# Perception-action map learning in controlled multiscroll systems applied to robot navigation

Paolo Arena,[a] Sebastiano De Fiore,[b] Luigi Fortuna, and Luca Patané[c]
*Dipartimento di Ingegneria Elettrica Elettronica e dei Sistemi, Universitá degli Studi di Catania, viale A. Doria 6, 95125 Catania, Italy*

(Received 24 April 2008; accepted 24 September 2008; published online 2 December 2008)

In this paper a new technique for action-oriented perception in robots is presented. The paper starts from exploiting the successful implementation of the basic idea that perceptual states can be embedded into chaotic attractors whose dynamical evolution can be associated with sensorial stimuli. In this way, it can be possible to encode, into the chaotic dynamics, environment-dependent patterns. These have to be suitably linked to an action, executed by the robot, to fulfill an assigned mission. This task is addressed here: the action-oriented perception loop is closed by introducing a simple unsupervised learning stage, implemented via a bio-inspired structure based on the motor map paradigm. In this way, perceptual meanings, useful for solving a given task, can be autonomously learned, based on the environment-dependent patterns embedded into the controlled chaotic dynamics. The presented framework has been tested on a simulated robot and the performance have been successfully compared with other traditional navigation control paradigms. Moreover an implementation of the proposed architecture on a Field Programmable Gate Array is briefly outlined and preliminary experimental results on a roving robot are also reported. © *2008 American Institute of Physics*. [DOI: 10.1063/1.3005783]

**Recently, a new family of chaotic systems, generating multiscroll attractors, was introduced. This type of chaotic dynamics was taken into consideration by the authors of this paper to model perception for action mechanisms, with application to autonomous mobile robot control. In this paper the authors complete the methodology by adding an unsupervised learning structure to the controlled multiscroll system. In this way the robot autonomously learns to answer, to a given environmental stimulus set, with that action which contributes to increase an *a priori* fixed reward function, representing the robot assigned mission. The simple learning strategy adopted is suitable to be implemented in hardware for real time working together with the already implemented multiscroll system.**

## I. INTRODUCTION

After the introduction of multiscroll chaotic systems, several circuit solutions for a real time implementation were proposed (see Refs. 1 and 2 for details). In particular, the potential role of multiscroll dynamics in perception was envisaged in Ref. 2. In fact, the peculiar chaotic attractor shown by a multiscroll system [as shown in Fig. 2(a) below], consists of a number of chaotic scrolls distributed all over the phase plane. These scrolls can be easily varied in number and position by modulating the system nonlinearity. In recent works,[3–5] a bio-inspired approach based on a new control technique, applied to multiscroll systems, was introduced to deal with navigation control in simulated and real environments. These works are based on a paradigm which considers perception no longer as a standalone process, but as a holistic and synergetic one, tightly connected to the motor and cognitive system.[6] Perception is now considered as a process indivisible from action; behavioral needs provide the context for the perceptual process, which, in turn, works out the information required for motion control. In this work a framework for action-oriented perception has been realized employing chaotic dynamical systems controlled through a new technique called *weak chaos control* (WCC).

The WCC technique allows us to create perceptual states directly related to the concept of embodiment and situatedness.[7] Moreover WCC allows a suitable control of the various scrolls, in front of the incoming environment stimuli. This fine control constitutes the basis for the emergence of environment-shaped dynamic patterns, that in this paper are used to face with the action-oriented perception task.

The approach proposed was partially inspired by Freeman's neurobiological theories. His approach, in the field of odor perception, recognizes the existence of stimulus-specific activity patterns.[8,9] Without entering in detail, the cerebral cortex processes information coming from objects identified in the environment from receptors by enrolling dedicated neural assemblies. These are nonlinear dynamical coupled systems, whose collective dynamics, joined to motivation-induced behavioral changes, creates stimulus-specific activity pattern shapes. Freeman and co-workers, in their extensive experimental studies on the dynamics of sensory processing in animals,[10] conceive a dynamical theory of

[a]Electronic mail: parena@diees.unict.it. Telephone: +39-095-7382307. Fax: +39-095-330793.
[b]Electronic mail: sdefiore@diees.unict.it.
[c]Electronic mail: lpatane@diees.unict.it.

perception. Through the electroencephalogram (EEG), Freeman evaluated the action potentials in the olfactory bulb and he noticed that the potential waves showed a typical chaotic behavior. The mental shaping of neural activity patterns, stimulus-specific, is the result of a chaotic dynamics in the sensory cortex in cooperation with the limbic system that implements the supporting processes of intention and attention.[9]

The application of chaotic models as basic blocks to reproduce adaptive behaviors[11–13] is an interesting aspect of the current research focussed at modeling intelligent behaviors and transferring them to robotics. The idea is that chaos provides the right properties in term of stability and flexibility, needed by systems that evolve among different cognitive states.

Skarda and Freeman[11] have investigated the role of chaos in the formation of meanings. According to their works, chaotic dynamics can constitute the normal background activity of neural systems. If the system (i.e., neuronal ensembles) is perturbed by sensory inputs, the result is a transition into a new attractor, depending on the state of the system, motivation, and environment. The advantage of a chaotic background activity with respect to noise is that noise cannot be easily controlled (e.g., stopped or started), whereas a chaotic dynamic can be easily enslaved and suppressed, switching among different attractors. Therefore chaotic dynamics can further improve reactive system capabilities; in fact chaotic systems generate a wide variety of attractors that can be controlled guiding the transit from one to another, similarly to the emergence of adaptive behaviors in living beings.

Freeman's studies led to a model, called K-sets, of the chaotic dynamics observed in the cortical olfactory system. This model has been used as a dynamic memory, for robust classification and navigation control of roving robots.[14–16]

The architecture proposed here, taking into consideration the relevant principles previously underlined, is based on the control of chaotic dynamics to learn adaptive behaviors in roving robots. The main aim is to formalize a new method of chaos control applied to solve problems of perceptual state formation.

The WCC approach, following these guidelines, uses a chaos control technique, applied to a multiscroll chaotic system.[1] WCC is a general technique that can be applied to several chaotic systems.[2,17] All sensory signals are mapped as different potential reference dynamics used to control the chaotic system. This creates associations among sensor information and a particular area located within the multiscroll phase plane, in a way that reflects the topological position of the robot within the sensed environment. This property, joined to the robot motivation defined through a suitable reward function, contributes to create perceptual meanings of the mental patterns, useful for behavior modulation. The term "perceptual meaning," within the framework of action-oriented perception, is here referred to the capability of a system to build a meaning, based both on sensor information and on the context of the action that the agent (animal or robot) performs to satisfy its motivation.

The multiscroll chaotic system has been chosen instead of other regular chaotic systems to exploit its modularity in the design of the multiscroll attractors. The multiscroll system can be designed depending on the robot embodiment. Each scroll can be related to the presence of one or more active sensors; in this way motion control can be used to improve the perceptual process by placing the robot sensors in the most appropriate positions. Consequently, the most important information can be selected, discarding what is less useful for the given task.

The aim of this paper is to assess the WCC navigation technique performance presented in previous works[3–5] in relation to the problem of action-oriented perception.

In fact in the referred papers above, the controlled system dynamics was linked to a given robot action using fixed rules, *a priori* selected by the designer. On the contrary a real perceptual architecture should be able to autonomously associate, to a given environmental state, as recorded by sensors the action that contributes to increase a given reward. In this manuscript we address such an issue by adding an associative learning layer (called in the following *action selection stage*). Since associations are incrementally learned via feedback through the environment, as recorded by sensors, placed in a given position in the considered robotic structure, this technique allows us to also implicitly consider, within the robot control system, information about the robot structure and dimensions, including the sensor position, thus situating the robot within the environment. The improvements provided by the action selection layer are shown in several works,[18,19] in relation to different applications.

With regards to the type of learning used in the *action selection stage*, a fundamental issue is that, given the high degree of information that the chaotic system can embed, in terms of sensor measures, the task to be solved by the associative layers should be really simple. Candidate algorithms are based on reinforcement learning (RL); this is used to find how to map situations to actions, so as to maximize a predefined reward signal. The system does not know which actions to take, but instead it must discover which actions yield the best reward by trying them.[33] In the proposed architecture, to close the loop, the perception-action link is plastically learned, introducing a simple reward-based mechanism, derived by the *motor map* paradigm. It is inspired by the paradigm of Kohonen nets[20] and is able to plastically react to localized excitation by triggering a movement (like the motor cortex or the superior colliculus in the brain).[21] The motor map paradigm has been already successfully applied to solve navigation control problems.[19,22] The proposed navigation control technique, based on the action-oriented perception paradigm, has been tested in a simulation environment and subsequently was implemented using a hardware platform.

As clearly outlined, our approach is focused to design a new methodology for action-oriented perception, with enhanced capabilities of real time implementation on moving robots. The strategy aims at autonomously learning fast, reflex-based "action-maps." This leads to the limitation that the approach is not designed to make any form of action planning or behavior planning, i.e., no possibility is allowed to learn chains of actions within any temporal horizon. This

