# Importing the Computational Neuroscience Toolbox into Neuro-Evolution—Application to Basal Ganglia

Jean-Baptiste Mouret
mouret@isir.upmc.fr

Stéphane Doncieux
doncieux@isir.upmc.fr

Benoît Girard
girard@isir.upmc.fr

ISIR, Université Pierre et Marie Curie-Paris 6, CNRS UMR 7222
4 place Jussieu, F-75252, Paris Cedex 05, France

## ABSTRACT

Neuro-evolution and computational neuroscience are two scientific domains that produce surprisingly different artificial neural networks. Inspired by the "toolbox" used by neuroscientists to create their models, this paper argues two main points: (1) neural maps (spatially-organized identical neurons) should be the building blocks to evolve neural networks able to perform cognitive functions and (2) well-identified modules of the brain for which there exists computational neuroscience models provide well-defined benchmarks for neuro-evolution.

To support these claims, a method to evolve networks of neural maps is introduced then applied to evolve neural networks with a similar functionality to basal ganglia in animals (i.e. action selection). Results show that: (1) the map-based encoding easily achieves this task while a direct encoding never solves it; (2) this encoding is independent of the size of maps and can therefore be used to evolve large and brain-like neural networks; (3) the failure of direct encoding to solve the task validates the relevance of action selection as a benchmark for neuro-evolution.

## Categories and Subject Descriptors

I.2.6 [**Artificial intelligence**]: Learning—*Connectionism and neural nets*

## General Terms

Algorithms

## Keywords

evolutionary algorithms; neural networks; computational neuroscience; basal ganglia; neuro-evolution.

## 1. INTRODUCTION

Evolution has been one of the prominent processes to shape the animal brain; imitating this process by employing evolution-inspired algorithms to design "artificial nervous systems" therefore drew considerable attention in artificial intelligence over the last two decades. This line of thought drives researchers to propose many methods to exploit evolutionary algorithms (EA) to design neural networks (neuro-evolution), from efficient direct encodings [44] to grammar-based generative systems [16, 20, 30, 31] and development-inspired approaches [4, 43]. Resulting neural networks can then control robots or artificial agents [28, 33, 31].

Concurrently with these researches, the field of computational neuroscience (see e.g. [6]) designs models of parts of the nervous system based on experimental data from biology. Among the published models, some of them are efficient enough to be useful in robots [23, 22, 47, 29, 25, 35, 15] and could provide the basic blocks to build a complete bio-inspired "artificial brain" for artificial intelligence.

Computational neuroscience and neuro-evolution therefore share a characteristic—creating artificial nervous systems; however, the comparison of the resulting neural networks is surprising: models produced in computational neuroscience display almost no similarities with evolved neural networks. The latter usually involve up to a few dozens of neurons and do not scale up to many more neurons, whereas computational neuroscience often manipulate hundreds of neurons in a very organized fashion. A detailed investigation of published models (especially [17, 18, 15, 45, 46, 40, 39]) reveals that the chosen building blocks constitute at least one fundamental difference; evolutionary methods mostly use individual neurons, ideally organized in modular fashion [20, 38, 8, 30, 31], but many neuroscience models (e.g. figure 1) rely on the concepts of *maps* (a $N \times M$ grid of neurons, in which $N$ and $M$ are free variables of the model) connected by regular connection schemes (either one to one connections or one to all with a regular assignation of weights). This allows such neural networks to scale up to larger maps (e.g. to handle higher-dimensional inputs) while maintaining the same overall structure. This description of neural networks as connected maps can be seen as the result of a developmental process in which a network of maps is developed to form a neural network.

Besides this analysis, it appears that computational neuroscience can also provide an efficient approach to benchmark neuro-evolution methods in the context of cognitive functions. On one hand, neuro-evolution ended up with substantially good results to control non-linear systems [16, 44, 8], but despite Beer's preliminary proposals [2, 41], no clear benchmark has been widely accepted to evaluate their potential for generating more cognitive functions. On the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*GECCO'10,* July 7–11, 2010, Portland, Oregon, USA.
Copyright 2010 ACM 978-1-4503-0072-8/10/07 ...$10.00.

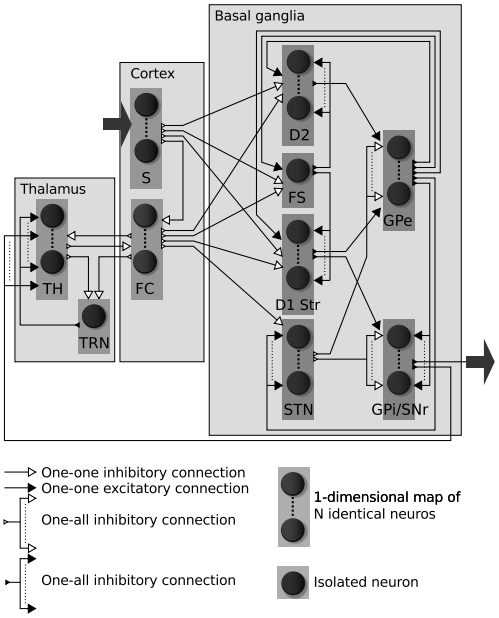



**Figure 1: Contracting basal ganglia model for action selection. D1/D2 Str: striatal neurons with D1 or D2 dopamine receptors; FC: frontal cortex; FS: fast spiking interneurons; GPe/GPi: external and internal part of the globus pallidus; S: input saliences; SNr: substancia nigra reticulata; STN: subthalamic nucleus; TH: thalamus; TRN: reticular thalamic nucleus.**

other hand, experimental neuroscience isolated several modules of the nervous systems, and precisely described their inputs and outputs such that computational neuroscience can model them. On the road to automatically design artificial nervous systems, the minimum benchmark for any neuro-evolution method should be to reproduce the functions modeled by neuro-scientists and to equal the efficiency of hand-designed neural networks.

