Movement Generation and Control with Generic Neural Microcircuits*

Prashant Joshi and Wolfgang Maass

Institute for Theoretical Computer Science
Technische Universitaet Graz
A-8010 Graz, Austria
{joshi,maass}@igi.tugraz.at

Abstract. Simple linear readouts from generic neural microcircuit models can be trained to generate and control basic movements, e.g., reaching with an arm to various target points. After suitable training of these readouts on a small number of target points; reaching movements to nearby points can also be generated. Sensory or proprioceptive feedback turns out to improve the performance of the neural microcircuit model, if it arrives with a significant delay of 25 to 100 ms. Furthermore, additional feedbacks of "prediction of sensory variables" are shown to improve the performance significantly. Existing control methods in robotics that take the particular dynamics of sensors and actuators into account("embodiment of robot control") are taken one step further with this approach which provides methods for also using the "embodiment of computation", i.e. the inherent dynamics and spatial structure of neural circuits, for the design of robot movement controllers.

1 Introduction

This article demonstrates that simple linear readouts from generic neural microcircuit models consisting of spiking neurons and dynamic synapses can be trained to generate and control rather complex movements. Using biologically realistic neural circuit models to generate and control movements is not so easy, since these models are made of spiking neurons and dynamic synapses which exhibit a rich inherent dynamics on several temporal scales. This tends to be in conflict with movement control tasks that require focusing on a relatively slow time scale.

Preceding work on movement control, has drawn attention to the need of taking the "embodiment of motor systems", i.e. the inherent dynamics of sensors and actuators into account. This approach is taken one step further in this article, as it provides a method for also taking into account the "embodiment of neural computation", i.e. the inherent dynamics and spatial arrangement of neural circuits that control the movements. Hence it may be seen as a first step in

* The work was partially supported by the Austrian Science Fond FWF, project # P15386.

A.J. Ijspeert et al. (Eds.): BioADIT 2004, LNCS 3141, pp. 258–273, 2004.

a long range program where abstract control principles for biological movement control and related models developed for artificial neural networks [Tani, 2003] can be implemented and tested on arbitrarily realistic models for the underlying neural circuitry.

The feasibility of our approach is demonstrated in this article by showing that simple linear readouts from a generic neural microcircuit model can be trained to control a 2-joint robot arm, which is a common benchmark task for testing methods for nonlinear control [Slotine and Li, 1991]. It turns out that both the spatial organization of information streams, especially the spatial encoding of slowly varying input variables, and the inherent dynamics of the generic neural microcircuit model have a significant impact on its capability to control movements. In particular it is shown that the inherent dynamics of neural microcircuits allows these circuits to cope with rather large delays for proprioceptive and sensory feedback. In fact it turns out that their performance is optimal for delays that lie in the range of 25 to 100 ms. Additionally it is shown that the generic neural microcircuit models used by us, possess significant amount of temporal integration capabilities. It is also demonstrated that this new paradigm of motor control provides generalization capabilities to the readouts. Furthermore, it is shown that the same neural microcircuit model can be trained simultaneously to predict the results of such feedbacks, and by using the results of these predicted feedbacks it can improve its performance significantly in cases where feedback arrives with other delays, or not at all.

This work complements preceding work where generic neural microcircuit models were used in an open loop for a variety of simulated sensory processing tasks ([Buonomano and Merzenich, 1995], [Maass et al., 2002], [Maass et al., 2003]). It turns out that the demands on the precision of real-time computations carried out by such circuit models are substantially higher for closed-loop applications such as those considered in this article. Somewhat similar paradigms for neural control based on artificial neural network models have been independently explored by Herbert Jaeger [Jäger, 2002].

The neural microcircuit model and the control tasks considered in this article are specified in the subsequent two sections. Results of computer simulations are presented in sections 4, 5, 6 and relations to theoretical results are discussed in section 7.

2 Generic Neural Microcircuit Models

In contrast to common artificial neural network models, neural microcircuits in biological organisms consist of diverse components such as different types of spiking neurons and dynamic synapses, that are each endowed with an inherently complex dynamics of its own. This makes it difficult to construct out of biologically realistic computational units, implementations of boolean or analog circuits that have turned out to be useful in the context of computer science or artificial neural networks. On the other hand, it opens the path towards alternative computational paradigms based on emergent computations in sparsely and

recurrently connected neural microcircuits, composed of diverse dynamic components [Maass et al., 2002]. In [Maass et al., 2002] a new computational model, the liquid state machine, has been proposed that can be used to explain and analyze the capabilities of such neural microcircuits for real-time computing. Consequences of this analysis for applications to closed loop control are discussed in section 7 of this article. Instead of constructing circuits for specific tasks, one considers here various probability distributions for neural connectivity. Such circuits have inherent capabilities for temporal integration of information from several segments of incoming input streams, and relevant computations that recombine pieces of this information in a nonlinear manner emerge automatically. To be exact, the information about the outputs of a very large class of computations on information contained in the input stream is automatically present in the "liquid state" of the dynamical system in the sense of [Maass et al., 2002], see [Natschläger and Maass, 2004].

The liquid state $\mathbf{x}(t)$ models that part of the current circuit state that is in principle "visible" to a readout neuron (see Fig. 1, a) that receives synaptic inputs from all neurons in the circuit. Each component of $\mathbf{x}(t)$ models the impact that a particular neuron v may have on the membrane potential of a generic readout neuron (see Fig. 2). Thus each spike of neuron v is replaced by a pulse whose amplitude decays exponentially with a time constant of 30 ms. In other words: $\mathbf{x}(t)$ is obtained by applying a low-pass filter to the spike trains emitted by the neurons in the generic neural microcircuit model. We will only consider information that can be extracted from the liquid state $\mathbf{x}(t)$ of the generic neural microcircuit model by a simple weighted sum[1] $\mathbf{w} \times \mathbf{x}(t)$. The weight vector $\mathbf{w}$ will be fixed for all t and all circuit inputs once the training of the (symbolic) readout neurons has been completed.

