Project no.: **027657**

Project full title: **Perception, Action & Cognition through Learning of Object-Action Complexes**

Project Acronym: **PACOPLUS**

Deliverable no.: D6.2

Title of the deliverable:

Chaining learning architectures in a simple closed-loop behavioural context

(scientific publication submitted to Biological Cybernetics)

Contractual Date of Delivery to the CEC:	**31/03-07**	
Actual Date of Delivery to the CEC:	**21/03-07**	
Organisation name of lead contractor for this deliverable:	**BCCN**	
Author(s):	**BCCN**	
Participants(s):	**BCCN**	
Work package contributing to the deliverable:	**WP6**	
Nature:	**R/D**	
Version:	**1.1**	
Total number of pages:	**25**	
Start date of project:	**1st Feb. 2006**	**Duration: 48 months**

Projectco-funded by the European Commission within the Sixth Framework Programme (2002-2006)
Dissemination Level

PU	Public
PP	Restricted to other programme participants (including the Commission Services)
RE	Restricted to a group specified by the consortium (including the Commission Services)
CO	Confidential, only for members of the consortium (including the Commission Services)

Abstract:

In this study we apply temporal sequence learning (Porr and Wörgötter, 2006) in a closed loop behavioural system where a driving robot learns to follow a line. Here for the first time we introduce simple chained learning architectures, demonstrating that stable behaviour can also be obtained in such architectures. Moreover, results also suggest that chained architectures can be employed and better behavioural performance can be obtained as compared to simple architectures in cases where we have sparse inputs in time and learning normally fails because of weak correlations.

Keyword list: Chaining, Temporal Sequence Learning, Closed Loop

1 Introduction

Normally many sensor events, which follow each other in time, are associated to a real life situation. However, 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 later following ones. For example, it is useful to react to a heat radiation signal and not to the later following pain on having finally touched a hot surface. Many similar sequences of sensor events are encountered during the life time 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 predictive with respect to the corresponding near-senses (Verschure and Coolen, 1991). Conceptionally this type of learning is related to classical and/or operant conditioning (Sutton and Barto, 1981, 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: 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 as 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's 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 in the following way. After presenting our ICO-learning rule (Porr and Wörgötter, 2006) and its embedding into a closed loop scenario we will first show results with a simple architecture. By this we would like 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 of 18 cm) two-wheeled Rug Warrior Pro driving robot for investigation which is shown in Fig. 1 A. Fig. 1 B 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 more at 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 implement only a crude, abrupt, and aversive steering reflex as soon as the image of the track moves over one of these near-sensor fields. As a consequence the robot will be forced

back to a situation where the track will remain mostly in the centre of the image. The learning goal is to increase the synaptic weights of the far-sensor fields in an appropriate way such that earlier, predictive, and smoother steering reactions will be elicited leading to the situation that the near-sensor fields will never be triggered again (hence avoiding the initial reflex).

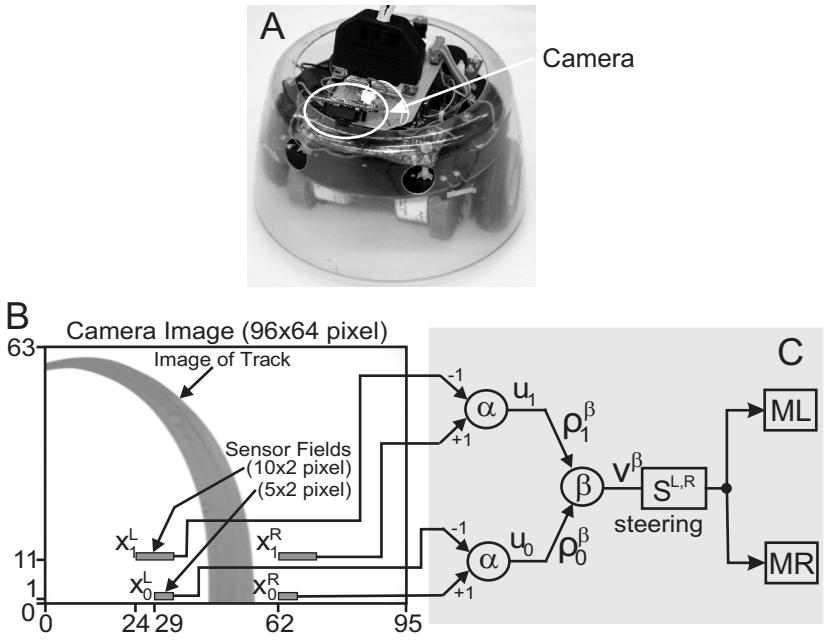

Figure 1: Physical and neuronal setup of the leaning. **A)** Image of the Rug Warrior Pro driving robot. **B)** Camera image with sensor fields marked by $x_1^{L,R}$ and $x_0^{L,R}$. **C)** The simple neuronal setup of the robot. Symbols α and β denote neurons, u denote filtered input signals x, ρ connection weight and v output of the neuron used for steering. v is calculated by the method shown in Fig. 2 B 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. 2 B) has inputs x_j which feed into a summation unit v. The output is calculated by

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

where $u = h * x$ is a 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 $b = \sqrt{(2\pi f)^2 - a^2}$, with f the frequency and $Q > 0.5$ the damping. This convolution allows correlating temporally non-overlapping signals (see Fig. 2 A).

The time delay T (see Fig. 3) between x_0 and x_1 depends on the speed of the robot. To accommodate some variability, x_1 is fanned out and fed into a filterbank of different filters h as indicated by the dashed lines in Fig. 2 B. 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

$f_{1,k} = 2.5/k\ Hz$, $k = 1, \ldots, 10$ for the filterbank in the x_1 pathway. Frequency of the x_0 pathway was $f_0 = 1.25\ Hz$. Damping parameter of all filters was $Q = 0.6$.

Weights change according 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 are discussed in a recent article (Porr and Wörgötter, 2006). *Note for the referees: This paper can be downloaded at: http://www.chaos.gwdg.de/~tomas/DrivingRobot/.*

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 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 grow ρ_1, such that the learner can use the earlier signal at x_1 to generate an anticipatory reaction. Learning stops and the weights stabilise at the condition $x_0 = 0$ when the reflex is not triggered anymore. The convergence properties of this kind of closed loop learning are discussed in (Porr and Wörgötter, 2006; Porr et al., 2003b).

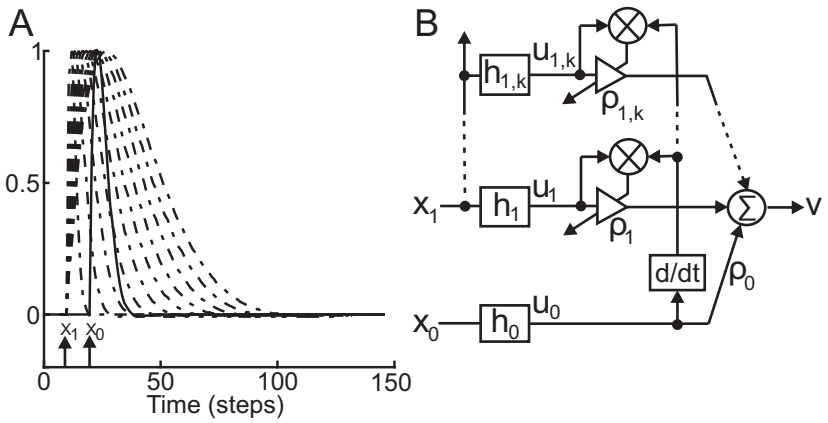

Figure 2: **A)** Resonator filters h_0 (solid line) for the input signal x_0 and $h_{1,k}$ (dashed lines) for the x_1 given by parameters $f_{1,k} = 2.5/k\ Hz$, $k = 1, \ldots, 10$ for the filterbank in the x_1 pathway. Frequency of the x_0 pathway was $f_0 = 1.25\ Hz$. Damping parameter of all filters 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 a temporal derivative. The amplifier symbol stands for a variable connection weight. Dashed lines indicate that input x_1 is fed into a filterbank.