Building on this inspiring parallel, this paper argues two main points:

- Maps (and not individual neurons) should be the building blocks to evolve neural networks able to perform cognitive functions.
- Well-identified modules of the brain for which there exists computational neuroscience models provide well-defined benchmarks for neuro-evolution.

To support these claims, a method to evolve networks of maps is described then applied to evolve neural networks with a similar functionality to basal ganglia (winner-takes-all selection by disinhibition). Results are compared with an optimized basal ganglia model from neuroscience and networks evolved with a classic direct encoding.

# 2. RELATED WORK

## 2.1 Evolving Neural Networks

Most of the work on the structural evolution of neural networks focuses on the definition of efficient encodings with their associated genetic operators. In direct encodings, genetic operators directly manipulate connections and neurons. The most successful direct encoding is undoubtedly NEAT [44], which is based on "innovation numbers" to improve cross-over and structural diversity preservation. Despite their simplicity and their efficiency to solve some control problems, direct encodings fail to scale up to large networks, supposedly because they cannot capture the regularity of the search space by using several times the same sub-networks.

Noticing that biological systems widely rely on the repetition and combination of hierarchically organized modules [20, 19, 30, 31], several researchers proposed to encode neural networks indirectly by employing a compact representation that is then *developed* into a neural network. One of the straightforward way to implement this idea is to use a list of modules as a genotype, as done in modular encodings [8, 38, 31]. A more complex but potentially more powerful approach is to evolve "construction programs" whose instructions are interpreted to build neural networks. This *modus operandi* was employed in cellular encoding [16] to design highly repetitive neural networks and in several other grammar-based generative systems [20, 30, 31].

A last method is to evolve chemical gradients in Cartesian space that can then be used to compute synaptic weights. Provided that these gradients displays some regularities (e.g. repetitive patterns or symmetry), the synaptic weights can mirror them. This process is currently instantiated in HyperNEAT [43], one of the rare neuro-evolution framework able to evolve networks of several hundred of neurons. However, the published method is not designed to evolve the topology of neural networks[1] and requires the experimenter to fix the number of neurons. Moreover, despite its inspiration from biological development, it remains to be proved that HyperNEAT can generate neural networks similar to those observed in real animals.

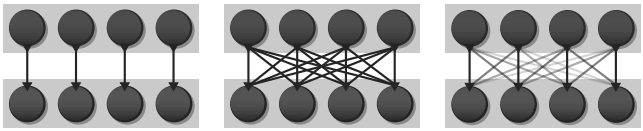
## 2.2 EAs and Computational Neuroscience

Current neuro-evolution methods mainly draw from computer science (L-systems [20, 19], formal grammars [16, 30], graph theory, machine learning, control theory[44]) and more rarely biological development [4, 43]. To our knowledge, only a few attempts to explicitly link this field to computational neuroscience have been published.

The main import from neuroscience to neuro-evolution is undoubtedly the neuron formulation. Beer et al. [3] analyzed the evolution of networks of leaky integrators, a neuron model initially introduced in computational neuroscience, for "minimal cognitive behaviors". Models at other abstraction levels, such as spiking neurons [10], nonlinear oscillators [27] and neural fields [21], have also been considered as the building blocks of evolved neural networks. Another important inspiration from neuroscience to neuro-evolution is on-line learning schemes such as Hebbian-like rules [12] and neuro-modulation [24, 42]. It should be emphasized that these papers mostly copied a brick from neuroscience and then evolved networks without any biological inspiration about the topology. As a consequence, the resulting neural networks are topologically different from those hand-designed by neuro-scientists.

Following a reasoning closer to the one argued here, Ijspeert

[1]HyperNEAT only evolves synaptic weights but connections with near zero weights can be removed.



(a) (b) (c)

**Figure 2: Three ways to connect two maps of neurons. (a) one to one; (b) one to all, with the same synaptic weight; (c) one to all with a Gaussian distribution of synaptic weights.**

et al. [23] imported from Ekeberg et al. [9] the overall organization and the neuron parameters—both extracted from animal experiments—of the neural network that generates the locomotion patterns of lampreys (the Central Pattern Generator, CPG). In a multi-stage evolutionary process, the authors first optimized the connectivity and the synaptic weights of segmental oscillators and then the connectivity between these oscillators. The obtained network oscillates over a wider range of frequency and amplitude than the hand-designed neural network [23] and, although they have only been investigated in simulation, they could be embedded in snake-like robots [22]. By constraining the evolutionary process with biological data, Ijspeert et al. managed to automatically design a large but regular neural network more efficient than a hand-designed controller. This result concurs with the approach presented here: using data from neuroscience (connectivity patterns, overall organization, common parameters) could lead neuro-evolution to a new level of functionality.

# 3. MAP-BASED ENCODING

## 3.1 Computational Neuroscience Toolbox

A careful examination of the published neuroscience models, and especially of those that could be employed in artificial intelligence, allows to extract some regularities in model descriptions. Neuro-scientists converged to a limited set of structures that can describe a wide range of biological neural networks while being simple enough to be analyzed and well understood. We list here the main features of this "computational neuroscience toolbox".

