Doctoral Thesis

On the role of feedback in enhancing the computational power of generic neural microcircuits

Prashant Joshi

Institute for Theoretical Computer Science
Technische Universität Graz,
Austria

Graz, November, 2006

Acknowledgments

I would like to thank the following people:

- Wolfgang Maass, my supervisor, for providing me with invaluable guidance throughout my research. This research would not have been possible without his mentoring, motivation, trust, criticisms, corrections, and a never-ending demand for excellence. I worked with him on topics described in Chapters 2 and 3.
- Helmut Pockberger for his friendly willingness to be the second referee of this thesis.
- Eduardo D. Sontag, with whom I worked on the topic presented in Chapter 2.
- Wulfram Gerstner, Auke Ijspeert, Herbert Jaeger, Konrad Koerding, Thomas Natschlaeger, Henry Markram, Gordon Pipa, Misha Tsodyks, Tony Zador, and Carlos Brody for their helpful comments on my research.
- Stefan Haeusler, Michael Pfeiffer and Robert Legenstein for general help and discussion.
- The whole "IGI-family" for making my stay here so enjoyable.
- My parents, for their love, affection, encouragement and support througout my education. I hope to make you proud.
- In addition I acknowledge support by following sources:
    1. "Fonds zur Förderung der wissenschaftlichen Forschung (FWF), Austrian Science Fund", projects #P15386, #S9102-N04, and #P17229-N04.
    2. PASCAL, project #IST2002-506778
    3. FACETS, project #FP6-015879

Abstract

Circuits of neurons in the brain perform diverse cortical computations in parallel, endowing the organism with diverse cortical modalities, e.g. motor control, vision, and audition; and higher order cognitive processes, e.g. planning, and decision making. It is believed that these computations are carried out by network of neurons in cortical microcircuits, where each microcircuit is composed of rather stereotypical circuit of neurons within a cortical column. A characteristic property of these cortical circuits is the presence of abundant feedback connections, be it on the level of *recurrent axon collaterals* projecting back onto the same neuron, or on the network level between different cortical areas. This thesis explores the functional role of neural feedback in enhancing the computational power of generic neural microcircuits. It is shown that feedback endows standard models for neural circuits with the capability to emulate arbitrary Turing machines. In fact, with a suitable feedback such circuits can simulate any dynamical system, in particular any conceivable analog computer. Under realistic noise conditions the computational power of these circuits is obviously reduced. However it is demonstrated through computer simulations that feedback also provides a significant gain in computational power for quite detailed models of cortical microcircuits with in-vivo-like high levels of noise. Furthermore neurocomputational models using generic neural microcircuits with feedback are explored in the context of motor control, decision making, and "action selection in presence of decisions".

Zusammenfassung

Schaltkreise von Neuronen im Hirn führen unterschiedliche Berechnungen parallel aus, welche den Organismus mit verschiedenen kortikalen Fähigkeiten, wie z.B. Motorkontrolle, Sehen, Hören und kognitiven Prozessen höherer Ordnung ausstatten. Es wird angenommen, dass diese Berechnungen von Netzwerken von Neuronen in kortikalen Mikroschaltkreisen durchgeführt werden, in welchen jeder Schaltkreis aus annähernd stereotypischen Schaltkreisen von Neuronen innerhalb einer kortikalen Säule zusammengesetzt ist. Eine charakteristische Eigenschaft dieser kortikalen Schaltkreise ist das Vorhandensein von mannigfaltigen Rückkopplungen, sei es auf dem Level von axonalen Kollateralen, welche zurück auf das selbe Neuron projizieren, oder auf dem Netzwerklevel zwischen verschiedenen kortikalen Arealen. Diese Arbeit untersucht die funktionelle Rolle von neuronalen Rückkopplungen für die Verbesserung der Rechenfähigkeiten von generischen neuronalen Mikroschaltkreisen. Es wird gezeigt, dass Rückkopplungen Standardmodelle von neuronalen Mikroschaltkreisen mit der Fähigkeit der Emulation von beliebigen Turingmaschinen ausstatten. Mit geeigneten Rückkopplungen können solche Schaltkreise jedes dynamische System simulieren, insbesondere jeden vorstellbaren analogen Computer. Unter realistischen Rausch-Bedingungen sind die Rechenfähigkeiten dieser Schaltkreise offensichtlich reduziert. Allerdings wird durch Computersimulationen gezeigt, dass Rückkopplungen auch zu einer signifikanten Steigerung der Rechenfähigkeiten von detaillierten Modellen von kortikalen Mikroschaltkreisen mit in-vivo artigen hohen Rauschleveln führt. Weiters werden Computermodelle neuronaler Informationsverarbeitung, welche generische neuronale Mikroschaltkreise mit Rückkopplungen verwenden, im Kontext von Motorkontrolle, Entscheidungsfindung und Handlungsselektion in Entscheidungssituationen untersucht.

Contents

Chapter 1

Introduction and Overview

The brain performs a wide range of complex computations in real-time, thereby endowing an organism with diverse cortical modalities essential for survival. It is believed that these computations are performed by network of neurons in cortical microcircuits, where each microcircuit is composed of rather stereotypical circuit of neurons within a cortical column. A characteristic property of these cortical circuits is the presence of abundant feedback connections, e.g. pyramidal neurons in the cortex typically have in addition to their long projecting axon, a large number of axon collaterals that provide feedback to the local circuit [1]. Abundant feedback connections also exist on the network level between different brain areas [2]. A vast body of existing neurophysiological evidence implies that diverse forms of neural feedback signals play an important role in cortical computations, for example:

- Deactivating the feedback signal from visuoparietal cortex to cat area 18 (part of the primary visual cortex), decreases the signal strength in both orientation and direction maps and almost destroys the global layout of direction maps, thereby suggesting the strong contribution of this feedback signal to the emergence of direction selectivity in early visual areas [3].

- Unilateral destruction of lateral olivocochlear (LOC) efferents, i.e. the feedback synapses from lateral superior olive (LSO) to cochlea corrupts the normal interaural correlation in response amplitudes to sounds of equal intensity, suggesting that lateral olivocochlear feedback maintains the binaural balance in neural excitability essential for correct localization of sounds in space [4]. Also, selective removal of these LOC efferent synapses has been shown to increase the probability of acute acoustic injury, implying the role of LOC feedback in modulating the cochlear nerve excitability [5].

- Sending a controlled external feedback that is consistent with a leaky or unstable integrator to the occulomotor neural integrator of goldfish over tens of minutes to hours, causes the occulomotor integrator to become progressively more unstable or leaky respectively. Normal external feedback signal returns the occulomotor integrator to stability, demonstrating that external visual feedback plays a vital role in gradually tuning the stability of the neural integrator [6].

An open research question concerning neural feedback is its exact functional role in computations carried out by cortical microcircuits. This work presents theoretical and simulation results that characterize the gain in computational power that neural feedback can provide in generic neural microcircuits[7, 8]. Furthermore, neurocomputational models are presented that demonstrate the role of feedback in tasks involving motor control [9, 10], working memory and decision making [11] and integrating the sense-think-decide-act phases involved in action selection in presence of decisions [12].

1.1 Computational role of neural feedback

The theoretical results described in chapter 2 show that feedback endows standard models for neural circuits with the capability to emulate arbitrary Turing machines. In fact, with a suitable feedback they can simulate any dynamical system, in particular any conceivable analog computer. Under realistic noise conditions the computational power of these circuits is necessarily reduced. It is demonstrated through computer simulations that feedback also provides a significant gain in computational power for quite detailed models of cortical microcircuits with in-vivo-like high levels of noise. In particular it enables generic cortical microcircuits to carry out computations that combine information from working memory and persistent internal states in real-time with new information from online input streams.

Quite demanding real-time computations with fading memory[1] can be carried out by generic cortical microcircuit models [13]. But many types of computations

[1] A map (or filter) F from input- to output streams is defined to have *fading memory* if its current output at time t depends (up to some precision ε) only on values of the input $\mathbf{u}$ during some finite time interval $[t-T,t]$. Formally, a filter F has *fading memory* if there exists for every $\varepsilon > 0$ some $\delta > 0$ and $T > 0$ so that $|(F\mathbf{u})(t) - (F\mathbf{v})(t)| < \varepsilon$ for any $t \in \mathbb{R}$ and any input functions $\mathbf{u}, \mathbf{v}$ with $\|\mathbf{u}(\tau) - \mathbf{v}(\tau)\| < \delta$ for all $\tau \in [t-T,t]$.

Note that fading memory is just a continuity property (that is characteristic for filters whose output is defined via one or several integrals over the input functions) which implies graceful decay of the output precision when there is noise on the input: the most relevant bits of the output value $(F\mathbf{u})(t)$ depend only on the most relevant bits of $\mathbf{u}(s)$ for arguments s from some finite domain $[t-T,t]$. It is easy to see that any linear filter, and each higher order term of a Volterra series, is time invariant and has fading memory.

in the brain, for example computations that involve memory or persistent internal states, cannot be modeled by such fading memory systems. On the other hand concrete examples of artificial neural networks [14] and cortical microcircuit models [10] suggest that their computational power can be enlarged through feedback from trained readouts. Furthermore the brain is known to have an abundance of feedback connections on several levels: within cortical areas, where pyramidal cells typically have in addition to their long projecting axon a number of local axon collaterals, between cortical areas, and between cortex and subcortical structures. But the computational role of these feedback connections has remained open. We present here a computational theory which characterizes the gain in computational power that a fading memory system can acquire through feedback from trained readouts, both in the idealized case without noise and in the case with noise. This theory simultaneously characterizes the potential gain in computational power resulting from training a few neurons *within* a generic recurrent circuit for a specific task. Applications of this theory to cortical microcircuit models provide a new way of explaining the possibility of real-time processing of afferent input streams in the light of learning-induced internal circuit states that might represent for example working memory or rules for the timing of behavior.

1.2 Movement generation and control

How can complex movements that take hundreds of milliseconds be generated by stereotypical neural microcircuits consisting of spiking neurons with a much faster dynamics? Chapter 3 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 chapter, 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 a long range program where abstract control principles for biological movement control and related models developed for artificial neural networks [15] can be implemented and tested on arbitrarily realistic models for the underlying neural circuitry.

The feasibility of this approach is demonstrated in this chapter 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 [16]. Such movement generation is independent of the arm-model used and the type of feedbacks that the circuit receives. This is demonstrated by considering two different models of a two-jointed arm, a standard model from robotics and a standard model from biology, that each generate different kinds of feedback. 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 50 to 280 ms. Additionally it is shown that the generic neural microcircuit models 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 ([17], [18], [19]). 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 chapter. Somewhat similar paradigms for neural control based on artificial neural network models have been independently explored by Herbert Jaeger [20].

1.3 Goal-directed movement in presence of decisions

Decision making lies towards the end-stage of one of the most taxing problems organisms face: action selection in an uncertain world [21]. The interval discrimination task [22, 23] is one of the classical experimental paradigms that is employed to study working memory and decision making. The experiment typically involves four phases, *viz.* the initial loading (L) of the first stimulus, maintaining (M) this stimulus in working memory till the subsequent stimulus is presented, making a binary decision (D), and finally acting (A) on this decision, usually by pressing one of the two buttons corresponding to the binary choice.

The precise computational and biophysical mechanism(s) through which the brain is able to execute this load-maintain-decide-act (LMDA) sequence is not understood. Several theoretical and modeling studies have tried to look at segregated phases of this sequence and give possible explanations for their working [24, 25, 26]. Typically neurocomputational models of working memory fail to address how the decisions made by neurons in the prefrontal cortex (PFC), are converted into motor commands, which are executed by the sensori-motor system [22, 27, 28]. Similarly, models for computational motor control tend to ignore the first three phases (LMD), that are responsible for generation of motor commands [29, 9, 10].

Several interesting modeling approaches for tasks involving working memory and decision making have been proposed recently [22, 28, 30]. These models propose different mechanisms e.g. precise tuning of mutual inhibition [22], fine-tuning of a heterogeneous recurrent network [30], using an integral feedback signal for inhibitory control [28]; to obtain the persistent neural activity which in turn stores information in the working memory. Despite existing evidence that shows synaptic learning as a responsible mechanism for working memory related tasks [23], all the models described above use static (no learning involved) neural circuits.

Chapter 4 proposes a neurocomputational architecture that uses synaptic learning mechanisms (simply linear regression), and is able to integrate the four phases (LMDA) involved in the process of action selection in presence of a decision, into a unified computational framework. Essentially the neural model described in this chapter integrates two distinct cortical functions *viz.*, working memory and decision making carried out by the neurons in PFC, and subsequent action selection executed by the sensorimotor system. More precisely, it is demonstrated that delayed-decision tasks that are followed by action selection, can be solved, if feedback from trained linear readouts is provided to generic neural microcircuits whose internal dynamics has not been optimized for any particular computational task. Two classical experimental paradigms for interval-discrimination task are modeled that use different mechanisms to encode external sensory inputs. For comparison with earlier models of working memory, the unified framework is used to build a spiking neural network model of two-interval discrimination [22]. Additionally, to demonstrate that this computational paradigm is task-independent, robust to how the external sensory inputs are encoded, and is capable of integrating the A phase, another spiking neural network model is presented for the delayed-match-to-sample task [23], followed by an arm movement to the decided goal position.

The core principles behind the working of this model make the assumption that the cortex can be thought of as an ultra-high dimensional dynamical system, where the afferent inputs arriving from thalamus and the recurrent cortical feedbacks are churned in a non-linear way to obtain a high-dimensional projection of the low-dimensional input space. Preceding work has demonstrated that such high dimensional transient dynamics provides the neural circuit with analog fading memory that provides the circuit enough computational power for performing open-loop sensory processing tasks [17, 13].

Analog fading memory by itself is not powerful enough to render the circuits the power to hold information in working memory. The obvious reason being that analog fading memory by itself has an upper limit on the order of tens of msec, depending on the time constants of synapses and neurons in the neural circuit [13], whereas typically the working memory holds information on the order of seconds. Recent results show that feedback from trained readout neurons that are part of generic neural circuit can induce multiple co-existing "partial attractors" in the circuit dynamics [7, 8]. This result is further extended in this chapter to demonstrate that even in the presence of feedback noise, such "partial attractor" states can be held by generic neural circuits on the time-scales of several seconds, that is obviously a requirement for tasks involving working memory.

Chapter 2

Computational aspects of feedback in neural circuits

It had previously been shown that generic cortical microcircuit models can perform complex real-time computations on continuous input streams, provided that these computations can be carried out with a rapidly fading memory. This chapter investigates the computational capability of such circuits in the more realistic case where not only readout neurons, but in addition a few neurons within the circuit have been trained for specific tasks. This is essentially equivalent to the case where the output of trained readout neurons is fed back into the circuit. It is shown that this new model overcomes the limitation of a rapidly fading memory. In fact, it is proved that in the idealized case without noise it can carry out any conceivable digital or analog computation on time-varying inputs. But even with noise the resulting computational model can perform a large class of biologically relevant real-time computations that require a non-fading memory. This chapter demonstrates these computational implications of feedback through computer simulations of detailed cortical microcircuit models that are subject to noise and have a complex inherent dynamics and also provides a summary of theoretical results. It is shown that the application of simple learning procedures (such as linear regression or perceptron learning) to a few neurons enables such circuits to represent time over behaviorally relevant long time spans, to integrate evidence from incoming spike trains over longer periods of time, and to process new information contained in such spike trains in diverse ways according to the current internal state of the circuit. In particular it is shown that such generic cortical microcircuits with feedback provide a new model for working memory that is consistent with a large set of biological constraints.

Although this chapter examines primarily the computational role of feedback in circuits of neurons, the mathematical principles on which its analysis is based apply to a variety of dynamical systems. Hence they may also throw new light on the computational role of feedback in other complex biological dynamical systems, such as for example genetic regulatory networks.

2.1 Introduction

The neocortex performs a large variety of complex computations in real-time. It is conjectured that these computations are carried out by a network of cortical microcircuits, where each microcircuit is a rather stereotypical circuit of neurons within a cortical column. A characteristic property of these circuits and networks is an abundance of feedback connections. But the computational function of these feedback connections is largely unknown. Two lines of research have been engaged in order to solve this problem. In one approach, which one might call the constructive approach, one builds hypothetical circuits of neurons and shows that (under some conditions on the response behavior of its neurons and synapses) such circuits can perform specific computations. In another research strategy, which one might call the analytical approach, one starts with data-based models for actual cortical microcircuits, and analyses which computational operations such "given" circuits can perform under the assumption that a learning process assigns suitable values to some of their parameters (e.g. synaptic efficacies of readout neurons). An underlying assumption of the analytical approach is that complex recurrent circuits, such as cortical microcircuits, cannot be fully understood in terms of the usually considered properties of their components. Rather, system level approaches that address directly the dynamics of the resulting recurrent neural circuits are needed to complement the bottom-up analysis. This line of research started with the identification and investigation of so called canonical microcircuits [31]. Several issues related to cortical microcircuits have also been addressed in the work of Grossberg; see [32] and the references therein. Subsequently it was shown that quite complex real-time computations on spike trains can be carried out by such "given" models for cortical microcircuits ([17, 33, 13, 34], see [35] for a review). A fundamental limitation of this approach was that only those computations could be modeled which can be carried out with a fading memory, more precisely only those computations that only require to integrate information over a time span of 200 or 300 ms (its maximal length depends on the amount of noise in the circuit and the complexity of the input spike trains [36]). In particular, computational tasks that require a representation of elapsed time between salient sensory events or motor actions [37], or an internal representation of expected rewards [38, 39, 40], working memory [41], accumulation of sensory evidence for decision making [42], the updating and holding of analog variables such as for example the desired eye position [6], and differential processing of sensory input streams according to attentional or other internal states of the neural system [43] could not be modeled in this way. Previous work on concrete examples of artificial neural networks [14] and cortical microcircuit models [10] had already indicated that these shortcomings of the model might arise only if one assumes that learning affects exclusively the synapses of readout neurons that project the results of computations to other circuits or areas, without giving feedback into the circuit from which they extract information. This scenario is in fact rather unrealistic from a biological perspective, since pyramidal neurons in the cortex typically have in addition to their long projecting axon a large number of axon collaterals that provide feedback to the local circuit [1]. Abundant feedback connections also exist on the network level between different brain areas [2]. This chapter shows that if one takes feedback connections from readout neurons (that are trained for specific tasks) into account, generic cortical microcircuit models can solve all of the previously listed computational tasks. In fact, one can demonstrate this also for circuits whose underlying noise levels and models for neurons and synapses are substantially more realistic than those which had previously been considered in models for working memory and related tasks.

It is shown in section 2.2.1 that the significance of feedback for the computational power of neural circuits and other dynamical systems can be explained on the basis of general principles. Theorem 2.2.1 implies that a large class of dynamical systems, in particular systems of differential equations which are commonly used to describe the dynamics of firing activity in neural circuits, gain universal computational capabilities for digital and analog computation as soon as one considers them in combination with feedback. A further mathematical result (Theorem 2.2.2) implies that the capability to process online input streams in the light of non-fading (or slowly fading) internal states is preserved in the presence of fairly large levels of internal noise. On the basis of this theoretical foundation one can explain why the computer models of generic cortical microcircuits, which are considered in section 2.2.2, are able to solve the previously mentioned benchmark tasks. These results suggest a new computational model for cortical microcircuits, which includes the capability to process online input streams in diverse ways according to different "instructions" that are implemented through high-dimensional attractors of the underlying dynamical system. The high-dimensionality of these attractors results from the fact that only a small fraction of synapses need to be modified for their creation. In comparison with the commonly considered low dimensional attractors, such high-dimensional attractors have additional attractive properties such as compositionality (the intersection of several of them is in general non-empty), and compatibility with real-time computing on online input streams within the same circuit.

The presentation of theoretical results for abstract models (without proofs) is given in section 2.2.1. Details to the computer simulations of more detailed cortical microcircuit models (discussed in section 2.2.2) can be found in section 2.4.1 of section 2.4. A discussion of the results described in this chapter is given in section 3.9.

2.2 Results

Two types of models for neural circuits were considered:

1. Mean field models, such as those defined by equation (2.6) in section 2.2.1, which model the dynamics of firing rates of neurons in neural circuits. These models have the advantage that they are theoretically tractable, but they have the disadvantage that they do not reflect many known details of cortical microcircuits. However it is shown in [8] that the theoretical results that are mentioned in section 2.2.1 hold for fairly large classes of dynamical systems. Hence they potentially also hold for some more detailed models of neural circuits.

2. Section 2.2.2 considers quite detailed models of cortical microcircuits consisting of spiking neurons (see the description in section 2.2.2 and 2.4.1). At present these models cannot be analyzed directly by theoretical methods, hence one can only present statistical data from computer simulations. The simulation results show that feedback has in these more detailed models a variety of computational consequences that have been derived analytically for the simpler models of section 2.2.1. This is not totally surprising insofar, as the computations that are considered in the more detailed models can be approximately described in terms of time-varying firing rates for individual neurons.

In both types of models the focus is on computations that transform time-varying input streams into time-varying output streams. The input streams are modeled in section 2.2.1 by time-varying analog functions $u(t)$ (that might for example represent time-varying firing rates of neurons that provide afferent inputs), and in section 2.2.2 by spike trains generated by Poisson processes with time-varying rates. Output streams are analogously modeled by time-varying firing rates, or directly by spike trains. I believe that such online computations, which transform time-varying inputs into time-varying outputs, provide a better framework for modeling cortical processing of information than computations that transform a static vector of numbers (i.e., a batch input) into a static output. Mappings from time-varying inputs to time-varying outputs are referred to as filters (or operators) in mathematics and engineering. A frequently discussed reference class of linear and nonlinear filters are those which can be described by Volterra- or Wiener series (see e.g. [44]). These filters can equivalently be characterized as those filters which are time-invariant (i.e., they are input-driven and have no "internal clock") and have a fading memory (see [13]). Fading memory means intuitively that the influence of any specific segment of the input stream on later parts of the output stream becomes negligible when the length of the intervening time interval is sufficiently large. It is shown in the next two subsections that feedback endows a circuit, that by itself can only carry out computations with fading memory, with flexible ways of combining fading-memory-computations on time varying inputs with computational operations on selected pieces of information in a non-fading memory.

