PERGAMON

AVAILABLE AT www.ComputerScienceWeb.com POWERED BY SCIENCE DIRECT®

Neural Networks 16 (2003) 161–182

Neural Networks

www.elsevier.com/locate/neunet

Self-organizing continuous attractor networks and motor function

S.M. Stringer, E.T. Rolls*,1, T.P. Trappenberg, I.E.T. de Araujo

Department of Experimental Psychology, Centre for Computational Neuroscience, Oxford University, South Parks Road, Oxford OX1 3UD, UK

Received 4 July 2001; accepted 11 November 2002

Abstract

Motor skill learning may involve training a neural system to automatically perform sequences of movements, with the training signals provided by a different system, used mainly during training to perform the movements, that operates under visual sensory guidance. We use a dynamical systems perspective to show how complex motor sequences could be learned by the automatic system. The network uses a continuous attractor network architecture to perform path integration on an efference copy of the motor signal to keep track of the current state, and selection of which motor cells to activate by a movement selector input where the selection depends on the current state being represented in the continuous attractor network. After training, the correct motor sequence may be selected automatically by a single movement selection signal. A feature of the model presented is the use of 'trace' learning rules which incorporate a form of temporal average of recent cell activity. This form of temporal learning underlies the ability of the networks to learn temporal sequences of behaviour. We show that the continuous attractor network models developed here are able to demonstrate the key features of motor function. That is, (i) the movement can occur at arbitrary speeds; (ii) the movement can occur with arbitrary force; (iii) the agent spends the same relative proportions of its time in each part of the motor sequence; (iv) the agent applies the same relative force in each part of the motor sequence; and (v) the actions always occur in the same sequence.

Keywords: Motor function; Continuous attractor neural networks; Self-organization; Trace learning

1. Introduction

Much effort has been devoted to the application of classical control theory to the understanding of motor control by the brain (Miall & Wolpert, 1996; Wolpert & Ghahramani, 2000; Wolpert, Miall, & Kawato, 1998). In these models the motor programme seeks to minimise the distance or error between the current state of the agent and a particular target state (or desired trajectory), where the distance is either determined by external visual or proprioceptive feedback, or estimated with internal models. In this paper we present an alternative, new, approach which is inspired from a dynamical systems perspective, rather than classical control theory. The model presented here is able to learn arbitrary dynamical motor sequences, execute such motor sequences at arbitrary speed, and perform the motor sequences in the absence of external visual or proprioceptive cues. These are important properties for models of motor control in biological agents (Bizzi & Polit, 1979; Laszlo, 1966, 1967; Polit & Bizzi, 1978; Schmidt, 1987, 1988). The system we describe is based on continuous attractor networks. These networks are used in a mode in which they perform 'path integration' on an efference copy of a motor signal. This enables them to learn arbitrary paths, which correspond to sequences of motor output commands generated by the network, in response to a movement selector signal. Moreover, they can execute the motor sequence at arbitrary speeds.

The learning of new motor skills appears to involve distinct stages, which a number of authors have attempted to identify and describe (Fitts & Posner, 1967; Gentile, 1972, 1987; Newell, 1985). When an animal begins to learn a new motor skill, the initial mechanisms used to execute the motor task during learning appear to be specific to the learning phase, and may involve for example attention and use of visual and proprioceptive feedback (Magill, 1998; Sections 3.2 and 4.2). Gradually the execution of the skill and the necessary sequence of movements becomes more

* Corresponding author. Tel.: +44-1865-271348; fax: +44-1865-310447.
E-mail address: edmund.rolls@psy.ox.ac.uk (E.T. Rolls).
[1] www.cns.ox.ac.uk

0893-6080/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
PII: S0893-6080(02)00237-X

automatic. In this paper we propose a general framework for modelling this learning process, and show how it provides an efficient means to learn from even noisy inputs such as those which may be present during the learning process.

A framework that introduces some of the properties that may be required for motor control is the *generalized motor program* (Schmidt, 1987, 1988). Schmidt proposed that a generalized motor program controls a class of actions, rather than a specific movement or sequence. Schmidt suggested that a particular class of actions has a common set of *invariant features* which define the class. Although many possible characteristics might be regarded as candidates for invariant features of motor programs, the three most commonly proposed invariant features are: the relative timing of the components of the skill; the relative force used in performing the skill; and the order or sequence of the components. In this paper we develop a network model that is able to demonstrate the key features of Schmidt's generalised motor control program. The model has the following key properties:

- the movement can occur at arbitrary speeds;
- the movement can occur with arbitrary force;
- the agent spends the same relative proportions of its time in each part of the motor sequence;
- the agent applies the same relative force in each part of the motor sequence;
- the actions always occur in the same sequence.

The model of motor function described here is based on self-organizing continuous attractor neural networks (CANNs). Continuous attractor networks are able to stably maintain a localised packet of neuronal firing activity (Amari, 1977; Taylor, 1999). The new model we describe makes use of our proposal of how path integration could be performed in a continuous attractor network (Stringer, Rolls, & Trappenberg, 2003; Stringer, Rolls, Trappenberg, & de Araujo, 2002b; Stringer, Trappenberg, Rolls, & de Araujo, 2002c). Use of this approach enables us to develop here a network that can learn temporal sequences of behaviour, and can perform the sequences at arbitrary speeds.

2. The model

The model is described with the neural architecture shown in Fig. 1. There are two connected networks. First, there is a network of state cells (with firing rate r_i^S for state cell i) which represent the current positional (or postural)

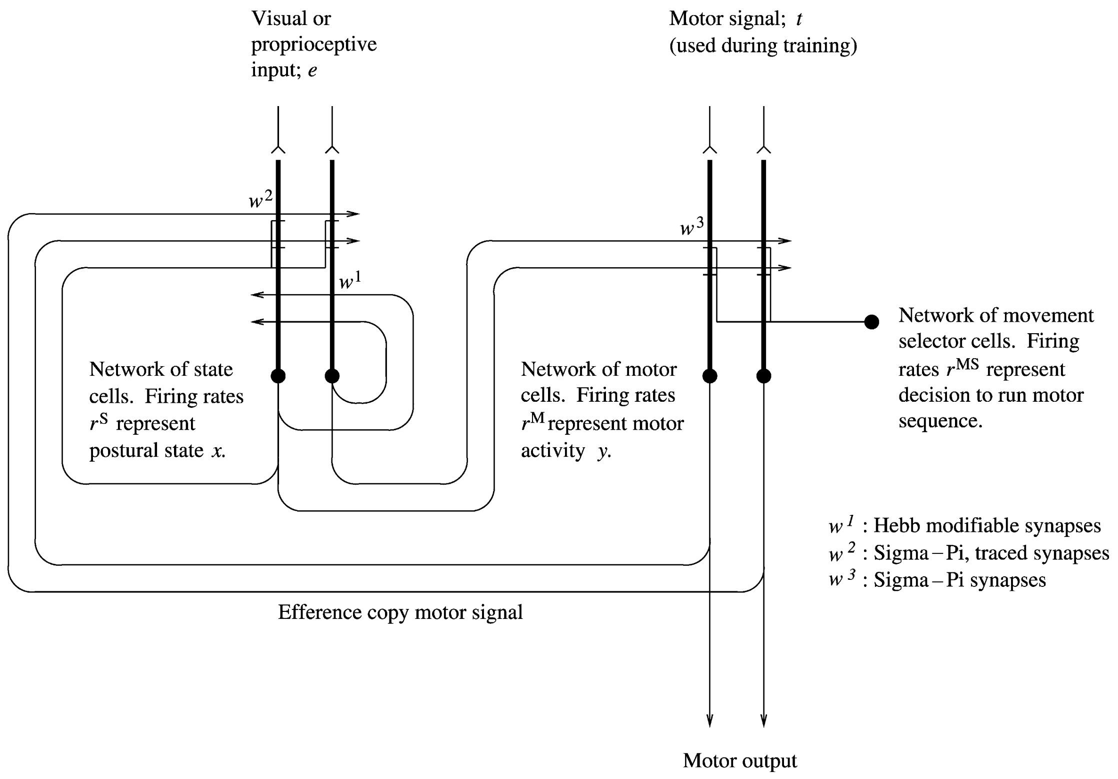

Visual or proprioceptive input; e
Motor signal; t (used during training)
w^2
w^3
w^1
Network of state cells. Firing rates r^S represent postural state x.
Network of motor cells. Firing rates r^M represent motor activity y.
Network of movement selector cells. Firing rates r^{MS} represent decision to run motor sequence.
w^1 : Hebb modifiable synapses
w^2 : Sigma–Pi, traced synapses
w^3 : Sigma–Pi synapses
Efference copy motor signal
Motor output

Fig. 1. General network architecture for continuous attractor network model of state representation and motor function. There is a network of state cells which represent the postural state of the agent, a network of motor cells which represent the motor activity, and a network of movement selector cells which represent the decision to perform a motor sequence. The forward model is implemented by the synaptic connections w^2 from Sigma–Pi couplings of state cells and motor cells to the state cells. The connections w^2 are able to update the representation in the network of state cells given the current patterns of firing in both the network of state cells and the network of motor cells. The inverse model is implemented by the synaptic connections w^3 from Sigma–Pi couplings of state cells and movement selector cells to the motor cells. Given a desired motor task or target state represented by the firing of the movement selector cells, the connections w^3 are able to drive the motor cells to perform the appropriate motor actions.

state of the agent. The positional state cell network operates as a movement path integrator using a CANN, which is a class of network that can maintain the firing of its neurons to represent any location along a continuous physical dimension representing the state of the agent (Amari, 1977; Taylor, 1999). Secondly, there is a network of motor cells (with firing rate r_i^{M} for motor cell i) which represent the current motor command. As the size of the activity packet in the motor network increases, this will increase the level of the current injected into muscle fibres or other motor neurons receiving efferent connections from these motor cells. Hence, the size of the activity packet in the network of motor cells may be viewed as representing the force, and hence the speed, of the movement of the agent itself.

During the initial learning phase, each positional state cell i receives an external input e_i, for example visual and proprioceptive input, which carries information about the state of the agent. When visual or proprioceptive cues are available, these external inputs dominate other excitatory inputs to the state cells, and force each positional state cell to respond best to a particular positional state of the agent, with less firing as the agent's state moves away from the preferred state. During training, the motor input signals t_i received by each motor cell i cause the agent to proceed through a set of positional states allowing the network to learn the relation between the motor firing r_i^{M} and the corresponding positional states r_i^{S} as forced on the network of state cells by the visual or proprioceptive inputs e_i.

During learning, the recurrent connections w_{ij}^1 from state cell j to i use associative synaptic modification so that the synaptic strengths between the state cells reflect the distance between the positional states of the agent represented by the state cells. The recurrent connectivity implemented by w_{ij}^1 allows the network of state cells to operate as a continuous attractor network and support stable patterns of firing in the absence of external visual or proprioceptive input, so that the agent can operate with incomplete sensory input or in the dark. However, the most important function performed by the positional state CANN is to enable an efference copy of the motor signal to update using the path integration the positional state CANN, and thus to implement a *forward model* of motor function. The path integration uses the connections w_{ijk}^2, which are learned by a traced Sigma–Pi learning rule, which allows a combination of the traced activity within the state cell network (which represents the preceding position) and the traced activity within the motor cell network (which represents the preceding motor command) to be associated with the current positional state. Thus, after training, the firing of a particular cluster of state cells and a particular cluster of motor cells should stimulate the firing of further state cells such that the pattern of activity within the network of state cells evolves continuously to faithfully reflect and track the changing state of the agent as it performs the motor sequence. Of course, a particular combination of the agent's state and motor activity should always lead to the same next state, regardless of which particular motor sequence is being executed. Thus, when the network is trained on multiple motor sequences, the firing of a particular cluster of state cells and a particular cluster of motor cells should always stimulate the firing of the same set of further state cells, regardless of which particular motor sequence is being executed. The synapses are Sigma–Pi in that there are three terms in their description w_{ijk}^2, where the i subscript refers to the postsynaptic position state cell i, the j subscript refers to the presynaptic position state cell j, and the k subscript refers to the input from the presynaptic motor cell k.

To develop the hypothesis more formally, the activation h_i^{S} of state cell i in the model is governed by

$$\tau\frac{\mathrm{d}h_i^{\mathrm{S}}(t)}{\mathrm{d}t} = -h_i^{\mathrm{S}}(t) + \frac{\phi_0}{C^{\mathrm{S}}}\sum_j (w_{ij}^1 - w^{\mathrm{INH}})r_j^{\mathrm{S}}(t) + e_i + \frac{\phi_1}{C^{\mathrm{S}\times\mathrm{M}}}\sum_{j,k} w_{ijk}^2 r_j^{\mathrm{S}} r_k^{\mathrm{M}}, \qquad (1)$$

where the activation h_i^{S} is driven by the following terms. The term r_j^{S} is the firing rate of state cell j, w_{ij}^1 is the excitatory (positive) synaptic weight from state cell j to state cell i, and w^{INH} is a global constant describing the effect of inhibitory interneurons within the layer of state cells.[2] Further terms in Eq. (1) are as follows. The term τ is the time constant of the system. The term e_i represents an external input to state cell i, which may be visual or proprioceptive. When the agent is denied visual or proprioceptive input, the term e_i is set to zero. Thus, in the absence of visual or proprioceptive input, the key term driving the state cell activations in Eq. (1) is a sum of coupled inputs from the state and motor cells $\sum_{j,k} w_{ijk}^2 r_j^{\mathrm{S}} r_k^{\mathrm{M}}$, where r_j^{S} is the firing rate of state cell j, r_k^{M} is the firing rate of motor cell k, and w_{ijk}^2 is the corresponding strength of connection from these cells.[3]

The issue of the biological plausibility of such synapses, which can be thought of as having two presynaptic terms which operate in combination (i.e. multiplicatively), is considered by Stringer et al. (2002c), where we note that there are several possible ways in which this could be implemented in the brain. The firing rate r_i^{S} of state cell i is determined from the activation h_i^{S} and the sigmoid

[2] The scaling factor (ϕ_0/C^{S}) controls the overall strength of the recurrent inputs to the layer of state cells, where ϕ_0 is a constant and C^{S} is the number of presynaptic connections received by each state cell from other state cells. Scaling the recurrent inputs $\sum_j (w_{ij}^1 - w^{\mathrm{INH}})r_j^{\mathrm{S}}(t)$ by $(C^{\mathrm{S}})^{-1}$ ensures that the overall magnitude of the recurrent input to each state cell remains approximately the same as the number of recurrent connections received by each state cell is varied. For a fully recurrently connected layer of state cells, C^{S} is equal to the total number of state cells, N^{S}.

