[2] J.J. Collins, I.N. Stewart, Coupled nonlinear oscillators and the symmetries of animal gaits, J. Nonlinear Sci. 3 (1993) 34992.
[3] K.V. Baev, Y.P. Shimansky, Principles of organization of neural systems controlling automatic movements in animals, Prog. Neurobiol. 39 (1992) 45–112.
[4] F. Brocard, S. Tazerart, L. Vinay, Do pacemakers drive the central pattern generator for locomotion in mammals? Neuroscientist 16 (2010) 139–155.
[5] J. Hertz, A. Krogh, R. Palmer, Introduction to the Theory of Neural Computation, Addison-Wesley Publishing Company, 1991.
[6] A.J. Ijspeert, Central pattern generators for locomotion control in animals and robots: a review, Neural Netw. 21 (2008) 642–653.
[7] S. Conforto, I. Bernabucci, G. Severini, M. Schmid, T. D'Alessio, Biologically inspired modelling for the control of upper limb movements: from concept studies to future applications, Front. Neurorobotics 3 (3) (2009).
[8] S. Steingrube, M. Timme, F. Wörgötter, P. Manoonpong, Self-organized adaptation of a simple neural circuit enables complex robot behaviour, Nat. Phys. 6 (2010) 224–230.
[9] J.J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proc. Natl. Acad. Sci. USA 79 (8) (1982) 2554–2558.
[10] Y. Kuramoto, Chemical Oscillations, Waves, Turbulence, Springer, New York, 1984.
[11] J.A. Acebrón, L.L. Bonilla, V.C.J. Pérez, F. Ritort, R. Spigler, The Kuramoto model: a simple paradigm for synchronization phenomena, Rev. Mod. Phys. 77 (2005) 137–185.
[12] T. Aoyagi, Network of neural oscillators for retrieving phase information, Phys. Rev. Lett. 74 (20) (1995) 4075–4078.
[13] T. Aonishi, K. Kurata, M. Okada, Statistical mechanics of an oscillator associative memory with scattered natural frequencies, Phys. Rev. Lett. 82 (13) (1999) 2800–2803.
[14] T. Nishikawa, Y.-C. Lai, F.C. Hoppensteadt, Capacity of oscillatory associative-memory networks with error-free retrieval, Phys. Rev. Lett. 92 (2004) 108101.
[15] T. Nishikawa, F.C. Hoppensteadt, Y.-C. Lai, Oscillatory associative memory network with perfect retrieval, Physica D 197 (2004) 134–148.
[16] T. Aonishi, Phase transitions of an oscillator neural network with a standard Hebb learning rule, Phys. Rev. E 58 (4) (1998) 4865–4871.
[17] M. Yoshioka, M. Shiino, Associative memory storing an extensive number of patterns based on a network of oscillators with distributed natural frequencies in the presence of external white noise, Phys. Rev. E 61 (5) (2000) 4732–4744.
[18] M. Yamana, M. Shiino, M. Yoshioka, Oscillator neural network model with distributed native frequencies, J. Phys. A 32 (1999) 3525–3533.
[19] P. Kaluza, H. Meyer-Ortmanns, Controlled pattern retrieval in a designed energy landscape, arxiv:1107.2294v1 (2011).
[20] P. Peretto, J.J. Niez, Long term memory storage capacity of multiconnected neural networks, Biol. Cybern. 54 (1986) 53–63.
[21] J.J. Arenzon, R.M.C. de Almeida, Neural networks with high-order connections, Phys. Rev. E 48 (1993) 4060–4069.
[22] S.C. Manrubia, A.S. Mikhailov, D.H. Zanette, Emergence of Dynamical Order: Synchronization Phenomena in Complex Systems, World Scientific Lecture Notes in Complex Systems, vol. 2, World Scientific, Singapore, 2004.
[23] M. Seki. Development of robotic upper limb orthosis with tremor suppressibility and elbow joint movability, in: 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2011, pp. 729–735.
[24] G. Severini, S. Conforto, I. Bernabucci, M. Schmid, T. D'Alessio. Tremor control during movement of the upper limb using artificial neural networks, in: 4th European Conference of the International Federation for Medical and Biological Engineering. IFMBE Proceedings, vol. 22, 2009, pp. 72–75.
[25] R.L. Honeycutt, Stochastic Runge–Kutta algorithms. I. White noise, Phys. Rev. A 45 (2) (1992) 600–603.
[26] G. Deco, E.T. Rolls, R. Romo, Stochastic dynamics as a principle of brain function, Prog. Neurobiol. 88 (1) (2009) 1–16.
[27] J. Kowalski, M. Strzelecki, H. Kim, Implementation of a synchronized oscillator circuit for fast sensing and labeling of image objects, Sensors 11 (4) (2011) 3401–3417.

Pablo Kaluza is a professor of physics at the Basic Science Institute of the National University of Cuyo in Mendoza, Argentina. He holds a bachelor and a master degree in physics from the Balseiro Institute (National University of Cuyo in Bariloche, Argentina). He has obtained a Ph.D. in Physics at the Technical University Berlin in Germany (he has been student at the Fritz Haber Institute of the Max Plank Society in Berlin, Germany). Also, this author has worked as postdoctoral researcher in several German institutions. His actual research focus on dynamical systems, complex networks and artificial neural networks.

Teodor Cioaca is currently a Computer Graphics and Simulation software developer within Essensys Software S.R.L. Romania. He holds a Mathematics and Computer Science License Diploma from University of Bucharest (2008), and a Master of Science degree from Jacobs University Bremen in Smart Systems (Computer Graphics and Robotics) (2010). Also, this author has worked as a Research Assistant for the German Center of Artificial Intelligence (DFKI Bremen) and as a Research Associate for Jacobs University Bremen between 2009 and 2011, focusing on Computer Graphics, Modeling, Physics based simulation and robot kinematics.