2.2.1 Theoretical Analysis

The dynamics of firing rates in recurrent circuits of neurons is commonly modeled by systems of nonlinear differential equations of the form

$$x_i'(t) \;=\; -\lambda_i x_i(t) \,+\, \sigma\,(\sum_{j=1}^{n} a_{ij}x_j(t) + b_i \cdot v(t))\,, \quad i = 1,\ldots,n\;, \tag{2.1}$$

or

$$x_i'(t) \;=\; -\lambda_i x_i(t) \,+\, \sigma\,(\sum_{j=1}^{n} a_{ij}x_j(t)) \,+\, b_i \cdot \sigma(v(t))\,, \quad i = 1,\ldots,n \tag{2.2}$$

([45, 46, 47, 48]). Here each $x_i, i = 1,\ldots,n$, is a real-valued variable which represents the current firing rate of the i^{th} neuron or population of neurons in a recurrent neural circuit, and $v(t)$ is an external input stream. The coefficients a_{ij}, b_i denote the strengths of synaptic connections, and the $\lambda_i > 0$ denote time constants. The function σ is some sigmoidal activation function (nondecreasing, with bounded range). In most models of neural circuits, the parameters are chosen so that the resulting dynamical system has a fading memory for preceding inputs. If one makes the synaptic connection strengths a_{ij} in (2.1) or (2.2) so large that recurrent activity does not dissipate, the neural circuit tends to exhibit persistent memory. But it is usually quite difficult to control the content of this persistent memory, since it tends to be swamped with minor details of external inputs (or initial conditions) from the distant past. Hence this chaotic regime of recurrent neural circuits (see [49] for a review) is apparently also not suitable for biologically realistic online computations that combine new information from the current input with selected (e.g., behaviorally relevant) aspects of external or internal inputs from the past.

Recurrent circuits of neurons (e.g. those described by equations (2.1) or (2.2)) are from a mathematical perspective special cases of dynamical systems. The subsequent mathematical results show that a large variety of dynamical systems, in particular also fading memory systems of type (2.1) or (2.2), can overcome in the presence of feedback the computational limitations of a fading memory without necessarily falling into the chaotic regime. In fact, feedback endows them with *universal* capabilities for *analog computing*, in a sense that can be made precise in the following way (see Fig. 2.1A-C for an illustration):

Theorem 2.2.1 *A large class $\mathcal{S}_n$ of systems of differential equations of the form*

$$x_i'(t) = f_i(x_1(t), \ldots, x_n(t)) + g_i(x_1(t), \ldots, x_n(t)) \cdot v(t), \quad i = 1, \ldots, n \tag{2.3}$$

are in the following sense universal for analog computing:

This system (2.3) *can respond to an external input $u(t)$ with the dynamics of any n^{th} order differential equation of the form*

$$z^{(n)}(t) = G(z(t), z'(t), z''(t), \ldots, z^{(n-1)}(t)) + u(t) \tag{2.4}$$

(for arbitrary smooth functions $G : \mathbb{R}^n \to \mathbb{R}$) if the input term $v(t)$ is replaced in (2.3) *by a suitable memoryless feedback function $K(x_1(t), \ldots, x_n(t), u(t))$, and if a suitable memoryless readout function $h(\mathbf{x}(t))$ is applied to its internal state $\mathbf{x}(t) = \langle x_1(t), \ldots, x_n(t) \rangle$: one can achieve then that $h(\mathbf{x}(t)) = z(t)$ for any solution $z(t)$ of* (2.4)*.*

Also the dynamic responses of all systems consisting of several higher order differential equations of the form (2.4) *can be simulated by fixed systems of the form* (2.3) *with a corresponding number of feedbacks.*

This result says more precisely that for any n^{th} order differential equation (2.4) there exists a (memory-free) feedback function $K : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ and a memory-free readout function $h : \mathbb{R}^n \to \mathbb{R}$ (which can both be chosen to be smooth, in particular continuous) so that, for every external input $u(t), t \geq 0$, and each solution $z(t)$ of the forced system (2.4) there is an input $u_0(t)$ with $u_0(t) \equiv 0$ for all $t \geq 1$, so that the solution $\mathbf{x}(t) = \langle x_1(t), \ldots, x_n(t) \rangle$ of the fixed system (2.3)

$$\mathbf{x}'(t) \;=\; f(\mathbf{x}(t)) + g(\mathbf{x}(t))K(\mathbf{x}(t), u(t) + u_0(t)), \quad \mathbf{x}(0) = \mathbf{0} \tag{2.5}$$

(for $f : \mathbb{R}^n \to \mathbb{R}^n$ consisting of $\langle f_1, \ldots, f_n \rangle$ and $g : \mathbb{R}^n \to \mathbb{R}^n$ consisting of $\langle g_1, \ldots, g_n \rangle$) is such that

$$h(\mathbf{x}(t)) = z(t) \quad \text{for all } t \geq 1 \,.$$

Note that the function $u_0(t)$, that is added to the input for $t < 1$ (whereas $u_0(t) = 0$ for $t \geq 1$), allows the system (2.3) (and (2.5)) to simulate with a standardized initial condition $\mathbf{x}(0) = \mathbf{0}$ any solution of (2.4) with arbitrary initial conditions.

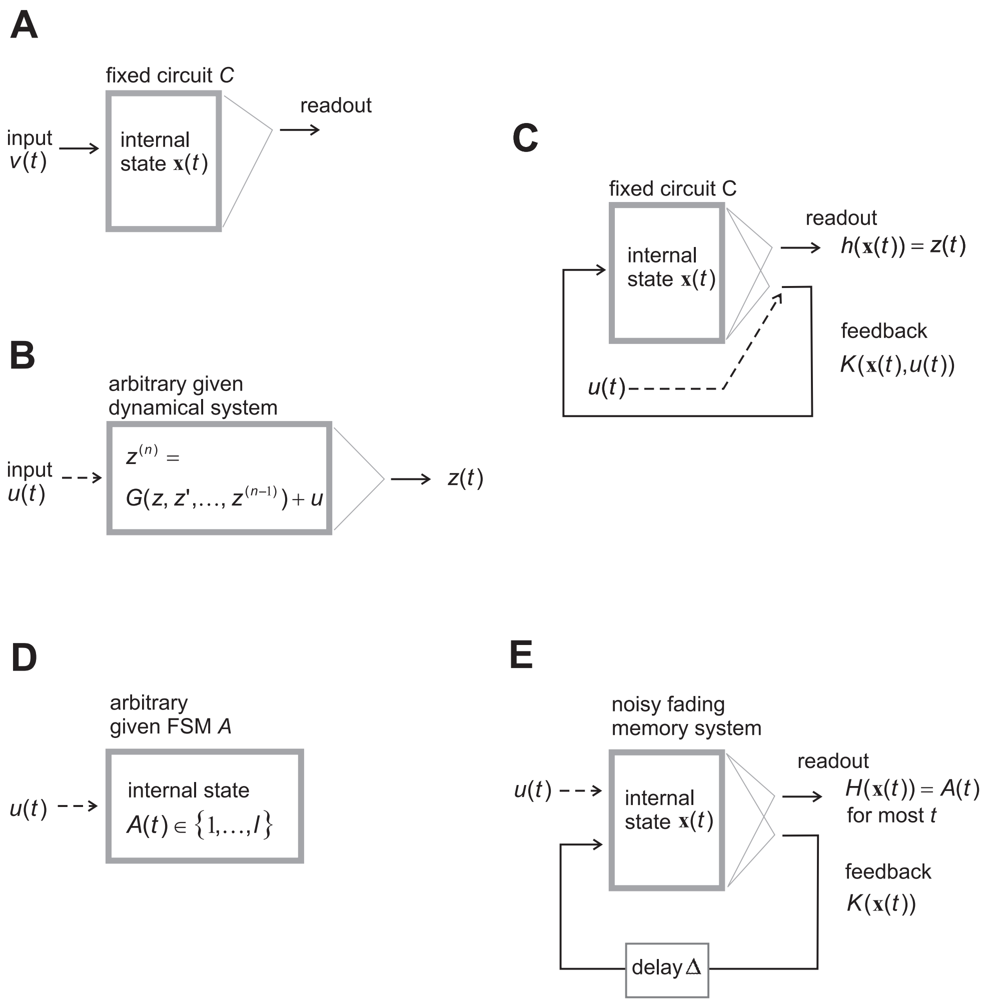
A
fixed circuit C
readout
input $v(t)$
internal state $\mathbf{x}(t)$
B
arbitrary given dynamical system
input $u(t)$
$z^{(n)} = G(z, z', \ldots, z^{(n-1)}) + u$
$z(t)$
C
fixed circuit C
readout $h(\mathbf{x}(t)) = z(t)$
internal state $\mathbf{x}(t)$
feedback $K(\mathbf{x}(t), u(t))$
$u(t)$
D
arbitrary given FSM A
$u(t)$
internal state $A(t) \in \{1, \ldots, l\}$
E
noisy fading memory system
$u(t)$
internal state $\mathbf{x}(t)$
readout $H(\mathbf{x}(t)) = A(t)$ for most t
feedback $K(\mathbf{x}(t))$
delay Δ

Figure 2.1: Computational architectures considered in Theorems 2.2.1 and 2.2.2. **(A)** A fixed circuit C whose dynamics is described by the system (2.3). **(B)** An arbitrary given n^{th} order dynamical system (2.4) with external input $u(t)$. **(C)** If the input $v(t)$ to circuit C is replaced by a suitable feedback $K(\mathbf{x}(t), u(t))$, then this fixed circuit C can simulate the dynamic response $z(t)$ of the arbitrarily given system shown in B, for any input stream $u(t)$. **(D)** Arbitrary given finite state machine (FSM) A with l states. **(E)** A noisy fading memory system with feedback can reliably reproduce the current state $A(t)$ of the given FSM A, except for time points t shortly after A has switched its state.

Theorem 2.2.1 implies that even if some fixed dynamical system (2.3) from the class $\mathcal{S}_n$ has fading memory, a suitable feedback K and readout function h will enable it to carry out specific computations with persistent memory. In fact, it can carry out *any* computation with persistent memory which could possibly be carried out by *any* dynamical system (2.4). To get a clear understanding of this universality property, one should note that the feedback function K and the readout function h depend only on the function G that characterizes the simulated system (2.4), but not on the external input $u(t)$ or the particular solution $z(t)$ of (2.4) that it simulates. Hence Theorem 2.2.1 implies in particular that any system (2.3) that belongs to the class $\mathcal{S}_n$ has in conjunction with several feedbacks the computational power of a universal Turing machine (see [50] or [51] for relevant concepts from computation theory). This follows from the fact that every Turing machine (hence any conceivable digital computation, most of which require a persistent memory) can be simulated by systems of equations of the form (2.4) (this was shown in [52] for the case with continuous time, and in [53, 54] for recurrent neural networks with discrete time; see [55] for a review). But possibly more relevant for applications to biological systems is the fact that any fixed system (2.3) that belongs to the class $\mathcal{S}_n$ is able to emulate any conceivable *continuous* dynamic response to an input stream $u(t)$ if it receives a suitable feedback $K(\mathbf{x}(t), u(t))$, where K can always be chosen to be continuous. Hence one may argue that these systems (2.3) are also universal for *analog* computing on time-varying inputs.

The class $\mathcal{S}_n$ of dynamical systems that become through feedback universal for analog computing subsumes systems of the form

$$x_i'(t) \;=\; -\lambda_i x_i(t) \,+\, \sigma\,(\sum_{j=1}^{n} a_{ij}\cdot x_j(t)) \,+\, b_i\cdot v(t)\,, \quad i=1,\ldots,n\,; \qquad (2.6)$$

for example if the λ_i are pairwise different and $a_{ij} = 0$ for all i, j, and all b_i are nonzero. Fewer restrictions are needed if more then one feedback to the system (2.6) can be used. Systems of the form (2.1) or (2.2) are of a slightly different form, since there the activation function σ (that has a bounded range) is applied to the term $v(t)$). But such systems (2.1), (2.2) can still be universal for all *bounded* analog responses of arbitrary dynamical systems (2.4), which are arguably the only ones of interest in a biological context. This follows from the fact that if the external input $u(t)$ of the system (2.4), as well as the resulting solution $z(t)$ and its derivatives $z^{(i)}t$ for $i \leq n-1$, stay within some bounded range, then the values of the feedback $v(t)$ that is needed for the simulation of (2.4) by (2.3) will also stay within a bounded range. More precisely, one has that:

For each constant $c > 0$ there is a constant $C > 0$ such that: for every external *input* $u(t), t \geq 0$, *and each solution* $z(t)$ *of the forced system* (2.4) *such that*

$$|u(t)| \leq c \text{ and } \left|z^{(i)}(t)\right| \leq c \quad \text{for all } i = 0, \ldots, n-1\,, \ \text{ for all } t \geq 0$$

the input u_0 *can be picked so that the feedback*

$$v(t) = K(\mathbf{x}(t), u(t) + u_0(t)) \quad t \geq 0$$

to (2.1) *or* (2.2) *satisfies:*

$$|v(t)| \leq C \quad \text{for all } t \geq 0\,.$$

Thus, if we know *a priori* that we will only deal with solutions of the differential equation (2.4) that are bounded by c, and inputs are similarly bounded, we could also consider instead of (2.3) a system such as $\mathbf{x}'(t) = f(\mathbf{x}(t)) + g(\mathbf{x}(t))\sigma(v(t))$ with $f, g : \mathbb{R}^n \to \mathbb{R}^n$, where some bounded activation function $\sigma : \mathbb{R} \to \mathbb{R}$ (e.g. $q \cdot \tanh(v)$, for a suitable constant q) is applied to the term $v(t)$ (like in (2.2)). The resulting feedback term $\sigma(K(\mathbf{x}(t), u(t) + u_0(t)))$ is then of a mathematical form which is adequate for modeling feedback in neural circuits.

Theorem 2.2.1 implies that a generic neural circuit may become through feedback a universal computational device, which can not only simulate any Turing machine, but also any conceivable model for analog computing with bounded dynamic responses. The "program" of such arbitrary simulated computing machine gets encapsulated in the static functions K that characterize the memoryless computational operations that are required from feedback units, and the static readout functions h. Since these functions are static, i.e. time-invariant, and continuous, they provide suitable targets for *learning*. More precisely, in order to train a generic neural circuit to simulate the dynamic response of an arbitrary dynamical system, it suffices to train - apart from readout neurons - a few neurons within the circuit (or within some external loop) to transform the vector $\mathbf{x}(t)$, that represents the current firing activity of its neurons, and the current external input $u(t)$ into a suitable feedback value $K(\mathbf{x}(t), u(t))$. This could for example be carried out by training a suitable feedforward neural network within the larger circuit, which can approximate any continuous feedback function K [56]. Furthermore I will show in section 2.2.2 that these feedback functions K can in many biologically relevant cases be chosen to be linear, so that it would in principle suffice to train a single neuron to compute K.

It is known that the memory capacity of such circuit is reduced to some finite number of bits if these feedback functions K are not learnt perfectly, or if there are other sources of noise in the system. More generally, no analog circuit with noise can simulate arbitrary Turing machines [57]. But the subsequent Theorem 2.2.2 shows that fading memory systems with noise and imperfect feedback can still achieve the maximal possible computational power within this a-priori limitation: they can simulate any given finite state machine (FSM). Note that any Turing machine with tapes of finite length is a special case of a FSM. Furthermore any existing digital computer is a FSM, hence the computational capability of FSM's is actually quite large.

In order to avoid the cumbersome mathematical difficulties that arise when one analyses differential equations with noise, Theorem 2.2.2 is formulated on a more abstract level, resorting to the notion of fading memory filters with noise. One assumes here that the input-output behavior of those dynamical systems with noise, for which one wants to determine the computational impact of (imprecise) state feedback, can be modeled by fading memory filters with additive noise on their output. The assumption that the amplitude of this noise is bounded is a necessary assumption according to [58]. Please refer to [33], [13], [59] for further discussions of the relationship between models for neural circuits and fading memory filters. In particular it was shown in [59] that every time-invariant fading memory filter can be approximated by models for neural circuits, provided that these models reflect the empirically found diversity of time constants of neurons and synapses.

Theorem 2.2.2 *Feedback allows linear and nonlinear fading memory systems, even in the presence of additive noise with bounded amplitude, to employ for real-time processing of time-varying inputs the computational capability and non-fading states of any given FSM (see Fig. 2.1D-E).*

The external input $u(t)$ can in this case be injected directly into the fading memory system, so that the feedback $K(\mathbf{x}(t))$ depends only on the internal state $\mathbf{x}(t)$ (see Fig. 2.1E).

2.2.2 Applications to Generic Cortical Microcircuit Models

This section examines the computational aspects of feedback in recurrent circuits of spiking neurons that are based on data from cortical microcircuits. The dynamics of these circuits is substantially more complex than the dynamics of circuits described by equ. (2.6), since it is based on action potentials (spikes) rather than firing rates. Hence one can expect at best that the temporal dynamics of firing rates in these circuits of spiking neuron is qualitatively similar to that of circuits described by (2.6).

The preceding theoretical results imply that it is possible for dynamical systems to carry out computations with persistent memory without acquiring all the computational disadvantages of the chaotic regime, where the memory capacity of the system is dominated by noise. Feedback units can create selective "loopholes" into the fading memory dynamics of a dissipative system, that can only be activated by specific patterns in the input or circuit dynamics. In this way the potential content of persistent memory can be controlled by feedback units that have been trained to recognize such patterns. This feedback may arise from a few neurons within the circuit, or from neurons within a larger feedback loop. The task to approximate a suitable feedback function K is less difficult than it may appear on first sight, since it suffices in many cases to approximate a *linear* feedback function. The reason is that sufficiently large generic cortical microcircuit models have an inherent kernel property [36], in the sense of machine learning [60]. This means that a large reservoir of diverse nonlinear responses to current and recent input patterns is automatically produced within the recurrent circuit. In particular, nonlinear combinations of variables $a, b, c, \ldots$ (that may result from the circuit input or internal activity) are automatically computed at internal nodes of the circuit. Consequently numerous low degree polynomials in these variables $a, b, c, \ldots$ can be approximated by *linear* combinations of outputs of neurons from the recurrent circuit. An example of this effect is demonstrated in Fig. 2.3G, where it is shown that the product of firing rates $r_3(t)$ and $r_4(t)$ of two independently varying afferent spike train inputs can be approximated quite well by a linear readout neuron. The kernel property of biologically realistic cortical microcircuit models is apparently supported by the fact that these circuits have many additional nonlinearities besides those that appear in the equations (2.1), (2.2), (2.6).

One formal difference between neurons in the mean field model (2.6) and more realistic models for spiking neurons is that the input to a neuron of the latter type consists of postsynaptic potentials, rather than of firing rates. Hence the time-varying input $\mathbf{x}(t)$ to a readout neuron is in this section not a vector of time-varying firing rates, but a smoothed version of the spike trains of all presynaptic neurons. This smoothing is achieved through application of a linear filter with an exponentially decaying kernel, whose time constant of 30 ms models time constants of receptors and postsynaptic membrane of a readout neuron in a qualitative fashion. Thus, if $\mathbf{w}$ is a vector of synaptic weights, then $\mathbf{w} \cdot \mathbf{x}(t)$ models the impact of the firing activity of presynaptic neurons on the membrane potential of a readout neuron.

The following refers to those neurons where the weights of synaptic connections from neurons within the circuit are adapted for a specific computational task (rather than chosen randomly from distributions that are based on biological data, like for all other synapses in the circuit) as *readout neurons*. The output of a readout neuron was modeled in most of the simulations simply by a weighted sum $\mathbf{w} \cdot \mathbf{x}(t)$ of the previously described vector $\mathbf{x}(t)$. Such output can be interpreted as the time-varying firing rate of a readout neuron. However I show in Fig. 2.3 that these readout neurons can (with a moderate loss in performance) also be modeled by spiking neurons, exactly like the other neurons in the simulated circuit. This demonstrates that not only those circuits that receive feedback from external readout neurons, but also generic recurrent circuits in which a few neurons have been trained for a specific task, acquire computational capabilities for real-time processing that are not restricted to computations with fading memory.

Theorem 2.2.2 suggests that the training of a few of its neurons enables generic neural circuits to employ persistent internal states for state-dependent processing of online input streams. Previous models for non-fading memory in neural circuits [61, 62, 41, 63] proposed that it is implemented through low-dimensional attractors in the circuit dynamics. These attractors tend to freeze or entrain the whole state of the circuit, and thereby shut it off from the online input stream (although independent local attractors could emerge in local subcircuits under some conditions [62]). In contrast, the generation of non-fading memory through a few trained neurons does not entail that the dynamics of the circuit is dominated by their persistent memory states. For example, when a readout neuron gives during some time interval a constant feedback $K(\mathbf{x}(t)) = c$, this only constrains the circuit state $\mathbf{x}(t)$ to remain in the sub-manifold $\{\mathbf{x} : K(\mathbf{x}) = c\}$ of its high-dimensional state space. This sub-manifold is in general high-dimensional. In particular, if $K(\mathbf{x})$ is a linear function $\mathbf{w} \cdot \mathbf{x}$, which often suffices as I will show, the dimensionality of the sub-manifold $\{\mathbf{x} : K(\mathbf{x}) = c\}$ differs from the dimension of the full state space only by 1. Hence several such sub-manifolds have in general a high-dimensional intersection, and their intersection still leaves sufficiently many degrees of freedom for the circuit state $\mathbf{x}(t)$ to also absorb continuously new information from online input streams. These sub-manifolds are in general not attractors in a strict mathematical sense. Rather, their effective attraction property (or noise-robustness) results from the subsequently described training process ("teacher forcing"). This training process produces weights $\mathbf{w}$ which have the property that the resulting feedback $\mathbf{w} \cdot \tilde{\mathbf{x}}(t)$ moves a trajectory of circuit states that goes through states $\tilde{\mathbf{x}}(t)$ in the neighborhood of the sub-manifold $\{\mathbf{x} : K(\mathbf{x}) = c\}$ closer to this sub-manifold.

Generic cortical microcircuit models were simulated consisting of 600 integrate-and-fire (I&F) neurons (for Fig. 2.3, 2.4), and circuits consisting of 600 Hodgkin-Huxley (HH) neurons (for Fig. 2.5), in either case with a rather high level of noise that reflects experimental data on the high conductance state in vivo [64]. These circuits were not constructed for any particular computational task. In particular, sparse synaptic connectivity between neurons was generated (with a biologically realistic bias towards short connections) by a probabilistic rule. Synaptic parameters were chosen randomly from distributions that depend on the type of pre- and postsynaptic neurons (in accordance with empirical data from [65], [66]). More precisely, biologically realistic models were used for dynamic synapses whose individual mixture of paired-pulse depression and facilitation (depending on the type of pre- and postsynaptic neuron) was based on these data. It has previously been shown in [36],[34] that the presence of such dynamic synapses extends the time span of the inherent fading memory of the circuit. However the computational tasks that are considered in this paper require, apart from a non-fading memory, only a fading memory with a rather short time span (in order to make the estimation of the current firing rate of input spike trains feasible). Therefore the biologically more realistic dynamic synapses could be replaced in this model by simple static synapses, without a change in the performance of the circuit for the subsequently considered tasks. All details of the simulated microcircuit models can be found in section 2.4.1. Details of the subsequently discussed computer experiments are given in sections 2.4.2 - 2.4.4.