In principle one can of course also view various parameters within the circuit as being subject to learning or adaptation, for example in order to optimize the dynamics of the circuit for a particular range of control tasks. However this has turned out to be not necessary for the applications described in this article. One advantage of just viewing the weight vector $\mathbf{w}$ as being plastic is that learning is quite simple and robust, since it just amounts to linear regression – in spite of the highly nonlinear nature of the control tasks to which this set-up is applied. Another advantage is that the same neural microcircuit could potentially be used for various other information processing tasks (e.g. prediction of sensory feedback, see section 6) that may be desirable for the same or other tasks.

The generic microcircuit models used for the closed loop control tasks described in this article were similar in structure to those that were earlier used for various sensory processing tasks. More precisely, we considered circuits consisting of 600 leaky-integrate-and-fire neurons arranged on the grid points of a $20 \times 5 \times 6$ cube in 3D (see Fig. 1, b). 20 % of these neurons were randomly chosen to be inhibitory. Synaptic connections were chosen according to a probability

[1] One constant component is added to $\mathbf{x}(t)$ to facilitate the implementation of a constant bias in terms of the form $\mathbf{w} \times \mathbf{x}(t)$.

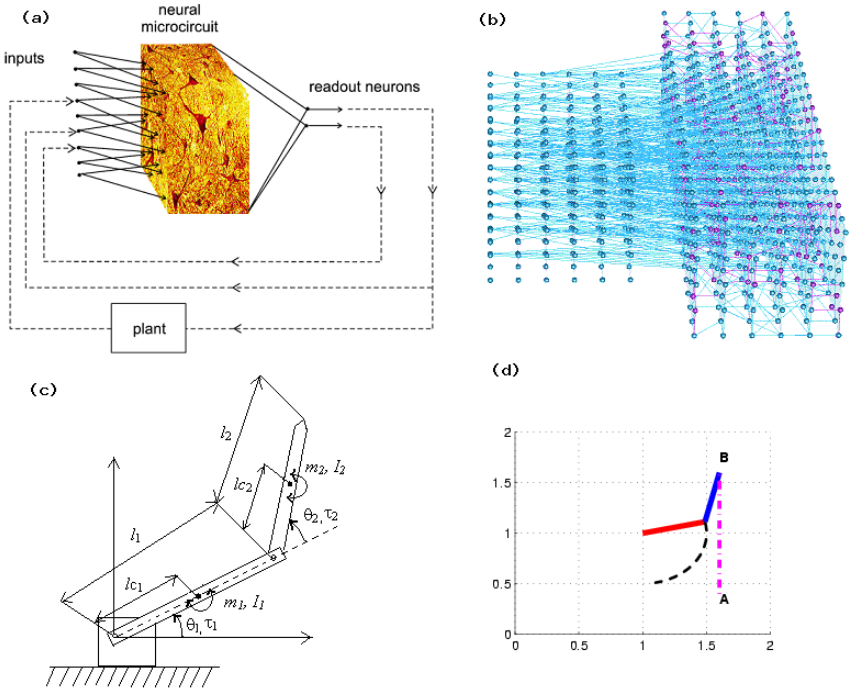

Fig. 1. a) Information flow diagram for a neural microcircuit model applied to a control task. The "plant" may be a motor system or some part of the environment. **b)** Spatial layout of neurons in the models considered in this article. The 6 layers on the left hand side are used for spatial coding of inputs to the circuit $(x_{dest}, y_{dest}, \theta_1(t-\Delta), \theta_2(t-\Delta), \tau_1(t), \tau_2(t))$. Connections between these 6 input layers, as well as between neurons in the subsequent 6 processing layers are chosen randomly according to a probability distribution discussed in the text. **c)** Standard model of a 2-joint robot arm **d)** Initial position A and end position B of the robot arm for one of the movements. The target trajectory of the tip of the robot arm and of the elbow are indicated by dashed lines.

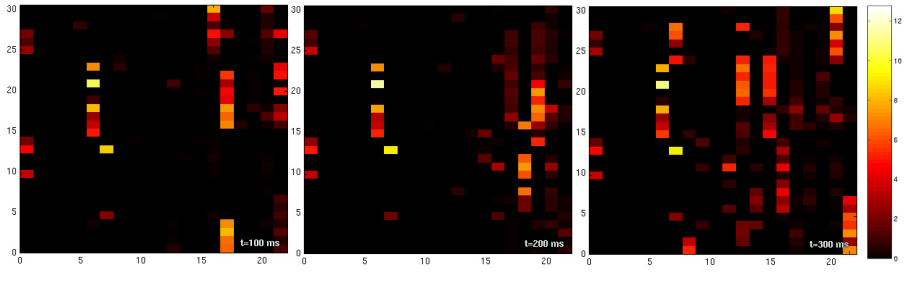

Fig. 2. Snapshots of the liquid state of a neural microcircuit model at 3 different time points. The circuit has 600 neurons which are shown in a 30×20 grid. Snapshots are taken at 100, 200 and 300 ms.

distribution that favored local connections[2]. Parameters of neurons and synapses were chosen to fit data from microcircuits in rat somatosensory cortex (based on [Gupta et al., 2000] and [Markram et al., 1998]).[3] We modeled the (short term) dynamics of synapses according to the model proposed in [Markram et al., 1998], with the synaptic parameters U (use), D (time constant for depression), F (time constant for facilitation) randomly chosen from Gaussian distributions that model empirically found data for such connections.[4]

In order to test the noise robustness of movement generation by the neural microcircuit model the initial condition of the circuit was randomly drawn (initial membrane potential for each neuron drawn uniformly from the interval [13.5 mV, 14.9 mV], where 15 mV was the firing threshold). In addition a substantial amount of noise was added to the input current of each neuron throughout the simulation at each time-step, a new value for the noise input current with mean 0 and SD of 1 nA was drawn for each neuron and added (subtracted) to its input current.