2.3 Embedding learning in a closed loop scenario

Fig. 3 shows how such a learning unit can be embedded in a closed-loop system. Initially (see panel A) the system is set up only to react to the near-sense x_0 by ways of a reflex. This reflex will after some behavioural delay reset the signal form the near-sensor again to its starting value (often zero) closing the loop. In more technical terms, this represents a negative feedback-loop controller.

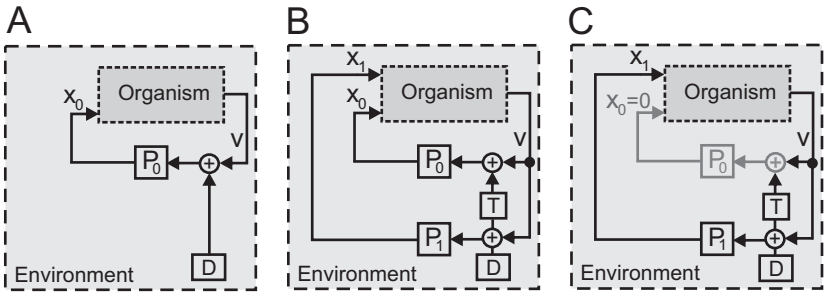

Figure 3: Schematic diagram of the control **(A)**, learning **(B)** and post-learning case **(C)**. Components of the learning system: sensor inputs x_0 and x_1, motor output v, P_0 denotes a reflexive pathway and P_1 predictive pathway. D - disturbance, T - time delay.

The learning system, however, contains a second predictive loop (panel B) from a sensor x_1 that receives an earlier signal (far-sensor). At the beginning of the learning, synapses (ρ_1) which convey information from the far-sense are zero and in Fig. 3 B only the inner loop (x_0) is functioning. During learning, synapses ρ_1 will get strengthened and the system will increasingly better react to the far-sense. As a consequence reactions occur earlier and the reflex based on x_0 will not be triggered anymore. Effectively, the inner loop has functionally been eliminated after learning (see Fig. 3 C). 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 Discussion*).

Intuitively the mechanism introduced in Fig. 3 will work with any aversive reflex. One should, however, note the same mechanisms can also be used to learn earlier attraction reactions. Already Braitenberg (1984) had nicely demonstrated that it is the sign-combination of the motor signals which 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 stop to change as soon as this condition ($x_0 = 0$) is fulfilled (Porr et al., 2003b; Porr and Wörgötter, 2006). Hence 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 the introduction, 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. Fig. 4 B shows a sequence of camera frames obtained during a left curve and the corresponding raw input signals (Fig. 4 A) obtained from the sensor fields $x_{0,1}^{L,R}$ (sum over all pixels within the sensor field). The vertical solid lines in panel A show that signals x_1 are indeed earlier than those at x_0. The sequence of camera frames in B demonstrates that the ego-motion of the robot creates quite some variability in the field of vision of the robot (see video camera.mpg on

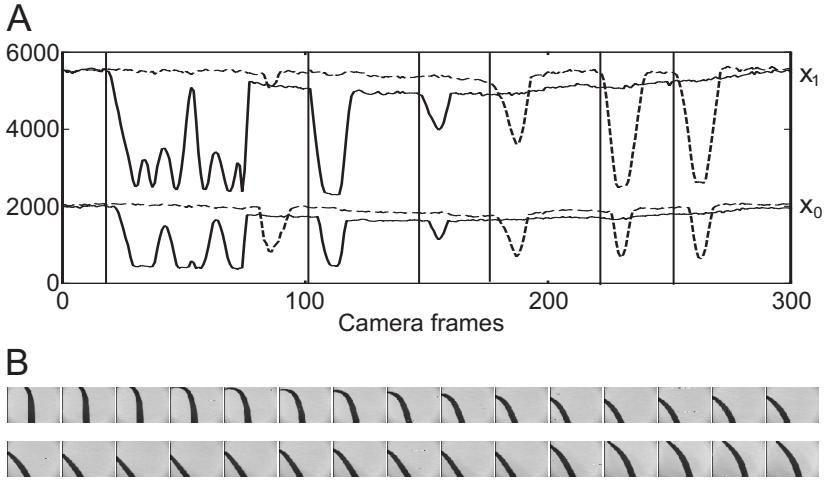

Figure 4: **A)** Input signals of the learning system $x_{0,1}$. Dashed lines represent signals from the right- (x^R) and solid lines those from the left sensor fields (x^L). Track layout is shown in Fig. 5 C. Signals before camera frame 150 come from the left turn, those after frame 150 from the right turn of the robot. **B)** Sequence of camera frames taken from the left curve.

http://www.chaos.gwdg.de/˜tomas /DrivingRobot/), for example the moving-out and moving-in of the bent line clearly visible in the second row in B. 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 of the robot is presented in Fig. 1 C. 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. Synaptic weight ρ_0^β are also set to a fixed value of 1 and only weights ρ_1^β of all ten filters (see Fig. 2 A) change. The output v^β is used to modify the motor signals of the robot. Note, in this experiment the setup for the weight development is symmetrical 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, left and right curves do not have any *a priori* bias. Hence, situations were, 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, since the robot is continuously driving, we perform on-line 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 $S = 0.125\ m/s$ in all experiments. This signal is modified by braking ($|v^\beta|$)

and steering ($\pm v^{\beta}$) according to:

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

where for the left motor we use "-" and for the right "+". Numerical constants as well as g are determined by the 12-bit resolution of the used DA-converter, where 0=maximal reverse speed and 4095=maximal forward speed. For the chained architectures, introduced later (see Fig. 10 B, C), we use v^{γ} instead of v^{β} in the equation (3).

3 Results

3.1 Basic behaviour of the simple system

The simple architecture was applied in the line following task and three different tracks were used in this experiment. Trajectories are shown for a left and a right curve. Weights and motor signals corresponding to the respective tracks are shown to their left. Sensor fields for predictor x_1 and reflex x_0 are as depicted in Fig. 1 B. 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. 5 A,B two learning trials (separated by a dashed line) are shown, between which connection weights were frozen and the robot 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 reflex and predictive response allows the robot to stay on the line already *during* the first learning trial (see Fig. 5 C trajectory T1). In the first learning trial the motor signal (Fig. 5 B) shows three leftward cumulative reflex+predictive reactions (large troughs) and seven (two leftward and five rightward) non-reflexive reactions. In the second trial only non-reflexive leftward and rightward steering signals occurred and the reflex was not triggered anymore. An appropriate steering reaction was learnt after three reflexes (reflected by the three peaks in the weight-curve in Fig. 5 A) during the first learning trial corresponding to about 50cm of the track (whole track is about 2 meters). Due to the symmetry of this setup (see Fig. 1 C), 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 smoother (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 (panels D-F) are similar to those from the previous experiment but for this track learning stopped already after two reflexes even with a lower learning rate of $\mu = 2.5 \times 10^{-6}$ as compared to the previous experiment where $\mu = 3 \times 10^{-6}$. As expected a much weaker steering reaction (Fig. 5 E) was learnt and weights are smaller. For movie of the learning behaviour see shallow.mpg.