I tested 3 different types of computational tasks for generic neural circuits with feedback. The same neural circuit can be used for each task, only the organization of input- and output streams needs to be chosen individually (see Fig. 2.2). The following procedure was applied to train readout neurons, i.e. to adjust the weights of synaptic connections from neurons in the circuit to readout neurons for specific computational tasks (while leaving all other parameters of the generic microcircuit model unchanged):

- First those readout neurons were trained that provide feedback, then the other readout neurons.
- During the training of readout neurons that provide feedback, their actual feedback was replaced by a *noisy* version of their target output ("teacher forcing").
- Each readout neuron was trained by linear regression to output at any time t a particular target value $f(t)$. Linear regression was applied to a set of data points of the form $\langle \mathbf{x}(t), f(t) \rangle$ for many time points t, where $\mathbf{x}(t)$ is a smoothed version of the spike trains of presynaptic neurons (as defined before).

Note that teacher forcing with noisy versions of target feedback values trains these readouts to correct errors resulting from imprecision in their preceding feedback (rather than amplifying errors). This training procedure is responsible for the robustness of the dynamics of the resulting closed-loop circuits, in particular for the "attractor" properties of the effectively resulting high-dimensional attractors.

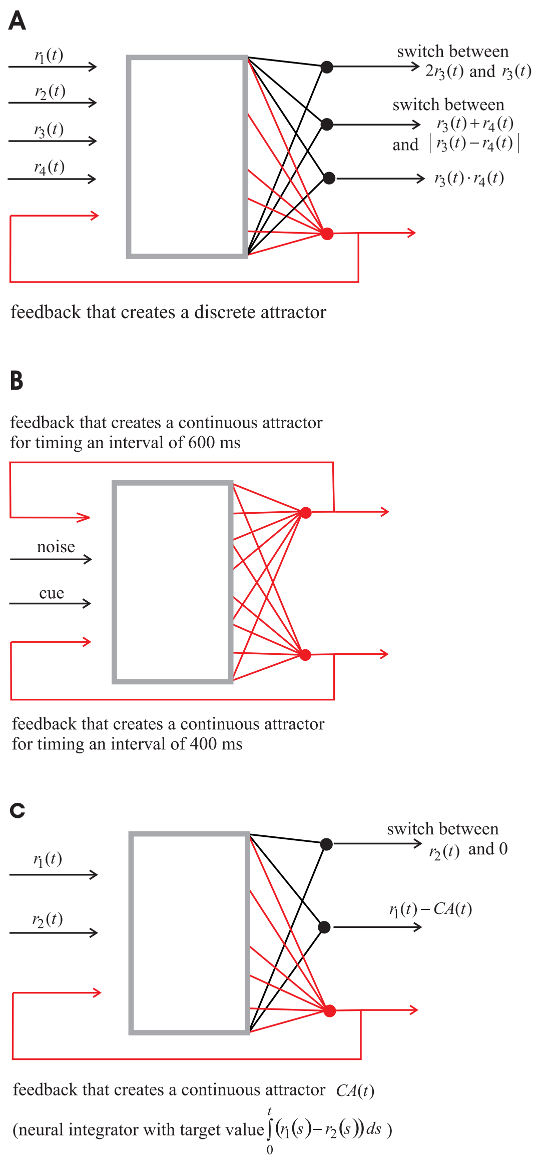
A
$r_1(t)$
$r_2(t)$
$r_3(t)$
$r_4(t)$
switch between $2r_3(t)$ and $r_3(t)$
switch between $r_3(t)+r_4(t)$ and $|r_3(t)-r_4(t)|$
$r_3(t)\cdot r_4(t)$
feedback that creates a discrete attractor
B
feedback that creates a continuous attractor for timing an interval of 600 ms
noise
cue
feedback that creates a continuous attractor for timing an interval of 400 ms
C
$r_1(t)$
$r_2(t)$
switch between $r_2(t)$ and 0
$r_1(t)-CA(t)$
feedback that creates a continuous attractor $CA(t)$
(neural integrator with target value $\int_0^t (r_1(s)-r_2(s))\,ds$)

Figure 2.2: Organization of input- and output streams for the 3 computational tasks considered in the computer simulations. Each input stream consisted of multiple spike trains, that provided synaptic inputs to individually chosen subsets of neurons in the recurrent circuit (which is indicated by a gray rectangle). In **(A)** and **(C)** these input streams consisted of multiple Poisson spike trains with a time-varying firing rate $r_i(t)$. In **(B)** the input consisted of a burst ("cue") in one spike train (which marks the beginning of a time interval) and independent Poisson spike train ("noise") in the other input channels. The actual outputs of the readouts (that were trained individually for each computational task) in panels **(A, B, C)** is shown in Figures 2.3, 2.4, 2.5.

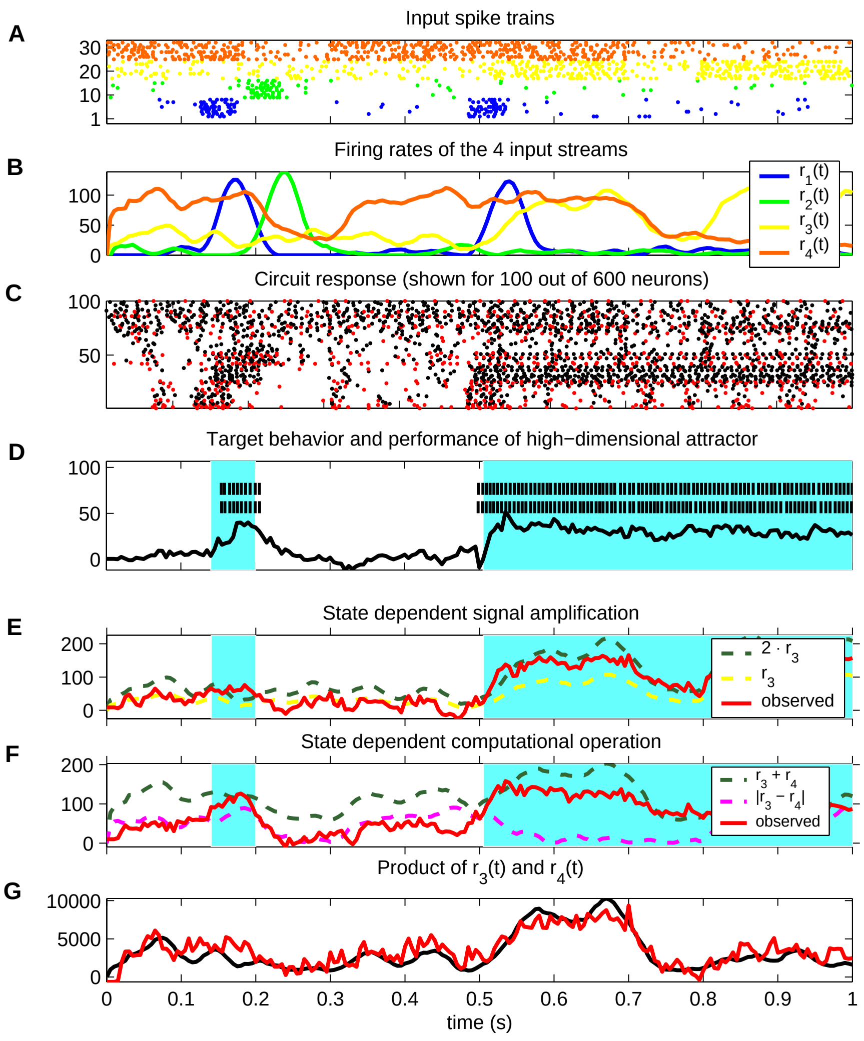

Figure 2.3: State-dependent real-time processing **(A)** 4 input streams, consisting each of 8 spike trains generated by Poisson processes with randomly varying rates $r_i(t), i = 1, \ldots, 4$ (rates plotted in **(B)**; all rates are given in Hz). The 4 input streams and the feedback were injected into disjoint sets of neurons in the circuit. **(C)** Resulting firing activity of 100 out of the 600 I&F neurons in the circuit. Spikes from inhibitory neurons marked in red. **(D)** Target activation times of the high-dimensional attractor (blue shading), spike trains of 2 of the 8 I&F neurons that were trained to create the high-dimensional attractor by sending their output spike trains back into the circuit, and average firing rate of all 8 neurons (lower trace). **(E and F)** Performance of linear readouts that were trained to switch their real-time computation task in dependence of the current state of the high-dimensional attractor: output $2 \cdot r_3(t)$ instead of $r_3(t)$ if the high-dimensional attractor is on (E), output $r_3(t) + r_4(t)$ instead of $|r_3(t) - r_4(t)|$ if the high-dimensional attractor is on (F). **(G)** Performance of linear readout that was trained to output $r_3(t) \cdot r_4(t)$, showing that another linear readout from the same circuit can simultaneously carry out nonlinear computations that are invariant to the current state of the high-dimensional attractor.

In the first computer experiment, readout neurons were trained to turn a high-dimensional attractor on or off (Fig. 2.3D), in response to bursts in 2 of the 4 independent input spike trains. More precisely, 8 neurons were trained to represent in their firing activity at any time the information in which of the input streams 1 or 2 a burst had most recently occurred. If it had occurred most recently in stream 1, they were trained to fire at 40 Hz, and if a burst had occurred most recently in input stream 2, they were trained not to fire. Hence these neurons were required to represent the non-fading state of a simple FSM, demonstrating in an example the computational capabilities predicted by Theorem 2.2.2. Fig. 2.3G demonstrates that the circuit retains its kernel property inspite of the feedback injected into the circuit by these readouts. But beyond the emulation of a simple FSM, the resulting generic cortical microcircuit is able to combine information stored in the current state of the FSM with new information from the online circuit input. For example, Fig. 2.3E shows that other readouts from the same circuit can be trained to amplify their response to specific inputs if the high-dimensional attractor is in the "on"-state. Readouts can also be trained to change the function that they compute if the high-dimensional attractor is in the on-state (Fig. 2.3F). This provides an example for an online reconfigurable circuit. The readout neurons that provide feedback had been modeled in this computer simulation like the other neurons in the circuit: by I&F neurons with in-vivo like background noise. Hence they can be viewed equivalently as neurons *within* an otherwise generic circuit.

Another difficult problem in computational neuroscience is to explain how neural circuits can implement a parametric memory, i.e. how they can hold and update an *analog* value, that may represent for example an intended eye-position that a neural integrator computes from a sequence of eye-movement commands [67], an estimate of elapsed time [37], or accumulated sensory evidence [42]. Various designs have been proposed for parametric memory in recurrent circuits, where continuous attractors (also referred to as line attractors) hold and update an analog value. But these approaches are inherently brittle [63], and have problems in dealing with high noise or online circuit inputs. On the other hand Fig. 2.4 shows that dedicated circuit constructions are not necessary, since feedback from readout neurons in *generic* cortical microcircuits models can also create high-dimensional attractors that hold and update an *analog* value for behaviorally relevant time spans. In fact, due to the high-dimensional character of the resulting high-dimensional attractors, two such analog values can be stored and updated independently (Fig. 2.4C,D), even within a fairly small circuit. In this example the readouts that provide feedback were simply trained to increase or reduce their feedback at each time point. Note that the resulting circuit activity is qualitatively consistent with recordings from neurons in cortex and striatum during reward expectation [38],[39],[40]. A similar ramp-like rise and fall of ac-

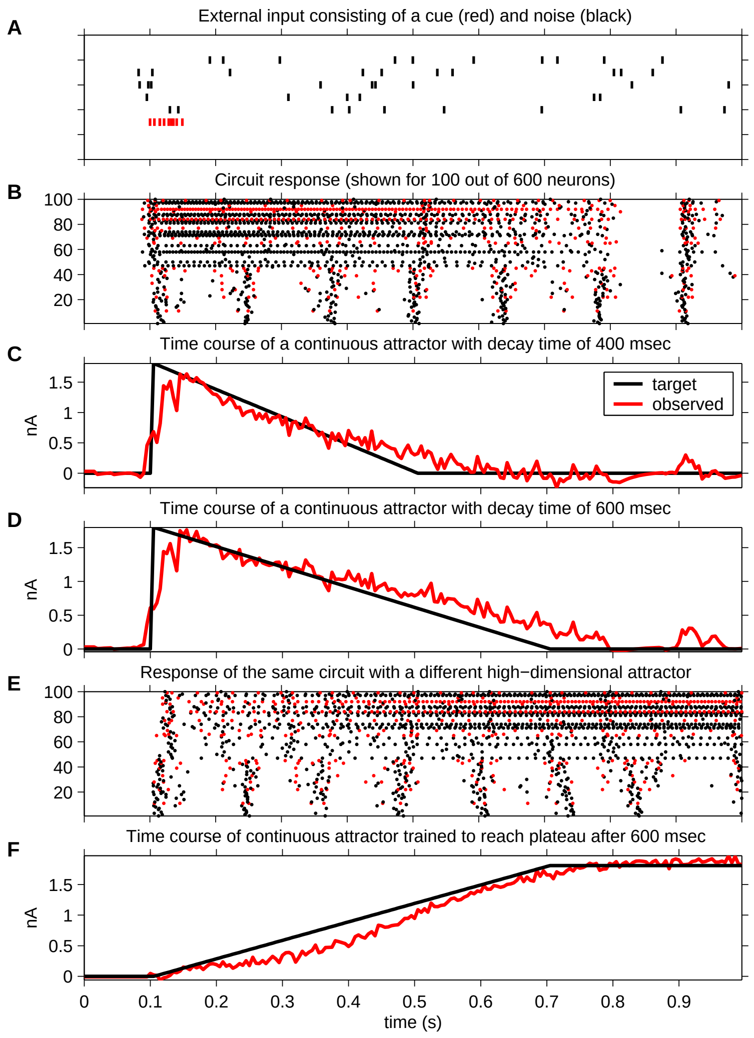

Figure 2.4: Representation of time for behaviorally relevant time spans in a generic cortical microcircuit model. **(A)** Afferent circuit input, consisting of a cue in one channel (red) and random spikes (freshly drawn for each trial) in the other channels. **(B)** Response of 100 neurons from the same circuit as in Fig. 2.3, which has here two co-existing high-dimensional attractors. The autonomously generated periodic bursts with a periodic frequency of about 8 Hz are not related to the task, and readouts were trained to become invariant to them. **(C and D)** Feedback from two linear readouts that were simultaneously trained to create and control two high-dimensional attractors. One of them was trained to decay in 400 ms (C), and the other in 600 ms (D) (scale in nA is the average current injected by feedback into a randomly chosen subset of neurons in the circuit). **(E)** Response of the same neurons as in (B), for the same circuit input, but with feedback from a different linear readout that was trained to create a high-dimensional attractor that increases its activity and reaches a plateau 600 ms after the occurrence of the cue in the input stream. (**F**) Feedback from the linear readout that creates this continuous high-dimensional attractor.

tivity as shown in panels C, D, F has also been recorded in neurons of posterior parietal cortex of the macaque in experiments were the monkey had been trained to classify the duration of elapsed time [37]. The high-dimensionality of the continuous attractors in this model makes it feasible to constrain the circuit state to stay simultaneously in more than one continuous attractor, thereby making it in principle possible to encode complex movement plans that require specific temporal relationships between individual motor commands.

This model for parametric memory in cortical circuits is consistent with high noise: Fig. 2.5G shows the typical trial-to-trial variability of a neuron in the simulated circuit of HH neurons with in-vivo like background noise. It qualitatively matches the "wide diversity of neural firing drift patterns in individual fish at all states of tuning" that was observed in the horizontal occulomotor neural integrator in goldfish [6], and the large trial-to-trial variability of neurons in prefrontal cortex of monkeys reported in [38]. In addition, this model is consistent with the surprising plasticity that has been observed even in quite specialized neural integrators [6], since continuous attractors can be created or modified in this model by changing just a few synaptic weights of neurons that are immediately involved. It does not require the presence of long-lasting postsynaptic potentials, NMDA-receptors, or other specialized details of biological neurons or synapses, although their inclusion in the model is likely to provide additional temporal stability [41]. Rather it points to complementary organizational mechanisms on the circuit level, that are likely to enhance the controllability and robustness of continuous attractors in neural circuits. The robustness of this learning-based model can be traced back to the fact that readout neurons can be trained to correct undesired circuit responses resulting from errors in their previous feedback. Furthermore such error correction is not restricted to linear computational operations, since the previously demonstrated kernel property of these generic circuits allows even linear neurons to implement complex nonlinear control strategies through their feedback. As an example I demonstrate in Fig. 2.5 that even under biologically realistic high noise conditions a linear readout can be trained to update a continuous attractor[1] (Fig. 2.5D), to filter out input activity[2] during certain time intervals in dependence of the current state of the continuous attractor (Fig. 2.5E), or to combine[3] the time-varying analog variable encoded by

[1]This readout was trained to output at any time t an approximation $CA(t)$ of the integral $\int_0^t (r_1(s) - r_2(s))ds$ over the difference of both input rates. Feedback values were injected as input currents into a randomly chosen subset of neurons in the circuit. Scale in nA shows average strength of feedback currents (also for Fig. 2.5 panel H).

[2]This readout was trained to output 0 as long as $CA(t)$ stayed below 0.83 nA, and to output $r_2(t)$ once $CA(t)$ had crossed this threshold, as long as $CA(t)$ stayed above 0.66 nA (i.e., in this test run during the shaded time periods).

[3]This readout was trained to output $r_1(t) - CA(t)$, i.e. a combination of external and internal

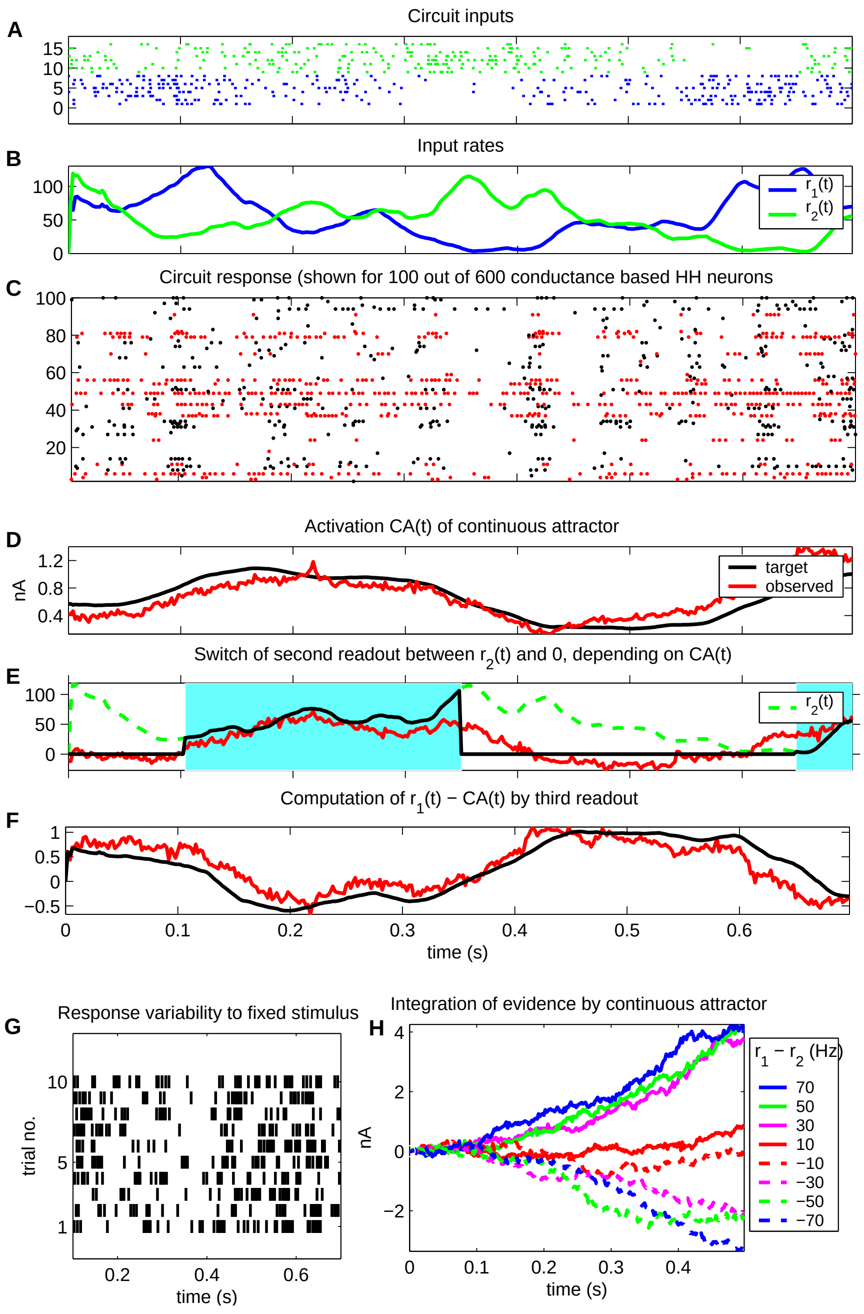

Figure 2.5: Analog real-time computation. **(A)** Two input streams and **(B)** their firing rates $r_1(t), r_2(t)$. **(C)** Resulting firing activity of 100 neurons in the circuit. **(D)** Performance of a neural integrator, generated by feedback. **(E)** Switching readout. **(F)** Performance of linear readout trained to output $r_1(t) - CA(t)$. **(G)** Response of a randomly chosen neuron in the circuit for 10 repetitions of the same experiment (with input spike trains generated by Poisson processes with the same time-course of firing rates), showing biologically realistic trial-to-trial variability. **(H)** Activity traces of a continuous attractor as in (D), but in 8 different trials for 8 different fixed values of r_1 and r_2 (shown on the right). For further details, see text.

the current state $CA(t)$ of the continuous attractor with a time-varying variable $r_1(t)$ that is delivered by an online spike input. Hence intention-based information processing [43] and other tasks that involve a merging of external inputs and internal state information can be implemented in this way. Fig. 2.5C shows that a high-dimensional attractor need not entrain the firing activity of neurons in a drastic way, since it just restricts the high-dimensional circuit dynamics $\mathbf{x}(t)$ to a slightly lower dimensional manifold of circuit states $\mathbf{x}(t)$ that satisfy $\mathbf{w} \cdot \mathbf{x}(t) = f(t)$ for the current target output $f(t)$ of the corresponding linear readout. On the other hand Fig. 2.5E shows that the activity level $CA(t)$ of the high-dimensional attractor can nevertheless be detected by other linear readouts, and can simultaneously be combined in a nonlinear manner with a time-varying variable $r_2(t)$ from one afferent circuit input stream, while remaining invariant to the other afferent input stream.

Finally, the same generic circuit also provides a model for the integration of evidence for decision making that is compatible with in-vivo like high noise conditions. Fig. 2.5H depicts the time course of the same neural integrator as in panel D, but here for the case where the rates r_1, r_2 of the 2 input streams assume in 8 trials 8 different constant values after the first 100 ms (while assuming a common value of 65 Hz during the first 100 ms). The resulting time course of the continuous attractor is qualitatively similar to the meandering path towards a decision threshold that has been recorded from neurons in area LIP where firing rates represent temporally integrated evidence concerning the dominating direction of random dot movements (see Fig. 5A in [42]).

