Available online at www.sciencedirect.com

SCIENCE DIRECT®

Neural Networks 19 (2006) 339–353

Neural Networks

www.elsevier.com/locate/neunet

A hybrid generative and predictive model of the motor cortex

Cornelius Weber *, Stefan Wermter, Mark Elshaw

Centre for Hybrid Intelligent Systems, School of Computing and Technology, University of Sunderland, Sunderland SR6 0DD, UK

Received 24 November 2003; accepted 13 October 2005

Abstract

We describe a hybrid generative and predictive model of the motor cortex. The generative model is related to the hierarchically directed cortico-cortical (or thalamo-cortical) connections and unsupervised training leads to a topographic and sparse hidden representation of its sensory and motor input. The predictive model is related to lateral intra-area and inter-area cortical connections, functions as a hetero-associator attractor network and is trained to predict the future state of the network. Applying partial input, the generative model can map sensory input to motor actions and can thereby perform learnt action sequences of the agent within the environment. The predictive model can additionally predict a longer perception- and action sequence (mental simulation). The models' performance is demonstrated on a visually guided robot docking manoeuvre. We propose that the motor cortex might take over functions previously learnt by reinforcement in the basal ganglia and relate this to mirror neurons and imitation.

Keywords: Motor cortex; Basal ganglia; Forward model; Unsupervised learning; Supervised learning; Reinforcement learning; Helmholtz machine; Continuous attractor network

1. Introduction

The prominent regions of motor skill learning are the basal ganglia (the largest part of which is known as striatum), the frontoparietal cortices and the cerebellum. Doya (1999) proposed that reinforcement learning appears in the basal ganglia, supervised learning in the cerebellum and unsupervised learning in the cortex. Hikosaka, Nakamura, Sakai, and Nakahara (2002) propose an integrated model in which cortical motor skill learning is optimised by a cortex basal ganglia loop taking advantage of reinforcement learning and a cortex–cerebellar loop taking advantage of supervised learning. Neurons in the sensorimotor striatum have been observed to be active only during early learning phases of a T-maze procedural task (Jog, Kubota, Connolly, Hillegaart, & Graybiel, 1999). Recently Pasupathy & Miller (2005) observed for monkeys learning an associative task, that striatal (specifically, caudate nucleus) activation progressively anticipates prefrontal cortical activity and that the cortical activity more closely parallels the monkey's behaviour improvement. The lead of striatal activity by as much as 140 ms indicates that the cortex may be a candidate structure to receive training by the basal ganglia (see also Ravel & Richmond, 2005, and the discussion).

Previously we have solved a robotic task by reinforcement learning mimicking basal ganglia function (Weber, Wermter, & Zochios, 2004). In this paper this performance will be copied by an unsupervised learnt generative model as well as with a predictive model trained supervised on the data given by the generative model. The task is that of a robot to approach a target in a pose suitable to grasp it ('docking') based on visual input and input conveying its heading angle. It corresponds perhaps to moving the limbs suitable to grasp an object by humans and monkeys. While we assume here that a module previously trained using reinforcements already performs the task, the generative and predictive motor cortex module shall learn it by observation.

The generative and the predictive models are identified with the hierarchically and laterally arranged cortical connections, respectively, an architecture which parallels a combined model of V1 simple and complex cells (Weber, 2001). The docking task intensively uses visual input, which makes it well suited to the function of the cortex. The cerebellum, however, also has, but rather indirect, access to visual information (Robinson, Cohen, May, Sestokas, & Glickstein, 1984).

The traditional role assigned to the motor cortex is to control movement: if e.g. neurons in area F5 of the motor cortex are

* Corresponding author. Tel.: +44 191 515 3274; fax: +44 191 515 3461.
E-mail address: cornelius.weber@sunderland.ac.uk (C. Weber).
URL: www.his.sunderland.ac.uk

0893-6080/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2005.10.004

Nomenclature

W^{bu}	bottom-up weight matrix conveying the input to the motor cortex
ΔW^{bu}	change of W^{bu} during learning
W^{td}	top-down weight matrix used to reconstruct input from hidden code
ΔW^{td}	change of W^{td} during learning
V	predictive weight matrix
ΔV	change of V during learning
$g(\cdot)$	neuronal transfer function of the neurons in the motor cortical area
s	parameter of $g(\cdot)$ which controls the maximum output
β	parameter of $g(\cdot)$ which controls the slope
k	parameter of $g(\cdot)$ which controls sparseness
η	weight learning rate
$\vec{p}$	input area activations encoding the perceived location of the target
$\vec{\Phi}$	input area activations encoding the robot heading direction
$\vec{m}$	input area activations encoding the motor activations
$\vec{f}=\vec{p}*\vec{\Phi}$	state space, used by the reinforcement-trained module
$\vec{x}=(\vec{p},\ \vec{\Phi},\ \vec{m})$	concatenation of the input area activations
$\vec{r}$	neuronal activations on the motor cortical area (hidden code)
$\vec{x}'$	reconstruction of the input activations by the network
$\hat{r}$	random hidden code, used in the sleep phase
$\hat{x}$	input activations generated from $\hat{r}$ in the sleep phase
$\hat{r}'$	reconstruction of the random hidden code in the sleep phase
$\vec{z}=(\vec{p},\vec{\Phi},\vec{m},\vec{r})$	concatenation of all network activations
$\tilde{z}=(\tilde{p},\tilde{\Phi},\tilde{m},\tilde{r})$	network activations constructed by the predictive weights
t	time; one time step denoting update of all neuronal activations (relaxation)
*	outer product

stimulated, then a limb movement will be executed. Many area F5 neurons, however, also fire in the absence of own motor action: if a monkey observes an action to be executed by another monkey or a human, then so-called mirror neurons fire action potentials (Gallese, Fadiga, Fogassi, & Rizzolatti, 1996). Also, action word representations are proposed to span language as well as motor areas in so-called ‘word webs’ (Pulvermüller, 2002). This leads to a distributed (population-) coding paradigm across several cortical areas, which are mostly topographically organised. Multi-modal connections of the motor areas to higher levels of the somato-sensory and visual cortex (Felleman & Van Essen, 1991) provide the necessary architecture.[1] With mirror neurons in motor cortical area F5 having also sensory neuron reponse properties it is therefore plausible to introduce learning principles of sensory cortical areas to the motor cortex. These considerations provide the backbone of our developmental model of one area of the motor cortex.

1.1. A generative model of the motor cortex

Monkeys in captivity sometimes imitate behaviours, which they observe from people, e.g. sweeping with a broom (Vittorio Gallese, personal communication). It is tempting to explain imitation behaviour with a generative model, as has been successfully applied to the well investigated lower visual area V1 of the cortex (Bell & Sejnowski, 1997; Olshausen & Field, 1997; Rao & Ballard, 1997; Weber, 2001). A sensory-motor generative model trained by error back-propagation has been proposed by Plaut & Kello (1998): it produces actions (speech), which lead via the environment to a similar sensory input distribution (sound) as previously experienced. However, since imitation behaviour is performed only by higher developed mammals, can we trace back a generative model to a more basic function of the motor cortex?

We go one step back and claim that the cortex reproduces experienced actions, which are originally produced by other parts of the brain rather than by other individuals. An action may originally be produced via phylogenetically older parts of the brain, e.g. the basal ganglia, and possibly via learning mechanisms such as reinforcement learning. Even though it is a powerful learning paradigm, reinforcement learning might leave actions stored inefficiently in the basal ganglia; in computational models a high-dimensional state space limits applications (see Section 4). When the cortex can take over, the basal ganglia would be available to learn new tasks.

An unsupervised model of the cortex as proposed by Doya (1999) is simple due to a similarity with sensory cortex models, and in that the direction of information flow (what is input and what is output) is not specified before learning. While a directed stimulus to response mapping, learnt supervised, might be more optimal, the motor cortex does not produce optimal movements, considering that damage to the cerebellum leads to lasting motor and movement difficulties (e.g. Sanes, Dimitrov, & Hallett, 1990).

Perception-action pairs might be input data to a generative model of the motor cortex, just like an image is the input to a generative model of the visual cortex. The cortical neurons represent this input as an internal activation pattern. This ‘hidden code’ can represent higher-order correlations within

[1] Given that layer 4 which receives bottom-up connections is poorly developed in the motor cortex, one might consider the motor cortex to be laterally connected to other cortical areas.

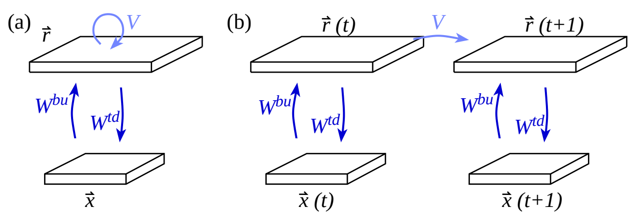

Fig. 1. The hybrid generative and predictive model. In (a) they are displayed as an architectural view and in (b) unfolded in time, over two time steps. The generative model consists of top-down weights W^{td} and it generates the input $\vec{x}$ from the hidden code $\vec{r}$ at any time step. Bottom-up weights W^{bu} are used to obtain the hidden code. The predictive model consists of lateral weights V and it predicts the hidden code at the next time step $\vec{r}(t+1)$ from the current hidden code $\vec{r}(t)$.

the input data and can trace back the input to meaningful, independent components. This constitutes an internal world model of the environment since the input can be generated from that hidden code. Fig. 1(a)) shows the model architecture (together with the predictive model, described next). The input $\vec{x}$ is the perception-action pair. It can be generated via the top-down weights W^{td} from the hidden representation $\vec{r}$. Perceptive, bottom-up weights W^{bu} are needed to obtain the hidden representation $\vec{r}$ from the data $\vec{x}$. Weights are trained via an unsupervised learning scheme described below.

