

Perception-Action map learning in controlled Multiscroll systems applied to robot navigation

Paolo Arena*, Sebastiano De Fiore*, Luigi Fortuna*,

Luca Patané*,

July 29, 2008

*Dipartimento di Ingegneria Elettrica Elettronica e dei Sistemi, Università degli Studi di Catania, viale A. Doria 6, 95125 Catania, Italy. E-mail: [parena, lpatane, sdefiore]@diees.unict.it, Tel: +39-095-7382307, Fax: +39-095-330793.

Abstract

In this paper a new technique based on the *sensing-perception-action loop* 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 depends on sensorial stimuli. In this way, it can be possible to encode an abstract and concise representation of the environment into the chaotic dynamics. This representation has to be suitably linked to an action, executed by the robot, to fulfill an assigned mission. This task is addressed here: the perception-action 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, exploiting the richness of environmental information 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 FPGA-based implementation of the proposed architecture is briefly outlined and preliminary experimental results on a roving robot are also reported.

Keywords Multi-scroll chaotic system, Sensing-perception-action loop, reflexive behaviours, Motor Maps, robot navigation.

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.

1 Introduction

After the introduction of Multiscroll chaotic systems [1], several circuit solutions were proposed, among which [2], for a real time implementation. In particular, the potential role of multiscroll dynamics in perception was envisaged in [2]. In fact, in more recent works [3, 4, 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 stand-alone 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 the perceptual system has been realized by means of chaotic dynamical systems controlled through a new technique called *Weak Chaos Control (WCC)*.

WCC technique allows to create perceptual states that can be managed by the control system because directly related to the concept of embodiment and situatedness [7]. The biological principles that inspired the proposed approach are based on Freeman's theories. His approach recognizes the existence of internal (mental) representation [8, 12]. 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 constitutes the mental representation of the stimulus. Freeman and co-workers, in their extensive experimental studies on the dynamics of sensory processing in animals [9, 10, 11, 12], conceive a dynamical theory of 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. So he came to the conclusion that an internal mental representation (cerebral pattern) of a stimulus 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 [12].

The application of chaotic models as basic blocks to reproduce adaptive behaviours [13, 14, 15] is an interesting aspect of the current research activity on this field. The idea is that chaos provides

the right properties in term of stability and flexibility, needed by systems that evolve among different cognitive states. In particular, perceptual systems dynamically migrate among different attractors that represent the meaning of the sensory stimuli coming from the environment.

Moreover other studies consider the role of noise as an added value to reactive systems that otherwise could not be able to escape from deadlock situations produced by local minima [16].

Skarda and Freeman [13] have investigated the role of chaos in the formation of perceptual 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 that represents the meaning of the sensory input, depending on the state of the system and its 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 behaviours in living beings.

Freeman's studies leaded to a model, called K-sets, of the chaotic dynamics observed in the cortical olfactory system. This model has been used as dynamic memory, for robust classification and navigation control of roving robots [17, 10, 18, 19].

The architecture here proposed, taking into consideration the relevant principles previously underlined, is based on the control of chaotic dynamics, to learn adaptive behaviours 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]. Therefore, the WCC is a general technique that can be applied to several chaotic system [2, 20]. 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.

The multiscroll chaotic system has been chosen instead of other regular chaotic systems to exploit its modularity in the design of the multiscroll attractors in fact, the 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. This well refines the ability to select the most important information 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, 4, 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 also to 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 [21, 22], in relation to different applications.

As regards 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 pre-defined 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 [36]. 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 [23] and is able to plastically react to localized excitation by triggering a movement

(like the motor cortex or the superior colliculus in the brain) [24]. The Motor Map paradigm has been already successfully applied to solve navigation control problems [22, 25]. 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.

In the next section we will describe the control architecture that reproduces the sensing-perception-action loop. Section 3 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 section 4 whereas section 5 describes a hardware implementation based on FPGA. Finally, in section 6 and 7 we will give some remarks and draw the conclusions.

2 WCC control architecture

The control architecture here proposed 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 sensing-perception-action 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 prospective 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 creates an internal representation of the environment from the sensory inputs;
- the action selection network, which learns a reward maximizing action at the motor layer.

The perceptual block is the unit where the agent creates an internal representation of the environment perceived through the distributed sensory system equipped on it. As discussed above, taking

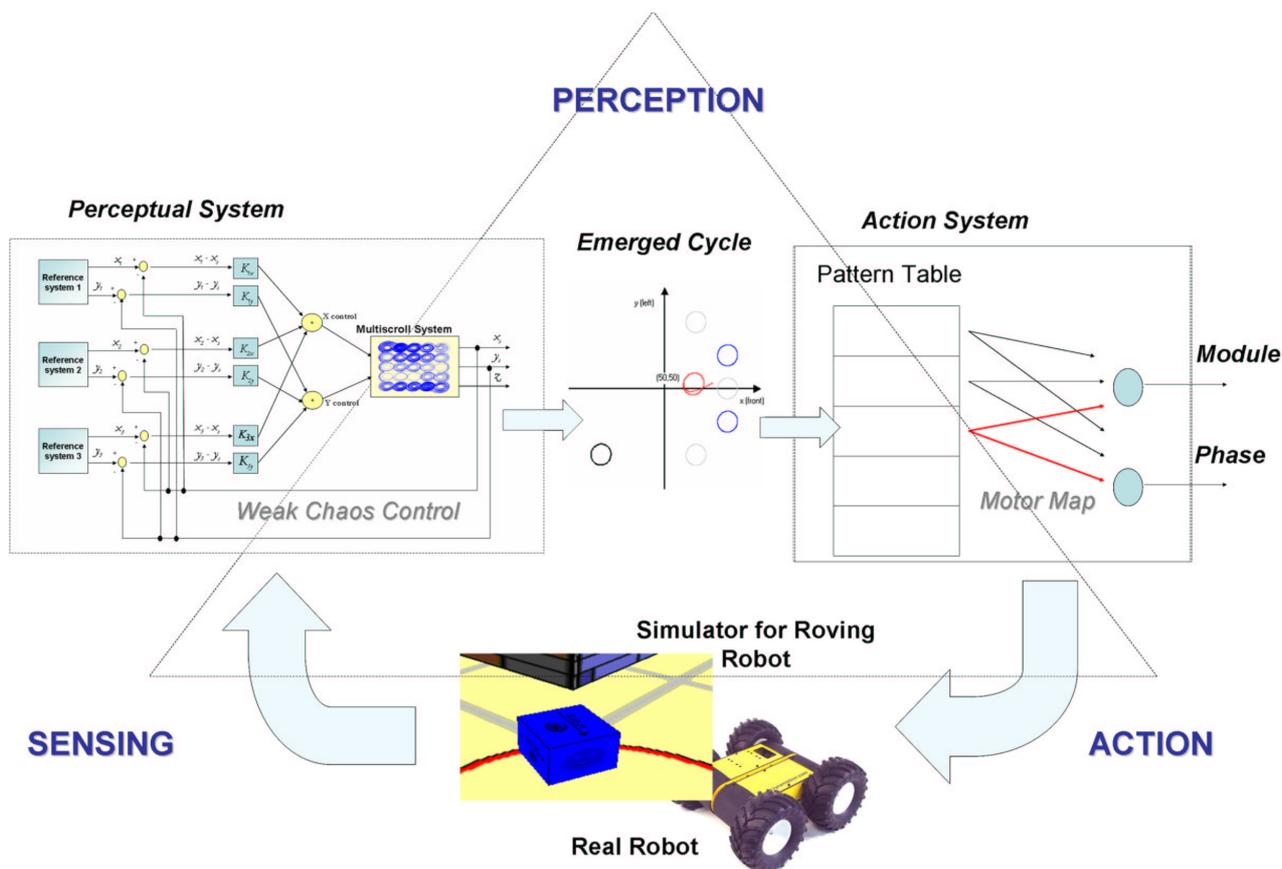


Figure 1: Control architecture.

inspiration from biological evidence, a chaotic dynamical system was adopted as an engine able to generate emergent solutions thanks to the richness of different behaviours that can arise [13, 14].