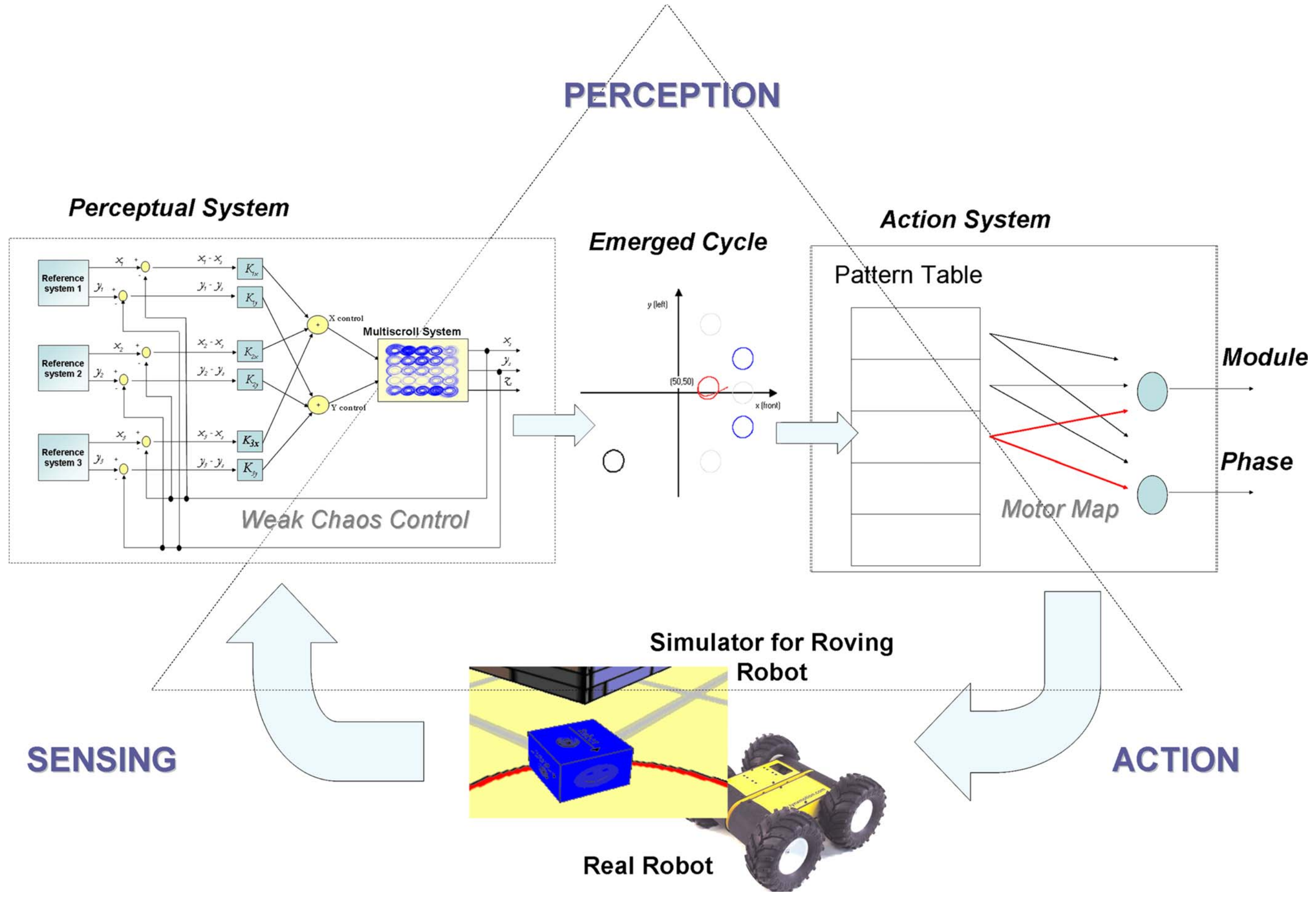


FIG. 1. (Color online) Control architecture.

would require more complex architectures and, of course, a much longer learning phase. This is outside of the scope of the present work.

In the next section we will describe the control architecture that reproduces the sensing-perception-action loop. Section III is devoted to illustrate a case of study; the application of the control architecture to a problem of autonomous navigation. Simulation results and comparisons with other traditional approaches are reported in Sec. IV, whereas Sec. V describes a hardware implementation based on Field Programmable Gate Arrays. Finally, in Secs. VI and VII we will give some remarks and draw the conclusions.

## II. WCC CONTROL ARCHITECTURE

The proposed control architecture is a general framework designed to deal with action-oriented perception mechanisms. The idea is that an agent with no *a priori* knowledge except for its body structure, can learn how to accomplish a given task performing the action-oriented perception loop. The architecture is general and can be used to control different kinds of agents (e.g., roving and legged robots, robotic arms) that need to interact with an external environment.

In this work the whole control architecture is described from a general perspective and an application in the case of autonomous roving robots dealing with basic navigation tasks is also reported and compared with other, more traditional, solutions.

The overall architecture for action-oriented perception is shown in Fig. 1, where two main blocks can be distinguished:

- the perceptual block, which, via the WCC technique, leads to the emergence of input-induced dynamics patterns (in our case periodic trajectories);
- the action selection network, where actions are learned, at the aim to increase the values of a predefined reward function, representing the robot motivation.

As discussed above, taking inspiration from biological evidence,[11,12] a chaotic dynamical system was adopted as an engine able to generate emergent solutions, thanks to the richness of different attractors that can arise in the controlled multiscroll.

The chaotic system, joined to the action selection network, is used here as a basic block to create a meaning of the sensory input by means of a control technique that can easily allow us to enslave and suppress the chaotic dynamic, switching among different attractors.

For autonomous navigation in a roving robot, when no stimuli are perceived (i.e., there are no active sensors) the multiscroll system evolves in a chaotic behavior and the robot continues to explore the environment performing an action determined by the chaotic evolution of the multiscroll system. When external stimuli are perceived, the controlled system converges to a cycle (i.e., a periodic pattern) that depends on the contribution of active sensors through the control gains. The corresponding action will plastically depend on the characteristics of the cycle, as a result of the learning algorithm taking place within the action selection layer. When the stimuli stop, the system falls back into a chaotic dynamic.

The learning mechanism is driven by a reward function (RF), designed on the basis of the mission to be accomplished (in our case navigation). Each block is described in the following, referring to Fig. 1.

### A. Perceptual system

The core of this layer is the weak chaos control method (WCC).[5] The crucial advantage of this approach is the possibility to create stimulus-dependent dynamic patterns, that can be used for real time generation of suitable actions. To model this behavior a chaotic system, proposed by Chen,[1] has been used as a plastic layer in which perceptual states can emerge. The chaotic behavior of Chen's multiscroll system can be enslaved to regular periodic patterns (here used as perceptual states) by using the sensory stimuli as reference control signals. The multiscroll system has been preferred to a regular chaotic one because we need to map the agent embodiment in the system dynamics and a very simple solution, that will be further discussed in the following, consists of a topological distribution of the sensory inputs in the system phase plane. Chen's system can be easily designed in order to increase the number and position of scrolls and this opportunity allows us to handle with a large number of distributed sensors.

The control mechanism has been realized via a feedback on the state variables $x$ and $y$ controlled to track the reference cycles. The equations of the controlled multiscroll system can be written as follows:

$$\dot{x} = y - \frac{d_2}{b} f(x;k_2;h_2;p_2,q_2) + \sum_i K_{x_i}(x_{m_i} - x),$$

$$\dot{y} = z + \sum_i K_{y_i}(y_{m_i} - y), \tag{1}$$

$$\dot{z} = -ax - by - cz + d_1 f(x;k_1;h_1;p_1,q_1) + d_2 f(y;k_2;h_2;p_2,q_2),$$

where $x_{m_i}$ and $y_{m_i}$ are the reference signals, $K_{x_i}$ and $K_{y_i}$ are the control gains and the following so-called saturated function series $f(x;k;h;p,q)$ has been used:

$$f(x;k;h;p;q) = \sum_{i=-p}^{q} f_i(x;k;h), \tag{2}$$

where $k>0$ is the slope of the saturated function, $h>2$ is called *saturated delay time*, $p$ and $q$ are positive integers, and

$$f_i(x;k;h) = \begin{cases} 2k & \text{if } x > ih+1, \\ k(x-ih)+k & \text{if } |x-ih| \leq 1, \\ 0 & \text{if } x < ih-1, \end{cases}$$

$$f_{-i}(x;k;h) = \begin{cases} 0 & \text{if } x > -ih+1, \\ k(x+ih)-k & \text{if } |x+ih| \leq 1, \\ -2k & \text{if } x < -ih-1. \end{cases}$$

The parameters used in the following ($a=b=c=d_1=d_2=0.7$, $k_1=k_2=50$, $h_1=h_2=100$, $p_1=p_2=1$, $q_1=q_2=2$) permit us to generate a 2D $5\times 5$ grid of scroll attractors.[3]

A key point of this approach is that the reference cycles distribution in the phase plane $x-y$ reflects the topological distribution of the sensory information taking into account the agent embodiment. Moreover, the sensor range depicts the current robot operating space, which is dynamically encoded within the phase space of the multiscroll system. The control acts only on the two state variables $x$ and $y$ and the link between reference signals and sensors is obtained though control gains. These control parameters are related to the amplitude of the sensory stimuli, so a regular periodic pattern emerges as a function of the sensor readings.[3,5]

The different dynamics shown by the controlled system are depicted in Fig. 2, where three different reference cycles are taken into account (i.e., $\mathrm{Re}\, f_i$). When the agent does not perceive any stimulus, the multiscroll system evolves chaotically [Fig. 2(a)]. When the control gains for the $\mathrm{Re}\, f_2$ are set to $K_{x_2}=K_{y_2}=0.1$, the chaotic evolution is confined in a part of the phase plane topologically related to the stimulus [Fig. 2(b)]. Furthermore if the control gain grows reaching a value $K_{x_2}=K_{y_2}=2$ (i.e., the agent is giving more importance to the sensors related to $\mathrm{Re}\, f_2$), the chaotic dynamics collapses to a cycle [Fig. 2(c)]. Finally, when another stimulus is perceived concurrently, for instance, the sensor associated with the $\mathrm{Re}\, f_3$ is very active ($K_{x_2}=K_{y_2}=2$ and $K_{x_3}=K_{y_3}=15$), the controlled system dynamics is influenced accordingly [Fig. 2(d)].

An important advantage given by the characteristic of the multiscroll system is the possibility to extend the grid of scrolls in a third dimension adding another simple piecewise linear function as described in Ref. 1. This solution can be useful when the three-dimensional distribution of sensors in the agent is important (e.g., in a multilegged robot or in a robotic arm). In this condition the control law can be easily extended to the state variable $z$.

In order to solve the robot navigation task, an action is performed by the robot according to the characteristics of the emerged pattern.

Each cycle that emerges from the control process (i.e., perceptual state) can be identified through its center position