The third experiment was performed using a track with very sharp corners (Fig. 5 I) and a relatively higher learning rate $\mu = 6.5 \times 10^{-6}$. This was done to demonstrate that fast and stable learning is possible even for such a sharp track. The results of three learning trials (separated by dashed lines) are presented in Fig. 5 G-H. The robot missed the track twice and finally succeeded in the third trial (see also sharp.mpg). As before, it can now use the learnt weights also 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 at some distance from the bottom of the camera image. Because steering necessarily always

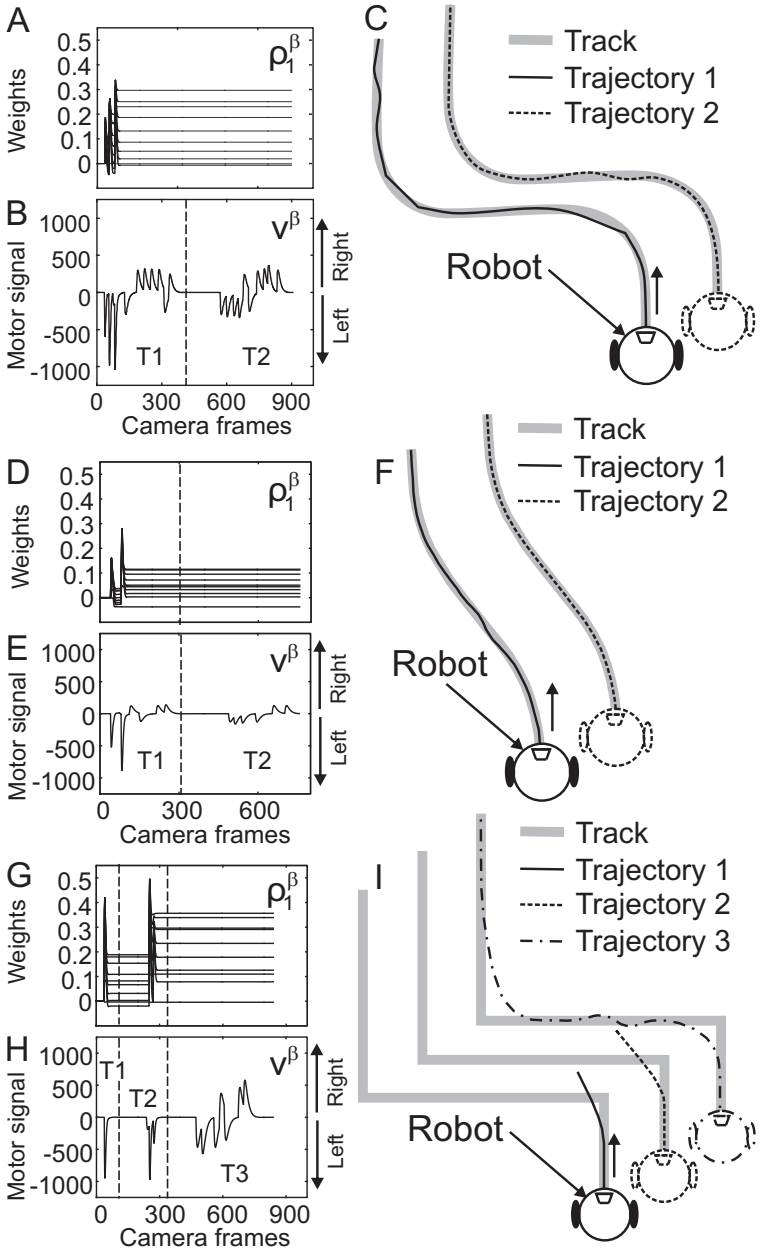

Figure 5: Results of the driving robot experiment using the simple architecture (see Fig. 1 C). **A-C)** Results for the intermediately steep track (C). Learning rate was: $\mu = 3 \times 10^{-6}$. **A)** Connection weights ρ_1^β, **B)** motor output v_β, and **C)** driving trajectories. Trajectory T1 during and T2 after learning. **D-F)** Results for the shallow track. Learning rate was $\mu = 2.5 \times 10^{-6}$. **D)** Connection weights ρ_1^β, **E)** motor output v_β, and **F)** driving trajectories. **G-I)** Results for the sharp track. Learning rate was $\mu = 6.5 \times 10^{-6}$. **G)** Connection weights ρ_1^β, **H)** motor output v_β, and **I)** driving trajectories.

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.

Fig. 6 shows results for two control experiments with a shallow left and an increasingly sharper right curve (see Fig. 6 E). Connection weights ρ_1^β (panel A) and motor output v_β (panel B) 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 even from following the very shallow left curve (see trajectory T1 in Fig. 6 E). In the second trial, the robot succeeded for the left curve and the beginning of the right curve but the learnt steering reaction still was not enough to allow it following the sharper parts of the right curve at the end of the spiral (see trajectory T2 in Fig. 6 E). In the third learning trial the robot succeeded to follow the whole trajectory completely (see trajectory T3 in Fig. 6 E) but still most of the time a mix of predictive and reflexive (large peaks) steering reactions occurred. The robot continued to improved its steering reactions in the fourth trial (trajectory not shown, but see video of whole experiment: spiral-low.mpg) where one can see more non-reflexive 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 yet finished at this stage and would need more trials until weights finally stabilise.

To speed-up the learning process a higher learning rate of $\mu = 1.5 \times 10^{-6}$ was used and three learning trials are presented in Fig. 6 C,D. In this case, the robot is able to stay on the line already during the first learning trial (trajectories not shown but see video spiral-high.mpg) but still more predictive+reflexive (large peaks) than non-reflexive steering reactions occurred (see panel D). In the second trial only two predictive+reflexive reactions occurred whereas in the last trial only non-reflexive steering reactions occurred and weights did not change anymore. When we use the final weights learnt with the sharp curve afterwards for driving a through the shallow left curve in a third trial the robot oversteers slightly 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. 6 F).

We also did an experiment to see how the robot behaves on a difficult track with different kinds of curvatures (see Fig. 7 C). The total length of track is approximately 14.5 m. Connection weights and the motor output are shown in Fig. 7 A,B. The robot had three reflexes at the beginning (Fig. 7 A, see arrows in C) while turning to the right and after that the reflex input was not triggered till it approached the crossing point where the robot turned to the right (see trajectory in Fig. 7 C) and the reflex was triggered twice again. After that the reflex was not triggered anymore and weights stopped changing. When the robot approached the crossing point for 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 uses the final weights learnt for the sharpest curve and over-steers a bit 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.