The chaotic system is here used to create a meaning of the sensory input by means of a control technique that can easily allow 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. When the stimuli stop, the multiscroll evolves in a chaotic way again.

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.

2.1 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 a compact representation of the input signals coming from the environment that can be used for real time generation of suitable actions. This block emulates the perceptual processes of the brain in which particular cerebral patterns emerge depending on the perceived sensorial stimuli. 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 the Chen's multiscroll system can be enslaved to regular periodic patterns (i.e. emergence of 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 sections, consists into a topological distribution of the sensory inputs in the system phase plane. The Chen's system can be easily designed in order to increase the number and position of scrolls and this opportunity allows to handle with a great number of distributed sensors.

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

$$\begin{cases} \dot{x} = y - \frac{d_2}{b} f(x; k_2; h_2; p_2, q_2) + \sum_i K_{x_i} (x_{m_i} - x) \\ \dot{y} = z + \sum_i K_{y_i} (y_{m_i} - y) \\ \dot{z} = -ax - by - cz + d_1 f(x; k_1; h_1; p_1, q_1) \\ \quad + d_2 f(y; k_2; h_2; p_2, q_2) \end{cases} \quad (1)$$

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:

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

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 to generate a 2-D 5×5 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. Ref_i). When the agent doesn't perceive any stimulus, the multiscroll system evolves chaotically (Fig. 2 (a)). When the control gains for the Ref_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 Ref_2), the chaotic dynamics collapses to a cycle (Fig. 2 (c)). Finally when another stimulus is perceived concurrently, for instance the sensor associated to the Ref_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)).

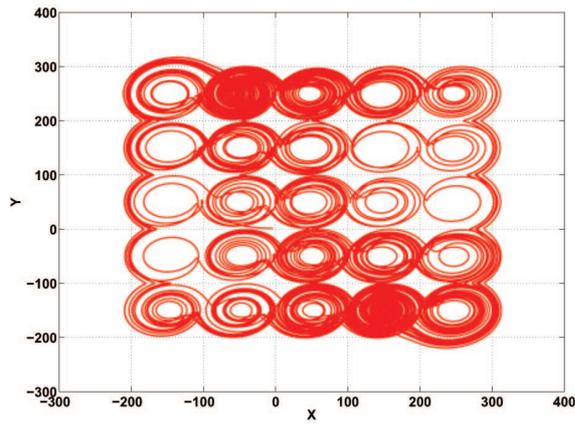
An important advantage given by the characteristic of the multiscroll system is the possibility to extend the grid of scrolls in a third dimension adding another simple piecewise linear function as described in [1]. This solution can be useful when the three-dimensional distribution of sensors in the agent is important (e.g. in a multi-legged 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.

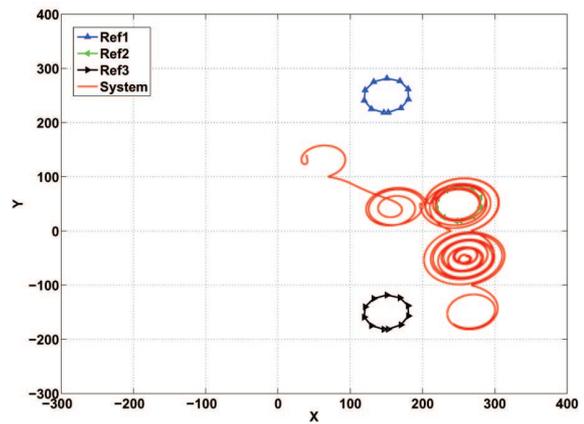
Each cycle that emerges from the control process (i.e. perceptual state) can be identified through its center position and shape. A code is then associated to each cycle and it is defined by the following parameters:

- 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;

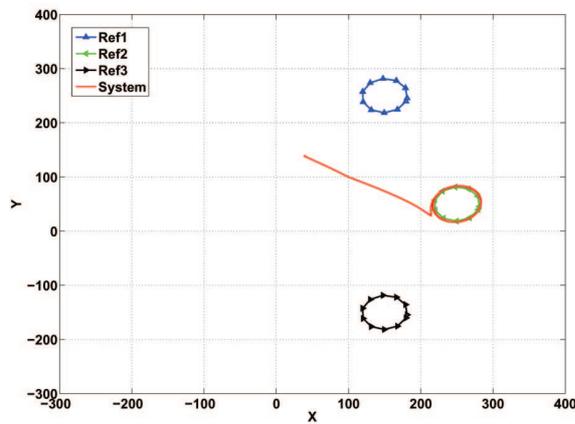
where q indicates the emerged cycle.



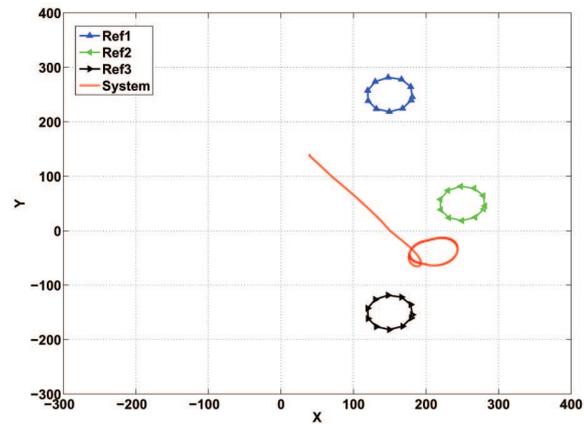
(a)



(b)



(c)



(d)

Figure 2: 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 $K_{x_3} = K_{y_3} = 15$

In this way, a few parameters provide an abstract and concise representation of the environment. This solution has been demonstrated to fit in the application to robot navigation. Other parameters could be taken into consideration to improve the representation process 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.

2.2 Action Selection Layer

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

$$action = (module, phase) \quad (3)$$

The module and phase of an action determine, respectively, the motion step and the rotation angle to be performed by the robot. To perform this task, the Motor Map (MM) paradigm was employed. MMs are suitable to control robot behaviours in an unknown environment because they are adaptive, unsupervised structures and simple enough to allow a real time learning. The MM is composed by 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:

$$\varphi : V \longrightarrow U \quad (4)$$

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 [26]. 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 [22]). The first difference is that relationship of proximity is not considered among the neurons, therefore, only the weights of the winner neuron are updated. Another and more important difference is that the afferent 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 pattern 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 synthesizing the perception of the environment. Therefore the winning neuron of the afferent layer is replaced by the element q of the pattern table which contains the last emerged cycle. Moreover the efferent (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:

$$A_q = (A_q(\text{module}), A_q(\text{phase})) = (w_{qm} + a_{s_q}\lambda_1, w_{qp} + a_{s_q}\lambda_2) \quad (5)$$

where λ_1 and λ_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:

$$RF = - \sum_i \frac{k_i}{D_i^2} - h_D D_T - h_A |\phi_T| \quad (6)$$

where D_i is the distance between the robot and the obstacle detected by the sensor i , D_T is the target-robot distance, ϕ_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 [27, 28].

3 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 an FPGA-based controller. To make easy the comparison of the proposed architecture with other more standard control techniques, a well-defined simulation set-up 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 behaviours.