Most neuroscience models use either spiking neurons [14] or leaky integrators, which only simulate the overall dynamic of a population of neurons [3]. This latter simple dynamic neuron has also been widely used in evolutionary robotics (see [11]) as it shows many different temporal behaviors but remains computationally inexpensive to simulate. Furthermore, in contrast to neurons employed in neuro-evolution or in machine learning, neurons manipulated in computational neuroscience are either inhibitory or excitatory.

In this work, we opt for the lPDS-based (locally Projected Dynamic System) neuron model [15], a variant of the classic leaky integrator with similar dynamics but which verify the dynamic property of contraction [26]. Using the Euler integration method, the output $y_{t+dt}$ of a lPDS neuron $i$ at time $t+dt$ is computed as follows:

$$p_t^{(i)} = \sum_{j \in C} w_{i,j} y_t^{(j)} \quad (1)$$

$$a_{t+dt}^{(i)} = \max(0, \min(1, a_t^{(i)} + \tfrac{1}{\tau}(p_t^{(i)} - a_t^{(i)} + T_i) \cdot dt)) \quad (2)$$

$$y_{t+dt}^{(i)} = \begin{cases} a_{t+dt}^{(i)} & \text{if i is excitatory} \\ -a_{t+dt}^{(i)} & \text{otherwise} \end{cases} \quad (3)$$

where $\tau$ is the time-constant of neuron $i$, $T_i$ a threshold value, $w_{i,j}$ the synaptic weight between neuron $i$ and $j$, $a_t^{(j)}$ the activation level of neuron $j$ and $dt$ the integration step (e.g. 0.001 s).

The main building blocks of the considered computational neuroscience models are either $N \times M$ maps of neurons or individual neurons. Maps are defined as spatially organized grids of *identical* neurons (same time constant, same threshold, same inhibitory status). Many models employ only maps with the same dimension, arbitrary fixed to the dimension of the input. At any rate, the number of map dimensions in a given model is very restricted as it typically covers only the dimension of the inputs and those of the outputs.

Connection schemes between maps are restricted to three cases (figure 2) in most models: (1) one to one connection with constant weights (neuron $i$ of map $M_1$ is connected to neuron $j$ of map $M_2$, with a positive weight identical for each connection), (2) one to all connections with constant weights (neuron $i$ of map $M_1$ is connected to each neuron of map $M_2$, with identical weights for all connections) and (3) one to all connections with weights following a Gaussian distribution computed as follows:

$$d_{i,j} = \frac{||i-j||^2}{N}, i \in M_1, j \in M_2 \quad (4)$$

$$w_{i,j} = \Gamma \exp(\frac{-d_{i,j}}{\sigma^2}), i \in M_1, j \in M_2 \quad (5)$$

where $N$ is the map size[2], $i$ is the $i$-th neuron of map $M_1$, $j$ the $j$-th of $M_2$ and $w_{i,j}$ the synaptic weight between $i$ and $j$. $\sigma$ (standard deviation) and $\Gamma$ (weight amplitude) are the only parameters that can be changed.

This regular assignation of weights (either constant or following a simple distribution) contrasts with typical neuro-evolution methods in which each synaptic weight is set-up separately. In effect, this adds numerous regularities in neural networks that are similar to those sought by modular and developmental encodings.

This computational neuroscience toolbox is sufficient to describe a wide range of models, for instance models of basal ganglia [17, 18, 15], of colliculus [45] and of visual attention [46, 40, 39]. Additions such as Hebbian learning and neural fields would allow to describe many other models.

## 3.2 Evolving a Graph of Neural Maps

Most computational neuroscience models can be described as a graph of neural maps in which each map and each connection is described by a set of parameters (time constant of neurons, connection scheme, synaptic weights, etc.). Such a labeled graph can be modified structurally (add/remove a connection or a node) and parametrically (change of a label). We first describe a basic method to evolve a generic

[2]It is here assumed that all the maps have the same size. This formula is easily generalized to the case of two maps with different sizes.

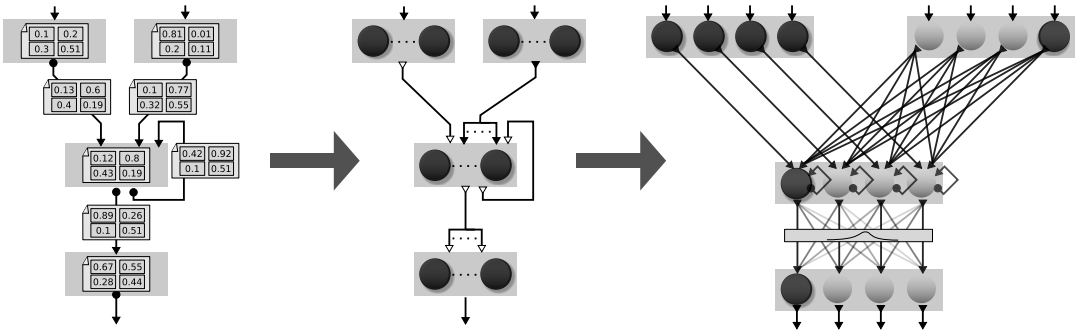


**Figure 3: Overview of the development process. From left to right: (1) the genotype is a labeled graph with evolvable labels; (2) the labels are interpreted to a neuroscience-inspired description of the neural network; (3) for a given size of maps, this neural network can be fully developed into a neural network (for instance to evaluate its fitness).**

labeled graph then we introduce the set of labels employed to describe a graph of neural maps.

#### 3.2.1 Evolving a Labeled Graph

The graph is represented as a classic adjacency list for which mutation operators have been designed; cross-over is not used.