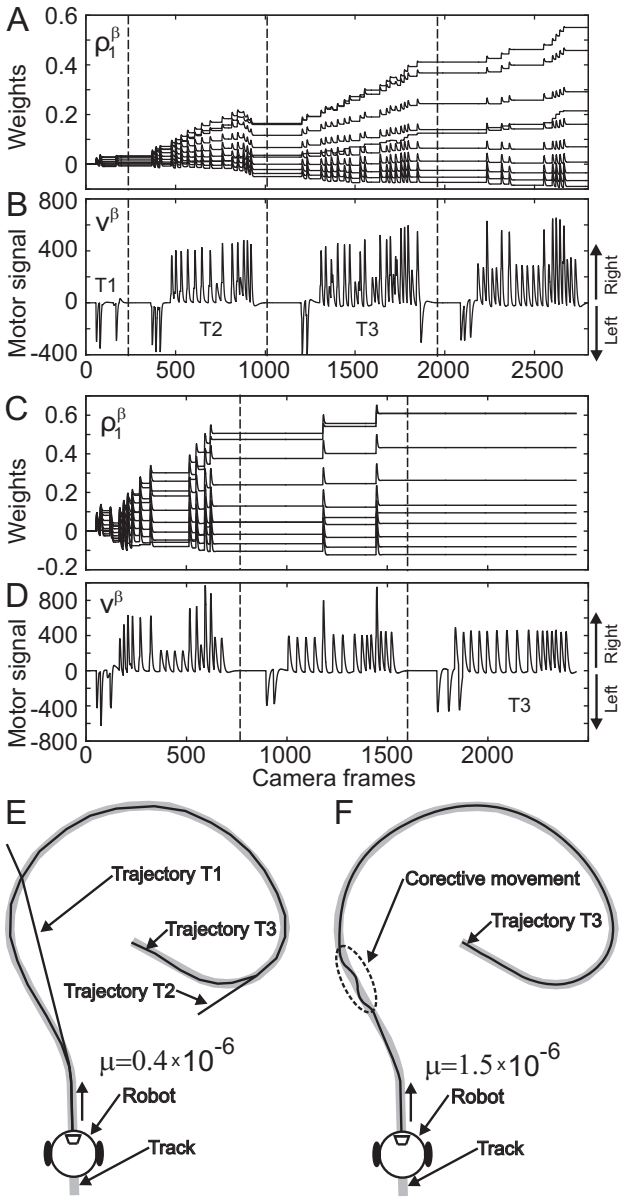

Figure 6: Results of the driving robot experiment using the simple architecture (see Fig. 1 C) on a spiral track. **A-B)** Results for a learning rate of $\mu = 0.4 \times 10^{-6}$. **A)** Connection weights ρ_1^β, **B)** motor output v_β. **C-D)** Results for a learning rate of $\mu = 1.5 \times 10^{-6}$. **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 trajectory 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, not-shown learning trajectories, when using the higher learning rate of $\mu = 1.5 \times 10^{-6}$. In this case we find weight stabilisation after two trials (see Fig. 6 C), but learnt weights will lead to too strong reactions for shallow curves which are compensated by corrective movements (see bottom part of the trajectory).

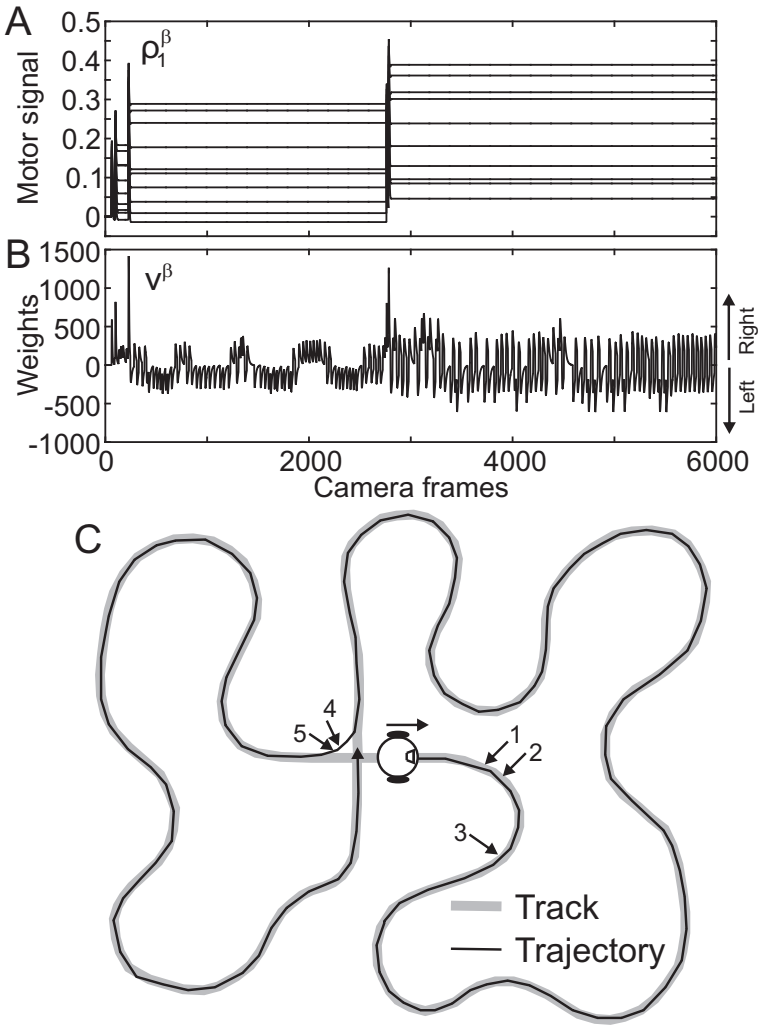

Figure 7: Results of the driving robot experiment using the simple architecture (see Fig. 1 C) on a maze track. **A)** Connection weights ρ_1^β, **B)** motor output v_β. Learning rate was $\mu = 3 \times 10^{-6}$, weights stabilised after five reflexes. **C)** Maze track and robot's driving trajectory for the first loop.

3.2 Statistical evaluation

In the experiments above it has become clear that our system performs on-line (and not batch) learning. Hence the most critical parameter affecting the convergence of learning is in which way the momentary behaviour will influence, or rather generate, the next learning experience. Ultimately this is given by the sequence of viewing angles which the robot creates due to its own driving. As a consequence the 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 learning rate as well as relative placement of the different sensor fields.

Thus, to investigate the robustness against these parameters we used a simulation and performed a set of more than one-thousand experiments where we let the simulated robot learn to follow left-right tracks with angles of 20, 45 and 90 degrees (see Fig. 8 A). 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. 8 B. We used the neuronal setup as presented in Fig. 1 C. The output of the neuron v^β modified by transformation function $S^{x,y}$ was used here to change the position of the robot in the environment. It was calculated according to the following equations:

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

$$S_t^y = Sp_t^y + r\ sin(\alpha_t),\ where \tag{5}$$

$$Sp_t^x = Sp_{t-1}^x + (1 - 0.001\ |v^\beta|)\ cos(\alpha_t), \tag{6}$$

$$Sp_t^y = Sp_{t-1}^y + (1 - 0.001\ |v^\beta|)\ sin(\alpha_t),\ and \tag{7}$$

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

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

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

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

If these three conditions are not fulfilled at the same time then we count an experiment as a failure. Results demonstrating the influence of the robot's position angle while placing the robot at the starting position are presented in Fig. 8 D. 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. The success is slightly decreasing if we increase the variance of the starting angle distribution σ_α^2, but we still get high performance (success rate $0.92 < succ \leq 0.99$ for all tracks). More reflexes are needed to accomplish the task if σ_α^2 is increased. Also, as expected, more reflexes are required for the sharp track as compared to shallower ones. Results of 100 experiments for different positions of the predictor sensor x_1 are shown in Fig. 8 E. Success rate decreases if the distance between inputs is getting larger for the sharp track whereas for the shallow and middle track decrease is less significant but also decreases when the distance is very large ($d = 9/10$. The number of necessary reflexes (NR) is increasing if the distance between x_1 and x_0 is getting 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 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 to steer the sharp track within 20 trials when distance between x_1 and x_0 was > 8. We also investigated the influence of the learning rate and results of 100 experiments are presented in Fig. 8 F. The learning rate does not affect performance except for the sharp track when the learning rate is relatively low and the robot does not succeed to steer the curve within 20 trials. As expected we find that with a higher learning rate less reflexes are needed to complete the task, because with a higher learning rate weights are growing faster and the task is learnt sooner.

