Emergence of perceptual states in nonlinear lattices: a new computational model for perception

Paolo Arena, Sebastiano De Fiore, Davide Lombardo and Luca Patané

***Abstract*— Insects show the ability to react to certain stimuli with simple reflexes using direct sensory-motor pathways, which can be considered as *basic behaviors*, while high brain regions provide secondary pathway allowing the emergence of a *cognitive behavior* which modulates the basic abilities. Taking inspiration from this evidence, a new general purpose perceptual control architecture is briefly presented and experimentally applied to a rover navigating in a cluttered environment. The core of the architecture is constituted by the Representation layer, where different stimuli, triggering competitive reflexes, are fused to form a unique abstract picture of the environment. Each representation induces a learnable modulation of the basic behaviors in order to determine the robot overall behavior. The representation is formalized by means of Reaction-Diffusion nonlinear partial differential equations, under the paradigm of the Cellular Neural Networks (CNNs), whose dynamics converges to steady-state Turing patterns. A suitable unsupervised learning leads to the shaping of the basins of attraction of the Turing patterns that, at the end of the leaning stage, represent a particular behavior modulation. Both simulations and robot experiments are drawn to demonstrate the potentiality and the effectiveness of the approach.**

I. Introduction

TAking inspiration from biological evidence, in the last years the research in the field of perception and robotics has developed a new paradigm known as Behavior-Based Robotics [1], in which the perceptual process is considered tightly interconnected with the agent behavioral needs.

Perception can be considered as an emerging complex phenomenon, where a large amount of heterogeneous information is fused to create an abstract and concise internal representation of the surrounding environment, which at the same time takes into account the needs and the motivation of the agent [2], while the whole process is mediated through a behavioral-dependent internal state [3].

Starting from these considerations and taking into account the latest results in the field of neurobiology [4] and the advancement in artificial cognitive system [5], we developed a general control architecture for implementing the sensing-perception-action cycle [6] to be potentially applied to different robotic platforms involved in several missions in cluttered environments. To this aim, we borrowed from the insect world neural structures and both simple and complex behaviors.

The internal representation of the external world, used for the action or behavior selection, is formalized by using Turing patterns [7], [8]; classical examples are animal coat patterns (stripes, spots and so on). In this work, Turing patterns are obtained in a nonlinear dynamical system, a Reaction-Diffusion CNN (RD-CNN) [9], as steady state conditions. More formally, they are attractors in nonlinear dynamical systems for particular sets of environmental stimuli and serve to modulate, through a reinforcement learning, competitive and concurrent basic behaviors. Learning is also introduced in the afferent layer to shape the basins of attraction of the Turing patterns in order to enhance this form of dynamic classification of the sensory events. This learning mechanism leads to the formation of abstract and flexible internal representations, mediated both by the environment and the agent needs. The second order cells within the RD-CNN mimick non-spiking neuron models, like neurons of group 12 in the pleural ganglia of the sea mollusk C*lione Limacina* [10]. These non spiking neurons are enrolled when a sudden speed variation has to take place, induced by external or even, as argued in [11], internal (e.g. humoral) motivations. These steady state plateau potentials in this neuron group lead to a suitable modulation of the animal motion, to fulfill a given motivation. From a structural point of view, in this work Turing patterns are generated within an array of non spiking neurons in a RD-CNN. They are used to form percepts, i.e internal representations of the external world information. It is to be pointed out that the same CNN cell neural structure, with a suitable modulation of its parameters, can generate spiking dynamics that were used to model the Central Pattern Generator in Bio-inspired robots [12]. Therefore, the RD-CNN structure can be considered as the basic unit to generate the suitable neural, self-organizing dynamics at different levels of an artificial brain architecture. It has to be outlined that several VLSI analog implementations of RD-CNNs have been developed [27]. Such chip prototypes are hosted within boards containing programmable digital hardware, in such a way that complex dynamics representing the solutions within the chip can be post processed allowing a real time implementation of the whole architecture for robot control.

In this work we assigned to the robot, as a simple case of study, a foraging task. To investigate the learning capability of the proposed architecture, both simulations in a virtual environment and experiments on a roving robot have been considered.

II. Control architecture

As in insects, the proposed perceptual architecture is organized in various control levels consisting of functional blocks, acting either at the same level, as competitors, or at distinct hierarchical levels showing the capability to learn more complex, experience-based behaviors [13]. Indeed a more complete insect brain architecture should include the

Paolo Arena, Sebastiano De Fiore, Davide Lombardo and Luca Patané are with the Dipartimento di Ingegneria Elettrica, Elettronica e dei Sistemi, Universitá degli Studi di Catania, Italy (email: {parena,lpatane}@diees.unict.it).
This work was supported by EU Project SPARK II

anticipation capability, to predict a sensory-motor path on the basis of another one. For example, exploiting phonotaxis, according to acoustic information, insects can predict to see the shape of another insect of the same species in a given space direction. This aspect was treated separately in a previous paper by using causal Hebbian rule in an array of spiking neurons [28]. In this paper, for the sake of brevity, we briefly discuss only the high level representation layer, where perception is formed, considering that the anticipation block can be added within this architecture to further enhance the capabilities.