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

- P_{new} : cumulative number of new perceptual states that emerge during learning;
- *Bumps*: cumulative number of collisions with obstacles;
- *Explored Area*: new area 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 process 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 an 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 can not be replaced if its a_{s_q} is under a fixed threshold (AS_{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's 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 1 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 frontal distance sensors in the reward function are higher than lateral sensors ones.

Table 1: 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_{Learn}	0.5
h_D, h_A	10	Tolerance	8%

3.1 Simulation Environment

To test the performance and the potential impact of the proposed architecture we developed a software tool for mobile robot simulations [29]. 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

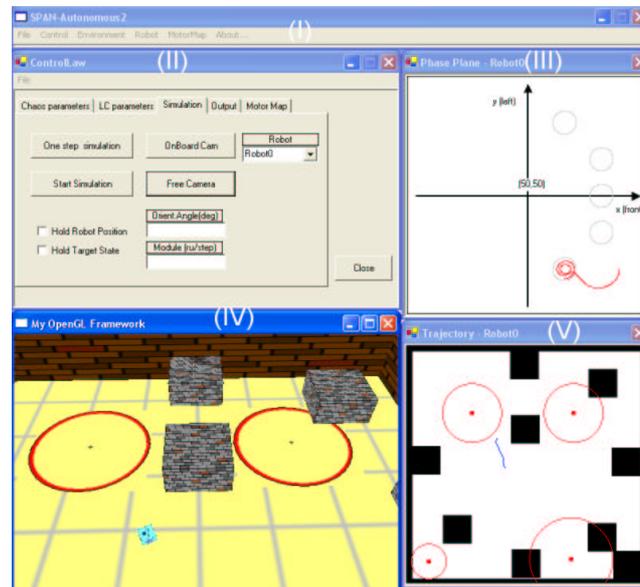


Figure 3: Interface of the 3D simulator used to compare robot navigation control algorithms: frame (I) is the menu that permits to set the robot and the environment characteristics, frame (II) permits to choose the simulation parameters, frame (III) represents the behavior of the WCC system and the active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

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 representation* and corresponding robot actions (see Fig. 3).

An example of the robot behaviour guided by the internal patterns generated through the WCC technique is shown in Fig.4. Movies of this simulation are available on the web [30].

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

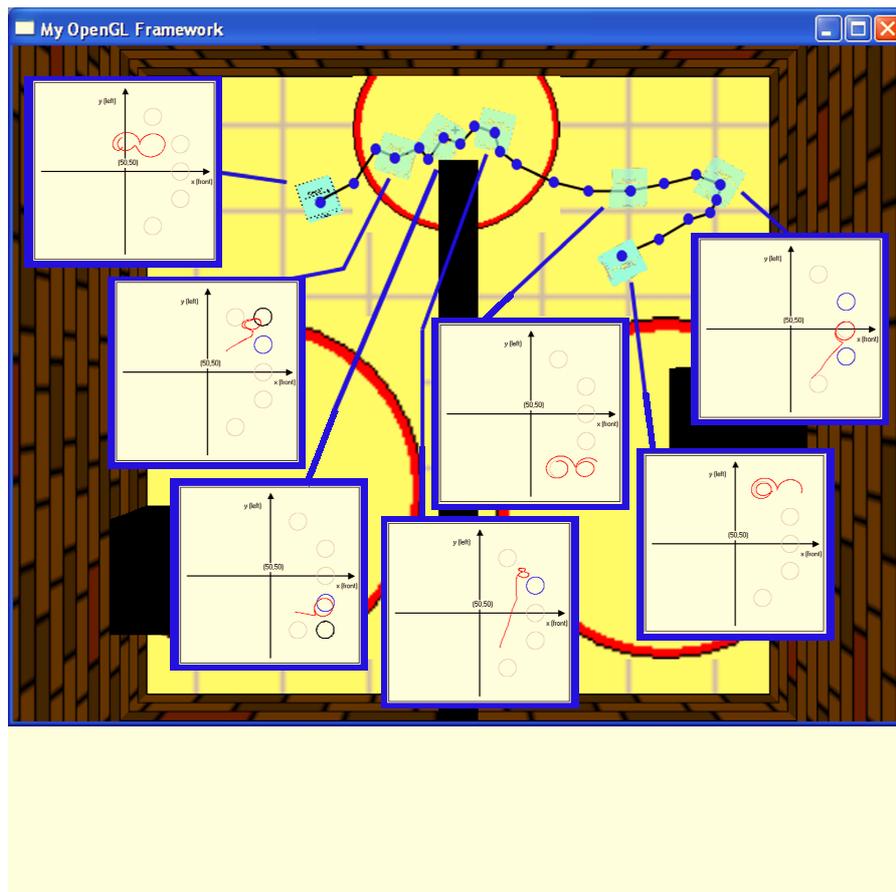


Figure 4: Example of a trajectory followed by the simulated robot and the corresponding mental patterns that emerge depending on the robot embodiment and situatedness.

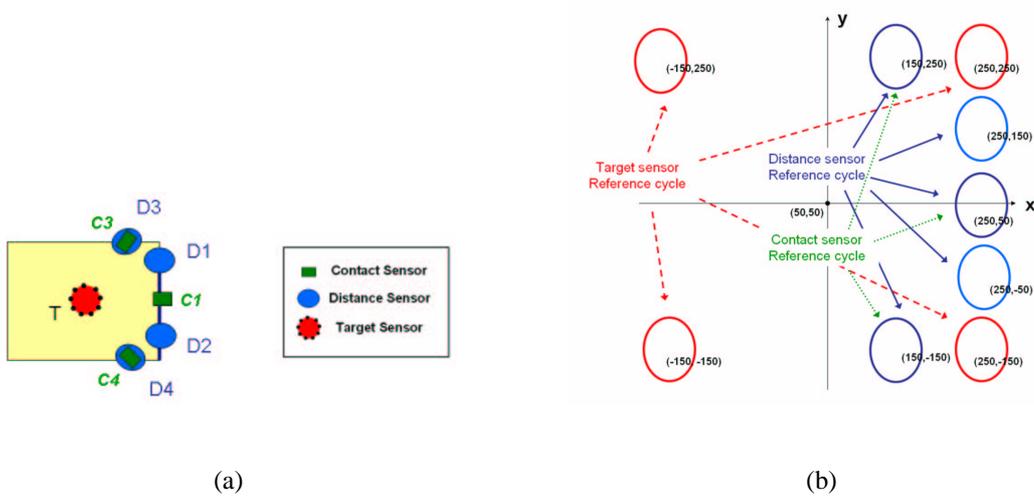


Figure 5: (a) Map of the sensors equipped on the robot. (b) Reference cycles linked to the robot sensors. Due to the common nature of contact and distance sensors, if they are placed in the same position on the robot (e.g. C3 and D3), an unique reference cycle is used. The target is considered as an 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.

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 to 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.

Fig. 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 to a robot behaviour through the MM-like structure that constitutes the action selection layer. The action selection network consists of one hundred entries and two output weights are associated to 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 to an empty slot or to a previously emerged pattern. The maximum number of non-empty entries was fixed to 100.

4 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, alternated 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 65000 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_{new}) is shown for both the environments. The learning process leads to a huge improvement of the robot behaviour for the situation (i.e. perceptual state) that more often occurs, while some other patterns can be not suitably learned if they seldom emerge.