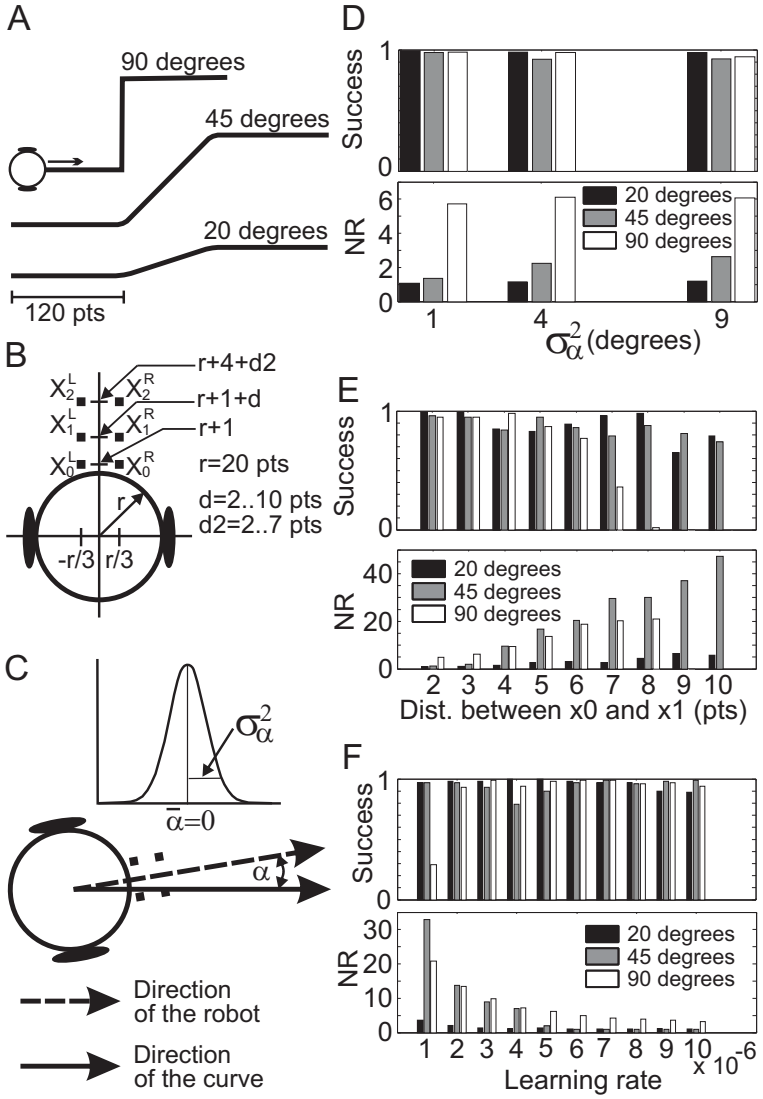

Figure 8: **A-C)** Setup of the simulation experiment. **A)** Tracks with 95, 45 and 20 degrees. **B)** Positions of the input signals $x_{0,1,2}^{L,R}$. **C)** Angle α of the robot at its starting position, given by the deviation from the direction of the track when placing the robot at the start. In the experiments a Gaussian distribution of α has been 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. 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. 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 $d = 3$.

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. 9 A. There is one reflex input x_0 and two predictive inputs x_1 and x_2. 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. 9 B) is named *honeycomb-chain*, to which its structure resembles. Output $v_1^{\beta,1}$ is used as the reflex input of the neuron γ and output $v_1^{\beta,2}$ as its predictive input. Weights $\rho_0^{\beta,1}$ and ρ_0^{γ} are set to a fixed value 1, all other weights are 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 both 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 $\pm$ 5 time steps and x_1 occurs 10 time steps earlier than x_0 with the same variability. This impulse sequence has been repeated every 50 time steps.

Simulation results for the linear-chain (Fig. 9 A) are presented in Fig. 9 B,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 respective 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 legend of Fig. 9 for equations).

Using this criterion, first the connection weights ρ_1^{β} stabilise and after some time ρ_1^{γ} stop to change. Results for the honeycomb-chain (Fig. 9 D) are presented in Fig. 9 E-G. In this situation first the connection weights $\rho_1^{\beta,1}$ stop to change and later both 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 is presented in Fig. 10. The neuronal setup for the linear-chain architecture is presented in Fig. 10 B and for the honeycomb-chain in Fig. 10 C. It is similar to those above (see Fig. 9 A,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. 10 B) are presented in Fig. 11 A-D and the results for the honeycomb-chain (Fig. 10 C) in Fig. 11 E-H. In the first learning trial the motor signal (Fig. 11 C) shows three leftward cumulative reflex+predictive reactions and two non-reflexive 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 where contributing to the change on ρ_1^{β} *and* ρ_1^{γ} (Fig. 11 A,B) but not the rightward reflexes, which only contributed to the change of ρ_1^{γ}.

In the second trial only non-reflexive leftward and rightward steering signals occurred and the reflex was not triggered anymore. The driving trajectories are shown in Fig. 11 D and in the video linear-chain.mpg. Weights at a certain neuron stabilise as soon as their corresponding reflex input remains silent. For the linear-chain (A-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 v_β remains longer active. Essentially the same is true for the honeycomb-chain (E-H). Here $\rho_1^{\beta,1}$ stops growing first, which gets the same reflex input u_0 as ρ_1^{β} in the linear-chain. Convergence of the