This generic neural microcircuit model received analog input streams from 6 sources (from 8 sources in the experiment with internal predictions discussed in Fig. 6, b and 7). The outcomes of the experiments discussed in this article

[2] The probability of a synaptic connection from neuron a to neuron b (as well as that of a synaptic connection from neuron b to neuron a) was defined as $C \cdot \exp(-D^2(a,b)/\lambda^2)$, where $D(a,b)$ is the Euclidean distance between neurons a and b and λ is a parameter which controls both the average number of connections and the average distance between neurons that are synaptically connected (we set $\lambda = 1.2$). Depending on whether the pre- or postsynaptic neuron were excitatory (E) or inhibitory (I), the value of C was set according to [Gupta et al., 2000] to 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II).

[3] *Neuron parameters*: membrane time constant 30 ms, absolute refractory period 3 ms (excitatory neurons), 2 ms (inhibitory neurons), threshold 15 mV (for a resting membrane potential assumed to be 0), reset voltage drawn uniformly from the interval [13.8, 14.5 mV] for each neuron, constant non-specific background current I_b uniformly drawn from the interval [13.5 nA, 14.5 nA] for each neuron, noise at each time-step I_{noise} drawn from a gaussian distribution with mean 0 and SD of 1nA, input resistance 1 MΩ. For each simulation, the initial conditions of each I&F neuron, i.e., the membrane voltage at time $t = 0$, were drawn randomly (uniform distribution) from the interval [13.5 mV, 14.9 mV].

[4] Depending on whether a and b were excitatory (E) or inhibitory (I), the mean values of these three parameters (with D,F expressed in seconds, s) were chosen according to [Gupta et al., 2000] to be .5, 1.1, .05 (EE), .05, .125, 1.2 (EI), .25, .7, .02 (IE), .32, .144, .06 (II). The SD of each parameter was chosen to be 50% of its mean. The mean of the scaling parameter A (in nA) was chosen to be 70 (EE), 150 (EI), -47 (IE), -47 (II). In the case of input synapses the parameter A had a value of 70 nA if projecting onto a excitatory neuron and -47 nA if projecting onto an inhibitory neuron. The SD of the A parameter was chosen to be 70% of its mean and was drawn from a gamma distribution. The postsynaptic current was modeled as an exponential decay $\exp(-t/\tau_s)$ with $\tau_s = 3$ ms ($\tau_s = 6$ ms) for excitatory (inhibitory) synapses. The transmission delays between liquid neurons were chosen uniformly to be 1.5 ms (EE), and 0.8 ms for the other connections.

were all negative if these analog input streams were directly fed into the circuit (as input current for selected neurons in the circuit). Apparently the variance of the resulting spike trains were too large to make the information about the slowly varying values of these input streams readily accessible to the circuit. Therefore the input streams were instead fed into the circuit with a simple form of spatial coding, where the location of neurons that were activated encoded the current values of the input variables. More precisely, each input variable is first scaled into the range $[0, 1]$. This range is linearly mapped onto an array of 50 symbolic input neurons. At each time step, one of these 50 neurons, whose number $n(t) \in \{1, \ldots, 50\}$ reflects the current value $i_n(t) \in [0, 1]$ which is the normalized value of input variable $i(t)$ (e.g. $n(t) = 1$ if $i_n(t) = 0$, $n(t) = 50$ if $i_n(t) = 1$). The neuron $n(t)$ then outputs at time step t, the value of $i(t)$. In addition the 3 closest neighbors on both sides of neuron $n(t)$ in this linear array get activated at time t by a scaled down amount according to a gaussian function (the neuron number n outputs at time step t the value $i(t)\ \frac{1}{\sigma\sqrt{2\pi}}\ e^{\frac{-(n-n(t))^2}{2\sigma^2}}$, where $\sigma = 0.8$). Thus the value of each of the 6 input variables is encoded at any time by the output values of the associated 50 symbolic input neurons (of which at least 43 neurons output at any time the value 0). The neuron in each of these 6 linear arrays are connected with one of the 6 layers consisting of 100 neurons in the previously described circuit of 100×6 ($(20 \times 5) \times 6$) integrate-and-fire neurons.[5] The spatial arrangement of the 6 input pools can be seen on the left of the 6 layers of the circuit of spiking neurons in Fig. 1, b.

3 A 2-Joint Robot Arm as a Benchmark Nonlinear Control Task

We used the generic neural microcircuit models in a closed loop (see Fig. 1, a and Fig. 3) as controllers for a 2-joint robot arm. More precisely, we trained them to control exactly the same model for a 2-joint robot arm (see Fig. 1, c) that is used in [Slotine and Li, 1991] as a standard reference model for a complex nonlinear control task (see in particular ch. 6 and 9). It is assumed that the arm is moving in a horizontal plane, so that gravitational forces can be ignored.

Using the well-known Lagrangian equation in classical dynamics, the dynamic equations for our arm model are given by equation 1:

$$\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -h\dot{\theta}_2 & -h(\dot{\theta}_1 + \dot{\theta}_2) \\ h\dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{1}$$

with $\boldsymbol{\theta} = [\theta_1\ \theta_2]^T$ being the two joint angles, $\boldsymbol{\tau} = [\tau_1\ \tau_2]^T$ being the joint input torques, and

$$H_{11} = m_1 {l_{c_1}}^2 + I_1 + m_2[{l_1}^2 + {l_{c_2}}^2 + 2\ l_1 l_{c_2} \cos\theta_2] + I_2$$

[5] with a value of 3.3 for λ in the formula for the connection probability given in footnote 2.