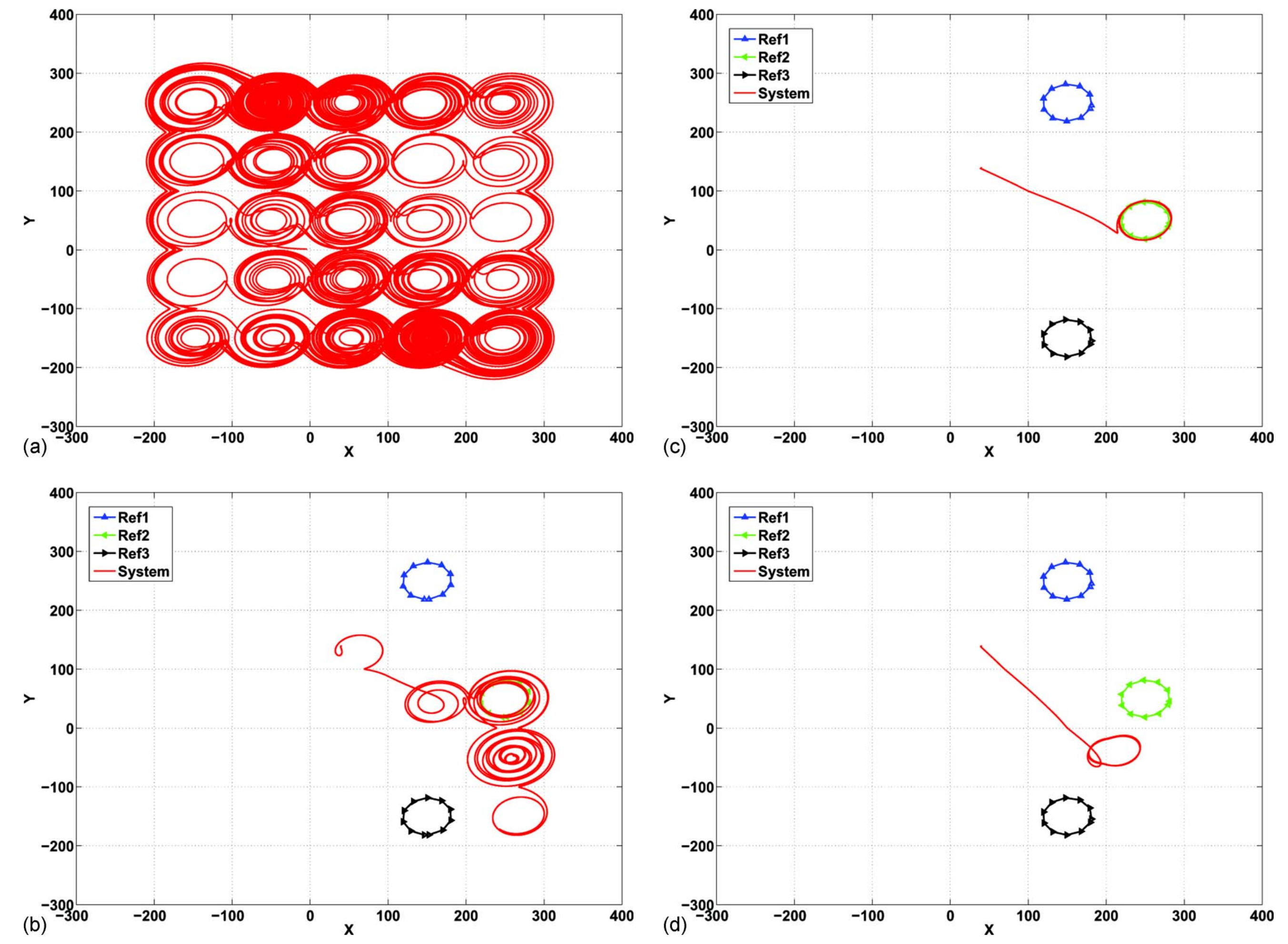


FIG. 2. (Color online) Dynamic evolution of the controlled multiscroll system for different values of the control gains. (a) $K_{x_2}=K_{y_2}=0$; (b) $K_{x_2}=K_{y_2}=0.1$; (c) $K_{x_2}=K_{y_2}=2$; (d) $K_{x_2}=K_{y_2}=2$ and $K_{x_3}=K_{y_3}=15$.

and shape. A code is then associated to each cycle and it is defined by the following parameters:

- $x_q$ and $y_q$: the center position in the phase plane $x-y$;
- $\bar{x}_q$: maximum variation of the state variable $x$ within the emerged cycle;
- $\bar{y}_q$: maximum variation of the state variable $y$ within the emerged cycle;

where $q$ indicates the emerged cycle.

The result of this stage is the emergence of a few parameters as compact information coming from sensors. Those linked to the following stage of action selection, will contribute to the realization of stimulus-dependent activity patterns. This approach has been suitably applied to robot navigation. Other parameters could be taken into consideration to improve the methodology for more complex tasks.

In the application dealing with autonomous navigation, only contact, distance, and target sensors have been used, although other sensors could be included. Distance sensors have a visibility range that describes the area where the robot is able to detect static and dynamic obstacles, while target sensors return the target detecting angle with respect to the frontal axis of the robot.

## B. Action selection layer

The parameters obtained through the WCC technique are processed by the action selection block (see Fig. 1). This block establishes the association between the parameters of the emerged cycle and the consequent robot action. For navigation purposes, an action consists of two elements:

$$\text{action} = (\text{module}, \text{phase}). \tag{3}$$

The module and phase of an action determine the motion step and the rotation angle to be performed by the robot, respectively. To perform this task, the motor map (MM) paradigm was employed. MMs are suitable to control robot behaviors in an unknown environment because they are adaptive, unsupervised structures and simple enough to allow a real time learning. The MM is composed of two layers: one ($V$) devoted to the storage of input weights and another ($U$) devoted to the output weights. This allows the map to perform tasks such as motor control. Formally, a MM can be defined as an array of neurons mapping the space of the input patterns into the space of the output actions:

$$\varphi : V \rightarrow U. \tag{4}$$

The learning algorithm is the key point to obtain a spatial arrangement of both the input and output weight values of the map. This is achieved by considering an extension of the Kohonen algorithm. At each learning step, when a pattern is given as input, the winner neuron is identified; this is the neuron which best matches the input pattern. Then, an update process of both the input and output weights for winner neuron and its neighbors is performed. The learning procedure is driven by a reward function that is defined on the basis of the final aim of the control process.[23] A MM although very efficient to be trained, could be difficult to be implemented in hardware because of the high number of afferent and efferent weights. Since this work is oriented towards a hardware implementation on real roving robot prototypes, a simplified version was adopted (for more details, see Ref. 19). The first difference is that the relationship of proximity among the neurons is not considered, therefore only the weights of the winner neuron are updated. Another and more important difference is that the input layer is substituted by a pattern table. Each emerged cycle, identified by a code, is stored in the pattern table (if it is not yet present) when the cycle emerges for the first time. Each element in the pattern table contains the emerged cycle code and the number of iterations from its last occurrence (defined as *age*). If the pattern table is full, the new element will overwrite the one containing the code of the pattern least recently used (LRU), i.e., that one with the highest *age* value. In the standard MM paradigm, the afferent layer $V$ represents the actual state of the system: in our application the pattern table performs the same function, by synthesizing information from the environment. Therefore the winning neuron of the input layer is replaced by the element $q$ of the pattern table which contains the last emerged cycle parameters. Moreover the output layer is now constituted by two weights for each element of the pattern vector. The element $q$ is connected to the weights $w_{qm}$ and $w_{qp}$ which represent, respectively, module and phase of the action associated with the pattern $q$ ($A_q$). At each step, the robot does not perform the exact action suggested by the weights of $q$ ($w_{qm}$ and $w_{qp}$), but the final action is

$$\begin{aligned} A_q &= [A_q(\text{module}), A_q(\text{phase})] \\ &= (w_{qm} + a_{s_q}\lambda_1, w_{qp} + a_{s_q}\lambda_2), \end{aligned} \tag{5}$$

where $\lambda_1$ and $\lambda_2$ are random variables uniformly distributed in the range $[-1;1]$. The parameter $a_{s_q}$ limits the searching area. Every time the pattern $q$ emerges, $a_{s_q}$ is reduced to focus the action search in a smaller range so to guarantee the convergence of the efferent weights. When there are no inputs, the perceptual core of the robot (the multiscroll system) behaves chaotically. This implies that there are no emerged cycles and no entries in the pattern table. In this case the robot explores the environment and its action depends on the position of the centroid of the chaotic dynamics shown by the system during the simulation step. Of course the exploration phase can be performed also using a forward motion, i.e., not considering the chaotic wandering.

The unsupervised learning mechanism that characterizes the MM algorithm, is based on a reward function (RF). This is a fitness function and it is the unique information that allows to determine the effectiveness of an action depending on the assigned task.

In a random foraging task, a suitable choice for the RF is

$$\mathrm{RF} = -\sum_i \frac{k_i}{D_i^2} - h_D D_T - h_A|\phi_T|, \tag{6}$$

where $D_i$ is the distance between the robot and the obstacle detected by the sensor $i$, $D_T$ is the target-robot distance, $\phi_T$ is the angle between the direction of the longitudinal axis of the robot and the direction connecting robot and target, and $k_i$, $h_D$, and $h_A$ are appropriate positive constants determined during the design phase.[24,25]

## III. APPLICATION TO ROBOT NAVIGATION

To demonstrate the applicability of the control scheme introduced above with a moving agent, an autonomous navigation task was taken into consideration.

The proposed framework has been firstly evaluated with a simulated roving robot and subsequently it was tested in a controller board based on a powerful Field Programmable Gate Array (FPGA) chip. To simplify the comparison of the proposed architecture with other more standard control techniques, a well-defined simulation setup has been followed. The simulation process consists of three main phases:

(1) *Learning*: during the learning phase the robot is placed into a training environment. While exploring, the system plastically changes the perception-action association on the basis of the overall goal (i.e., defined through the RF).
(2) *Test in the same environment as in the learning phase*: the learning mechanisms are stopped and the robot performances are evaluated.
(3) *Test in an unknown environment*: the robot is placed into a series of new environments, different from the learning ones, to show the generality of the learned behaviors.

During the learning phase a growing set of emerged cycles, arise in response to different environmental states, and are associated with suitable actions through the MM algorithm. In order to evaluate the robot performances in a quantitative way, the following parameters have been considered:

- $P_{\text{new}}$: cumulative number of new perceptual states that emerge during learning;
- *Bumps*: cumulative number of collisions with obstacles;
- *Explored area*: new areas covered (i.e., exploration capability);
- *Retrieved targets*: number of targets retrieved in the environment.

Initially each new pattern is associated with a random action, but the continuous emerging of such a pattern leads the action selection network to tune its weights in order to optimize the association between the perceptual state and the action to be performed. It is also desirable that new patterns occur only during the first learning steps (i.e., epochs). To guarantee the convergence of the algorithm, the learning pro-

TABLE I. Relevant simulation parameters of the MM-like structure.

| RF parameters | | Learning parameters | |
|---|---|---|---|
| $k_i$ for frontal distance sensors | 15 | $a_s$ start value | 0.6 |
| $k_i$ for lateral distance sensors | 10 | $a_s$ decrement factory | 0.01 |
| $k_i$ for contact sensors | 20 | $AS_{\text{Learn}}$ | 0.5 |
| $h_D, h_A$ | 10 | Tolerance | 8% |

cess cannot be considered ended while new patterns continue to emerge with a high frequency. Moreover, in order to solve the robot navigation problem, it is necessary that a pattern occurs several times, since the robot learns by trial and error.

Since the term $a_{s_q}$ gives information about the stability of the action associated with the pattern $q$, this was used to evaluate the convergence of the learning process. The LRU algorithm (that manages the pattern table) was modified to consider the $a_{s_q}$. The pattern $q$ cannot be replaced if its $a_{s_q}$ is under a fixed threshold ($AS_{\text{Learn}}$) that is determined during the design phase.

The code that identifies an emerged cycle is constituted by the four parameters $x_q$, $y_q$, $\bar{x}_q$, and $\bar{y}_q$ that take on continuous values because they depend on the evolution of the state variables of the controlled system.

The choice of the tolerance to distinguish among different patterns is a crucial problem during the design phase. If the tolerance increases, the number of patterns representing the robot perception of the environment decreases. Then, the learning time is reduced but the perception-action association is more rough. On the contrary, if the tolerance is reduced the number of actions increases, producing a wider range of different solutions for the navigation task. In this way it is feasible to reach a better but more time consuming solution.

Table I shows the value of the most relevant parameters used during the learning phase. In the RF, the contact sensors are dominant with respect to the distance and target ones since it was considered more important to preserve the robot safety. For the same reason the weights of the front distance sensors in the reward function are higher than lateral sensors.