The generative model (consisting of W^{bu} and W^{td}) can be used for input–output mappings as we have done recently on a Kohonen network (Wermter & Elshaw, 2003). For this purpose its input $\vec{x}$ is logically split into a perceptive input $\vec{y}$ and a motor output $\vec{m}$, which can be expressed as $\vec{x}=(\vec{y},\vec{m})$. If we now test the model using 'incomplete' input $(\vec{y},0)$ where there is perception but zero motor action, we can still find a hidden representation $\vec{r}'$ using W^{bu}. Based on $\vec{r}'$ the model can generate a virtual input $\vec{x}'=(\vec{y}',\vec{m}')$ using W^{td}. This motor representation $\vec{m}'$ is then the most appropriate action, which belongs to the perception $\vec{y}$. Having learnt perception-action mappings, the model can automatically generate action sequences within an environment, because an action will lead to a new perception, which will itself trigger an appropriate action.

1.2. A predictive model of the motor cortex

A predictive model generates its future input or its future state, unlike a generative model, which generates its current input. An advantage of a predictive model is that it can compensate for a delay of its input. Furthermore, since the consequences of an action are known without that the action is actually performed, a predictive model allows for mental simulations of action sequences (Oztop, Wolpert, & Kawato, 2003). Prediction allows the observer to determine if the actor is helpful, unhelpful or threatening (Gallese & Goldman, 1998) and offers the opportunity to react to an action sequence before it has ended (Demiris & Hayes, 2002).

The hierarchically directed weights W^{bu} and W^{td} which constitute the generative model generate the incoming data, which enters the model at the same time instance. On the other hand, the weights V of our predictive model are directed primarily horizontally, or laterally. As can be seen in Fig. 1(a), these weights are recurrent connections within one neural sheet. Fig. 1(b) which depicts the model unfolded over two time steps illustrates that V connect hidden representations at different time steps—making them hetero-associative weights. Note that a trained generative model is needed to obtain the hidden representations, which the predictive model uses for training and for action.

Our model choice to separate feature extraction (the generative model) from prediction originates from modelling constraints in the visual cortex. Simple cell-like localised edge detectors in W are obtained by a training paradigm where the same image has to be reconstructed from the hidden representation, which gave rise to it, but not if, for example, a shifted version of that image should be reconstructed. The prediction task was therefore separated out to lateral weights V, which predict the next hidden representation $\vec{r}(t+1)$ rather than directly the next input data point. At each relaxation time step, the input images had been moved slightly into a random direction. As a consequence the lateral weights predicted a hidden code that was slightly shift invariant, yielding V1 complex cell properties (Weber, 2001).

Lateral weights V have been extended to connect also different cortical areas, in order to associate 'what'- and 'where' representations of an object (Weber & Wermter, 2003). Here we will extend them so that they are allowed to connect to all areas involved, including the input areas (as shown later in Fig. 4). They can then directly predict the future input (consisting of perception and action) and therefore anticipate a learnt motor action. This corresponds to the findings that long-range horizontal connections in the cortex (which originate as well as terminate in layers ⅔) are also found between areas of different hierarchical levels, even though they are strongest within one hierarchical level (Felleman & Van Essen, 1991).

Such a two-tier architecture of feature extracting cells with a layer designed to compute invariances on top resembles the Neocognitron of Fukushima et al. (1983), a biologically inspired model of visual pattern recognition where several hierarchical levels are arranged successively. A more complex architecture is the Helmholtz machine through time by Hinton, Dayan, To, and Neal (1995a). Its recognition component (bottom–up connections) and generative component (top-down connections) are each completed by additional lateral connections which are trained to predict a future state. While such lateral connections yielded shift invariance in (Weber, 2001), contrasting models achieve invariances by vertical connections (e.g. Riesenhuber & Poggio, 2002).

Taking the generative and the predictive model together, we are proposing such a two-tier architecture for a motor cortical area. As in the Helmholtz machine through time we train the lateral connections to predict the future activations on the hidden layer. Since this is done using recurrent relaxations of activations, this introduces the competition as between the attractors of a continuous attractor network. In contrast to a maximum operator or a convolution with a fixed kernel, this is a soft competition involving trained, irregular weights. Having applied our model to the visual system we will here demonstrate its ability to perform sensation driven motor control and mental action simulation in a real world scenario.

2. Methods

In the following we will briefly describe the task, and then a reinforcement trained model, which can solve the task. Next, we will introduce the model of the motor cortex and explain its components, the generative model and the predictive model, which are trained to copy the solution to the task from the reinforcement trained model.

2.1. Task

The scenario is to dock a robot to a table so that it can grasp a visually perceived object on it (Fig. 2). The camera is fixed to the robot, so the task is accomplished if the target is perceived at a certain location in the visual field. In addition, however, for geometrical reasons the robot has to approach the table at a perpendicular angle (defined below as 0°), so that its short gripper can reach over the table (see Fig. 2). For simplicity, the gripper's height is constant just above table level. The two-wheeled robot can perform 'go forward', 'go backward', 'turn left' and 'turn right' motor commands.

A simulator for the environment was implemented which computes the future perceived target position vector $\vec{p}$ and robot heading direction encoded in $\vec{\Phi}$ based on present values and on the motor command $\vec{m}$. In previous work, a network trained on this simulator with a reinforcement scheme was able to control a real robot in the corresponding task (Weber et al., 2004). Here too, we rely on the simulator for training but run demonstrations also on the robot.

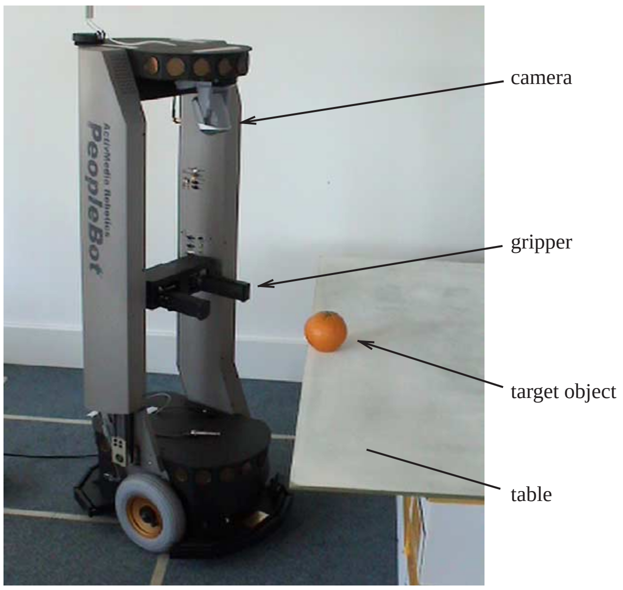

Fig. 2. The robot performing the 'docking' action. Note that it cannot reach the orange fruit if it approaches the table at an angle, because of its short gripper and its side columns. This corresponds perhaps to situations where fingers, arm or hand are in an inappropriate position for grasping.

2.2. Reinforcement model structure

The network, which performs the docking action, is depicted in Fig. 3. Here, we will give a short account of how it was trained, as we have described in detail in Weber et al. (2004). Learning consists of three steps.

First, the lower-level vision module's weights W^{bu} and W^{td} are trained from natural colour image patches using a sparse coding Helmholtz machine (the names W^{bu} and W^{td} here are intended only for this context, however, the model is basically the same as the generative model of the motor cortex for which we again use these names). As a result, the input image $\vec{I}$ is coded as a hidden code $\vec{u}$ by units, which are similar to V1 simple cells (localised edge detectors, some colour selective cells, retino-topography).

Second, lateral weights V are trained to generate a set of continuous attractors, which approximate the common representation $(\vec{u}, \vec{p})$. For this task, a specific object of interest (an orange-coloured disk, corresponding to an orange fruit) is artificially inserted into each image $\vec{I}$ while $\vec{p}$ is constructed to represent a Gaussian hill of activity at the corresponding location of the object ($\vec{p}$ is the same size as $\vec{I}$, both are vectors of 24×16 elements, arranged two-dimensionally). As a result, whenever an image is presented with an object of interest and when $\vec{p}$ is initially unknown, then a best matching continuous attractor will arise in which $\vec{p}$ reflects the visually perceived position of the object. This irregular hill of activation is then replaced by a Gaussian around the maximum value, but for simplicity we will reuse the symbol $\vec{p}$ for this.

Third, $\vec{p}$ is one of two inputs to the module, which is trained using reinforcement. The other input is $\vec{\Phi}$, a vector with 7 elements which represent 15° increments in the robot heading direction angle from -45 to 45°, where 0° is suitable for grasping. The elements are arranged one-dimensionally to encode a Gaussian profile around the robot direction angle. The robot angle is given by its odometry system and $\vec{\Phi}$ might be encoded in head direction cells in a biological system. Since different combinations of $\vec{p}$ and $\vec{\Phi}$ make up different states, we multiply these by their outer product $\vec{f} = \vec{p} * \vec{\Phi}$ to obtain a $24\times16\times7$-dimensional state vector, depicted in Fig. 3 as a three-dimensional block. $\vec{f}$ encodes one Gaussian hill of activity where neighbourhood is defined by the three dimensions of that block. It is an almost localist code since it is roughly defined by the most active unit.

During reinforcement learning a critic unit assigns any given state a value c, which reflects the future expected rewards (Foster, Morris, & Dayan, 2000). A 'reward' is delivered only at the end of each trial. It is positive when the target is perceived at the robot's gripper and if the robot heading direction angle is approximately zero. The 'reward' is negative when the target is lost out of sight or when the robot hits the table. The mapping of the state $\vec{f}$ to the critic value c via the weights W^{c} constitutes the value function. After training it increases if the robot gets nearer to its target.