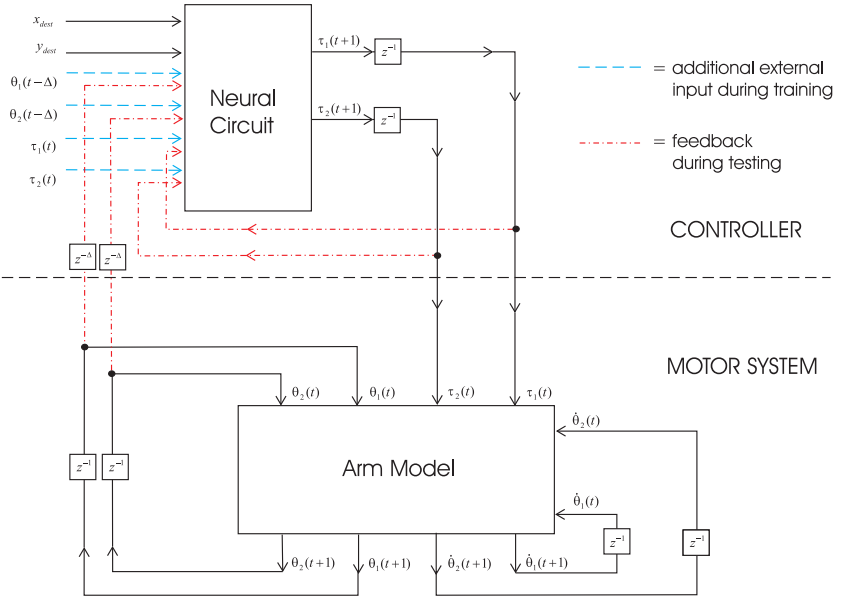

Fig. 3. During training the neural circuit receives the values $\theta_1(t-\Delta), \theta_2(t-\Delta), \tau_1(t), \tau_2(t)$ as additional external inputs. During validation the circuit is run in a closed loop with the arm model, and these 4 external inputs are replaced by internal feedbacks as indicated. The only external inputs during validation are the constant inputs x_{dest}, y_{dest} that give in cartesian coordinates, the target position of the tip of the robot arm after the movement has been completed. All the dynamics needed to generate the movement is then provided by the inherent dynamics of the neural circuit in response to the switching on of these constant inputs (and in response to the dynamics of the feedbacks).

$$H_{12} = H_{21} = m_2 l_1 l_{c_2} \cos\theta_2 + m_2 {l_{c_2}}^2 + I_2$$

$$H_{22} = m_2 {l_{c_2}}^2 + I_2$$

$$h = m_2 l_1 l_{c_2} \sin\theta_2$$

Equation 1 can be compactly represented as:

$$H(\theta)\,\ddot{\boldsymbol{\theta}} + C(\theta, \dot{\theta})\,\dot{\boldsymbol{\theta}} = \boldsymbol{\tau}$$

where H represents the inertia matrix, and C represents the matrix of Coriolis and centripetal terms. The values of the parameters that were used in our simulations were: $m_1 = 1, m_2 = 1, l_{c_1} = 0.25, l_{c_2} = 0.25, I_1 = 0.03, I_2 = 0.03$.

The closed loop control system that was used to generate the results discussed in this article is shown in Fig. 3. During training of the readouts from the generic neural microcircuit model the circuit was used in an open loop with target values for the output torques provided by equation 1 (for a given target trajectory $\{\theta_1(t), \theta_2(t)\}$), and feedbacks from the plant replaced by the target values of

these feedbacks for the target trajectory. More precisely, for each such target trajectory 20 variations of the training samples were generated, where for each time step t[6], a different noise value of $0.01 \times \rho$ was added to each of the input channels where ρ is a random number drawn from a gaussian distribution with mean 0 and SD 1, multiplied by the current value of input channel.

The purpose of this extended training procedure was to make the readout robust with regard to deviation from the target trajectory caused by imprecision in earlier torque outputs (see section 7). Each target trajectory had a time duration of 500 ms.

4 Teaching a Generic Neural Microcircuit Model to Generate Basic Movements

As a first task the generic neural microcircuit model described in section 2 was taught to generate with the 2-joint arm described in section 3 , 4 separate movements. One of these movements is shown in Fig. 1, d. In each case the task was to move the tip of the arm from point A to point B on a straight line, with a biologically realistic bell-shaped velocity profile.The 2 readouts of the neural microcircuit model are trained by linear regression to output the joint torques required for movement. More precisely, 20 noisy variations of each of the 4 target movements were used for the training of the two readouts by linear regression. Note that each readout is simply modeled as a linear gate with weight vector $\mathbf{w}$ applied to the liquid state $\mathbf{x}(t)$ of the neural circuit. This weight vector is fixed after training, and during validation all 4 movements are generated with this fixed weight vector at the readouts.

When the circuit receives as input the coordinates $\langle x_{dest}, y_{dest} \rangle$ of the endpoint B of one of the target movements, the circuit autonomously generates in a closed loop the torques needed to move the 2-joint arm from the corresponding initial point A to this endpoint B.

Obviously temporal integration capabilities of the controller are needed for the control of many types of movements. Generic neural microcircuit models have inherent temporal integration capabilities (see [Maass et al., 2003]), and hence can even generate movements that require to stop for a certain period, and then to move on autonomously. Fig. 4 shows the results for the case where the readouts from the neural microcircuit have been trained to generate a stop-and-start kind of motion, with a stop from 225 to 275 ms (see the velocity profile at the bottom). The initiation of the continuation of the movement at time $t =$ 275 ms takes place without any external cue, just on the basis of the inherent temporal integration capability of the trained neural circuit. Average deviation of the tip of the robot arm from the target end point over 20 validation runs: 6.86 cm. For the sake of demonstration purposes we chose for the experiment reported in Fig. 4 a feedback delay of just 1 ms, so that all circuit inputs are

[6] all time steps were chosen to have a length of 2 ms, except for the experiment reported in Fig. 4, where a step size of 1 ms was used.

constant during 49 ms of the 50 ms while the controller has to wait, forcing the readouts to recognize just on the basis of the inherent circuit dynamics when to move on.