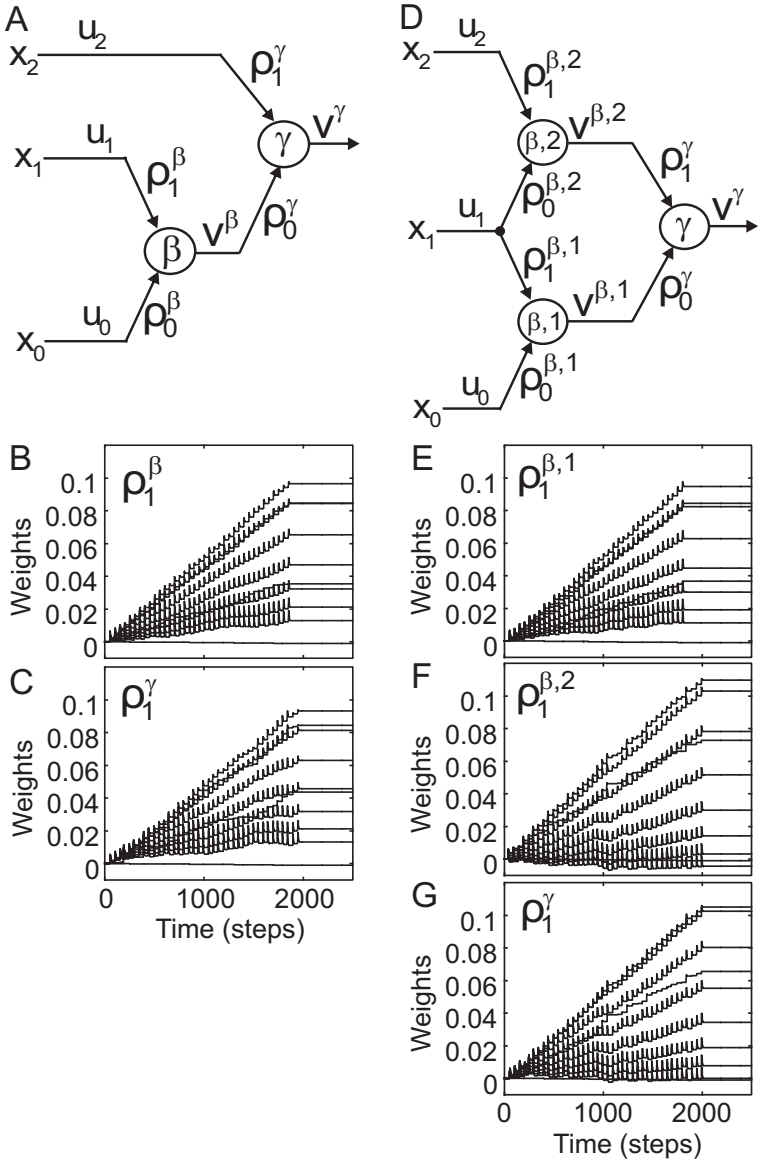

Figure 9: Chained neuronal architectures **(A, D)** and simulation results for the open loop case **(B, C, E-G)**. **A)** Linear-chain and **D)** honeycomb-chain. Learning rate for both architectures was $\mu = 10^{-7}$. **B, C)** Results for the linear-chain **(A)**. Connection weights ρ_1^β and ρ_1^γ. Weights ρ_1^β stop growing at the condition $x_0 = 0$ and ρ_1^γ stop growing when $x_1 = 0$. We have set $x_0 = 0$ when the sum of weights over all ten filters is $\sum_{k=1}^{10} \rho_{1,k}^\beta \geq 0.5$ and $x_1 = 0$ when $\sum_{k=1}^{10} \rho_{1,k}^\gamma \geq 0.5$. **E-G)** Results for the honeycomb-chain **(D)**. Connection weights $\rho_1^{\beta,1}$, $\rho_1^{\beta,2}$, ρ_1^γ. Weights $\rho_1^{\beta,1}$ stop growing at the condition $x_0 = 0$, $\rho_1^{\beta,2}$ and ρ_1^γ stop growing when $x_1 = 0$. We have set $x_0 = 0$ when sum of weights over all ten filters is $\sum_{k=1}^{10} \rho_{1,k}^{\beta,1} \geq 0.5$ and $x_1 = 0$ when $\sum_{k=1}^{10} \rho_{1,k}^{\beta,2} \geq 0.5$.

weights $\rho_1^{\beta,2}$ is controlled by reflex input u_1 which also contributes to the signal $v^{\beta,1}$, being the reflex input to neuron γ. Hence weights $\rho_1^{\beta,2}$ and ρ_1^γ behave in the same way and stabilise later (similar to ρ_1^γ of the linear-chain).

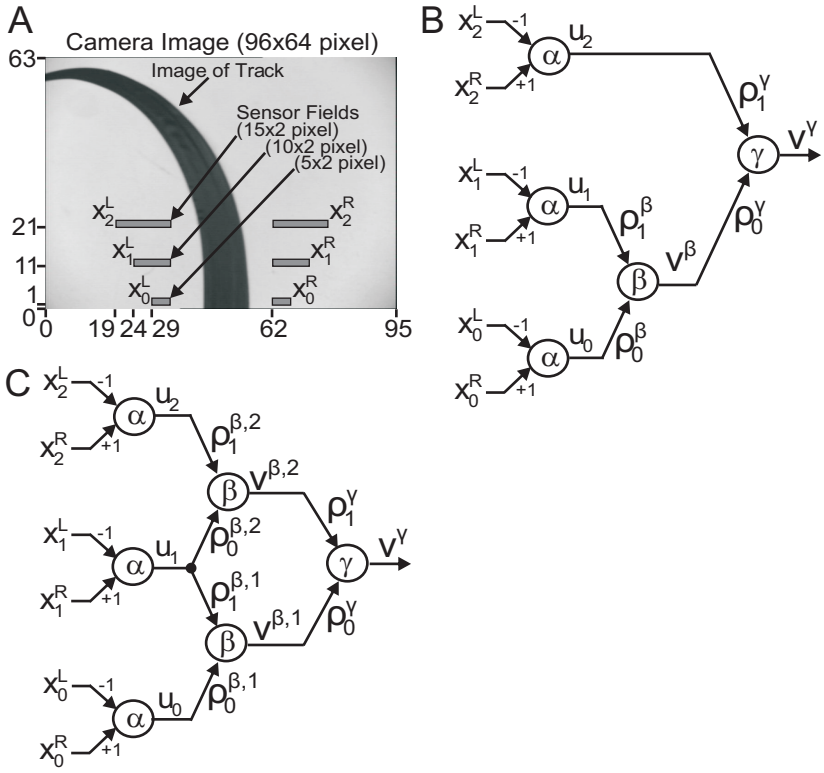

Figure 10: The physical **A)** and neuronal **B,C)** setups of the chained learning system for the closed loop case. **A)** Camera image with sensor fields marked by $x_{1,2}^{L,R}$ and $x_0^{L,R}$. **B)** Linear-chain and **C)** honeycomb-chain.

3.4 Statistical evaluation

We also did simulations using chained architectures in order to a make comparison with the simple setup. The simulation setup for the chained architectures is shown in Fig. 8 B. Positions of sensor fields $x_1 L, R$ and $x_0 L, R$ were fixed (distance was 3 pts) and we only varied the position of sensor field $x_2 L, R$. The influence of the robot's position angle while placing the robot at the starting position are presented in Fig. 12 A-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 (panel A) as compared to the simple architecture where 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 as compared to the simple setup (success rate $0.86 < succ < 0.96$ for all tracks). Also, as for using simple setup, more reflexes are required for the sharp track as compared to shallower ones. For the honeycomb-chain (panel D) performance was again lower: success rate $0.71 < succ \leq 0.94$ for the shallow and middle track where for the sharp track we got very low performance (success rate $succ < 0.1$). 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 (we kept positions of x_1 and x_0 fixed) in Fig. 8 D. 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