## A. Simulation environment

To test the performance and the potential impact of the proposed architecture we developed a software tool for mobile robot simulations.[26] The framework has been designed to evaluate and compare the performance of different control strategies applied to the navigation of autonomous roving robots. The tool permits to create a 3D environment in which the exploring capabilities of a robot executing a navigation task such as a food retrieval task, can be evaluated. The behavior of the system is monitored with the help of a real-time visualization tool that gives a complete idea of the system evolution in terms of *internal state* and corresponding robot actions (see Fig. 3).

An example of the robot behavior guided by the internal patterns generated through the WCC technique is shown in Fig. 4. Movies of this simulation are available on the web.[27]

The modular structure of the simulator allows us to easily integrate new blocks as the action selection module. The

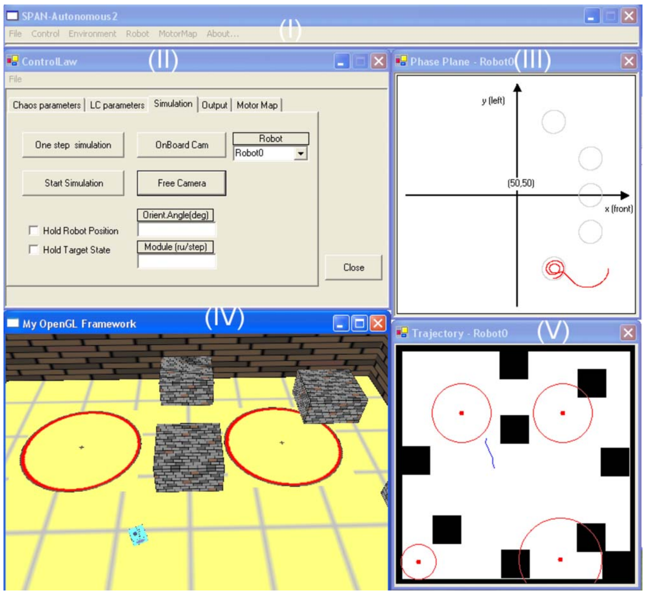


FIG. 3. (Color online) Interface of the 3D simulator used to compare robot navigation control algorithms: frame (I) is the menu that permits us to set the robot and the environment characteristics, frame (II) permits us to choose the simulation parameters, frame (III) represents the behavior of the WCC system and the active reference cycle, frames (IV) and (V) show a 3D and 2D view of the environment and the robot.

simulated environment, where the robot is placed, is made up of obstacles, walls, and targets. When the robot finds a target, this one is disabled until another target is found. This mechanism allows the robot to visit different targets exploring the whole environment. A target is represented by a point surrounded by a circle that indicates its visibility range. Obstacles are represented by walls and by rectangular objects distributed in the environment.

The simulated robot has a cubic shape and the dimension of each edge is 1 robot unit (ru). The sensory system equipped on the robot consists of four distance sensors, three contact sensors, and one target sensor. The detecting distance is set to 2.5 ru for the frontal distance sensors, 2.0 ru for the lateral sensors. The target sensor is able to detect the distance and the orientation between the robot and the detected target only when the robot reaches the influence area associated with each target.

Distance sensors are associated with the reference cycles that reflect the topological position. A similar strategy has been adopted for the target. When a target is within the range of robot visibility, it is considered as an obstacle located in a position symmetric with respect to the motion direction. The aiming action is guided by a reference cycle with a low gain, so that obstacle avoidance is a priority over reaching a target.

Figure 5 shows the link between sensors and reference signals. Sensors with the same position on the robot are associated with reference cycles with the same position in the phase plane.

The four parameters ($x_q$, $y_q$, $\bar{x}_q$, and $\bar{y}_q$) used to characterize the perceptual state (i.e., cycle) that emerges in the controlled chaotic system representing our perceptual system, are associated with a robot behavior through the MM-like structure that constitutes the action selection layer. The

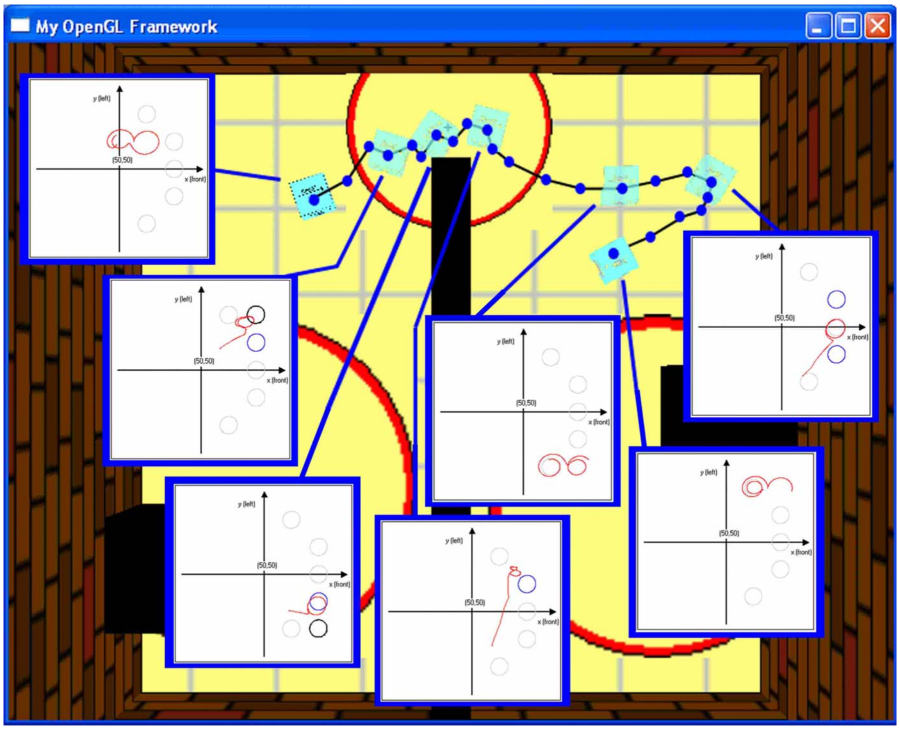


FIG. 4. (Color online) Example of a trajectory followed by the simulated robot and the corresponding mental patterns that emerge depending on the robot embodiment and situatedness.

action selection network consists of 100 entries and two output weights are associated with each input. Each entry is constituted by the set of the four parameters above, which describe an emerged cycle. So, when a cycle emerges in the perceptual system, on the basis of its identification parameters, is associated with an empty slot or to a previously emerged pattern.

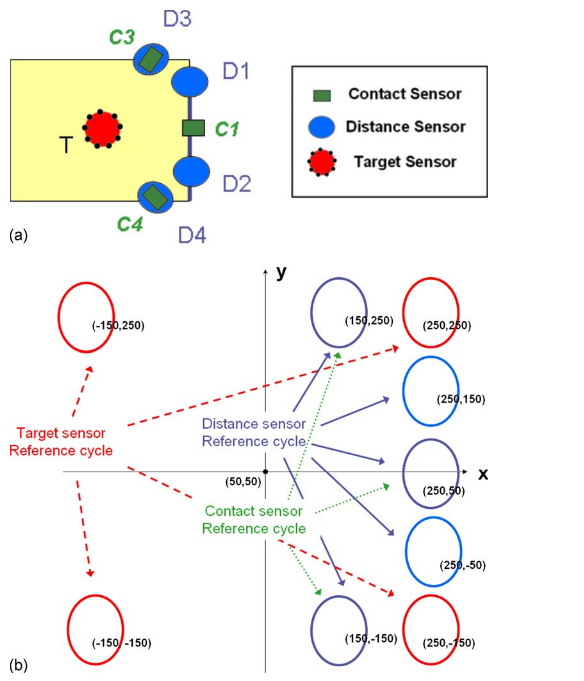


FIG. 5. (Color online) (a) Map of the sensors equipped on the robot. (b) Reference cycles linked to the robot sensors. Due to the common nature of contact and distance sensors, if they are placed in the same position on the robot (e.g., C3 and D3), a unique reference cycle is used. The target is considered as an obstacle located in a symmetrical position with respect to the motion direction. The target sensor is omnidirectional and for this reason it is associated to four reference cycles distributed in the corners of the scroll grid.

## IV. SIMULATION RESULTS AND COMPARISONS

In order to test the capabilities of the proposed architecture, the learning phase has been done in two environments with different characteristics. The first, called $E_1$, is shown in Fig. 6(a). It consists of two rooms, with a target in a room and two obstacles in the other one. In order to guarantee an easy passage through the rooms, another target is placed in the communication door between them. The second environment, called $E_2$, contains a series of obstacles, alternating with several targets [see Fig. 6(b)].

For each environment a set of five learning trials was performed with the MM structure randomly initialized. The learning phase is stopped at 65 000 actions (i.e., epochs). Each robot simulation step (i.e., epoch) corresponds to a single robot action: this is determined simulating the dynamical system for 2000 steps with an integration step equal to 0.1. These parameters guarantee the convergence of the multiscroll system to a stable attractor when external stimuli are perceived by the robot.

During the learning phase, a sequence of new patterns emerges and the robot learns how to behave in the current situation. To evaluate the convergence of the learning phase, in Fig. 7 the trends of the cumulative number of new patterns that arise ($P_{\text{new}}$) is shown for both environments. The learning process leads to a huge improvement of the robot behavior for the situation (i.e., perceptual state) that more often occurs, while some other patterns cannot be suitably learned if they seldom emerge.

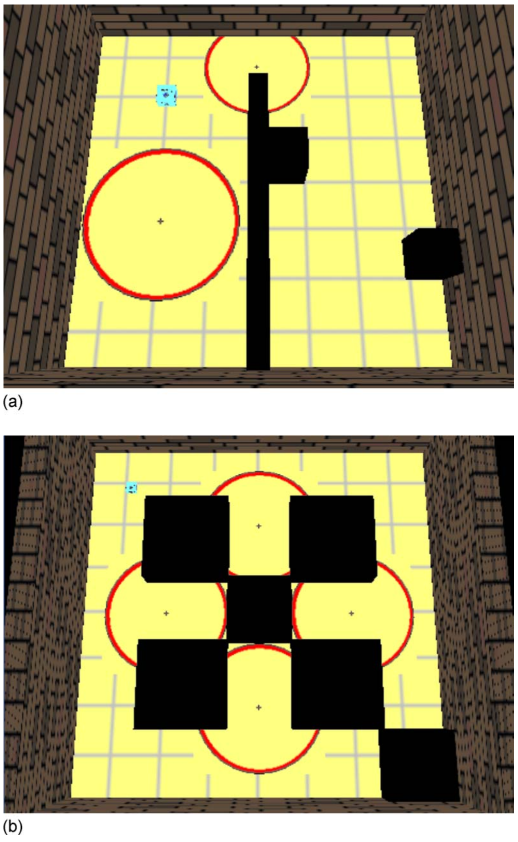


FIG. 6. (Color online) 3D view of the environments used during the learning phase: $E_1$ (a) and $E_2$ (b). Black boxes are obstacles, the circles represent the target detection area. The dimensions of $E_1$ are $20\times 20$ ru$^2$, whereas $E_2$ dimensions are $36\times 36$ ru$^2$.