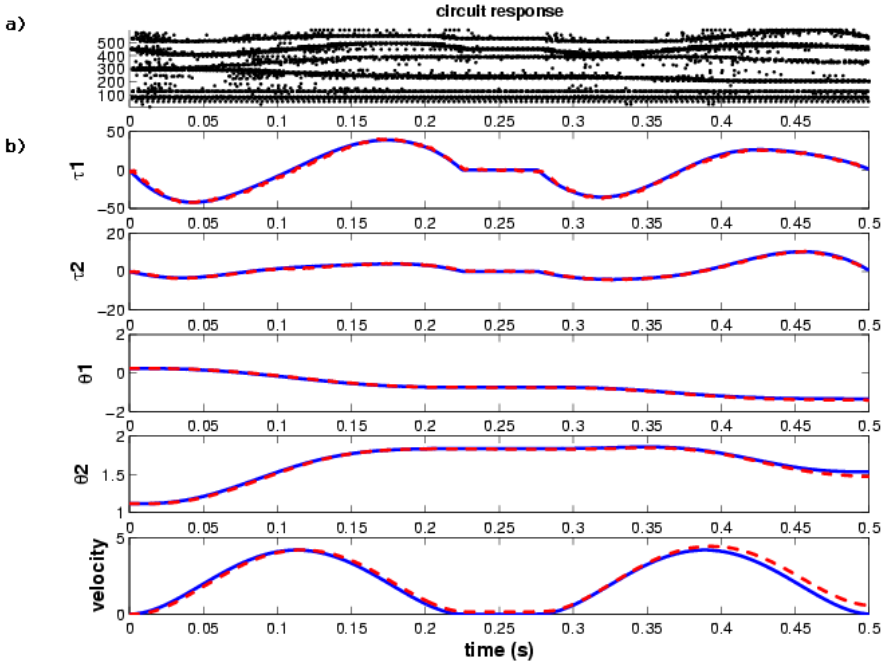

Fig. 4. Demonstration of the temporal integration capability of the neural controller. Shown is a validation run for a circuit that has been trained to generate a movement that requires an intermediate stop and then autonomous continuation of the movement after 50 ms. Target trajectories are shown as solid lines, actual trajectories as dashed lines.

5 Generalization Capabilities

For the experiment reported in Fig. 5, the circuit was trained to generate from a common initial position reaching movements to 8 different target positions, given in terms of their cartesian coordinates as constant inputs $\langle x_{dest}, y_{dest} \rangle$ to the circuit. After training the circuit was able to generate with fairly high precision reaching movements to other target points never used during training. The autonomously generated reaching movements moved the robot arm on a rather straight line with bell-shaped velocity profile, just as for those reaching movements to targets that were used for training.

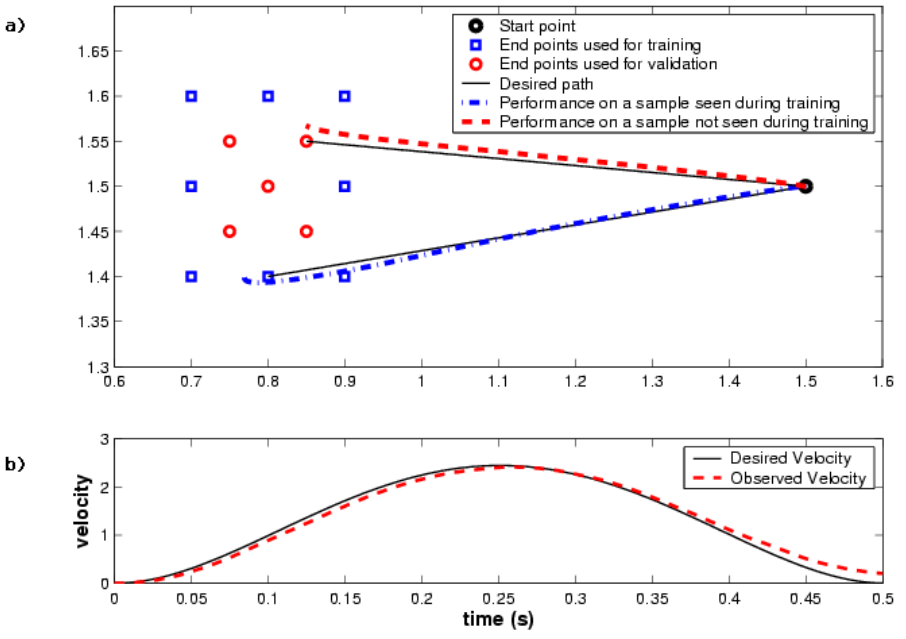

Fig. 5. a) Generalization of movement generation to 5 target end points (small circles) which were not among the 8 target end points that occurred during training (small squares). Movements to a new target end point was initiated by giving its cartesian coordinates as inputs to the circuit. Average deviation for 15 runs with new target end points: 10.3 cm (4.8 cm for target end points that occurred during training). **b)** The velocity profile for one of the generalized movements (target - solid, actual - dashed).

6 On the Role of Feedback and Sensory Prediction for Movement Generation

Our model assumes that the neural circuit receives as inputs in addition to efferent copies of its movement commands (i.e. torques $u_1(t), u_2(t)$), with little delay also simulated proprioceptive or visual feedback that provides at time t, information about the actual values of the angles of the joints at time $t - \Delta$. Whereas it is quite difficult to construct circuits or other artificial controllers that can benefit significantly from substantially delayed feedbacks, we show in Fig. 6 a, that generic neural microcircuit models are able to generate and control movements even for substantially delayed feedbacks. In fact, Fig. 6 a, shows that the best performance is achieved not when this delay Δ has an value of 0, but for a range of delays between 25 and 100 ms. In another experiment which is reported in Fig. 6 b, we used a generic neural microcircuit with 800 neurons ($20 \times 5 \times 8$). This microcircuit had two additional readouts which were trained to estimate at time t, the values of the joint angles θ_1 and θ_2 at time t - 200 ms.