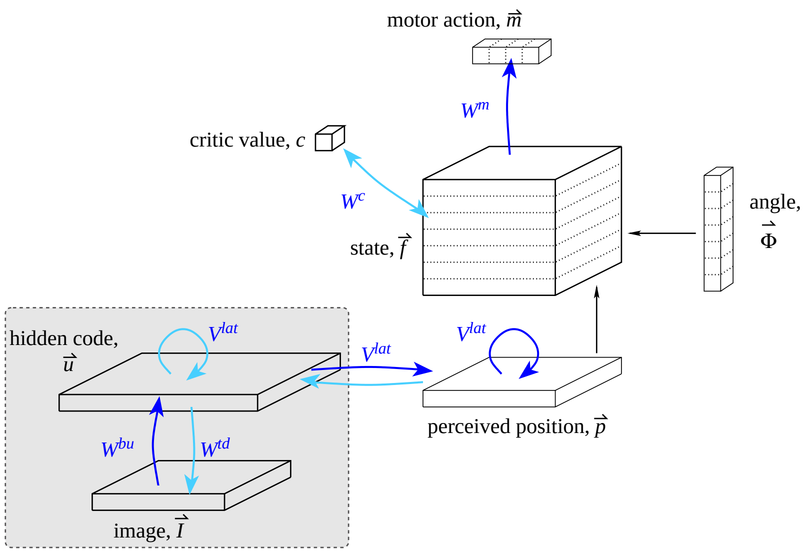

Fig. 3. The network which performs the docking action based on reinforcement training (after Weber et al., 2004). $\vec{I}$ is the camera image. $\vec{u}$ is the hidden representation ('what') of the image. $\vec{p}$ contains the perceived location ('where') of the target within the image. $\vec{\Phi}$ has a Gaussian profile centred on the angle of the robot. $\vec{f}$ is the perceptual space, made up by multiplying $\vec{p}$ and $\vec{\Phi}$. c, the critic, holds the value function which is assigned to each perceptual state $\vec{f}$ unit. $\vec{m}$ are the four motor unit activations. Thick arrows denote trained weights W and V. Only the ones depicted dark are used during performance while those depicted light are involved in training. The lower level vision module is shaded to highlight its similarity to the generative model of Figs. 1 and 4.

The critic is needed only to train the motor weights W^m which are trained such that the motor actions given by $\vec{m}$ maximise the value c. A motor action at any time step is defined by the maximum of the four values of $\vec{m}$ which represent the robot movements 'go forward', 'go backward', 'turn left' and 'turn right'.

While this network performs the docking action, the target representation $\vec{p}$, the angle vector Φ and the motor action $\vec{m}$ are used to train the motor cortical network, described next.

2.3. The cortical model

The model has two logical constituents. First, feature extraction via weights W of the generative model gives a higher level representation $\vec{r}$ of its input. Second, hetero-association via predictive weights V predicts the representation $\vec{r}(t+1)$ at the next time step given the current representation $\vec{r}(t)$. The weights W are trained via unsupervised learning, because the higher level representation $\vec{r}$ is initially not given at any time. The weights V are trained afterwards to predict $\vec{r}(t+1)$ from $\vec{r}(t)$. Since, $\vec{r}$ is then already given at all times, this resembles supervised learning.

Let us for now assume that a lower level (basal ganglia) model already exists which produces the desired interactive action sequences. The model of the motor cortex which shall learn to generate these sequences is shown in Fig. 4. During lower-level sequence production it receives bottom-up input from perceived vision $\vec{p}$, the heading angle coded in $\vec{\Phi}$ and the motor action $\vec{m}$. It produces a hidden representation $\vec{r}$ from its bottom-up input using weights W. Prediction of future hidden and input states is done by the weights V.

2.3.1. Training the generative model

The unsupervised learning scheme is that of the wake–sleep algorithm (Hinton, Dayan, Frey, and Neal, 1995b) used to approximate training of a Helmholtz machine (Dayan, Hinton, Neal, & Zemel, 1995). It alternates between the wake phase in which one data point is presented to the network and the sleep phase which does not require any data.

The wake phase is used to train the generative, top-down weights W^{td}. First, a data point $\vec{x}=(\vec{p},\vec{\Phi},\vec{m})$ which logically concatenates the three input areas is presented. Then, the hidden code is computed:

$$\vec{r} = g\left(W^{bu}\vec{x}\right) \quad \text{with} \quad g(a) := s\frac{e^{\beta a}}{e^{\beta a}+k}$$

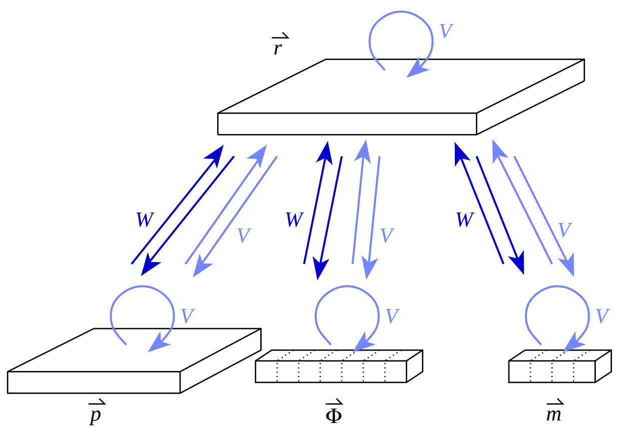

Fig. 4. The hybrid generative and predictive model of the motor cortex applied to the docking problem. The generative model consists of weights W (where W summarises W^{bu} and W^{td}), the predictive model of weights V. $\vec{r}$ is the hidden representation. The inputs $\vec{p}$, Φ and $\vec{m}$ are described in Fig. 3.

where the logistic, positive-only transfer function g is applied at each hidden unit. It has parameters $s=1$ which controls the maximum output (saturation), $\beta=2$ which controls the slope and $k=64$ which if made larger decreases the output value at a given small input and thus promotes sparseness in the representation $\vec{r}$. The product $W^{\mathrm{bu}}\vec{x}$ is between the concatenated bottom-up weights from the three input areas to the hidden area and the concatenated inputs. As the next step, the reconstructed input $\vec{x}'$ is generated from the hidden code:

$$\vec{x}' = W^{\mathrm{td}}\vec{r} \tag{1}$$

where similar to above, W^{td} denotes the concatenated top-down weight matrices. Then, Hebbian-type delta learning is done using as post-synaptic term the difference between the original and the reconstructed data:

$$\Delta W^{\mathrm{td}} = \eta\left(\vec{x}-\vec{x}'\right)*\vec{r} \tag{2}$$

with learning rate $\eta=0.001$.

The sleep phase is used to train the recognition, bottom-up weights W^{bu}. It mirrors the wake phase with $\vec{x}$ and $\vec{r}$ exchanged. First, a 'fantasy' hidden code $\hat{r}$ is generated. It is binary: most elements are zero and a few are set to 0.75 (the value 0.75 can easily be reconstructed by the transfer function g unlike the value 1 below which g saturates). These are selected randomly under the envelope of a randomly placed Gaussian to impose a functional topographical relationship. Then, the imagined input which belongs to the fantasy hidden code is computed:

$$\hat{x} = W^{\mathrm{td}}\hat{r}.$$

No additional transfer function is applied on the inputs. As the next step, the reconstructed fantasy hidden code is obtained from the imagined input:

$$\hat{r}' = g\left(W^{\mathrm{bu}}\hat{x}\right)$$

where g is the same transfer function with its parameters as in the wake phase. Then, Hebbian-type delta learning is done using the difference between the fantasy—and the reconstructed hidden code:

$$\Delta W^{\mathrm{bu}} = \eta\left(\hat{r}-\hat{r}'\right)*\hat{x} \tag{3}$$

with the same learning rate η as above.

There are additional constraints on the weights. (i) All weights W^{bu} and W^{td} are rectified to have only positive values. This reflects the fact that the data $\vec{x}$ and hidden code $\vec{r}$ have positive-only values and excludes subtractive components in the construction of these vectors via the weights. (ii) A weight regularisation term which should prevent a few single weights from becoming very large adds a small weight decay of $-0.015\,\eta w$ on every weight w in Eqs. (2) and (3).

2.3.2. Training the predictive model

The predictive model functions like an associator neural network: the current activations of the network units lead via a (almost) full recurrent connectivity to the next activations, and so on. The connectivity structure is shown in Fig. 4: there are predictive weights V within every area as well as between the hidden area and every input area. We omitted direct weights between different input areas in order to save computation time. For convenience of notation, let us define V the full weight matrix between all areas, with the entries of V where defined and with zeros where there are no connections. Furthermore, let us concatenate all areas' activations to one large vector $\vec{z}(t) = (\vec{p}(t), \vec{\Phi}(t), \vec{m}(t), \vec{r}(t))$. Then one step of the dynamics of the predictive model is:

$$\tilde{z}(t+1) = g(V\tilde{z}(t)) \tag{4}$$

where the transfer function g is the same as above, where parameters $s=0.7$, $\beta=2$ and $k=8$ aid maintain recurrent activations. We use the 'tilde' symbol for the vector $\tilde{z}$ which is obtained by the predictive weights in order to distinguish it from $\vec{z}$ which is given by the data.

The predictive model is logically trained after the generative model, because the correct hidden code must be known for training. However, for convenience of implementation, we applied a learning step of the predictive model after every learning step of the generative model (wake- and sleep phase) throughout training. The values $\vec{z}$ are taken from the wake phases.

Note that two consecutive data points are not independent: since goal-directed action sequences are performed during acquisition of the data, a data point $\vec{z}(t)$ will lead to a causally linked consequence $\vec{z}(t+1)$. The prediction has as parameters (i) the progress in time, i.e. how much does the physical world progress between time steps t and $t+1$ and (ii) the number of neural updates made in this time. Assuming that data are continuously available, we can choose a progression in time that is sufficiently small to be aligned with just one neural update step, speeding up training and avoiding issues of unstable attractor dynamics during learning. The repetitive relaxation of predicted values of Eq. (4) is thus replaced by a one-step prediction at each time step:

$$\tilde{z}(t+1) = g(V\vec{z}(t)).$$

The learning rule is:

$$\Delta V = \eta(\vec{z}(t+1) - \tilde{z}(t+1))*\vec{z}(t) \tag{5}$$

with learning rate $\eta=0.01$. Thus, in each training step, two consecutive time steps are involved: the first for the given pre-synaptic value $\vec{z}(t)$ and the second for the post-synaptic difference between the given target value $\vec{z}(t+1)$ and the predicted value $\tilde{z}(t+1)$. Since $\tilde{z}(t+1)$ saturated at 0.7 due to its transfer function parameter s, we constrained the target values $\vec{z}(t+1)$ at the visible components $\vec{p}$, $\vec{\Phi}$ and $\vec{m}$ (but not $\vec{r}$) to not exceed 0.8.