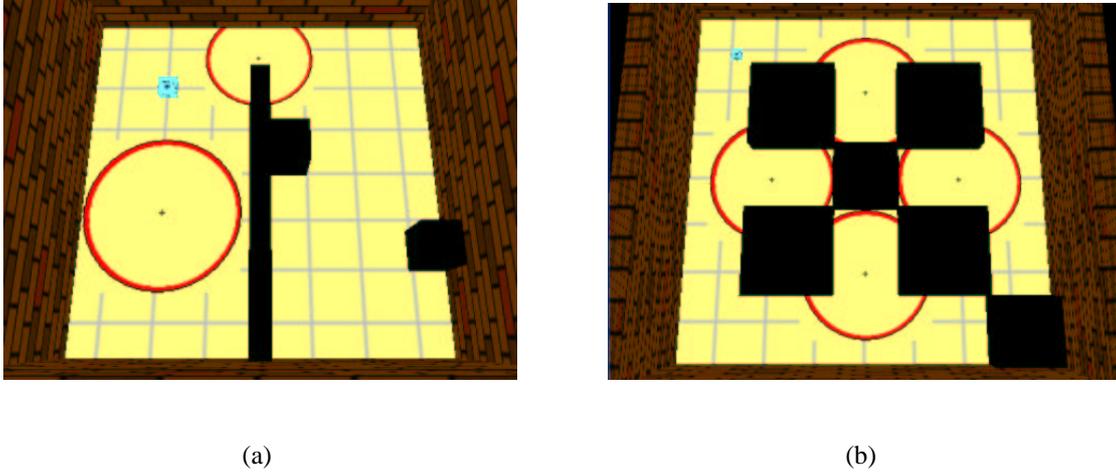


Figure 6: 3D view of the environments used during the learning phase: E_1 (a) and E_2 (b). Black boxes are obstacles, the circles represent the target detection area. The dimensions of E_1 are $20 \times 20 \text{ ru}^2$ whereas E_2 dimensions are $36 \times 36 \text{ ru}^2$.

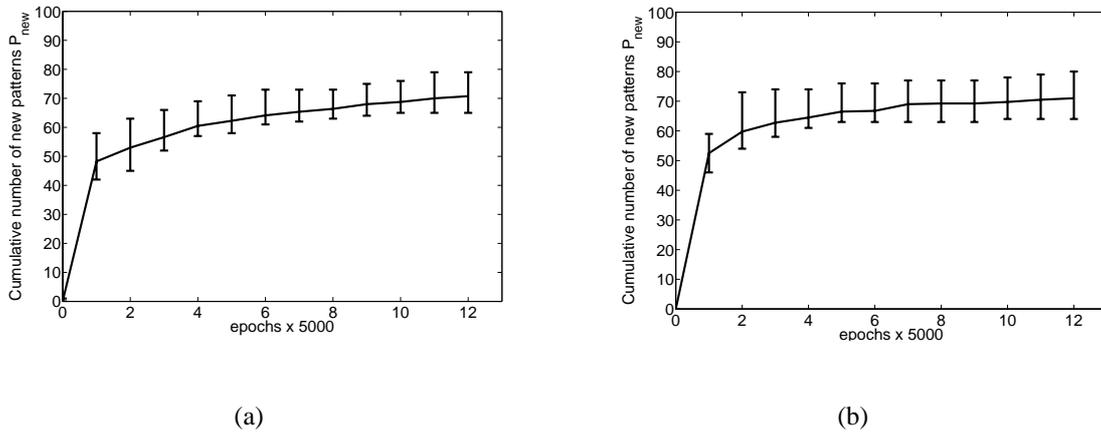


Figure 7: Cumulative number of new patterns that emerge during the learning phase calculated in windows of 5000 epochs for the environments E_1 (a) and E_2 (b). The bars indicate the minimum and maximum value whereas the solid line is the mean value of the set of five simulations performed.

After the first 5000 learning epoches, the number of new emerging patterns is very low and tends to reach a stable value. These results were obtained adopting the learning parameters defined in Tab. 1. In particular the total number of patterns that emerges, 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.

Fig. 8 shows the evolution of the learning parameter a_s in the two environments (i.e. E_1 and E_2). In Fig. 8 (a) and (b) the trend for a single simulation and for each pattern is shown. The duration of the learning phase, set to 65000 epoches, 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 Fig. 8 (c) and (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 Fig. 9 and 10, the cumulative number of bumps and targets found is shown for the two learning environments, comparing the behaviour 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 epoches), a significant difference in term of performance is evident. The robot behaviour is completely different as shown in Fig. 11 where the trajectories followed by the robot with and without learning can be compared.

The comparison between the performance with and without learning 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

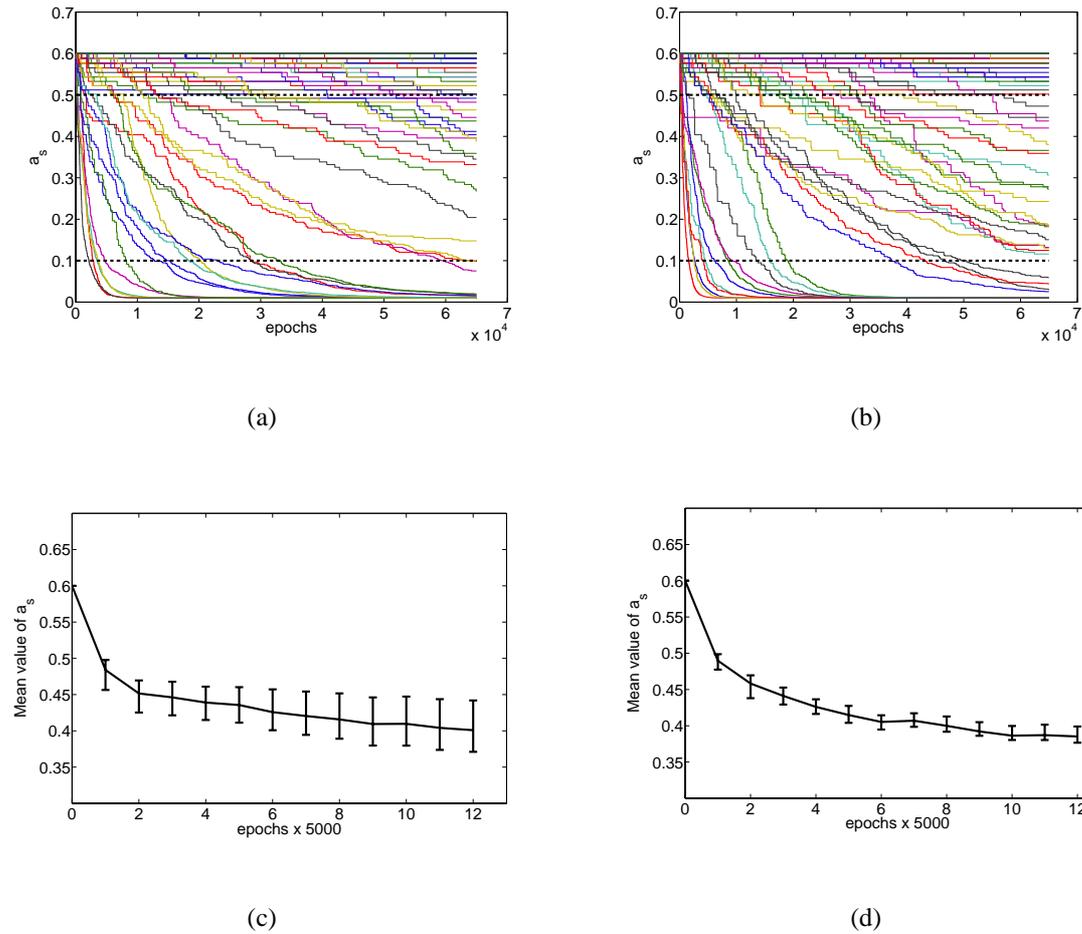
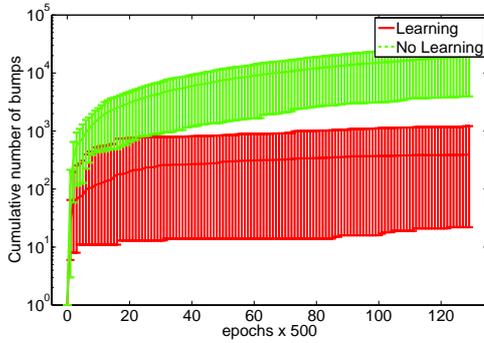
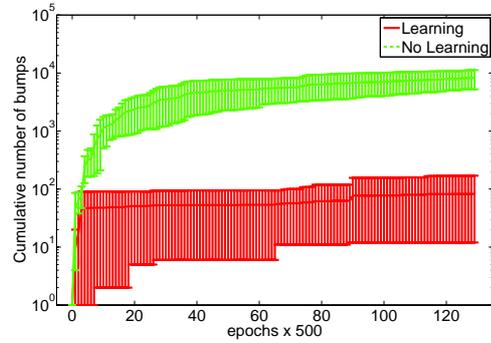