[3] The scaling factor $\phi_1/C^{\mathrm{S}\times\mathrm{M}}$ controls the overall strength of the motor inputs, where ϕ_1 is a constant, and $C^{\mathrm{S}\times\mathrm{M}}$ is the number of Sigma–Pi connections received by each state cell. Scaling the motor inputs by $(C^{\mathrm{S}\times\mathrm{M}})^{-1}$ ensures that the overall magnitude of the motor input to each state cell remains approximately the same as the number of coupled state and motor connections received by each state cell is varied. For a fully connected network, $C^{\mathrm{S}\times\mathrm{M}}$ is equal to the number of state cells, N^{S}, times the number of motor cells, N^{M}.

activation function

$$r_i^{\mathrm{S}}(t) = \frac{1}{1 + \mathrm{e}^{-2\beta(h_i^{\mathrm{S}}(t)-\alpha)}}, \tag{2}$$

where α and β are the sigmoid threshold and slope, respectively.

The recurrent synapses w_{ij}^1 in the state cell continuous attractor are trained by a local associative (Hebb) rule

$$\delta w_{ij}^1 = k^1 r_i^{\mathrm{S}} r_j^{\mathrm{S}}. \tag{3}$$

This rule increases the strength of the synaptic connections between state cells that represent nearby states of the agent, and which tend to be co-active due to broadly tuned, overlapping receptive fields. The learning rule used to update the synapses w_{ijk}^2 can be expressed by

$$\delta w_{ijk}^2 = k^2 r_i^{\mathrm{S}} \bar{r}_j^{\mathrm{S}} \bar{r}_k^{\mathrm{M}}, \tag{4}$$

where $\bar{r}_j^{\mathrm{S}}$ refers to a memory trace of the firing r_j^{S}, and $\bar{r}_k^{\mathrm{M}}$ refers to a memory trace of the firing of r_k^{M}. The trace value $\bar{r}$ of the firing rate r of a cell is calculated according to

$$\bar{r}(t + \delta t) = (1 - \eta)r(t + \delta t) + \eta\bar{r}(t), \tag{5}$$

where η is a parameter in the interval [0,1] which determines the relative contributions of the current firing and the previous trace. For $\eta = 0$ the trace becomes just the present firing rate, and as η is increased the contribution of preceding firing at times earlier than the current timestep is increased (Stringer et al., 2002b,c; Sutton & Barto, 1981). Possible ways in which such traces of previous neuronal activity could be implemented include short-term memory related firing in networks, and biophysical processes within neurons (Stringer et al., 2002c).

The motor cells are driven by the Sigma–Pi synapses w_{ijk}^3 which allow the selection of motor cell firing r_i^{M} by movement selector cells r_k^{MS} to be dependent on the current activity of the position state cells r_j^{S}. Of course, if visual input is available, visual information may be used directly to guide the motor activity. However, in primates, for example, only a relatively small amount of the environment is analysed in detail by the visual system at any moment, and so most movement is probably guided by internal representations of the positional state of the agent in its environment that in our idealised model are supported by the network of state cells. The firing of the movement selector cells r_k^{MS} represents the instruction to perform a particular motor sequence. In particular, the force, and hence speed, of the movement may be controlled by varying the firing rates of the movement selector cells. As the firing rates of the movement selector cells increase, the size of the activity packet within the motor network increases and the movement is performed with greater force, and hence speed. The synapses w_{ijk}^3 are Sigma–Pi in that there are three terms in their description, where the i subscript refers to the postsynaptic motor cell i, the j subscript refers to the presynaptic position state cell j, and the k subscript refers to the input from the presynaptic movement selector cell k. The synaptic weights w_{ijk}^3 are set up during training by a learning rule which associates the co-firing of the movement selector cells r_k^{MS} and a particular cluster of state cells r_j^{S}, with the firing of the appropriate cluster of motor cells r_i^{M} produced during the training by an external motor signal t_i (Fig. 1). Then, after training, the co-firing of the movement selector cells and a particular cluster of position state cells stimulates the relevant cluster of motor cells, and thus produces motor firing which is appropriate given the current position of the agent. The movement selector specifies the desired movement (which may be a final target position for the agent to reach, or simply a desired motor sequence like saying a word), and the motor firing needed to produce this can be learned by the network because it takes into account the current position state when the movement selection command is issued. The synaptic connections w_{ijk}^3 thus implement an *inverse model* of motor function: given a desired motor task or target state represented by the firing of the movement selector cells, the synaptic connections w_{ijk}^3 drive the motor cells to perform the appropriate motor actions.

More formally, the activation h_i^{M} of motor cell i is governed by

$$\tau\frac{\mathrm{d}h_i^{\mathrm{M}}(t)}{\mathrm{d}t} = -h_i^{\mathrm{M}}(t) + t_i + \frac{\phi_2}{C^{\mathrm{S\times MS}}}\sum_{j,k} w_{ijk}^3 r_j^{\mathrm{S}} r_k^{\mathrm{MS}}, \tag{6}$$

where the activation h_i^{M} is driven by the following terms. The first term driving the activations of the motor cells in Eq. (6) is the motor training signal t_i for each motor cell i. This term is present only during the training phase. The input term t_i models an initial mechanism used during learning, perhaps involving active attentional processes, to stimulate the neurons associated with the new motor sequence being learned. After the training phase is completed, the input terms t_i are set to zero for the subsequent testing phase. One possible mechanism for generating the training signal t_i during the learning phase is discussed below in Section 5, and involves the stimulation, during training, of an appropriate temporal sequence of movement selector cells that encodes simpler motor primitives that represent component motor features of the full motor sequence to be learned.

The second term driving the activations of the motor cells in Eq. (6) is the input from couplings of the state cells and movement selector cells $\sum_{j,k} w_{ijk}^3 r_j^{\mathrm{S}} r_k^{\mathrm{MS}}$, where r_j^{S} is the firing rate of state cell j, r_k^{MS} is the firing rate of movement selector cell k, and w_{ijk}^3 is the corresponding strength of the connection from these cells. The driving term $\sum_{j,k} w_{ijk}^3 r_j^{\mathrm{S}} r_k^{\mathrm{MS}}$ initiates activity in the network of motor neurons, and then drives the activity packet along a particular path through the motor network.[4] The synapse connecting the state and movement selector cells to the motor cells has a Sigma–Pi form in that it computes a weighted sum of the products of the firing rates of the state cells and movement selector cells. This ensures the activity within the network of motor cells is driven by the state cells if and only if the movement selector cells are also active. If the movement selector cells stop firing then the activity in the motor network decays to zero according to the time constant τ. The firing rate r_i^{M} of motor cell i is determined from the activation h_i^{M} and the sigmoid activation function

$$r_i^{\mathrm{M}}(t) = \frac{1}{1 + \mathrm{e}^{-2\beta(h_i^{\mathrm{M}}(t)-\alpha)}}, \tag{7}$$

where α and β are the sigmoid threshold and slope, respectively.

The synaptic weights w_{ijk}^3 from the state cells and movement selector cells to the motor cells are updated during learning according to

$$\delta w_{ijk}^3 = k^3 r_i^{\mathrm{M}} r_j^{\mathrm{S}} r_k^{\mathrm{MS}}. \tag{8}$$

During the learning phase, the agent performs the desired motor sequence to be learned. As the agent performs the motor task, the state cells are driven by the visual or proprioceptive inputs e_i, the motor cells are driven by the training signal t_i, and the synaptic weights w_{ij}^1, w_{ijk}^2 and w_{ijk}^3 are updated according to the simple learning rules discussed earlier. During repeated learning cycles of the motor sequence, even with some amount of error in the training signal t_i for the motor cells, the synaptic connectivity of the network self-organizes such that, after training, the correct motor sequence may be stimulated solely by stimulating the particular set of movement selector cells. The overall network architecture is thus able to learn the following. First, the network learns a forward model of motor control, implemented through the synaptic connections w_{ijk}^2, which uses the motor signal from the motor cell firing to update the position state CANN to implement path integration. Secondly, the network learns an inverse model of motor control, implemented through the synaptic weights w_{ijk}^3, which enables a movement command (or target-position movement command) signal represented by the firing of the movement selector cells r_k^{MS} to select the appropriate motor cell firing r_i^{M} given the current position or postural state represented by the firing of the state cells r_j^{S}. Effectively, the movement selector cells enable a sequence of motor cell firing to be produced, where the motor sequence is guided by the changing pattern of activity within the state CANN.

This model is capable of implementing various types of motor programs. For example, the model developed here may be used to learn movement sequences which involve the agent seeking to reach a fixed positional target, as in prehension, where the target position is specified by the particular pattern of firing among the movement selector cells. Alternatively, the motor sequence may not involve a particular target state for the agent. Instead the performance of the motor sequence itself may be the goal as in the case of, for example, saying a word. Furthermore, the network can implement cyclic motor programs, involving continuous cycling through the position state space of the agent, if the state space associated with the movement is toroidal (as might be required for example to implement locomotion). The network can implement movements in higher dimensional spaces, because the nature of the position state space is stored in the recurrent synaptic connections w_{ij}^1, which may become self-organised during learning to represent spaces of arbitrary dimensionality (Stringer et al., 2002b,c). However, for simplicity of illustration, in the simulations performed in this paper the state space of the agent, represented by the network of state cells, is one-dimensional.

2.1. Stabilization of the activity packet within the continuous attractor network of state cells when the agent is stationary

As described for the models presented by Stringer et al. (2002b,c), the recurrent synaptic weights within the continuous attractor network may be corrupted by a certain amount of noise from the learning regime. This in turn can lead to drift of the activity packet within the continuous attractor network of state cells when there is no external visual or proprioceptive input available even when the agent is not moving. Stringer et al. (2002c) proposed that in real nervous systems this problem may be solved by enhancing the firing of neurons that are already firing. This might be implemented through mechanisms for short term synaptic enhancement (Koch, 1999), or through the effects of voltage dependent ion channels in the brain such as NMDA receptors. In the model presented in this paper, we simulate these effects using an additional non-linearity in the activation functions (2) and (7) (such as might be implemented by NMDA receptors, see Wang (1999) and Lisman, Fellous, and Wang (1998)) by adjusting the sigmoid threshold α_i for each state and motor cell i according to

$$\alpha_i = \begin{cases} \alpha^{\mathrm{HIGH}} & \text{if } r_i < \gamma \\ \alpha^{\mathrm{LOW}} & \text{if } r_i \geq \gamma \end{cases}, \tag{9}$$

where γ is a firing rate threshold. This helps to reinforce the current position of the activity packet within the continuous attractor network of state cells. The sigmoid slopes are set to a constant value, β, for all cells i.

[4] The scaling factor $\phi_2/C^{\mathrm{S\times MS}}$ controls the overall strength of the inputs from couplings of state and movement selector cells, where ϕ_2 is a constant, and $C^{\mathrm{S\times MS}}$ is the number of connections received by each motor cell from couplings of the state and movement selector cells. Scaling the inputs from the state and movement selector cells by $(C^{\mathrm{S\times MS}})^{-1}$ ensures that the overall magnitude of the input from the state and movement selector cells to each motor cell remains approximately the same as the number of connections received by each motor cell from couplings of state and movement selector cells is varied. For a fully connected network, $C^{\mathrm{S\times MS}}$ is equal to the number of state cells, N^{S}, times the number of movement selector cells, N^{MS}.

2.2. Training and testing procedures

We next present numerical (simulation) results to prove and illustrate the main properties of the model, including its ability to learn arbitrary motor sequences. Fig. 1 shows the details of the network architecture used for the simulations. In most of the simulations (except Experiment 4) there were 200 state cells, 200 motor cells, and 200 movement selector cells. In the numerical simulations presented below we demonstrated the model working with a one-dimensional state space, but the concepts readily generalise to higher dimensions. We assumed the state space x of the agent was a finite one-dimensional space from $x = 0-1$. That is, the state of the agent was defined by the parameter $x \in [0,1]$. In the model simulations the state cells were mapped onto a regular grid of different states, where for each state cell i there was a unique preferred state x_i of the agent for which the cell was stimulated maximally by the visual and proprioceptive cues. Similarly, we assumed the motor space y of the agent was a finite one-dimensional space from $y = 0-1$. That is, the instantaneous motor activity of the agent was defined by the parameter $y \in [0,1]$. In the simulations of the model the motor cells were mapped onto a regular grid of different instantaneous motor activities, where for each motor cell i there was a unique preferred instantaneous motor activity y_i for which the cell was stimulated maximally.