Additional constraints on the weights V are: (i) a weight decay term is applied which adds the term $-\eta\ 0.001\ v$ for every weight v in Eq. (5) and (ii) self-connections are set to zero.

In contrary to the generative model weights W, we allow weights V to be negative, so that they can maintain an activity pattern as a recurrent attractor network. For example, a centre-excitatory and surround-inhibitory weight structure (in feature- or geometrical space) can maintain a localised patch of activity.

2.4. Implementation

One million training steps, each consisting of a wake phase and a sleep phase, were done and the learning step size η was reduced linearly to zero during the last quarter of training, long after no weight changes were visible any more. At each step, the module previously trained using reinforcements (Weber et al., 2004) performed a motor action dependent on its input. After on average about 20 steps the goal was reached (or a border of the visual field was hit by the perceived object) and the robot was placed at a new random continuous position, with initial rotation angle zero and so that the target object was inside the visual field. Transitions between sequences were not specifically treated, thus if $\vec{z}(t)$ represents a goal state then $\vec{z}(t+1)$ is unrelated, because it represents a new random initial position. The speed of the robot was set such that the perceived object moved along a distance of 0.9 pixels during one step, or, if it turned, then by 18° to the left or to the right.

The size of the three input areas matches by the underlying reinforcement-trained model: the perceived location $\vec{p}$ is represented on a grid of 24×16 units, the robot heading angle is represented on seven units and there are four motor units. For the hidden area we chose a size of 16×16 units. $\vec{p}$ has the shape of a two-dimensional Gaussian with $\sigma=1.5$ and height 1 on the grid position of the target; $\vec{\Phi}$ has the shape of a one-dimensional Gaussian with $\sigma=0.5$ and height 2 centered according to the robot rotation angle; $\vec{m}$ is 1 for the active motor unit and zero else. Because $\vec{m}$ is relatively small, the input is not very much distorted if all units are set to zero as in the testing condition. Furthermore, no bias is introduced as all motor units have the same magnitude. One might, however, compensate for the missing input by setting all four values to 0.25. Training took roughly one day on a 2.2 GHz Intel Pentium 4 Linux machine, but performance is fast enough for real-time application.

3. Results

3.1. Anatomy of the generative model

As a result of training, Fig. 5(a–c) together show the recognition weights W^{bu}. They are made up of the three components to receive the $\vec{p}$, $\vec{\Phi}$ and $\vec{m}$ and components of the input, respectively. Adjacent neurons on the hidden area have similar receptive fields in any of the three input areas. This realises topographic mappings. Accordingly, there are regions on the hidden area which are specialised to receive input from similar regions in $\vec{p}$-space, $\vec{\Phi}$-space and $\vec{m}$-space. The central region in $\vec{p}$-space is represented by more neurons than peripheral regions, because according to the turning and movement of the simulated robot toward the target, the target was more often perceived centrally during training. While each hidden neuron codes only one angle and only one action, a few neurons shown at the top right of Fig. 5(a) with bi-lobed receptive fields code for target object positions at the lower right and also at the lower left of the visual field. These neurons have weak angle input, as the missing weights there in Fig. 5(b) suggest, and code for backward movement, Fig. 5(c). They obviously represent the situation where the robot is close to the table, but with the target object at the left or the right, so it has to move back before doing further grasping action.

The component of the generative weight matrix W^{td} which projects to the four motor units is shown in Fig. 5(d). Their receptive fields in the hidden area originate from several sub-regions, except for the second unit which codes the 'go backward' command and which receives its input from one single region in the hidden area. Since W^{td} and W^{bu} invert each other functionally, the transpose of W^{bu} looks very much like W^{td}, and accordingly, the structure of Fig. 5(c) can be easily matched with Fig. 5(d). We can see easily in Fig. 5(d) that the neurons in the upper right of the hidden area are the only cluster coding for backward movement.

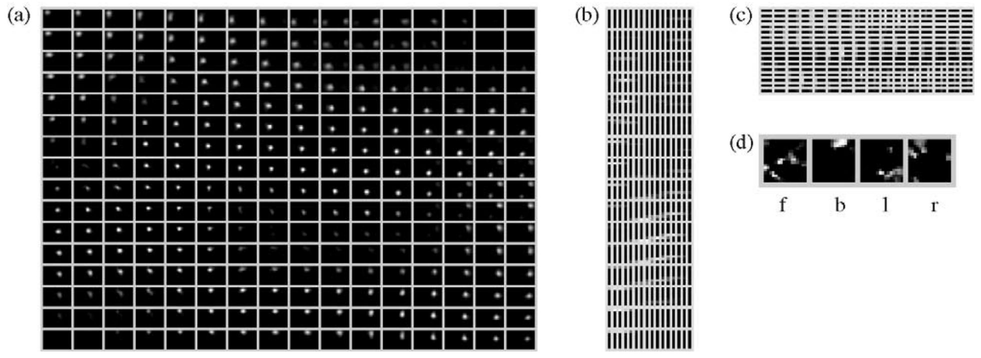

Fig. 5. A selection of trained weights W of the generative model. Each little rectangle shows the receptive field of one neuron in its respective input area. Black denotes zero- and white strong connection strengths. (a) Shows recognition (bottom-up) weights from the perceived vision area to the 16×16-unit hidden area, (b) those from the angle area (the 7 angle units within each tiny rectangle displayed vertically arranged) and (c) those from the motor area (the 4 motor units within each tiny rectangle horizontally arranged) to the hidden area, respectively. (d) Shows the generative (top-down) weights from the hidden area to the four units of the motor area, which functionally invert the weights shown in (c). The four motor units are labelled for 'forward', 'backward', 'left turn' and 'right turn'.

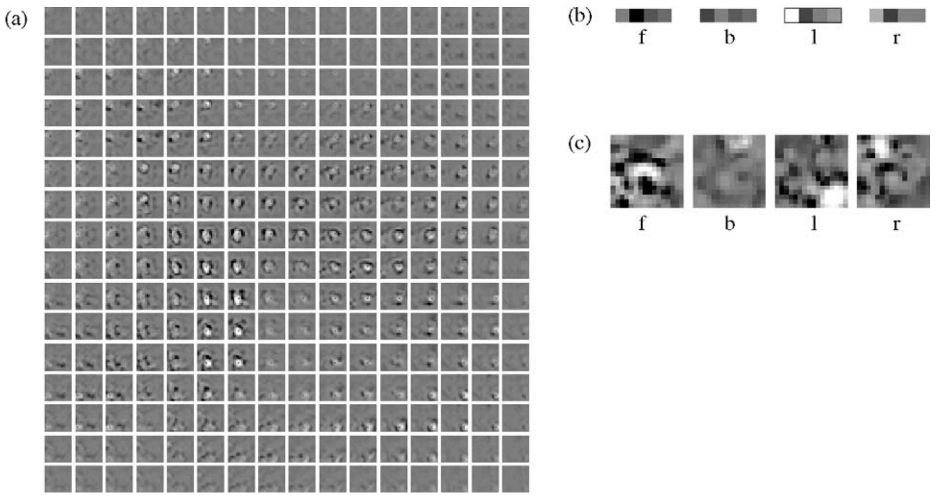

Fig. 6. A selection of trained weights V of the predictive model. Negative weights are depicted as dark, positive weights as light. In (a) and (c) contrast is enhanced so that weights larger than 0.5 times the maximum value are white; weights smaller than 0.5 times the minimum are depicted black. (a) Shows inner-area lateral weights of the 16×16-unit hidden area. Their irregular centre-excitatory, surround-inhibitory structure produces shifting activation patterns $\tilde{r}$ as shown in Fig. 10. (b) shows inner-area weights of the motor area. Note that self-connections in (a) and (b) were set to zero. (c) Shows predictive weights from the hidden area to the four motor units. These can be loosely matched to the feedback, generative connections which area shown in Fig. 5 (d) positive weights originate in similar regions of the hidden area.

3.2. Anatomy of the predictive model

The lateral predictive weights within the hidden area shown in Fig. 6(a). They have developed an approximate centre-excitatory surround-inhibitory structure, an organisation principle found across the entire cortex. The structure is not regular and symmetric but the inhibitory surround has a directional bias. This produces activation patterns that have a slight offset to the existing patterns, which will result in their movement. Lateral weights within the motor area are shown in Fig. 6(b). Here, too, the weights are not symmetric, i.e. $V_{ij} \neq V_{ji}$ for weights between units i and j. For example, the 'go forward' unit receives only negative connections from the three other units while the 'left turn' and 'right turn' units each receive a positive connection from the 'go forward' unit. This reflects different probabilities between a transition from one state to another and a transition into the opposite direction.

The sample of predictive weights V from the hidden area to the motor area shown in Fig. 6(c) matches roughly the corresponding one from the generative matrix W, shown in Fig. 5(d). Differences are first, that the predictive weights have also negative components, which is necessary, because they need to maintain a line attractor. Secondly, the shapes also of the positive components of V differ slightly from those of W, because the predictive weights have to generate a slightly different activation pattern, which lies one step ahead in the future. Predictive weights are more blurred, probably because the network is unable to predict the exact future state but must average between several possible future states.

3.3. Error curves

Fig. 7 shows the errors of the generative and predictive weights during training in the case of when the motor output is fed into the network (as during training) and in the case without motor input (as during performance). The errors of the generative weights W are considerably larger when the correct motor input is missing. In case of the predictive weights, the values given as input were those of the previous time step only, including the previous correct motor values. The error of the predictive weights V is also larger without motor input.