The control architecture is reported in Fig.1. It consists of series of parallel sensory-motor pathways modulated by the representation layer, working as a nonlinear feedforward complex loop, whose output is trained to combine the basic behaviors. These are pre-wired and give knowledge baseline to the system. The loop is finally closed through the robot body and the environment. The control process can be divided into functional blocks: at the lowest level, we place the parallel pathways representing the basic behaviors, each one triggered by a specific sensor; at a higher level we introduce a representation layer that processes all the sensory information in order to define the final behavior. At the highest layer we introduce a lattice of non spiking neurons. This neural lattice shows distinct characteristics of a complex dynamical systems. The associated emerging neural states take on the meaning of percepts. These ones are then associated to suitable modulations of the basic behaviors, driven by a Reward function. In such a way, as it happens in insects, the basic behaviors, which are often life-saving sensory-motor pathways, are progressively enriched with emerging capabilities which incrementally increase the animal skills. The main focus is therefore on the application of complex dynamics to obtain a proper, complex, context-learned modulation of the basic skills. This process is the main characteristic of our approach which makes it different from the other control strategies, based on the subsumption architecture proposed by Brooks [14]. The latter in fact, uses a high level approach to face with the design of both basic behaviors and the coordination block. Here, complex dynamical systems are successfully used. Both architectures use a behavioral decomposition of the system to exploit parallel computation although the Subsumption network makes a rigid hierarchy among the basic behaviors: the lower ones cannot influence the upper ones, while the latter can act on the former. In our scheme, taking inspiration from the insect brain organization, all the basic behaviors are sensory-motor pathways elicited by only one sensory modality and on the same hierarchical level: knowledge is incrementally built upon their modulation, giving importance to one or the other, depending on the context. Under this perspective the proposed architecture resembles the Motor Schemas, introduced by Arkin [1]. Turing Patterns in RD-CNN are hosted, in our architecture, within a layer here called *Representation Layer*. This term is here not referred to a place where a predictive model of

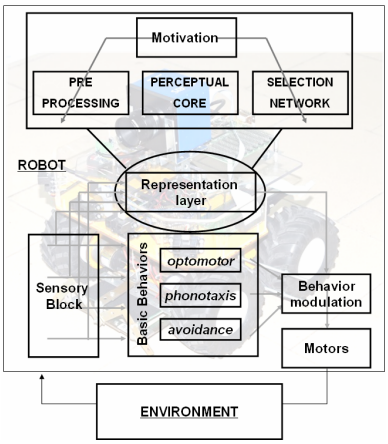

Fig. 1. Functional block diagram of the implemented control architecture. The interaction between the robot and the environment is realized by direct sensory-motor pathways, the *basic behaviors*, which are modulated by the representation layer. This high level function consists of a *preprocessing block*, a *perceptual core*, a *selection network*, while the *motivation* drives the learning process.

the body-environment interaction is learned. This is rather a layer where the single-sensory motor modalities, constituted by the parallel sensory motor pathways, are modulated in a feedforward way, taking into account all the incoming sensory stimuli. This leads to the emergence of a contextually self organising activity, focusing at modulating the basic behaviors. Within the *Representation Layer*, a motivation-driven learning is used to associate environment conditions to internal states (i.e. Turing patterns) that modulate the system behavior to fulfill the assigned task. The RD-CNN layer leads to the emergence of a *concise representation*. In fact all the sets of environment driven multisensory information leading to one rewarding behavior modulation are collected into a unique basin of attraction, represented by its steady state condition, depicted as a pattern. This pattern is a binary image, suitable for a very compact coding. It has to be outlined that the number of different patterns that are able to emerge from the neural RD lattice could be very high (on the order of some hundreds in a square $4x4$ network). So the number of different behavior modulations could be as large as needed to cope for very complicated and cluttered environment. The result of the behavior modulation leads to a particular robot motion, at each time t. This is formalized with a final action $A_F(t)$ that consists of a variable turning movement (rotation) and a fixed-length forward movement. The main characteristics of the cognitive architecture are described in the following subsections.

A. Sensory block

To face with the problem of autonomous navigation, the robot is provided with three distance sensors (covering the front, left and right hand side of the robot) for obstacle detection. Moreover, the robot receives information on the angle between the robot orientation and the direction robot-target and, in some simulations, also on the distance between the robot and the target. A graphic overview of the sensory

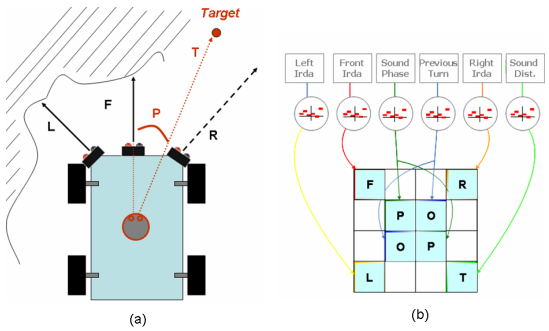

Fig. 2. (a) The robot is equipped with three distance sensors (F:Front, L:Left, R:Right) and a target sensor providing the phase (P) and distance (T) between robot and the target. (b) Initialization for the first layer CNN cells in the representation layer. The corner cells are set by obstacle stimuli (Front, Left, Right obstacle distance sensors) and by the target distance sensor, if present. The central cells are set by the previous executed rotation (O) and by the angle between the robot heading and the direction robot-target.

apparatus is sketched in Fig. 2.

B. Basic Behaviors

With *basic behaviors*, we refer to some "genetically" pre-wired reflexes, triggered by specific sensory events through direct sensory-motor pathways. Referring to crickets, these behaviours are: the capability showed by crickets to recover heading during walking, called *optomotor reflex* [15]; the female ability to follow the sound chirp emitted by a male, named *phonotaxis* [16]; and the ability to avoid obstacles, e.g. detected by the antennae.