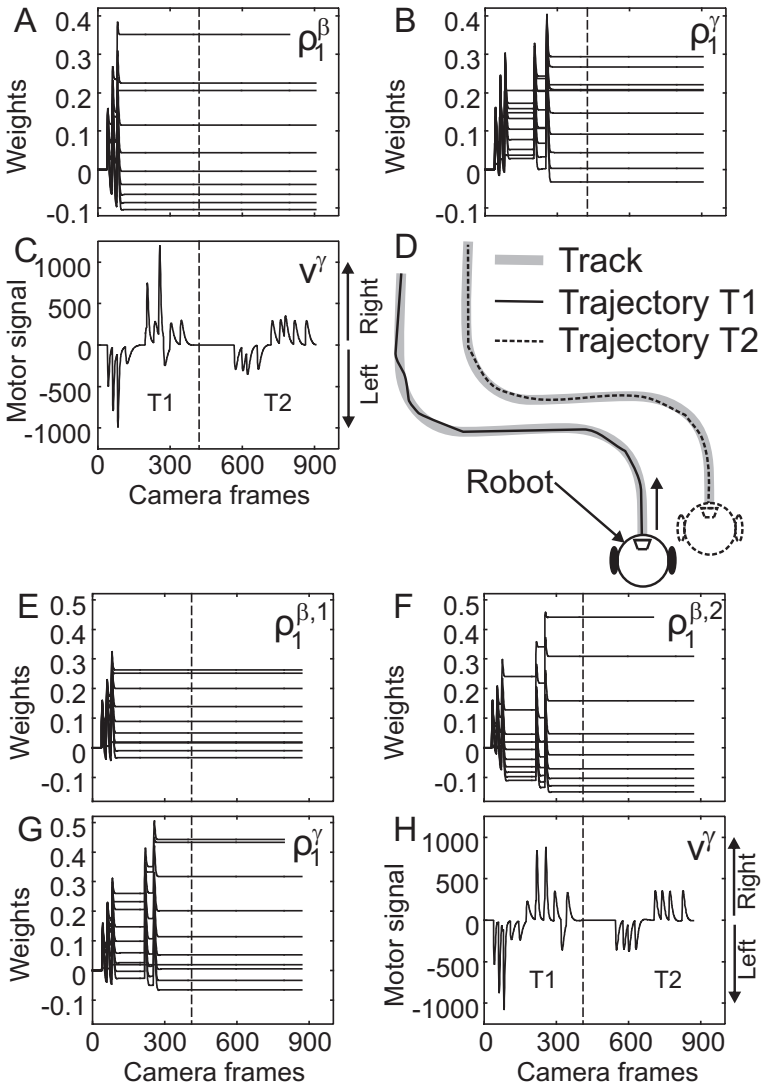

Figure 11: Results of the driving robot experiment using chained architectures. Learning rate for both experiments was $\mu = 2.5 \times 10^{-6}$. **A-D)** Results for the linear-chain (see Fig. 10 B). **A,B)** Connection weights ρ_1^β and ρ_1^γ. **C)** Motor output v_γ, and **D)** driving trajectories. Trajectory T1 during and T2 after learning. **E-H)** Results for the honeycomb-chain (see Fig. 10 C). **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 not shown.

(success rate $0.70 \leq succ \leq 0.96$ for all tracks) where for the middle track the importance of the position of the sensor fields is not significant (except for the smallest distance between x_2 and x_1 given $d_2 = 2$ pts). For the linear-chain setup (panel C) we obtained the same results as for the simple one. Success rate decreases if the distance between inputs is getting larger only for the sharp track whereas for the shallow and middle track decrease is not significant. We also observed that the number of necessary reflexes (see panel C-D) is increasing if the distance between x_1 and x_0 is getting larger except very small distances between x_2 and x_1 when using honeycomb-chain setup (panel D).

We can summarise that better performance is obtained with the simple setup as compared

to the chained architectures. The performance does not crucially depend on the starting angle position. It decreases only slightly if the variance of starting angle position increases. In general we observed that only for the honeycomb-chain architecture performance depends on the position of the sensor fields (distance between sensor fields). The learning rate does also not affect performance itself. The robot only needs more reflexes to learn the task if we use relatively low learning rates.

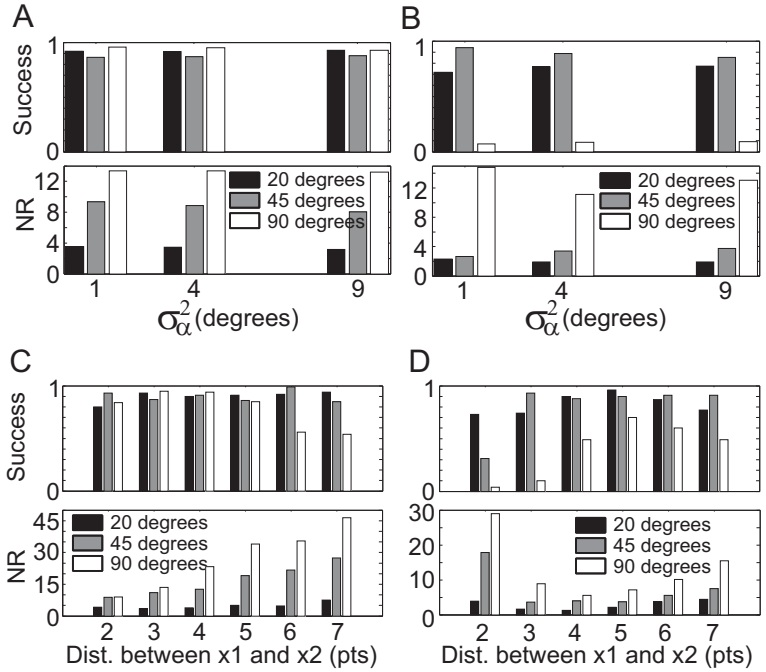

Figure 12: **A-D)** Results of the simulation experiments using chained architectures. **A-B)** Success in 1000 experiments and average number of reflexes (NR) at the motor output neuron γ needed to accomplish the task within successful trials are plotted against the variance σ^2_α: **A)** linear-chain, **B)** honeycomb-chain. Learning rate was $\mu = 5 \times 10^{-6}$ and 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. Distance between x_1 and x_0 was fixed and was d=3. Learning rate was $\mu = 5 \times 10^{-6}$ and the variance was $\sigma^2_\alpha = 4$.

4 Simple architecture vs chained architectures

Previously we summarised that with the simple setup we get better performance as compared to the chained architectures. This is true only for cases where we have good input correlations (small distances between inputs) in the simple setup. Performance decreases if the distance between inputs is very large (see Fig. 8 E) for the shallow and middle track and the robot never managed to steer the sharp curve when the distance between inputs was > 8. However, the robot managed to steer the sharp curve when chained architectures were used (see Fig. 12 C,D) where the distance between inputs x_2 and x_1 was > 5 and between x_1 and x_0 was 3 (total distance between x_2 and x_0 was > 8). To test the hypothesis whether chained architectures

are advantageous for bad correlations because of sparse inputs we did an experiment where we compared the performance of all three architectures on the middle curve (45 deg.). The setup of the input configuration for the simple architecture is shown in Fig. 13 A and for the chained architectures in Fig. 13 B. Distance between inputs x_1 and x_0 in the simple setup was 15 pts and in the chained architectures it was 8 between inputs x_2 and x_1 and 7 between x_1 and x_0 (total distance between x_2 and x_0 was 15 pts). A comparison between all three architectures is presented in Fig. 13 C,D where we plot the success rate in 500 experiments (panel C) and the average number of trials (NT) within successful experiments together with confidence intervals (95%) needed to accomplish the task (see panel D). From the results we can conclude that chained architectures indeed perform better (success rate for the linear-chain 0.87 and for the honeycomb-chain 0.92) whereas for the simple architecture we obtained a success rate of only 0.57 (see panel C). We also needed less trials to complete learning when using chained architectures as compared to the simple setup (see panel D).