3.4. Actions and mental simulation

As a result of training, the robot undertakes one of the four possible actions, 'go forward', 'go backward', 'turn left' and 'turn right' dependent on its relative position to the target $\vec{p}$ and its heading angle $\vec{\Phi}$. Fig. 8 shows these actions as flow fields for

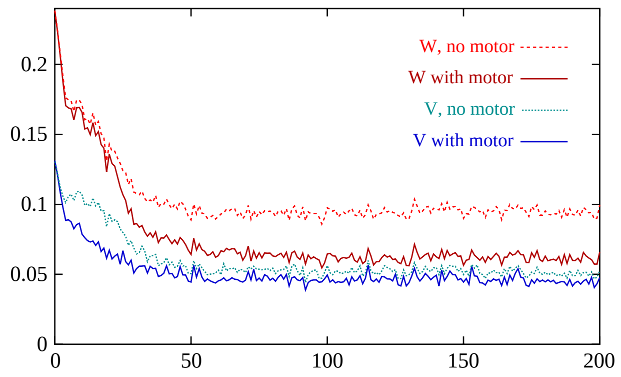

Fig. 7. Errors during the training progress. Each point along the x-axis denotes 1000 sampling points over which the average squared errors, y-axis, over the 4 motor units' outputs were averaged. From top to bottom, errors made by: generative weights W if they have not received motor input, generative weights after receiving full input including the correct motor values, predictive weights V which have received only sensory input $\vec{p}$ and $\vec{\Phi}$ at the previous time step and predictive weights which have received full input at the previous time step. Note that the predictive model's output using the transfer function of Eq. (4) cannot be directly compared to the generative model's linear reconstruction of Eq. (1).

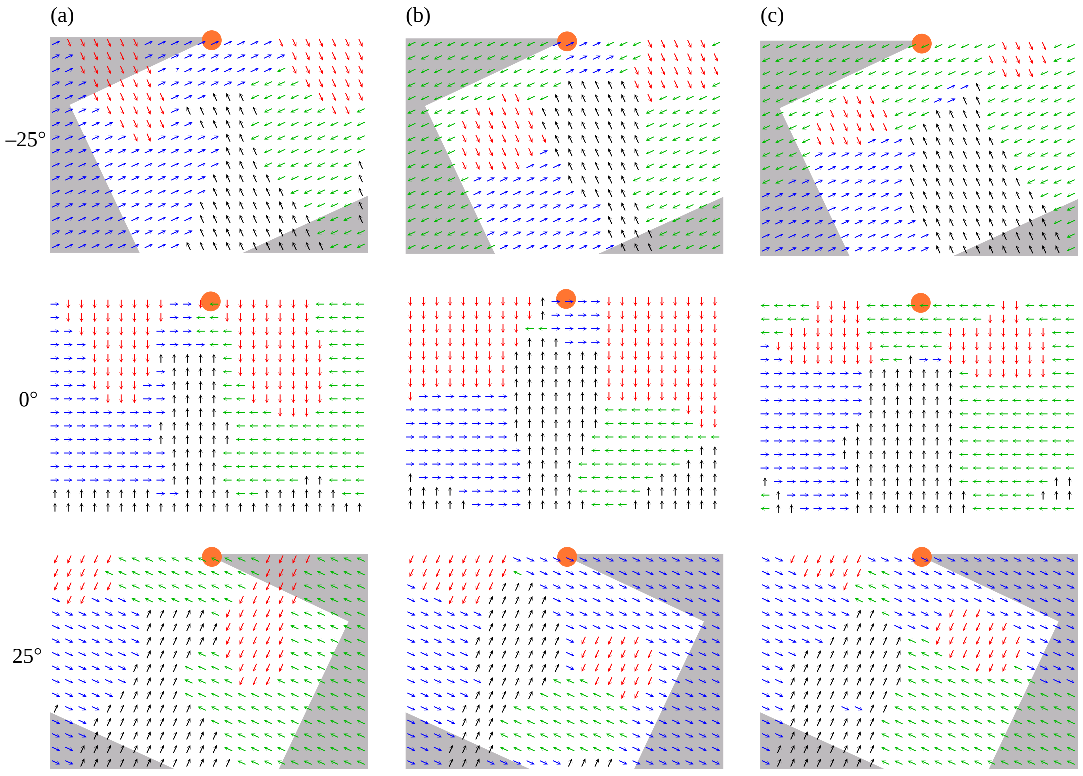

Fig. 8. Mappings from state space to actions. In each field, the *x*, *y*-plane denotes the location of the robot; the target location is depicted as a disk at the centre of the upper edge. Action fields are depicted for three different robot heading angles, −25, 0 and 25°. In the grey shaded regions, the robot would not see the target, because it is outside of the field of vision. These regions have been omitted during training. The arrows at each point denote the robot action: the arrow facing in the robot heading direction (primarily upward) corresponds to 'go forward', the downward arrow to 'go backward', the right arrow corresponds to 'turn right', and the left arrow to 'turn left'. (a) For the reinforcement-trained module, (b), for the generative motor cortex model, *W* and (c) for the predictive model, *V*.

three heading angles. In column (a) they are depicted for the reinforcement-trained module which served as teacher for the motor cortex module. Column (b) shows these for the generative cortical model and column (c) for the predictive model. The actions depicted are defined by that motor unit among $\vec{m}$ which was strongest activated by the hidden code via W^{td} (or V in case of the predictive model), after the hidden code was obtained from the 'incomplete' input $\vec{x}=(\vec{p},\vec{\Phi},(\vec{m}=0))$ via W^{bu}. Actions thereby defined are deterministic given the input. The flow fields of the generative and the predictive models match those of the reinforcement-trained module in large parts near the centre but differ in some surrounding areas. Training was done while the simulated robot performed docking actions; thus training data occurred more often near the centre where the robot path would lead than near the edges of the field. No training was done when the target was not visible (grey shaded regions in Fig. 8) and the models differ considerably here.

Based on Fig. 8 we counted how often the actions of the generative and predictive models would match those of the reinforcement-trained module. Across the three depicted angles, there are 948 data points which fall inside the white regions, which received training. There the generative model performs in 59.92% of all cases similar to the reinforcement-trained module (at angle 0° the number is 67.45%), the predictive model performs in 60.23% of cases similarly (at angle 0°: 64.32%). The models perform thus better in regions which the robot has visited more often during training (e.g. it was always initially placed at an angle of 0°). 204 remaining data points are in shaded regions where there was no training. There the generative model would perform correctly only in 14.71% of the cases (random decisions would lead to 25% correspondency). Note however that the reinforcement-trained module is itself a neural network which was not trained in these regions. Since it has a different architecture the models are expected to generalise differently and it is not defined which model is 'correct'. With 47.06% similar performance, the predictive model generalises into these regions more similarly to the reinforcement-trained module.

Examples of docking sequences are shown in Fig. 9. The generative model sequences in (b) were obtained similarly to the reinforcement-trained sequences depicted in (a): at each point in $(\vec{p},\vec{\Phi})$-space, the motor action was obtained from the maximum active of the four motor units and the simulated robot was moved accordingly. The resulting movement sequences of both models are largely similar. The first depicted example is unusual in that the generative model even outperforms its teacher: while the reinforcement-trained module utilises an unsuccessful strategy where the robot will hit the table with its left side, the generative model decides to move a little back before turning.

The third example of Fig. 9 shows some limitations in the model's capabilities: after initial good performance, the

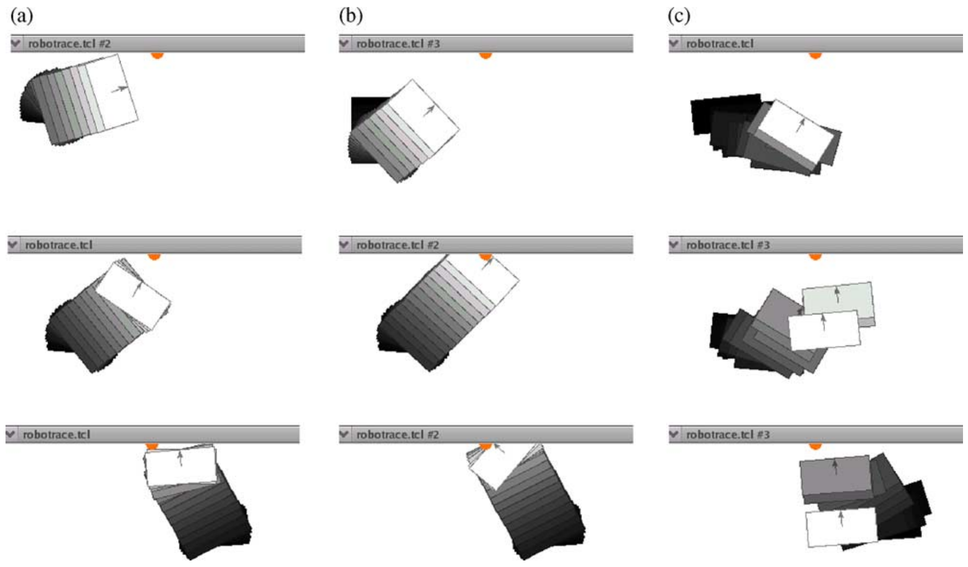

Fig. 9. Action sequences. In each row, from a given starting point, 20 consecutive time steps are displayed such that the positions of the robot at later times are displayed in lighter shades. (a) 'Correct' sequences performed by the reinforcement-trained module, interacting with the environment. (b) Reconstruction: learnt sequences performed by W^{td}, interacting with the environment. (c) Mental simulation: imagined sequences 'performed' by relaxation via V given the starting point, but without any further feedback from the environment (see also Fig. 10). For display, the strongest activations of $\vec{p}$ and $\vec{\Phi}$ at each time step of the relaxation were taken to project the mental state into the 'real world'.

simulated robot bumps into the table and then continues turning into it. In that region of the state space, of course, there was no training data. Note also that the generative model simply copies an action without knowing about a goal or a reinforcement signal. Finally, the learning rule Eq. (2) minimises the squared error between each component of the generative model's output vector and the values given by the data, while the action is defined differently, by the maximum value of $\vec{m}$.

Projections of the predictive model's 'mental simulations' into the real world are shown in Fig. 9(c). It predicts the robot to move toward the goal for several time steps even in total absence of any further input from the real world. Since the imagined future $(\tilde{p}, \tilde{\Phi})$-states are not corrected by the environment's reaction to the predictive motor action $\tilde{m}$, the imagined movement is not physically constrained: side shifts of the robot are possible in its imagination.