At each time step t, the *optomotor reflex* tries to compensate for the previously executed rotation, as occurs in crickets that try to compensate its leg asymmetry to maintain the heading. Even if a detailed neural network was developed to carefully model the neural control system for such behavior [17], in this case a very simple rule was adopted consisting in: $A_o(t) = -A_F(t-1)$, where $A_o(t)$ is the rotation triggered by the optomotor reflex at the time step t and $A_F(t-1)$ is the turn executed by the robot at the previous time step.

The *obstacle avoidance* behavior guides the robot in avoiding obstacles perceived by distance sensors: $A_a(t) = f_a(d_F(t), d_L(t), d_R(t))$, here $A_a(t)$ is the rotation triggered by the obstacle avoidance, $f_a(\cdot)$ is a simplified version of the traditional potential field navigation algorithm [20] and $d_F(t)$, $d_L(t)$, $d_R(t)$ are the distances provided by the three distance sensors.

Finally, *phonotaxis* proposes a rotation, $A_p(t)$, aiming to compensate for the phase between the robot heading and the robot-target direction: $A_p(t) = f_p(p(t))$, where $p(t)$ is the phase between the robot and the sound source. The function $f_p(\cdot)$, used in this application, is a simplified version of the model for phonotaxis behavior, reported in [18].

C. Representation layer

Growing up from the basic behaviors, we consider as a *complex behavior* the ability to interpret "situations" in terms of robot-environment interaction (i.e. perception for action). The robot perceives using its sensory apparatus and processes at a cognitive level to optimize its behavior in relation to the mission assigned. The aim of the Representation layer, the highest control level within the whole cognitive process, is to achieve context dependent decisions. To this aim, all the available sensory modalities, each one separately being responsible of each single basic behavior, have to constitute the input to this layer. They are here incrementally transformed into environment *representations*, which lead to the modulation of the basic behaviors. These mechanisms are plastically modified by experience. This layer consists of a *preprocessing block*, a *perceptual core*, a *selection network* and a *motivation* layer, responsible for driving the learning process. Fig.1 shows its main components of the representation layer.

1) Preprocessing Block: The sensorial inputs, normalized in the range $[-1, 1]$, enter the preprocessing block: each stimulus is the input for a Sensing Neuron (SN) with piece-wise linear activation function, made-up of 10 amplitude-varying steps learned in an unsupervised way. Finally, each output of the SNs sets the initial condition for a cell of the nonlinear dynamical system that realizes the perceptual core of the Representation layer.

2) Perceptual Core: The creation of a concise representation of the environment is crucial for the cognitive process, since it is the result of the dynamic processing of the external stimuli.

In this work the CNN has been designed to generate, on the basis of information coming from sensory events, Turing patterns. At the afferent (i.e. input) level, an unsupervised learning algorithm plastically shapes the basins of attraction of the Turing patterns in order to adjust the classification of the information with respect to the robot motivation. To implement this feature, we use a nonlinear partial differential equation, discretised in space as a neural lattice made-up of second order cells. This constitutes a two-layers RD-CNN, able to generate Turing patterns [7]. The dimension of the network has been fixed to 4×4 on the basis of a previous work [21]. Each cell $c(i, j)$ of the two-layers RD-CNN has state variables ($x_{1;i,j}$ for the first layer and $x_{2;i,j}$ for the second layer, with $i, j = 1, .., 4$) and reads:

$$\begin{aligned} \dot{x}_{1;i,j} &= -x_{1;i,j} + (1 + \mu + \varepsilon)y_{1;i,j} - sy_{2;i,j} + D_1\nabla^2 x_{1;i,j} \\ \dot{x}_{2;i,j} &= -x_{2;i,j} + sy_{1;i,j} + (1 + \mu - \varepsilon)y_{2;i,j} + D_2\nabla^2 x_{2;i,j} \\ y_{h;i,j} &= \tfrac{1}{2}(|x_{h;i,j} + 1| - |x_{h;i,j} - 1|) \end{aligned} \tag{1}$$

where $y_{h;i,j}$ ($h = 1, 2$) is the output of the layer h of the cell $c(i, j)$ and D_1, D_2, μ, ε and s are parameters of the model. To satisfy the analytical conditions to obtain Turing pattern the parameters have been set to: $\mu = -0.7$, $\varepsilon = 1.1$, $s = 0.9$, $D_1 = 0.05$, $D_2 = 15$, $\gamma = 1/D_1 = 20$ [21].

As shown in Fig. 2.b, the output of each SN sets the initial conditions for the state variable of two central cells or a corner cell, which have been proven to have higher control than the other cells [21]. The initial conditions for the state variables of the second layer are set to zero for all the cells.

The RD-CNN evolves towards the condition in which all

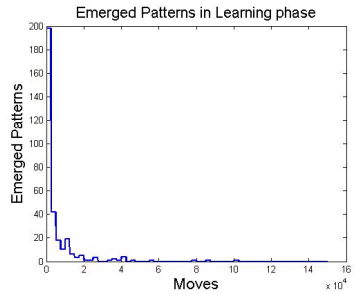

Fig. 3. Number of new patterns that emerge during learning when $\gamma = 20$.

the state variables of the first layer, i.e. the $x_{1;i,j}$, saturate at a value greater than 1 or lesser than -1. In this case, each output variable $y_{1;i,j}$ will be either 1 or -1, a condition that we consider a Turing pattern.