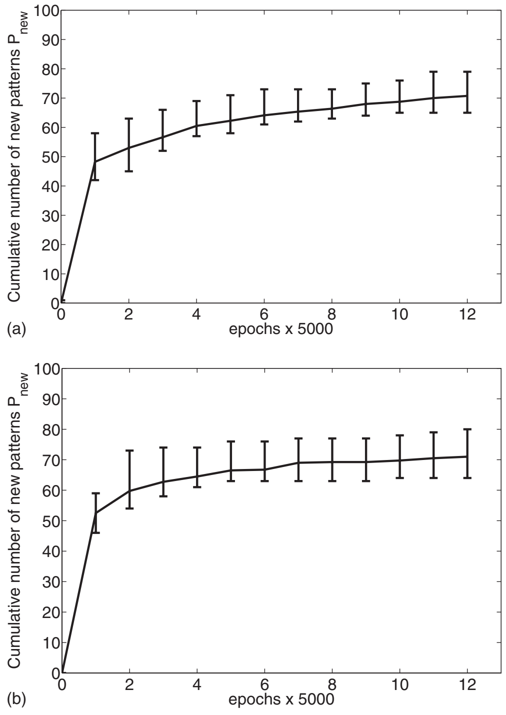


FIG. 7. Cumulative number of new patterns that emerge during the learning phase calculated in windows of 5000 epochs for the environments $E_1$ (a) and $E_2$ (b). The bars indicate the minimum and maximum value, whereas the solid line is the mean value of the set of five simulations performed.

After the first 5000 learning epochs, the number of new emerging patterns is very low and tends to reach a stable value. These results were obtained by adopting the learning parameters defined in Table I. In particular, the total number of emerging patterns, around 70, is directly related to the tolerance parameter that has been set to 8%. In other simulations carried out, increasing the tolerance factor to 15%, the number of emerged patterns was reduced to around 30.

A reduced number of emerged patterns leads to speed up the learning phase, but decreases the specialization of the robot actions with a consequent lowering of the performances.

To evaluate the efficacy of the learning phase, an important parameter is $a_s$. The value of $a_s$ is directly related to the stability of the association among the perceptual state and the robot action.

Figure 8 shows the evolution of the learning parameter $a_s$ in the two environments (i.e., $E_1$ and $E_2$). In Figs. 8(a) and 8(b) the trend for a single simulation and for each pattern is shown. The duration of the learning phase, set to 65 000 epochs, guarantees a sufficient stability and reliability for the perception-action association. In fact the total number of patterns emerged is about 70 and more than 60% of the patterns have an $a_s < 0.5$: this corresponds to more than 100 updates of the associated action following the indication of the reward function, and about the 25% of the patterns have an $a_s < 0.1$ that correspond to more than 500 updates.

In Figs. 8(c) and 8(d) the average value for the whole simulation campaign for each environment is given.

The learning process guided by the reward function significantly improves the robot capabilities evaluated in terms of number of bumps and target retrieved. In Figs. 9 and 10, the cumulative number of bumps and targets found is shown for the two learning environments, comparing the behavior of the system during learning with the same architecture when the learning is not activated. The results show that since the first stage of the learning (i.e., about 5000 epochs), a significant difference in term of performance is evident. The robot behavior is completely different as shown in Fig. 11, where the trajectories followed by the robot with and without learning can be compared.

This comparison outlines the capability of the control system to create a suitable link among perception and action.

To further outline the results of the testing phase for a learned architecture, information on the association between perceptual patterns and corresponding actions is reported. In particular, in Fig. 12, the final emerged actions associated with the mostly used patterns are shown. This result is related to a learning carried out in the environment $E_1$. Figure 12(a) shows the $x$-$y$ phase plane of the multiscroll system, together with the internal patterns emerged during a learning phase. For sake of clarity, each class is reported only with a marker indicating its position (i.e., parameters $x_q$ and $y_q$). The vector associated with each pattern, shows module and phase of the corresponding action performed by the robot, with respect to the $x$-axis that indicates the front direction of the robot motion. In Fig. 12(b), a typical target approaching maneuver made by the robot is shown. The sequence of in-

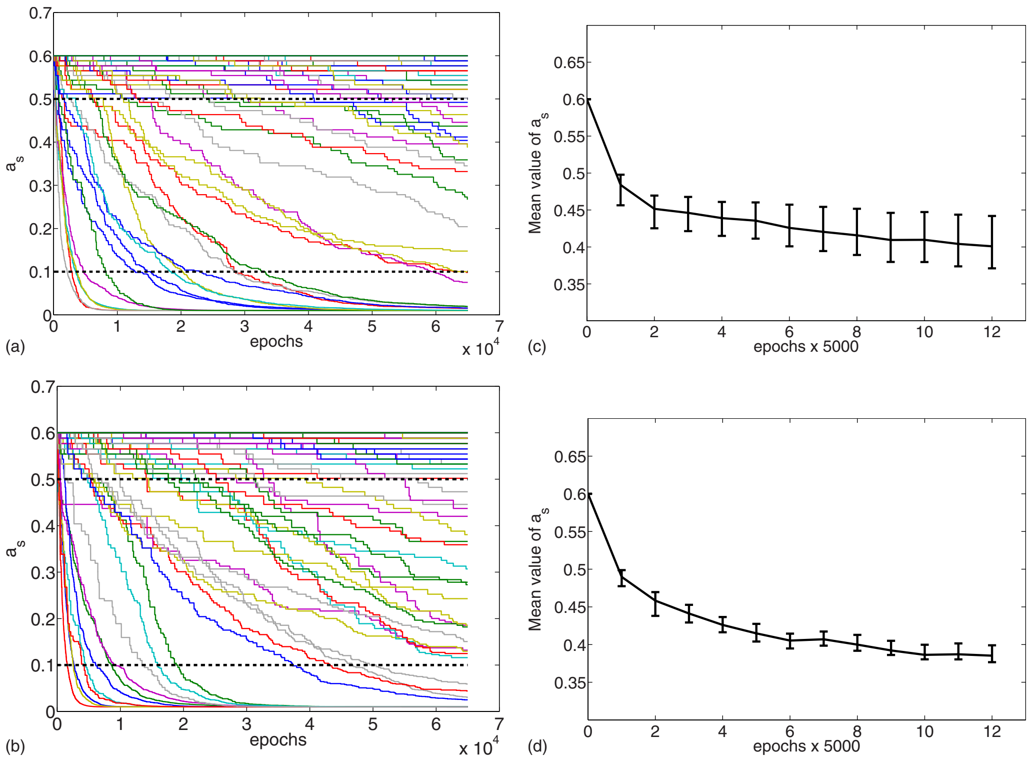


FIG. 8. (Color online) An example of the evolution of all the $a_s$ during the learning phase in the environments $E_1$ (a) and $E_2$ (b). In (c) and (d) the average value of $a_s$ among five simulations, calculated in windows of 5000 epochs for the two environments is shown. The bar indicates the minimum and maximum value for each window.

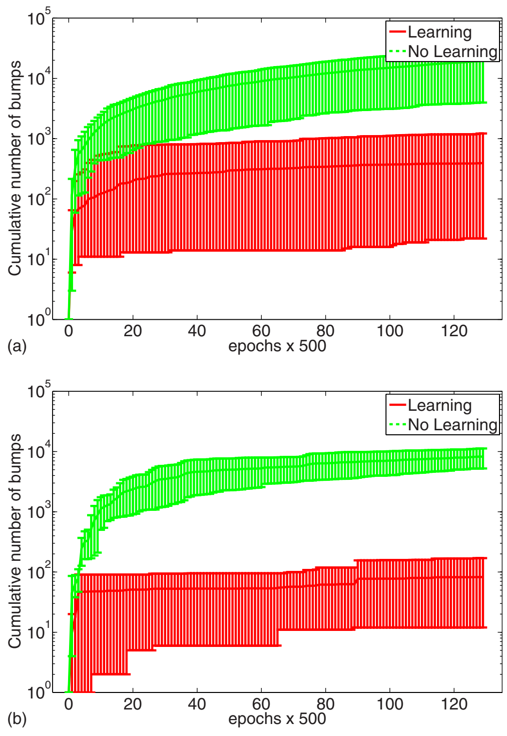


FIG. 9. (Color online) Cumulative value of bumps calculated in windows of 500 epochs in the conditions of learning and no-learning for the environments $E_1$ (a) and $E_2$ (b). The two trends span among the minimum and maximum value for each window.

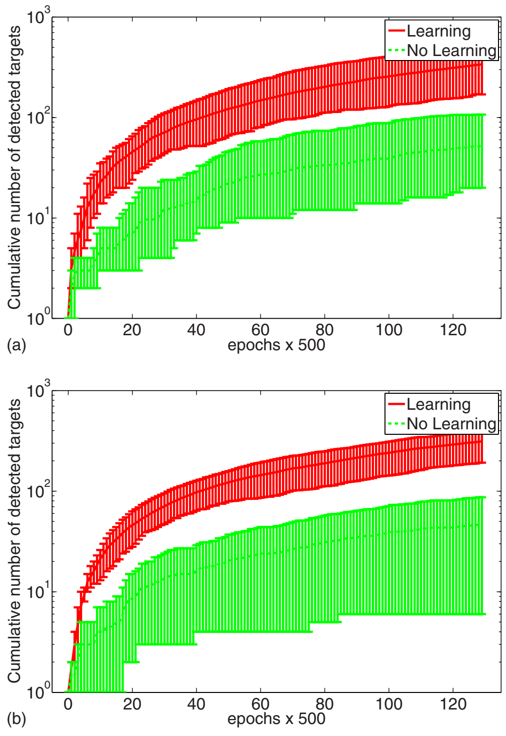


FIG. 10. (Color online) Cumulative value of retrieved targets calculated in windows of 500 epochs in the conditions of learning and no-learning for the environments $E_1$ (a) and $E_2$ (b). The two trends span among the minimum and maximum value for each window.