3.5. Physiology

Neuronal activations during 'mental simulation' of the third sequence of Fig. 9(c) are shown in Fig. 10. All activations $(\tilde{p}, \tilde{\Phi}, \tilde{r}, \tilde{m})$ together form one non-stationary line attractor in time, governed by repetitive application of Eq. (4). In the depicted example, $\tilde{p}$ and $\tilde{r}$ die out after about 12 iteration steps. When $\tilde{p}$ is small, the maximally active unit remains somewhere in the centre, as the final imagined robotic positions which are depicted white in Fig. 9(c) suggest. $\tilde{p}$ usually dies out eventually. This is probably due to unpredictability of the perceived location by the model: in the shown example, the target is perceived near the goal position at time step 12. During training, a new random starting position which cannot be predicted was selected whenever the goal was reached. Furthermore, $\tilde{p}$ is mainly kept 'alive' by the intra-area lateral weights V within the area which represents $\vec{p}$. These do not consider the robot heading angle $\vec{\Phi}$ and can only predict an average outcome by which an originally Gaussian hill of activation will be flattened. Eventually, it cannot maintain itself any more.

A remedy against $\tilde{p}$ dying out in time may be simply not to implement any lateral weights V within this area, and possibly not within the other input areas as well. We have tried this out (results not shown) and decided against it, because the activations on the $\vec{p}$-area would not be refined to a narrow Gaussian, but become broadly distributed over the whole input area.

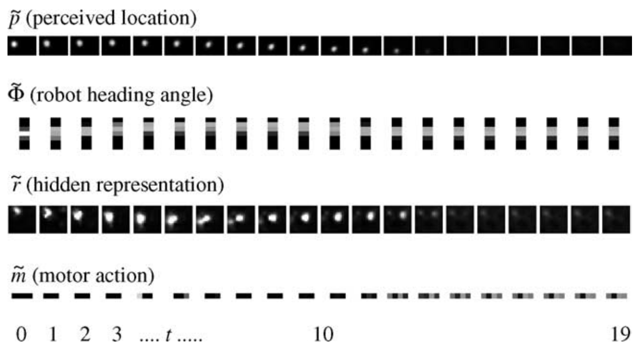

Fig. 10. Activation sequences during mental simulation. After initialisation with $\vec{p}, \vec{\Phi}, \vec{m} = 0$ and $\vec{r}$ (the latter computed using W^{bu} weights) at time $t=0$ the activation sequences at 'future' time steps, produced by the predictive weights V and counted by the numbers below, are shown. Black denotes zero activation, white strong activation. The depicted area sizes are at different scales (the $\vec{p}$-area is reduced most) for display reasons. This example sequence belongs to the one shown in the third row of Fig. 9(c).

3.6. Application to the robot

The network was run on the PeopleBot robot (Fig. 2) where input $\vec{\Phi}$ was retrieved from the robot odometry and $\vec{p}$ from its camera using the localisation algorithm described in Weber & Wermter (2003). Gripper action was not considered. Example movies of the robot using the generative weights W can be downloaded at the following URL: http://www.his.sunderland.ac.uk/supplements/NN04/.

Variations in the robotic movement arise first from the cortical visual target localisation network, second, from the reinforcement-trained teacher network and finally, from the motor cortex learner network. Shortcomings w.r.t. the teacher network (Weber et al., 2004) are visible in the final phase when the robot adjusts itself insufficiently parallel to the table. It would be conceivable that the reinforcement network remains active and plastic in such phases, countervailing the cortex' deviations until the cortex can succeed.

4. Discussion

In addition to generative models already explaining sensory cortical areas very well, we have demonstrated here that a generative model can account also for motor cortical function. It relies on a working module trained by reinforcement which resides outside of the cortex. A generative motor theory forms a bridge between a generative sensor theory and a reinforcement based motor theory. It is important to have such a link, since sensory and motor representations are largely overlapping in the brain.

4.1. From reptilian dorsal cortex to the mammalian cortex

During evolution, too, it would be an advantage to evolve sensory and motor systems together, since they must always remain linked together. The six-layered mammalian cortex might have evolved from the reptilian dorsal cortex, a more primitive structure with three layers (for a review see Aboitiz, Morales, & Montiel, 2003). If the mammalian cortex functions as a generative model, we may ask, to what degree is the reptilian dorsal cortex already such a model?

Strictly speaking, a generative model must have feedback connections from the hidden layer to the input layer. They allow to reconstruct the input from the hidden code and obtain the reconstruction error which can be used for optimisation of the hidden code as well as for training. Models which develop localised edge detectors as found in V1 from natural image input have a recurrent connectivity between the simulated V1 and its thalamic input area, the lateral geniculate nucleus (LGN) (Olshausen & Field, 1997; Weber, 2001). An alternative are non-local (thus biologically implausible) learning rules like the standard independent component analysis (ICA) algorithm (Bell & Sejnowski, 1997) which effectively incorporate feedback effects. Biologically plausible models based solely on forward connections remain impaired.

Mammalian cortex allows for such feedback since areas are generally connected bi-directionally in an organised fashion (Felleman & Van Essen, 1991): neurons in layer 4 on a higher-level cortical area receive input from neurons in layers 2/3 on a lower-level area while feedback involves layers 5/6 on both areas. Does the reptilian dorsal cortex with its different layer structure allow for comparable feedback? It has been considered that layers 2, 3, 4 and possibly 5 of the mammalian cortex are newly acquired (Aboitiz et al., 2003). This is evidence for an evolved recurrent connectivity but it is linked to the hierarchical information processing of the mammalian cortex.

A statistically optimal model through the use of feedback may become important, if several processing steps follow in a series such as in hierarchical information processing. This implies that the advancement of the mammalian cortex is linked to its hierarchical arrangement of areas.

4.2. Imitation and mirror neurons

So far, our model is capable of self-imitation by copying the behaviour of another internal module, the reinforcement trained module. But it should also be able to explain the imitation of high level actions of higher developed mammals, if it receives appropriate inputs which enable the individual to understand the actions of another. These inputs would come from higher-level visual and auditory areas, where other individuals and their actions are recognised and in the case of humans also from language areas where actions are represented. In order for imitation to work, the perceived action of another individual and the own action need a common representation (see Sauser & Billard (2005) for a functional model achieving this by frame of reference transformations). This is found in mirror neurons in the motor area F5 of primates: they fire when the primate performs a specific action as well as when it sees or hears the action performed (Kohler, Keysers, Umilta, Fogassi, Gallese and Rizzolatti, 2002; Rizzolatti & Arbib, 1998).

The recognition of actions such as holding, hand grasping and tearing using the mirror system must involve as input to the system the perception of the goal or object; the motor system does not solely control movement (Gallese & Goldman, 1998). The importance of this goal-oriented component of the mirror neuron system can be understood from that the observer can still recognise high level actions even when the final section is hidden or the high level action understanding requires interaction with the actual object (Gallese & Goldman, 1998; Umilta, Kohler, Gallese, Fogassi, Fadiga and Keysers, 2001). The presence of the goal and its understanding allows the observer to share the internal state and representation of the performer of the action and so can imitate him (Rizzolatti & Luppino, 2001). The mirror neuron system was also observed in humans (Gallese & Goldman, 1998) and the association of some mirror neurons with Broca's area indicates a role in the evolution of language (Rizzolatti & Arbib, 1998).

4.3. Biological evidence for a two-stage learning process

In the following, we review evidence that the cortex might take over functions which have been previously acquired by the basal ganglia. Jog et al. (1999) demonstrated on rats in a T-maze task the role of the striatum, a large part of the basal ganglia, during learning (see Fig. 11). At early learning stages, neurons fired during the turn at the junction, i.e. while the rat executed the decision to turn left or right, in response to an acoustic signal. Later, during 'over-learning', responses went radically down at the turn, suggesting that the learnt decision is executed in other neural structures than the striatum. Interestingly, striatal neuronal activities increased at the beginning and at the end of the task, indicating (i) that the striatum may code for whole sequences of behaviour or (ii) that the striatum would at this point be ready to learn a more complex task, possibly comprising the one learnt by the motor cortex as a sub-task.

Brashers-Krug, Shadmehr, and Bizzi, (1996) have found evidence that newly learnt skills and consolidated skills are stored in separate parts of the brain: people had to acquire two conflicting motor maps for reaching movements with a robotic manipulandum, with conflicting (opposite) force fields applied. If the two situations were learnt with just minutes in between, then performance of both tasks was decreased wrt when just one task was learnt. If the two situations were learnt with more than 4 h in between, then the negative mutual influences were significantly decreased, suggesting that consolidated memory does not interfere with newly acquired skills. In a PET study (Shadmehr & Holcomb, 1997) monitoring regional cerebral blood flow, they associate memory consolidation with a shift from prefrontal regions of the cortex to premotor, posterior parietal and cerebellar structures. The striatum was not investigated here, but the above-mentioned studies of Jog et al. (1999) suggest its temporary involvement.

On the other hand, the orbitofrontal cortex (OFC) stores the value of olfactory/taste related rewards (Rolls, 2000), possibly learnt using its associative capabilities. OFC can thereby itself be involved in reinforcement learning while the striatum is involved in maintaining habits: rats with dorsolateral striatum

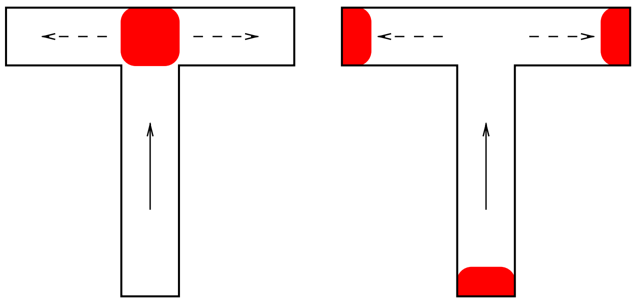

Fig. 11. Responses of basal ganglia neurons during early (left) and late (right) task acquisition phases in a T-maze. Activation levels were measured when the rat was at corresponding positions in the maze and high activation levels are depicted dark. Arrows indicate the directions in which the rat may move. At the beginning of learning, activity is strong if the rat is at the junction while at a late stage, activity is then strong at the beginning and at the end of the task, but not at the junction (drawn after Jog et al., 1999).

lesions which learn a conditioned lever-press for food, un-learn it better than non-lesioned animals if the food is devalued (Yin, Knowlton, & Balleine, 2004). The sensitivity to post-training devaluation might here be attributed to associations between stimulus properties that are learnt outside of the BG (Blundell, Hall, & Killcross, 2003). Functionalities beyond the simple motor primitive execution can also be found in the motor cortex in area F6 which has a modulating influence on F5 (Rizzolatti & Luppino, 2001). In contrary, the striatum may not be so easily controllable by (prefrontal) cortex (Ravel & Richmond, 2005), as its quick response in the experiments of Pasupathy & Miller (2005) suggests. Our model which addresses the learning of motor primitives as in neurons of F5, is therefore distinguished from habit formation in the basal ganglia on the one hand, as well as from task selection and higher functions in more frontal cortical regions on the other hand.