Figure 8: An example of the evolution of all the a_s during the learning phase in the environments E_1 (a) and E_2 (b). In (c) and (d) the average value of a_s among five simulations, calculated in windows of 5000 epochs for the two environments is shown. The bar indicates the minimum and maximum value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

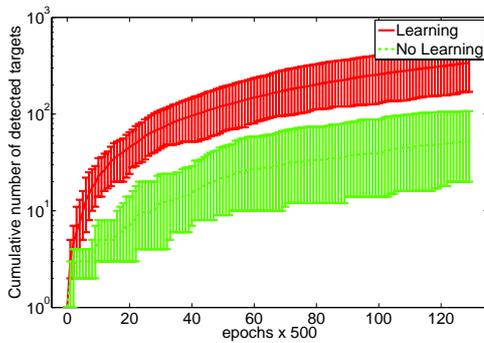


(a)

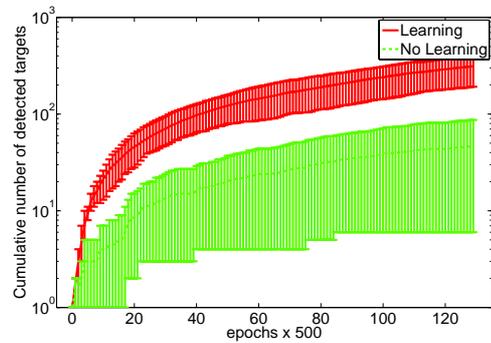


(b)

Figure 9: 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.



(a)



(b)

Figure 10: 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.

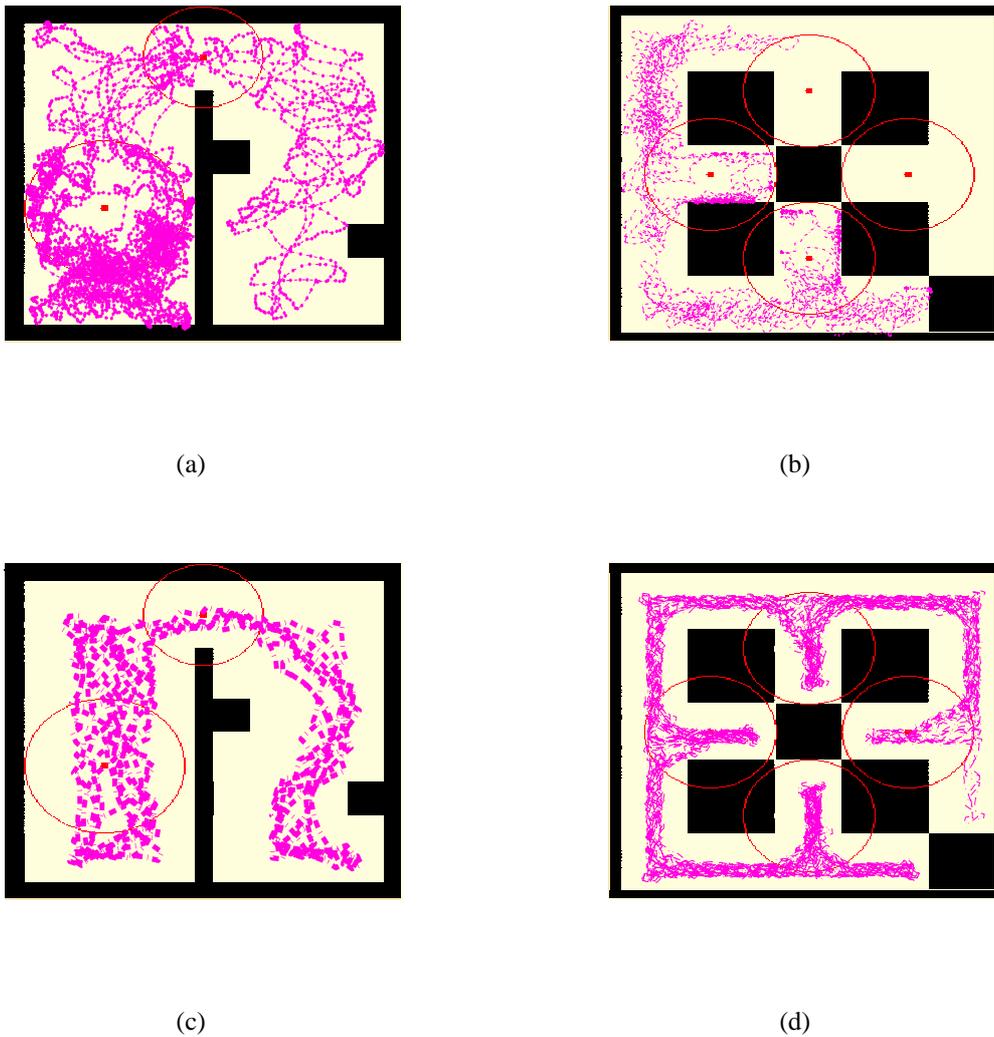


Figure 11: Trajectories followed by the robot during a test in the learning environments. (a) and (b) behaviour of the robot without learning; (b) and (d) trajectories followed after the learning phase. The simulation time is 10000 epochs.

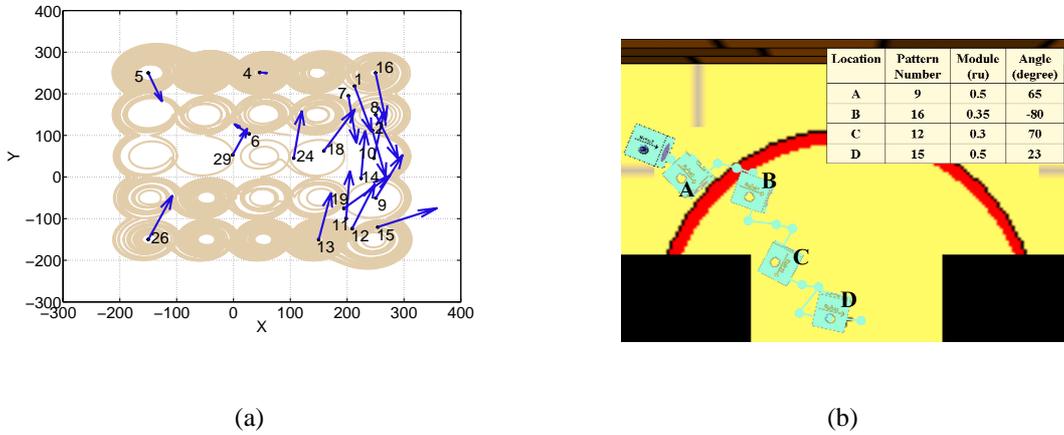


Figure 12: (a) Creation of an *Action Map*. Each vector indicates the phase action associated with the emerged patterns indicated only with their position in the phase plane. (b) Sequence of perceptual patterns and associated actions generated by the control architecture.