During the initial learning phase it was assumed that the external input e_i to the state cells would dominate all other excitatory inputs. Therefore, in the simulations presented below we employed the following modelling simplification. During the learning phase in which the agent received external visual or proprioceptive input, rather than implementing the dynamical Eqs. (1) and (2), we set the firing rates of the state cells according to the following Gaussian response profile

$$r_i^{\mathrm{S}} = \exp(-(s_i^{\mathrm{S}})^2/2(\sigma^{\mathrm{S}})^2), \tag{10}$$

where s_i^{S} was the absolute value of the difference between the actual state x of the agent and the preferred state x_i for state cell i, and σ^{S} was the standard deviation. For each state cell i, s_i^{S} was given by

$$s_i^{\mathrm{S}} = |x_i - x|. \tag{11}$$

During the initial learning phase we assumed that the agent would employ specific neural mechanisms, perhaps involving active attentional processes, to stimulate the particular sequence of motor neurons associated with the new motor task. These mechanisms were modelled by the incorporation of the training signal terms t_i in Eq. (6). However, in the simulations presented below we employed the following modelling simplification. During the learning phase, rather than implementing the dynamical Eqs. (6) and (7), the firing rate r_i^{M} of each motor cell i was set according to the following Gaussian response profile

$$r_i^{\mathrm{M}} = \exp(-(s_i^{\mathrm{M}})^2/2(\sigma^{\mathrm{M}})^2), \tag{12}$$

where s_i^{M} was the absolute value of the difference between the actual motor activity y of the agent and the preferred motor activity y_i for motor cell i, and σ^{M} was the standard deviation. For each motor cell i, s_i^{M} was given by

$$s_i^{\mathrm{M}} = |y_i - y|. \tag{13}$$

The numerical simulations began with the learning phase in which the synaptic weights, w_{ij}^1, w_{ijk}^2 and w_{ijk}^3, were self-organized. At the start of the learning, the synaptic weights were initialized to zero (but could have been random positive values). Then learning proceeded with the agent running through the motor training sequence. Different motor tasks might involve slightly different training and testing procedures. However, the basic process was as follows. The motor task involved the motor activity of the agent y and the state of the agent x simultaneously moving through their respective one-dimensional spaces in a particular sequence that defined the motor program. That is, during learning, the state x of the agent and motor activity y ran through the sequence

$$x = f(t) \qquad \text{and} \qquad y = g(t), \tag{14}$$

where f and g were the functions of time t that defined the motor sequence to be learned. This was the motor task to be learned by the movement selector cells. (Although, in some of the simulations described later, we demonstrated how the network was able to learn the correct motor sequence, defined by Eq. (14), even when the motor training sequence used during learning contained some error). As the agent performed the motor sequence (14), the firing rates of the state cells were set according to Eq. (10), the firing rates of the motor cells were set according to Eq. (12), and the firing rates of the relevant subset of movement selector cells were set to 1. This led to activity packets in the state and motor networks moving through their respective networks. During this, the synaptic weights were updated as described above according to the learning rules (3), (4) and (8). At the start of training all trace values were initialised to zero.

After the learning phase was completed, the simulations continued with the testing phase in which the agent had to perform the motor sequence without the motor training signal, and without the visual and proprioceptive inputs. The aim was to demonstrate that the population of movement selector cells had learned to produce the desired motor sequence once the relevant movement selector cells were activated. For the testing phase, the full 'leaky-integrator' dynamical Eqs. (1), (2), (6) and (7), were implemented. In the simulations, the differential Eqs. (1) and (6) were approximated by Forward Euler finite difference schemes. At the start of the testing phase, all of the firing rates, r_i^S, r_i^M, and r_i^{MS}, were set to zero. Then the agent was simulated with visual input available, with the agent in an initial state $x = f(0)$, but with no movement selector cells active, for 500 timesteps. While the agent maintained this state, the visual input term e_i for each state cell i was set to a Gaussian response profile identical (except for scaling) to that used for the state cell during the learning phase given by Eq. (10). Next the visual input was removed by setting all of the e_i terms to zero, and the agent was allowed to rest in the same state $x = f(0)$ for a further 500 timesteps. This process led to a stable packet of activity within the layer of state cells that represented the initial state of the agent.

To start to perform the motor activity, the firing rates r_i^{MS} of the relevant movement selector cells were set to 1. When the movement selector cells fired, the co-firing of the movement selector cells and the state cells stimulated a relatively large activity packet in the motor network. Then the co-firing of the motor cells and the state cells stimulated further state cells in the appropriate direction in the state network to reflect the altering state of the agent given that the agent was moving. That is, the state representation was updated by motor efference copy. As the activity packet within the state network moved, the activity packet in the motor network, which was stimulated by the co-firing of the state cells and movement selector cells, also moved. Thus, the two networks moved in tandem while the movement selector cells were active. In this way the network repeated the motor sequence that was learned during training. When the movement selector cells stopped firing, the driving input to the motor cells disappeared, and the motor cells ceased to fire. Then the activity packet within the state network was no longer updated by the motor efference copy, and the state representation remained static.

3. Simulation results

The aim of Experiment 1 was to demonstrate, test, and elucidate the fundamental properties of the model. In Experiment 2, we investigate ways of altering the velocity with which movements are performed. In Experiment 3 we illustrate motor behaviour in which the same motor command reflected in the firing of the motor cells can occur during different parts of a motor sequence in which the current postural state may be different. In Experiment 4 we investigate the operation of the model when it is required to reach a target postural state from different arbitrary starting positions.

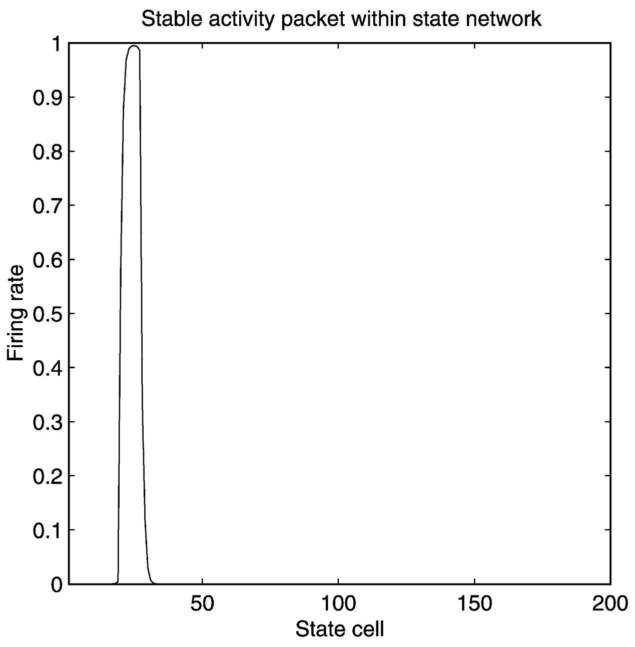


Fig. 2. Experiment 1: firing rates of state cells before movement. There was a stable packet of activity within the state network that represented the current state of the agent.

3.1. Experiment 1: learning a motor sequence

The investigations of Experiment 1 are shown in Figs. 2–4.[5] In Experiment 1, the network was trained with the agent running through a motor sequence, with the postural state x of the agent and current motor activity y running in lock-step from $x = 0.1$ to 0.9, and from $y = 0.1 to 0.9$, with $x = y$. During the training, the selection of this movement was represented by setting the firing rates r^{MS} of movement selector cells 1–5 to 1, with the firing rates of the remaining movement selector cells 6–200 set to 0. (Any subset of movement selector cells could have been chosen to represent the movement). The result of the training (to be shown in Fig. 3) is that movement selector cells 1–5 came to represent the movement just described.

After the training was completed, the agent was simulated with visual input available, with the agent

[5] In this simulation we used the following parameter values. The parameters governing the response properties of the state and motor cells during learning were: $\sigma^S = 0.02$ and $\sigma^M = 0.02$. Further parameters governing the learning were: $\eta = 0.9$, $k^1 = 0.001$, $k^2 = 0.001$, and $k^3 = 0.001$. The parameters governing the leaky-integrator dynamical equations (1) and (6) were: $\tau = 1$, $\phi_0 = 3 \times 10^5$, $\phi_1 = 5 \times 10^6$, $\phi_2 = 2.5 \times 10^6$, and $w^{INH} = 0.0055$. The parameters governing the sigmoid activation functions were as follows. For the state cells we used: $\alpha^{HIGH} = 0.0$, $\alpha^{LOW} = -20.0$, $\gamma = 0.5$, and $\beta = 0.1$. For the motor cells we used: $\alpha^{HIGH} = 10.0$, $\alpha^{LOW} = 10.0$, $\gamma = 0.5$, and $\beta = 0.3$. Since we set $\alpha^{HIGH} = \alpha^{LOW}$ for the motor cells, there was no enhancement of the firing rates of motor neurons with already high firing rates, and the parameter γ was redundant. Finally, for the numerical simulations of the leaky-integrator dynamical Eqs. (1) and (6) we employed a Forward Euler finite difference method with the timestep set to 0.2.

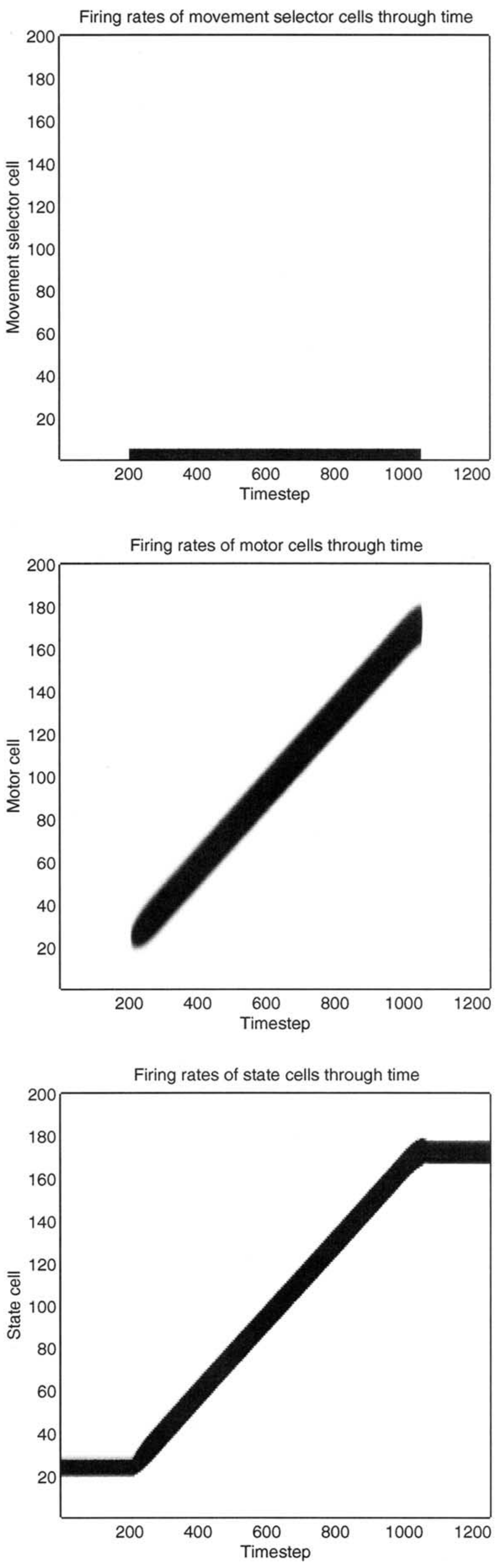


Fig. 3. Experiment 1: firing rate profiles within the movement selector network, state network and motor network through time. Top: firing rates of movement selector cells through time. From timesteps 1 to 200, all movement selector cells were quiescent. At timestep 201 movement selector cells 1–5 became active, and remained active until timestep 1050. The activity of these movement selector cells initiated the learned motor sequence during this time interval. From timesteps 1051 to 1250 all movement selector cells were again quiescent. Middle: firing rates of motor cells through time. From timesteps 1 to 200, all motor cells were quiescent. At timestep 201, the appropriate motor cells were stimulated by the movement selector cells, and the agent began to move through the learned motor sequence. Bottom: firing rates of state cells through time. From timesteps 1 to 200, there was a stationary, stable activity packet within the state network centred around $x = 0.1$. When the movement selector cells became active at timestep 201, the activity packet within the state network started to move to track the state of the agent as the motor sequence was performed. When the motor sequence was completed by timestep 1051 and the movement selector cells became inactive, the activity packet within the state network remained stationary at $x = 0.9$ reflecting the end state of the agent after the movement. During the movement, the activity packets in the state and motor networks moved in synchrony, with the network firing patterns running through the learned motor sequence maintaining the relation $x = y$. From timesteps 1051 to 1250 the movement selector cells stopped firing, and all motor cells were again quiescent.

maintained in an initial state $x = 0.1$, but with no movement selector cells active, for 500 timesteps. (As described in Section 2.2, the visual input term e_i was set to the same Gaussian response profile as that used for state cell i during the learning phase given by Eq. (10)). Next the visual input was removed by setting all of the e_i terms to zero, and the agent was allowed to rest in the same state for a further 500 timesteps. It is shown in Fig. 2 that 500 timesteps after the external input was set to zero, that the firing rates of the postural state cells r^S maintained a stable packet of activity representing the (continuing, remembered) steady postural state. This continued firing among the state cells is supported by the recurrent synaptic connections w^1.

In Fig. 3 we show the firing rate profiles within the movement selector network, state network and motor network through time as the agent performs the learned motor sequence in the absence of the motor training signal t, and without the external (visual or proprioceptive) input e. We show in Fig. 3 that applying steady activity to movement selector cells 1–5 from timestep 200 to 1050 first activates (through the synapses w^3) the motor cells, the firing of which alters the state being represented by the network of state cells via connections w^2, which in turn shifts the motor state represented by the motor cells through synapses w^3. The results produced are continuously moving motor and postural states, as shown in Fig. 3. In this way, the activity packets in the state and motor networks moved in synchrony, with the network firing patterns running through the learned motor sequence maintaining the relation $x = y$. From timesteps 1051 to 1250 the movement selector cells stopped firing, and all motor cells were again quiescent. Finally, in additional simulations, even when the speed of the activity packets was varied by altering the firing rates