To simplify the successive processing, we associate a simple integer code for each Turing pattern as already discussed in [21]. Once preprocessed the external stimuli, we reset the CNN, set the initial conditions of the selected cells through the outputs of the SNs (Fig.2.b) and let the CNN evolve and generate a Turing pattern. Its code is stored in a *Pattern Vector* at the first occurrence. Each element of the pattern vector contains the *Pattern Code* and the step of its last occurrence (*Occurrence Lag*). The effect in terms of trend of new emerged patterns during learning is shown in Fig. 3.

The use of Turing patterns as steady states of a dynamical system implies a form of sensor fusion, i.e. we synthesize heterogeneous sensory information into a single attractor. At each step, the information coming from sensors is fused to form a unique abstract and concise representation of the environment, as discussed in the Section II.

3) Selection Network: The *Selection Network* associates each element q of the pattern vector with a set of three parameters (k_o^q, k_a^q, k_p^q) At the first occurrence of the pattern q, they are randomly chosen in the range $[0, 1]$ with the constraint that: $k_o^q + k_a^q + k_p^q = 1$. Then, the parameters are modified under the effect of the learning process acting at the efferent (i.e. output) stage of the Representation layer as explained in the following. After completed the learning process, at each time step t, once generated the Turing pattern $q(t)$, the corresponding modulation parameters are selected and the behavior that emerges is the weighted sum of the actions suggested by the basic behaviors at that time: $A_F(t) = k_o^q \cdot A_o(t) + k_a^q \cdot A_a(t) + k_p^q \cdot A_p(t)$.

4) Motivation layer and learning process: The association between Turing patterns and modulation parameters is learned through a reward-based reinforcement learning implemented by a simplified Motor Map (MM) [19], [21], whereas the fitness of each action is evaluated by means of a Reward Function (RF) defined as follows:

$$RF(t) = \sum_i h_i \cdot RF_i(t) \tag{2}$$

where RF_i represents the degree of satisfaction related to the basic behavior *i* where $i = o, a, p$:

$$\begin{array}{c} RF_o(t) = r_o(|A_F(t-1)|) \\ RF_a(t) = \sum_i r_i(e^{d_i(t)}) \\ RF_p(t) = r_p(|p(t)|) \end{array} \tag{3}$$

Here $A_F(t)$ is the action performed at time t, $d_i(t)$ is the distance between the robot and the obstacle detected by the sensor i ($i = Front(F), Right(R), Left(L)$) and $p(t)$ is the phase between the robot orientation and direction robot-target. The goodness of the behavior can be evaluated comparing the RF at each step by $DRF(t) = RF(t) - RF(t-1)$. A positive (negative) value for $DRF(t)$ indicates a successful (unsuccessful) behavior. Successful behaviors are followed by reinforcement, like in the Skinner's experiments [22] in order to maximize the RF. More in details, when the Turing pattern q emerges at the time step t, the behavior performed by the motor layer is:

$$A_F(t) = \sum_i (k_i^q + g_i^q(\xi)) \cdot A_i(t) \tag{4}$$

where $g_i^q(\xi)$ ($i = o, a, p$) are gaussian variables (zero-mean and unitary variance), the variance (σ_q^2 associated with the pattern q) determines the range of the *random search* for the optimal modulation parameters. After the execution of the behavior defined in (4), the $DRF(t)$ is evaluated and, in case it is greater than the average increase in the RF generated by q, called b_q, the modulation parameters are updated in direction the suggested by the random variable according to:

$$k_i^q(new) = k_i^q(old) + \varepsilon g_i^q(\xi) \tag{5}$$

where $\varepsilon = 0.1$ is the learning rate. Furthermore, the variance of the gaussian variable is decreased exponentially. In case $DRF < b_q$, the modulation parameters do not change.

If $DRF < 0$, the learning process acts on the afferent (input) association, realized by the SNs, between the stimuli and the initial conditions for the CNN cells aiming to establish the correct association between the sensory events and the internal representations (Turing patterns). In particular, our choice for the SNs activation function consists in an increasing function constituted by ten variable amplitude steps, θ_i ($1 \leq i \leq 10$), covering the whole input range $[-1, 1]$. At the beginning of the learning phase, all the steps have zero amplitude and, when we want to punish the system due to a $DRF < 0$, the step amplitudes are modified randomly in order to try to change pattern. The idea is that, when the action associated with the previous situation is no longer able to make the robot succeed in accomplishing the current task, a new pattern should emerge and the suitable action to this new environmental condition has to be learned by the robot. In such a way the sensorial stimuli will be divided into classes, associating different situations with patterns that generate rewarding behaviors. More in detail, if the action associated with the currently emerged pattern is unsuccessful (i.e. $DRF(t) < 0$), then the learning algorithm for each SN acts as follows:

- determine which of the RF components has suffered the highest decrease (e.g. the component associated with the Front side obstacle detector);
- for the selected SN determine the step amplitude θ_i related to the current input value;
- extract a number rnd from a zero-mean, uniformly distributed random variable r;
- the step amplitude θ_j is modified as: $\theta_i(new) = \theta_i(old) + rnd$, provided that it lies in the range $[-3, 3]$.