association between perceptual patterns and corresponding actions is reported. In particular, in Fig. 12, the final emerged actions associated to the mostly used patterns are shown. This result is related to a learning carried out in the environment E_1 . Fig. 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 to each pattern, shows module and phase of the corresponding action performed by the robot, with respect to the x-axis that indicates frontal direction of the robot motion. In Fig. 12 (b), a typical target approaching manoeuvre made by the robot is shown. The sequence of internal patterns and the corresponding actions, used to avoid the collision 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 patterns which are used as a concise environment representation. Since the environment conditions reflect situations where the robot locally interact with targets and/or obstacles, the capability to learn a motion rule can be generalized to new environments. So we will refer to this “generalization capability” shown in the following section.

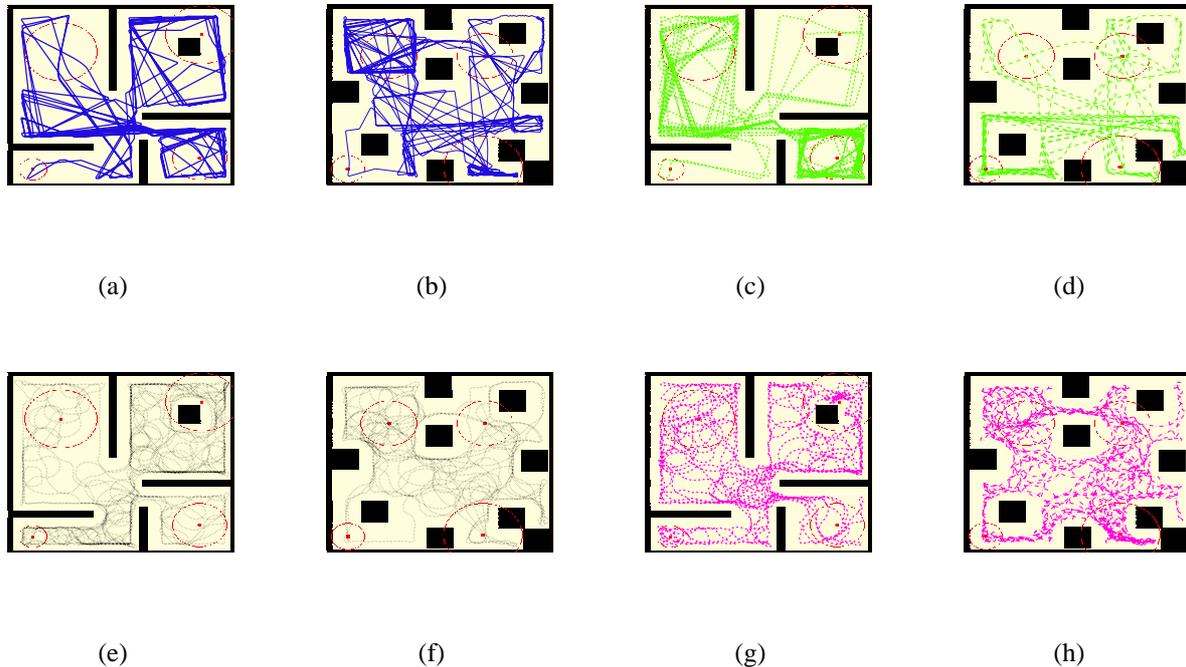


Figure 13: Trajectories followed by the robot controlled through: (a) (b) local Potential Field; (c)(d) WCC with forward exploration strategy; (e)(f) WCC with chaotic exploration behaviour; (g)(h) WCC with action selection layer based on a MM structure.

4.1 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) [31, 32]. 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 [29, 5]. The difference between the two versions is limited to the behaviour of the robot during the exploration phase (i.e. when no stimuli are perceived). The former implements a very simple behaviour that consists into a forward movement with the speed set to its maximum value (i.e. WCC_f), whereas the latter consid-

ers the chaotic evolution of the multiscroll system to determine the action of the robot exploring the environment (i.e. 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 enclosed in the MM-structure with the other control scheme, 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 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 $40 \times 40 r_u^2$.

A typical trajectory followed by the robot in each case is shown in Fig. 13. The figure qualitatively shows the robot behaviour 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 behaviour for 10000 actions.

In the case of the proposed architecture with the MM-based action selection layer and the chaotic exploration movements (WCC_{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 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 [30].

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 [19].

The performance obtained with the WCC_{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

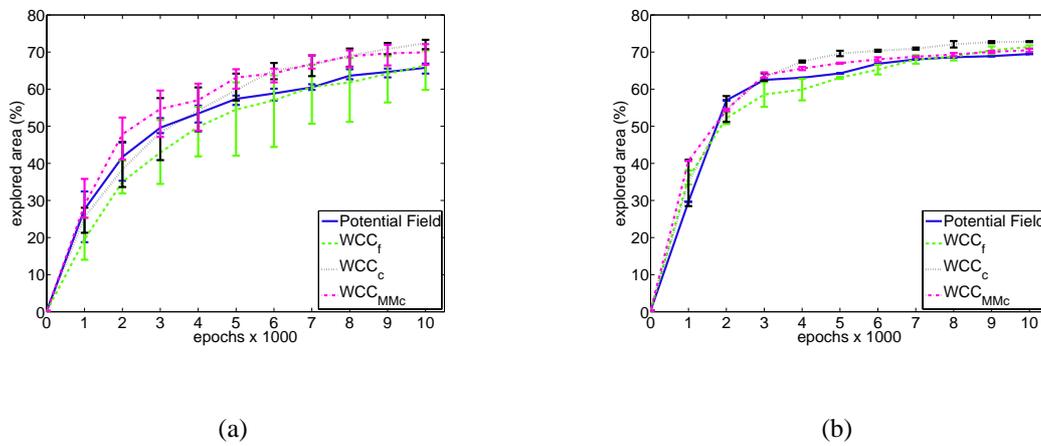


Figure 14: Value of the explored area obtained using three different control algorithms in two environments: a four-rooms environment (E_3) (a) and a less structured one (E_4)(b). The arena which dimension is 50x50 robot units, has been divided into locations of 2x2 robot units. The simulation time is 10000 epochs and the mean value of area explored, mediated over 5 simulations, calculated with time windows of 1000 epochs, is indicated. The bars show the minimum and maximum explored area.