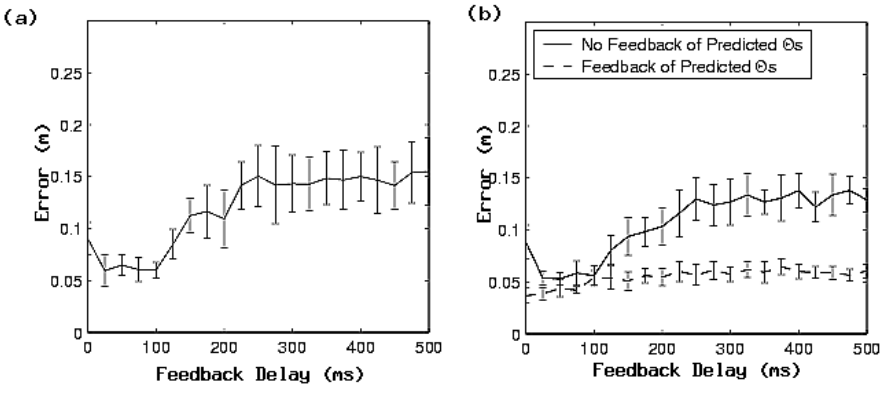

Fig. 6. Deviation of end point of movement averaged over 400 runs (10 randomly constructed circuits were used; 40 runs per circuit), for the same 4 movements but with 21 different values of the delay Δ for the feedback $\theta_1(t-\Delta), \theta_2(t-\Delta)$. The vertical bars show the SD of the data. **a)** Circuit-size – 600 neurons. No readouts assigned to predict the feedbacks. **b)** Circuit-size – 800 neurons. Two additional readouts have been trained to predict $\theta_1(t-200$ ms$), \theta_2(t-200$ ms$)$. The solid line shows the average performance when these predictions are not fed back to the circuit. The dashed line shows the average performance when these predictions were also supplied to the circuit. Note that the SD in the data when these predictions were fed-back to the circuit is considerably less than that of the case when no such predictions were availaible.

The top solid line in Fig. 6 b shows the result (computed in the same way as Fig. 6 a) for the case when the information about these estimates was not fed-back to the circuit. The bottom dashed line shows the result when these estimates were available to the circuit via feedback. Although these additional feedbacks do not provide any new information to the circuit, but only condensate and reroute (see Fig. 7) information, that was previously spread out all over the circuit dynamics; this additional feedback significantly improved the performance of the simulated neural microcircuit for all values of Δ. The values for $\Delta = 500$ ms shows the improvement achieved by using predicted sensory feedback in case when no feedback arrives at all, since the total movements lasted for 500 ms.

7 Theoretical Analysis

Some theoretical background for the analysis of the computational power of neural microcircuits in an open loop, such as online speech recognition (see [Maass et al., 2002]), is provided by two mathematical theorems in the appendix of [Maass et al., 2002] (see [Maass and Markram, 2002] for details). The main result is that a sufficiently large neural microcircuit model (which contains sufficiently diverse dynamic components to satisfy the separation property) can in principle uniformly approximate any given time-invariant fading memory filter F. However these theoretical results do not address the problems caused by inter-

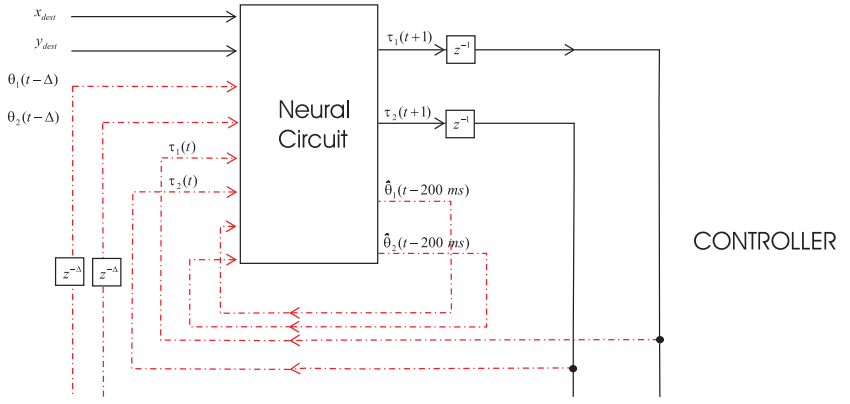

Fig. 7. Information flow for the case of autonomously generated estimates $\hat{\theta}(t-200$ ms) of delayed feedback $\theta(t-200$ ms). Rest of the circuit as in Fig. 3.

nal noise in neural microcircuits, which require a more quantitative complexity-theoretic analysis that takes into account that differences in liquid states at a given time t, which are caused by two different input histories $u(\cdot)$ and $v(\cdot)$, have to be *sufficiently large* (instead of just $\neq 0$) in order to be distinguishable from state differences caused by internal noise, and it is only meaningful to require such sufficiently large separation for input histories $u(\cdot), v(\cdot)$ that are *significantly* different (hence the dynamics of the neural microcircuit has to be non-chaotic). Fortunately generic neural microcircuit models tend to have these properties (see [Maass et al., 2002]).