To guarantee the convergence of the algorithm, the variable rnd varies in the range [-m,m] where m, initially sets to 0.5, decreases at each step with an aging coefficient $m(new) = 0.999 \cdot m(old)$. The result is that the association between sensorial stimuli and Turing patterns is dynamically tuned by modulating the basins of attraction of the steady state patterns. The effect is that, at the beginning of the learning phase, a lot of pattern-action associations arise which are stabilized at later stages. This strategy, already effective, is going to be improved including the dependence on the Reward function fluctuations. More details on the whole mathematical model are given in [21].

III. Simulation Results

A. Simulation Setup

The software simulation environment, developed in $C++$, allows to create an arena constituted by walls, obstacles and targets. In the arena a robot equipped with a distributed sensory system can be simulated. The dimensions of the arena are 300×300 $pixels$: the learning and the test have been done with different configurations of obstacles. The simulated robot is equipped with three distance sensors, and one target sensor providing the phase between the robot orientation and the direction robot-target. The front side sensor detects obstacles within a limited range of 40 pixels, while the other two obstacle sensors are oriented at $-45°$ and $45°$ with respect to the robot heading (Fig. 2.a). All the sensors have a visual conus of $[-30°, 30°]$. It is to be noticed that, for all the distance sensors, the output is saturated to the limit of the detection range, so even if no obstacles are detected, the output of the sensor would be 40 $pixels$ for the front distance sensor, and 20 $pixels$ for the other two distance sensors. The target sensor has an unlimited range and provides the angle between the robot orientation and the robot-target direction. All the sensor outputs are scaled in the range $[-1, 1]$. The component of the RF in Eq. (3) were heuristically defined as: $r_o(t) = -A_F(t-1)$, $r_p(t) = -|p(t)|$, $r_F(t) = -e^{-8(d_F(t)+1)}$, $r_L(t) = -e^{-8(d_L(t)+1)}$, $r_R(t) = -e^{-8(d_R(t)+1)}$, where $d_F(t)$, $d_R(t)$, $d_L(t)$ are the distances detected by the sensors F, R, L, while $p(t)$ is the angle between the robot heading and the direction robot-target and $A_F(t-1)$ is the rotation made by the robot in the time step $t-1$. In the following simulations, the choice for the other parameters in Eq. (2) is $h_o = 1, h_a = 10, h_p = 10$. In this way more importance is given to the contribution of the obstacle information than to the target one, because the former is crucial to preserve the robot integrity. In particular the output coming from the front side obstacle sensor has the greatest weight in the RF. Through the definition of this reward function, we give to the robot knowledge about the task to be fulfilled, but it has no *a priori* knowledge about the correct way to interact with the environment. So the phase of the actions associated with each pattern is randomly initialized within the range $[-20°, 20°]$.

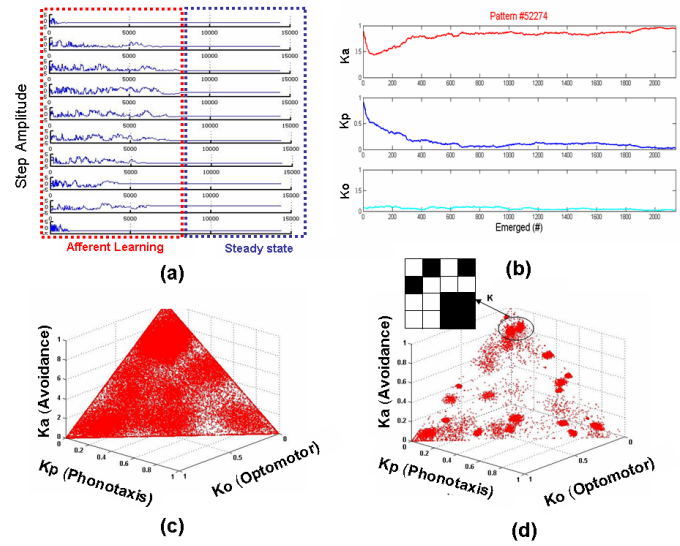

Fig. 4. a) Step amplitude for the activation functions of one SN (i.e. SN related to frontal distance sensor) during learning in a typical simulation. b) Evolution of the k_i^f for the most frequently emerged pattern (52274) in the first 8000 occurrences. c) The modulation parameters used in the first 30000 movements. d) The parameters used in the last 30000 movements with the indication of the region associated with the pattern 52274.

B. Learning phase

As far as the simulated robot is concerned, the task assigned to the robot consists in aiming a target avoiding obstacles. When the target is found, a new target appears in a random position. The learning phase lasts until one of the two conditions occurs: either the a_q averaged on the last 1000 patterns drops below 10^{-4} or 5000 targets have been found. At the beginning of the learning phase, the robot randomly modulates the basic behaviors due to the random initialization of the modulation parameters k_i^q $(i = a, o, p)$, which determine the robot heading. During the learning process, the Motor Map-like algorithm corrects the parameters associated with each pattern. Fig. 4 shows the evolution of the step amplitudes of one SN activation function during learning, the evolution of the k_i^f for the most frequently emerged pattern, i.e. $f = 52274$, and the modulation parameters used in the first and in the last 30000 actions of a typical simulation phase.

C. Testing phase

The testing phase is made every 30000 actions and consists of 10 target findings with the targets placed in different positions within the testing arena. To evaluate the benefit of the learning process, we match the result of the test with the

TABLE I

SIMULATION RESULTS.