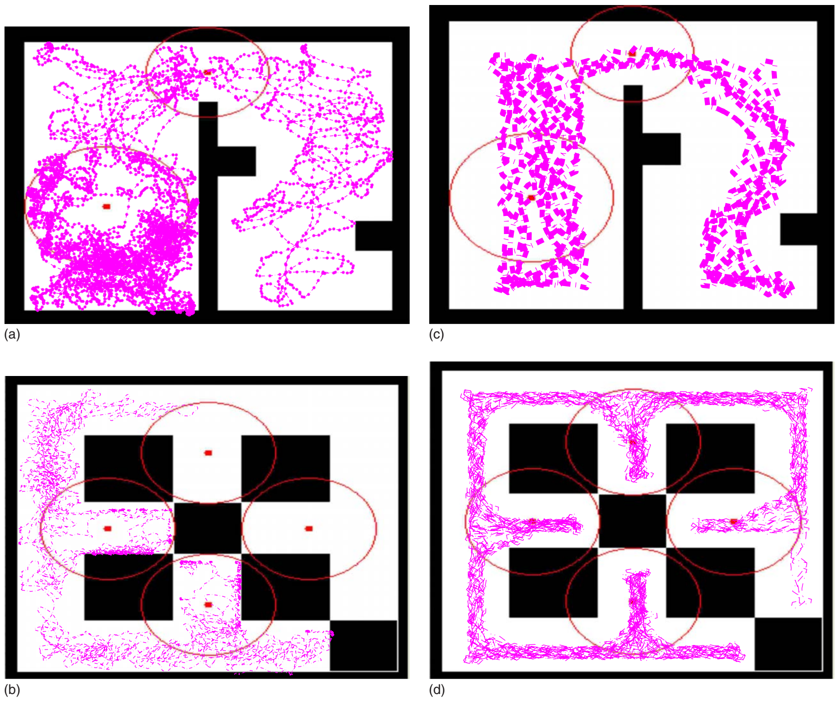
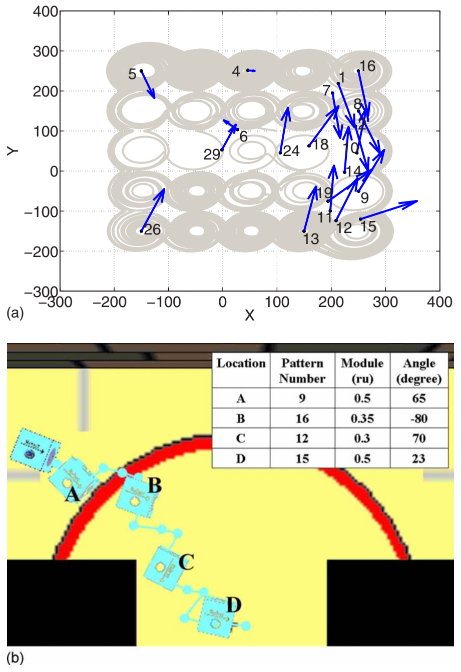


FIG. 11. (Color online) Trajectories followed by the robot during a test in the learning environments. (a) and (b) Behavior of the robot without learning; (c) and (d) trajectories followed after the learning phase. The simulation time is 10 000 epochs.

ternal patterns and the corresponding actions, used to avoid collisions with an obstacle and to find a target, are depicted.

A main issue in neural network learning is to ensure that the network, during the learning phase, is able to extract a "motion rule" from the environment patterns. Since the environment conditions reflect situations where the robot locally interacts with targets and/or obstacles, the capability to learn a motion rule can be generalized to new environments.



| Location | Pattern Number | Module (ru) | Angle (degree) |
|---|---|---|---|
| A | 9 | 0.5 | 65 |
| B | 16 | 0.35 | -80 |
| C | 12 | 0.3 | 70 |
| D | 15 | 0.5 | 23 |

FIG. 12. (Color online) (a) Creation of an *action map*. Each vector indicates the phase action associated with the emerged patterns indicated only with their position in the phase plane. (b) Sequence of perceptual patterns and associated actions generated by the control architecture.

So we will refer to this "generalization capability," shown in the following section.

## A. Comparisons with other navigation strategies

To further evaluate the performances of the control architecture, we tested the learned structure in new environments making a comparison with other navigation strategies. In particular, a basic version of the traditional method of the quadratic potential field (PF) has been considered. Also in this case the robot can use only local information, acquired from its sensory system to react to the environment conditions (i.e., local potential field).[28,29] For the comparison, two other control schemes based on the WCC technique have been taken into account. For these algorithms we adopt a different action system based on a deterministic action selection procedure.[26,5] The difference between the two versions is limited to the behavior of the robot during the exploration phase (i.e., when no stimuli are perceived). The former implements a very simple behavior that consists of a forward movement with the speed set to its maximum value (i.e., $\mathrm{WCC}_f$), whereas the latter considers the chaotic evolution of the multiscroll system to determine the action of the robot exploring the environment (i.e., $\mathrm{WCC}_c$).

The parameters of all the algorithms taken into consideration (e.g., robot speed, constraints for the movements) have been chosen in order to allow a comparison among them.

To compare the results obtained with the reward-driven learning in the MM structure with the other control schemes, a learned structure with average performances was chosen. To compare the navigation capabilities to explore the environment avoiding obstacles and retrieving the target found, two different scenarios were considered. The former (i.e., $E_3$) is a well structured environment consisting of four rooms

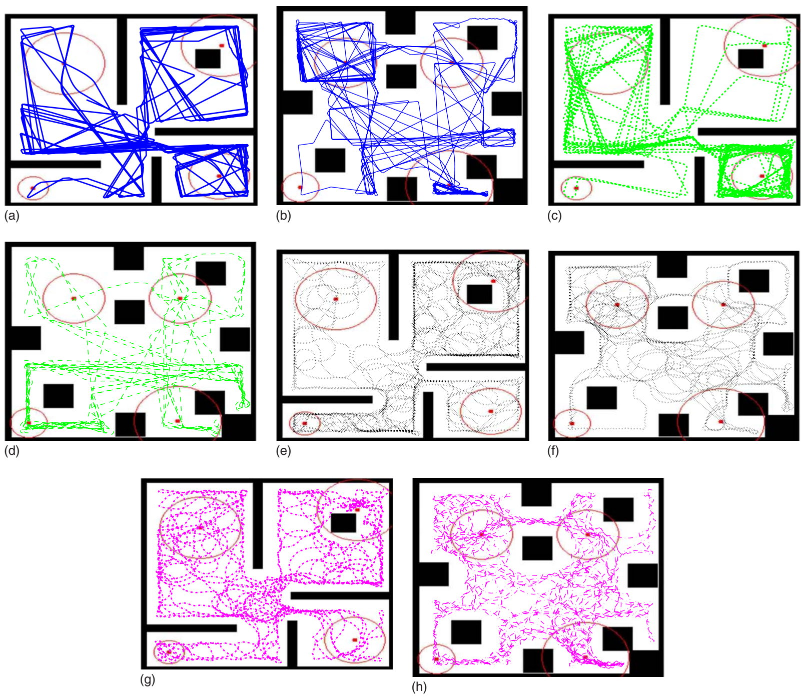


FIG. 13. (Color online) Trajectories followed by the robot controlled through: (a) (b) local potential field; (c) (d) WCC with forward exploration strategy; (e) (f) WCC with chaotic exploration behavior; (g) (h) WCC with action selection layer based on a MM structure.

with a target placed in each room; the latter (i.e., $E_4$) is a more complex environment filled with randomly placed obstacles and targets. The dimensions of both the environments are $40 \times 40$ ru$^2$.

A typical trajectory followed by the robot in each case is shown in Fig. 13. The figure qualitatively shows the robot behavior for the four considered algorithms. For each simulation, the robot is randomly placed in the environment, and the three control methods are applied monitoring the robot behavior for 10 000 actions.

In the case of the proposed architecture with the MM-based action selection layer and the chaotic exploration movements (WCC$_{\mathrm{MMc}}$), the parameters used for both the environments $E_3$ and $E_4$, have been obtained with a learning phase carried out in the environment $E_1$. This was done to demonstrate that the knowledge acquired by the system during learning can be used also in different environmental situations. Movies of these simulations are available on the web.[27]

To compare the performances of the algorithms we consider, as performance indexes, the cumulative number of targets found and the area explored by the robot.[16]

The performance obtained with the WCC$_{\mathrm{MMc}}$ architecture in terms of explored area, are comparable with the best results obtained with the other navigation schemes, as shown in Fig. 14.

It is important to notice that comparing the results between WCC$_f$ and WCC$_c$, in terms of explored area, the effect of chaos during the exploration phase is positive and allows to obtain the best performance.

Furthermore as far as the cumulative number of detected targets is concerned, the results are excellent for the environment $E_4$ [see Fig. 15(b)]. Instead the simulations performed in $E_3$ indicate a retrieving problem as illustrated in Fig.

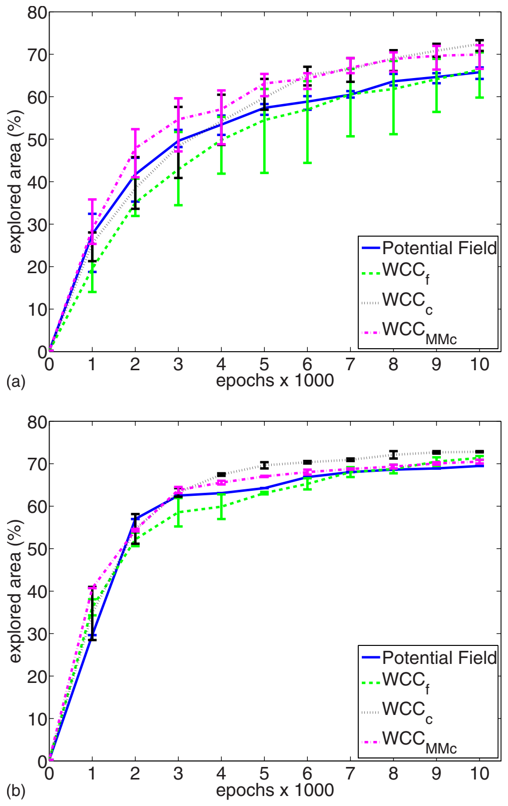
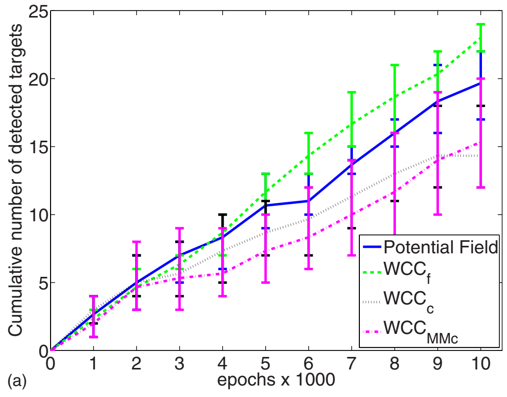


FIG. 14. (Color online) Value of the explored area obtained using three different control algorithms in two environments: a four-room environment ($E_3$) (a) and a less structured one ($E_4$) (b). The arena which dimension is $50 \times 50$ robot units, has been divided into locations of $2 \times 2$ robot units. The simulation time is 10 000 epochs and the mean value of the area explored, mediated over five simulations, calculated with time windows of 1000 epochs, is indicated. The bars show the minimum and maximum explored area.