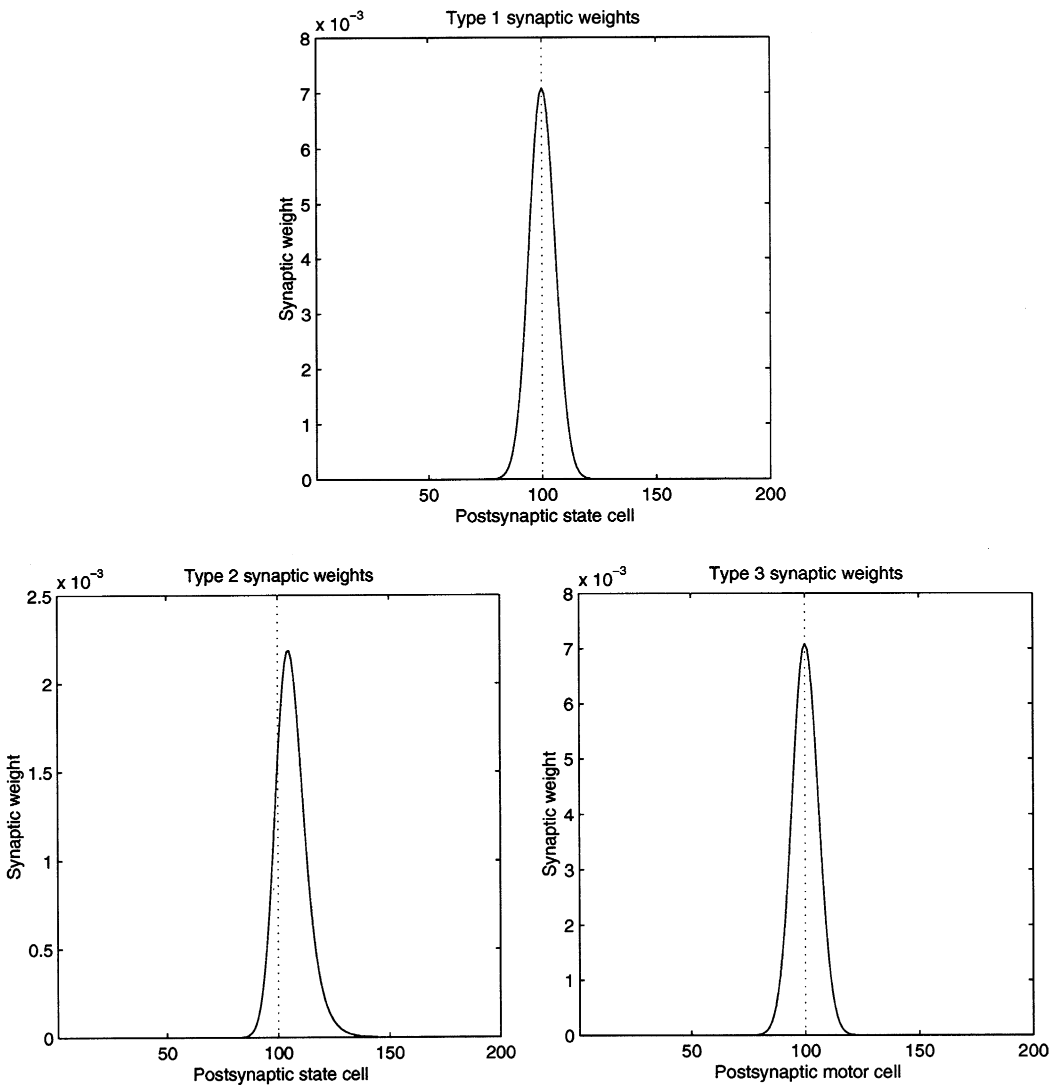


Fig. 4. Experiment 1: synaptic weight profiles for type 1, 2 and 3 synaptic weights. Top: type 1 synaptic weights w^1_{ij} where the presynaptic state cell was $j = 100$ which fired maximally when the state of agent was approximately $x = 0.5$ during the learning phase. The value of the weights w^1_{ij} with $j = 100$ is shown for all postsynaptic state cells $i =\in [1, 200]$. Bottom left: type 2 synaptic weights w^2_{ijk} where the presynaptic state cell j was $j = 100$ which fires maximally when the state of agent was approximately $x = 0.5$ during the learning phase, and the presynaptic motor cell was $k = 100$ which fired maximally when the motor activity of agent was approximately $y = 0.5$ during the learning phase. The value of the weights w^2_{ijk} with $j = 100$ and $k = 100$ is shown for all postsynaptic state cells $i =\in [1, 200]$. Bottom right: type 3 synaptic weights w^3_{ijk} where the presynaptic state cell j was $j = 100$ which fired maximally when the state of agent was approximately $x = 0.5$ during the learning phase, and the presynaptic movement selector cell was $k = 1$ which was one of the five movement selector cells that were associated with the learned motor sequence. The value of the weights w^3_{ijk} with $j = 100$ and $k = 1$ is shown for all postsynaptic motor cells $i =\in [1, 200]$.

r_i^{MS} of the movement selector cells, the activity packets in the state and motor networks remained in synchrony, maintaining the relation $x = y$.

Three important observations from the results shown in Fig. 3, and from the additional simulations in which the firing rates r_i^{MS} of the movement selector cells were varied, were as follows. First, the actions always occurred in the same sequence. Secondly, for a particular fixed firing rate of the movement selector cells, the size of the activity packet within the motor network, which we take to govern the force of the movement, remained the same throughout the motor sequence, and so the agent applied the same relative force in each part of the motor sequence. Thirdly, for a particular fixed firing rate of the movement selector cells, the speeds of the activity packets within the state and motor networks, which represent the speed of the agent's movement, remained the same throughout the motor sequence, and so the agent spent the same relative proportions of its time in each part of the motor sequence.

In Fig. 4 we show the recurrent synaptic weights w^1_{ij} between state cells, the synaptic weights w^2_{ijk} from Sigma–Pi couplings of the state cells and the motor cells to the state cells, and the synaptic weights w^3_{ijk} from the Sigma–Pi couplings of the movement selector cells and state cells to the motor cells. In the top plot of Fig. 4 we show the type 1 synaptic weights w^1_{ij} where the presynaptic state cell was $j = 100$ which fired maximally when the state of agent was approximately $x = 0.5$ during the learning phase. The value of the weights w^1_{ij} with $j = 100$ is shown for all postsynaptic state cells $= i \in [1, 200]$. It can be seen that the learning resulted in nearby state cells in the state space, which need not be at all close to each other in the brain, developing stronger recurrent synaptic connections than state cells that were more distant in the state space. Furthermore, it can be seen that the graph of the recurrent weights is symmetric about the central node and is approximately a Gaussian function of the distance between the state cells in the state space. This is important for stably supporting the activity packet at a particular location when the agent is stationary. In the bottom left plot of Fig. 4 we show the type 2 synaptic weights w^2_{ijk} where the presynaptic state cell j was $j = 100$ which fired maximally when the state of agent was approximately $x = 0.5$ during the learning phase, and the presynaptic motor cell was $k = 100$ which fired maximally when the motor activity of agent was approximately $y = 0.5$ during the learning phase. The value of the weights w^2_{ijk} with $j = 100$ and $k = 100$ is shown for all postsynaptic state cells $i =\in [1, 200]$. The synaptic weights w^2_{ijk}, which were calculated with a trace learning rule, were asymmetric. The asymmetry of the weights w^2_{ijk}, introduced by trace learning, played a key rôle in moving the activity packets in state and motor networks through their correct sequences when the movement selector cells fired in the dark. In the bottom right plot of Fig. 4 we show the type 3 synaptic weights w^3_{ijk} where the presynaptic state cell j was $j = 100$ which fired maximally when the state of agent was approximately $x = 0.5$ during the learning phase, and the presynaptic movement selector cell was $k = 1$ which was one of the five movement selector cells that were associated with the learned motor sequence. The value of the weights w^3_{ijk} with $j = 100$ and $k = 1$ is shown for all postsynaptic motor cells $i =\in [1, 200]$. The synaptic weights w^3_{ijk}, which were calculated according to a Hebb rule without trace terms, were symmetric.

3.2. Experiment 2: controlling the speed and force of movement

The investigations of Experiment 2 are illustrated in Figs. 5 and 6.[6] In Experiment 2, we demonstrate how increasing the firing rates of the movement selector cells increases the force, and hence speed, of the motor action, and how the network is able to compensate by increasing the speed of the path integration within the state network so that the agent's internal representation of its state may keep in step with its actual state. This is necessary in order for the agent to perform the correct motor action when the agent is in a particular state.

Fig. 5 shows the sizes of the activity packets within the continuous attractor network of state cells (left) and the network of motor cells (right) for different firing rates r^{MS}_i of the relevant movement selector cells. When the firing rates of the movement selector cells were less than or equal to 0.5, there was a moderate amount of activity within the network of state cells, but only a very low level of activity within the motor network. However, when the firing rates of the movement selector cells increased above 0.5 the sizes of the activity packets within the state and motor networks started to rise sharply with the firing rates of the movement selector cells. However, while the size of the activity packet within the state network increased by just under a factor of two as the firing rates of the movement selector cells increased from 0.5 to 1.0, the size of the activity packet within the motor network increased by an order of magnitude. In a sense, once the firing rates of the movement selector cells reached a threshold of approximately 0.5, the motor network 'switched on' to perform the movement. The key observation from the results shown in Fig. 5 is that the size of the activity packet within the motor network, which we take to govern the force of the movement, could be controlled through control of the firing rates of the relevant movement selector cells. As the firing rates of the movement selector cells were increased, the size of the activity packet within the motor network increased, and the desired movement was performed with greater force. Thus, a movement could be performed with arbitrary force. Furthermore, as shown in Fig. 3, for a particular fixed firing rate of the movement selector cells, the size of the activity packet within the motor network remained the same throughout the motor sequence, and so the agent applied the same relative force in each part of the motor sequence.

Fig. 6 shows the speeds of the activity packets within the continuous attractor network of state cells (left) and the network of motor cells (right) for different firing rates r^{MS}_i of the relevant movement selector cells. Because the motor cells i were stimulated through the w^3_{ijk} synaptic connections by the co-firing of the corresponding state cells j with $y_i = x_j$ and the movement selector cells k, the location of the activity packet within the motor network reflected the location of the activity packet within the state network, and hence the two plots were almost identical. When the firing rates of the movement selector cells were less than or equal to 0.5, the activity packet within the network of state cells was held stably at its initial location due to the non-linearity within the sigmoid activation functions described by Eq. (9), and the speeds of the activity packets within the state and motor networks were therefore zero. However, when the firing rates of the movement selector cells increased above 0.5 the speeds of the activity packets within the state and motor networks started to rise sharply with the firing rates of the movement selector cells. The reason for this was as follows. The state cells received two kinds of inputs: $\sum_j w^1_{ij} r^{\mathrm{S}}_j$ and $\sum_{j,k} w^2_{ijk} r^{\mathrm{S}}_j r^{\mathrm{M}}_k$. The input

[6] The model parameters used for Experiment 2 were the same as those used for Experiment 1.

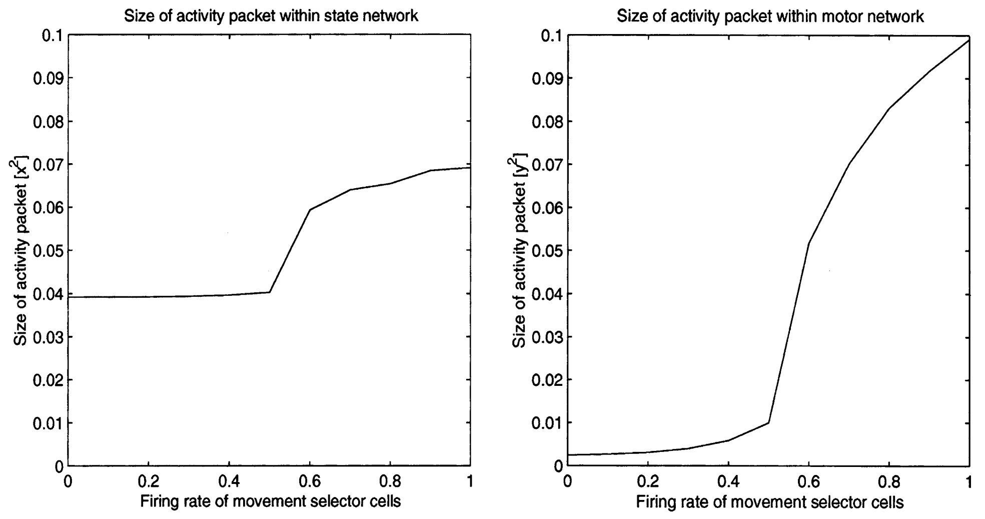


Fig. 5. Experiment 2: the plot shows the sizes of the activity packets within the continuous attractor network of state cells (left) and the network of motor cells (right) for different firing rates r_i^{MS} of the relevant movement selector cells. (The size of the activity packets was calculated by numerically integrating the neuronal firing rates over the relevant space. That is, the firing rates of the state cells were integrated over the state space $x \in [0, 1]$, and the firing rates of the motor cells were integrated over the motor space $y \in [0, 1]$.) When the firing rates of the movement selector cells were less than or equal to 0.5, there was a moderate amount of activity within the network of state cells, but only a very low level of activity within the motor network. However, when the firing rates of the movement selector cells increased above 0.5 the sizes of the activity packets within the state and motor networks started to rise sharply, and rose monotonically with the firing rates of the movement selector cells. However, while the size of the activity packet within the state network increased by just under a factor of two as the firing rates of the movement selector cells increased from 0.5 to 1.0, the size of the activity packet within the motor network increased by an order of magnitude.

term responsible for moving the activity packet within the state network was the second term $\sum_{j,k} w_{ijk}^2 r_j^{\mathrm{S}} r_k^{\mathrm{M}}$, which involved asymmetric w_{ijk}^2 synaptic connections from Sigma–Pi couplings of state cells j and motor cells k. As the size of the activity packet within the motor network increased, so did the size of this input term, and so the activity packet within the state network moved faster. Thus, the speed of the activity packet within the state network directly reflected the size of the activity packet within the motor network and hence the force of

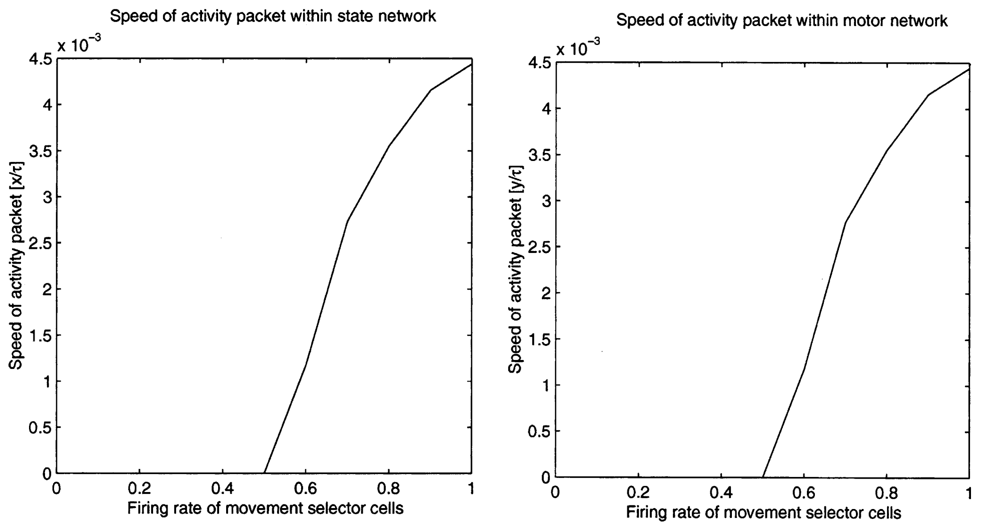


Fig. 6. Experiment 2: the plot shows the speeds of the activity packets within the continuous attractor network of state cells (left) and the network of motor cells (right) for different firing rates r_i^{MS} of the relevant movement selector cells. Because the motor cells i were stimulated through the w_{ijk}^3 synaptic connections by the co-firing of the corresponding state cells j with $y_i = x_j$ and the movement selector cells k, the location of the activity packet within the motor network reflected the location of the activity packet within the state network, and hence the two plots were almost identical. When the firing rates of the movement selector cells were less than or equal to 0.5, the activity packet within the network of state cells was held stably at its initial location due to the non-linearity within the sigmoid activation functions described by Eq. (9), and the speeds of the activity packets within the state and motor networks were therefore zero. However, when the firing rates of the movement selector cells increased above 0.5 the speeds of the activity packets within the state and motor networks started to rise sharply, and rose monotonically with the firing rates of the movement selector cells.