Control experiments (see Fig. 2.6) show that the feedback is essential for the performance of the circuit for these computational tasks.

2.3 Discussion

A theoretically founded model is presented in this chapter for real-time computations on complex input streams with persistent internal states in generic cortical microcircuits. This model does not require a handcrafted circuit structure or biologically unrealistic assumptions such as symmetric weight distributions, static synapses that do not exhibit pair-pulsed depression or facilitation, or neuron models with low levels of noise that are not consistent with data on in-vivo conditions. This model only requires the assumption that adaptive procedures (synaptic plasticity) in generic neural circuits can approximate linear regression. Furthermore, in contrast to classical learning paradigms for attractor neural networks, it is here not required that a large fraction of synaptic parameters in the circuit are

variables, at any time t (both r_1 and CA normalized into the range $[0, 1]$).

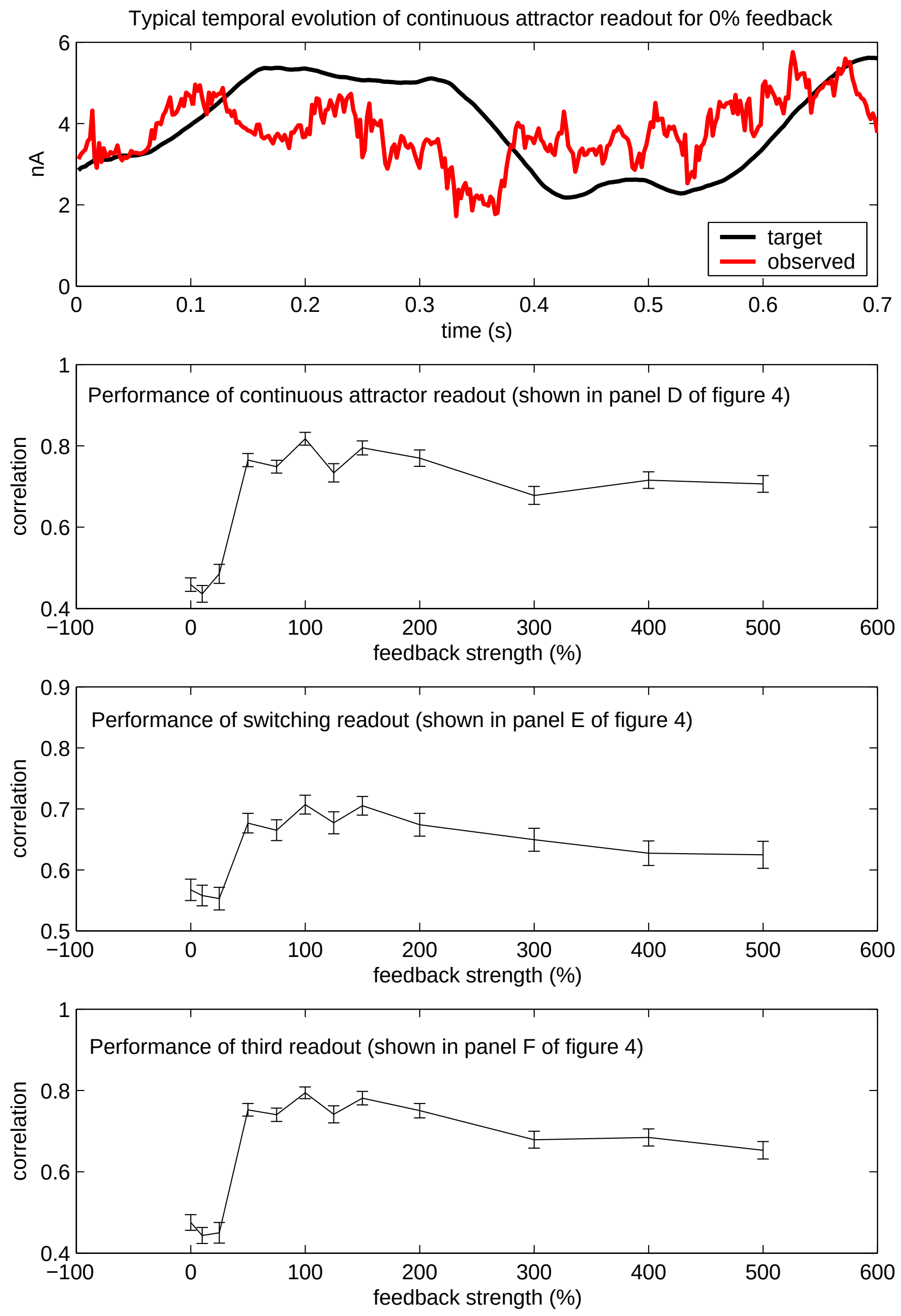

Figure 2.6: Evaluation of the dependence of the performance of the circuit in Fig. 2.5 on the feedback strength (i.e., on the mean amplitude of current injection from the readout back into neurons in the circuit). For each feedback strength that was evaluated, the readouts were trained and tested for this feedback strength like for the preceding experiments. Error bars in B-D denote standard error. These control experiments show that the feedback is essential for the performance of the circuit.

changed when a new computational task is introduced, or a new item is stored in working memory. Rather, it suffices if those neurons that provide the circuit output and a few neurons that provide feedback are subject to synaptic plasticity. Such minimal circuit modifications have the advantage that thereby created attractors of the circuit dynamics are high-dimensional. It is shown that the circuit state can be simultaneously in several of such high-dimensional attractors, and still retain sufficiently many degrees of freedom to absorb and process new information from online input streams. In particular, I have shown in Fig. 2.3 and 2.5 how bottom-up processing can be reconfigured in dependence of discrete internal states (implemented through high-dimensional attractors) by turning certain input channels on or off, and by changing the computational operations that are applied to input variables. Furthermore I have shown in Fig. 2.5 that analog variables, which are extracted from an online input stream, can be combined in real-time computations with analog variables that are stored in high-dimensional continuous attractors. This provides in particular a model for the implementation of intention-based information processing [43] in cortical microcircuits.

It remains open how learning signals can induce neurons in a biological organism to compute specific linear feedback functions. But at least this this problem is reduced to the feasibility of perceptron-like learning (or more abstractly: to linear regression) for single neurons. Subsequent research will have to determine whether these learning requirements (which can partially be reduced to spike-timing dependent plasticity [68]) can be justified on the basis of results on unsupervised learning and reinforcement learning [69] in biological organisms.

Whereas it was previously already known that one can construct specific circuits that have universal computational capabilities for real-time computing on analog input streams, Theorems 2.2.1 and 2.2.2 of this chapter imply that a large variety of dynamical systems (in particular generic cortical microcircuits) can acquire through feedback such universal capabilities for computations that map time-varying inputs to time-varying outputs. It should be noted that these universal computational capabilities differ from the well known but much weaker universal approximation property of feedforward neural networks (see [56]), since not only the static output of an arbitrary continuous static function is approximated, but the dynamic response of arbitrary differential equations of higher order to time-varying inputs.

The theoretical results of this chapter also provide an explanation for the astounding computational capability and flexibility of echo state networks [14]. In addition they can be used to analyze computational aspects of feedback in other biological dynamical systems besides neural circuits. Several such systems, for example genetic regulatory networks, are known to implement complex maps from time-varying input streams (e.g. external signals) onto time-varying outputs (e.g.

transcription rates). But little is known about the way in which these maps are implemented. Whereas feedback in biological dynamical systems is usually only analyzed and modeled from the perspective of control, I propose that an analysis of its computational aspects is likely to yield a better understanding of the computational capabilities of such systems.

2.4 Materials and Methods

2.4.1 Details of the Cortical Microcircuit Models

This section complements the general description of the simulated cortical microcircuit models from section 2.2.2, providing in particular all missing data that are needed to reproduce the simulation results. The original code that was used for these simulations is online available from http://www.lsm.tugraz.at/research/index.html.

Each circuit consisted of 600 neurons, which were placed on the integer grid points of a $5 \times 5 \times 24$ grid. 20% of these neurons were randomly chosen to be inhibitory. 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 $\lambda = 3$). Depending on whether the pre- or postsynaptic neuron were excitatory (E) or inhibitory (I), the value of C was set according to [66] to 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II), yielding an average of 10900 synapses for the chosen circuit size. External inputs and feedbacks from readouts were connected to populations of neurons in the circuit with randomly chosen connection strengths.

I&F neurons: A standard leaky-integrate-and-fire neuron model was used, where the membrane potential V_m of a neuron is given by:

$$\tau_m \frac{dV_m}{dt} = -(V_m - V_{resting}) + R_m \cdot (I_{syn} + I_{inject} + I_{noise}) \tag{2.7}$$

where t_m is the membrane time constant (30 ms), which subsumes the time constants of synaptic receptors as well as the time constant of the neuron membrane. Other parameters: absolute refractory period 3 ms (excitatory neurons), 2 ms (inhibitory neurons), threshold 15 mV (for a resting membrane potential $V_{resting}$, assumed to be 0), reset voltage drawn uniformly from the interval [13.8, 14.5 mV] for each neuron, input resistance R_m, 1 MΩ, constant non-specific background current I_{inject} uniformly drawn from the interval [13.5 nA, 14.5 nA] for each neuron, an additional time-varying noise input current I_{noise} was drawn every 5 ms from a Gaussian distribution with mean 0 and SD chosen for each neuron randomly from the uniform distribution over the interval [4.0 nA, 5.0 nA]. For each simulation, the initial condition of each I&F neuron, i.e., its membrane voltage at time $t = 0$, was drawn randomly (uniform distribution) from the interval [13.5 mV, 14.9 mV]. Finally, $I_{syn}(t)$ is the sum of input currents supplied by the explicitly modeled synapses.

HH-neurons: Single compartment HH neuron models were used with passive and active properties modeled according to [70, 71]. The membrane potential was modeled by

$$C_m \frac{dV}{dt} = -g_l(V - E_l) - I_{Na} - I_{Kd} - I_M - \frac{1}{a} I_{noise} - I_{syn} \ , \tag{2.8}$$

where $C_m = 1\ \mu F/cm^2$ is the specific membrane capacitance, $g_L = 0.045\ mS/cm^2$ is the leak conductance density, $E_L = -80\ mV$ is the leak reversal potential, and $I_{syn}(t)$ is the input current supplied by explicitly modeled synapses (see the definition below). The membrane area a of the neuron was set to be 34636 μm^2 as in [70]. The term $I_{noise}(t)$ (see the precise definition below) models smaller background input currents from a large number of more distal neurons, causing a depolarization of the membrane potential and a lower input resistance commonly referred to as 'high conductance state' (for a review see [64]).

In accordance with experimental data on neocortical and hippocampal pyramidal neurons ([72, 73, 74, 75]) the active currents in the HH neuron model comprise a voltage dependent Na^+ current I_{Na} ([76]) and a delayed rectifier K^+ current I_{Kd} ([76]). For excitatory neurons a non-inactivating K^+ current I_M ([77]) responsible for spike frequency adaption was included in the model.

The voltage-dependent Na^+ current was modeled by:

$$I_{Na} = \overline{g}_{Na} m^3 h (V - E_{Na})$$

$$\frac{dm}{dt} = \alpha_m(V)(1 - m) - \beta_m(V) m$$

$$\frac{dh}{dt} = \alpha_h(V)(1 - h) - \beta_h(V) h$$

$$\alpha_m = \frac{-0.32(V - V_T - 13)}{exp[-(V - V_T - 13)/4] - 1}$$

$$\beta_m = \frac{0.28(V - V_T - 40)}{exp[(V - V_T - 40)/5] - 1}$$

$$\alpha_h = 0.128\ exp[-(V - V_T - V_S - 17)/18]$$

$$\beta_h = \frac{4}{1 + exp[-(V - V_T - V_S - 40)/5]}$$

where $V_T = -63\ mV$, and the inactivation was shifted by 10 mV toward hyperpolarized values ($V_S = -10\ mV$) to reflect the voltage dependence of Na^+ currents in neocortical pyramidal cells [78]. The peak conductance densities for the I_{Na} current was chosen to be $500pS/\mu m^2$.

The delayed rectifier K^+ current was modeled by:

$$I_{Kd} = \overline{g}_{Kd} n^4 (V - E_K)$$

$$\frac{dn}{dt} = \alpha_n(V)(1 - n) - \beta_n(V)n$$

$$\alpha_n = \frac{-0.032(V - V_T - 15)}{exp[-(V - V_T - 15)/5] - 1}$$

$$\beta_n = 0.5\ exp[-(V - V_T - 10)/40]$$

The peak conductance densities for the I_{Kd} current was chosen to be $100pS/\mu m^2$.

The noninactivating K^+ current was modeled by:

$$I_M = \overline{g}_M n (V - E_K)$$

$$\frac{dn}{dt} = \alpha_n(V)(1 - n) - \beta_n(V)n$$

$$\alpha_n = \frac{0.0001(V + 30)}{1 - exp[-(V + 30)/9]}$$

$$\beta_n = \frac{-0.0001(V + 30)}{1 - exp[(V + 30)/9]}$$

The peak conductance density for the I_M current was chosen to be $5pS/\mu m^2$.

For each simulation the initial condition of each neuron, i.e. the membrane voltage at time $t = 0$, was drawn randomly (uniform distribution) from the interval [-70, -60] mV.

The total **synaptic background current**, $I_{noise}(t)$, was a sum of two independent currents:

$$I_{noise}(t) = g_e(t)(V - E_e) + g_i(t)(V - E_i) \ ,$$

where $g_e(t)$ and $g_i(t)$ are time-dependent excitatory and inhibitory conductances. The values of respective reversal potentials were $E_e = 0 \ mV$ and $E_i = -75 \ mV$.

The conductances $g_e(t)$ and $g_i(t)$ were modeled according to [70] as a one-variable stochastic process similar to an Ornstein-Uhlenbeck process:

$$\frac{dg_e(t)}{dt} = -\frac{1}{\tau_e}[g_e(t) - g_{e0}] + \sqrt{D_e}\chi_1(t)$$

$$\frac{dg_i(t)}{dt} = -\frac{1}{\tau_i}[g_i(t) - g_{i0}] + \sqrt{D_i}\chi_2(t)$$

where $g_{e0} = 0.012 \ \mu S$ and $g_{i0} = 0.057 \ \mu S$ are average conductances, $\tau_e = 2.7 \ ms$ and $\tau_e = 10.5 \ ms$ are time constants, $D_e = 0.0067 \ \mu S^2/s$ and $D_i = 0.0083 \ \mu S^2/s$ are noise diffusion constants, $\chi_1(t)$ and $\chi_2(t)$ are Gaussian white noise of zero mean and unit standard deviation.

Since these stochastic processes are Gaussian, they can be integrated by an exact update rule:

$$g_e(t + \Delta t) = g_{e0} + [g_e(t) - g_{e0}] \ exp(-\Delta t/\tau_e) + A_e \ N_1(0, 1)$$

$$g_i(t + \Delta t) = g_{i0} + [g_i(t) - g_{i0}] \ exp(-\Delta t/\tau_i) + A_i \ N_2(0, 1)$$

where $N_1(0, 1)$ and $N_2(0, 1)$ are normal random numbers (zero mean, unit SD) and A_e and A_i are amplitude coefficients given by:

$$A_e = \sqrt{\frac{D_e \ \tau_e}{2}\left[1 - exp\left(\frac{-2\Delta t}{\tau_e}\right)\right]}$$

$$A_i = \sqrt{\frac{D_i \ \tau_i}{2}\left[1 - exp\left(\frac{-2\Delta t}{\tau_i}\right)\right]} \ .$$

According to [70], this model captures the spectral and amplitude characteristics of the input conductances of a detailed biophysical model of a neocortical pyramidal cell that was matched to intracellular recordings in cat parietal cortex *in vivo*. Furthermore the ratio of the average contributions of excitatory and inhibitory background conductances was chosen to be 5 in accordance with experimental studies during sensory responses [79, 80, 81]. The maximum conductances of the synapses were chosen from a Gaussian distribution with a SD of 70% of its mean (with negative values replaced by values chosen from an uniform distribution between 0 and two times the mean).

The (short term) **dynamics of synapses** was modeled according to the model proposed in [65], 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 (see supplementary information). This model predicts the amplitude A_k of the EPSC for the k^{th} spike in a spike train with interspike intervals $\Delta_1, \Delta_2, \ldots, \Delta_{k-1}$ through the equations

$$\begin{aligned}
A_k &= w \cdot u_k \cdot R_k \\
u_k &= U + u_{k-1}(1-U)exp(-\Delta_{k-1}/F) \\
R_k &= 1 + (R_{k-1} - u_{k-1}R_{k-1} - 1)exp(-\Delta_{k-1}/D)
\end{aligned}$$

with hidden dynamic variables $u \in [0,1]$ and $R \in [0,1]$ whose initial values for the first spike are $u_1 = U$ and $R_1 = 1$ (see [82] for a justification of this version of the equations, which corrects a small error in [65]).

The postsynaptic current for the k^{th} spike in a presynaptic spike train, that had been generated at time t_k, is modeled for $t \geq t_k + \Delta$ (where Δ is the transmission delay) by $A_k \exp(-(t - t_k - \Delta)/\tau_s)$ with $\tau_s = 3\,\text{ms}$ ($\tau_s = 6\,\text{ms}$) for excitatory (inhibitory) synapses. The transmission delays Δ between neurons were chosen uniformly to be 1.5 ms for EE-connections, and 0.8 ms for the other connections. The total **synaptic input current** $I_{syn}(t)$ was modeled by the sum of these currents for all synapses onto a neuron.

Synaptic parameters: Depending on whether a and b were excitatory (E) or inhibitory (I), the mean values of the three parameters U, D, F (with D,F expressed in seconds, s) were chosen according to [66] to be .5, 1.1, .05 (EE), .05, .125, 1.2 (EI), .25, .7, .02 (IE), .32, .144, .06 (II). The SD of each of these parameters was chosen to be 50% of its mean. The mean of the scaling parameter w (in nA) was chosen to be 70 (EE), 150 (EI), -47 (IE), -47 (II). The SD of the parameter w was chosen to be 70% of its mean and was drawn from a gamma distribution. In the case of input synapses the parameter w had a value of 70 nA if projecting onto a excitatory neuron and -47 nA if projecting onto an inhibitory neuron.

The synaptic weights $\mathbf{w}$ of **readout neurons** were computed by linear regression to minimize the mean squared error $(\mathbf{w}\cdot\mathbf{x}(t)-f(t))^2$ with regard to a specific target output function $f(t)$ (which is described for each case in the text or figure legends) for a series of randomly generated time-varying circuit input streams $\mathbf{u}(t)$ of length up to 1 second. Up to 200 such time-varying input streams $\mathbf{u}(t)$ were used for training, amounting to at most 200 seconds of simulated biological time for training the readouts.

The performance of trained readouts was evaluated by measuring the correlation between $\mathbf{w}\cdot\mathbf{x}(t)$ and the target function $f(t)$ during separate testing episodes where the circuit received new input streams $\mathbf{u}(t)$ (that were generated by the same random process as the training inputs).

All simulations were carried out with the software package CSIM [83], which is freely available from http://www.lsm.tugraz.at. It uses a C^{++}-kernel with Matlab interfaces for input generation and data analysis. As simulation time step I chose 0.5 ms.

2.4.2 Technical Details to Figure 2.3

4 randomly generated test input streams, each consisting of 8 spike trains (see Fig. 2.3A), were injected into 4 disjoint (but interconnected) subsets of $5\times5\times5=125$ neurons in the circuit consisting of 600 neurons. Feedbacks from readouts were injected into the remaining 100 neurons of the circuit. The set of 100 neurons for which the firing activity is shown in Fig. 2.3C contained 20 neurons from each of the resulting 5 subsets of the circuit.

Generation of input streams for training and testing: The time-varying firing rate $r_i(t)$ of the 8 Poisson spike trains that represented input stream i was chosen as follows. The baseline firing rate for streams 1 and 2 (see the lower half of Fig. 2.3A) was chosen to be 5 Hz, with randomly distributed bursts of 120 Hz for 50 ms. The rates for the Poisson processes that generated the spike trains for input streams 3 and 4 were periodically drawn randomly from the two options 30 Hz and 90 Hz. The actual firing rates (i.e. spike counts within a 30 ms window) resulting from this procedure are plotted in Fig. 2.3B.

In order to demonstrate that readouts that send feedback into the circuit can just as well represent neurons *within* the circuit, we had chosen the readout neurons that send feedback to be I&F neurons with noise, like the other neurons in the circuit. Each of them received synaptic inputs from a slightly different randomly chosen subset of neurons within the circuit. Furthermore the signs of weights of these synaptic connections were restricted to be positive (negative) for excitatory (inhibitory) presynaptic neurons.

The 8 readout neurons that provided feedback were trained to represent in their firing activity at any time the information in which of input streams 1 or 2 a burst had most recently occurred. If it occurred most recently in input stream 1, they were trained to fire at 40 Hz, and they were trained not to fire whenever a burst had occurred most recently in input stream 2. The training time was 200 s (of simulated biological time). After training, their output was correct 86% of the time (average over 50 s of input streams; counting the high-dimensional attractor as being in the on-state if the average firing rate of the 8 readout neurons was above 34 Hz). It was possible to train these readout neurons to acquire such persistent firing behavior, although they only received input from a circuit with fading memory, because they were actually trained to acquire the following behavior: fire whenever the rate in input stream 1 becomes higher than 30 Hz, or if one can detect in the current state $\mathbf{x}(t)$ of the circuit traces of recent high feedback values, provided the rate of input stream 2 stayed below 30 Hz. Obviously this definition of the learning target for readout neurons only requires a fading memory of the circuit.

The readouts for the other 3 tasks achieved in 50 tests for new inputs over 1 s (that had been generated by the same distribution as the training inputs, see the preceding description) the following average performance:

Task of panel E:	Mean correlation:	0.85
Task of panel F:	Mean correlation:	0.63
Task of panel G:	Mean correlation:	0.86 .

2.4.3 Technical Details to Figure 2.4

The same circuit as for Fig. 2.3 was used. First 2 linear readouts with feedback were simultaneously trained to become highly active after the occurrence of the cue in the spike input, and then to linearly reduce their activity, but each within a different time span (400 versus 600 ms). Their feedback into the circuit consisted of 2 time-varying analog values (representing time-varying firing rates of 2 population of neurons), which were both injected (with randomly chosen amplitudes) into the same subset of 350 neurons in the circuit. Their weights $\mathbf{w}$ were trained by linear regression for a total training time of 120 s (of simulated biological time), consisting of 120 runs of length 1 s with randomly generated input-cues (a burst at 200 Hz for 50 ms) and noise inputs (5 spike trains at 10 Hz).

