Biol Cybern (2007) 97:363–378
DOI 10.1007/s00422-007-0176-y

ORIGINAL PAPER

Biological
Cybernetics

Chained learning architectures in a simple closed-loop behavioural context

Tomas Kulvicius · Bernd Porr · Florentin Wörgötter

Received: 25 January 2007 / Accepted: 20 August 2007 / Published online: 3 October 2007

Abstract
Objective Living creatures can learn or improve their behaviour by temporally correlating sensor cues where near-senses (e.g., touch, taste) follow after far-senses (vision, smell). Such type of learning is related to classical and/or operant conditioning. Algorithmically all these approaches are very simple and consist of single learning unit. The current study is trying to solve this problem focusing on chained learning architectures in a simple closed-loop behavioural context.
Methods We applied temporal sequence learning (Porr B and Wörgötter F 2006) in a closed-loop behavioural system where a driving robot learns to follow a line. Here for the first time we introduced two types of chained learning architectures named linear chain and honeycomb chain. We analyzed such architectures in an open and closed-loop context and compared them to the simple learning unit.
Conclusions By implementing two types of simple chained learning architectures we have demonstrated that stable behaviour can also be obtained in such architectures. Results also suggest that chained architectures can be employed and better behavioural performance can be obtained compared to simple architectures in cases where we have sparse inputs in time and learning normally fails because of weak correlations.

Keywords Temporal sequence learning · Sparse inputs · Weak correlations · Line-following · Driving robot ·

1 Introduction

Normally many sensor events, which follow each other in time, are associated to a real-life situation. However, reacting to only a few will improve the behaviour. This situation can be addressed by mechanisms of temporal sequence learning. These mechanisms rest on the assumption that it is, in most cases, advantageous to react to the earliest of such sensor events, not having to wait for following ones. For example, it is useful to react to a heat radiation signal and not to the later pain on having finally touched a hot surface. Many similar sequences of sensor events are encountered during the lifetime of a creature as the consequence of the existing far senses (e.g. vision, hearing, smell) and near-senses (touch, taste, etc.). Generically one observes that the trigger of a near-sense is preceded by that of a far sense (smell precedes taste, vision precedes touch, etc.). Far-senses act predictively with respect to the corresponding near-senses (Verschure and Coolen 1991). Conceptually this type of learning is related to classical and/or operant conditioning (Sutton and Barto 1981; Sutton and Barto 1990; Wörgötter and Porr 2005). Algorithmically all these approaches (Sutton and Barto 1981; Kosco 1986; Klopf 1988; Porr and Wörgötter 2003a) share the property that they are built in a very simple way, in general only consisting of a single learning unit.

Here we will apply temporal sequence learning to a driving robot that is supposed to learn to better follow a line painted on the ground. We will try to answer two questions:

T. Kulvicius · F. Wörgötter (✉)
Bernstein Centre for Computational Neuroscience,
University of Göttingen,
Bunsenstr. 10, 37073 Göttingen, Germany
e-mail: worgott@bccn-goettingen.de

T. Kulvicius
Department of Informatics, Vytautas Magnus University,
Kaunas, Lithuania
e-mail: tomas@bccn-goettingen.de

B. Porr
Department of Electronics and Electrical Engineering,
University of Glasgow, Glasgow, GT12 8LT, Scotland, UK

(1) whether it is possible to design simple chains of learning units while at the same time still guaranteeing behavioural stability, and (2) can chained architectures be employed in order to obtain better behavioural performance compared to the simple architecture in cases where we have sparse inputs in time and weak correlations.

We believe that the embedding of learning architectures into behaving systems, which close the loop between perception and action, is an important field of investigation leading away from the pure stimulus–response paradigm to a more-ecological system perspective. The current study is meant to provide a specific contribution to the solution of this problem focusing on chained learning architectures in a simple closed-loop behavioural context.

The paper is organised as following. After presenting our input–input correlation (ICO) learning rule (Porr and Wörgötter 2006) and its embedding into a closed-loop scenario we will first show results with a simple architecture. In this way we want to demonstrate the efficiency and stability of the ICO rule and fast learning in the line-following task using relatively high learning rates. Next we will introduce two simple chained architectures and present the behaviour of these architectures in an open-loop case. Finally, we will show results for chained architectures in a closed-loop context and compare these architectures with the simple setup.

2 Methods

2.1 Robot setup

We used a small (diameter 18 cm) two-wheeled Rug Warrior Pro driving robot for the investigation (Fig. 1a). Figure 1b shows the physical setup used for learning. A camera mounted at the front of the robot produces images of the track like the one shown. Since the robot drives forward, obviously sensor fields towards the top of the image ($x_1^{L,R}$) represent far-sensors, while those at the bottom ($x_0^{L,R}$) can be regarded as near-sensors. Initially we implemented only a crude, abrupt, and aversive steering reflex as soon as the image of the track moved over one of these near-sensor fields. As a consequence the robot was forced back to a situation where the track remained mostly in the centre of the image. The learning goal was to learn predictive and smoother steering reactions. This was achieved by changing the synaptic weights of the far-sensor fields in an appropriate way such that earlier and smoother steering reactions are elicited, leading to the situation that the near-sensor fields will never be triggered (hence avoiding the initial reflex).

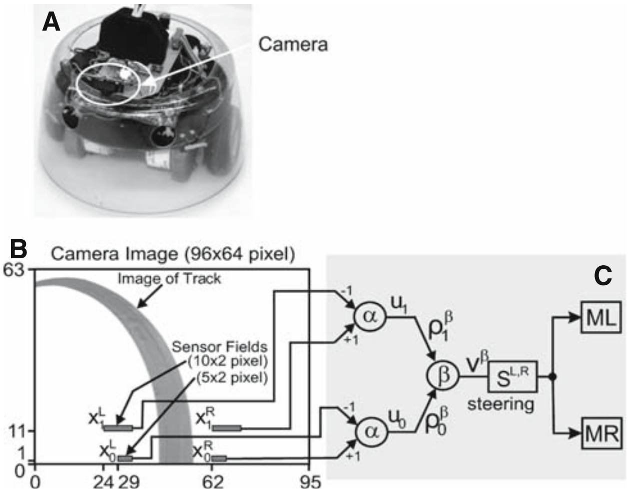

Fig. 1 Physical and neuronal setup of the learning. **a** Image of the Rug Warrior Pro driving robot. **b** Camera image with sensor fields marked by $x_1^{L,R}$ and $x_0^{L,R}$. **c** The simple neuronal setup of the robot. The symbols α and β denote neurons, u denotes the filtered input signals x, ρ the connection weight, and v output of the neuron used for steering. v is calculated by the method shown in Fig. 2b and its corresponding Eq. 1. $S^{L,R}$ is given in Eq. 3 and transforms v to the motor output

2.2 Learning algorithm

The learner (Fig. 2b) has inputs x_j that feed into a summation unit v. The output is calculated as

$$v = \sum_j \rho_j u_j \ , \tag{1}$$

where $u = h \times x$ is a temporal convolution of the input x with a resonator h. We define $h(t) = \frac{1}{b} e^{at} \sin(bt)$, $a = -\pi f / Q$, and $b = \sqrt{(2\pi f)^2 - a^2}$, where f is the frequency and $Q > 0.5$ the damping. This convolution correlates temporally nonoverlapping signals (Fig. 2a).

The time delay T (Fig. 2a) between x_0 and x_1 depends on the speed of the robot. To accommodate for some variability, x_1 is fanned out and fed into a filterbank of different filters h as indicated by the dashed lines in Fig. 2b. As shown in our older studies, the number of filters is not critical and we use 10 (Porr and Wörgötter 2003a, 2006). The robot's base speed of 0.125 m/s together with the camera frame rate of 25 Hz used in all experiments leads to $f_{1,k} = 2.5/\text{kHz}$, $k = 1, \ldots, 10$ for the filterbank in the x_1 pathway. The frequency of the x_0 pathway was $f_0 = 1.25$ Hz. The damping parameter of all the filters was $Q = 0.6$.

Weights change according to an input–input correlation (ICO) rule (Porr and Wörgötter 2006):

$$\dot{\rho}_j = \mu u_j \dot{u}_0, \quad j > 0 \ , \tag{2}$$

which is a modification of the ISO learning rule (Porr and Wörgötter 2003a). The behaviour of this rule and its convergence properties were discussed in a recent article (Porr and Wörgötter 2006).

The weight ρ_0 is set to a fixed value; all other weights are initially zero. As discussed above, this learning rule is specifically designed for a closed-loop system where the output of

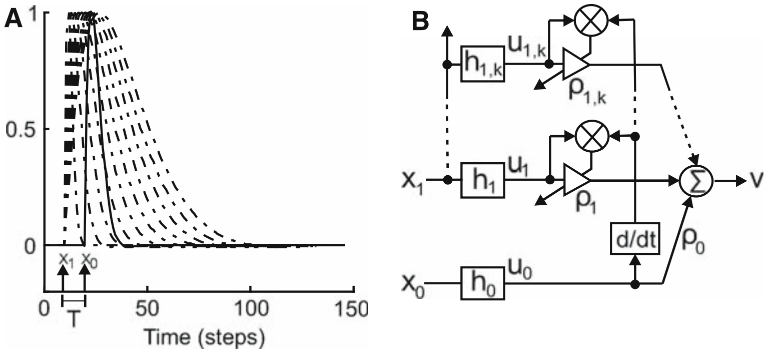

Fig. 2 **a** Resonator filters h_0 (*solid line*) for the input signal x_0 and $h_{1,k}$ (*dashed lines*) for the x_1 given by parameters $f_{1,k} = 2.5/k$Hz, $k = 1, \ldots, 10$ for the filterbank in the x_1 pathway. The frequency of the x_0 pathway was $f_0 = 1.25$ Hz. The damping parameter of all filters was $Q = 0.6$. **b** Schematic diagram of the learning system: inputs x, resonator filters h, connection weights ρ, output v. The symbol $\otimes$ denotes a multiplication, d/dt represents a temporal derivative. The amplifier symbol stands for a variable connection weight. *Dashed lines* indicate that the input x_1 is fed into a filterbank

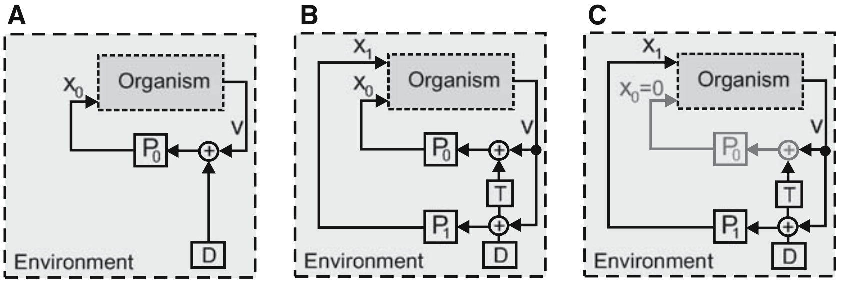

Fig. 3 Schematic diagram of the control (**a**), learning (**b**), and post-learning case (**c**). Components of the learning system are: sensor inputs x_0 and x_1, motor output v; P_0 denotes a reflexive pathway and P_1 a predictive pathway. D is a disturbance and T is a time delay

the learner v feeds back to its inputs x_j after being modified by the environment (see Fig. 3).