	Fixed	Random	Learned
Average number of actions	166.8	176.3	28
Average number of collisions	95.4	40.7	5

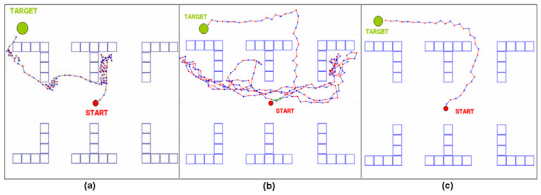

Fig. 5. Trajectories in the testing arena in the case of constant (a), randomly chosen (b) and learned (c) modulation parameters. The first two architectures (a-b) take a lot of time to reach the target and suffer from many collisions. From the learned modulation parameters, a very straightforward, although safe, behavior emerges.

case of using constant modulation parameters or randomly chosen modulation parameters.

The compared results, in terms of Average Number of Actions and Average Number of Collisions needed to find a target for Fixed, Random and Learned modulation parameters, are reported in Tab. I. the learning process leads to a dramatic reduction both in the average number of actions needed to reach a target and in the average number of collisions, demonstrating the effectiveness of the control architecture and its capability to generalize the representations. In particular, this feature has been proven by performing the test in a scenario that is different from that one used during learning. Fig. 5 shows examples of trajectories followed during the testing phase in case of fixed, random and learned modulation parameters.

IV. EXPERIMENTAL RESULTS

A. Experimental Setup

To cross-check experimentally the promising results obtained in numerical simulations we use a roving robot moving in a real environment. The experimental platform consists of a mobile robot interfaced, via Bluetooth and a $RS232$ serial communication, to a computer, where the control architecture has been implemented. The roving robot is a classical four wheel drive rover controlled through a differential drive system. The robot dimensions are $35 \times 35\ cm^2$. Each distance sensor equipped on board, is a pair of short and long-distance infrared sensors providing the distance to obstacles in the range $3 - 80$ cm. The frontal side is covered by two pair of sensors with a unique output that mediates the two data. The roving robot is attracted by a sound source (target) emitting the cricket characteristic sound chirp.

The measure on the angle between the robot heading and the direction robot-target is provided by two microphones and an analog board capable to recognize the cricket chirp and to provide two output signals proportional to the intensity of the sound at each microphone. The robot is controlled by a $STR7x$ family microcontroller, which acquires analog signals via ADC at $7.2\ kHz$ and the bearing with respect to the magnetic North provided by a digital compass. Furthermore the microcontroller drives four torque-controlled DC motors and handles the communication with the computer. This platform has been already used as test bed for experiments of new cognitive architectures [25], [26].

B. Experimental Results

The experiments have been carried out within a $3 \times 3\ m^2$ arena where two obstacles and a sound source have been placed. In the experiments, we test the architecture with the learned modulation parameters compared with the case of fixed modulation parameters. The constant behavior modulation parameters were chosen through a manual tuning aiming at optimize the global performance of the robot. The parameters used in the following experiments are: $K_a = 0.35, K_p = 0.1, K_o = 0.05$. For each case, we let the robot perform different trials starting from the same position. Fig. 6 shows the trajectory followed in the best and in the worst trial for the two cases. The path length to reach the target, obtained with fixed modulation parameters, after several trials, has a mean value $P = 4.7m$ and a standard deviation $\sigma = 1.01m$ whereas the introduction of the representation layer produces an improvement obtaining $P = 2.7m$ and a $\sigma = 0.26m$. Even in these experiments, the performance improvement is evident in terms of path length to reach the target and robustness of the behavior against the noise present in the environment, which can be observed by the low variance in the path length along the different trials. It is to be noticed that using the fixed modulation parameters as a benchmark, means to eliminate the RD-CNN from the architecture, and exploiting only the basic behaviors, whose role in the action selection is constant throughout the experiment. It is apparent that the robot, basing only into its basic behaviors, could solve the problem assigned (except in the case in which a very bad selection of constant and random modulation parameters is chosen), succeeding, sooner or later, in reaching the target. The importance of the representation layer, introduced in this work, is to add to the architecture incremental capabilities of building knowledge, based on the environment. At the beginning of the learning

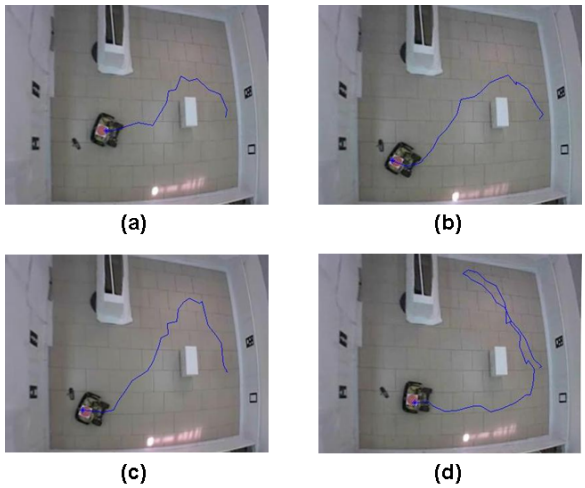

Fig. 6. Best and worst trial in the case of learned $(a-b)$ and constant $(c-d)$ modulation.