2.4.4 Technical Details to Figure 2.5

Time-varying firing rates for the two input streams (consisting each of 8 Poisson spike trains) were drawn randomly from values between 10 and 90 Hz. The 16 spike trains from the 2 input streams, as well as feedback from trained readouts were injected into randomly chosen subsets of neurons. In contrast to the experiment for Fig. 2.3, these circuit inputs were not injected into spatially concentrated clusters of neurons, but to a sparsely distributed subset of neurons scattered throughout the 3-dimensional circuit. As a consequence, the firing activity $CA(t)$ of the high-dimensional attractor (see Fig. 2.5D) cannot be readily detected from the spike raster in Fig. 2.5C. Both the linear readout that sends feedback, and subsequently the other two linear readouts (whose output for a test input to the circuit is shown in Fig. 2.5E,F), were trained by linear regression during 140 s of simulated biological time.

Average performance of linear readouts on 100 new test inputs of length 700 ms (that had been generated from the same distribution as the training inputs):

Task of panel D:	Mean correlation:	0.82
Task of panel E:	Mean correlation:	0.71
Task of panel F:	Mean correlation:	0.79 .

Chapter 3

Movement generation with circuits of spiking neurons

How can complex movements that take hundreds of milliseconds be generated by stereotypical neural microcircuits consisting of spiking neurons with a much faster dynamics? This chapter shows that linear readouts from generic neural microcircuit models can be trained to generate basic arm movements. Such movement generation is independent of the arm-model used and the type of feedbacks that the circuit receives. It is demonstrated by considering two different models of a two-jointed arm, a standard model from robotics and a standard model from biology, that each generate different kinds of feedback. Feedbacks that arrive with biologically realistic delays of 50–280 ms turn out to give rise to the best performance. If a feedback with such desirable delay is not available, the neural microcircuit model also achieves good performance if it uses internally generated estimates of such feedback. Existing methods for movement generation in robotics that take the particular dynamics of sensors and actuators into account ("embodiment of motor systems") are taken one step further with this approach, which provides methods for also using the "embodiment of motion generation circuitry", i.e., the inherent dynamics and spatial structure of neural circuits, for the generation of movements.

3.1 Introduction

Using biologically realistic neural circuit models to generate 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 tasks that require sequences of precise motor commands on a relatively slow time scale. However it is shown that without the construction of any particular circuit, training a linear readout to take a suitable weighted sum (with *fixed* weights after training) of the output activity of a fairly large number of neurons in a generic neural microcircuit model provides a very general paradigm for movement generation. It is obviously reminiscent of a number of experimental results (see e.g. [84]) which show that a suitable weighted sum of the activity from a fairly large number of cortical neurons in monkeys predicts quite well the trajectory of hand positions for a variety of arm movements. Obviously the neural microcircuit model assumes here a similar role as a kernel for support vector machines in machine learning (see [85] and [86] for details).

This chapter demonstrates that controllers made from generic neural microcircuits are functionally "generic" in the sense that readouts from such circuits can learn to control the arm irrespective of the model that is used to describe the arm dynamics, the type of feedbacks used (visual or proprioceptive), and also the type of movements that are generated. This is shown here by teaching the same generic neural circuit to generate reaching movements for two different models, with different kinds of feedbacks. The first model used (Model 1) is the standard model of a 2-joint robot arm described in [16]. The other model [87, 88] comes from biology and relates the activity of neurons in the cortical motor area M1 to the kinematics of the arm (Model 2).

It turns out that both the spatial organization of information streams, especially the population coding of slowly varying input variables, and the inherent dynamics of the generic neural microcircuit model have a significant impact on its capability to generate 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 the performance of this generic neurocontroller is optimal for feedback delays that lie in the biologically realistic range of 50 to 280 ms. Furthermore, it is shown that other readout neurons from the same neural microcircuit model can be trained simultaneously to estimate results of such feedbacks, and that in the absence of real feedbacks the precision of reaching movements can be improved significantly if the circuit gets access to these estimated feedbacks.

This work complements preceding work where generic neural microcircuit models were used in an open loop for a variety of sensory processing tasks ([17], [18], [85]). 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 chapter. The paradigm for movement generation discussed in this chapter is somewhat related to preceding work [89], where a fixed parametrized system of differential equations was used instead of neural circuits, and to the melody-generation and prediction of chaotic time series with artificial neural networks in discrete time of [20, 14]. In these other models no effort is 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 chapter to make the process by which the neural circuit model (more specifically: the readouts from this circuit) learns to generate specific movement primitives in a biologically realistic fashion. Hence the results of this chapter only provide evidence that a generic neural microcircuit can hold the information needed to generate certain movement primitives, and that it can generate a suitable slow dynamics with high precision.

The structure of this chapter is as follows: Section 3.2 describes the neural microcircuit model. This is followed by the description of the robot arm model (Model 1) in section 3.3. Sections 3.4, 3.5, and 3.6 present results of computer simulations for Model 1. Section 3.7 repeats the experiment described in section 3.4 for the biologically motivated arm model (Model 2). Finally section 3.8 discusses robustness issues related to this new paradigm for movement generation.

A preliminary version of some results from this chapter (for movements of just one fixed temporal duration, and without Model 2) have previously been presented at a conference [10].

3.2 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 neural circuits out of biologically realistic computational units that solve specific computational problems, such as generating arm movements to various given targets. In fact, the generation of a smooth arm movement appears to be particularly difficult for a circuit of spiking neurons, since the dynamics of arm movements takes place on a time scale of hundreds of milliseconds, whereas the inherent dynamics of spiking neurons takes place on a much faster time scale. This chapter demonstrates that this problem can be solved, even with a generic neural microcircuit model whose internal dynamics has not been adjusted or specialized for the task of creating arm movements, by taking as activation command for a muscle at any time t a weighted sum $\mathbf{w} \times \mathbf{z}(t)$ of the vector $\mathbf{z}$ that describes the current firing activity of all neurons in the circuit.[1] The weight vector $\mathbf{w}$, which remains fixed after training, is the only part that needs to be specialized for the generation of

[1]As usual a constant component is formally included in $\mathbf{z}(t)$ so that the term $\mathbf{w} \times \mathbf{z}(t)$ may contain some fixed bias.

a particular movement task. Each component of $\mathbf{z}(t)$ models the impact that a particular neuron v may have on the membrane potential of a generic readout neuron. Thus each spike of neuron v is replaced by a pulse of unit amplitude 1 that decays exponentially with a time constant of 30 ms. In other words: $\mathbf{z}(t)$ is obtained by applying a low-pass filter to the spike trains emitted by the neurons in the generic neural microcircuit model. Note that it is already known that hand trajectories of monkeys can be recovered from the current firing activity $\mathbf{z}(t)$ of neurons in motor cortex through the same types of weighted sums as considered in this chapter [84].

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 chapter, although it remains an interesting open research problem how unsupervised learning could optimize a circuit for motor control tasks. 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 3.6) that may be desirable for the same or other tasks.

The generic microcircuit models used for the closed loop control tasks described in this chapter were similar in structure to those that were earlier used for various sensory processing tasks in an open loop. More precisely, the circuits considered consisted of 600 leaky-integrate-and-fire neurons arranged on the grid points of a $20\times5\times6$ cube in 3D (see Fig. 3.1). 20 % of these neurons were randomly chosen to be inhibitory. Synaptic connections were chosen according to a biologically realistic probability distribution that favored local connections but also allowed some long range connections. Biologically realistic models for dynamic synapses were employed instead of the usual static synapses of artificial neural network models. Parameters of neurons and synapses were chosen to fit data from microcircuits in rat somatosensory cortex (based on [66] and [65]), see section 3.10.

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.

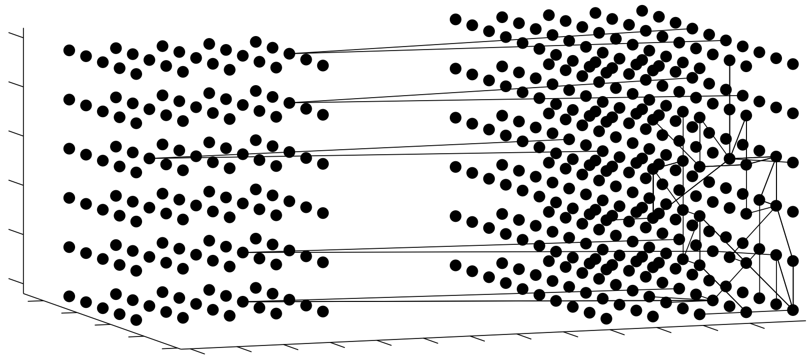

Figure 3.1: Spatial arrangement of neurons in the neural microcircuit models considered in this chapter. The neurons in the 6 layers on the left hand side encode the values of the 6 input/feedback variables $x_{dest}, y_{dest}, \theta_1(t-\Delta), \theta_2(t-\Delta), \tau_1(t), \tau_2(t)$ in a standard population code. Connections from these 6 input layers (shown for a few selected neurons), as well as connections between neurons in the subsequent 6 processing layers are chosen randomly according to a probability distribution discussed in the text (a typical example is shown).

The neural circuit receives in the case of the arm model that is considered in sections 3.3 - 3.6 (Model 1) analog input streams from 6 sources (from 8 sources in the experiment with internal predictions discussed in Fig. 3.8 and 3.9). A critical factor for the performance of these neurocontrollers is the way in which these time-varying analog input streams are fed into the circuit. The outcomes of the experiments discussed in this chapter would have been all negative if these analog input streams were fed into the circuit as time-varying input currents. 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 a standard form of population coding ([90] was applied. Each of the 6 time varying input variables was mapped onto an array of 50 symbolic input neurons with bell-shaped tuning curves (see section 3.10). 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 neurons in each of these 6 input arrays are connected[2] with one of the 6 layers consisting of 100 neurons in the circuit of 100×6 $((20 \times 5) \times 6)$ integrate-and-fire neurons, providing a time-varying input current to a randomly selected subset of integrate-and-fire neurons on that layer; see Fig. 3.1.

[2]$\lambda = 3.3$, in the formula for the connection probability given in the section 3.10.

3.3 A 2-joint robot arm as a benchmark nonlinear control task

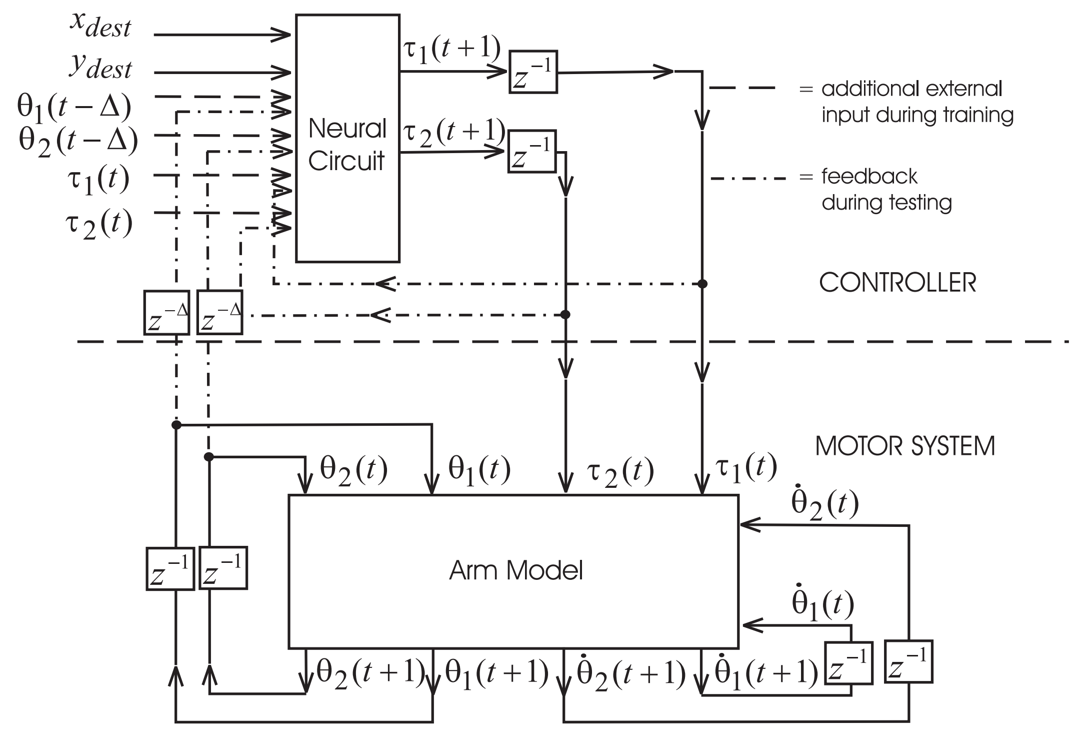

Figure 3.2: Closed loop application of a generic neural microcircuit. The weight vectors of the linear readouts from this circuit that produce the next motor commands $\tau_1(t{+}1), \tau_2(t{+}1)$ are the only parameters that are adjusted during training. After training the neural circuit receives in this closed loop as inputs a target position x_{dest}, y_{dest} for the tip of the robot arm (in cartesian coordinates; these input remain constant during the subsequent arm movement) as well as feedback $\theta_1(t-\Delta), \theta_2(t-\Delta)$ from the arm representing previous values of joint angles delayed by an amount Δ, as well as "efferent copies" $\tau_1(t), \tau_2(t)$ of its preceding motor commands. 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 the constant inputs (and in response to the dynamics of the feedbacks). During training of the readouts from the generic neural circuit the proprioceptive feedbacks $\theta_1(t{-}\Delta), \theta_2(t{-}\Delta)$ and the efferent copies of previous motor commands $\tau_1(t), \tau_2(t)$ are replaced by corresponding values for a target movement which are given as external inputs to the circuit ("imitation learning").

Initially a generic neural microcircuit model was trained (see Fig. 3.1 and Fig. 3.2) to control a standard model for a 2-joint robot arm (Model 1), see Fig. 3.3. This model is used in [16] 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.

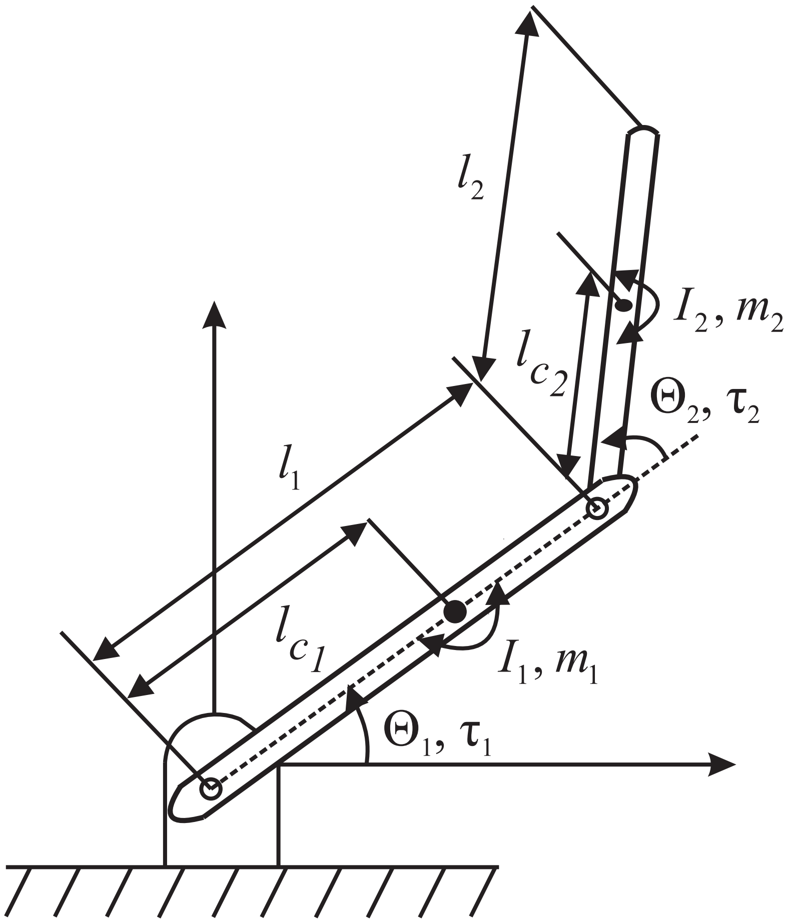

Figure 3.3: Standard model of a 2-joint robot arm.

Using the well-known Lagrangian equation in classical dynamics, the dynamic equations for this arm model are given by equation 4.1:

$$\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -h\dot{\theta}_2 & -h(\dot{\theta}_1 + \dot{\theta}_2) \\ h\dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{3.1}$$

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 to the two joints, and

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

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

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

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

Equation 4.1 can be compactly written 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. I_1, I_2 are the moments of inertia of the two joints. The values of the parameters that were used in these 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 is shown in Fig. 3.2. During training of the weights of the linear 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 4.1 (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. The delay Δ of the proprioceptive or sensory feedback is assumed to have a fixed value of 200 ms, except for section 3.6 which describes the impact of this value for the precision of the movement. For each such target trajectory 20 variations of the training samples were generated, for which at each time step[3] t a different noise value of $10^{-5} \times \rho$ 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 that input channel. The purpose of this extended training procedure was to make the readout robust with regard to deviations from the target trajectory caused by faulty earlier torque outputs given by the readouts from the neural circuit (see section 3.8). Each target trajectory had a time duration of 500 ms.

3.4 Teaching a generic neural microcircuit model to generate basic movements

As a first task, the generic neural microcircuit model described in section 3.2, was taught to generate with the 2-joint arm described in section 3.3, the 4 movements indicated in Fig. 3.4. 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 two readouts from the neural microcircuit model

[3]All time steps were chosen to have a length of 2 ms, except for the experiment reported in Fig. 3.6, where a step size of 1 ms was used to achieve a higher precision.

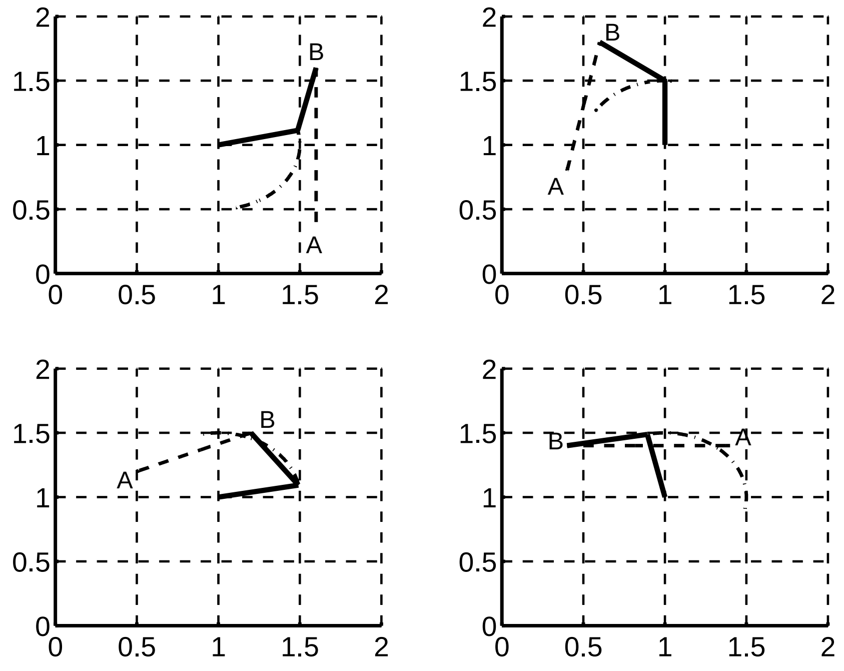

Figure 3.4: Initial position A and end position B of the robot arm (Model 1) for 4 target movements, scaled in meters. The target trajectory of the tip of the robot arm and of the elbow are indicated by dashed and dashed/dotted lines. One sees clearly that even simple linear movements of the tip to the arm require quite nonlinear movements of the elbow.

were trained by linear regression to output the joint torques required for each of these movements.[4]

20 noisy variations of each of the 4 target movements were used for the training of the two readouts by linear regression, as specified in section 3.3. 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

[4] Training data were generated as follows: For a given start point $\langle x_{start}, y_{start}\rangle$ and target end point $\langle x_{dest}, y_{dest}\rangle$ of a movement (both given in cartesian coordinates) an interpolating trajectory of the tip of the arm was generated according to the following equation given in [91]:

$$x(t) = x_{start} + (x_{start} - x_{dest}) \cdot (15\tau^4 - 6\tau^5 - 10\tau^3)$$

$$y(t) = y_{start} + (y_{start} - y_{dest}) \cdot (15\tau^4 - 6\tau^5 - 10\tau^3)$$

where $\tau = t/MT$ and MT is the target movement time (in this case $MT = 500$ ms). From this target trajectory for the endpoint of the robot arm, target trajectories of the angles Θ_1, Θ_2 of the robot arm were generated by applying standard equations from geometry (see e.g. [92]). From these the target trajectories of the torques were generated according to equ. (4.1).

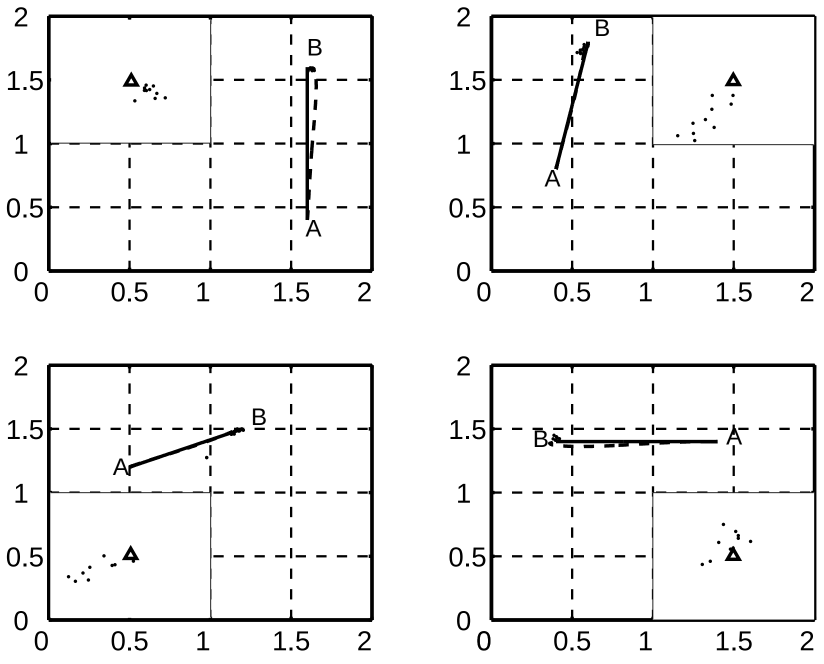

Figure 3.5: Target trajectories of the tip of the robot arm as in Fig. 3.4 (solid) and resulting trajectories of the tip of the robot arm in a closed loop for one of the test runs (dashed). The dots around the target end points show the end-points of the tip of the robot arm for 10 test runs for each of the movements (enlarged inserts show a 20 cm × 20 cm area with target end point B marked by a black open triangle. Differences are due to varying initial conditions and simulated inherent noise of the neural circuit. Nevertheless all movement trajectories converged to the target, with an average deviation from the target end point of 4.72 cm, and the SD of 0.85 cm (scale of figures in m).

at the readout.