The goal of the learning is to learn predictive steering reactions in such a way that the initial reflex is avoided. This is achieved by changing the connection weights ρ_1 such that the learner can use the earlier signal at x_1 to generate an anticipatory reaction. The weights stabilise and learning stops at the condition $x_0 = 0$ when the reflex is no longer triggered. The convergence properties of this kind of closed-loop learning are discussed in Porr and Wörgötter (2006) and Porr et al. (2003b).

2.3 Embedding learning in a closed-loop scenario

Figure 3 shows how such a learning unit can be embedded in a closed-loop system. Initially (Fig. 3a) the system is set up only to react to the near-sense x_0 by way of a reflex. This reflex will, after some behavioural delay, reset the signal form the near-sensor to its starting value (often zero), closing the loop. In more-technical terms, this represents a negative-feedback loop controller. The learning system, however, contains a second predictive loop (Fig. 3b) from a sensor x_1 that receives an earlier signal (the far-sensor). At the beginning of the learning, the synapses ρ_1 that convey information from the far-sensor are zero and in Fig. 3b only the inner loop x_0 is functioning. During learning, the synapses ρ_1 will become strengthened and the system will react better to the far-sense. As a consequence reactions occur earlier and the reflex based on x_0 will no longer be triggered. Effectively, the inner loop has functionally been eliminated after learning (Fig. 3c). A forward model of the reflex has been built (Porr et al. 2003a). The learning of a forward model makes this approach appear similar to feedback-error learning as introduced by Gomi and Kawato (1993), but there are distinctive differences as will be discussed later (see Sect. 5).

Intuitively the mechanism introduced in Fig. 3 will work with any aversive reflex. One should, however, note that the same mechanisms can also be used to learn earlier attraction reactions. Braitenberg (1984) has nicely demonstrated that it is the sign combination of the motor signals that determines the resulting reaction (aversion versus attraction) in his vehicles. Here, similarly, we can define the behavioural outcome by ways of the motor signals leaving the learning mechanism unaffected (see Porr and Wörgötter (2003b,2006) for examples of attraction reflexes). Regardless of the motor signs, the learning goal is always to *avoid the earlier, near-sense-triggered reflex*, leading to a situation where $x_0 = 0$. We were able to prove mathematically that synaptic weights will no longer change as soon as this condition ($x_0 = 0$) is fulfilled (Porr et al. 2003b; Porr and Wörgötter 2006). Hence

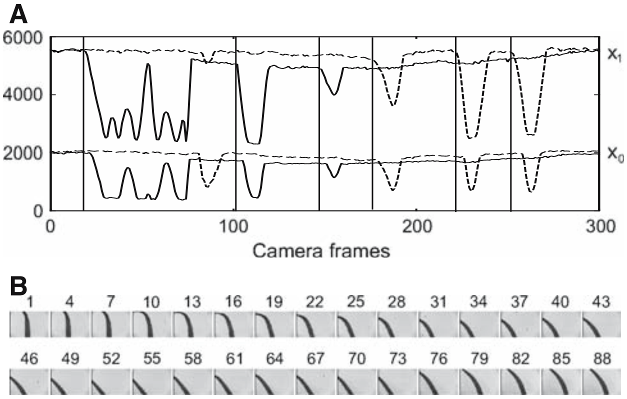

Fig. 4 **a** The input signals $x_{0,1}$ of the learning system. *Dashed lines* represent signals from the right (x^R) and *solid lines* those from the left sensor fields (x^L). The track layout is shown in Fig. 5c. Signals before camera frame 150 come from the left turn, those after frame 150 from the right turn of the robot. **b** Sequence of camera frames taken from the *left curve*

learning terminates as soon as the newly learnt behaviour is successful, which creates a nice self-stabilising property of such systems.

2.4 Input signals

As described in Sect. 1, a far-sensor (predictive) pathway and a near-sensor (reflexive) loop can be defined from sensor fields in the image of a forward-pointing camera on the robot. Figure 4b shows a sequence of camera frames obtained during a left curve and the corresponding raw input signals (Fig. 4a) obtained from the sensor fields $x_{0,1}^{L,R}$ (the sum over all pixels within the sensor field). The vertical solid lines in panel a show that the signals at x_1 are indeed earlier than those at x_0. The sequence of camera frames in Fig. 4b demonstrates that the ego-motion of the robot creates a degree of variability in the field of vision of the robot (see video camera.mpg[1]), for example the moving out and in of the bent line is clearly visible in the second row in Fig. 4b. This creates a temporally inverted sequence of input events. Learning needs to be robust against such effects as well as against other problems that arise from this behaviourally self-generated variability.

2.5 Simple architecture

A simple neuronal setup for the robot is presented in Fig. 1c. It has three neurons: two are essentially only summation nodes, which we for consistency also call neurons α. They have fixed weights (+1 for right-side inputs and −1 for left-side inputs). In addition there is one neuron β with changing synapses on which all signals converge. The synaptic weights ρ_0^β are also set to a fixed value of 1 and only the weights ρ_1^β of the 10 filters (Fig. 2a) change. The output v^β is used to modify the motor signals of the robot. Note that in this experiment the setup for the weight development is symmetric but with inverted signs for the left versus the right curve. Hence only one set of weights ρ_1^β develops. This is motivated by the fact that, in a natural setup, the left and right curves do not have any a priori bias. Situations where, for example, left curves are always on average sharper than right curves are not realistic. Hence, weights learnt for a left curve might as well be applied, with inverted sign, to a right curve (and vice versa), where learning will commence if the learnt weights are not sufficient. Given that the learning algorithm is linear, it would not make any difference if inputs were all converging directly onto β. Note that, since the robot is continuously driving, we perform online and not batch learning.

2.6 Motor outputs

The robot has a left and a right motor, which receive a certain forward drive leading to a constant speed of $S = 0.125\,\text{m/s}$ in all experiments. This signal is modified by braking ($|v^\beta|$) and steering ($\pm v^\beta$) according to:

$$S^{L,R} = 1.1905 \times 10^{-4}(3097 - g(|v^\beta| \pm v^\beta)) - 0.2437\,\text{m/s}, \tag{3}$$

where for the left motor we use the minus and for the right the positive sign. Numerical constants as well as g are determined by the 12-bit resolution of the digital-to-analog (DA) converter used, where 0 is the maximal reverse speed and 4095 is the maximal forward speed. For the chained architectures, introduced later (Fig. 10b, c), we use v^γ instead of v^β in Eq. (3).

3 Results

3.1 Basic behaviour of the simple system

The simple architecture (Fig. 1b, c) was applied in the line-following task and three different tracks were used in this experiment (see Fig. 5). Trajectories are shown for a left and a right curve (Fig. 5d, g, j). The weights and motor signals corresponding to the tracks are shown to their left. The sensor fields for the predictor x_1 and the reflex x_0 are depicted in Fig. 1b. The late and weak reflex response by itself is not enough to assure line-following behaviour; therefore the robot misses the line whenever it drives without learning (not shown, but see video control.mpg). In Fig. 5a, b two learning trials (separated by a dashed line) are shown, between which connection weights were frozen and the robot

[1] Videos can be downloaded at http://www.nld.ds.mpg.de/~tomas/DrivingRobot/

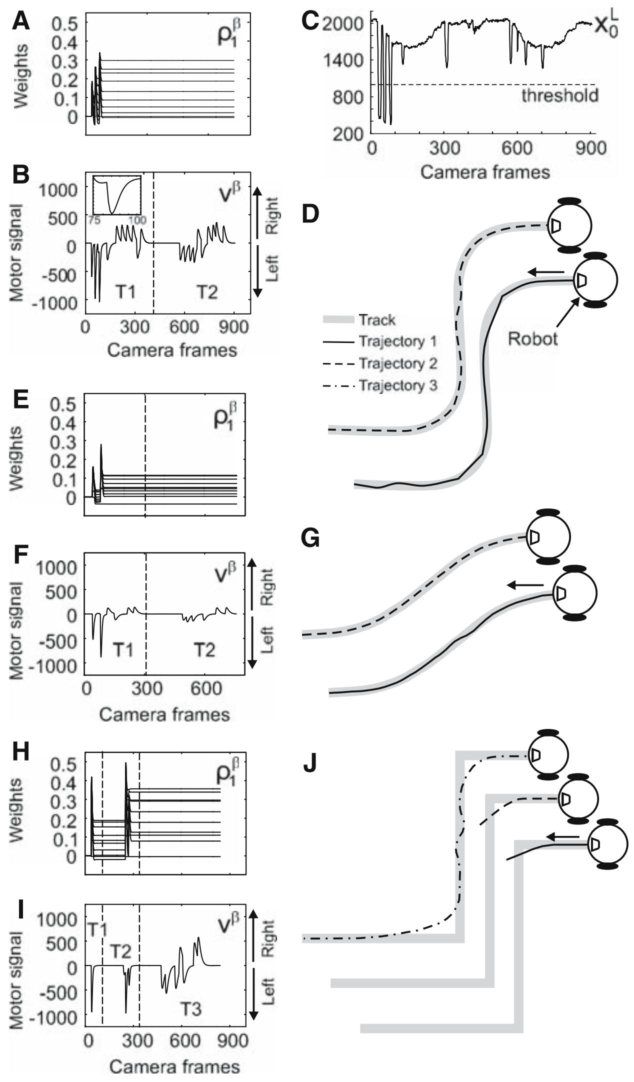

Fig. 5 Results of the driving robot experiment using the simple architecture (see Fig. 1c). **a–d** Results for the intermediately steep track (**d**); the learning rate was $\mu = 3 \times 10^{-6}$. **a** Connection weights ρ_1^β: **b** motor output v^β, **c** reflex signal x_0^L, and **d** driving trajectories; trajectory T1 during, and T2 after, learning. **e–g** Results for the shallow track (**g**); the learning rate was $\mu = 2.5 \times 10^{-6}$: **e** connection weights ρ_1^β, **f** motor output v^β, and **g** driving trajectories. **h–j** Results for the sharp track (**j**); the learning rate was $\mu = 6.5 \times 10^{-6}$: **h** connection weights ρ_1^β, **i** motor output v^β, and **j** driving trajectories