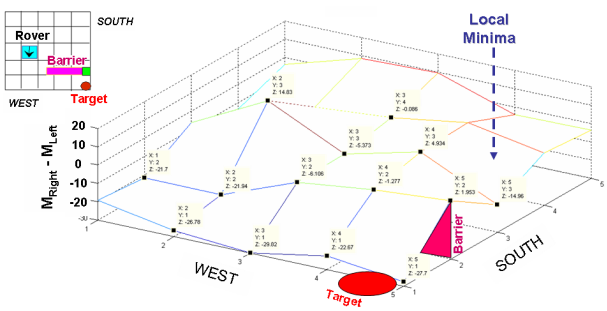

Fig. 7. Characterization of the phonotaxis behavior when the sound source is behind a barrier. The value reported in each point of the grid represents the turning angle (a positive value correspond to a right turn) produced by the basic behavior of phonotaxis when the robot is placed in that position in the arena and oriented toward west. A local minima and consequently a trapping condition for this behaviour is indicated in the map.

phase, the role of the Turing Pattern Generator is negligible, and the robot moves only according to the basic behaviors. As learning proceeds, the robot acquire the capability to exploit the space-varying combination of the basic behavior to improve its performance in relation to its motivation.

It is well known that one of the major problems in standard navigation strategies (e.g. Potential field) is to be stacked in local minima. For this reason we performed a test in an environment where the sound target is placed behind a barrier whose height is enough to be detected by distance sensors, allowing, at the same time the robot to sense the sound target. This creates a clear local minimum, since the presence of an obstacle creates an avoidance reaction by means of the avoidance behaviour, while the target is still attractive

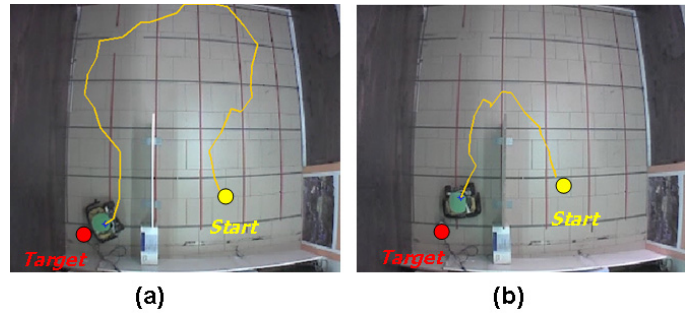

Fig. 8. Robot trajectory obtained with fixed (a) and variable (b) modulation of the basic behaviors.

through the phonotaxis behaviour. Fig. 7 shows a scheme of the arena used for this experiment (in the left upper part). To estimate the sound distribution in the arena, a series of acquisitions were made in different locations (discretising the arena in a $5x5$ grid). In particular the robot, oriented toward West, was placed in each point of the grid and the output of the target sound sensor was acquired. The $M_{right} - M_{left}$ variable reported in Fig. 7 indicates the rotation angle (e.g. a positive angle indicates a right turn) produced by the phonotaxis behavior. It could be noticed the presence of a local minimum in this scenario. As shown in Fig. 8 the robot is able to escape from this blind alley both with and without the representation layer. However, in the case of fixed behavior modulation, the solution is found thanks to a very careful choice of the gains. In particular the high enough value of the avoidance gain ($K_a = 0.35$) allows to discard the phonotaxis information to escape the deadlock but, at the same time, this high preference in avoiding obstacles increases the mean path to reach the sound target (P= 5.6 m). When the previously learned representation layer is used to modulate the basic behaviors, the deadlock is suitably overcame by using the avoidance behavior that however is predominant only in the first part of the trajectory while afterwards the other behaviors and in particular phonotaxis become more important leading to a drastically reduction of the mean path to reach the target (P= 2.7 m).

Videos, including tests in very noisy and dynamically changing environments, are available on the web [23], together with the high resolution version of the figures reported in this paper.

V. Remarks and Conclusions

In this paper a new control architecture for the sensing-perception-action loop in robots is described and validated through simulations and experiments on autonomous navigation. The control architecture is based on some predefined

basic abilities, called basic behaviors, which are modulated by the *Representation Layer*, which learns to associate set of sensory events with specific Turing patterns and with modulation parameters. Here, unlike similar approaches referring to behavior based robotics, we used complex dynamics to explore attractor based nonlinear computation and a simple reward based learning, to associate rewarding behavior modulation to contextual information coming from sensors. The whole sensory system depicts the environment scene as perceived by the robot. It is clear that within this information, the salient details about the robot body and position in the environment are naturally used to achieve an efficient, embodied and situated knowledge. It is to be underlined that algorithms dedicated to face with navigation tasks could even give better results: the potentiality of our approach lies in its generality. In fact the approach can be easily migrated to other robotic platforms, redefining the basic behaviors, and to other applications, redesigning the reward function. The approach, for example, is being actually applied to a more complex structure, an hexapod robot [24], where the control actions are much more complex, and the basic behaviors could include, for instance, not only avoiding obstacles by turning, but also climbing over steps. In this case patterns can indicate the particular scheme of leg motions, which should be applied in front of particular environment conditions. Presently the use of complex dynamics to achieve contextualization does not enable the capability to make prediction on sequences of behaviors useful to reach the target. Indeed this could be inserted very easily by implementing chains of successful behavior modulations, but we are currently working at exploiting the complex dynamics within the Turing Pattern generator to include prediction capabilities. The above described framework is suitable to be included in a more complex bio-inspired architecture aiming to emulate an insect brain at least from a functional point of view. A wider set of heterogeneous sensors such as cameras could be included. The implementation of the whole architecture on board on the robot in view of a more complete and autonomous interaction with complex and cluttered environments is also envisaged.

REFERENCES

[1] R. C. Arkin, *Behaviour Based Robotics*, MIT Press, Cambridge, 1991.