In the model of Hikosaka et al. (2002) the cortex basal ganglia loop and the cortex-cerebellar loop are both duplicated to represent a sequence of motor movements in two ways—one using a spatial coordinate system and another using a motor coordinate system. The former learns fast, accurately and in high awareness while the latter learns slowly, for speedy and unconscious performance. This addresses a shift of learning progress *within* each of the cortex, the basal ganglia and the cerebellum, that may coincide with a shift *between* the basal ganglia and the cortex, which we address.

4.4. Considerations about optimal reinforcement learning

One motivation for our research was to find a method that the large state space $\vec{f}$ would not be needed. Our model makes it obsolete, since the cortex model produces the action sequence as another mapping from sensory input to motor response. This would release the basal ganglia so that they can learn other sequences, however, only after these have been used during the initial training. The findings of Jog et al. (1999) which showed that striatal neuronal activities increased at the beginning and at the end of a learnt task suggest that the basal ganglia pay attention to more complex sequences which are composed of learnt sub-parts.

Another advantage could be that the cortex might be able to store a sequence more efficiently than the basal ganglia. We would suggest this from our robotic experiments where the reinforcement training paradigm (which we associate with the basal ganglia) requires a high-dimensional state space. This is because of localist coding in the state space by which every state is assigned one unit.

In order to adjust its strategy, the reinforcement network would need feedback about the motor cortex' progress. Then it may become inactive in situations where the motor cortex' behaviour parallels its action and the inactive neurons of the state space would become available to learn another task. Such a task can become more complex if the reinforcement network makes use of the motor primitives that the cortex has already acquired. It would then learn to select among these motor primitives—a function attributed in the adult to motor area F6 which has a modulatory influence on F5 neurons (Rizzolatti & Luppino, 2001).

If the action-selection region has access to the goal value, then setting the goal value can influence the model's behaviours. In a multi-modal model, the value might for example be set by a language instruction or by new obstacles which the robot faces during performance. Hence, the robot would behave in a teleological (Dickinson, 1985) rather than a mechanistic stimulus-response manner.

There exist other methods of circumventing that state space a priori. One approach is to approximate the critic's values as a learnt function of favourably chosen inputs. In our example the critic value c would be some function of input vectors $\vec{p}$ and $\vec{\Phi}$ which could even be reduced without loss of information to one three-dimensional vector (x,y,ϕ) of the perceived target location in Cartesian coordinates x, y and the robot heading angle ϕ. (The mapping to the motor action $\vec{m}$ technically does not need the state space, because the optimal motor action can always be inferred if the critic value is available. A critic's value function approximation therefore already renders the state space superfluous.) This approach can be prone to convergence problems, as pointed out by Boyan & Moore (1995), caused by small systematic errors in the approximating function and a resulting false action strategy.

Other methods reduce the state space to a minimum rather than avoiding it. The U Tree (McCallum, 1995), continuous U Tree (Uther & Veloso, 1998) and porti-game (Moore & Atkeson, 1995) algorithms translate a high-dimensional or continuous sensory state space to a low-dimensional discrete space. If the agent makes similar experiences in a large region of the state space (i.e. a constant critic value and similar actions), then this region is encouraged to be represented by just one state.

Even if these methods are applied to learn the desired behaviour with reasonable effort, our generative and predictive models remain applicable to copy and perform the input–output relation. Whether a lot of resources can be saved is probably problem dependent and would require an architecture which is optimised for a particular purpose.

4.5. Modelling constraints and biological plausibility

Our implementation uses connectionist units with continuous outputs which are updated in discrete time steps. The output of one such unit may correspond to an average spike rate of an underlying spiking neuron population or to the firing probability, over a fixed time interval, of one spiking neuron. For the generative model, discrete time steps are appropriate as the model associates the location and angle with the motor control at a specific time and does not rely on previous decisions. For the predictive model, however, the development of activations over time is learnt to match the real world progress that is determined by the clock cycle. The speed of the attractor activations across the hidden layer is intrinsically encoded in the asymmetric components of the lateral weights (Zhang, 1996). The speed could be regulated by scaling this component, e.g. through an external input which arrives via Sigma-Pi synapses, as in Stringer, Rolls, Trappenberg, and de Araujo (2003), which is, however, not in the scope of our model. We omitted a parameter which controls differential neuron update and which could be used to adjust to different time scales. Instead, the activations of our memory-less neurons at the next time step depend only on their weighted input (which excludes self-input since self-connections are set to zero).

The model achieves global competition via the predictive weights using purely local learning rules without adaptive global variables. More biologically detailed spike timing dependent learning (STDP) rules cannot be resolved with connectionist units. One might reason that STDP might be suitable to calculate the difference between target- and reconstruction activation values, which is required in the learning terms, Eqs. (2, 3 and 5). Then, the spikes representing the target value would have to arrive earlier and might trigger postsynaptic spiking (positive learning sign), while the spikes representing the reconstruction would have to arrive after the postsynaptic spike (negative learning sign). A different opportunity would be that different cortical layers might represent different terms. Eq. (2) can be decomposed into a Hebbian term $\vec{x}*\vec{r}$ involving the original data and an anti-Hebbian term $-\vec{x}'*\vec{r}$ involving the reconstructed data. The related Boltzmann machine model (Ackley, Hinton, & Sejnowski, 1985) separates Hebbian and anti-Hebbian learning in time into its wake and sleep phases, respectively.

Our asymmetric model architecture assigns the motor cortex to a hierarchically higher position than its sensory and motor input/output areas. Since, layer 4 which receives forward projections is absent in motor cortical areas (but not in prefrontal cortex), one might consider them being laterally related to each other and lateral to (or even hierarchically lower than) other structures. The related Boltzmann machine would account for this laterality between input/output and its hidden units. Its recurrent relaxations furthermore make it better suitable for pattern completion. However, this would come at a cost of increased computation time, and its strict weight symmetry would not allow to implement a forward model.

4.6. Relation to other models

Considering performance, we needed a model which approximates an input–output relation: given sensory input $\vec{p}$ from vision and $\vec{\Phi}$ from an internal angle representation, we need a mapping to the motor output vector $\vec{m}$. This could be solved by supervised learning, e.g. using the error back-propagation algorithm (Rumelhart, Hinton, & Williams, 1986). We used instead a biologically inspired, generative model which does not distinguish input and output layers: $\vec{p}$, $\vec{\Phi}$ and $\vec{m}$ and are all treated as the input which is to be generated from a hidden layer code, $\vec{r}$. This allows for two modes of operation: (i) reconstruct one input component (e.g. $\vec{m}$) from the others (function approximation) or (ii) find a consistent joint representation $(\vec{p}, \vec{\Phi}, \vec{m})$ from a noisy multi-modal input $(\tilde{p}, \tilde{\Phi}, \tilde{m})$ which is possibly conflicting (cue combination).

A continuous attractor network can in simple cases solve these without involving a hidden code $\vec{r}$. The task of function approximation would be referred to as pattern completion.

But the application becomes more demanding, if there are no direct correlations between the inputs. For example, from the knowledge of the perceived target position $\vec{p}$ alone, one cannot infer the motor action $\vec{m}$, because the heading angle encoded in $\vec{\Phi}$ modulates the mapping. Direct weights between $\vec{p}$ and $\vec{m}$ therefore, cannot perform the input–output mapping and a hidden code $\vec{r}$ has to be introduced. An attractor network model of choice involving hidden units is the Boltzmann machine (Ackley et al., 1985). However, we chose a related variant based on the Helmholtz machine, because the Boltzmann machine does not allow for structured hierarchical information processing as it happens on the several hierarchical levels of the cortex.

A network for function approximation and cue combination has been proposed by Deneve, Latham, and Pouget (2001) for the simplified case that every input $\vec{p}$, $\vec{\Phi}$ and $\vec{m}$ is a Gaussian hill of activation on a one-dimensional neural sheet. The relaxation procedure within their attractor network solved each task in an optimal way, since a relaxation procedure using an attractor network has been shown to retrieve a stored activation pattern such that its properties encode the data with maximum likelihood (Deneve, Latham, & Pouget, 1999). An intuitive reason for the virtues of a relaxation procedure is that if parts of the network's activation are in conflict with the overall activation pattern, then the most optimal, consistent solution will be found by relaxation of the activations into an attractor. In the proposed network, however, weights were geometrically determined. Training is difficult, because the hidden layer code $\vec{r}$ is not given by the data (only pairs of $\vec{p}$, $\vec{\Phi}$ and $\vec{m}$ are accessible).

We have previously proposed a model based on the self-organising map (Kohonen, 2001) in which a hidden code is learnt given multi-modal input (Wermter & Elshaw, 2003). It reconstructs a missing input component based on the input vector of the hidden unit which has been most activated by the remaining inputs. This model can in principle generate interactive motor sequences. However, some restrictions are (i) the localist code of a winner-take-all network and (ii) a missing relaxation procedure.

Stringer et al. (2003) avoid the use of hidden units. They use Sigma-Pi connections of the state cells and the movement selector cells into the motor cells and a temporal activity trace learning rule to learn action sequences. Using abstract data, their model produces motor sequences with arbitrary speed and force.