was manually returned to its starting position. A rather high learning rate $\mu = 3 \times 10^{-6}$ was chosen to demonstrate fast learning. The cumulative action of the reflex and predictive response already allows the robot to stay on the line *during* the first learning trial (see Fig. 5d, trajectory T1). In the first learning trial the motor signal (Fig. 5b) shows three leftward cumulative reflexive+predictive reactions (large troughs) and seven (two leftward and five rightward) nonreflexive (predictive) reactions. Note that cumulative responses consist of two components: the first component, smaller in amplitude, is the predictive response, whereas the second, larger in amplitude, is the reflexive response (see the inset in Fig. 5b). In the second trial only nonreflexive leftward and rightward steering signals occurred and the reflex was no longer triggered. An appropriate steering reaction was learnt after three reflexes (reflected by the three peaks in the weight curve in Fig. 5a) during the first learning trial corresponding to about 50 cm of the track (the whole track is about 2 m). The left reflex signal x_0^L is shown in Fig. 5c, where we observe that the reflex was triggered three times (three troughs below the threshold), which correspond to three learning experiences. To ensure weight stabilisation (at the condition $x_0 = 0$) we employ a threshold where values of x_0 above the threshold were set to zero (similarly to the mechanical arm experiment in Porr and Wörgötter 2006). Due to the symmetry of this setup (Fig. 1c), results from the learnt left curve could be equally applied to the right curve and no more reflexes are triggered after these first three learning experiences. Also we observe that after learning the robot steers more smoothly (see video simple.mpg).

In addition two more-extreme tracks were chosen to demonstrate the robustness of these findings. The results for a shallower track (Fig. 5e–g) are similar to those from the previous experiment but for this track learning stopped after just two reflexes, even with a lower learning rate of $\mu = 2.5 \times 10^{-6}$ compared to the previous experiment where $\mu = 3 \times 10^{-6}$. As expected a much weaker steering reaction (Fig. 1f) was learnt and the weights were smaller (for a movie of the learning behaviour see shallow.mpg).

The third experiment was performed using a track with very sharp corners (Fig. 5j) and a higher learning rate of $\mu = 6.5 \times 10^{-6}$. This was done to demonstrate that fast stable learning is possible even for such a sharp track. The results of three learning trials (separated by dashed lines) are presented in Fig. 5h–j. The robot missed the track twice and finally succeeded in the third trial (see also sharp.mpg). As before, it can now also use the learnt weights for the right curve. Note, however, as a consequence of the general arrangement, the robot now cuts corners. This is a result of the fact that the predictive sensor field is some distance from the bottom of the camera image. Because steering necessarily always consists of a sequence of short, straight trajectories, the robot will always take shortcuts if the curves are too sharp and/or if the predictive sensor field is high up in the camera image.

In general we observed that the robot can learn the task fast even with a low learning rate as long as the track is shallow but needs higher rates to be able to follow the sharp track after about the same number of reflexes. If the same learning rate is chosen for all tracks then more reflexes are needed for the sharp track than for the shallow one.

Figure 6 shows the results for two control experiments with a shallow left and an increasingly sharp right curve

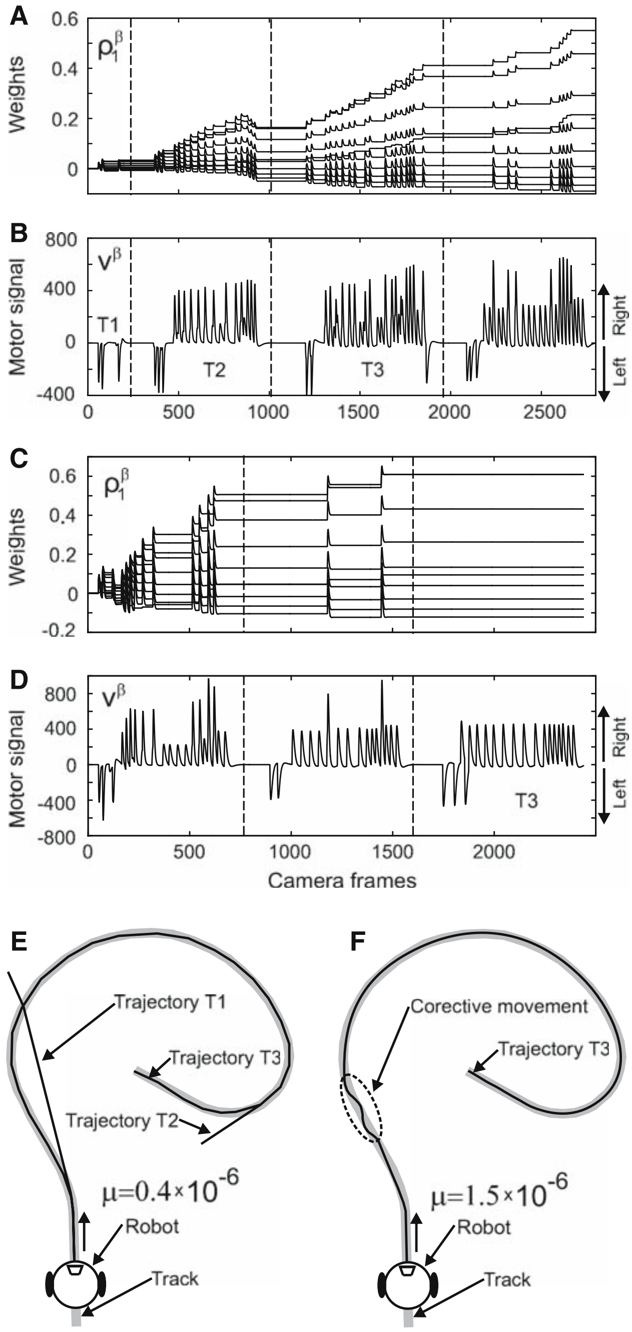

Fig. 6 Results of the driving robot experiment using the simple architecture (see Fig. 1c) on a spiral track. **a, b** Results for a learning rate of $\mu = 0.4 \times 10^{-6}$: **a** connection weights ρ_1^β, **b** motor output v^β. **c, d** Results for a learning rate of $\mu = 1.5 \times 10^{-6}$: **c** connection weights ρ_1^β, **d** motor output v^β. **e–f** Spiral track and robot trajectories belonging to the different learning rates used in Fig. 6. **e** Ongoing learning with rate $\mu = 0.4 \times 10^{-6}$, where we show trajectories T1, T2, and T3 during learning. Note, learning has not yet finished after T3, but improves gradually towards a smooth trajectory. **f** Final stage T3 reached after two learning trajectories (not shown) when using the higher learning rate of $\mu = 1.5 \times 10^{-6}$. In this case we find weight stabilisation after two trials (Fig. 6c), but the learnt weights will lead to too strong reactions for shallow curves which are compensated by corrective movements (see near the start of the trajectory)

(see Fig. 6e). The connection weights ρ_1^β (Fig. 6a) and motor output v^β (Fig. 6b) of four learning trials (separated by dashed lines) are shown for a relatively low learning rate $\mu = 0.4 \times 10^{-6}$. At the beginning, the low learning rate prevents the robot from even following the very shallow left curve (see trajectory T1 in Fig. 6e). In the second trial, the robot succeeded for the left curve and the beginning of the right curve but the learnt steering reaction was still not sufficient to allow it to follow the sharper parts of the right curve at the end of the spiral (see trajectory T2 in Fig. 6e). In the third learning trial the robot succeeded to follow the whole trajectory completely (see trajectory T3 in Fig. 6e) but still most of the time a mix of predictive and reflexive (large peaks) steering reactions occurred. The robot continued to improve its steering reactions in the fourth trial (trajectory not shown, but see video of whole experiment spiral-low.mpg) where one can see more nonreflexive reactions (smaller peaks) and less predictive + reflexive reactions than in the third trial. As expected from the linearity of our learning rule, in the right curve the system can use the weights learnt during the left curve up to the point where the right curvature remains below the left curvature (three leftward reactions and then two rightward reactions in the fourth trial) after which weights will continue to grow (large peaks). However, learning is not finished at this stage and would need more trials until the weights finally stabilise.

To speed up the learning process a higher learning rate of $\mu = 1.5 \times 10^{-6}$ was used; three learning trials are presented in Fig. 6c, d. In this case, the robot was already able to stay on the line during the first learning trial (trajectories not shown but see video: spiral-high.mpg) but still more predictive + reflexive (large peaks) than nonreflexive steering reactions occurred (see Fig. 6d). In the second trial only two predictive + reflexive reactions occurred, whereas in the last trial only nonreflexive steering reactions occurred and the weights did not change anymore. When we use the final weights learnt with the sharp curve to drive along the shallow left curve in a third trial the robot oversteers slightly to the left curve and then makes a right–left–right corrective movement, however, without triggering reflexes, in order to remain on the line (see trajectory T3 in Fig. 6f).