[2] R. A. Brooks, "Intelligence without reason", *in 12th Int. Conference on Artificial Intelligence*, eds. J. Mylopoulos and R. Reiter, San Mateo, CA; 1994.

[3] S. Nolfi, "Power and Limits of Reactive Agents", *Neurocomputing*, vol. 42, no.1, pp. 119-145, 2002.

[4] W.J. Freeman, "How and why brains create meaning from sensory information", *Int. Jour. of Bif. and Chaos*, vol.14, pp.515-530, 2004.

[5] D. Vernon, G. Metta and G. Sandini, "A Survey of Artificial Cognitive System: Implications for the Autonomous Development of Mental Capabilities in Computational Agents", *IEEE Transections on Evolutionary Computation*, vol. 11, no.2, pp. 151-180, 2007.

[6] K. Lynch *The Image of the City*, MIT Press, Cambridge, 1960.

[7] A. M. Turing, "The chemical basis of morphogenesis", *Phil. Trans. Roy. Soc. Lond. B*, vol. 237, pp. 37-72, 1952.

[8] J. D. Murray, *Mathematical Biology I: An Introduction (third edition)*, Springer-Verlag, NJ; 2002.

[9] L. Goras and L. Chua, "Turing Patterns in CNNs - Part I, II. *IEEE Trans. Circuits and Systems - I*, vol. 42, pp. 602-626, 1995.

[10] G. Orlovsky, T. G. Deliagina, S. Grillner, *Neuronal Control of Locomotion: From Mollusc to Man*, Oxford Neuroscience, 1999.

[11] Yu.I. Arshavsky, G.N. Orlovsky, Yu.V. Panchin, and G.A. Pavlova, "Control of locomotion in marine mollusc Clione limacina VII. Reexamination of type 12 interneurons", *Exp Brain Res.*, vol. 78, pp. 398-406, 1989.

[12] P. Arena, L. Fortuna, M. Branciforte, "Reaction-Diffusion CNN Algorithms to Generate and Control Artificial Locomotion", *IEEE Trans. on Circuits and Systems, I*, vol. 46(2), pp.259-266, 1999.

[13] J. Wessnitzer, B. Webb, "Multimodal sensory integration in insects - towards insect brain control architectures", *Bioinspiration and Biomimetics* vol. 1, pp. 63-75, 2006.

[14] R.A. Brooks, "A robust layered control system for a mobile robot", *IEEE Jour. of Robotics and Automation*, Vol. 2, no. 1 pp. 14- 23, 1986.

[15] H. Böhm, K. Schildberger, F. Huber, "Visual and acoustic course control in the cricket Gryllus-bimaculatus", *Journal of Experimental Biology* vol. 159, pp. 235-248, 1991.

[16] B. Webb and T. Scutt, "A simple latency dependent spiking neuron model of cricket phonotaxis", *Biological Cybernetics*, vol. 82(3), pp. 247-269, 2000.

[17] P. Russo, B. Webb, R. Reeve, P. Arena, L. Patané, "A Cricket-Inspired Neural Network for FeedForward Compensation and Multisensory Integration", *in IEEE Conference on Decision and Control and European Control Conference 2005*, Sevilla, Spain, 2005.

[18] A. Horchler, R. Reeve, B. Webb and R. Quinn, "Robot phonotaxis in the wild: a biologically inspired approach to outdoor sound localization", *Advanced Robotics*, vol.18, no.8, pp. 801-816, 2004.

[19] K. Schulten, "Theoretical biophysics of living systems". In H. Ritter, T. Martinetz and K. Schulten (Eds.), *Neural computation and self-organizing maps: An introduction*. Addison-Wesley, New York; 1992.

[20] J. Borenstein and Y. Koren, "The vector field histogram fast obstacle avoidance for mobile robots", *IEEE Journal of Robotics and Automation*, vol. 7, pp 278-288, 1991.

[21] P. Arena, P. Crucitti, L. Fortuna, M. Frasca, D. Lombardo and L. Patané, "Turing patterns in RD-CNNs for the emergence of perceptual states in roving robots", *Int. Journal of Bifurcation and Chaos*, vol. 17, no.1, pp. 107-127, 2007.

[22] B. F. Skinner, *About behaviorism*, Alfred Knopf, New York; 1974.

[23] EU Project SPARK II, online at *www.spark2.diees.unict.it/BehaviorModulation.html*.

[24] EU Project SPARK I, online at*www.diees.diees.unict.it/Robot.html*.

[25] P. Arena, L. Fortuna, D. Lombardo, L. Patané, "Perception for action: Dynamic spatiotemporal patterns applied on a roving robot", *Adaptve Behavior*, vol. 16 (2/3), pp. 104-121, 2008.

[26] P. Arena, S. de Fiore, L. Fortuna, L. Patané, "Perception-action map learning in controlled multiscroll system applied to robot navigation", *Chaos*, vol. 18, 043119 pp. 1-16, 2008.

[27] P. Arena, L. Fortuna, M. Frasca, L. Patané, "A CNN-based chip for robot locomotion control", *IEEE Trans. On Circuits and Systems-I*, Vol. 52, No. 9, pp. 1862-1871, 2005.

[28] P. Arena, L. Fortuna, M. Frasca, L. Patané, "Learning anticipation via spiking networks: application to navigation control", *IEEE Trans. on Neural Networks*, vol. 20(2), pp. 202-216, February 2009.