In this work, we employ straightforward mutation operators that apply each of the possible changes with a user-defined probability (see appendix to know the chosen values). Three structural mutation operators, inspired by those defined in NEAT [44], have been designed:

- addition of a node on an existent connection, with random labels; the connection is split in two and the two parts keep the same labels;
- removal of a random node and its associated connections;
- addition/removal of a connection between two random neurons.

Nodes and connections can be labeled by a list of real parameters that represent weights, threshold, neuron type, ... These parameters are mutated using polynomial mutation [7].

To initialize a random graph, a random number of nodes (with random labels) are first created then connected by a random number of connections (also with random labels). Connections cannot be doubled. Last, graphs are simplified by removing each sub-graph not connected to both an input and an output.

#### 3.2.2 Map-based Encoding

To evolve a network of maps inspired by the neuroscience toolbox, the previously defined operators are employed with a particular labeling of nodes and connections (figure 3). This representation is independent of the size of maps, that is the same genotype can be developed into a neural network with an arbitrary number of neurons.

In the current implementation, nodes are labeled with four evolved numbers $\{e, v, T, a\} \in [0, 1)^4$ that are scaled or binarized to obtain nodes' parameters. They are interpreted as follows[3]:

[3] $\lfloor x \rfloor$ denotes the floor of $x$.

$$\text{Type:} \quad \begin{cases} \text{inhibitory if } e < 0.5 \\ \text{excitatory otherwise.} \end{cases} \tag{6}$$

$$\text{Time constant } (\tau): \quad \tau = v_{\lfloor \tau' \times 4 \rfloor} \tag{7}$$

$$\text{where} \quad v = [5 \cdot 10^{-3}, 10^{-3}, 20 \cdot 10^{-3}, 40 \cdot 10^{-3}]$$

$$\text{Threshold:} \quad 10 \cdot (T - 0.5) \tag{8}$$

$$\text{Map type:} \quad \begin{cases} \text{isolated neuron if } a > 0.75 \\ \text{standard map otherwise.} \end{cases} \tag{9}$$

Similarly, connections are labeled with four evolved real numbers $\{c, k, \Gamma, \sigma\} \in [0, 1)^4$ that are interpreted as follows:

$$\text{Scheme:} \quad \begin{cases} \text{1 to 1, if } c < 0.5 \\ \text{1 to all, otherwise} \end{cases} \tag{10}$$

$$\text{Function:} \quad \begin{cases} \text{Gaussian if } c > 0.5 \text{ and } k < 0.5 \\ \text{constant otherwise} \end{cases} \tag{11}$$

$$\text{Parameter 1:} \quad \begin{cases} \Gamma \text{ if } c > 0.5 \text{ and } k < 0.5 \\ \frac{1}{5} w \text{ (synaptic weight) otherwise.} \end{cases} \tag{12}$$

$$\text{Parameter 2:} \quad \begin{cases} \sigma \text{ if } c > 0.5 \text{ and } k < 0.5 \\ \text{otherwise: ignored} \end{cases} \tag{13}$$

### 3.3 Classic Direct Encoding

The previously described evolvable labeled graph can also be employed in a more classic fashion to directly define a neural network. This leads to a direct encoding of neural networks. In this case, each node describes a neuron (instead of a map). Labels used to describe neurons are the same as those used to describe maps (time constant, etc.). Connections are labeled with a single real number interpreted as the synaptic weight. This direct encoding will be used as a reference encoding in the next section.

## 4. EXPERIMENTS

### 4.1 Action Selection in Basal Ganglia

Having introduced the map-based encoding, we now focus on evaluating its ability to generate brain-like structures; to that aim, we chose to reproduce the function performed by the basal ganglia (BG).

More precisely, the basal ganglia are a set of interconnected subcortical nuclei [36], that are thought to be involved in action selection [37, 34], i.e. the problem, for an agent, of choosing which action to perform within a repertoire, given internal and external sensory information, in order to achieve its goals. Solving this generic resource allocation problem seems to be a central cognitive function as the BG circuitry is common to all vertebrates, and as it contains duplicated circuits [1] dedicated to many critical functions (skeleton movements, eye movements, sequence learning, working memory, navigation, planning, etc.). The BG performs the two main aspects of action selection: the central process of selection of one action only among conflicting ones, similar to a winner-takes-all (WTA), and the learning process necessary to bias the selection process towards the most profitable option. We focus here on the WTA-like process.

The BG circuits are subdivided in parallel channels, which are supposed to be associated to the actions of the repertoire, and which receive convergent inputs from the cortex. At rest, the BG output is tonically active and inhibits its targets. The selection of an action, caused by a strong convergent input on the associated channel signaling its urgency, causes a pause of the channel output inhibition, allowing the activation of the targeted brain region [5]. Various models of this WTA process have been proposed; we consider here the CBG model proposed in [15], which derives from the new functional interpretation of the circuitry proposed by [17], but adds some previously unused connections (figure 1).

In the following, we test the ability of our new map-based encoding to evolve circuits exhibiting the WTA ability expected from BG models, and compare the obtained results with CBG models whose parameters[4] are optimized by a basic evolutionary algorithm.