We also carried out an experiment to see how the robot behaves on a difficult track with different kinds of curvatures (Fig. 7c). The total length of track was approximately 14.5 m. The connection weights and motor output are shown in Fig. 7a, b. The robot had three reflexes at the beginning (Fig. 7a, see the arrows in c) while turning to the right and then the reflex input was not triggered until it approached the crossing point, where the robot turned to the right (see trajectory in Fig. 7c) and the reflex was triggered twice more. The reflex was then not triggered again and the weights stopped changing. When the robot approached the crossing point for

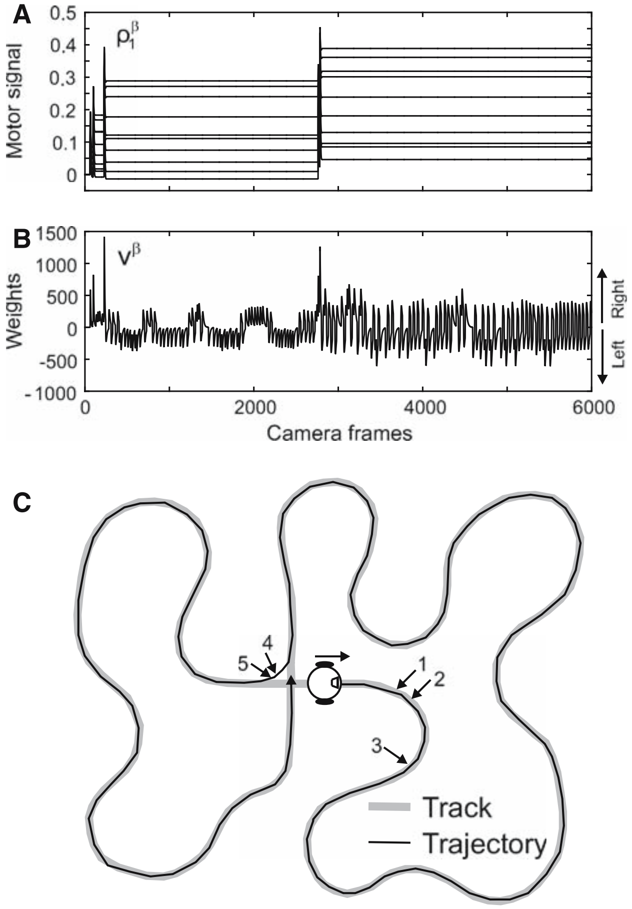

Fig. 7 Results of the driving robot experiment using the simple architecture (see Fig. 1c) on a maze track: **a** connection weights ρ_1^β, **b** motor output v^β. The learning rate was $\mu = 3 \times 10^{-6}$. The weights stabilised after five reflexes. **c** The maze track and the robot's driving trajectory for the first loop

the second time it went straight and for the third time (trajectory not shown) it turned to the left (see video maze.mpg). In general we obtained the same results as on the spiral track where the robot used the final weights learnt for the sharpest curve and oversteers slightly when driving on the shallower curves. Note, as the robot does not use any assumptions about track smoothness (similar to a known Gestalt law), for the machine both solutions, driving straight or turning, are equivalent at the crossing point in the centre of the track and the selection of a certain behaviour only depends on the status of its sensor inputs.

3.2 Statistical evaluation

From these experiments it became clear that the system performs online (and not batch) learning. Hence the most critical parameter affecting the convergence of learning is the way in which the instantaneous behaviour will influence, or rather generate, the next learning experience. Ultimately this is given by the sequence of viewing angles that the robot creates due to its own driving. Therefore, investigation of the influence of the viewing angle on the learning should provide the most relevant information about the robustness of this system. Other relevant parameters are the learning rate and the relative placement of the sensor fields.

Thus, to investigate the robustness against these parameters we used a simulation and performed a set of more than 1000 experiments in which we let the simulated robot learn to follow left-right tracks with angles of 20°, 45° and 90° (see Fig. 8a). The total length of the tracks was 360 points while its thickness was one point. The radius of the robot was $r = 20$ points and the positions of the sensor inputs $x_{0,1}^{L,R}$ (1x1 point) were defined as shown in Fig. 8b. We used the neuronal setup as presented in Fig. 1c. The output of the neuron v^β modified by the transformation function $S^{x,y}$ was used to change the position of the robot in the environment. It was calculated according to the following equations:

$$S_t^x = \mathrm{Sp}_t^x + r\cos(\alpha_t), \tag{4}$$

$$S_t^y = \mathrm{Sp}_t^y + r\sin(\alpha_t), \quad \text{where} \tag{5}$$

$$\mathrm{Sp}_t^x = \mathrm{Sp}_{t-1}^x + (1 - 0.001\,|v^\beta|)\cos(\alpha_t), \tag{6}$$

$$\mathrm{Sp}_t^y = \mathrm{Sp}_{t-1}^y + (1 - 0.001\,|v^\beta|)\sin(\alpha_t), \quad \text{and} \tag{7}$$

$$\alpha_t = \alpha_{t-1} - 0.01\,v^\beta, \quad t = 0, \ldots, N, \tag{8}$$

where α_0 is the angle in radians of the robot's starting position (Fig. 8c). We used a filter bank of 10 filters for the inputs $x_{0,1}^{L,R}$, given by the parameters $f_0 = 0.25$ for x_0 and $f_0 = 0.5/k$, $k = 1, \ldots, 10$, for x_1. The damping parameters of all the filters were $Q = 0.6$.

To evaluate the robot's performance we defined three (AND-connected) conditions to measure success:

1. The correlation coefficient between the robot's trajectory and the whole track is > 0.90.
2. The reflex is not triggered in three consecutive trials after the connection weights have stopped changing.
3. The robot completed the task within 20 trials (20 full tracks).

If these three conditions are not fulfilled at the same time then we count the experiment as a failure. Results demonstrating the influence of the robot's angle at the starting position are presented in Fig. 8d. We plot the success rate in 1000 experiments and the average number of reflexes (NR) needed to accomplish the task (in successful experiments) against the variance of the distribution of the starting angle σ_α^2. Success slightly decreases as we increase the variance of the starting angle distribution σ_α^2, but we still get high levels of performance (success rate $0.92 < \text{succ} \leq 0.99$ for all tracks). More reflexes are needed to accomplish the task if σ_α^2 is increased. Also, as expected, more reflexes are required

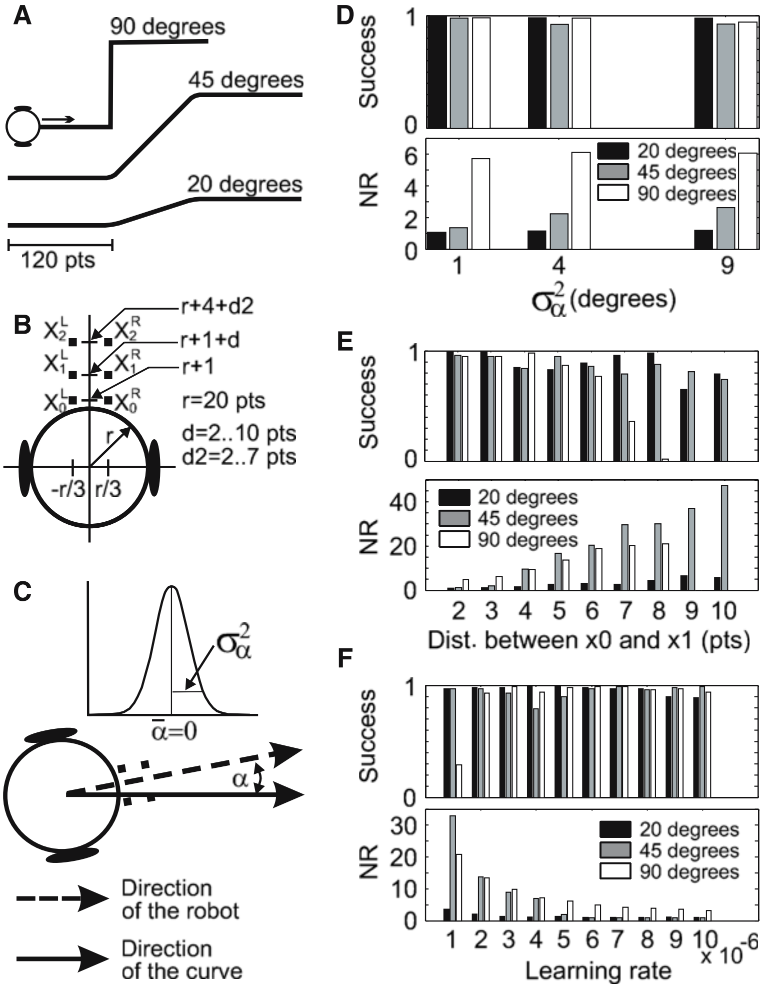

Fig. 8 **a–c)** Setup of the simulation experiment. **a** Tracks with 90°, 45°, and 20°. **b** Positions of the input signals $x_{0,1,2}^{L,R}$. **c** Angle α of the robot at its starting position, given by the deviation from the direction of the track. A Gaussian distribution of α was used with mean $\overline{\alpha}$ of zero and different variances σ_α^2. **d–f** Results of the simulation experiments using simple neuronal setup. **d** Success in 1000 experiments and average number of reflexes (NR) needed to accomplish the task within successful experiments are plotted against the variance σ_α^2; the learning rate was $\mu = 5 \times 10^{-6}$ and distance between x_1 and x_0 was $d = 3$. **e** Success in 100 experiments and average NR plotted against the distance between x_1 and x_0; the learning rate was $\mu = 5 \times 10^{-6}$ and the variance was $\sigma_\alpha^2 = 4$. **f** Success in 100 experiments and average NR plotted against the learning rate; the variance was $\sigma_\alpha^2 = 4$ and the distance between x_1 and x_0 was $d = 3$

for the sharp track than for the shallower ones. The results of 100 experiments for different positions of the predictor sensor x_1 are shown in Fig. 8e. The success rate decreases if the distance between the inputs becomes larger for the sharp track whereas for the shallow and intermediate track the decrease is less significant when the distance is very large ($d = 9/10$). The number of necessary reflexes (NR) increases as the distance between x_1 and x_0 becomes larger. This is due to the weight change curve of the ICO learning rule (Porr and Wörgötter 2006). If the inputs are spaced further apart in time then the correlations are weaker, the connection weights do not change so fast, and more repetitions are needed to complete learning. Due to this the robot never succeeded in steering along the sharp track within 20 trials when the distance between x_1 and x_0 was > 8. We also investigated the influence of the learning rate; the results of 100 experiments are presented in Fig. 8f. The learning rate does not affect performance except for the sharp track. When the learning rate is relatively low the robot does not succeed in steering along the curve within 20 trials. As expected we find that, with a higher learning rate, fewer reflexes are needed to complete the task, because the weights grow faster and the task is learnt sooner.