The performance of the trained neural microcircuit model during validation in the closed loop (see Fig. 3.2) is demonstrated in Fig. 3.5. When the circuit receives as input the coordinates $\langle x_{dest}, y_{dest} \rangle$ of the endpoint B of one of the target movements shown in Fig. 3.4, the circuit autonomously generates in a closed loop the torques needed to move the tip of the 2-joint arm from the corresponding initial point A to this endpoint B [5].

[5]In these experiments no effort was made to stabilize the endpoint of the arm at or near the target position. Rather the movement was externally halted at the end of the allotted time period of 500 ms. Hence the neural circuit model acts as a movement generator, rather than as a controller. However I am not aware of a fundamental obstacle which would make it impossible to teach a circuit to stabilize the arm once it has reached the target position (which the circuit receives as an extra input).

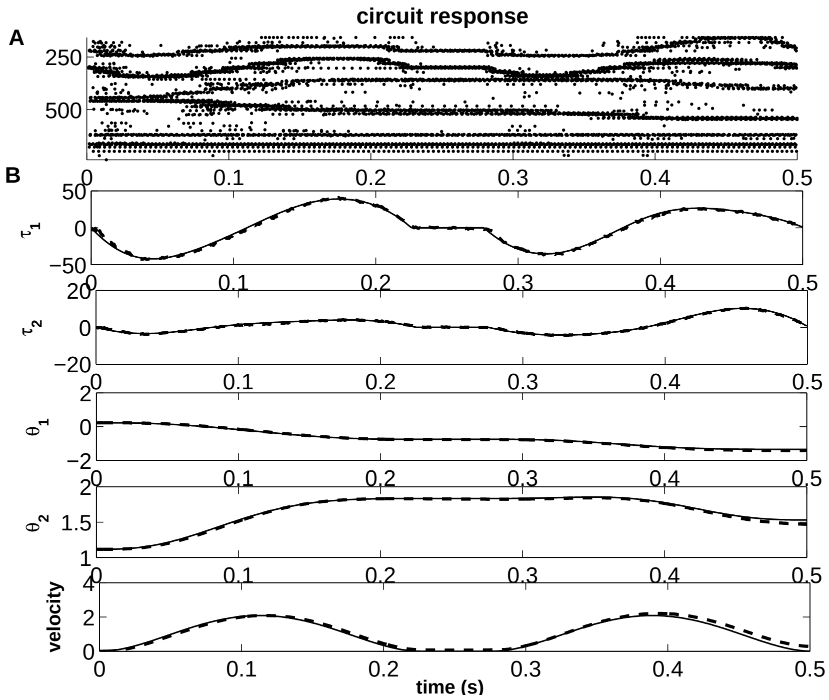

Figure 3.6: Demonstration of the temporal integration capability of the neural controller. The data shown are for a validation run for a circuit that has been trained to generate a movement that requires an intermediate stop and then autonomous continuation of the movement after 50 ms. **a)** Spike raster of the 600 neurons on the right hand side of Fig. 3.1. Note that the readout neurons receive at time t only information about the last few spikes before time t (more precisely: they receive at time t the liquid state $x(t)$ of the circuit as their only input). **b)** Target time courses of the joint angles θ_1, θ_2, joint torques τ_1, τ_2 and of the velocity of the tip of the robot arm are shown as solid line, actual time courses of these variables during a validation run in closed loop as dashed lines.

Obviously temporal integration capabilities of the controller are needed for the control of many types of movements. The next experiment was designed to test explicitly this capability of neurocontrollers constructed from generic circuits of spiking neurons. Fig. 3.6 shows results for the case where the readouts from the neural microcircuit have been trained to generate an arm movement with an intermediate stop of all movement from 225 to 275 ms (see the velocity profile at the bottom of Fig. 3.6). The initiation of the continuation of the movement at time $t = 275$ ms has to take place without any external cue, just on the basis of the inherent temporal integration capability of the neural circuit. For the sake of demonstration purposes I chose for the experiment reported in Fig. 3.6 a feedback delay of just 1 ms, so that all circuit inputs are constant during 49 ms of the 50 ms while the controller has to wait, forcing the readouts to decide just on the basis of the inherent circuit dynamics when to move on. Nevertheless the average deviation of the tip of the robot arm for 20 test runs (with noisy initial conditions and noise on feedbacks as before) was just 6.86 cm, and the bottom part of Fig. 3.6 shows (for a sample test run) that the tip of the robot arm came to a halt during the period from 225 to 275 ms, and then autonomously continued to move.

3.5 Generalization capabilities

The trained neurocontroller (with the weights of the linear readouts being the only parameters that were adjusted during training) had some limited capabilities to generate arm-reaching movements to new targets. For the experiment reported in Fig. 3.7, 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, provided that they were located between target points used for training. The autonomously generated reaching movements moved the tip of 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.

3.6 On the role of feedback delays and autonomously generated feedback estimates

Our model assumes that the neural circuit receives as inputs in addition to the constant target end points and efferent copies $\tau_1(t), \tau_2(t)$ of its movement commands with very little delay, also proprioceptive or visual feedback that provides at time t information about the values of the angles of the joints at time $t - \Delta$. Whereas it is quite difficult to construct circuits or other artificial controllers for imitating movements that can benefit significantly from feedback (for example with the approach of [89]), especially if this feedback is significantly delayed, it is shown in Fig. 3.8 that neurocontrollers built from generic neural microcircuit models are able to generate and control movements for feedbacks with a wide range of delays. In fact, Fig. 3.8 shows that the smallest deviation between the target end point $\langle x_{dest}, y_{dest}\rangle$ and the actual end point of the tip of the robot arm is not achieved when this delay Δ has a value of 0, but for a range of delays between 50 and 280 ms. In order to make sure that this surprising result is not an

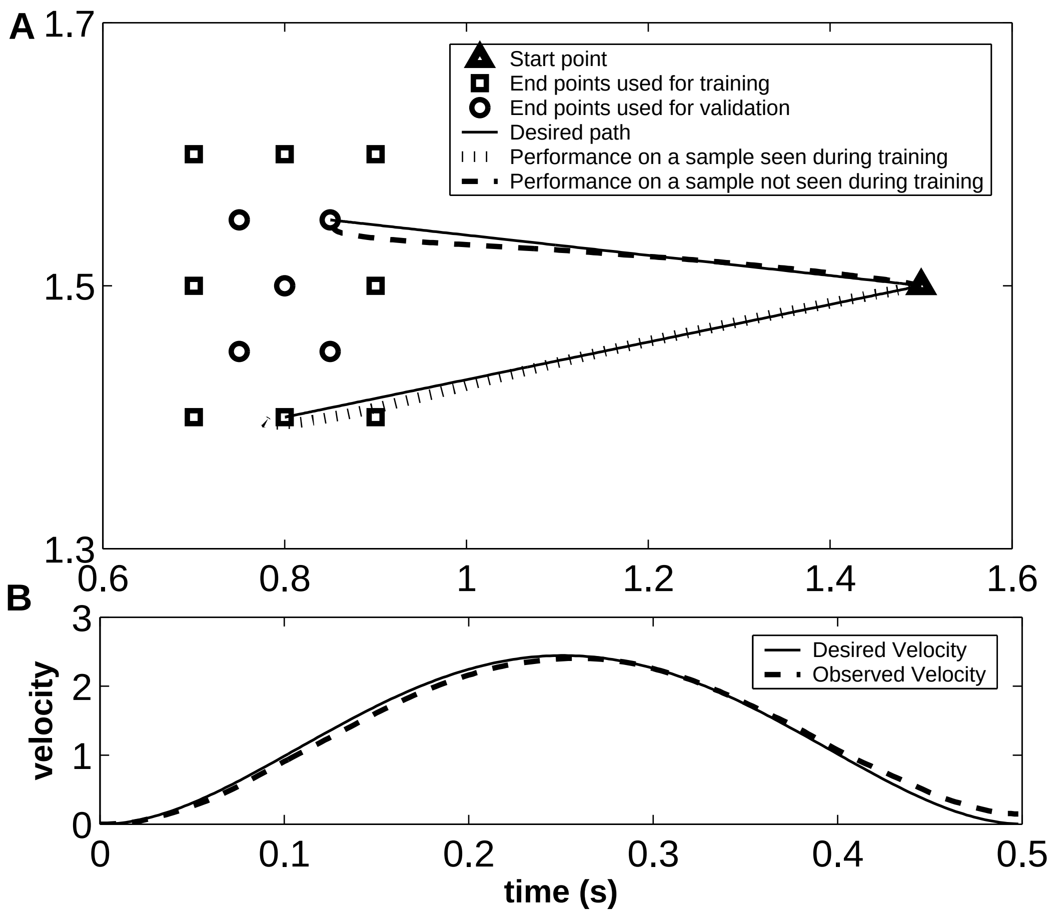

Figure 3.7: Generation of reaching movements to new target end points that lie between end points used for training. **a)** Generalization of movement generation to 5 target end points (small circles) which were not among the 8 target end points (small squares) that occurred during training. Movements to a new target end point was initiated by giving its Cartesian coordinates as constant 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 movements to a new target end point (solid line is ideal bell-shaped velocity profile, actual - dashed).

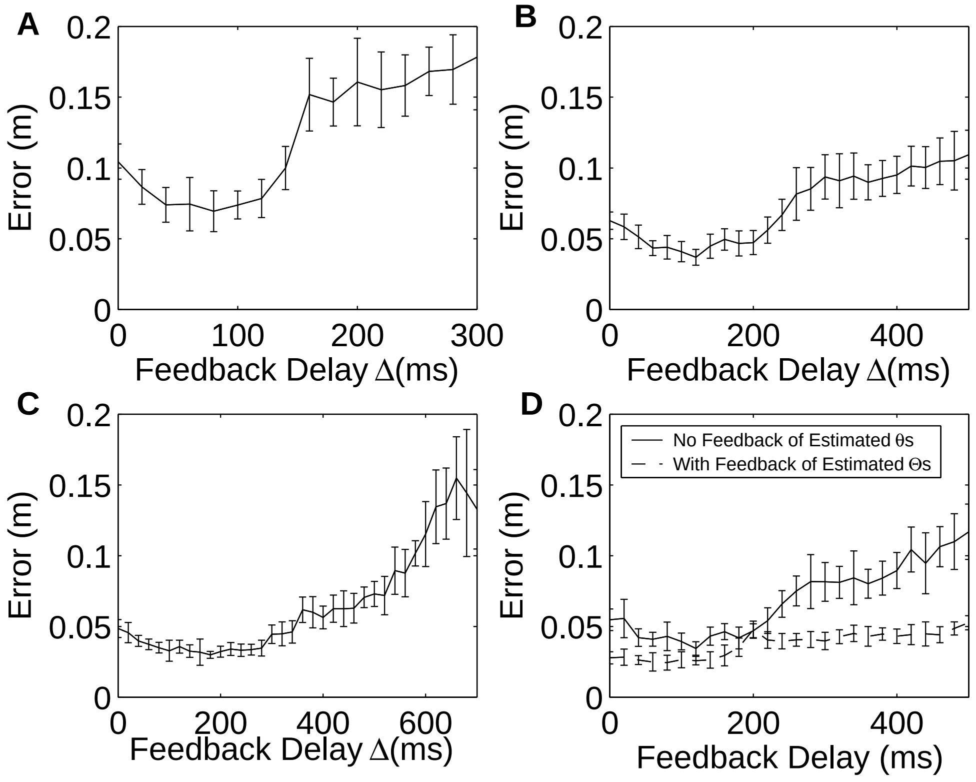

Figure 3.8: Influence of feedback delay Δ on movement error. Error is defined as the difference of desired and observed end-point of movement. The delay Δ is for proprioceptive feedbacks $\theta_1(t-\Delta), \theta_2(t-\Delta)$. The curves show the averages and the vertical bars show the SD of the data achieved for 400 movements for each value of Δ (4 different movements as shown in Fig. 3.4 repeated 10 different times with different random initial conditions of the circuit and different online noise for each of 10 randomly drawn generic neural microcircuit models). Panels a), b), c) show these data for 3 different movement durations: 300, 500, and 700 ms. Panel d) shows in the upper curve results for a slightly larger neural circuit (consisting of 800 instead of 600 early integrate-and-fire neurons). The lower (dashed) curve in d) shows the performance of the same circuits when internally generated estimates of proprioceptive feedbacks (for a delay of 200 ms) were fed back as additional inputs to the neural circuit. Note that the use of such internally estimated feedbacks not only improves the movement precision for all values of the actual feedback delay Δ expect for $\Delta = 200\,\text{ms}$, but also reduces the SD of the precision achieved for different circuits considerably.

artifact of some particular randomly drawn neural microcircuit model or a particular arm movement, it has been tested on each of 10 randomly drawn neural microcircuits with 12 different movements (4 trajectories as shown in figure 3.4, each created at 3 different speeds, resulting in movement times of 300, 500 and 700 ms). The result of these statistical experiments are reported in Fig. 3.8 a), b), c). The rightmost point on each of the 3 curves shows the performance achieved without any feedback (since for this point the delay of the feedback is as large as the duration of the whole movement). Compared with that, feedback with a suitable delay reduces the imprecision of the movement by at least 50 %. Altogether these data show that the best values for the feedback delay lie in the range of 50 to 280 ms. The upper bound for this interval depends somewhat on the duration of the movement. A possible explanation for the fact that feedbacks with a delay of less than 50 ms are less helpful is that in this case the current target circuit output is very similar to the currently arriving feedback, and hence it is more difficult for the circuit to learn the map from current feedback to current target output in a noise-robust fashion. In addition a delayed feedback complements the inherent temporal integration property of the neural microcircuit model (see [85]), and therefore tends to enlarge the time constant for the fading of memory in the closed loop system. Hence these neurocontrollers perform best for a range of feedback delays that contain typical values of actual delays for proprioceptive and visual feedback measured in a variety of species (e.g. 120 ms for proprioceptive feedback and 200 ms for visual feedback is reported in [93]).

In another computer experiment I have examined the potential benefit of using estimated feedback for the neurocontroller under consideration. Estimation of feedback is very easy for such neural architecture, since the generic neural microcircuit model that generates (via suitable readouts) the movement commands has not been specialized in any way for this movement generation task, and can simultaneously be used as information reservoir for estimating feedbacks. More precisely, 2 additional readouts were added and trained to estimate at any time t the values of the joint angles θ_1 and θ_2 at time $t - 200\,\text{ms}$, i.e., 200 ms earlier. These delayed values were chosen as targets for these 2 additional readouts during training, since the previously reported results (see in particular Fig. 3.8b)) show that a feedback of the actual values of θ_1 and θ_2 with a delay of 200 ms is quite beneficial for the precision of the movement that is generated. After training, the weights of these 2 additional readouts were frozen (like for the first 2 readouts which produce the movement commands).[6] The outputs of these 2 additional

[6]Since neither the training of the readouts for movement commands nor the training of the readouts for retrodiction of sensory feedback changes the neural circuit itself, it does not matter whether these readouts are trained sequentially or in parallel. In these experiments both types of readouts were trained simultaneously, while the target values of both $\theta_1(t-\Delta), \theta_2(t-\Delta)$ and $\theta_1(t-200), \theta_2(t-200)$ were given to the circuits as additional inputs during training (where Δ

readouts were also fed back into the circuit (without delay), see Fig. 3.9. Compared with the architecture shown in Fig. 3.2 the neural circuit receives now 2 additional time-varying inputs. These were fed into the circuit in the same way as the other 6 inputs (described in section 3.2). Thus 2 additional arrays consisting of 50 neurons each were used for a population coding of these time-varying input variables, and 2 “columns” consisting of 100 neurons each were added of the neural circuit that received the outputs of these 2 additional input-arrays.

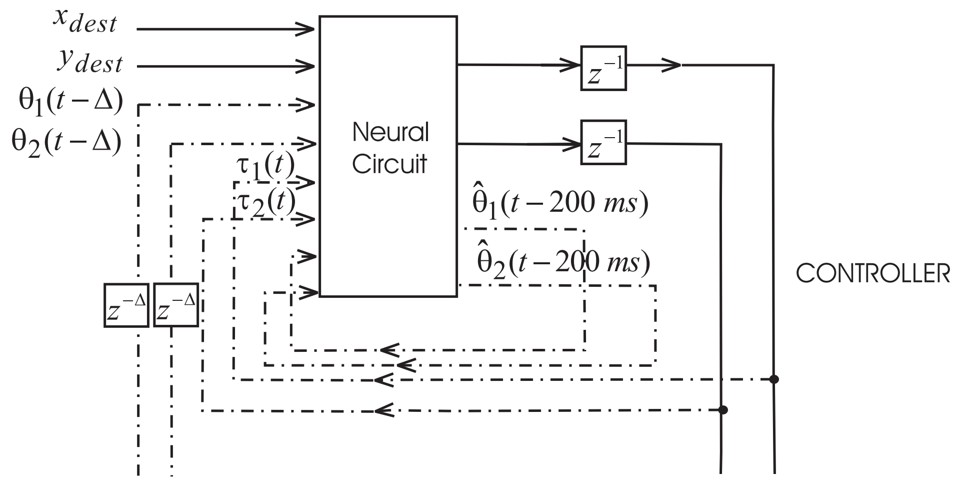

Figure 3.9: Information flow for the case of autonomously generated estimates $\hat{\theta}(t-200\text{ ms})$ of delayed feedback $\theta(t-200\text{ ms})$. Rest of the circuit as in Fig. 3.2.

The top solid line in Fig. 3.8 d) shows the result (computed in the same way as in the other panels of Fig. 3.8 for the case when the values of the estimates of $\theta_1(t-200\text{ ms})$ and $\theta_2(t-200\text{ ms})$ produced by the 2 additional readouts were not fed back into 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 collect and redistribute information within the neural circuit; this additional feedback significantly improved the performance of the neurocontroller for all values of the actual delay Δ of feedback about the values of θ_1 and θ_2 (except for $\Delta = 200$ ms). The value on the rightmost point of the lower curve for $\Delta = 500$ ms shows the improvement achieved by using estimated sensory feedback in case when no feedback arrives at all, since the total movements lasted for 500 ms. Altogether

is the assumed actual feedback delay plotted on the x-axis in Fig. 3.8 d)).

the use of internally estimated feedback improved the precision of the movement by almost 50 % for most values of the delay of the actual feedback.

3.7 Application to a biological model for arm control

Whereas in the preceding section I have focused on a model for a robot arm as a standard example for a highly nonlinear control task, I will demonstrate in this section that the same paradigm for movement generation can also be applied to a well known model for cortical control of arm movements in primates [88, 87]. This model proposes a direct relationship between the firing rate c_j of individual neuron j in primary motor cortex M1 (relative to some baseline firing rate C) and the kinematics (in Cartesian coordinates) and endpoint force f_{ext} of the hand, which is viewed here simply as the tip of a 2-joint arm:

$$c_j(t-d) = \frac{\mathbf{u}_j^T}{2}\left(F^{-1}\mathbf{f}_{ext}(t) + m\ddot{\mathbf{x}}(t) + k\mathbf{x}(t)\right) + b\lfloor \mathbf{u}_j^T\dot{\mathbf{x}}(t)\rfloor \ . \tag{3.2}$$

The vector $\mathbf{u}_j$ denotes the direction in which the end point force is generated due to activation of muscles by neuron j (assuming cosine tuning of neurons). In these simulations I simply took 4 unit vectors $\mathbf{u}_j$ pointing up, down, left, right. $\mathbf{f}_{ext}(t)$ is the endpoint force that the hand applies against external objects. $\mathbf{x}, \dot{\mathbf{x}}$, and $\ddot{\mathbf{x}}$ are the position, velocity and acceleration of the hand respectively (we usually write $\langle x, y\rangle$ for the hand position $\mathbf{x}$ in a 2-dimensional space).

Although the precise relationship between the activity of neurons in motor cortex and the activation of individual muscles is extremely complicated and highly nonlinear, a derivation given in in [88] suggested that equation 3.2 provides a quite good (almost linear) local approximation to multijoint kinematics over a small workspace. As a consequence I have applied this model only for arm movements when the hand moves on the boundaries of a 28.28 cm $\times$ 28,28 cm square.

Since we are only concerned with the movement of the hand in its workspace, and do not require the hand to exert an endpoint force on external world objects, $\mathbf{f}_{ext}(t)$ can be set to 0.

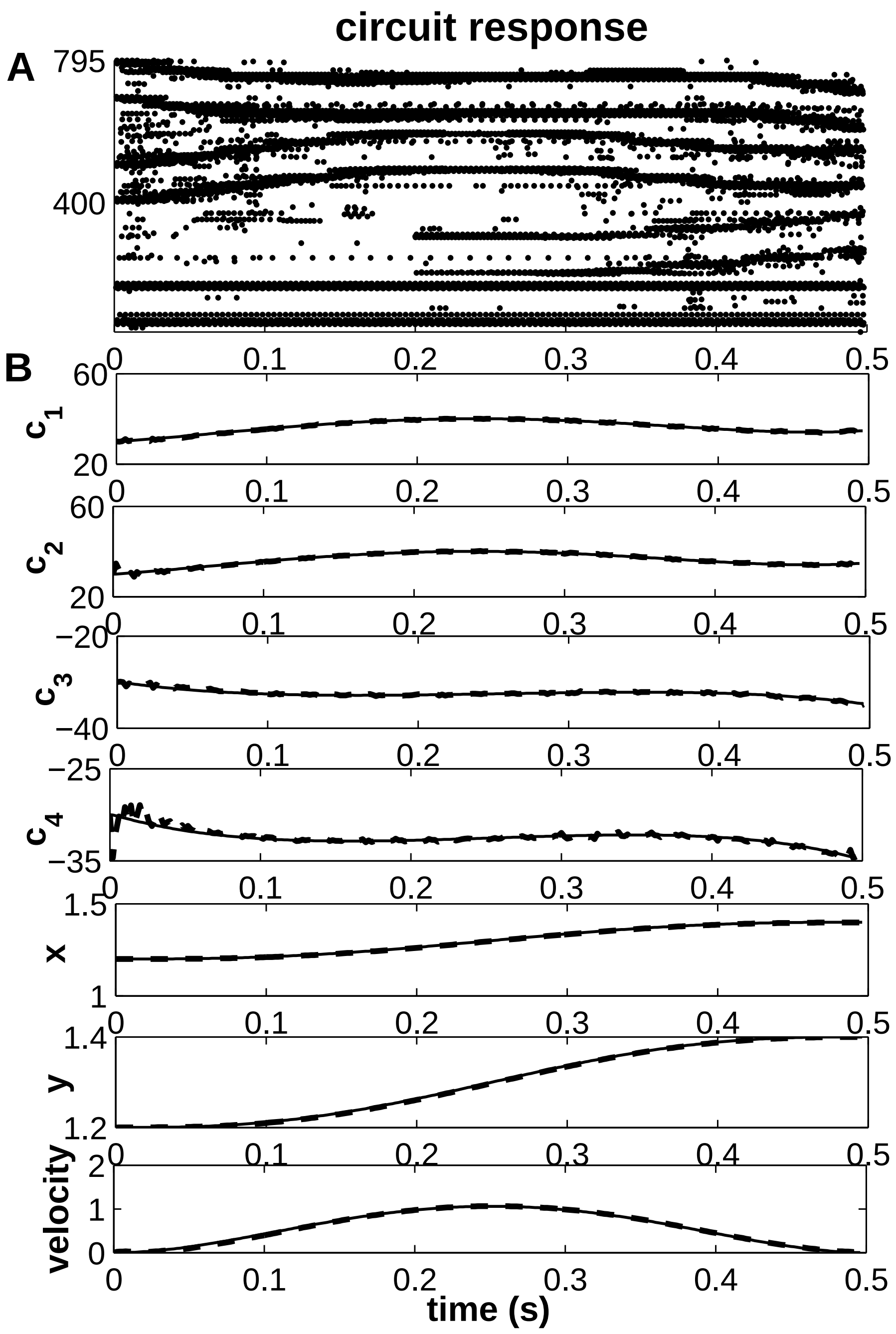

Figure 3.10: Generation of an arm movement for a biological model for cortical control of muscle activations. **a)** Spike raster analogous to Fig. 3.6. **b)** Solid lines denote target values and dashed lines show performance of simulated readouts $c_1, \ldots, c_4$ from a simulated microcircuit in motor cortex that receives significantly delayed information about earlier hand positions as feedback (simulating visual feedback to motor cortex). Scales for $c_1, \ldots, c_4$ in N, for x, y in m, for the velocity of the hand in m/s.