## 4.2 Fitness

In the remaining text, the following notations are used:

- $x$: a developed individual (a neural network);
- $k$: number of inputs (also the number of outputs);
- $T$: the maximum simulation duration;
- $\gamma(x, \mathbf{v}, t)_i$: activation level of the output neuron $i$ ($i \in \{1, ..., k\}$) at time $t$ ($t \in [0, T]$), given the input vector $\mathbf{v}$ of saliences ($\mathbf{v} \in \mathbb{R}^k$);
- $T_c(x, \mathbf{v})$: the duration of the simulation for the input vector $\mathbf{v}$ (see $K(x)$ below);
- $\gamma_c(x, \mathbf{v})_i$: activation level of the output neuron $i$ ($i \in \{1, ..., k\}$) at the end of the simulation (i.e. $t = T_c(x, \mathbf{v})$);
- $\gamma_0(x)$ the average activation level of the output neurons when the inputs are null, at the end of the experiment:

$$\gamma_0(x) = \frac{1}{k} \sum_{i=1}^{k} \gamma_c(x, [0, \cdots, 0], 1)_i \quad (14)$$

Each individual is simulated until its output converges to a constant vector or until it reaches the maximum number of time-steps ($t = T$). From a practical viewpoint, a neural network is considered to have converged when 10 successive outputs have a difference of less than $\varepsilon$ (in these experiments, $\varepsilon = 10^{-6}$). To compute $T_s$, we first define the "convergence function" $K(x, t, \mathbf{v})$:

$$K(x, t, \mathbf{v}) = \begin{cases} 0 \text{ if } \left|\gamma(x, \mathbf{v}, t)_i - \gamma(x, \mathbf{v}, t-n)_i\right| < \varepsilon, \\ \quad \forall n \in \{1, \cdots, 10\}, \forall i \in \{1, \cdots, k\} \\ 1 \text{ otherwise} \end{cases} \quad (15)$$

$T_s$ can now be defined as:

$$T_s(x, \mathbf{v}) = t \text{ such as } K(x, t, \mathbf{v}) = 1, t \in [0, \cdots, T] \quad (16)$$

$$T_c = \min(T, T_s(x, \mathbf{v})) \quad (17)$$

### 4.2.1 Objective Function

The main objective function (fitness) aims at checking that the maximum salience corresponds to the minimum activation in the corresponding output neuron (and consequently to the less inhibited action). Furthermore, we are interested in the best contrast possible, that is the selected action should be as little inhibited as possible and the other actions should be as inhibited as possible. To formulate this idea, let first define $m_{in}(\mathbf{v})$ (the index of the maximum value of $\mathbf{v}$) and $m_{out}(x, \mathbf{v})$ (the index of the minimum value of the output vector, for the input $\mathbf{v}$):

$$m_{in}(\mathbf{v}) = \arg\max_{i \in 1, \cdots, k}(\mathbf{v}_i) \quad (18)$$

$$m_{out}(x, \mathbf{v}) = \arg\min_{i \in 1, \cdots, k}(\gamma_c(x, \mathbf{v})_i) \quad (19)$$

We also need the index of the second minimum value of the output vector, $m^{(2)}_{out}(x, \mathbf{v})$:

$$m^{(2)}_{out}(x, \mathbf{v}) = \arg\min_{i \in 1, \cdots, k, i \neq m_{out}} (x, \mathbf{v})(\gamma_c(x, \mathbf{v})_i) \quad (20)$$

Let $I$ a set of $N$ random salience vectors in $[0.1, 1]$, uniformly distributed. Given that an action will be selected if the corresponding output channel is below $\gamma_0(x)$, we define the fitness to be maximized as the average selection success (the correct channel was disinhibited) weighted by the difference between the minimum output (the selected action) and the second minimum output (the first inhibited action). If all outputs except the minimum one are above $\gamma_0(x)$, $\gamma_0(x)$ is used to compute the difference. We thus foster the maximum inhibition of the selected channel but let the network free to use any value above $\gamma_0(x)$ for the other channels. The fitness is normalized such that the maximum value is 1.

$$F(x) = \frac{1}{|I| \cdot \gamma_0(x)} \sum_{\mathbf{v} \in I} \left(\delta_{m_{in}, m_{out}}(x, \mathbf{v}) \cdot \left|\gamma_c(x, \mathbf{v})_{m_{out}} ... \right.\right. \quad (21)$$

$$\left.\left. ... - \min(m^{(2)}_{out}(x, \mathbf{v}), \gamma_0(x))\right|\right)$$

where $\delta_{m_{in}, m_{out}}(x, \mathbf{v})$ denotes the Kronecker's delta:

$$\delta_{m_{in}, m_{out}}(x, \mathbf{v}) = \begin{cases} 1 \text{ if } m_{out}(x, \mathbf{v}) = m_{in}(x, \mathbf{v}) \\ 0 \text{ otherwise} \end{cases} \quad (22)$$

In these experiments, $N$ was fixed to $N = 1000$ and the same vectors were employed to evaluate all individuals.

### 4.2.2 Constraints

The search is restricted to networks that converge during the simulation time and that do not select anything if all the

[4] the CBG is parameterized by 22 synaptic weights and 3 neuron thresholds

**Table 1: Summary of launched experiments.**

| | Search space | Genotype | Channels |
|---|---|---|---|
| 1 | synaptic weights of CBG | 25 real numbers | 6 |
| 2 | topology and param. | direct encoding | 6 |
| 3 | topology and param. | map-based encoding | 6 |
| 4 | synaptic weights of CBG | 25 real numbers | 15 |
| 5 | topology and param. | direct encoding | 15 |
| 6 | topology and param. | map-based encoding | 15 |

**Table 2: Parameters used in the experiments.**

| Parameter / Genotype | Map-based | Direct enc. |
|---|---|---|
| min./max. nb. of nodes (rand. gen.) | 3 / 5 | 18 / 25 |
| min/max. nb. of links (rand. gen.) | 3 / 5 | 18 / 25 |
| prob. to add/remove a node | 0.05 / 0.05 | 0.05 / 0.05 |
| prob. to add/remove link | 0.05 / 0.05 | 0.05 / 0.05 |
| prob. to change each label | 0.1 | 0.1 |
| $\sigma$ for gaussian mutation | 0.05 | 0.05 |

input saliences are low. These restrictions are implemented using three constraints.