3.3 Chained architectures

3.3.1 Open-loop case

Two types of chained architectures were developed by the modification of the simple neuronal setup and were simulated in the open-loop case before applying them in the line-following task (closed-loop case). The neuronal setup of the first type of chained architecture, called the *linear chain*, is presented in Fig. 9a. There is one reflex input x_0 and two predictive inputs x_1 and x_2. The output v^β is used as the reflex input of the neuron γ. The weights $\rho_0^{\beta,\gamma}$ are set to a fixed value 1; all other weights are initially zero. The second type of chained architecture (Fig. 9d) is called a *honeycomb chain* due to its shape. The output $v^{\beta,1}$ is used as the reflex input of the neuron γ and the output $v^{\beta,2}$ as its predictive input. Note, that the output $v^{\beta,2}$, similarly to inputs x_1 and x_2, is fed into a filterbank h of different filters as indicated by the dashed lines in Fig. 2b. The output v^{γ} is calculated by

$$v^{\gamma} = \rho_0^{\gamma} v^{\beta,1} + \rho_1^{\gamma} u^{\beta,2}, \tag{9}$$

where $u^{\beta,2} = h \times v^{\beta,2}$ is a temporal convolution of the output $v^{\beta,2}$ with a resonator h. The resonator filters h_k for $v^{\beta,2}$ are determined by the parameters $f_{1,k} = 2.5/\text{kHz}$, $k = 1, \ldots, 10$ for the filterbank h, and the damping parameter was set at $Q = 0.6$. The weights $\rho_0^{\beta,1}$ and ρ_0^{γ} are set to a fixed value 1; all other weights were initially zero. The connection weight $\rho_0^{\beta,2}$ is given by $\rho_0^{\beta,2} = \sum_{k=1}^{10} \rho_{1,k}^{\beta,1}$, where k denotes the number of the filter in the filter bank. We note that the two architectures are identical if we set $\rho_0^{\beta,2} = 0$ and $\rho_1^{\beta,2} = 1$.

Inputs for the open-loop case were generated as follows. Input x_2 occurs 20 time steps earlier than input x_0 with a variability of up to ±5 time steps and x_1 occurs 10 time steps earlier than x_0 with the same variability. This impulse sequence was repeated every 50 time steps.

Simulation results for the linear chain (Fig. 9a) are presented in Fig. 9b, c. The variability in the pulse sequences leads to uneven growth. In the open-loop case we have to enforce weight stabilisation by setting the inputs x_0 to zero at some points. This was done whenever the growing input weights ρ_1 at this neuron, summed over the whole filter bank, exceeded a threshold of 0.5 (see the legend of Fig. 9 for the equations).

Using this criterion, first the connection weights ρ_1^{β} stabilise and after some time the ρ_1^{γ} stop changing. The results for the honeycomb chain (Fig. 9d) are presented in Fig. 9e–g. In this situation first the connection weights $\rho_1^{\beta,1}$ stop changing and later both the weights $\rho_1^{\beta,2}$ and ρ_1^{γ} stabilise.

3.3.2 Closed-loop case

The physical and neuronal setups of the learning system for the chained architectures are presented in Fig. 10. The neuronal setup for the linear chain is presented in Fig. 10b and for the honeycomb chain in Fig. 10c. These are similar to those above (see Fig. 9a, d), only that we add left and right inputs with inverted signs before this signal finally arrives at neurons β.

These chained architectures were applied in the line-following task and results similar to those in the simulated open-loop case were obtained for both architectures. The results for the learning task using the linear chain (Fig. 10b) are presented in Fig. 11a–d and for the honeycomb chain (Fig. 10c) in Fig. 11e–h. In the first learning trial the motor signal (Fig. 11c) shows three leftward cumulative reflex + predictive reactions and two nonreflexive reactions, as well as two cumulative rightward reactions and three non reflexive reactions. Note, by chance in this trial the three leftward reflexes were elicited by triggering x_0^L, whereas the two rightward reflexes came from x_1^R. Hence the leftward reflexes were contributing to changes of ρ_1^{β} and ρ_1^{γ} (Fig. 11a, b) but not the rightward reflexes, which only contributed to the change of ρ_1^{γ}.

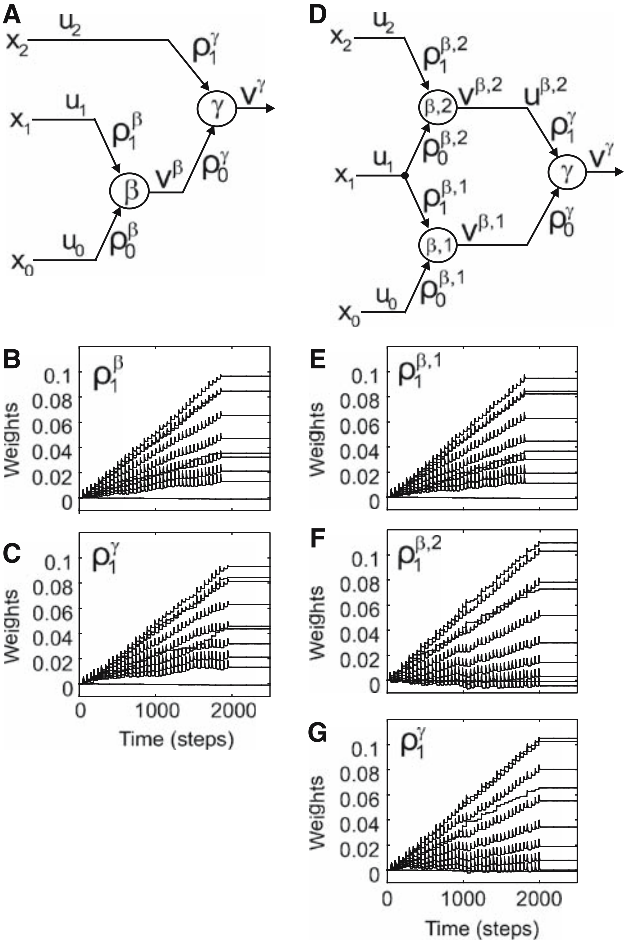

Fig. 9 Chained neuronal architectures (**a, d**) and simulation results for the open-loop case (**b, c, e–g**): **a** Linear chain and **d** honeycomb chain. The learning rate for both architectures was $\mu = 10^{-7}$. **b, c** Results for the linear chain (**a**) with connection weights ρ_1^{β} and ρ_1^{γ}; the weights ρ_1^{β} stop growing at the condition $x_0 = 0$ while the ρ_1^{γ} stop growing when $x_1 = 0$. We set $x_0 = 0$ when the sum of weights over all 10 filters was $\sum_{k=1}^{10} \rho_{1,k}^{\beta} \geq 0.5$ and $x_1 = 0$ when $\sum_{k=1}^{10} \rho_{1,k}^{\gamma} \geq 0.5$. **e–g** Results for the honeycomb chain (**d**) with connection weights $\rho_1^{\beta,1}$, $\rho_1^{\beta,2}$, and ρ_1^{γ}. The weights $\rho_1^{\beta,1}$ stop growing at the condition $x_0 = 0$, $\rho_1^{\beta,2}$ while the ρ_1^{γ} stop growing when $x_1 = 0$. We set $x_0 = 0$ when the sum of weights over all 10 filters was $\sum_{k=1}^{10} \rho_{1,k}^{\beta,1} \geq 0.5$ and $x_1 = 0$ when $\sum_{k=1}^{10} \rho_{1,k}^{\beta,2} \geq 0.5$

In the second trial only nonreflexive leftward and rightward steering signals occurred and the reflex was not triggered anymore. The driving trajectories are shown in Fig. 11d and in the video linear-chain.mpg. The weights at a certain neuron stabilise as soon as their corresponding reflex input remains silent. For the linear chain (Fig. 11a–d) this happens earlier for ρ_1^{β}, where x_0 becomes zero after about 150 camera frames, and later for ρ_1^{γ}, because its reflex input

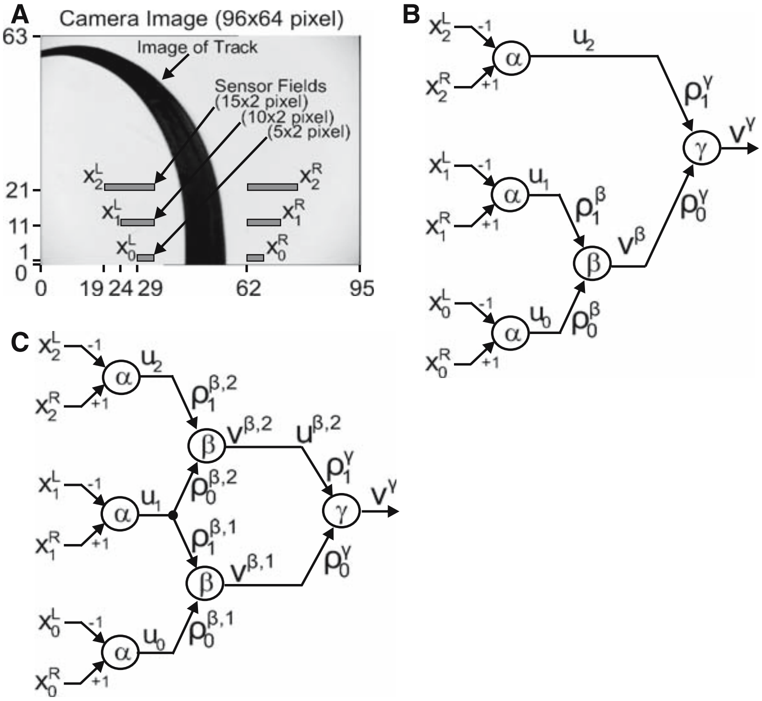

Fig. 10 The physical (**a**) and neuronal (**b, c**) setups of the chained learning system for the closed-loop case. **a** Camera image with sensor fields marked by $x_{1,2}^{L,R}$ and $x_0^{L,R}$. **b** Linear chain and **c** honeycomb chain

v^β remains active for longer. Essentially the same is true for the honeycomb chain (Fig. 11e–h). Here $\rho_1^{\beta,1}$ stops growing first, which becomes the same reflex input u_0 as ρ_1^β in the linear chain. The convergence of the weights $\rho_1^{\beta,2}$ is controlled by the reflex input u_1, which also contributes to the signal $v^{\beta,1}$, the reflex input to neuron γ. Hence the weights $\rho_1^{\beta,2}$ and ρ_1^γ behave in the same way and stabilise later (similar to ρ_1^γ in the linear chain).