Models on high level imitation and mirror neurons have also already been presented. The 'FARS' model (Fagg & Arbib, 1998) details the functional involvement of areas AIP and F5 in grasping. It highlights visually derived motor *affordances* by which the object is represented at the level of the motor system by features that are relevant for grasping it, such as size and orientation. It was extended to learn the recognition of action using the mirror system (Oztop & Arbib, 2002) of which a forward model was proposed (Oztop et al., 2003). Demiris & Hayes (2002) used the concept of the mirror neuron system to develop an architecture for robot imitation. In their model, the simulated imitating agent is given directly the coordinates of the demonstrator, bypassing the need for a sophisticated visual recognition system. A forward model predicts the outcome of the output of a behaviour model in order to compare this to the actual state of the demonstrator and to produce an error signal which can be used to identify a particular behaviour. According to the authors, robot imitation offers the opportunity for learning through demonstration and so allows non-robot programmers the chance to teach a robot.

5. Conclusion

We have set up a biologically realistic model of a motor cortical area which can take over and predict actions proposed to be learnt earlier via a reinforcement scheme in the basal ganglia. All network connections are trained, only local learning rules are used and the resulting connectivity structure is topographic where lateral predictive weights are centre-excitatory, surround-inhibitory. This work bridges the wide gap between neuroscience and robotics and motivates the development of the motor cortex per se as a generative and predictive model, a model class central to the description of sensory cortices.

Currently we are extending the framework to include several actions and additional language input. We expect mirror neuron properties to arise among hidden area neurons which eventually will enable a robot to perform learnt actions based on language instruction.

Acknowledgements

This work is part of the MirrorBot project supported by the EU in the FET-IST programme under grant IST- 2001-35282. We thank Christo Panchev, Alexandros Zochios, Hauke Bartsch as well as the anonymous reviewers for useful contributions to the manuscript.

References

Aboitiz, F., Morales, D., & Montiel, J. (2003). The evolutionary origin of the mammalian isocortex: Towards an integrated developmental and functional approach. *The Behavioral and Brain Sciences*, *26*(5), 535–552.

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, *9*, 147–169.

Bell, A., & Sejnowski, T. (1997). The 'independent components' of natural scenes are edge filters. *Vision Research*, *37*(23), 3327–3338.

Blundell, P., Hall, G., & Killcross, S. (2003). Preserved sensitivity to outcome value after lesions of the basolateral amygdala. *The Journal of Neuroscience*, *23*(20), 7702–7709.

Boyan, J., & Moore, A. (1995). Generalization in reinforcement learning: Safely approximating the value function. In G. Tesauro, D. Touretzky, & T. Lee (Eds.), *Neural information processing systems 7* (pp. 369–376). Cambridge, MA: MIT Press.

Brashers-Krug, T., Shadmehr, R., & Bizzi, E. (1996). Consolidation in human motor memory. *Nature*, *382*, 252–255.

Dayan, P., Hinton, G. E., Neal, R., & Zemel, R. S. (1995). The Helmholtz machine. *Neural Computation*, *7*, 1022–1037.

Demiris, Y., & Hayes, G. (2002). Imitation as a dual-route process featuring prediction and learning components: A biologically-plausible computational model. In K. Dautenhaln, & C. Nehaniv (Eds.), *Imitation in animals and artifacts* (pp. 327–361). Cambridge, MA: MIT Press.

Deneve, S., Latham, P., & Pouget, A. (1999). Reading population codes: A neural implementation of ideal observers. *Nature Neuroscience*, *2*(8), 740–745.

Deneve, S., Latham, P., & Pouget, A. (2001). Efficient computation and cue integration with noisy population codes. *Nature Neuroscience*, *4*(8), 826–831.

Dickinson, A. (1985). Actions and habits: The development of behavioural autonomy. *Philosophical Transactions of the Royal Society of London. Series B*, *308*, 67–78.

Doya, K. (1999). What are the computations of the cerebellum, the basal ganglia and the cerebral cortex? *Neural Networks*, *12*, 961–974.

Fagg, A., & Arbib, M. (1998). Modeling parietal–premotor interactions in primate control of grasping. *Neural Networks*, *11*, 1277–1303.

Felleman, D., & Van Essen, D. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, *1*, 1–47.

Foster, D., Morris, R., & Dayan, P. (2000). A model of hippocampally dependent navigation, using the temporal difference learning rule. *Hippocampus*, *10*, 1–16.

Fukushima, K., Miyake, S., & Ito, T. (1983). Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, *13*, 826–834.

Gallese, V., Fadiga, L., Fogassi, L., & Rizzolatti, G. (1996). Action recognition in the premotor cortex. *Brain*, *119*, 593–609.

Gallese, V., & Goldman, A. (1998). Mirror neurons and the simulation theory of mind-reading. *Trends in Cognitive Sciences*, *2*(12), 493–501.

Hikosaka, O., Nakamura, K., Sakai, K., & Nakahara, H. (2002). Central mechanisms of motor skill learning. *Current Opinion in Neurobiology*, *12*, 217–222.

Hinton, G., Dayan, P., To, A., & Neal, R. (1995). The Helmholtz machine through time. In: F. Soulie, & R. Gallinari (Eds.), *Proceedings of the ICANN* (pp. 483–490).

Hinton, G. E., Dayan, P., Frey, B. J., & Neal, R. (1995b). The wake–sleep algorithm for unsupervised neural networks. *Science*, *268*, 1158–1161.

Jog, M., Kubota, Y., Connolly, C., Hillegaart, V., & Graybiel, A. (1999). Building neural representations of habits. *Science*, *286*, 1745–1749.

Kohler, E., Keysers, C., Umilta, M., Fogassi, L., Gallese, V., & Rizzolatti, G. (2002). Hearing sounds, understanding actions: Action representation in mirror neurons. *Science*, *297*, 846–848.

Kohonen, T. (2001). *Self-organizing maps*. Berlin: Springer.

McCallum, A. (1995). *Reinforcement learning with selective perception and hidden state*. PhD thesis, University of Rochester.

Moore, A., & Atkeson, C. (1995). The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, *21*, 1–36.

Olshausen, B., & Field, D. (1997). Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, *37*, 3311–3325.

Oztop, E., & Arbib, M. (2002). Schema design and implementation of the grasp-related mirror neuron system. *Biological Cybernetics*, *87*, 116–140.

Oztop, E., Wolpert, D., & Kawato, M. (2003). Mirror neurons: Key for mental simulation? In: *Twelfth annual computational neuroscience meeting CNS (abstracts)* (p. 81).

Pasupathy, A., & Miller, E. (2005). Different time courses of learning-related activity in the prefrontal cortex and striatum. *Nature*, *433*, 873–876.

Plaut, D. C., & Kello, C. T. (1998). The emergence of phonology from the interplay of speech comprehension and production: A distributed connectionist approach. In: B. MacWhinney (Ed.), *The emergence of language* (pp. 381–415).

Pulvermüller, F. (2002). A brain perspective on language mechanisms: From discrete neuronal ensembles to serial order. *Progress in Neurobiology*, *67*, 85–111.

Rao, R., & Ballard, D. (1997). Dynamic model of visual recognition predicts neural response properties of the visual cortex. *Neural Computation*, *9*(4), 721–763.

Ravel, S., & Richmond, B. (2005). Where did the time go? *Nature Neuroscience*, *8*(6), 705–707.

Riesenhuber, M., & Poggio, T. (2002). Neural mechanisms of object recognition. *Current Opinion in Neurobiology*, *12*, 162–168.

Rizzolatti, G., & Arbib, M. (1998). Language within our grasp. *Trends in Neurosciences*, *21*(5), 188–194.

Rizzolatti, G., & Luppino, G. (2001). The cortical motor system. *Neuron*, *18*(2), 889–901.

Robinson, F., Cohen, J., May, J., Sestokas, A., & Glickstein, M. (1984). Cerebellar targets of visual pontine cells in the cat. *The Journal of Comparative Neurology*, *223*(4), 471–482.

Rolls, E. (2000). The orbitofrontal cortex and reward. *Cerebral Cortex*, *10*(3), 284–294.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In: Parallel distributed processing (Vol. 1, pp. 318–362). Cambridge, MA: MIT Press.

Sanes, J., Dimitrov, B., & Hallett, M. (1990). Motor learning in patients with cerebellar dysfunction. *Brain*, *113*(1), 103–120.

Sauser, E., & Billard, A. (2005). Three dimensional frames of references transformations using recurrent populations of neurons. *Neurocomputing*, *64*, 5–24.

Shadmehr, R., & Holcomb, H. (1997). Neural correlates of motor memory consolidation. *Science*, *277*, 821–825.

Stringer, S., Rolls, E., Trappenberg, T., & de Araujo, I. (2003). Self-organising continuous attractor networks and motor function. *Neural Networks*, *16*, 161–182.

Umilta, M., Kohler, E., Gallese, V., Fogassi, L., Fadiga, L., Keysers, C., et al. (2001). I know what you are doing: A neurophysical study. *Neuron*, *31*, 155–165.

Uther, W., & Veloso, M. (1998). Tree based discretization for continuous state space reinforcement learning. In: *Proceedings of AAAI-98* (pp. 769–774).

Weber, C. (2001). Self-organization of orientation maps, lateral connections, and dynamic receptive fields in the primary visual cortex. In G. Dorffner, H. Bischof, & K. Hornik (Eds.), *Proceedings of the ICANN* (pp. 1147–1152). Berlin: Springer.

Weber, C., & Wermter, S. (2003). Object localization using laterally connected 'what' and 'where' associator networks. In: *Proceedings of the ICANN* (pp. 813–820). Berlin: Springer.

Weber, C., Wermter, S., & Zochios, A. (2004). Robot docking with neural vision and reinforcement. *Knowledge-Based Systems*, *17*(2–4), 165–172.

Wermter, S., & Elshaw, M. (2003). Learning robot actions based on self-organising language memory. *Neural Networks*, *16*(5–6), 661–669.

Yin, H., Knowlton, B., & Balleine, B. (2004). Lesions of dorsolateral striatum preserve outcome expectancy but disrupt habit formation in instrumental learning. *The European Journal of Neuroscience*, *19*(1), 181–189.

Zhang, K. (1996). Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: A theory. *Journal of Neuroscience*, *16*, 2112–2126.