For the sake of simplicity, the transmission delay d from cortex to muscles is also set to 0 (but this model would work just as well for other values of d). This simplifies the model to:

$$c_j(t) = \frac{\mathbf{u}_j^T}{2}\left(m\ddot{\mathbf{x}}(t) + k\mathbf{x}(t)\right) + b\lfloor \mathbf{u}_j^T \dot{\mathbf{x}}(t) \rfloor \tag{3.3}$$

In the computer experiments this model was simulated with the parameter values $b = 10$ Ns/m, $k = 50$ N/m, and $m = 1$kg suggested in [88].

In order to produce a paradigm for arm control by cortical circuits I took a generic cortical microcircuit model consisting of 800 neurons as described in section 3.2. 4 readouts that received inputs from all neurons in this microcircuit model were trained by linear regression to assume the role of these 4 neurons in motor cortex that directly control arm muscles resulting in hand movements according to equation 3.3.[7] These readout neurons were trained to produce 4 different hand movements along the edges of a 28.28 cm $\times$ 28.28 cm square whose diagonals were parallel to the x- and y-axis respectively.

The inputs to the neural microcircuit were the coordinates $\langle x_{dest}, y_{dest}\rangle$ of the desired target end point of the hand, efferent copies of the outputs $c_1(t), \ldots, c_4(t)$ of motor neurons $1, \ldots, 4$, and feedback $x(t-200)$ ms, $y(t-200)$ ms about preceding hand positions with a delay of 200 ms that is biologically realistic for visual feedback into motor cortex. The values of these 8 inputs were fed into the 800 neuron microcircuit model in the same way as for the 8 input circuit discussed at the end of the preceding section. The results for this experiment are shown in 3.10. The average deviation over 40 runs of the tip of the arm from the desired end-point was 0.13 cm with a SD of 7.2295×10^{-2} cm.

It is interesting to note that the generic neural microcircuit can also learn to generate movements for this quite different arm model. Another point of interest is that the control performance of the generic neural microcircuit is independent of the kind of feedbacks that it is receiving (*c.f.* angles in the earlier model and position coordinates in this model).

3.8 How to make the movement generation noise-robust

Mathematical results from approximation theory (see the appendix of [18] and [51] for details) imply that a sufficiently large neural microcircuit model (which con-

[7]Target trajectories of the endpoint of the arm were generated for training as described in footnote 4. Target outputs $c_j(t)$ for the readouts were generated from these trajectories by equation 3.3.

tains sufficiently diverse dynamic components to satisfy the separation property) can in principle (if combined with suitable static readouts) uniformly approximate any given time-invariant fading memory filter F.

Additional conditions have to be met for successful applications of neural microcircuit models in closed-loop movement generation tasks, such as those considered in this chapter. First of all, one has to assume that the approximation target for the neural microcircuit, some movement generator F for a plant P, is a time-invariant fading memory filter (if considered in an open loop). But without additional constraints on the plant and/or target movement generator F one cannot guarantee that neural microcircuits L that uniformly approximate F in an open loop can successfully generate similar movements of the plant P. Assume that F can be uniformly approximated by neural microcircuit models L, i.e., 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 movement generator. 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 movement commands[8] and $f(t)$ is the feedback (both $u_0(t)$ and $f(t)$ are in general multi-dimensional). Assume that such microcircuit model L has been chosen for some extremely small $\varepsilon > 0$. Even if the plant P has the common bounded input bounded output (BIBO) property, it 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 for these two cases 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 $f_F(s), f_L(s)$ as inputs may become much larger than ε, and hence the outputs of F and L with plant P may eventually diverge in this closed loop. 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.

This instability problem can be solved by training the readout from the neural circuit L to create an "attractor" around the trajectory generated by F in the noise-free case. This is possible because the current liquid state of the circuit depends not just on the most recent feedback to the circuit, but also on the preceding stream of feedbacks (therefore the liquid state also contains information about which particular part of the movement has to be currently carried out) as well as on the target end-position $\langle x_{dest}, y_{dest}\rangle$. If one trains the readout from

[8]In these 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.

circuit L to ensure that $\|(Lu_F)(t)-(Lu_L)(t)\|$ stays small when $u_F(\cdot)$ and $u_L(\cdot)$ did not differ too much at preceding time steps, one can bound $\|(Fu_F)(t)-(Lu_L)(t)\|$ by $\|(Fu_F)(t)-(Lu_F)(t)\|+\|(Lu_F)(t)-(Lu_L)(t)\| \leq \varepsilon+\|(Lu_F)(t)-(Lu_L)(t)\|$ and thereby avoid divergence of the trajectories caused by F and L in the closed-loop system.

This makes clear why it was necessary to train the readouts of the neural microcircuit models L to produce the desired trajectory not just for the ideal feedback $u_F(t)$ but also for noisy variations of $u_F(t) = \langle u_0(t), f_F(t)\rangle$ that represent possible functions $u_L(t)$ that arise if the approximating circuit L is used in the closed loop.

3.9 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 ([17], [18], [85], [94]). Previously only sensory processing tasks such as speech recognition or visual movement analysis ([17], [18], [95]) were considered in this context as benchmark tests for real-time computing. In this chapter I have applied such generic neural microcircuit models for the first time in a biologically more realistic closed loop setting, where the output of the neural microcircuit model directly influences 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 recent 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.

It is demonstrated that the capability of the neural circuit to generate reaching movements generalizes to novel target end positions of the tip of the arm that lie between those which occured during training (see Fig. 3.7). The velocity profile for these autonomously generated new reaching movements exhibits a bell-shaped velocity profile, like for the previously taught movements. It is proposed to view the basic arm movements that are generated in this way as possible implementations of muscle synergies, i.e. of rather stereotypical movement templates [96]. In this interpretation, the learning of a larger variety of arm movements requires superposition of time-shifted versions of several different basic movement templates of the type as are considered in this chapter. Such learning on a higher level is a topic of current research.

Surprisingly the performance of the neural microcircuit model for generating movements not only deteriorates if the (simulated) proprioceptive feedback is delayed by more than 280 ms, or if no feedback is given at all, but also if this feedback arrives *without* any delay. Our computer simulations suggest that the best performance of such neurocontrollers is achieved if the feedback arrives with a biologically realistic delay in the range of 50 to 280 ms. If the delay assumes other values, or is missing altogether, a significant improvement in the precision of the generated reaching movements can be achieved if additional readouts from the same neural microcircuit models that generate the movements are 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. 3.8 d).

Apart from these effects resulting from the interaction of the inherent circuit dynamics with the dynamics of externally 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 externally 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 population coding (see description in section 3.2) as implemented for the experiments reported in this chapter.

Another interesting point to be noted is that this model for motor control can successfully learn to control the arm movement irrespective of the model that is used to describe the dynamics of the arm-movement and the types of feedbacks that the circuit is receiving. One of the two arm models that was tested (see section 3.7) is a model for cortical control of muscle activation. Hence this model also provides a new hypothesis for the computational function of neural circuits in the motor cortex.

Altogether the results presented in this chapter may be viewed as a first step towards an exploration of the role of the "embodiment of motion generation circuitry", i.e., of concrete spatial neural circuits and their inherent temporal dynamics, in motor control. This complements the already existing work on the relevance of the embodiment of actuators to motor control [97].

3.10 Materials and Methods

3.10.1 Specification of Generic Neural Microcircuit Models

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].

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 [66] to 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II).

The (short term) dynamics of synapses was modeled according to the model proposed in [65], 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.

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 [66] 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 $\tau_s = 3\,\text{ms}$ ($\tau_s = 6\,\text{ms}$) for excitatory (inhibitory) synapses. The transmission delays between neurons were chosen uniformly to be 1.5 ms (EE), and 0.8 ms for the other connections.

The following input convention was applied. 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 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 $i(t) \cdot \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(n-n(t))^2}{2\pi^2}}$, where $\sigma = 0.8$).

Chapter 4

From memory based decisions to decision based movements

Interval discrimination task is a classical experimental paradigm that is employed to study working memory and decision making and typically involves four phases. First the subject receives a stimulus, then holds it in the working memory, then makes a decision by comparing it with another stimulus, and finally acts on this decision, usually by pressing one of the two buttons corresponding to the binary decision. This chapter demonstrates that simple linear readouts from generic neural microcircuits that send feedback of their activity to the circuit, can be trained using identical learning mechanisms to perform quite separate tasks of decision making and generation of subsequent motor commands. In this sense, the neurocomputational algorithm presented here is able to integrate the four computational stages into a single unified framework. The algorithm is tested using two-interval discrimination and delayed-match-to-sample experimental paradigms as benchmarks.

4.1 Introduction

'Decision making followed by action selection' is one of the most demanding and recurring event in our day to day lives [21]. For example, to choose a drink from a vending machine, you need to browse through the possible choices, hold the interesting ones in your working memory, decide which of these choices you want to buy, and finally press the button corresponding to your choice. The interval discrimination task [22, 23] is one of the classical experimental paradigms that is employed to study working memory and decision making. The experiment typically involves four phases, *viz.* the initial loading (L) of the first stimulus, maintaining (M) this stimulus in working memory till the subsequent stimulus is presented, making a binary decision (D), and finally acting (A) on this decision, usually by pressing one of the two buttons corresponding to the binary choice.

The precise computational and biophysical mechanism(s) through which the brain is able to execute this load-maintain-decide-act (LMDA) sequence is not understood. Several theoretical and modeling studies have tried to look at segregated phases of this sequence and give possible explanations for their working [24, 25, 26]. Typically neurocomputational models of working memory fail to address how the decisions made by neurons in the prefrontal cortex (PFC), are converted into motor commands, which are executed by the sensori-motor system [22, 27, 28]. Similarly, models for computational motor control tend to ignore the first three phases (LMD), that are responsible for generation of motor commands [29, 9, 10].

Several interesting modeling approaches for tasks involving working memory and decision making have been proposed recently [22, 28, 30]. These models propose different mechanisms e.g. precise tuning of mutual inhibition [22], fine-tuning of a heterogeneous recurrent network [30], using an integral feedback signal for inhibitory control [28]; to obtain the persistent neural activity which in turn stores information in the working memory. Despite existing evidence that shows synaptic learning as a responsible mechanism for working memory related tasks [23], all the models described above use static (no learning involved) neural circuits.

This chapter proposes a neurocomputational architecture that uses synaptic learning mechanisms (simply linear regression), and is able to integrate the four phases (LMDA) involved in the process of action selection in presence of a decision, into a unified computational framework. Essentially the neural model described here integrates two distinct cortical functions *viz.*, working memory and decision making carried out by the neurons in PFC, and subsequent action selection executed by the sensorimotor system. More precisely, it is demonstrated that delayed-decision tasks that are followed by action selection, can be solved, if feedback from trained linear readouts is provided to generic neural microcircuits whose internal dynamics has not been optimized for any particular computational task. Two classical experimental paradigms for interval-discrimination task are modeled that use different mechanisms to encode external sensory inputs. For comparison with earlier models of working memory, the unified framework is used to build a spiking neural network model of two-interval discrimination [22]. Additionally, to demonstrate that this computational paradigm is task-independent, robust to how the external sensory inputs are encoded, and is capable of integrating the A phase, another spiking neural network model is presented for the delayed-match-to-sample task [23], followed by an arm movement to the decided goal position.

The core principles behind the working of this model make the assumption that the cortex can be thought of as an ultra-high dimensional dynamical system, where the afferent inputs arriving from thalamus and the recurrent cortical feedbacks are churned in a non-linear way to obtain a high-dimensional projection of the low-dimensional input space. Preceding work has demonstrated that such high dimensional transient dynamics endows the neural circuit with analog fading memory (see Appendix) that can provide the circuit enough computational power for performing open-loop sensory processing tasks [17, 13, 59].

Analog fading memory by itself is not powerful enough to render the circuits the power to hold information in working memory. The obvious reason being that analog fading memory by itself has an upper limit on the order of tens of milliseconds, depending on the time constants of synapses and neurons in the neural circuit [13], whereas typically working memory holds information on the order of seconds. Recent results show that precisely tuned synaptic feedback can be a possible mechanism for maintaining persistent memory[98, 99], and that feedback from trained readout neurons can induce multiple co-existing "partial attractors" in the circuit dynamics [7, 8]. These results are further extended here to demonstrate that even in the presence of feedback noise, such "partial attractor" states can be maintained by generic neural circuits on the time-scales of several seconds, that is obviously a requirement for tasks involving working memory. The results presented in this chapter indicate that simple linear readouts from generic neural microcircuit models, that send their output as a feedback signal to the circuit, can be a plausible model of how the interval discrimination task is executed, and a subsequent action is chosen.

4.2 Generic Neural Microcircuit Models

In contrast to artificial neural networks, neural microcircuits in biological organisms are composed of diverse components such as different types of spiking neurons and dynamic synapses, each endowed with an inherently complex dynamics of its own. This poses a challenge to construct neural circuits out of biologically realistic computational units that solve specific computational problems, e.g. decision making or motor control. In fact, decision making and motor control are particularly challenging, since these tasks occur on the time scales of seconds, where as the inherent dynamics of spiking neurons takes place on a much faster timescale. This chapter demonstrates that this problem can be solved, if feedback from trained readouts is available to generic neural microcircuits whose internal dynamics has not been adjusted or specialized for any particular computational task, by taking at any time t, a weighted sum $\mathbf{w} \times \mathbf{y}(t)$ of the vector $\mathbf{y}$ that describes the current firing activity of all neurons in the neural circuit. The weight vector $\mathbf{w}$, which remains fixed after training, is the only part that needs to be optimized or specialized for the generation of any particular computational task (decision making or motor control). Each component of $\mathbf{y}(t)$ models the impact that a particular neuron v may have on the membrane potential of a generic linear readout neuron, which is trained for some specific computational task. The value of $\mathbf{y}(t)$ is obtained by applying a low-pass filter to the spike trains emitted by neurons in the generic neural microcircuit model. Note that this is not biologically unrealistic as it has been shown previously that such weighted sums contain behavior-relevant information [84].

For the experiments described in this chapter, generic cortical microcircuit models consisting of integrate-and-fire neurons were used, with a high level of noise that reflects experimental data. Biologically realistic models of dynamic synapses were used whose individual mixture of pair-pulsed depression and facilitation (depending on the type of pre- and postsynaptic neuron) was based on experimental data [65, 66]. These circuits were not created for any specific computational task. Sparse synaptic connectivity between neurons was generated (with a biologically realistic bias towards short-range connections) by a probabilistic rule, and synaptic parameters were chosen randomly from distributions that depended on the type of pre- and postsynaptic neurons (in accordance with empirical data from [65, 66]). The neurons in the generic cortical microcircuit models were placed on the integer-points of a 3-D grid, and 20% of these neurons were randomly chosen to be inhibitory. 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. Depending on whether the pre- and postsynaptic neurons were excitatory (E) or inhibitory (I), the value of C was set according to [66] to 0.3 (EE), 0.2 (EI), 0.4 (IE), 0.1 (II).

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, constant non-specific background current I_b uniformly drawn from the interval $[13.5, 14.5]$ nA for each neuron, noise at each time-step I_{noise} drawn from a Gaussian distribution with mean 0 and SD chosen for each neuron randomly from a Gaussian distribution over the interval $[4.0, 5.0]$ nA, input resistance 1 $M\Omega$, the initial condition for each neuron, i.e. its membrane potential at time $t = 0$, was drawn randomly (uniform distribution) from the interval $[13.5, 14.9]$ mV.

Synaptic Parameters. The short term dynamics of synapses was modeled according to the model proposed in [65], 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. This model predicts the amplitude A_k of the EPSC for the k^{th} spike in a spike train with interspike intervals $\Delta_1, \Delta_2, \ldots, \Delta_{k-1}$ through the equations

$$\begin{aligned} A_k &= w \cdot u_k \cdot R_k \\ u_k &= U + u_{k-1}(1-U)exp(-\Delta_{k-1}/F) \\ R_k &= 1 + (R_{k-1} - u_{k-1}R_{k-1} - 1)exp(-\Delta_{k-1}/D) \end{aligned}$$

with hidden dynamic variables $u \in [0,1]$ and $R \in [0,1]$ whose initial values for the first spike are $u_1 = U$ and $R_1 = 1$ (see [82] for a justification of this version of the equations, which corrects a small error in [65]). Depending on whether a and b were excitatory (E) or inhibitory (I), the mean values of the three parameters U, D, F (with D,F expressed in seconds, s) were chosen according to [66] to be 0.5, 1.1, 0.05 (EE), 0.05, 0.125, 1.2 (EI), 0.25, 0.7, 0.02 (IE), 0.32, 0.144, 0.06 (II). The SD of each of these parameters was chosen to be 50% of its mean. The mean of the scaling parameter w (in nA) was chosen to be 70 (EE), 150 (EI), -47 (IE), -47 (II). In the case of input synapses the parameter w 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 parameter w was chosen to be 70% of its mean and was drawn from a gamma distribution. The postsynaptic current was modeled by an exponential decay $\exp(-t/\tau_s)$ with $\tau_s = 3\,\text{ms}$ ($\tau_s = 6\,\text{ms}$) for excitatory (inhibitory) synapses. The transmission delays between neurons were chosen uniformly to be 1.5 ms (EE), and 0.8 ms for the other connections.

4.3 Results

The experiments for tasks described in this chapter consisted of two distinct phases. In the first phase linear readouts that received inputs from the circuit, were trained in an open-loop fashion to perform diverse computational tasks (e.g. to make decisions, to generate a motor command or to predict the joint angles during the arm movement). During open-loop training, the feedback from readouts performing diverse computational tasks were simulated by a *noisy* version of their target output ("teacher forcing"). More precisely, at each time-step t, a different noise value of $0.0001 \times \rho \times f(t)$ was added, where ρ is a random number drawn from a Gaussian distribution with mean 0 and SD 1, and $f(t)$ is the current value of the feedback signal (signal-dependent noise). Note that teacher forcing with noisy versions of target feedback values trains these readouts to correct errors resulting from imprecision in their preceding feedback (rather than amplifying errors). Each readout neuron was trained by linear regression to output at any time t, a particular target value $f(t)$. Linear regression was applied to a set of data points of the form $\langle \mathbf{y}(t), f(t) \rangle$, for many time points t, where $\mathbf{y}(t)$ is the output of low-pass filters applied to the spike-trains of pre-synaptic neurons, and $f(t)$ is the target output. Note that this form of training keeps the internal dynamics of the neural circuit intact, and only modifies the weight vector $\mathbf{w}$, that corresponds to the projection of the circuit dynamics onto a linear readout for the generation of any particular computational task. Performance of a readout was measured in terms of the correlation value of its target and observed signals during any trial for all experiments reported in this chapter.

The training phase was followed by a subsequent closed-loop validation phase. During validation, teacher-signals to the generic neural microcircuits were replaced by actual feedback from trained readouts. A standard form of population coding [100] was employed to encode feedback signals. Each feedback signal was mapped onto an array of 50 symbolic neurons with bell-shaped tuning curves[1]. Thus the value of feedback signal is encoded at any time by the output values of the associated 50 symbolic neurons (of which at least 43 neurons output at any time the value 0).

In principle, if significantly larger circuits are used, thereby providing the neural microcircuits with more robust kernels, then one can also send this feedback directly to the neurons in the generic neural microcircuit. Nevertheless, population coding of feedback variables is suitable in context of this chapter as previous studies demonstrate that neurons in PFC process information via a population code [101, 102, 103]. Also population coding in M1 neurons has been shown to predict movement direction in 2-D [104], as well as 3-D [105, 106, 107, 108] space, movement trajectories [109, 110, 111], and several other variables crucial for motor control [112, 113, 114, 115].

4.3.1 The two-interval discrimination task.

In the actual experimental protocol for two-interval discrimination task, a pre-trained subject is presented with two frequencies f_1 and f_2, separated by a certain delay interval. Initially the frequency f_1 is loaded into working memory, and its value is maintained during the delay phase, and on presenting the f_2 frequency, the subject is required to decide whether "$f_1 > f_2$?". A key feature of these

[1]The following convention was applied. Each feedback signal is first scaled into the range [0,1]. This range is linearly mapped onto an array of 50 symbolic neurons. At each time step, one of these 50 neurons, whose number $n(t) \in \{1, \ldots, 50\}$ reflects the current value $f_n(t) \in [0, 1]$ which is the normalized value of the feedback signal $f(t)$ (e.g. $n(t) = 1$ if $f_n(t) = 0$, $n(t) = 50$ if $f_n(t) = 1$). The neuron $n(t)$ then outputs at time t the value of $f(t)$. 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 $f(t) \cdot \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(n-n(t))^2}{2\pi^2}}$, where $\sigma = 0.8$).

vibrotactile sensory inputs arriving from secondary somatosensory cortex (S2) to PFC is that they arrive through identical neural pathways [28], and are encoded by simple monotonic coding [22, 116, 117]. Two kinds of neurons have been observed in PFC which show opposite activities in response to the above question [22]. The first type of neurons, called "+" neurons from now, show an increase in their activity during the D phase, when the answer to the above question is "yes", and the second type of neurons, called "-" neurons from now, show an increase in their neural activity when the answer to the above question is "no". The information required to carry out the task is present in the firing activity of "+" and "-" neurons independently, and the reason for the simultaneous presence of both these sets is till now unknown [22], though it has been suggested by Romo's group that this may be important for robustness in decision making.