the movement. Furthermore, the curve showing the speed of the activity packet within the motor network was identical to the curve showing the speed of the activity packet within the state network, as discussed earlier. Indeed, it can be seen that the curves showing the speeds of the activity packets in the state and motor networks as shown in Fig. 6 were remarkably similar to the curve showing the size of the activity packet within the motor network as shown in the right plot of Fig. 5. Thus, the forward model implemented by the synapses w^2_{ijk} is able to take into account the force with which the motor sequence is performed, and hence the speed of movement of the agent, when updating the representation within the state network. This is a fundamentally important property of the model, and is the mechanism by which the network is able to operate across different forces, and hence speeds, of movement. The network is able to achieve this even when it is only trained with one speed of movement. If the mechanism responsible for updating the state network in the absence of external visual or proprioceptive input was not able to take into account the force, and hence speed, with which the agent was moving, the state representation would become inaccurate and fail to reflect the true state of the agent. In this case, for each momentary physical state of the agent, the agent would not be able to stimulate the correct motor neurons for the desired motor sequence. Instead, the agent would continue to stimulate the motor neurons that were associated during learning with the state cells currently active (which would no longer reflect the true state of the agent), and so the motor sequence would not be performed correctly.

The key observation from the results shown in Fig. 6 is that the speed of the activity packets within the state and motor networks, which represent the speed of

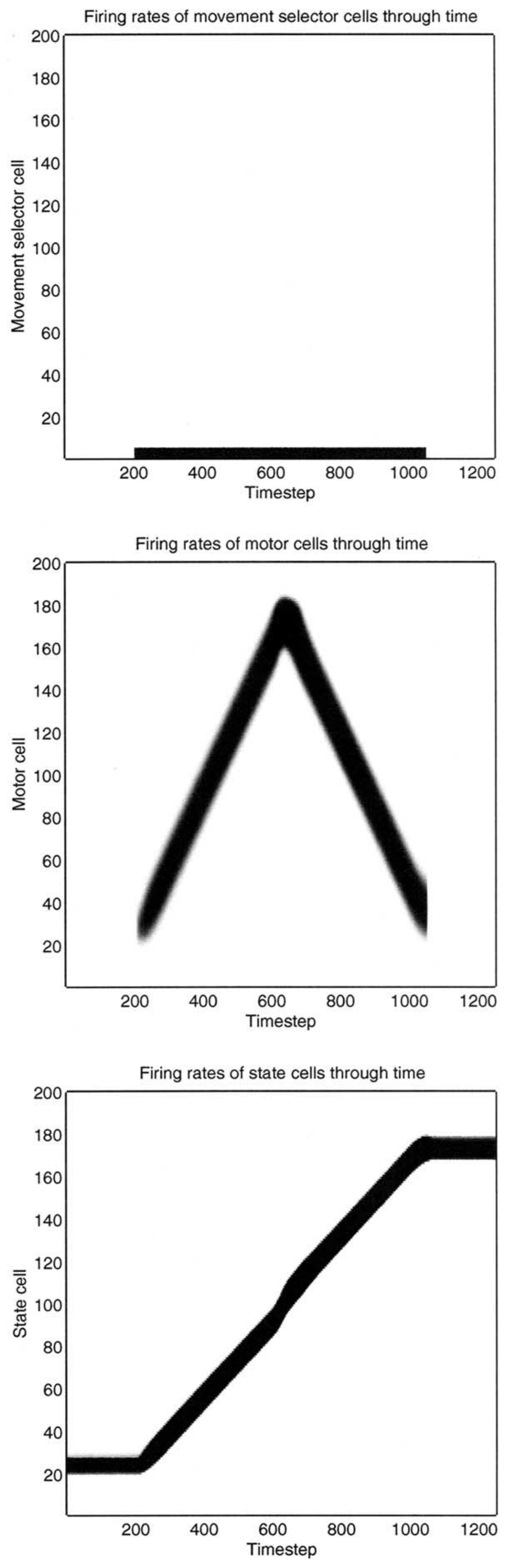


Fig. 7. Experiment 3: firing rate profiles within the movement selector network, state network and motor network through time. Top: firing rates of movement selector cells through time. From timesteps 1 to 200, all movement selector cells were quiescent. At timestep 201 movement selector cells 1–5 became active, and remained active until timestep 1050. The activity of these movement selector cells initiated the learned motor sequence during this time interval. From timesteps 1051 to 1250 all movement selector cells were again quiescent. Middle: firing rates of motor cells through time. From timesteps 1 to 200, all motor cells were quiescent. At timestep 201, the appropriate motor cells were stimulated by the movement selector cells, and the agent began to move through the learned motor sequence. Bottom: firing rates of state cells through time. From timesteps 1 to 200, there was a stationary, stable activity packet within the state network centred around $x = 0.1$. When the movement selector cells became active at timestep 201, the activity packet within the state network started to move to track the state of the agent as the motor sequence was performed. When the motor sequence was completed by timestep 1051 and the movement selector cells became inactive, the activity packet within the state network remained stationary at $x = 0.9$ reflecting the end state of the agent after the movement. During the movement, the activity packets in the state and motor networks moved with the same relationship that occurred during the learning phase. From timesteps 1051 to 1250 the movement selector cells stopped firing, and all motor cells were again quiescent.

the agent's movement, could be controlled through control of the firing rates of the relevant movement selector cells. As the firing rates of the movement selector cells were increased, the size of the activity packet within the motor network increased and the desired movement was performed with greater force. However, the increase in the size of the activity packet within the motor network led to an increase in the speeds of the activity packets within the state and motor networks, and so the network was able to operate at greater speeds. Thus, a movement could be performed with arbitrary speed. Furthermore, as shown in Fig. 3, for a particular fixed firing rate of the movement selector cells, the speeds of the activity packets within the state and motor networks, which represent the speed of the agent's movement, remained the same throughout the motor sequence, and so the agent spent the same relative proportions of its time in each part of the motor sequence.

From the above simulations we have demonstrated how our model captures the key criteria of Schmidt's generalised motor control program. That is, key properties of the model are as follows: (i) the movement can occur at arbitrary speeds; (ii) the movement can occur with arbitrary force; (iii) the agent spends the same relative proportions of its time in each part of the motor sequence; (iv) the agent applies the same relative force in each part of the motor sequence; (v) the actions always occur in the same sequence.

3.3. Experiment 3: motor sequences with the same motor commands executed during different parts of the motor programme

The results from Experiment 3 are shown in Fig. 7.[7] In Experiment 3, the network was trained on a different motor sequence in which the state x of the agent and the motor activity y do not run in phase. For this simulation, during learning the activity packet within the motor network moved back and forth, at twice the speed of the activity packet in the state network. This was done to demonstrate the ability of the model to learn more general motor sequences.[8] In particular, we demonstrate that the network can learn a motor sequence involving an association between the state $x = f(t)$ and motor activity $y = g(t)$ where $(f(t), g(t))$ is a non-monotonic curve. As the agent's state x moves in the positive x-direction, the motor activity y first moves in the positive y-direction, and then moves in the opposite direction. For this motor sequence, each motor activity y occurred for more than one state x of the agent. As for Experiment 1, during this movement, the firing rates of movement selector cells 1–5 were set to 1, and the firing rates of the remaining movement selector cells 6–200 were set to 0. Thus, the motor sequence was learned by movement selector cells 1–5, and it was these cells that needed to be activated after training in order to perform the learned motor sequence. In Fig. 7 we show the firing rate profiles within the movement selector network, state network and motor network through time as the agent performs the learned motor sequence in the absence of external input. During the movement, the activity packets in the state and motor networks moved with the same relationship that occurred during the learning phase. Thus, the network successfully learned this second motor sequence in which each motor activity y occurred for more than one state x of the agent. Finally, in additional simulations, even when the speed of the activity packets was varied by altering the firing rates r_i^{MS} of the movement selector cells, the activity packets in the state and motor networks maintained the relation which was learned during training.

3.4. Experiment 4: learning to reach a target state

Numerical results from Experiment 4 are shown in Figs. 8 and 9.[9] In Experiment 4, we demonstrated the ability of the network to encode multiple motor programmes, where each motor programme involved the agent moving to a final target state. Such motor programmes might represent motor activities like prehension, where the agent has to reach to a target location. In this case, the firing of the state cells would represent the position of the hand in space, and the firing of the movement selector cells would represent the target location to which the agent decided to reach.

In the simulations, the 200 movement selector cells were mapped onto a regular grid of different target states x^T, where for each movement selector cell i there was a unique preferred target state x_i^T for which the cell was stimulated maximally. Then, during both the learning and testing phases, the firing rates of the movement selector cells were set according to the following Gaussian response profile

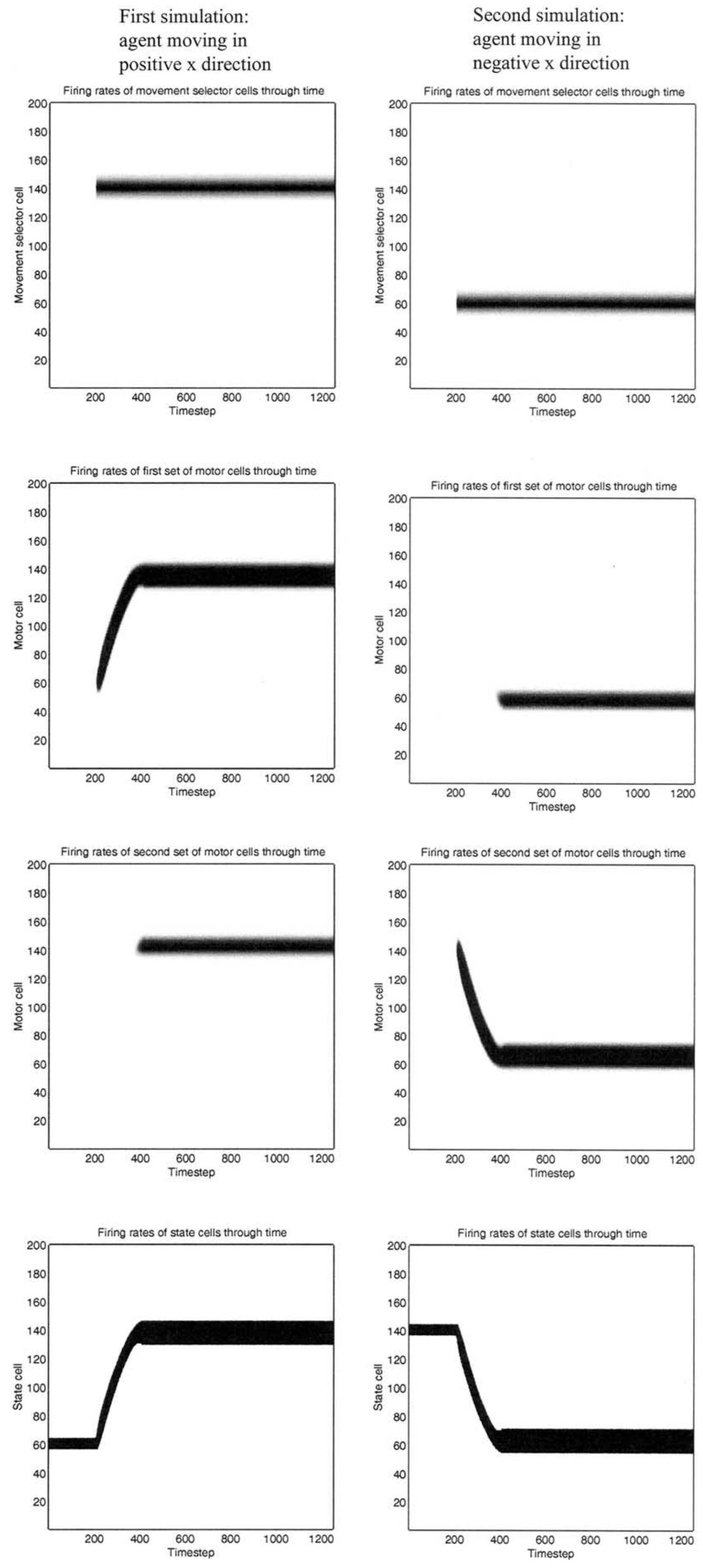
$$r_i^{MS} = \exp(-(s_i^{MS})^2/2(\sigma^{MS})^2), \tag{15}$$

where s_i^{MS} was the absolute value of the difference between the actual target state x^T of the agent and the preferred target state x_i^T for movement selector cell i, and σ^{MS} was

[7] The model parameter values used for Experiment 3 were the same as those used for Experiment 1, except for two exceptions. In Experiment 3, we set $\phi_1 = 6.2 \times 10^6$ and $\phi_2 = 3.1 \times 10^6$.

[8] For Experiment 3, the network was trained with the agent running through a motor sequence, with the state x of the agent running from $x = 0.1$ to 0.9. However, during this, the motor activity y moved at twice the speed of the state x. As the state x of the agent moved from $x = 0.1$ to 0.5, the motor activity y moved from $y = 0.1$ to 0.9. Then, while the state x of the agent moved from $x = 0.5$ to 0.9, the motor activity y moved backwards from $y = 0.9$ to 0.1.