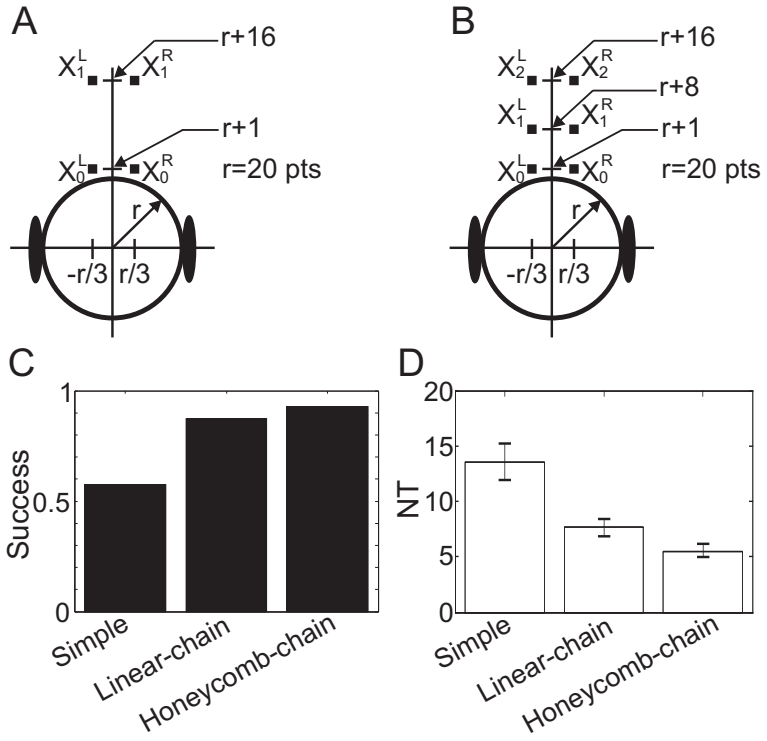

Figure 13: **A, B)** Setup of the simulation experiment. **A)** Simple setup. Positions of the input signals $x_{0,1}^{L,R}$. **A)** Chained architectures. Positions of the input signals $x_{0,1,2}^{L,R}$. **C,D)** Results of the simulation experiments using different neuronal setups on the middle track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

5 Discussion

This study has addressed two specific aspects in the field of temporal sequence learning. We have focused on: 1) chaining learning architectures in a closed loop perception-action system and 2) the question whether chained architectures can be employed in order to obtain better behavioural performance as compared to the simple architecture in cases where we have sparse inputs in time and correlations between these inputs are weak. In the following we would like to discuss how the open- and closed-loop situation compares to biological and other artificial systems, how our methods relate to other approaches for RF-development, and where there are relations to some aspects of reinforcement learning.

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. 5 C), while it does not follow the line if only the reflex alone is employed (see video control.mpg). Hence, already during the first learning reflex synaptic weights 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 a stable behavioural domain (Porr and Wörgötter, 2006), provided the input correlations are robust enough.

Biological systems are generally operating in close conjunction with their environment. This so-called ecological embedding has already been discussed by theoreticians very early as also essential for autonomous artificial agents (Ashby, 1956; McFarland, 1971; Wiener, 1961). On the more practical side the work of 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 had been built, which do not adapt but instead react to a stimulus by ways of reflex-like behaviour. 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 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” performing pure reflex-based control, one above an ”adaptive layer” performing predictive learning much in the sense

of classical or operant conditioning and finally on top a "contextual layer" for higher level adaptation (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, called feedback-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 which are available to the (late) feedback system. A simple example is feedback loop which 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 could be made faster to by adding an adaptive controller which receives a copy of the disturbance itself or the output of the feedback controller. Because we have got an overdamped system, FEL would learn to become the derivative of the disturbance. In other words, FEL would adaptively learn to add the "D" to a PI controller. ICO or ISO learning, however, are fundamentally different because they use 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 as such.

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 this, however, means that the reflex has been functionally eliminated and has now been successfully replaced by an earlier anticipatory action. This property allows controlling the homeostasis of learning and behaviour at the same time, which is more difficult to achieve with most other architectures.

In this study we were concerned with designing simple chained architectures of our learning modules. This was motivated by the fact that the sensor information in animals internally progresses along many stages until a motor output is generated. It is unknown, how such complex sensor-motor loops maintain behavioural stability, let alone behavioural stability during changing synaptic strengths between these different stages. 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 (Wörgötter 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, 2004) making these algorithms better compatible to neuronal architectures.

Indeed, some Actor-Critic algorithms have been also 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 there is no generic "recipe" existing, 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 (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 (Wörgötter et al., 2007).

Hence, one starting point for this study was the assumption that input control should allow designing 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 in order to obtain better behavioural performance as compared to the simple architecture where learning fails because of weak correlations.

Acknowledgements

The European Project "PACO+" is being acknowledged.

References

Agostini, A. Celaya, E. (2004). Trajectory tracking control of a rotational joint using feature-based categorization learning. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan. IEEE.

Ashby, W. R. (1956). *An introcduction to cybernetics*. Methnen+Co LTD, London.

Bailey, C. H., Giustetto, M., Huang, Y. Y., Hawkins, R. D., and Kandel, E. R. (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., editor, *Handbook of Brain Theory and Neural Networks*, pages 809–812. MIT Press.

Barto, A. G., Sutton, R. S., and Anderson, C. W. (1983). Neuronlike elements that can solve difficult learning control problems. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume 13, pages 835–846.

Braitenberg, V. (1984). *Vehicles: Experiments in Synthetic Psychology*. MIT Press.

Gomi, H. and 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., and 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., and 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*, 44:8187.

Jay, T. (2003). Dopamine: a potential substrate for synaptic plasticity and memory mechanisms. *Prog Neurobiol*, 69(6):375390.

Jodogne, S., Scalzo, F., and Piater, J. H. (2005). Task-driven learning of spatial combinations of visual features. In *Proc. of the IEEE Workshop on Learning in Computer Vision and Pattern Recognition*, San Diego (CA, USA). IEEE.

Kelley, A. E. (1999). Functional specificity of ventral striatal compartments in appetitive behaviors. *Ann N Y Acad Sci*, 877:71–90.

Klopf, A. H. (1988). A neuronal model of classical conditioning. *Psychobiol.*, 16(2):85–123.

Kosco, B. (1986). Differential Hebbian learning. In Denker, J. S., editor, *Neural networks for computing: AIP Conference Proc. proceedings*, volume 151. New York: American Institute of Physics.

McClelland, J. L., Rumelhart, D. E., and Hinton, G. E. (1987). *Parallel Distributed Processing - Vol. 1*. MIT Press.

McFarland, D. J. (1971). *Feedback Mechanisms in Animal Behaviour*. Academic Press, London.

Montague, P. R., Dayan, P., Person, C., and Sejnowski, T. J. (1995). Bee foraging in uncertain environments using predictive hebbian learning. *Nature*, 377:725–728.

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

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

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

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

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

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

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

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

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

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

Sutton, R. S. and Barto, A. G. (1990). Time-derivative models of Pavlovian reinforcement. In Gabriel, M. and Moore, J., editors, *Learning and computational neuroscience:Foundation of adaptive networks*. MIT Press.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Bradford Books, MIT Press, Cambridge, MA, 2002 edition.

Tsukamoto, M., Yasui, T., Yamada, M. K., Nishiyama, N., Matsuki, N., and 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. and Althaus, P. (2003). A real-world rational agent: unifying old and new AI. *Cognitive Science*, 27:561–590.

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

Walter, W. G. (1950). An imitation of life. *Scient. Am.*, 182:42–45.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. PhD thesis, University of Cambridge, Cambridge, England.

Watkins, C. J. C. H. and 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*. The M.I.T. Press, Cambridge, Massachusetts, 2 edition.

Witten, I. H. (1977). An adaptive optimal controller for discrete-time Markov environments. *Information and Control*, 34:86–295.

Wörgötter, F., Kolodziejski, C., and Porr, B. (2007). Comparing neuronal approaches for temporal sequence learning. *Natural Computing, in press*.

Wörgötter, F. and 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.