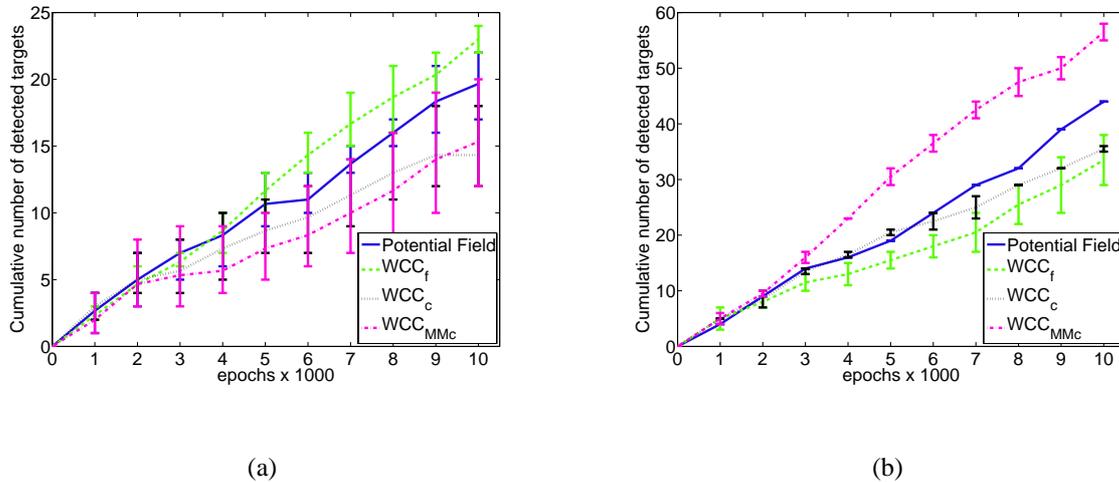


Figure 15: Cumulative number of targets found with the considered control algorithms in E_3 (a) and in a E_4 (b). The simulation time is 10000 epochs and the mean number of targets, mediated over 5 simulations, calculated with time windows of 1000 epochs, is indicated. The bars show the minimum and maximum number of targets.

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. 15 (a) in which the performances of the WCC_{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 indicated because they adopt forward actions during the exploration phase. Nevertheless the performances of the WCC_{MMc} can be improved if a learning session is performed in the testing environment. In fact, as shown in Fig. 16, the WCC_{MMc} agent, after learning on the 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 dealing with.

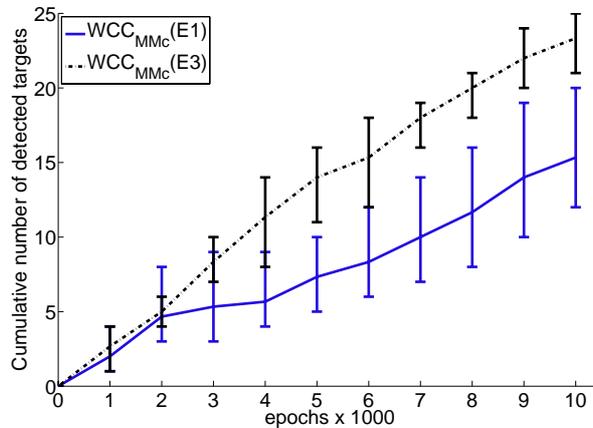


Figure 16: Comparison between the cumulative number of targets found in the environment E_3 by an agent that performed a learning phase in E_1 and another that creates its perception-action map directly in E_3 . The simulation time is 10000 epochs and the mean number of targets, mediated over 5 simulations, calculated with time windows of 1000 epochs, is indicated. The bars show the minimum and maximum number of targets found.

5 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, an 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 [33].

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 phonotaxis in crickets. A scheme of the hardware framework is shown in Fig. 17.

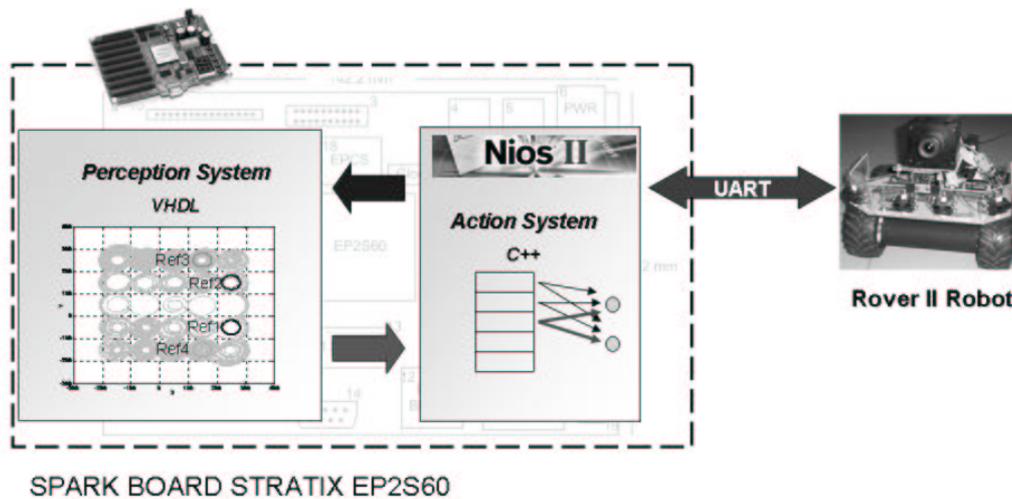


Figure 17: Description of the framework used during the experiments. An FPGA-based board equipped on a roving robot.

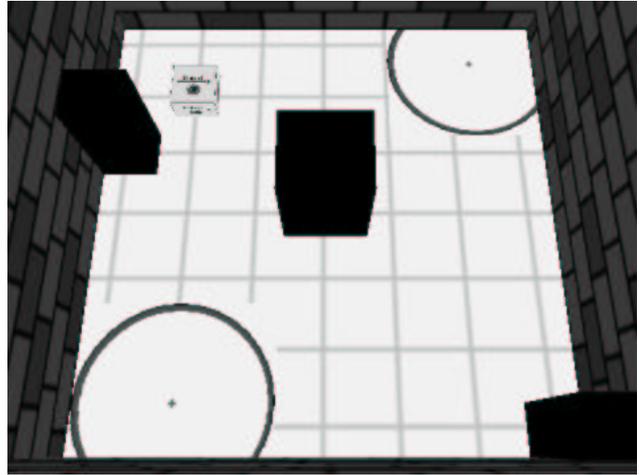
In the following experiments, to fulfill a food retrieval task, distance sensors were used for obstacle avoidance, whereas a cricket-inspired hearing board was considered for the target detection issues.

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 $10 \times 10 \text{ ru}^2$ 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.



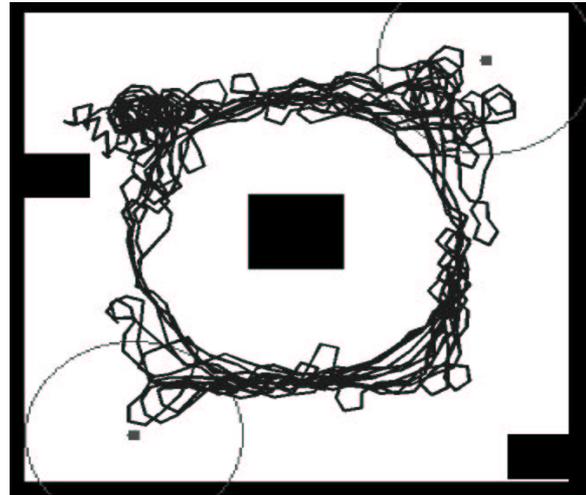
(a)



(b)



(c)



(d)

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

6 Remarks

Among the very few attempts to use dynamical systems in robot perception the Dual Dynamics (DD) scheme is worth of interest [34]. The key idea is that a robotic agent can work in different modes, which lead to qualitatively different behavioral patterns. Mathematically, transitions between modes 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 robotics 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 that can therefore 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 Coupled Map Lattice (CML), exploited for their rich dynamical properties [35], are used in experiments with a baby model [21]. Coupled chaotic system such as CML can change their behaviour 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 [36], by implementing the simplest version that leads to good results being also suitable for a reliable hardware implemen-

tation, due to its simplicity.

Another important consideration is that the WCC technique here discussed is based on a state feedback approach and the control gains are here chosen in order to grant the controlled system stability under stimulation with reference signals (see [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 evident when the robot is trapped in a deadlock situation. In this situation standard solutions proposed in 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 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 *behaviour 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 [8]), 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 here introduced within a more general scheme for robot perception in unknown conditions and environments.