[9] The model parameter values used for Experiment 4 were the same as those used for Experiment 1, except for the following exceptions. In Experiment 4 we set $\phi_1 = 4 \times 10^7$ and $\phi_2 = 1.3 \times 10^5$. In addition, since the increased amount of learning involved in Experiment 4 leads to larger recurrent synaptic weights w_{ij}^1, the lateral inhibition constant w^{INH} was increased to 1.1.


First simulation: agent moving in positive x direction
Second simulation: agent moving in negative x direction
Firing rates of movement selector cells through time
Movement selector cell
Firing rates of first set of motor cells through time
Firing rates of second set of motor cells through time
Motor cell
Firing rates of state cells through time
State cell
Timestep

the standard deviation. For each movement selector cell i, s_i^{MS} was given by

$$s_i^{\mathrm{MS}} = |x_i^{\mathrm{T}} - x^{\mathrm{T}}|. \quad (16)$$

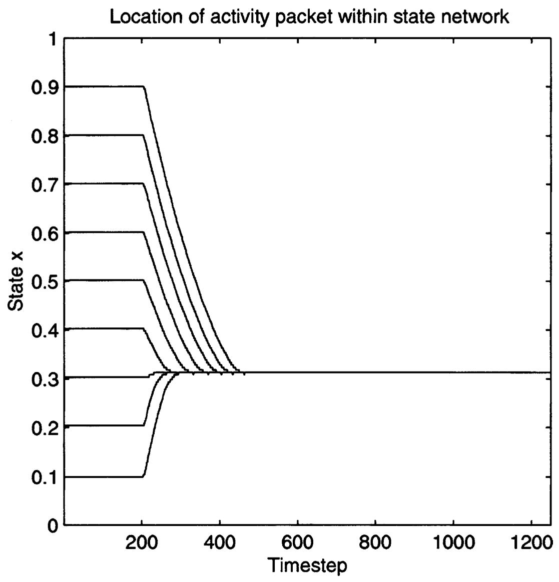
Since the motor programmes in Experiment 4 involved movement of the state of the agent in the positive and negative x-directions, an additional set of 200 motor cells were needed to encode the movements of the agent in the negative x direction. This was because a fundamental aspect of the model is that a particular combination of the agent's state and motor activity should always lead to the same next state, regardless of which particular motor sequence is being executed. Hence, different motor cells were required for when the agent was moving in the positive or negative x directions. The enlarged network thus had two sets of motor cells, where the first set of 200 motor cells encoded movements in the positive x direction, and the second set of 200 motor cells encoded movements in the negative x direction. Both sets of motor cells were mapped onto two separate regular grids of different instantaneous motor activities, where for each motor cell i there was a unique preferred instantaneous motor activity y_i for which the cell was stimulated maximally. Then, during learning, the firing rates of the two sets of motor cells were set as follows. Firstly, the firing rates of the first set of motor cells were set according to their Gaussian response profiles (12) whenever the agent was moving in the positive x direction, and were set to zero whenever the agent was moving in the negative x direction. Secondly, the firing rates of the second set of motor cells were set according to their Gaussian response profiles (12) whenever the agent was moving in the negative x direction, and were set to zero whenever the agent was moving in the positive x direction.

The learning phase proceeded as follows. For each of the 200 target states represented by the movement selector cells, the agent was simulated performing the following two separate motor sequences: (i) moving in the positive x direction from $x = 0$ to $x = x^{\mathrm{T}}$, and (ii) moving in the negative x direction from $x = 1$ to $x = x^{\mathrm{T}}$. During movements in the positive x direction, the state x of the agent, and the instantaneous motor activity y represented by the first set of motor cells moved in lockstep with $x = y$. During movements in the negative x direction, the state x of the agent, and the instantaneous motor activity y represented by the second set of motor cells moved in lockstep with $x = y$. Training the network with many target locations x^{T} led to the network learning the ability to move the agent towards any desired target location within the state space $x \in [0, 1]$ of the agent, where such motor programmes may be initiated after training by stimulating a packet of activity at the desired location within the network of movement selector cells. In this experiment, the representation provided by the movement selector cells is a continuous representation of the target location space, and we suggest that such networks could themselves operate as continuous attractor networks.

The testing phase began with a period of rest in which the agent remained still, and then proceeded with a period of movement in which a target state x^{T} was chosen, and the movement selector cells were set to fire according to their Gaussian response profiles (15). In these simulations, the movement selector cells were set to continue firing even after the agent had reached its target state. The aim here was to simulate the agent maintaining the postural target state through continued motor stimulation. An example of such a motor programme in biological agents might be reaching and maintaining its target position, where maintaining the target position requires continued muscle stimulation. A number of simulations were performed with different initial states $x(0)$ and different target states x^{T} represented by the network of movement selector cells.

In Fig. 8 we show two simulations demonstrating the ability of the network to learn to move the agent to chosen target states. On the left are results from the first simulation showing the evolution of the network from initial state

Fig. 8. Experiment 4: two simulations demonstrating the ability of the network to learn to move the agent to chosen target states. On the left are results showing the evolution of the network from initial state $x(0) = 0.3$ to target state $x^{\mathrm{T}} = 0.7$, and on the right are results showing the evolution of the network from initial state $x(0) = 0.7$ to target state $x^{\mathrm{T}} = 0.3$. For each simulation we show the firing rate profiles through time within the movement selector network, the first set of motor cells which encode movements in the positive x direction, the second set of motor cells which encode movements in the negative x direction, and the state network. Top row: firing rates of movement selector cells through time. From timesteps 1 to 200, all movement selector cells were quiescent. At timestep 201 the movement selector cells were set to fire according to their Gaussian response profiles given a target state x^{T}, and these movement selector cells remained active until the end of the simulation at timestep 1250. The activity of these movement selector cells initiated the learned motor sequence during this time interval. Second row: firing rates of first set of motor cells through time. Third row: firing rates of second set of motor cells through time. From timesteps 1 to 200, all motor cells were quiescent. At timestep 201, the appropriate motor cells were stimulated by the movement selector cells, and the agent began to move through the learned motor sequence. For the simulation shown on the left, the state of the agent moved in the positive x direction from $x(0) = 0.3$ to $x^{\mathrm{T}} = 0.7$, and so the relevant motor cells from the first set of motor cells were stimulated. However, when the agent reached the target state $x^{\mathrm{T}} = 0.7$, the relevant motor cells from the second set of motor cells were stimulated. For the simulation shown on the right, the state of the agent moved in the negative x direction from $x(0) = 0.7$ to $x^{\mathrm{T}} = 0.3$, and so the relevant motor cells from the second set of motor cells were stimulated. However, when the agent reached the target state $x^{\mathrm{T}} = 0.3$, the relevant motor cells from the first set of motor cells were stimulated. Bottom row: firing rates of state cells through time. From timesteps 1 to 200, there was a stationary, stable activity packet within the state network centred around the initial state $x(0)$. When the movement selector cells became active at timestep 201, the activity packet within the state network started to move to track the state of the agent as the motor sequence was performed. When the motor sequence was completed, even with the movement selector cells still active, the activity packet within the state network remained stationary at the target state x^{T} reflecting the end state of the agent after the movement. During the movement, the activity packets in the state and the two sets of motor cells moved with the same relationship that occurred during the learning phase.



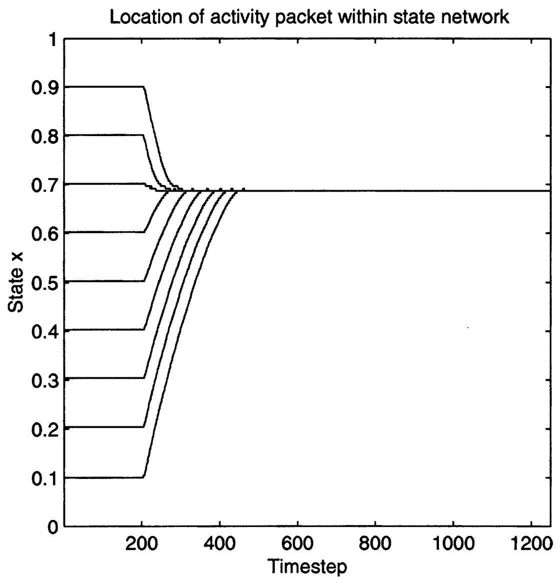


Fig. 9. Experiment 4: examples of the agent moving to two different target states x^T from many different initial states $x(0)$. Both top and bottom plots show many time courses of the position of the activity packet within the state network for different initial locations $x(0)$. For the top plot, the final target state represented by the firing of the movement selector cells is $x^T = 0.3$. While for the bottom plot, the final target state represented by the firing of the movement selector cells is $x^T = 0.7$. In all simulations, the movement selector cells were quiescent from timesteps 1 to 200. At timestep 201 the movement selector cells were set to fire according to their Gaussian response profiles given the relevant target state, $x^T = 0.3$ or 0.7, and these movement selector cells remained active until the end of the simulation at timestep 1250.

$x(0) = 0.3$ to target state $x^T = 0.7$, and on the right are results from the second simulation showing the evolution of the network from initial state $x(0) = 0.7$ to target state $x^T = 0.3$. For each simulation we show the firing rate profiles through time within the movement selector network, the first set of motor cells which encode movements in the positive x direction, the second set of motor cells which encode movements in the negative x direction, and the state network.

At timestep 201 the movement selector cells were set to fire according to their Gaussian response profiles given a target state x^T, and these movement selector cells remained active until the end of the simulation at timestep 1250. The activity of these movement selector cells initiated the learned motor sequence during this time interval. At timestep 201, the appropriate motor cells were stimulated by the movement selector cells, and the agent began to move through the learned motor sequence. For the simulation shown on the left, the first set of motor cells were stimulated and the state of the agent moved in the positive x direction from $x(0) = 0.3$ to $x^T = 0.7$. However, when the agent reached the target state $x^T = 0.7$, certain motor cells from the second set of motor cells were stimulated. This is because, during the learning phase, when the agent was learning to move to the target state $x^T = 0.7$ in the negative x direction, at the end of the movement the network learned to associate the co-firing of the movement selector cells that represented the target state $x^T = 0.7$ and the state cells that represented the states near to $x = 0.7$, with the firing of the corresponding motor cells within the second set of motor cells that represented instantaneous motor activities near to $y = 0.7$. Hence, during testing, the co-firing of the movement selector cells that represented the target state $x^T = 0.7$ and the state cells that represented the states near to $x = 0.7$, stimulated the firing of the corresponding motor cells from the second set of motor cells that represented instantaneous motor activities near to $y = 0.7$. The activation of the second set of motor cells, which encode movement in the negative x direction, after the agent reached the target state $x^T = 0.7$ helped to stop the agent from continuing to move through the target state without stopping. For the simulation shown on the right, motor cells from the second set of motor cells were stimulated, and the state of the agent moved in the negative x direction from $x(0) = 0.7$ to $x^T = 0.3$.

In Fig. 9 we show examples of the agent moving to two different target states x^T from many different initial states $x(0)$. Both top and bottom plots show many time courses of the position of the activity packet within the state network for different initial locations $x(0)$. For the top plot, the final target state represented by the firing of the movement selector cells is $x^T = 0.3$. While for the bottom plot, the final target state represented by the firing of the movement selector cells is $x^T = 0.7$. In all simulations, the movement selector cells were quiescent from timesteps 1 to 200. At timestep 201 the movement selector cells were set to fire according to their Gaussian response profiles given the relevant target state, $x^T = 0.3$ or 0.7, and these movement selector cells remained active until the end of the simulation at timestep 1250.

4. Learning mechanisms to remove the effects of errors present in the motor sequence during training

In reality biological agents may initially perform the motor task poorly at the start of training. That is, the motor sequence stimulated during the initial learning phase may contain significant error such that the agent does not follow the desired motor sequence perfectly during training. In this section we propose some learning mechanisms that may help to solve this problem.

First, one way of reducing the effects of motor error during learning might be through 'explicit learning', which involves making use of an explicit reward signal r^{R} that signifies when the motor sequence is being performed well by the agent. Let us assume there is a signal r^{R} available which represents the reward, where $r^{\mathrm{R}} = 1$ when the motor sequence is performed correctly, and $r^{\mathrm{R}} = 0$ when the motor sequence is performed incorrectly. Then, the synaptic weights are only updated during learning when the agent is currently performing well and the reward signal r^{R} is set to 1. In this case, the learning rules are modified as follows. Firstly, learning rule (3) becomes

$$\delta w^1_{ij} = k^1 r^{\mathrm{S}}_i r^{\mathrm{S}}_j r^{\mathrm{R}}, \tag{17}$$

where we have scaled the synaptic update by the term r^{R}. Secondly, in a similar way, learning rule (4) becomes

$$\delta w^2_{ijk} = k^2 r^{\mathrm{S}}_i \bar{r}^{\mathrm{S}}_j \bar{r}^{\mathrm{M}}_k r^{\mathrm{R}}. \tag{18}$$

Thirdly, learning rule (8) becomes

$$\delta w^3_{ijk} = k^3 r^{\mathrm{M}}_i r^{\mathrm{S}}_j r^{\mathrm{MS}}_k r^{\mathrm{R}}. \tag{19}$$

The effect of these modifications is to allow the networks to learn only when the agent is performing the motor task correctly.