3.4 Statistical evaluation

We also carried out simulations using chained architectures in order to make a comparison with the simple setup. The simulation setup for the chained architectures is shown in Fig. 8b. The positions of the sensor fields $x_1^{L,R}$ and $x_0^{L,R}$ were fixed (3 pts) and we only varied the position of the sensor fields $x_2^{L,R}$. The influence of the robot's starting angle are presented in Fig. 12a, b. We plot the success rate in 1000 experiments and the average number of reflexes (NR) needed to accomplish the task (in successful experiments) against the variance of the distribution of the starting angle σ_α^2. We obtained similar results using the linear chain (Fig. 12a) as with the simple architecture; success is slightly decreasing and more reflexes are needed to accomplish the task if we increase the variance σ_α^2. We get a slightly reduced performance compared to the simple setup (success rate $0.86 < \text{succ} < 0.96$ for all tracks). Also, as for the simple setup, more reflexes are required for the sharp track compared to the shallower ones. For the honeycomb chain (Fig. 12b) the performance was again lower, with a success rate of $0.71 < \text{succ} \leq 0.94$ for the shallow and intermediate track where for the sharp track we got very low performance (success rate $\text{succ} < 0.1$). This is due to the fact that the honeycomb chain architecture is sensitive to the position of the sensor fields. We plot the results of 100 experiments for different positions of the predictor sensor x_2 (keeping the positions of x_1 and x_0 fixed) in Fig. 12d. Here we can see that we get the best performance for the shallow and sharp track when the distance between x_2 and x_1 is $d_2 = 5$ pts (success rate $0.70 \leq \text{succ} \leq 0.96$ for all tracks), where for the intermediate track the importance of the position of the sensor fields is not significant (except for the smallest distance between x_2 and x_1 of $d_2 = 2$ pts). For the linear chain setup (Fig. 12c) we obtained the same results as for the simple one. The success rate decreases as the distance between the inputs becomes larger only for the sharp track, whereas for the shallow and intermediate track the decrease is not significant. We also observed that the number of necessary reflexes (see Fig. 12c–d) increased if the distance between x_1 and x_0 became larger except for very small distances between x_2 and x_1 when using honeycomb chain setup (Fig. 12d).

We can summarise that better performance is obtained with the simple setup compared to the chained architectures. The performance does not depend crucially on the starting angle. It decreases only slightly if the variance of starting angle position increases. In general we observed that only for the honeycomb chain architecture does performance depend on the position of the sensor fields (the distance between the sensor fields). The learning rate does also not affect the

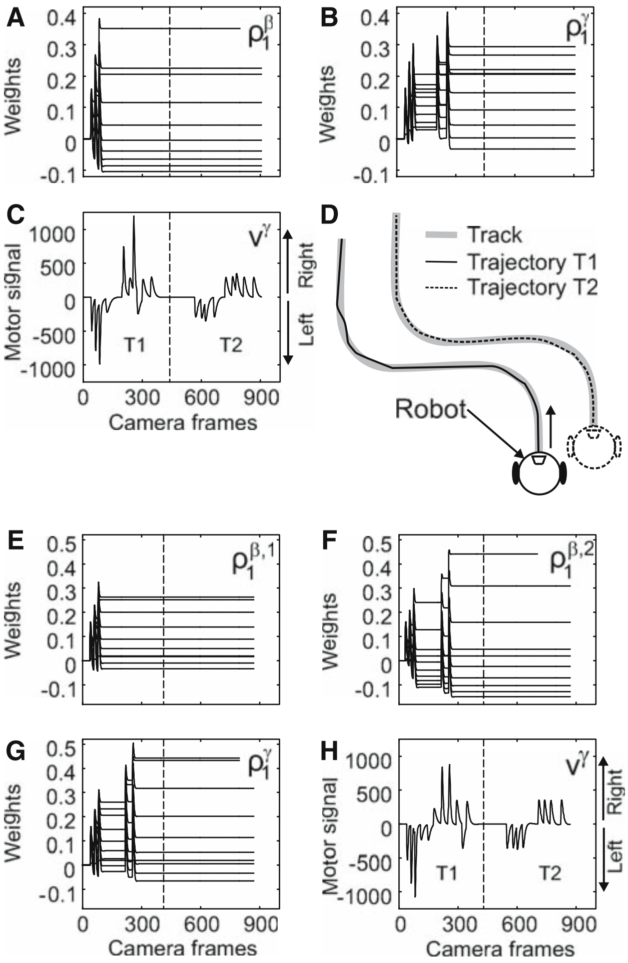

Fig. 11 Results of the driving robot experiment using chained architectures. The learning rate for both experiments was $\mu = 2.5 \times 10^{-6}$. **a–d** Results for the linear chain (see Fig. 10b): **a, b** the connection weights ρ_1^β and ρ_1^γ, **c** motor output v^γ, and **d** the driving trajectories. Trajectory T1 during, and T2 after, learning. **e–h** Results for the honeycomb chain (see Fig. 10c): **e–g** connection weights $\rho_1^{\beta,1}$, $\rho_1^{\beta,2}$, and ρ_1^γ; **h** motor output v^γ. The trajectories are similar to the previous experiment (**d**) and are not shown

performance itself. The robot only needs more reflexes to learn the task if we use lower learning rates.

4 Simple versus chained architectures

Previously we summarised that with the simple setup we got better performance compared to the chained architectures. This is true only for cases where we have good input correlations (small distances between the inputs) in the simple setup. The performance decreased if the distance between the inputs was very large (see Fig. 8e) for the shallow and intermediate track and the robot never managed to steer along the sharp curve when the distance between inputs was > 8. However, the robot managed to steer along the sharp curve when the chained architectures were used (see Fig. 12c, d) where the distance between the inputs x_2 and x_1 was >5 and between x_1 and x_0 was 3 (the total distance between x_2 and x_0 was > 8). To test the hypothesis that chained architectures are advantageous for bad correlations because of sparse inputs we carried out an experiment in which we compared the performance of all three architectures on the intermediate track (45°). The setup of the input configuration for the simple architecture is shown in Fig. 13a and for the chained architectures in Fig. 13b. The distance between the inputs x_1 and x_0 in the simple setup was 15 pts and in the chained architectures it was 8 between the inputs x_2 and x_1 and 7 between x_1 and x_0 (the total distance between x_2 and x_0 was 15 pts). A comparison between all three architectures is presented in Fig. 13c, d where we plot the success rate in 500 experiments (Fig. 13c) and the average number of trials (NT) within successful experiments together with confidence intervals (95%) needed to accomplish the task (see Fig. 13d). From these results we can conclude that chained architectures indeed perform better (with a success rate for the linear chain of 0.87 and for the honeycomb chain of 0.92) whereas for the simple architecture we obtained a success rate of only 0.57 (see Fig. 13c). We also needed fewer trials to complete learning when using the chained architectures compared to the simple setup (see Fig. 13d).

5 Discussion

In this study we have introduced a specific closed-loop robotics system that can adaptively improve its line-following behaviour, performing reflex-avoidance learning by ways of replacing late responses to sensor fields at the base of a camera image with earlier ones triggered by sensors higher up in the field of vision. A new learning rule (ICO) was employed, which is able to correlate sequences of temporal events and the system has been tested in a restricted set of scenarios far less complex than those in a real-world navigation task. Thus, the system has been specifically designed for this task and cannot easily be compared with more-general navigation systems (see Sect. 5.3 below). These restrictions, however, are justified by the focus of our study, which is twofold: (1) we wanted to investigate the properties of chained ICO learning, and (2) we were interested in finding out whether chained learning could be beneficial in cases of sparse and noisy inputs. Note, more-general applications of single-module (no chaining) ICO learning can be found in (Porr and Wörgötter 2006, 2003a, b; Manoonpong et al. 2007). These studies should support the general versatility of this type of learning. In the following we would like to discuss how the open- and closed-loop situation compares to biological and other artificial systems, compare our approach to other approaches

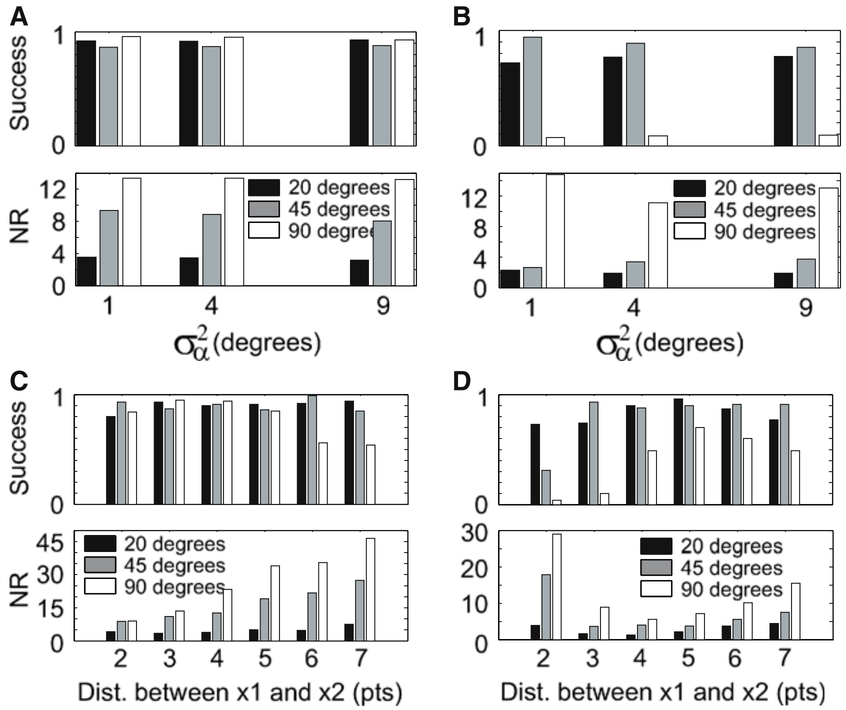

Fig. 12 **a–d** Results of the simulation experiments using chained architectures. **a, b** Success in 1000 experiments and average number of reflexes (NR) at the motor output neuron γ needed to accomplish the task within successful trials plotted against the variance σ_α^2: **a** linear chain, **b** honeycomb chain. The learning rate was $\mu = 5 \times 10^{-6}$ and the distance between x_1 and x_0 and between x_2 and x_1 was $d = d_2 = 3$. **c, d** Success in 100 experiments and average NR plotted against the distance between x_2 and x_1: **c** linear chain, **d** honeycomb chain. The distance between x_1 and x_0 was fixed and was $d = 3$. The learning rate was $\mu = 5 \times 10^{-6}$ and the variance was $\sigma_\alpha^2 = 4$