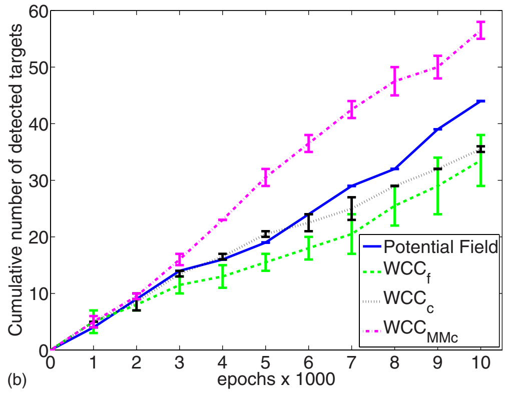


FIG. 15. (Color online) Cumulative number of targets found with the considered control algorithms in $E_3$ (a) and in $E_4$ (b). The simulation time is 10 000 epochs and the mean number of targets, mediated over five simulations, calculated with time windows of 1000 epochs, is indicated. The bars show the minimum and maximum number of targets.

15(a), in which the performances of the WCC$_{\mathrm{MMc}}$ are not impressive. A justification can be found in the particular distribution of the targets in $E_3$ that are confined in four well distinct rooms. In this condition an approach like the PF or the WCC$_f$ is more suitable because they adopt forward actions during the exploration phase. Nevertheless the performances of the WCC$_{\mathrm{MMc}}$ can be improved if a learning session is performed in the testing environment. In fact, as shown in Fig. 16, the WCC$_{\mathrm{MMc}}$ agent, after learning on the

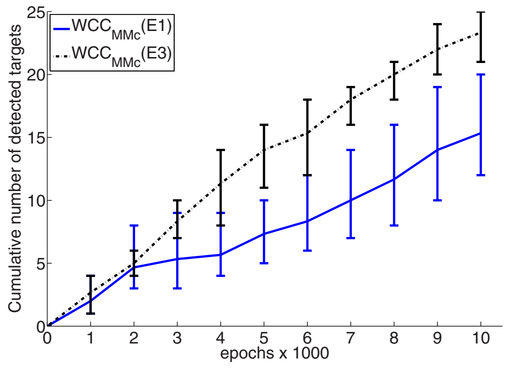


FIG. 16. (Color online) Comparison between the cumulative number of targets found in the environment $E_3$ by an agent that performed a learning phase in $E_1$ and another that creates its perception-action map directly in $E_3$. The simulation time is 10 000 epochs and the mean number of targets, mediated over five simulations, calculated with time windows of 1000 epochs, is indicated. The bars show the minimum and maximum number of targets found.

environment $E_3$, improves its target retrieving capabilities with respect to the previous one (i.e., learning in $E_1$). This final result shows that the proposed architecture can generalize the learned motion rules and the performance can be improved extending the learning to the specific environment we are dealing with.

## V. FPGA-BASED IMPLEMENTATION

The choices done during the design of the control architecture were oriented to the implementation of the strategy in a high performing hardware, embedded on a roving robot. In the following some preliminary results on an experimental platform are reported.

To realize the WCC approach for navigation control in hardware, a FPGA-based board was considered. The control algorithm was implemented on the Nios II, a 32-bit RISC digital soft-processor, using also customized VHDL (Very High Speed Integrated Circuits Hardware Description Language) blocks for the most time consuming tasks.[30]

The roving robot *Rover II* used for the experiments is a classic four wheeled drive rover controlled through a differential drive system. Rover II is equipped with four infrared distance sensors, with low level target sensors (i.e., able to detect black spots on the ground used as targets) and with a board for sonorous target detection inspired by phototaxis in crickets. A scheme of the hardware framework is shown in Fig. 17.

In the following experiments, to fulfill a food retrieval task, distance sensors were used for obstacle avoidance, whereas a cricket-inspired hearing board was considered for the target detection issues.

The simulation and control of the multiscroll system is performed directly in a VHDL entity, implementing a fourth order Runge–Kutta algorithm (RK4). On the other side, the NiosII microprocessor is devoted to handle the sensory system, to execute the action selection layer and to supervise the activities of the VHDL entities implementing the weak chaos control. When the simulation ends, the NiosII reads the parameters that identify the emerged cycle. Then it calculates the command to drive the roving robot. The simulation process implemented in the VHDL entity lasts about 4.2 ms (the time is referred to the generation of 2000 samples with an integration step of 0.1) and the control algorithm running on NiosII about 80 ms.

The environment used to perform tests and comparisons, is a $10\times 10$ ru$^2$ room with three obstacles and two targets. The simulated arena and the real arena are shown in Fig. 18, where the trajectories followed by the robot after the learning process are shown and compared with the trajectories shown by a simulated robot working in a virtual environment mimicking the real one.

## VI. REMARKS

Among the very few attempts to use dynamical systems in robot perception, the dual dynamics (DD) scheme is of interest.[31] The key idea is that a robotic agent can work in different modes, which lead to qualitatively different behavioral patterns. Mathematically, transitions between modes

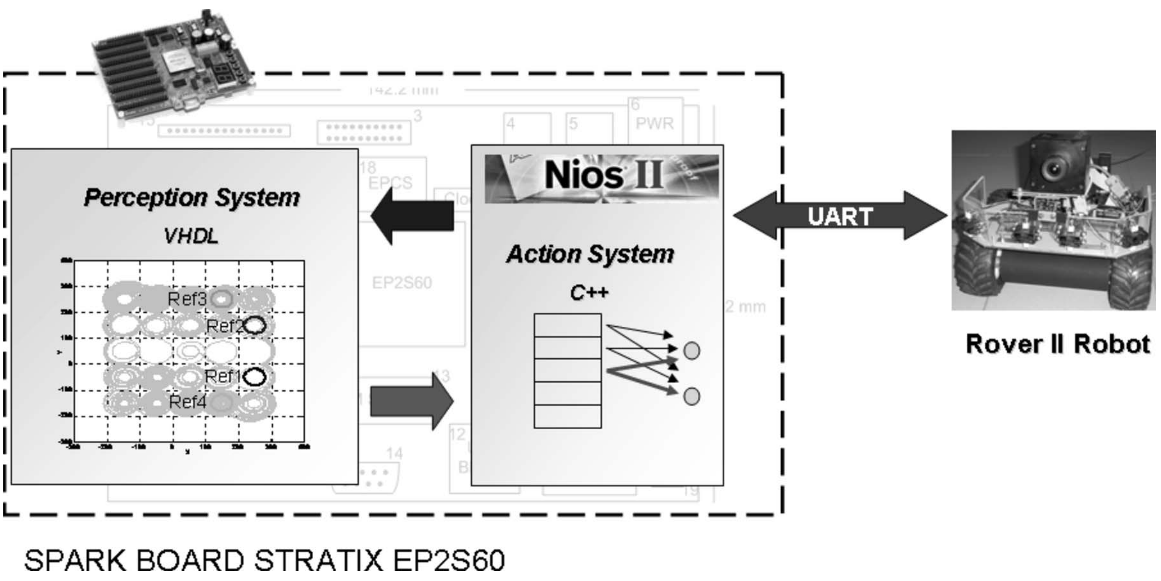


FIG. 17. Description of the framework used during the experiments. A FPGA-based board equipped on a roving robot.

can be considered as bifurcations in the control system. One of the key aspects of the DD scheme is that the overall behavior of the agent is the result of a concurrent and weighted activation of different behaviors, each one modeled by a dynamical system. All the behaviors are active at the same time. In this way the approach tries to formulate in an alternative way the very well known behavior based robotic methodology.

Our approach, as outlined above, is completely different. In fact only one behavior, at the end of the learning phase, is active. Moreover, we fully exploit the richness shown by chaotic wanderings, by controlling the chaotic system in a "weak mode." In fact, in this way, even slightly different sensory information can lead to a different emerged cycle. The "meaning" of this cycle is gained through a very simple associative layer, reward driven, that can be implemented in hardware at low cost and with real time performance.

An application of chaos theory to control robotic systems is included in Kuniyoshi's works, in which the coupled map lattice (CML), exploited for their rich dynamical properties,[32] are used in experiments with a baby model.[18] Coupled chaotic system such as CML, can change their behavior to reflect external information via coupling with the environment, and can explore the multiple dynamics embedded, and get temporarily entrapped in them.

In this work, we adopt an extremely simple dynamical system, that can be designed and developed in a modular way to facilitate the scalability of the architecture. Moreover, the chosen model can be implemented in both in analog[2] and in digital[5] hardware to be embedded in a really working robot prototype.

The learning algorithm used to attain a suitable association between the emerged cycle characteristics and a successful action is drawn from the reinforcement learning theory,[33] by implementing the simplest version that leads to good results being also suitable for a reliable hardware implementation, due to its simplicity.

Another important consideration is that the WCC technique discussed here is based on a state feedback approach and the control gains are chosen here in order to grant the controlled system stability under stimulation with reference signals (see Ref. 5 for details). The term *weak chaos control* refers to a strategy that does not aim to the exact matching between the reference and the controlled signal; instead the chaotic signal has to collapse to an orbit near the reference signal. At this point, the amplitude value of the control gains is related to the matching degree between the reference and the controlled signal. The control gain value could be therefore used as an additional choice to weight a kind of "degree of attention" that the learning system could pay to the corresponding sensor signal. If learning is used also to choose the control gains, at the end of the learning phase the robot could be allowed to discard useless sensor signals and to "pay attention" to the important ones. This approach is currently under active investigation.

Another advantage, given by the chaotic nature of the perceptual core, is apparent when the robot is trapped in a deadlock situation. In this situation, standard solutions proposed in the literature consider either the introduction of a higher layer dedicated to deal with the local minima or to add noise to escape from the deadlock. In our case the robot entrapped in a difficult situation, without succeeding in reaching the assigned target for a long time can decrease its level of attention to sensory inputs (e.g., reducing the assigned control gain, this effect is similar to "boredom" in living beings) allowing the chaotic dynamics to reemerge. The result is to increase the exploration movements to avoid the local minima.