Additional conditions have to be met for successful applications of neural microcircuit models in closed-loop control tasks, such as those considered in this article. First of all, one has to assume that the approximation target for the neural microcircuit, some successful controller F for the plant P[7] (the robot arm from Fig. 1 c, is the example of a plant P discussed in this article), is again a time-invariant fading memory filter. But without additional constraints on the plant and/or target controller F one cannot guarantee that neural microcircuits L that uniformly approximate such successful controller F can also successfully control the plant P. In this context we say that F can be uniformly approximated by neural microcircuit model L if there exists for every $\varepsilon > 0$ some neural microcircuit model L so that $\|(Fu)(t)-(Lu)(t)\| \leq \varepsilon$ for all times t and all input functions $u(\cdot)$ that may enter the controller. Note that the feedback f from the plant has to be subsumed by these functions $u(\cdot)$, so that $u(t)$ is in general of the form $u(t) = \langle u_0(t), f(t)\rangle$, where $u_0(t)$ are external control commands (both

[7] P should satisfy the well-known bounded input, bounded output (BIBO) criteria in control theory.

$u_0(t)$ and $f(t)$ are in general multi-dimensional).[8] Assume that such microcircuit model L has been chosen for some extremely small $\varepsilon > 0$. Nevertheless the plant P may magnify the differences $\leq \varepsilon$ between outputs from F and outputs from L (which may occur even if F and L receive initially the same input u) and produce feedback functions $f_F(s), f_L(s)$ whose difference is fairly large. The difference between the outputs of F and L for these different feedbacks may become much larger than ε, and the outputs of F and L (and of the plant) in this closed loop may eventually diverge. This situation does in fact occur in the case of a 2-joint arm as plant P. Hence the assumption that L approximates F uniformly within ε cannot guarantee that $\|(Fu_F)(t) - (Lu_L)(t)\| \leq \varepsilon$ for all t (where $u_F(t) := \langle u_0(t), f_F(t)\rangle$ and $u_L(t) := \langle u_0(t), f_L(t)\rangle$), since even $\|(Fu_F)(t) - (Fu_L)(t)\|$ may already become much larger than ε for sufficiently large t. On the other hand if one assumes that the target controller F is "contracting" in the neighborhood of its intended working range so that $\|(Fu_F)(t) - (Fu_L)(t)\|$ stays small if $u_F(\cdot)$ and $u_L(\cdot)$ did not differ too much at preceding time steps, one can bound $\|(Fu_F)(t) - (Lu_L)(t)\|$ by $\|(Fu_F)(t) - (Fu_L)(t)\| + \|(Fu_L)(t) - (Lu_L)(t)\|$ and thereby avoid divergence of the trajectories caused by F and L in the closed-loop system.

The assumption that the target control filter F is "contracting" in the neighborhood of its intended working range, i.e. for external controls $u_0(t)$ that are actually used, was met by the target filters F that the neural microcircuit models were trained to approximate for the experiments reported in this article: these target filters F, which were used to generate the target values for training the readouts of the neural microcircuit models L were assumed to give for small variations of their input functions $u(\cdot)$ the *same* output. Hence they were contracting in the neighborhood of their intended working range. Through these considerations it now becomes clear why it was necessary to train the readouts of the neural microcircuit models L not just for single functions $u_F(t)$ but also for noisy variations of the ideal input function $u_F(t) = \langle u_0(t), f_F(t)\rangle$ that arise from functions $f_F(t)$ produced by the plant P in response to the outputs of target controller F: without this training procedure it would have been impossible to train the neural circuit models to approximate a *contracting* controller.

8 Discussion

Whereas traditional models for neural computation had focused on constructions of neural implementations of Turing machines or other offline computational models, more recent results have demonstrated that biologically more realistic neural microcircuit models consisting of spiking neurons and dynamic synapses are well-suited for real-time computational tasks ([Buonomano and Merzenich, 1995], [Maass et al., 2002], [Maass et al., 2003], [Natschläger and Maass, 2004]). Whereas related work had so far focused on sensory processing tasks such as speech recognition or visual

[8] In our experiments $u_0(t)$ was a very simple 2-dimensional function with value $\langle 0, 0\rangle$ for $t < 0$ and value $\langle x_{dest}, y_{dest}\rangle$ for $t \geq 0$. All other external inputs to the circuit were only given during training.

movement analysis ([Buonomano and Merzenich, 1995], [Maass et al., 2002], [Legenstein et al., 2003]) we have applied such models here for the first time in a biologically more realistic closed loop setting, where the output of the neural microcircuit model directly influence its future inputs. Obviously closed loop applications of neural microcircuit models provide a harder computational challenge than open loop sensory processing, since small imprecisions in their output are likely to be amplified by the plant to yield even larger deviations in the feedback, which is likely to increase even further the imprecision of subsequent movement commands. This problem can be solved by teaching the readout from the neural microcircuit during training to ignore smaller deviations reported by feedback, thereby making the target trajectory of output torques an attractor in the resulting closed-loop dynamical system. After training, the learned reaching movements are generated completely autonomously by the neural circuit once it is given the target end position of the tip of the robot arm as (static) input. Furthermore the capability of the neural circuit to generate reaching movements automatically generalizes to novel target end positions of the tip of the robot arm that did not occur during training (see Fig. 5). Furthermore the velocity profile for these autonomously generated new reaching movements exhibits a bell-shaped velocity profile, like for the previously taught movement primitives. Surprisingly the performance of the neural microcircuit model for generating movement primitives not only deteriorates if the (simulated) proprioceptive feedback is delayed by more than 250 ms, or if no feedback is given at all, but also if this feedback arrives without any delay. The best performance is achieved if the feedback arrives with a significant delay in the range of 25 to 100 ms. If the delay assumes other values, or is missing altogether, a significant improvement in the precision of the generated reaching movements is achieved after additional readouts from the same neural microcircuit models that generate the movements have been taught to estimate the values of the feedback with an optimal delay of 200 ms, and if the results of these internally generated feedback estimates are provided as additional inputs to the circuit (see Fig. 6 b). Apart from these effects resulting from the interaction of the inherent circuit dynamics with the dynamics of external or internally generated feedbacks, also the spatial organization of information streams in the simulated neural microcircuit plays a significant role. The capability of such a circuit to generate movements is quite bad if information about slowly varying input variables (such as external or internally generated feedback) is provided to the circuit in the form of a firing rate of a single neuron (not shown), rather than through the firing activity of a spatially extended array of inputs (see description in section 2) as implemented for the experiments reported in this article. Thus altogether these results may be viewed as a first step towards an exploration of the role of the "embodiment of neural computation" in concrete spatially extended neural circuit models and their resulting inherent temporal dynamics. This may complement the already existing work on the relevance of the embodiment of actuators to motor control [Pfeifer, 2002], and might possibly lead to a better understanding of biological motor control, and also provide new ideas for the design of robot controllers. The