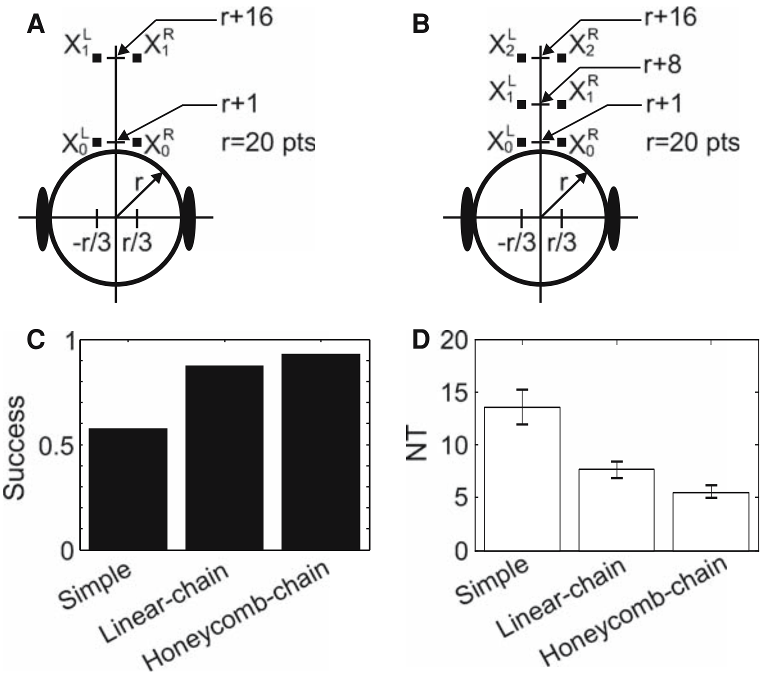

Fig. 13 **a, b** Setup of the simulation experiment. **a** Simple setup. Positions of the input signals $x_{0,1}^{L,R}$. **b** Chained architectures. Positions of the input signals $x_{0,1,2}^{L,R}$. **c, d** Results of the simulation experiments using different neuronal setups on the middle track (45°). **c** Success in 500 experiments. **d** Average number and confidence intervals (95%) of trials (NT) needed to accomplish the task within the successful experiments. The learning rate for all the experiments was $\mu = 5 \times 10^{-6}$. The distance between x_1 and x_0 in the simple setup was 15 pts whereas the distances between x_1 and x_0 and between x_2 and x_1 in chained architectures were 7 and 8 pts, respectively

for steering control, and discuss where there are relations to some aspects of reinforcement learning.

5.1 Relation of ICO learning to synaptic plasticity in real neurons

The ICO learning rule has been chosen because of its robust convergence properties (Porr and Wörgötter 2006) even with high learning rates. ICO learning changes its weights by correlating inputs only. This can be interpreted as heterosynaptic plasticity or as modulatory plasticity. In biological systems, pure heterosynaptic learning is only found at a few specialised synapses (mossy fibre, amygdala, Humeau et al. 2003; Tsukamoto et al. 2003), where the mossy fiber synapse between dentate gyrus and CA3 in the hippocampus can indeed create fast and strong changes similar to those induced by ICO learning with a high learning rate. More often, however, heterosynaptic influences are thought to be mainly modulatory (Kelley 1999; Ikeda et al. 2003; Bailey et al. 2000; Jay 2003). Here we are not really concerned with the possible biological implications of such a learning rule (see Wörgötter and Porr 2005; Porr and Wörgötter 2006 for a more-detailed discussion). Instead we have used it as a tool to employ fast learning in a closed-loop scenario. This property is visible when learning succeeds after the first trial in keeping the robot on track for an intermediately steep track (Fig. 5a–d), while it does not follow the line if only the reflex alone is employed (see video control.mpg). Hence, during the first learning trial synaptic weights already adjust quickly and, in turn, immediately influence the output leading to successful behaviour. This behaviour is generically observed for the ICO rule, which thereby approaches the limit of one-shot learning in stable behavioural domain (Porr and Wörgötter 2006), provided the input correlations are robust enough.

5.2 Closed-loop context: combining control and learning

Biological systems generally operate in close conjunction with their environment. This so-called ecological embedding has was discussed by theoreticians very early as also essential for autonomous artificial agents Ashby 1956; McFarland 1971; Wiener 1961. On the more-practical side W. G. Walter was probably the first to create an operational, autonomous, cybernetic control system when he built his two robots Elmer and Elsie. These machines could already perform homing as well as different forms of photokinesis (Walter 1950). In the following the ecological perspective had been widened most notably by the work of Braitenberg (1984) on his *vehicles* and for invertebrates by Webb (2002).

In most of the older work typical feedback loop control systems that do not adapt but instead react to a stimulus by ways of reflex-like behaviour were built. Stable feedback loop control is in itself a difficult problem, in particular when there are multiple inputs and outputs. It is however known that even very simple animals can learn and adapt to new situations. Hence we are now faced with the augmented problem of how to combine control with learning in a stable way. Specifically we are confronted with the question of how animals arrive at useful, reproducible and, hence, stable behavioural patterns, while they are at the same time able to learn something new. Recently Verschure suggested that such systems should contain several layers for control and learning: at the bottom a reactive layer performs pure reflex-based control, an adaptive layer above performs predictive learning much in the sense of classical or operant conditioning, and finally a top contextual layer carries out higher-level adaptation (the DAC architecture, Verschure and Althaus 2003). In our study we are concerned with the first two layers only.

There is another class of learning setups based on feed back-error learning (FEL, Gomi and Kawato 1993; Nakanishi and Schaal 2004), which appear to be related to closed-loop ICO. However, in contrast to ICO learning FEL does not use additional predictive inputs $x_1, x_2, \ldots$ to compensate for a disturbance. It rather improves the feedback loop itself by using the signals that are available to the (late) feedback system. A simple example is a feedback loop that is set up as an overdamped system (PI controller) so that the reaction of the loop to a disturbance or a change in the setpoint leads to a low-pass-filtered impulse response of the system. With the help of FEL the reaction can be made faster by adding an adaptive controller that receives a copy of the disturbance itself or the output of the feedback controller. Because the system is overdamped, FEL learns to become the derivative of the disturbance. In other words, FEL adaptively learns to add the D stage to a PI controller. ICO or ISO learning, however, is fundamentally different because it uses the derivative as a predictor to learn *another* predictive input, which is then used to eliminate the disturbance and eventually eliminates the feedback loop itself. FEL on the other hand does not replace the feedback loop by a forward controller but rather improves the performance of the feedback controller itself.

In all such architectures, however, one must ask how, in the process of learning, synaptic weights are stabilised in conjunction with behavioural success. Stability in our approach rests on the assumption that the reflex eliciting signal (x_0) really represents an error signal. Hence, ICO learning stabilises as soon as this error signal is eliminated, as has been rigorously shown in Porr and Wörgötter (2006). On the behavioural side, however, this means that the reflex has been functionally eliminated and has now been successfully replaced by an earlier anticipatory action. This property enables control of the homeostasis of learning and behaviour at the same time, which is more difficult to achieve with most other architectures.

5.3 Comparison to other approaches on navigation learning

Wyss et al. (2004) used a neural model to control a robot that learned to follow accurately lines drawn on the floor using visual information provided by a camera. For this task they used a form of reinforcement learning where the sensory input was mapped to the output. The reinforcement signal was derived by computing the temporal derivative of the summed activity from a small receptive field in the lower centre part of the camera image. Compared to our approach this learning algorithm takes a relatively long time and many learning experiences (a general drawback of reinforcement learning approaches). In the work of Pomerleau (1996) an autonomous land vehicle in a neural network (ALVINN) system learnt to steer a vehicle in response to visual input from a forward-facing camera. ALVINN uses a single hidden-layer feedforward neural network that applied the back-propagation learning algorithm to learn an appropriate behaviour according to human reactions. This differs from our approach because it is supervised learning and the learning also does not take place in a complete closed-loop setting since the output of the learner is not used to drive the car. This differs from a recent study by McKinstry et al. (2006), who were able to close the loop and derive path-following behaviour in a robot driven by a complex multilayer neuronal system supposed to mimic parts of the cerebellar system. The system learns, as in our case, reflex avoidance. This is done by the simulated neural system containing 27,688 neuronal units and ≈1.6 million synaptic connections that adapt following a delayed eligibility trace learning rule. Synaptic weights develop at several stages in the network, but it appears that this type of learning will not lead to their final stabilisation. Land (2001) analysed which part of the road scene is needed for steering control. He observed that, when only the top part of the simulated road image was presented as visual input, the driving trajectory matched the curvature well, but lane-keeping performance was poor (corresponding to predictive control), and when only the bottom part of the scene was visible, lane keeping was better but steering became unstable and jerky (corresponding to reflexive control). In our approach sensory inputs are below the horizontal centre line of the image as, if the sensors are located at the top part of image, the robot takes shortcuts, since we do not have delays in the motor actions with respect to sensory input (as can also be observed on sharp curvatures, see Fig. 5j). Thus, in our system we have reactive (reflexive) and proactive (in our system predictive) control compared to the reactive and real predictive control in the study of Land (2001).

In this study we were concerned with designing simple chained architectures of our learning modules. This was inspired by second-order conditioning in animals (Rescorla 1980; Gewirtz and Davis 2000) and humans (Jara et al. 2006). Secondary conditioning requires a similar situation where the primary correlation between conditioned (early, CS) and unconditioned (late, US) stimulus is first learned and then in a second learning stage replaced by a newly learned correlation between secondary conditioned stimulus (earlier yet) and CS (resp. US). This situation is conceptually similar to our chained learning units and the same problems, for example, less-reliable correlation patterns, arise in both situations.