Moreover, the strategy introduced in the paper was applied to the apparently simple task of autonomous navigation

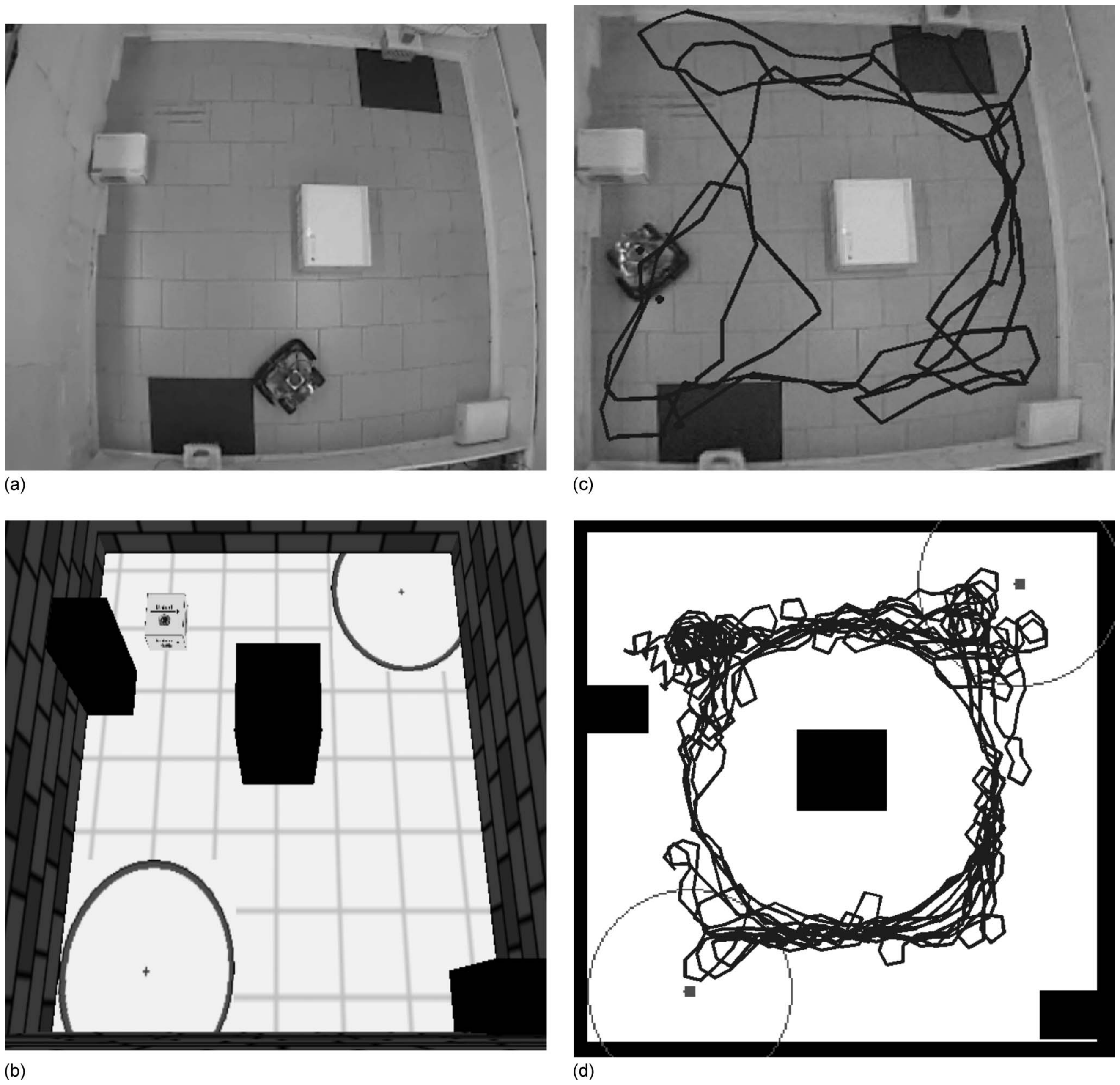


FIG. 18. Environment used during the learning phase in real experiments (a) compared with simulations (b). The dimensions of the environment are $10 \times 10$ ru$^2$. Trajectories followed during a test in the learning environments by the real robot (c) and the simulated one (d).

learning. The clear advantages over the classical approaches, for example, related to the potential field, are that the control structure, based on the multiscroll system, is quite general. The fact that the results obtained are comparable with those of the potential field is relevant. This means that a general approach to learn the sensing-perception-action cycle using the power of information embedding typical of chaotic systems, applied to a traditional task, succeeds in reaching the same results as a technique peculiarly designed to solve that task. The WCC-MM approach can in fact be applied to learn an arbitrary *action-map*, or in general a *behavior map*. In particular, we exploit the rich information embedding capability of a chaotic system with a simple learning that gives a "meaning" to the embedded information (taking inspiration from Ref. 9), within the context of the robot action. This contextualization is decided through the reward function. The definition of this function has to be designed *a priori*, based on the task to be accomplished. According to the best of the authors' knowledge, it is the first time that the chaotic circuits and system theory, linked to a simple neural learning, is used to approach problems relevant in robot perception, even in the simple case of navigation. In fact, here navigation is treated as a perceptual task. Several other examples are currently under investigation to further generalize the approach and a considerable theoretical effort is being paid nowadays to include the strategy introduced here within a more general scheme for robot perception in unknown conditions and environments.

## VII. CONCLUSION

In this paper a new perception-action system has been proposed to deal with robot navigation problems in unknown environments. The weak chaos control technique permits us to synthesize the perception schema in a compact form easy to be processed. The introduced action layer allows us to generate new behaviors through a very simple unsupervised learning driven by a reward function. The architecture has been tested in different simulated environments showing the increase in terms of robot capabilities obtained during the learning phase.

The simulation results confirmed that the proposed solution is suitable to resolve the obstacle avoidance and target retrieving tasks. The performance indexes adopted have been compared with other navigation strategies showing that a learned structure can be used in different kinds of environments and that reactivating the learning mechanisms improves the performance. Finally the designed and assessed architecture is suitable for a hardware implementation based on FPGA. The perceptual core and the learning mechanism were implemented in hardware and embedded on a roving robot for a real-time learning of the navigation tasks defined through the reward function.

Further developments will include the introduction in the real robot of other kinds of sensors needed to improve the capabilities to explore and draw more details from the environment for perceptual purposes.

## ACKNOWLEDGMENTS

The authors acknowledge the support of the European Commission under the project SPARK II "*Spatial-temporal patterns for action-oriented perception in roving robots: An insect brain computational model.*"

[1]J. Lu, G. Chen, X. Yu, and H. Leung, IEEE Trans. Circuits Syst., I: Regul. Pap. **51**, 2476 (2004).

[2]M. Hulub, M. Frasca, L. Fortuna, and P. Arena, Chaos **16**, 013121 (2006).

[3]P. Arena, L. Fortuna, M. Frasca, G. Lo Turco, L. Patané, and R. Russo, "Perception-based navigation through weak chaos control," in Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC 2005), Seville (2005).

[4]P. Arena, S. De Fiore, L. Fortuna, M. Frasca, L. Patané, and G. Vagliasindi, "Weak chaos control for action-oriented perception: Real time implementation via FPGA," in Proceedings of the Biomedical Robotics and Biomechatronics, February 2006, BioRob 2006, pp. 555–560.

[5]P. Arena, S. De Fiore, L. Fortuna, M. Frasca, L. Patané, and G. Vagliasindi, Special Issue on Bio-Inspired Sensory-Motor Coordination, Auton. Rob. **25**, 123 (2008).

[6]R. C. Arkin, *Behaviour Based Robotics* (MIT Press, Cambridge, 1997).

[7]*The Artificial Life Route to Artificial Intelligence: Building Embodied Situated Agents*, edited by L. Steels and R. A. Brooks (Lawrence Erlbaum Associates, Hillsdale, NJ, 1995).

[8]W. J. Freeman and C. A. Skarda, "Representations: Who needs them?" in *Brain Organization and Memory Cells*, edited by J. L. McGaugh, N. Weinberger, and G. Lynch (Oxford University Press, New York, 1990), pp. 375–380.

[9]W. J. Freeman, Int. J. Bifurcation Chaos Appl. Sci. Eng. **14**, 515 (2004).

[10]W. J. Freeman, Int. J. Bifurcation Chaos Appl. Sci. Eng. **10**, 2307 (2000).

[11]C. A. Skarda and W. J. Freeman, Behav. Brain Sci. **10**, 161 (1987).

[12]I. Aradi, G. Barna, and P. Erdi, Int. J. Intell. Syst. **1091**, 89 (1995).

[13]P. Arena, L. Fortuna, M. Frasca, and L. Patané, Int. J. Neural Syst. **13**, 469 (2003).

[14]W. J. Freeman and R. Kozma, Behav. Brain Sci. **23**, 401 (2000).

[15]R. Gutierrez-Osuna and A. Gutierrez-Galvez, IEEE Trans. Neural Netw. **14**, 1565 (2003).

[16]D. Harter and R. Kozma, IEEE Trans. Neural Netw. **16**, 565 (2005).

[17]M. E. Yalcin, J. A. K. Suykens, J. Vandewalle, and S. Ozoguz, Int. J. Bifurcation Chaos Appl. Sci. Eng. **12**, 23 (2002).

[18]Y. Kuniyoshi and S. Sangawa, Biol. Cybern. **95**, 589 (2006).

[19]P. Arena, P. Crucitti, L. Fortuna, M. Frasca, D. Lombardo, and L. Patané, Int. J. Bifurcation Chaos Appl. Sci. Eng. **17**, 107 (2007).

[20]T. Kohonen, Prog. Biocybern **43**, 59 (1972).

[21]J. D. Murray, *Mathematical Biology* (Springer, Berlin, 2002).

[22]P. Arena, L. Fortuna, F. Frasca, R. Pasqualino, and L. Patané, "CNNs and motor maps for bio-inspired collision avoidance in roving robots," in 8th IEEE International Workshop on Cellular Neural Networks and Their Applications (CNNA 2004), Budapest, 2004.

[23]H. Ritter, T. Martinetz, and K. Schulten, *Neural Computing and Self-Organizing Maps Reading* (Addison-Wesley, Reading, 1992).

[24]P. Arena, L. Fortuna, M. Frasca, D. Lombardo, and L. Patané, "Learning efference in CNNs for perception-based navigation control," in Proceedings of the International Symposium on Nonlinear Theory and its Applications, NOLTA'05, Bruges, Belgium, 18–21 October 2005.

[25]P. Arena, L. Fortuna, M. Frasca, L. Patané, and M. Pavone, "Towards autonomous adaptive behavior in a bio-inspired CNN-controlled robot," in Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS 2006), Island of Kos, Grecia, 21–24 April 2006.

[26]P. Arena, L. Fortuna, M. Frasca, G. Lo Turco, L. Patané, and R. Russo, "A new simulation tool for action-oriented perception systems," in Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation, (ETFA 2005), Catania, Italy, 19–22 September 2005.

[27]Spark home page (online) available at http://www.spark.diees.unict.it/WCC2.html.

[28]O. Khatib, Int. J. Robot. Res. **5**, 90 (1986).

[29]R. Beard and T. McClain, "Motion planning using potential fields," Electrical and Computer Engineering, Brigham Young University, Provo, Utah, 2003; http://www.ee.byu.edu/ugrad/srprojects/robotsoccer/labs/skills/potential.pdf.

[30]P. Arena and L. Patané, *Spatial Temporal Patterns for Action Oriented Perception in Roving Robots*, Cognitive Systems Monographs (Springer, Berlin) (to be published).

[31]H. Jaeger and Th. Christaller, "Dual dynamics: Designing behavior systems for autonomous robots," in Proceedings of the International Symposium on Artificial Life and Robotics (AROB'97), edited by S. Fujimura and M. Sugisaka (1997), pp. 76–79.

[32]K. Kaneko and I. Tsuda, *Complex Systems: Chaos and Beyond* (Springer, Berlin, 2001).

[33]R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction, a Bradford Book* (MIT Press, Cambridge, 1998).