The first constraint $C_1(x)$ ensures that $\gamma_0(x)$ is greater than a minimum value $\gamma_{min}$, fixed to 0.09 in this work (the resting activity of the CBG model is 0.096). This means that all the actions should be sufficiently inhibited if all the saliences are null.

$$C_1(x) = \begin{cases} 0 \text{ if } \gamma_0(x) > \gamma_{min} \\ 1 \text{ otherwise} \end{cases} \quad (23)$$

The second constraint $C_2(x)$ ensures that small values are filtered (this behavior is observed in animals' basal ganglia) by sampling random saliences in $[0, 0.1]$ and checking that the outputs are greater than $\gamma_0(x)$:

$$C_2(x) = \begin{cases} 0 \text{ if } \gamma_c(x, \mathbf{v})_i > \gamma_0(x), \\ \quad \forall \mathbf{v} \in [0, 0.1]^k, \forall i \in \{1, \cdots, k\} \\ 1 \text{ otherwise} \end{cases} \quad (24)$$

Last, $C_3(x)$ checks that the tested neural network converges to a constant output vector before the end of the experiment, for all the tests performed on the neural network ($C_1(x)$, $C_2(x)$ and $F(x)$):

$$C_3(x) = \max_{\mathbf{v} \in I}(K(x), \mathbf{v}) \quad (25)$$

These constraints are enforced with the penalty method [7]: an arbitrary large penalty is added to the fitness each time a constraint is violated. Instead of maximizing $F(x)$, we thus maximize $F_c(x)$:

$$F_c(x) = -K(C_1(x) + C_2(x) + C_1(x)) + F(x) \quad (26)$$

where $K$ is an arbitrary large constant (e.g. $10^{10}$).

## 4.3 Experiments

To evaluate the relevance of the proposed approach to solve the action selection task, we ask three questions: (1) how well does the proposed map-based encoding (section 3.2.2) solve this task compared to an optimized CBG ([15], section 4.1)? (2) compared to a basic direct encoding (section 3.3)? and (3) does it scale up to high numbers of neurons?

To that aim, six different experiments have been launched (table 1), each of those consisting of 10 independent evolutionary runs, each of them with a budget of 200,000 evaluations. The same evolutionary algorithm, the same fitness and the same model of neurons were employed in all experiments; only the genotype/phenotype mapping was changed. The chosen evolutionary algorithm is a single-objective implementation of NSGA-2 [7], an elitist tournament-based evolutionary algorithm. Parameters are provided in table 2 and the source code is available online[5]. Experiments were carried in the Sferes$_{v2}$ framework [32].

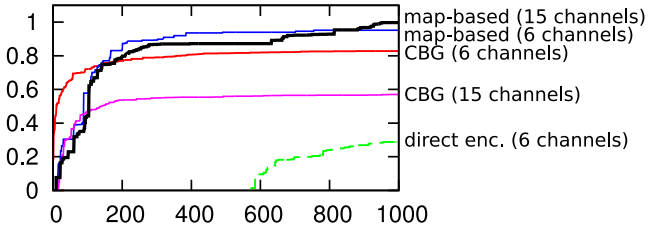



**Figure 4: Median fitness with regards to generation number (200 evaluations for each generation). The map-based encoding achieve the highest fitness while the direct encoding does not solve the task (median fitness is negative for direct encoding with 15 channels, therefore it does not appear on the graph).**

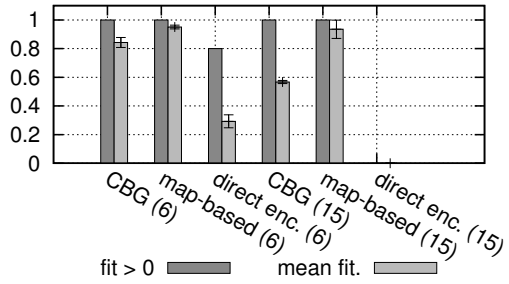



**Figure 5: Proportion of runs that satisfy the constraints (fitness $> 0$) after 200,000 evaluations and average fitness of those runs. All differences are statistically significant (Student T-test, $p < 0.01$) except between map-based (6) and map-based (15).**

# 5. RESULTS

For six channels, the optimized CBGs reach a fitness of about 0.8 (after 200,000 evaluations, median: 0.83, mean: 0.84, s.d.: 0.035; see figures 4 and 5). This means that it performs well, but some saliences are not well selected or the contrast is not maximum. Surprisingly, the map-based encoding leads to better results than this reference point by achieving a fitness of almost 1 with a low standard deviation (median: 0.951, mean: 0.950, s.d: 0.013). Overall, the map-based encoding easily and reliably solves the task of action selection. The discrepancy between map-based encoding and the reference point (optimized CBG) probably stems from the many biological constraints on the design of the CBG.