5.4 Some relations to reinforcement learning

Our approach is to some degree related to reinforcement learning, not so much to machine learning methods like Q-learning (Watkins 1989; Watkins and Dayan 1992), but rather to actor–critic loop architectures (Witten 1977; Barto et al. 1983; Barto 1995), which have been employed in simulated neural systems. Indeed, if one uses the x_0 signal as a reward one can create a structural similarity between some of these algorithms and our ICO rule (for a detailed comparison see Kolodziejski et al. 2007). Also, we note that the strict state and action space tiling used in traditional Q-learning approaches has in some approaches been replaced by more-adaptive self-defining processes, which span the state and action space through exploration (Jodogne et al. 2005; Agostini and Celaya 2004), making these algorithms more compatible with neuronal architectures.

Indeed, some actor–critic algorithms have also been used to guide the learning of biologically inspired agents (Montague et al. 1995; Suri and Schultz 1998; Schultz and Suri 2001; Niv et al. 2002) but—to our knowledge—it has not been attempted to chain actor–critic loops so far. Apart from the fact that no generic recipe exists, the problem may be even more fundamental. Actor–critic architectures usually rely (in their Critic) on the TD algorithm (Sutton 1988; Sutton and Barto 1998) to assess the value of an action of the actor. The prediction error δ in TD learning equals zero as soon as the output v accurately estimates the future expected reward $r(t+1)$ using: $\delta(t) = r(t+1) + v(t+1) - v(t)$. To fulfil this convergence condition, the output v needs to take on a certain value (the output control). In any single-loop architecture, outputs will be fairly directly transferred to inputs by ways of the environment (e.g. Fig. 3). In a nested or chained loop, however, a problem may arise. To guarantee the convergence of each individual stage of the chain its output needs to be directly conveyed backward to compare it to the reward, which, necessarily is an input to the regarded stage. Effectively this amounts to some kind of error back-propagation, a commonly used principle in artificial neural networks (McClelland et al. 1987), but hard to justify in biological networks, where the role of internal feedback does not seem to be related to any error back-propagation mechanism. Architectures based on our correlation-based learning rule(s) perform strict input control, because they converge as soon as the error signal of the reflex, x_0, equals zero, regardless of the value of the output. This condition, hence, does not require error back-propagation and may prove to be easier to handle for the design of more-complex nested of chained loops as compared to actor–critic architectures.

Hence, one starting point for this study was the assumption that input control should allow the design of more-complex structures with predictable stability properties. Therefore, here we have for the first time implemented a simple-layered structure and obtained stable behaviour in a closed-loop scenario. While the two chained architectures are still rather simple, we believe that this is nevertheless an important step towards more-advanced networks of correlation-based learning units. Furthermore, we conclude that chained architectures can be employed to obtain better behavioural performance compared to the simple architecture where learning fails because of weak correlations.

Acknowledgements Prof. F. Wörgötter acknowledges support of the European Project DRIVSCO. T. Kulvicius thanks Dr. M. Tamosiunaite for fruitful discussions.

References

Agostini E, Celaya A (2004) Trajectory tracking control of a rotational joint using feature-based categorization learning. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, IEEE, Sendai, Japan

Ashby WR (1956) An introduction to cybernetics. Methnen, London

Bailey CH, Giustetto M, Huang YY, Hawkins RD, Kandel ER (2000) Is heterosynaptic modulation essential for stabilizing Hebbian plasticity and memory. Nat Rev Neurosci 1(1):11–20

Barto A (1995) Reinforcement learning in motor control. In: Arbib M (ed.) Handbook of brain theory and neural networks. MIT Press, Cambridge, pp 809–812

Barto AG, Sutton RS, Anderson CW (1983) Neuronlike elements that can solve difficult learning control problems. IEEE Trans Syst Man Cybern 13:835–846

Braitenberg V (1984) Vehicles: experiments in synthetic psychology. MIT Press, Cambridge

Gewirtz JC, Davis M (2000) Using pavlovian higher-order conditioning paradigms to investigate the neural substrates of emotional learning and memory. Learn Mem 7(5):257–266

Gomi H, Kawato M (1993) Neural network control for a closed-loop system using feedback-error-learning. Neural Netw 6(7):933–946

Humeau Y, Shaban H, Bissiere S, Luthi A (2003) Presynaptic induction of heterosynaptic associative plasticity in the mammalian brain. Nature 426(6968):841–845

Ikeda H, Akiyama G, Fujii Y, Minowa R, Koshikawa N, Cools A (2003) Role of AMPA and NMDA receptors in the nucleus accumbens shell in turning behaviour of rats: interaction with dopamine and receptors. Neuropharmacology 44:81–87

Jara E, Vila J, Maldonado A (2006) Second-order conditioning of human causal learning. Learn Motiv 37:230–246

Jay T (2003) Dopamine: a potential substrate for synaptic plasticity and memory mechanisms. Prog Neurobiol 69(6):375–390

Jodogne S, Scalzo F, Piater JH (2005) Task-driven learning of spatial combinations of visual features. In: Proceedings of the IEEE workshop on learning in computer vision and pattern recognition, IEEE, San Diego (CA, USA)

Kelley AE (1999) Functional specificity of ventral striatal compartments in appetitive behaviors. Ann NY Acad Sci 877:71–90

Klopf AH (1988) A neuronal model of classical conditioning. Psychobiology 16(2):85–123

Kolodziejski C, Wörgötter F, Porr B (2007) Mathematical properties of neuronal TD-rules and differential Hebbian learning: A comparison. Biol Cybern (submitted)

Kosco B (1986) Differential Hebbian learning. In: Denker JS (ed) Neural networks for computing: AIP Conference Proceedings, vol. 151. American Institute of Physics, New York

Land MF (2001) Does steering a car involve perception of the velocity flow field. In: Zeil JMZJ (ed) Motion vision—computational, neural, and ecological constraints, pp. 227–235

Manoonpong P, Geng T, Kulvicius T, Porr B, Wörgötter F (2007) Adaptive, fast walking in a biped robot under neuronal control and learning. PLoS Comput Biol 3(7):e134 doi:10.1371/journal.pcbi.0030,134

McClelland JL, Rumelhart DE, Hinton GE (1987) Parallel distributed processing, vol 1. MIT Press, Cambridge

McFarland DJ (1971) Feedback mechanisms in animal behaviour. Academic, London

McKinstry JL, Edelman GM, Krichmar JL (2006) A cerebellar model for predictive motor control tested in a brain-based device. Proc Natl Acad Sci USA 103(9):3387–3392

Montague PR, Dayan P, Person C, Sejnowski TJ (1995) Bee foraging in uncertain environments using predictive Hebbian learning. Nature 377:725–728

Nakanishi J, Schaal S (2004) Feedback error learning and nonlinear adaptive control. Neural Netw 17:1453–1465

Niv Y, Joel D, Meilijson I, Ruppin E (2002) Evolution of reinforcement learning in uncertain environments: a simple explanation for complex foraging behaviors. Adapt Behav 10(1):5–24

Pomerleau D (1996) Neural network vision for robot driving. In: Nayar S, Poggio T (eds) Early visual learning. Oxford University Press, New York, pp 161–181

Porr B, Wörgötter F (2003a) Isotropic sequence order learning. Neural Comp 15:831–864

Porr B, Wörgötter F (2003b) Isotropic sequence order learning in a closed loop behavioural system. R Soc Phil Trans Math Phys Eng Sci 361(1811):2225–2244

Porr B, Wörgötter F (2006) Strongly improved stability and faster convergence of temporal sequence learning by utilising input correlations only. Neural Comp 18(6):1380–1412

Porr B, Ferber C, Worgotter F (2003a) Iso-learning approximates a solution to the inverse controller problem in an unsupervised behavioural paradigm. Neural Comp 15:865–884

Porr B, von Ferber C, Wörgötter F (2003b) ISO-learning approximates a solution to the inverse-controller problem in an unsupervised behavioral paradigm. Neural Comp 15:865–884

Rescorla RA (1980) Pavlovian second-order conditioning: studies in associative learning. Erlbaum, Hillsdale

Schultz W, Suri RE (2001) Temporal difference model reproduces anticipatory neural activity. Neural Comp 13(4):841–862

Suri RE, Schultz W (1998) Learning of sequential movements by neural network model with dopamine-like reinforcement signal. Exp Brain Res 121:350–354

Sutton R, Barto A (1981) Towards a modern theory of adaptive networks: expectation and prediction. Psychol Rev 88:135–170

Sutton RS (1988) Learning to predict by the methods of temporal differences. Mach Learn 3:9–44

Sutton RS, Barto AG (1990) Time-derivative models of Pavlovian reinforcement. In: Gabriel M, Moore J (eds) Learning and computational neuroscience: foundation of adaptive networks. MIT Press, Cambridge

Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge

Tsukamoto M, Yasui T, Yamada MK, Nishiyama N, Matsuki N, Ikegaya Y (2003) Mossy fibre synaptic NMDA receptors trigger non-Hebbian long-term potentiation at entorhino-CA3 synapses in the rat. J Physiol 546(3):665–675

Verschure P, Althaus P (2003) A real-world rational agent: unifying old and new AI. Cogn Sci 27:561–590

Verschure P, Coolen A (1991) Adaptive fields: distributed representations of classically conditioned associations. Network 2:189–206

Walter WG (1950) An imitation of life. Sci Am 182:42–45

Watkins CJCH (1989) Learning from delayed rewards. PhD Thesis, University of Cambridge, Cambridge, England

Watkins CJCH, Dayan P (1992) Technical note: Q-Learning. Mach Learn 8:279–292

Webb B (2002) Robots in invertebrate neuroscience. Nature 417:359–363

Wiener N (1961) Cybernetics—or control and communication in the animal and the machine, 2nd edn. The MIT Press, Cambridge

Witten IH (1977) An adaptive optimal controller for discrete-time Markov environments. Inf Control 34:86–295

Wörgötter F, Porr B (2005) Temporal sequence learning for prediction and control - a review of different models and their relation to biological mechanisms. Neural Comp 17:245–319

Wyss R, König P, Verschure PFMJ (2004) Involving the motor system in decision making. Proc Biol Sci 271(Suppl 3):50–52