7 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 to synthesize the perception schema in a compact form easy to be processed. The introduced action layer allows to generate new behaviours 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 obstacles avoidance and target retrieving tasks and 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 per-

ceptual purposes.

Acknowledgment

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*”.

References

- [1] J. Lu, G. Chen, X. Yu, H. Leung, “Design and Analysis of Multiscroll Chaotic Attractors from Saturated Function Series”, *IEEE Trans. Circuits Syst., I: Regular Paper*, Vol. 51, 2004.
- [2] M. Hulub, M. Frasca, L. Fortuna, P. Arena, “Implementation and synchronization of 3x3 grid scroll chaotic circuits with analog programmable devices”, *Chaos*. 2006 Mar ;16 (1):13121 16599752 (P,S,E,B)
- [3] P. Arena, L. Fortuna, M. Frasca, G. Lo Turco, L. Patané, R. Russo, “Perception-Based Navigation through Weak Chaos Control”, in *Proc. of 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 *Proc. of Biomedical Robotics and Biomechatronics*, 2006. BioRob 2006, pp 555-560, February 2006.
- [5] P. Arena, S. De Fiore, L. Fortuna, M. Frasca, L. Patané, G. Vagliasindi, “Reactive navigation through Multiscroll Systems: from theory to real-time implementation”, to be published on the Special Issue on Bio-Inspired Sensory-Motor Coordination of the Journal *Autonomous Robots*, DOI: 10.1007/s10514-007-9068-1.
- [6] R. C. Arkin, *Behaviour Based Robotics*, MIT Press, 1997.

- [7] L. Steels and R.A. Brooks, editors, *The Artificial Life Route to Artificial Intelligence: Building Embodied Situated Agents*, Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, 1995.
- [8] W.J. Freeman , “How and Why Brains Create Meaning from Sensory Information”, *International Journal of Bifurcation and Chaos*, Vol. 14, No. 2, 2004.
- [9] W.J. Freeman, “Characteristics of the Synchronization of Brain Activity Imposed by Finite Conduction Velocities of Axons”, *International Journal of Bifurcation and Chaos*, Vol. 10, No. 10, 1999.
- [10] R. Kozma, W. J. Freeman, “Chaotic Resonance - Methods and Applications for Robust Classification of Noisy and Variable Patterns”, *International Journal of Bifurcation and Chaos*, Vol. 11, No. 6, 2000.
- [11] W. J. Freeman, “A Neurobiological Theory of Meaning in Perception. Part I: Information and Meaning in Nonconvergent and Nonlocal Brain Dynamincs”, *International Journal of Bifurcation and Chaos*, Vol. 13, No. 9, 2003.
- [12] W. J. Freeman, “The physiology of perception,” *Sci. Am.* 264, 78-85, 1991.
- [13] C.A. Skarda and W.J. Freeman, “How brains make chaos in order to make sense of the world,” *Behav. Brain Sci.*, vol. 10, pp. 161195, 1987.
- [14] I. Aradi, G. Barna, and P. Erdi, “Chaos and learning in the olfactory bulb,” *Int. J. Intell. Syst.*, vol. 1091, pp. 89117, 1995.
- [15] P. Arena, L. Fortuna, M. Frasca, L. Patané, “Sensory feedback in CNN-based CPG,” *International Journal of Neural Systems*, Vol. 13 No. 6 December 2003.
- [16] V.A. Makarov, N.P. Castellanos, and M.G. Velarde, “Simple agents benefits only from simple brains,” *Trans. Engn., Computing and Tech.*, vol.15, pp. 25-30, 2006.

- [17] W. J. Freeman and R. Kozma, “Local-global interactions and the role of mesoscopic (intermediate-range) elements in brain dynamics,” *Behav. Brain Sci.*, vol. 23, no. 3, p. 401, 2000.
- [18] R. Gutierrez-Osuna and A. Gutierrez-Galvez, “Habituation in the KIII olfactory model with chemical sensor arrays,” *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1565-1568, Nov. 2003.
- [19] D. Harter, R. Kozma, “Chaotic Neurodynamics for autonomous agents”, *IEEE Trans. on Neural Networks*, 16(3), pp. 565–579, 2005.
- [20] M. E. Yalcin, J. A. K. Suykens, J. Vandewalle, and S. Ozoguz, “Families of scroll grid attractors”, *Int. J. Bifurc Chaos*, vol. 12, no. 1, pp. 23-41, (Jan 2002).
- [21] Y. Kuniyoshi and S. Sangawa, “Early motor development from partially ordered neural-body dynamics: Experiments with a cortico-spinal-musculo-skeletal model”. *Biological Cybernetics*, 95(6):589-605, December 2006.
- [22] 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”, *International Journal of Bifurcation and Chaos*, pp 107-127, January 2007.
- [23] T. Kohonen, “Self-organized formation of topologically correct feature maps”, *Bio. Cybern.*, 43, 59-69, 1972.
- [24] J.D. Murray, “Mathematical Biology”, Springer, 2002.
- [25] Arena P., Fortuna L., Frasca F., Pasqualino R., Patané L., “CNNs and Motor Maps for Bio-inspired Collision Avoidance in Roving Robots”, *8th IEEE Int. Workshop on Cellular Neural Networks and their Applications (CNNA 2004)*, Budapest, 2004.
- [26] H. Ritter, T. Martinetz and K. Schulten, *Neural Computing and Self-Organizing Maps Reading*, MA: Addison-Wesley, 1992

- [27] P. Arena, L. Fortuna, M. Frasca, D. Lombardo, L. Patané, “Learning efference in CNNs for perception-based navigation control”, *in proc. of International Symposium on Nonlinear Theory and its Applications, NOLTA'05* , Bruges, Belgium, October 18-21, 2005.
- [28] P. Arena, L. Fortuna, M. Frasca, L. Patané, M. Pavone, “Towards Autonomous Adaptive Behavior in a bio-Inspired CNN-Controlled Robo”, *in proc. of IEEE International Symposium on Circuits and Systems (ISCAS 2006)*, Island of Kos, Grecia, 21-24 April 2006.
- [29] P. Arena, L. Fortuna, M. Frasca, G. Lo Turco, L. Patané and R. Russo, “A new simulation tool for actionoriented perception systems”, *in proc. of 10th IEEE International Conference on Emerging Technologies and Factory Automation, (ETFA 2005)*, Catania, Italy, September 19-22, 2005.
- [30] “Spark home page” [Online]. Available: <http://www.spark.diees.unict.it/WCC2.html>
- [31] O. Khatib, “Real-time Obstacle Avoidance for Manipulators and Mobile Robots,” *in proc. International Journal of Robotics Research*, vol. 5, no. 1, pp 90–98, , 1986.
- [32] R. Beard and T. McClain, “Motion Planning Using Potential Fields,” BYU, 2003.
- [33] P. Arena and L. Patané, “Spatial Temporal Patterns for Action Oriented Perception in Roving Robots”, Springer Tracts in Cognitive Systems, Vol. 1, 2008 in press.
- [34] H. Jaeger and Th. Christaller, “Dual dynamics: Designing behavior systems for autonomous robots”. In S. Fujimura and M. Sugisaka, editors, *Proc. Int. Symposium on Artificial Life and Robotics (AROB'97)*, pp. 76–79, 1997.
- [35] K. Kaneko and I. Tsuda, *Complex Systems: Chaos and Beyond*, Springer, 2001.
- [36] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, a Bradford Book, MIT Press Cambridge, Massachusetts London, England, 1998.