[5] `http://www.isir.fr/evorob_db`

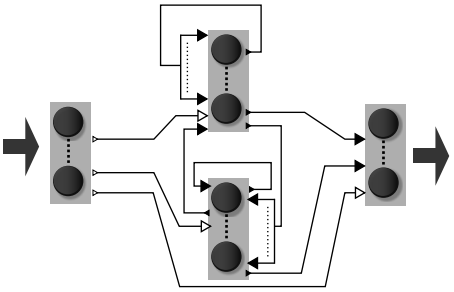


**Figure 6: Example of a neural network obtained with the map-based encoding for 6 channels (fitness: 1.0).**

With the basic direct encoding, about 600 generations (120,000 evaluations) were required to obtain a positive median fitness (i.e. more than half of the runs obtained at least one individual that satisfies the constraint). This contrasts with the map-based encoding runs, in which only a few generations were necessary to satisfy the same constraints. After 200,000 evaluations (1000 generations), 80% of the direct encoding runs satisfy the constraints (figure 5); however, the mean fitness is low (0.3) and corresponds to individuals that do not solve the task.

A typical neural network obtained with the map-based encoding is drawn on figure 6. In addition to the input and output maps, this network is made of only two hidden maps. They are linked using a mix of 8 "one to one" and 2 "one to all" connections (no Gaussian "one to all" connections are employed; as in the CBG model, they are useless to solve this problem). This neural network is therefore described by $4 \times 4 + 10 \times 4 = 56$ real numbers and graph topology. The same network described using a direct encoding would require (for six channels) $3+8\times6+2\times6 = 132$ real numbers in addition to graph topology. This illustrates the intuitive difference in size of the search spaces using map-based and direct encoding. The low score achieved by direct encoding suggests that the subspace explored with the map-based encoding is a useful restriction to solve the targeted cognitive function, in this case action selection.

The 15 channels experiments demonstrate how well the map-based encoding scales up. Since the neural network description is independent of the size of the map, results obtained with 6 or 15 channels are statistically not different. No runs managed to satisfy the constraints using the direct encoding. Surprisingly, although the description of the CBG is also based on maps, the CBG obtained lower fitness values with 15 channels than with 6 channels. We are investigating this issue.

# 6. CONCLUSIONS AND FUTURE WORK

A new encoding to evolve neural networks has been introduced in this paper. It is based on the "computational neuroscience toolbox" and especially on maps of neurons connected with regular connections schemes. This encoding is designed to explore with evolutionary algorithms a subspace of possible neural networks similar to the one explored in computational neuroscience. The proposed encoding has been compared to a basic direct encoding and to a hand-designed model on the task of action selection, an important function of basal ganglia in animals. Results show that:

- the map-based encoding easily and reliably solves the action selection task whereas a basic direct encoding never solves it;
- this description is independent of the size of maps and can therefore be used to evolve large and brain-like neural networks;
- the failure of direct encoding to solve the task validates the relevance of action selection as a benchmark.

The described map-based encoding could easily be improved by adding more "tools" from the computational neuroscience literature. Learning rules [12, 24, 42] are probably the most useful addition. Moreover, other basic tasks of the brain have to be investigated with the same encoding such as auto-calibration, working memory [13] or selective attention [41, 46, 40].

# 7. ACKNOWLEDGMENTS

This project was funded by the ANR EvoNeuro project, ANR-09-EMER-005-01.

# 8. REFERENCES

[1] G. E. Alexander, M. R. DeLong, and P. L. Strick. Parallel organization of functionally segregated circuits linking basal ganglia and cortex. *Annual Review of Neuroscience*, 9:357–381, 1986.

[2] R. Beer. Toward the evolution of dynamical neural networks for minimally cognitive behavior. In *From Animals to Animats 4*, pages 421–429, 1996.

[3] R. Beer and J. Gallagher. Evolving dynamical neural networks for adaptive behavior. *Adaptive behavior*, 1(1):91, 1992.

[4] A. Cangelosi, D. Parisi, and S. Nolfi. Cell division and migration in a 'genotype' for neural networks. *Network: Computation in Neural Systems*, 5(4):497–515, 1994.

[5] G. Chevalier and M. Deniau. Disinhibition as a basic process of striatal functions. *Trends in Neurosciences*, 13:277–280, 1990.

[6] P. Dayan, L. Abbott, and L. Abbott. *Theoretical neuroscience: Computational and mathematical modeling of neural systems*. MIT Press, 2001.

[7] K. Deb. *Multi-objectives optimization using evolutionnary algorithms*. Wiley, 2001.

[8] S. Doncieux and J.-A. Meyer. Evolving modular neural networks to solve challenging control problems. In *Proc. of EIS 2004*, 2004.

[9] O. Ekeberg. A combined neuronal and mechanical model of fish swimming. *Biological cybernetics*, 69(5):363–374, 1993.

[10] D. Floreano, Y. Epars, J. Zufferey, and C. Mattiussi. Evolution of spiking neural circuits in autonomous mobile robots. *International Journal of Intelligent Systems*, 21(9):1005–1024, 2006.

[11] D. Floreano and C. Mattiussi. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. MIT Press, 2008.

[12] D. Floreano and J. Urzelai. Evolution of plastic control networks. *Autonomous Robots*, 11(3):311–317, 2001.

[13] M. J. Frank, B. Loughry, and R. C. O'Reilly. Interactions between frontal cortex and basal ganglia in working memory: a computational model. *Cognitive, Affective, and Behavioral Neuroscience*, 1(2):137–160, 2001.

[14] W. Gerstner and W. Kistler. *Spiking Neuron Models: An Introduction*. Cambridge University Press, 2002.

[15] B. Girard, N. Tabareau, Q. C. Pham, A. Berthoz, and J.-J. Slotine. Where neuroscience and dynamic system theory meet autonomous robotics: a contracting basal ganglia model for action selection. *Neural Networks*, 21(4):628–641, 2008.