paradigm for movement generation discussed in this article is somewhat related to preceding work [Ijspeert et al., 2003], where abstract systems of differential equations were used, and to the melody-generation with artificial neural networks in discrete time of [Jäger, 2002]. In these preceding models no effort was made to choose a movement generator whose inherent dynamics has a similarity to that of biological neural circuits. It has not yet been sufficiently investigated whether feedback; especially feedback with a realistic delay, can have similarly beneficial consequences in these other models. No effort was made in this article to make the process by which the neural circuit model (more specifically: the readouts from this circuit) learn to generate specific movement primitives in a biologically realistic fashion. Hence the results of this article only provide evidence that a generic neural microcircuit can hold the information needed to generate certain movement primitives, and once it has this information it can automatically use it to generate movements to other given targets. In some organisms such information is provided through the genetic code, often in combination acquired from observation or trial-and-error. It remains to be explored to what extent a neural microcircuit model can also learn through trial-and-error (i.e., reinforcement learning) to execute basic movements, or to combine movement primitives to yield more complex movements. We believe that the control framework presented in this article, based on a model for a neural system that can be chosen to be as complex and biologically realistic as one wants to, provides a quite fruitful platform for investigating the possible role and interaction of genetic information as well as various biological learning mechanisms, since it allows us to explore the role of biologically realistic models for neural system in the context of a functional closed-loop model where complex real-world movement control tasks can be addressed.

Possibly some of the results reported in this article also provide new ideas for tackling complex robot control tasks even in contexts where superior performance rather than biological realism in required. As indicated in [Maass et al., 2003], there are various methods for abstracting salient computational functions of generic neural microcircuits in ways that can be implemented much more efficiently on digital computers than a straightforward simulation of a neural microcircuit (see [Nessler and Maass, 2003]). In this way the inspiration from biological movement control may give rise to new methods for real-time adaptive robot control that help to solve some of the challenging open problems in that area.

Acknowledgment. We would like to thank Auke Ijspeert and Henry Markram for stimulating discussions and anonymous reviewers for helpful comments.

References

[Buonomano and Merzenich, 1995] Buonomano, D. V. and Merzenich, M. M. (1995). Temporal information transformed into a spatial code by a neural network with realistic properties. *Science*, 267:1028–1030.

[Gupta et al., 2000] Gupta, A., Wang, Y., and Markram, H. (2000). Organizing principles for a diversity of GABAergic interneurons and synapses in the neocortex. *Science*, 287:273–278.

[Ijspeert et al., 2003] Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2003). Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems (NIPS 2002)*, volume 15. MIT Press.

[Jäger, 2002] Jäger, H. (2002). Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach. GMD Report 159, German National Research Center for Information Technology.

[Legenstein et al., 2003] Legenstein, R. A., Markram, H., and Maass, W. (2003). Input prediction and autonomous movement analysis in recurrent circuits of spiking neurons. *Reviews in the Neurosciences (Special Issue on Neuroinformatics of Neural and Artificial Computation)*, 14(1–2):5–19. Online available as #140 from http://www.igi.tugraz.at/maass/publications.html.

[Maass and Markram, 2002] Maass, W. and Markram, H. (2002). On the computational power of recurrent circuits of spiking neurons. *submitted for publication*. Online available as #135 from http://www.igi.tugraz.at/maass/publications.html.

[Maass et al., 2002] Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560. Online available as #130 from http://www.igi.tugraz.at/maass/publications.html.

[Maass et al., 2003] Maass, W., Natschläger, T., and Markram, H. (2003). Computational models for generic cortical microcircuits. In Feng, J., editor, *Computational Neuroscience: A Comprehensive Approach*, chapter 18. CRC-Press. to appear. Online available as #149 from http://www.igi.tugraz.at/maass/publications.html.

[Markram et al., 1998] Markram, H., Wang, Y., and Tsodyks, M. (1998). Differential signaling via the same axon of neocortical pyramidal neurons. *Proc. Natl. Acad. Sci.*, 95:5323–5328.

[Natschläger and Maass, 2004] Natschläger, T. and Maass, W. (2004). Information dynamics and emergent computation in recurrent circuits of spiking neurons. *Proc. of NIPS 2003, Advances in Neural Information Processing Systems*, 16, MIT Press. to appear. Online available as #150 from http://www.igi.tugraz.at/maass/publications.html.

[Nessler and Maass, 2003] Nessler, B. and Maass, W. (2003). A brain-like architecture for real-time computing. *in preparation*.

[Pfeifer, 2002] Pfeifer, R. (2002). On the role of embodiment in the emergence of cognition: Grey walter's turtles and beyond. In *Proc. of the Workshop "The Legacy of Grey Walter"*, Bristol. see http://www.ifi.unizh.ch/groups/ailab/.

[Slotine and Li, 1991] Slotine, J. J. E. and Li, W. (1991). *Applied Nonlinear Control*. Prentice Hall, Englewood Cliffs, New Jersey.

[Tani, 2003] Tani, J. (2003). Symbols and dynamics in embodied cognition: Revisiting a robot experiment. In Butz, M. V., Sigaud, O., and Gerard, P., editors, *Anticipatory Behavior in Adaptive Learning Systems*, pages 167–178. Springer.