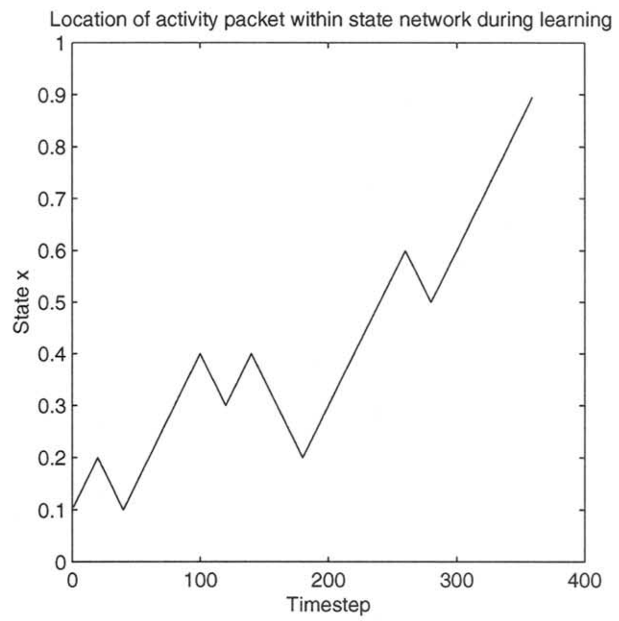
A second way in which the effects of motor error during learning may be reduced is through 'implicit learning' (Pfeifer & Scheier, 1999), which relies on the noisy motor training signal containing a systematic component of the correct signal which is relatively strong compared to any error components. Whether the agent is able to generate such a training signal will depend on the agent's physical morphology and the current level of organisation of its neural architecture. Thus, implicit learning would operate in a 'bootstrapping' manner in that the learning of a new motor programme would rely on previously learned motor competence. In the case where the motor training signal contains a relatively strong component of the correct signal, the errors may wash out during learning and the movement selector cells are able to learn the correct motor sequence. Hence, after training, the movement selector cells stimulate the correct sequence of motor cells with minimal error.

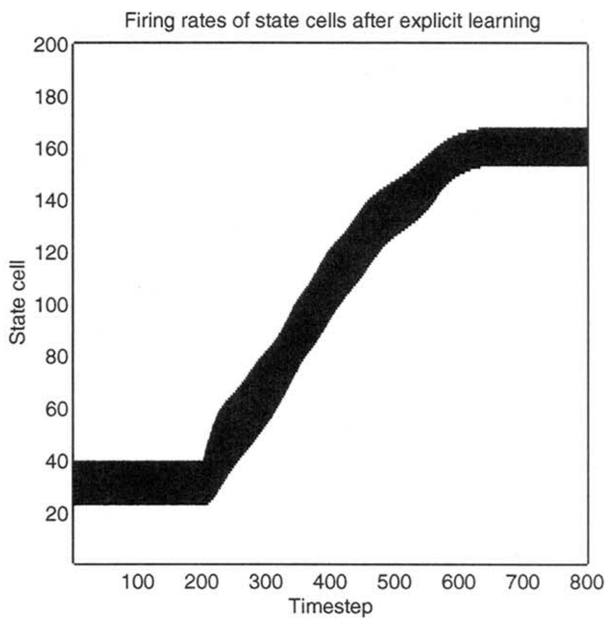
4.1. Simulation results with errors present in motor sequence during training

In this section we present results for Experiment 5, in which the simulations were performed with errors present in motor sequence during training. In the simulations performed here, the aim was for the agent to learn to perform the same motor sequence as used for Experiment 1, with the postural state x of the agent and current motor activity y running in lock-step from $x = 0.1$ to 0.9, and from $y = 0.1$ to 0.9, with $x = y$. During the training, the selection of this movement was represented by setting the firing rates r^{MS} of movement selector cells 1–5 to 1, with the firing rates of the remaining movement selector cells 6–200 set to 0. However, for Experiment 5, the motor training signal was corrupted by noise in the following way. The network was trained in a similar manner to Experiment 1, except that at every 20th timestep the agent had a 2/3 probability of beginning a 20 timestep movement in the positive x direction, and a 1/3 probability of beginning a 20 timestep movement in the negative x direction. Thus, the agent effectively performed a random walk from $x = 0.1$ to 0.9, with a constant bias of moving in the positive x direction. During the simulation of explicit learning with a reward signal, the reward signal r^{R} was set to 1 when the agent was moving in the correct positive x direction, and was set to 0 when the agent was moving in the incorrect negative x direction. This effectively corresponds to giving the agent a reward signal whenever it was getting closer to its end state $x = 0.9$.

Results for Experiment 5 are shown in Fig. 10.[10] In the top left is shown an example of the time course of the position of the activity packet within the state network during one epoch of the learning phase. For each learning epoch, the agent moved in a random walk from $x = 0.1$ to 0.9, with a constant bias of moving in the positive x direction. During the learning phases for both the explicit and implicit learning simulations, the agent performed 100 such learning epochs. In the top right are shown the firing rates of the state cells through time as the agent performs the motor programme after explicit learning with a reward signal. In the bottom row are shown the firing rates of the state cells through time as the agent performs the motor programme after implicit learning without a reward signal. For both simulations with explicit and implicit learning, the agent was able to learn to perform the correct motor sequence fairly well given the amount of error in the training sequence. The activity packets within the state and motor networks moved in synchrony, with the network patterns running through the correct motor sequence maintaining the relation $x = y$.

[10] The model parameter values used for Experiment 5 were the same as those used for Experiment 1, except for the following exceptions. In the simulation with explicit learning (top right of Fig. 10) we set $\phi_1 = 10^7$, $\phi_2 = 1.3 \times 10^5$, and $w^{\mathrm{INH}} = 1.3$. In the simulation with implicit learning (bottom row of Fig. 10) we set $\phi_0 = 2.1 \times 10^5$, $\phi_1 = 6 \times 10^8$, $\phi_2 = 2.5 \times 10^3$, and $w^{\mathrm{INH}} = 1.7$. In addition, for both simulations of explicit and implicit learning, we set $\alpha^{\mathrm{LOW}} = \alpha^{\mathrm{HIGH}} = 0.0$ for the state cells. Since we set $\alpha^{\mathrm{LOW}} = \alpha^{\mathrm{HIGH}}$ for the state cells, there was no enhancement of the firing rates of state cells with already high firing rates, and the parameter γ was redundant.





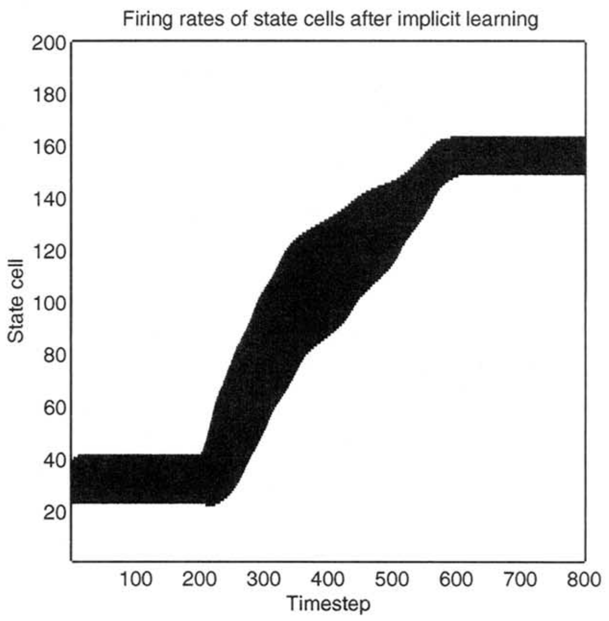


Fig. 10. Experiment 5: simulations with error present in motor training signal during learning phase. Top left: example of the time course of the position of the activity packet within the state network during one epoch of the learning phase. For each learning epoch, the agent moved in a random walk from $x = 0.1$ to 0.9, with a constant bias of moving in the positive x direction. During the learning phases for both the explicit and implicit learning simulations, the agent performed 100 such learning epochs. Top right: firing rates of the state cells through time as the agent performs the motor programme after explicit learning with a reward signal. Bottom row: firing rates of the state cells through time as the agent performs the motor programme after implicit learning without a reward signal. For both simulations with explicit and implicit learning, the agent was able to learn to perform the correct motor sequence fairly well given the amount of error in the training sequence. The activity packets within the state and motor networks moved in synchrony, with the network patterns running through the correct motor sequence maintaining the relation $x = y$.

5. A hierarchical model of motor function

In the model presented earlier, the network learned to use the movement selector cells' input to produce a temporal sequence of motor output, using during training the training motor signal t (Fig. 1). Let the firing of one set of movement selector cells by training come to produce sequence one. Let the firing of a second set of movement selector cells by training come to produce sequence two. Now present a third movement selector cell input pattern throughout the presentation of both movement selector inputs 1 and 2. If the synapses from the movement selector cells to the motor cells are associatively modifiable, then after a few associative pairings, movement selector cell input pattern 3 will produce the complex sequence of movements that was produced by movement selector input 1 followed by movement selector cell input 2. In this way, new, high level, movement selection commands can be automatically generated by the system, without any error feedback.

Another way of implementing hierarchical motor control would be to add another network of the type shown in Fig. 1 hierarchically above the first network. The states in the higher order state attractor could represent high level states such as where one is in a room, and as the high level attractor follows a trained trajectory through its state space, its motor cells with rates r^{MH} would provide an appropriate sequence of movement selector cell inputs r^{MS} for the first (lower) network. The higher level network would have its own set of high level motor signals t^{H} (e.g. a motor plan to move to a new location in the room), and its own set of visual or proprioceptive inputs e^{H} which might represent for example surrounding space, but not information such as limb position which would be represented in the lower level continuous attractor. Such higher level networks might be located in separate cortical areas, for example, in parts of the prefrontal cortex. The concept of high level motor selector cells stimulating directly the lower level motor selector cells may offer a more powerful framework for self-organizing motor control.

Thus, the models presented here may be viewed as a way of self-organizing a hierarchical pyramid of motor programs with increasing levels of complexity (Stringer & Rolls, 2003).

6. Discussion

The acquisition of new motor skills may initially involve specific neural mechanisms that involve a heavy neural processing burden in terms of active attention, and that may also perform poorly in terms of error in the initial attempts at the motor action. However, with practice a motor skill may be mastered such that, after training, little active attention is required to perform the task, and in addition the task may be performed with much reduced error. In this paper we present a network model that is able to learn arbitrary motor sequences and simulates the acquisition of motor skills by biological agents as outlined earlier.

In this paper we have presented a model of motor control in which the synaptic connectivity is able to self-organize during an initial training phase using biologically plausible learning rules. After training, the network is able to produce whole sequences of motor cell firing, to implement particular learned movements, even when the state cells receive no visual or proprioceptive input. In the models presented here the synaptic connectivity of the network self-organizes during learning such that, after training, the correct motor sequence may be stimulated automatically by the firing of a single set of 'movement selector' cells.

We showed that the models developed here are able to demonstrate the key features of Schmidt's Generalised Motor Control Program (Schmidt, 1987). That is, we showed that: (i) the movement can occur at arbitrary speeds; (ii) the movement can occur with arbitrary force; (iii) the agent spends the same relative proportions of its time in each part of the motor sequence[11]; (iv) the agent applies the same relative force in each part of the motor sequence; (v) the actions always occur in the same sequence. In the numerical simulations, we showed that the force, and hence speed, of movement can be controlled by varying the firing rate of the relevant movement selector cells.

The models of motor function developed here are based on self-organizing CANNs. Continuous attractor networks are able to stably maintain a localised packet of neuronal firing activity. This enables neural activity to progress through motor networks at arbitrary speeds, and so allows an agent to control the speed of its movement. A further feature of the models presented here is their use of trace learning rules which incorporate a form of temporal average of recent cell activity. Such rules are able to build associations between different patterns of neural activities that tend to occur in temporal proximity. This form of temporal learning underlies the ability of the networks to learn temporal sequences of behaviour.

The model developed here is capable of implementing various types of motor programs. For example, the model may be used to learn movement sequences which involve the agent seeking to reach a fixed positional target, as in prehension, where the target position is specified by the particular pattern of firing among the movement selector cells. Alternatively, the motor sequence may not involve a particular target state for the agent. Instead the performance of the motor sequence itself may be the goal as in the case of, for example, saying a word. In addition, the network can implement cyclic motor programs, involving continuous cycling through the position state space of the agent, if the state space associated with the movement is toroidal (as might be required for example to implement locomotion).

A key property of the model is that the network is able to learn both forward and inverse models of motor control. The forward model, implemented by the synapses w^2_{ijk}, allows much quicker updating of the representation in the state network than would be possible by visual or proprioceptive cues alone. In particular, in the simulation results we showed that the forward model is able to take into account the force with which the motor sequence is executed, and hence the speed of movement of the agent, when updating the representation within the state network. Such a property

[11] Although this part of Schmidt's hypothesis does receive considerable experimental support (Carter & Shapiro, 1984; Kelso, Putnam, & Goodman, 1983; Schmidt, 1987; Shapiro & Schmidt, 1982; Terzuolo & Viviani, 1980), some studies have found that the proportion of time spent in each part of a motor program is not always maintained over different speeds (Gentner, 1987). Further, for some kinds of multi-step motor programs, which consist of a series of distinct stages, the proportional duration model may apply to each component of a movement, but not necessarily to an entire movement sequence (Shapiro, 1976).

is of fundamental importance for the practical operation of a forward model. The inverse model is implemented by the synapses w^3_{ijk}. Given a desired motor task or target state represented by the firing of the movement selector cells, the synaptic connections w^3_{ijk} drive the motor cells to perform the appropriate motor actions. We note that the model addresses the issue of how the automatic motor system learns, and does not itself use feedback control. However, the other part of the motor system, shown as providing the motor training signal t in Fig. 1, is presumably—though this is not part of the model we describe—able to use visual and proprioceptive feedback in a classical type of control loop. To the extent that this occurs, the model of the learning of automatic skills described here could be part of a larger system which does utilise error feedback control from, e.g. visual and proprioceptive signals during training. We emphasise that the subsystem we describe does not actually use feedback control itself as part of its processing—any feedback control effects have the role of affecting the motor training signal t, which is an input to the model we describe.

A key property of both biological and artificial control systems is robustness. We propose that it should be possible for the model described in this paper to learn to perform a motor program robustly in the sense that the synaptic weights are self-organized such that, during testing small perturbations from the ideal motor sequence will result in the network being drawn back to the correct sequence. That is, for example, in the case of a cyclic motor sequence, the whole motor sequence becomes an asymptotically stable limit cycle, forming a basin of attraction for all nearby perturbations from the ideal motor sequence. This might be achieved through learning with an explicit reward signal, where the reward signal r^{R} is set to 1 whenever the agent has moved from a perturbed state back onto the correct motor sequence.

In the simulations performed in this paper, the state space of the agent, represented by the network of state cells, is one-dimensional. However, in principle, the network can implement movements in higher dimensional spaces, because the nature of the position state space is stored in the recurrent synaptic connections w^1_{ij}, which may become self-organised during learning to represent spaces of arbitrary dimensionality (Stringer et al., 2002b,c).