[16] F. Gruau. Automatic definition of modular neural networks. *Adaptive Behaviour*, 3(2):151–183, 1995.

[17] K. Gurney, T. J. Prescott, and P. Redgrave. A computational model of action selection in the basal ganglia. I. A new functional anatomy. *Biological cybernetics*, 84(6):401–410, 2001.

[18] K. Gurney, T. J. Prescott, and P. Redgrave. A computational model of action selection in the basal ganglia. II. Analysis and simulation of behaviour. *Biological cybernetics*, 84(6):411–423, 2001.

[19] G. S. Hornby. Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. In *Proc. of GECCO*, pages 1729–1736, 2005.

[20] G. S. Hornby and J. B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246, 2002.

[21] C. Igel, W. Erlhagen, and D. Jancke. Optimization of neural field models. *Neurocomputing*, 36(1-4):225–233, 2001.

[22] A. Ijspeert, A. Crespi, D. Ryczko, and J. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315(5817):1416–1420, 2007.

[23] A. Ijspeert, J. Hallam, and D. Willshaw. Evolving swimming controllers for a simulated lamprey with inspiration from neurobiology. *Adaptive Behavior*, 7(2):151–172, 1999.

[24] T. Kondo. Evolutionary design and behavior analysis of neuromodulatory neural networks for mobile robots control. *Applied Soft Computing Journal*, 7(1):189–202, 2007.

[25] J. Krichmar and G. Edelman. Brain-based devices for the study of nervous systems and the development of intelligent machines. *Artificial Life*, 11(1-2):63–77, 2005.

[26] W. Lohmiller and J.-J. E. Slotine. Contraction analysis for nonlinear systems. *Automatica*, 34(6):683–696, 1998.

[27] D. Marbach and A. J. Ijspeert. Online optimization of modular robot locomotion. In *Proc. of ICMA'2005*, pages 248–253, 2005.

[28] J.-A. Meyer, S. Doncieux, D. Filliat, and A. Guillot. *Biologically Inspired Robot Behavior Engineering*, chapter Evolutionary Approaches to Neural Control of Rolling, Walking, Swimming and Flying Animats or Robots, pages 1–43. Springer-Verlag, 2002.

[29] J.-A. Meyer, A. Guillot, B. Girard, M. Khamassi, P. Pirim, and A. Berthoz. The Psikharpax project: Towards building an artificial rat. *Robotics and Autonomous Systems*, 50(4):211–223, 2005.

[30] J.-B. Mouret and S. Doncieux. MENNAG: a modular, regular and hierarchical encoding for neural-networks based on attribute grammars. *Evolutionary Intelligence*, 1:187–207, 2008.

[31] J.-B. Mouret and S. Doncieux. Evolving modular neural-networks through exaptation. In *Proc. of IEEE-CEC*, 2009.

[32] J.-B. Mouret and S. Doncieux. Sferes$_{v2}$: evolvin' in the multicore world. In *Proc. of IEEE-CEC*, 2010.

[33] S. Nolfi and D. Floreano. *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. Bradford Book, 2004.

[34] T. Prescott. Action selection. *Scholarpedia*, 3(2):2705, 2008. doi:10.4249/scholarpedia.2705.

[35] T. J. Prescott, F. Montes-Gonzalez, K. Gurney, M. D. Humphries, and P. Redgrave. A robot model of the basal ganglia: Behavior and intrinsic processing. *Neural Networks*, 19:31–61, 2006.

[36] P. Redgrave. Basal ganglia. *Scholarpedia*, 2(6):1825, 2007.

[37] P. Redgrave, T. Prescott, and K. N. Gurney. The basal ganglia: A vertebrate solution to the selection problem? *Neuroscience*, 89:1009–1023, 1999.

[38] J. Reisinger, K. O. Stanley, and R. Miikkulainen. Evolving reusable neural modules. In *Proc. of GECCO*. Springer, 2004.

[39] N. Rougier. Dynamic neural field with local inhibition. *Biological cybernetics*, 94(3):169–179, 2006.

[40] N. Rougier and J. Vitay. Emergence of attention within a neural population. *Neural Networks*, 19(5):573–581, 2006.

[41] A. Slocum, D. Downey, and R. Beer. Further experiments in the evolution of minimally cognitive behavior: From perceiving affordances to selective attention. In *From Animals to Animats 6*, pages 430–439, 2000.

[42] A. Soltoggio, J. A. Bullinaria, C. Mattiussi, P. Dürr, and D. Floreano. Evolutionary Advantages of Neuromodulated Plasticity in Dynamic, Reward-based Scenarios. *Artificial Life*, 11:569, 2008.

[43] K. Stanley, D. D'Ambrosio, and J. Gauci. A Hypercube-Based encoding for evolving Large-Scale neural networks. *Artificial Life*, 15(2):185–212, 2009.

[44] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2), 2002.

[45] T. Trappenberg, M. Dorris, D. Munoz, and R. Klein. A model of saccade initiation based on the competitive integration of exogenous and endogenous signals in the superior colliculus. *Journal of Cognitive Neuroscience*, 13(2):256–271, 2001.

[46] J. Vitay, N. Rougier, and F. Alexandre. A distributed model of spatial visual attention. In *Biomimetic Neural Learning for Intelligent Robots*, pages 54–72. Springer, 2005.

[47] B. Webb. Robots in invertebrate neuroscience. *Nature*, 417(6886):359–363, 2002.