The model developed in this paper offers an interesting perspective on the rôle of visual feedback in guiding motor sequences like prehension, where the movement selector cells represent a target position for the agent. We note the following points. When visual input is available, the state cell representation stays accurate. However, the model can operate without visual or proprioceptive input. In this case, the network relies on the representation of the agent's state within the continuous attractor network of state cells, which is updated by efference copy from the motor network. This property of the model is in accordance with evidence that animals (Bizzi & Polit, 1979; Polit & Bizzi, 1978) and humans (Laszlo, 1966, 1967) are able to perform some tasks such as reaching in the absence of visual or proprioceptive feedback, although in more ecological situations where several joints are involved (Rothwell et al., 1982) or where there are several segments to a movement (Cordo, 1990; Cordo & Flanders, 1990; Sainburg, Ghilardi, Poizner, & Ghez, 1995), some impairment is more typical. In the model, impairments in accurate performance can be accounted for by path integration errors in the update of the state network when only efference copy is available. This means that the motor cells will receive inaccurate information from the state cells, which will lead to errors in the motor sequence executed, and hence a loss of accuracy in reaching the target position. However, if visual cues are made available during the course of a movement then the state network may be updated to represent the current position of the agent more accurately. This will then allow the correct motor sequence to be performed by the agent, allowing the agent to reach the target position more accurately. However, this mechanism produces a more accurate movement to the target position as a result of a more accurate state representation rather than the availability of a measure of error between the current state of the agent and its target position. The earlier discussion perhaps suggests a new view of how visual feedback is used in motor tasks like prehension. Rather than directly using an explicit error signal representing the distance between the hand and the target, the accuracy with which the agent reaches a target position may depend instead on the accuracy of its state representation, which is then used to drive the correct motor sequence. That is, the role of visual cues may be to ensure an accurate representation in the state network, which in turn is used to stimulate the correct dynamic in the motor network given the current target position represented by the movement selector cells. This viewpoint may become important in the future interpretation of data from neurophysiological studies of motor function.

The networks described in this paper are intended primarily to introduce and illustrate a new approach to understanding how motor function might be implemented in neural systems. However, it is of interest to consider which brain areas might correspond to some of the parts of the network described here. One relevant brain area is the parietal cortex, which has a set of representations of the body and of egocentric visual space (Colby, 1999). Further, damage to the parietal cortex can result in impairments in both spatial perception and related forms of motor action (Milner & Goodale, 1995). In relation to the state representation described in this paper, it is relevant that neurons in the anterior intraparietal area (AIP) respond not only in relation to the shape of a grasp that is required, but also, in a memory guided reaching task, when the monkey is remembering an object with the neuron's preferred object shape (Murata, Gallese, Kaseda, & Sakata, 1996). Further, if area AIP is inactivated, this reduces the monkey's ability to properly control the shape its of its hand in order to grasp particular objects (Gallese, Murata, Kaseda, Niki, & Sakata, 1994). Thus, the spatial representation maintained in area AIP appears to be for the purpose of guiding grasping movements with the hand, and is used by the premotor cortex to control hand shape and grip (Gallese, Fadiga, Fogassi, Luppino, & Murata, 1997; Jeannerod, Arbib, Rizzolatti, & Sakata, 1995). Another brain region with a state representation is the hippocampus, and in this case the representation in primates is of visual space. In particular, primate spatial view cells respond when the monkey is looking towards a particular location in space (Georges-François, Rolls, & Robertson, 1999; Robertson, Rolls, & Georges-François, 1998; Rolls, Robertson, & Georges-François, 1997), and are updated by idiothetic cues such as eye movements made in darkness (Robertson et al., 1998), thus reflecting path integration. Moreover, hippocampal spatial representations are important in movement, as shown by the deterioration in the ability of the rat to navigate through its environment after hippocampal damage (Morris, Schenk, Tweedie, & Jarrard, 1990).

The training signals t^{MS} used to guide the firing of movement selector cells determine the command patterns represented in the movement selector layer, and hence the overlap and interference between different movement commands/motor programs. If two separate movement commands have significantly overlapping distributed representations in the layer of movement selector cells, then executing one motor program may stimulate some motor neurons associated with the other program. There are a number of ways in which this problem may be ameliorated. First, the firing threshold of the motor neurons may be set sufficiently high for these cells to ignore partial signals from incomplete movement commands in the layer of movement selector cells. Second, the training signal t^{MS} may operate to establish a continuous space of commands within the movement selector cell network, as demonstrated in Experiment 4, in which case there will be no interference between the commands represented in that network. Third, the use of even higher order synapses w^3 may help to make the motor cells respond more selectively to movement commands represented in the layer of movement selector cells.

In Section 4 we discussed various mechanisms that might help to remove the effects of errors present in the motor sequence during training. First, in numerical simulations we demonstrated that one way of reducing the effects of motor error during learning might be through explicit learning, which involves making use of an explicit reward signal r^{R} that signifies when the motor sequence is being performed well by the agent. However, questions remain about how the agent might judge when the motor sequence was being performed well, and how this might be converted into some sort of useful reward signal in the brain. In further simulations we demonstrated that a second way in which the effects of motor error during learning may be reduced is through implicit learning, which relies on the noisy motor training signal containing a systematic component of the correct signal which is relatively strong compared to any error components. However, implicit learning relies on the presence of a relatively strong component of the motor training signal relating to the desired motor sequence. The question remains as to how the agent may generate such a motor training signal in the first place during learning. For both of the above learning mechanisms, a fundamental requirement is that during learning the agent is able to generate a motor training signal t_i that is good enough, at least for part of the time, for the motor cells to be able to extract and learn the correct motor sequence. The details of how this might be achieved for the learning mechanisms described earlier is an important question for future research.

In conclusion, we have used a dynamical systems perspective to show how complex motor sequences could be learned. The network uses a continuous attractor network architecture to perform path integration on an efference copy of the motor signal to keep track of the current state, and selection of which motor cells to activate by a movement selector input where the selection depends on the current state being represented in the continuous attractor network.

Acknowledgements

This research was supported by the Medical Research Council, grant PG9826105, by the Human Frontier Science Program, and by the MRC Interdisciplinary Research Centre for Cognitive Neuroscience.

References

Amari, S. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, *27*, 77–87.

Bizzi, E., & Polit, A. (1979). Processes controlling visually evoked movements. *Neuropsychologia*, *17*, 203–213.

Carter, M. C., & Shapiro, D. C. (1984). Control of sequential movements: evidence for generalized motor programs. *Journal of Neurophysiology*, *52*, 787–796.

Colby, C. L. (1999). Parietal cortex constructs action-orientated spatial representations. In N. Burgess, K. J. Jeffery, & J. O'Keefe (Eds.), *The hippocampal and parietal foundations of spatial cognition* (pp. 104–126). Oxford: Oxford University Press.

Cordo, P. J. (1990). Kinesthetic control of a multijoint movement sequence. *Journal of Neurophysiology*, *63*(1), 161–172.

Cordo, P. J., & Flanders, M. (1990). Time-dependent effects of kinesthetic input. *Journal of Motor Behaviour*, *22*(1), 45–65.

Fitts, P. M., & Posner, M. I. (1967). *Human performance*. Belmont, CA: Brooks/Cole.

Gallese, V., Fadiga, L., Fogassi, L., Luppino, G., & Murata, A. (1997). A parietofrontal circuit for hand grasping movements in the monkey: Evidence from reversible inactivation experiments. In P. Thier, & H. O. Karnath (Eds.), *Parietal lobe contributions to orientation in 3D space* (pp. 619–631). Heidelberg: Springer.

Gallese, V., Murata, A., Kaseda, M., Niki, N., & Sakata, H. (1994). Deficit of hand preshaping after muscimol injection in monkey parietal cortex. *Neuroreport*, *5*, 1525–1529.

Gentile, A. M. (1972). A working model of skill acquisition with application to teaching. *Quest, Monograph*, *17*, 3–23.

Gentile, A. M. (1987). Skill acquisition: Action, movement, and the neuromotor processes. In J. H. Carr, R. B. Shepherd, J. Gordon, A. M. Gentile, & J. M. Hind (Eds.), *Movement science: Foundations for physical therapy in rehabilitation*. Rockville, MD: Aspen.

Gentner, D. R. (1987). Timing of skilled motor performance: tests of the proportional duration model. *Psychological Review*, *94*, 255–276.

Georges-François, P., Rolls, E. T., & Robertson, R. G. (1999). Spaital view cells in the primate hippocampus: allocentric view not head direction or eye position or place. *Cerebral Cortex*, *9*, 197–212.

Jeannerod, M., Arbib, M. A., Rizzolatti, G., & Sakata, H. (1995). Grasping objects: the cortical mechanisms of visuomotor transformation. *Trends in Neuroscience*, *18*, 314–320.

Kelso, J. A. S., Putnam, C. A., & Goodman, D. (1983). On the spacetime structure of human interlimb co-ordination. *Quarterly Journal of Experimental Psychology*, *35A*, 347–375.

Koch, C. (1999). *Biophysics of computation*. Oxford: Oxford University Press.

Laszlo, J. L. (1966). The performance of a single motor task with kinesthetic sense loss. *Quarterly Journal of Experimental Psychology*, *18*, 1–8.

Laszlo, J. L. (1967). Training of fast tapping with reduction of kinesthetic, tactile, visual, and auditory sensation. *Quarterly Journal of Experimental Psychology*, *19*, 344–349.

Lisman, J. E., Fellous, J. M., & Wang, X. J. (1998). A role for NMDA-receptor channels in working memory. *Nature Neuroscience*, *1*, 273–275.

Magill, R. A. (1998). *Motor learning*. Boston: McGraw-Hill.

Miall, R. C., & Wolpert, D. M. (1996). Forward models for physiological motor control. *Neural Networks*, *9*, 1265–1279.

Milner, A. D., & Goodale, M. A. (1995). *The visual brain in action*. Oxford: Oxford University Press.

Morris, R. G. M., Schenk, F., Tweedie, F., & Jarrard, L. E. (1990). Ibotenate lesions of hippocampus and/or subiculum dissociating components of allocentric spatial learning. *European Journal of Neuroscience*, *2*, 1016–1028.

Murata, A., Gallese, V., Kaseda, M., & Sakata, H. (1996). Parietal neurons related to memory-guided hand manipulation. *Journal of Neurophysiology*, *75*, 2180–2186.

Newell, K. M. (1985). Coordination, control and skill. In D. Goodman, R. B. Wilberg, & I. M. Franks (Eds.), *Differing perspectives in motor learning, memory and control* (pp. 295–317). Amsterdam: North-Holland.

Pfeifer, R., & Scheier, C. (1999). *Understanding intelligence*, Cambridge, MA: MIT Press.

Polit, A., & Bizzi, E. (1978). Processes controlling arm movements in monkeys. *Science*, *201*, 1235–1237.

Robertson, R. G., Rolls, E. T., & Georges-François, P. (1998). Spatial view cells in the primate hippocampus: effects of removal of view details. *Journal of Neurophysiology*, *79*, 1145–1156.

Rolls, E. T., Robertson, R. G., & Georges-François, P. (1997). Spatial view cells in the primate hippocampus. *European Journal of Neuroscience*, *9*, 1789–1794.

Rothwell, J. C., Traub, M. M., Day, B. L., Obeso, J. A., Thomas, P. K., & Marsden, C. D. (1982). Manual motor performance in a deafferented man. *Brain*, *105*, 515–542.

Sainburg, R. L., Ghilardi, M. F., Poizner, H., & Ghez, C. (1995). Control of limb dynamics in normal subjects and patients without proprioception. *Journal of Neurophysiology*, *73*, 820–835.

Schmidt, R. A. (1987). *Motor control and learning: A behavioural emphasis* (2nd ed.). *Human kinetics*, IL: Champaign.

Schmidt, R. A. (1988). Motor and action perspectives on motor behaviour. In O. G. Meijer, & K. Roth (Eds.), *Complex motor behaviour: The motor-action controversy* (pp. 3–44). Amsterdam: Elsevier.

Shapiro, D. C. (1976). A preliminary attempt to determine the duration of a motor program. In D. M. Landers, & R. W. Christina (Eds.), *Psychology of sport and motor behaviour* (*Vol. 1*) (pp. 17–24). *Human kinetics*, IL: Champaign.

Shapiro, D. C., & Schmidt, R. A. (1982). The schema theory: Recent evidence and developmental implications. In J. A. S. Kelso, & J. E. Clark (Eds.), *The development of movement control and coordination* (pp. 113–150). New York: Wiley.

Stringer, S. M., & Rolls, E. T (2003). Hierarchical dynamical models of motor function in press.

Stringer, S. M., Rolls, E. T., & Trappenberg, T. P (2003). Self-organizing continuous attractor network models of hippocampal spatial view cells in press.

Stringer, S. M., Rolls, E. T., Trappenberg, T. P., & de Araujo, I. E. T. (2002b). Self-organizing continous attractor networks and path integration: two-dimensional models of place cells. *Network: Computation in Neural Systems*, *13*, 429–446.

Stringer, S. M., Trappenberg, T. P., Rolls, E. T., & de Araujo, I. E. T. (2002c). Self-organizing continuous attractor networks and path integration: one-dimensional models of head direction cells. *Network: Computation in Neural Systems*, *13*, 217–242.

Sutton, R. S., & Barto, A. G. (1981). Towards a modern theory of adaptive networks: expectation and prediction. *Psychological Review*, *8*, 135–170.

Taylor, J. G. (1999). Neural 'bubble' dynamics in two dimensions: foundations. *Biological Cybernetics*, *80*, 393–409.

Terzuolo, C. A., & Viviani, P. (1980). Determinants and characteristics of motor patterns used for typing. *Neuroscience*, *5*, 1085–1103.

Wang, X. J. (1999). Synaptic basis of cortical persistent activity: the importance of NMDA receptors to working memory. *Journal of Neuroscience*, *19*, 9587–9603.

Wolpert, D. M., & Ghahramani, Z. (2000). Computational principles of movement neuroscience. *Nature Neuroscience*, *3*(suppl.), 1212–1217.

Wolpert, D. M., Miall, R. C., & Kawato, M. (1998). Internal models in the cerebellum. *Trends in Cognitive Sciences*, *2*, 338–347.