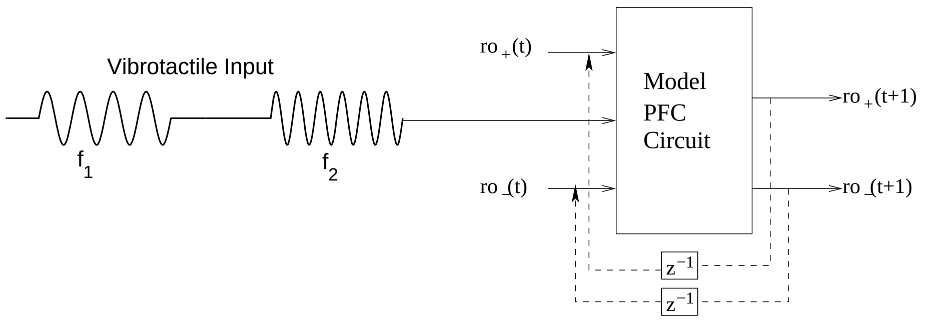

Figure 4.1: Closed-loop setup for the two-interval discrimination task. The weight vectors of the "+" and "-" readouts from this circuit, that produce the decision signal, are the only parameters that are adjusted during training. The model PFC circuit is made of 300 integrate-and-fire neurons arranged on the integer points of a $20 \times 5 \times 3$ cube. The circuit receives 2 frequencies $\langle f_1,\ f_2 \rangle$ as external input through the same input lines that have been encoded using a simple monotonic code, and two feedback signals $\langle ro_+(t),\ ro_-(t) \rangle$ from the "+" and "-" readouts. The task is to answer the question if "$f_1 > f_2$? The "+"("-") neurons show an increase in their activity when the answer to the above question is "yes"("no"). The notation z^{-1} denotes a unit time-step delay in the feedback signal.

In this setup, the "+" and "-" neurons are modeled as simple linear readouts that send feedback of their activity to the neural circuit (see Fig. 4.1). The generic neural microcircuit used in this experiment consisted of 300 integrate-and-fire neurons arranged on the grid-points of a $20 \times 5 \times 3$ cube in $3D$. In addition to the feedback, the circuit receives external sensory input composed of 2 frequencies (f_1 and f_2). These vibrotactile frequencies f_1 and f_2 are presented during the L and D phases respectively (see Fig. 4.1 and also Fig. 4.2 A)

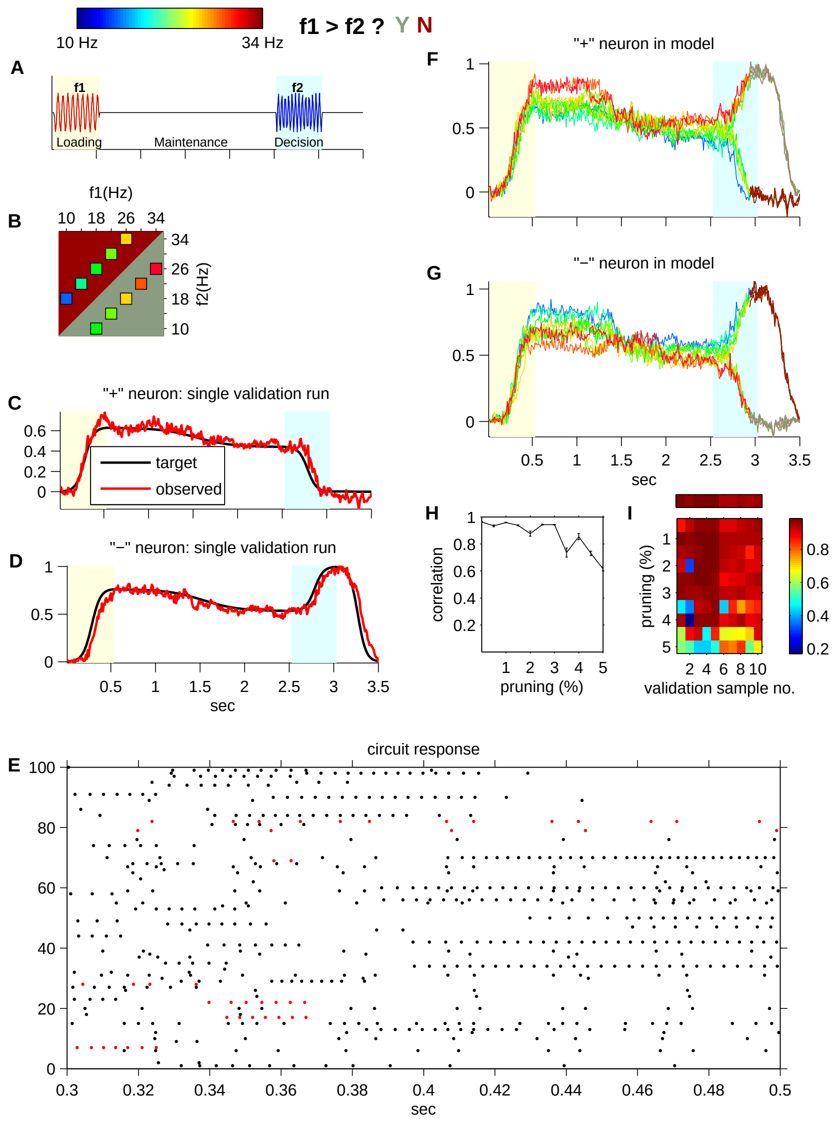

Figure 4.2: **(A)** External frequencies $\langle f_1 = 18, f_2 = 26\rangle$ Hz) for one of the close-loop validation trials. **(B)** Stimulus frequencies used in this study. The target (black) and observed (red) values for the **(C)** "+" readout, and the **(D)** "-" readout. **(E)** A blowup of 200 ms of resulting firing activity of 100 randomly chosen neurons in the circuit. Responses of the **(F)** "+" and **(G)** "-" readouts for each of the frequency pairs. The colorbar at upper left indicates the value of f_1 used in each trial. **(H)** Mean and standard error of correlation values (between the target and observed values) of "+" readout for trials with progressively higher pruning percentage. **(I)** The resulting matrix of correlation values where each square shows the mean correlation value over 10 runs for a particular frequency pair, for a particular pruning percentage. The control correlation values (no pruning) are shown in the row on the top.

through the same input lines, and are coded using simple monotonic coding[2]. In the training phase the circuit was trained for 100 trials with the training data composed of 10 noisy versions of each of the 10 input pairs $\langle f_1, f_2 \rangle$ (see Fig. 4.2 B). The target functions of "+" and "-" readouts are as described above (see Appendix for details). During each trial[3], f_1 and f_2 are presented for 0.5 s each, during the L and D phases respectively. In the closed-loop validation phase, the performance was validated for 100 trials, with the external sensory inputs for these validation trials being 10 noisy versions of each of the 10 input frequency pairs (not seen during training).

Fig. 4.2 E shows a 200 ms blowup of the circuit response of 100 randomly chosen neurons (activity of inhibitory neurons shown in red) during one of the closed-loop validation runs ($f_1 = 18$ Hz, $f_2 = 26$ Hz). The panels C and D of Fig. 4.2 show the target (black) and observed (red) values of the "+" and "-" readouts during this run[4]. Panel F and G show the closed-loop validation response of the "+" and "-" readouts for the 10 pairs of input frequencies, $\langle f_1, f_2 \rangle$ (note the similarity to Fig. 1, panels C and D in [22], which show the actual data from "+" and "-" neurons in PFC during the two-interval discrimination task).

Robustness. To test the robustness of the neural model, experiments were done where after the readouts have been trained, a subset κ_n (κ_n progressively increased from 0.5% to 5% in 10 trials such that $\kappa_n \subset \kappa_{n+1}$) of synapses converging onto the "+" readouts were randomly chosen and pruned (synaptic weight set to 0). The resulting setup was tested for 100 trials using 10 noisy versions of each of the 10 frequency pairs $\langle f_1, f_2 \rangle$[5]. Panels H and I of Fig. 4.2 show the result of these robustness tests. Panel H shows the mean and standard error of correlation values between the target and observed values of "+" neuron for progressively higher levels of pruned synapses[6]. Panel I shows the resulting matrix of same correlation values, where each square shows the mean correlation value for 10 validation runs using noisy versions of a particular frequency pair $\langle f_1, f_2 \rangle$, for a particular pruning percentage. The control correlation values (no pruning) are shown in the row on the top (over 100 validation runs, mean = 0.9625,

[2]The external input composed of f_1 and f_2 was encoded by a simple monotonic code by a set of 50 input neurons with variable transmission delays Δ, uniformly drawn from the interval [0 10] ms.

[3]Each trial lasted for 3.5 s, with a simulation time-step of 10 ms. The duration of L, M, and D phases were 0.5, 2.0, and 0.97 seconds respectively. The cue stimulus f_1 was presented for 0.5 s during the L phase (starting at $t = 0.03$ s), which was followed by the M phase (starting at $t = 0.53$ s). The probe stimulus was presented for 0.5 s from the start of D phase (starting at $t = 2.53$ s).

[4]With the correlation values between target and observed signals being 0.97976 for the "+" neuron and 0.954118 for the "-" neuron respectively.

[5]Using the same 100 pairs of input frequencies $\langle f_1, f_2 \rangle$ for all levels of pruning.

[6]For 100 validation runs for each level of pruning.

SD = 0.0173). Results indicate that the model shows graceful degradation in presence of suboptimal inputs, and the performance of this model is comparable to other more traditional models of attractor based neural computation [25].

4.3.2 Delayed-match-to-sample followed by a goal-directed arm movement.

The decisions taken in the PFC region are acted upon by the motor commands issued by the M1 neurons of the sensorimotor system. To demonstrate that the neurocomputational paradigm presented in this chapter is capable of integrating the A phase, and is task independent, delayed-match-to-sample was chosen as an alternative interval discrimination paradigm. In this task, a pre-trained subject is presented with a cue visual stimulus during the L phase (for example a small colored square on a predetermined region of the screen), which is followed by a delay period (M phase), and subsequently in the D phase two simultaneous probe stimuli are presented at two different places on the screen, and the subject is required to decide which of the probe stimuli has the same color as the cue stimulus, by pressing one of the two buttons corresponding to the binary choice in the A phase [23].

Fig. 4.3 shows the setup used in this experiment[7]. For added biological realism, the neurocomputational function of PFC and M1 were modeled using two separate generic cortical microcircuits. The circuit modeling PFC function consisted of 500 integrate-and-fire neurons arranged on the grid points of a $20 \times 5 \times 5$ cube and the circuit modeling M1 was made of 1000 integrate-and-fire neurons arranged on the grid points of a $20 \times 5 \times 10$ cube.

The model PFC circuit received 3 external inputs (C_{cue}, the cue color; C_{left}, the left probe color; C_{right}, the right probe color), and 2 feedback signals (from the “left” and “right” readouts). The cue stimulus was presented during the L phase and the probe stimuli were presented during the D phase, at 3 different locations on the workspace of the arm (see Fig. 4.5 panels A and B). The neurons in the model PFC circuit made projections to two linear readouts (called “left” and “right” readouts from now on) which had similar functionality as the “+” and “-” readouts in the two-interval discrimination task. The “left” readout showed high activity during the D phase, if the answer to the question “$C_{cue} = C_{left}$?” was positive. The “right” readout behaved exactly opposite to this and showed increased activity in the D phase, if the probe stimulus shown on the right

[7]For this experiment, the external inputs were encoded using population coding in the similar fashion as the feedback from trained readouts. For details of the coding paradigm, see section 4.2.

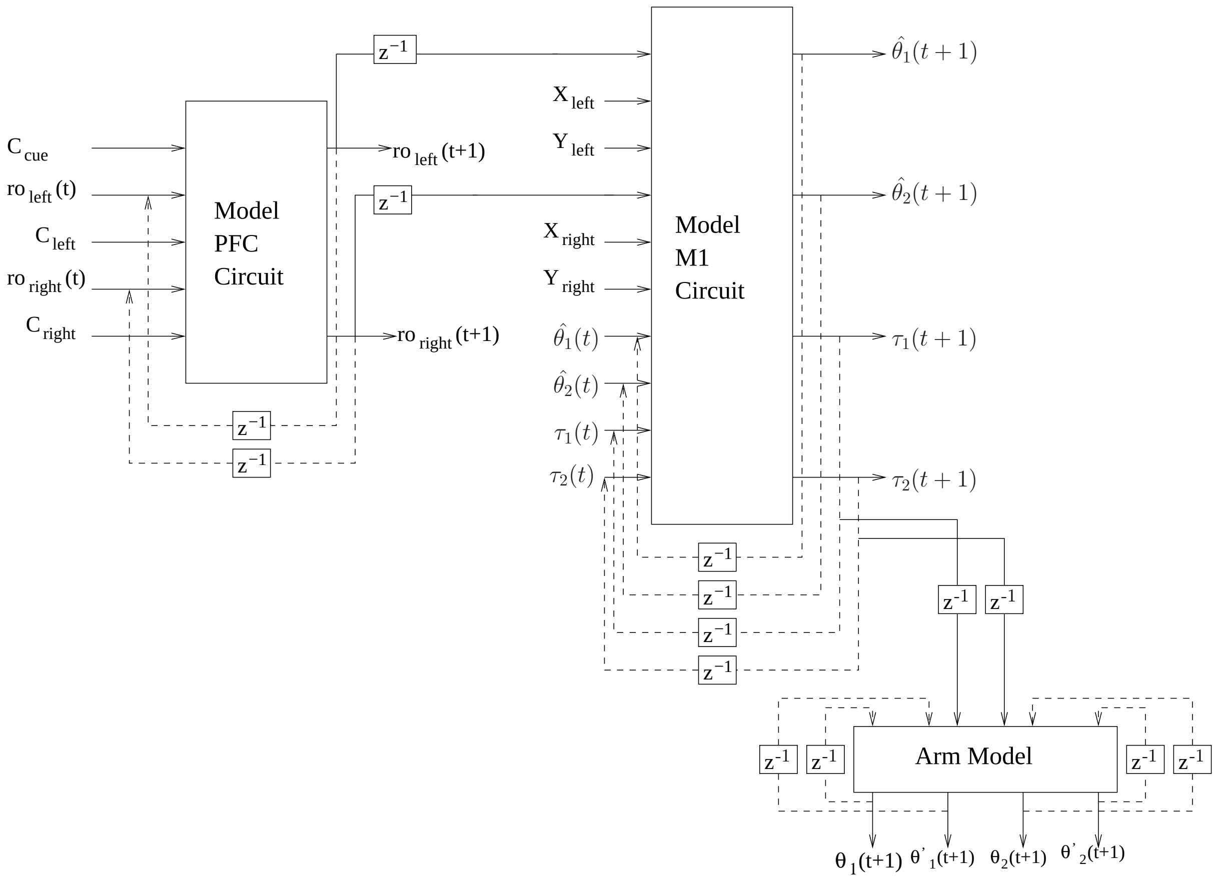

Figure 4.3: Block diagram depicting the setup used in the delayed-match-to-sample experiment. The circuit modeling PFC function consisted of 500 integrate-and-fire neurons arranged on the grid points of a $20\times5\times5$ cube and the circuit modeling M1 was made of 1000 integrate-and-fire neurons arranged on the grid points of a $20\times5\times10$ cube. The model PFC circuit received 3 color stimulus $\langle C_{cue}, C_{left}, C_{right}\rangle$ as external inputs, and 2 feedback signals $\langle ro_{left}(t), ro_{right}(t)\rangle\rangle$ from the "left" and "right" readouts. The "left"("right") neurons show an increase in their activity when $C_{cue} = C_{left}(C_{right})$. In addition to the response of the "left" and "right" readouts, the model M1 circuit receives 4 additional external inputs (X_{left}, Y_{left}, X_{right}, Y_{right}, the X and Y coordinates of the center of the left and right probe regions), and 4 feedback signals ($\hat{\theta_1}(t)$, $\hat{\theta_2}(t)$, $\tau_1(t)$, $\tau_2(t)$). The task is to move the arm to the center of the probe region that matches in color to the cue stimuli.

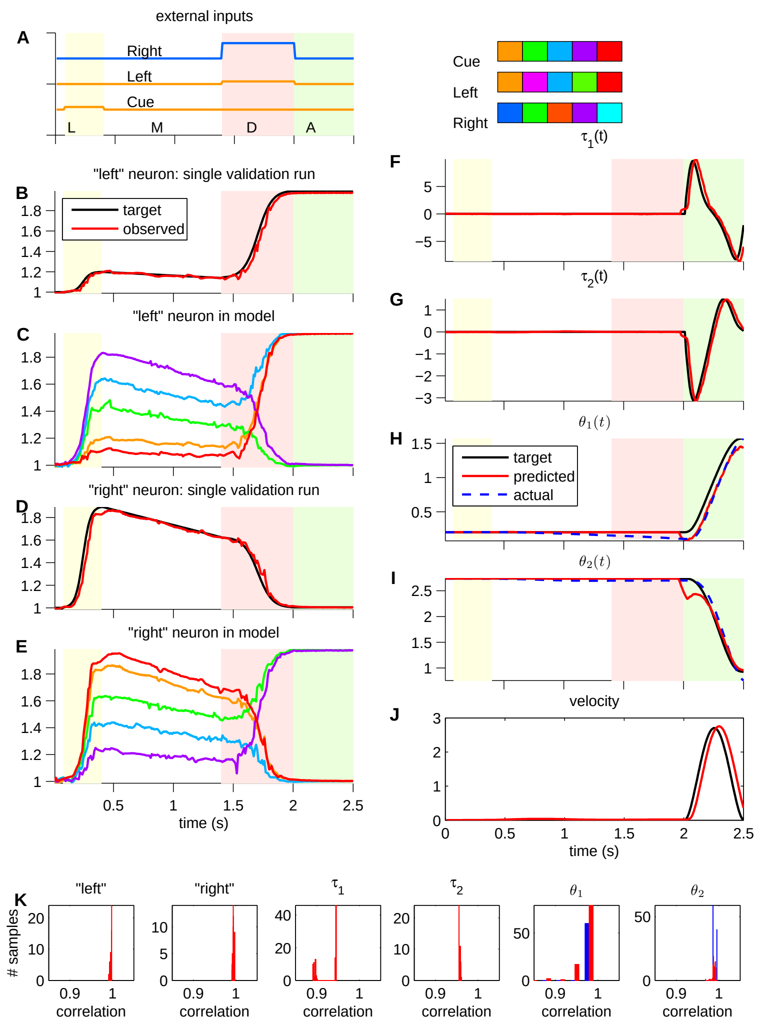

Figure 4.4: **(A)** The left column shows external stimuli to the model PFC circuit during one of the closed loop validation runs. The right column shows the colors of the cue, left and right stimuli used in trials. The target (black) and observed (red) values for the **(B)** "left" readout, and the **(D)** "right" readout. Responses of the **(C)** "left" and **(E)** "right" readouts for each of the color triplets. Target (black) and observed (red) motor commands (joint torques) generated in the A phase for the **(F)** shoulder and **(G)** elbow joints. The target (black), predicted (red) and actual (blue dashed) values for the joint angles for the **(H)** shoulder and **(I)** elbow joints. **(J)** The movement of the tip of the robot arm from the resting phase to the decided end-point occurred with a biologically realistic bell-shaped velocity profile. **(K)** Histogram of correlation values between target and observed signals for readouts that made decisions ("left" and "right"), issued motor commands $\langle \tau_1(.), \tau_2(.) \rangle$, and predicted the joint angles $\langle \hat{\theta}_1(.), \hat{\theta}_2(.) \rangle$. The last two plots from the right in panel K also show the histogram of correlation for between the actual joint angles and their target values (shown in blue).

matched the cue stimulus. Please see Appendix for details about target functions for these readouts.

The output of the "left" and "right" readouts were sent as inputs to the model M1 circuit. In addition, the circuit also received 4 external inputs (X_{left}, Y_{left}, X_{right}, and Y_{right}; the X and Y coordinates of the center of the left and right probe regions), and 4 feedback signals (2 motor and 2 proprioceptive feedbacks). The arm model used in this experiment (for details, see Appendix) is the standard model of a 2-joint robot arm described in [16].

The neurons in the model M1 circuit made projections to 4 linear readouts. Two of these readouts $\langle\hat{\theta_1}(.),\ \hat{\theta_2}(.)\rangle$ were trained to predict the current values of the shoulder and elbow joint angles. The other 2 readouts $\langle\tau_1(.),\ \tau_2(.)\rangle$ were trained to generate joint torques to drive the arm from its resting position to the center of the probe region that matched the stimuli (see Fig. 4.5 panels A and B).

During the open-loop training, the circuit was trained for 100 trials[8] using 10 noisy versions of each of the 5 input color triplets $\langle C_{cue}, C_{left}, C_{right}\rangle$ (see Fig. 4.4 A, right side). The target trajectory followed by the tip of the arm was generated using the minimum-jerk model (for details see Appendix) described in [91]. For the closed loop validation phase, 100 trials were performed using 20 noisy versions of each of the 5 input color triplets.

Fig. 4.4 shows the result of a closed loop validation run. The target (black) and observed (red) response of the "left" and "right" readouts are shown in panel B and D respectively. The panels C and E of Fig. 4.4 show the response of the "left" and "right" readouts for the 5 input stimuli (each line drawn in the color of corresponding cue stimulus). Panels F and G show the target (black) and observed (red) motor commands (joint torques) generated in the A phase. Panels H and I show the target (black), predicted (red) and actual (blue dashed) values for the joint angles. The movement of the tip of the robot arm from the resting phase to the decided end-point occurred with a biologically realistic bell-shaped velocity profile shown in panel J.

Robustness. The columns in panel K of Fig. 4.4 demonstrate the performance of this setup over 100 closed-loop validation runs. More precisely, these result show the histogram of correlation values between the target and observed signals for the "left" (mean = 0.9962, SD = 0.0018) and "right" (mean = 0.9942, SD

[8]Each trial lasted for 2.5 s, with a simulation time-step of 10 ms. The duration of L, M, D and A phases were 0.32, 1.0, 0.6, and 0.5 seconds respectively. The cue stimulus C_{cue} was presented for 0.32 s during the L phase (starting at $t = 0.08$ s), which was followed by the M phase (starting at $t = 0.4$ s). C_{left} and C_{right} are presented simultaneously for 0.6 s from the start of D phase (starting at $t = 1.4$ s). The A phase (starting at $t = 2.0$ s) consisted of moving the arm from the initial resting position to the decided position in 0.5 s (see Fig. 4.4, panel A, left side)

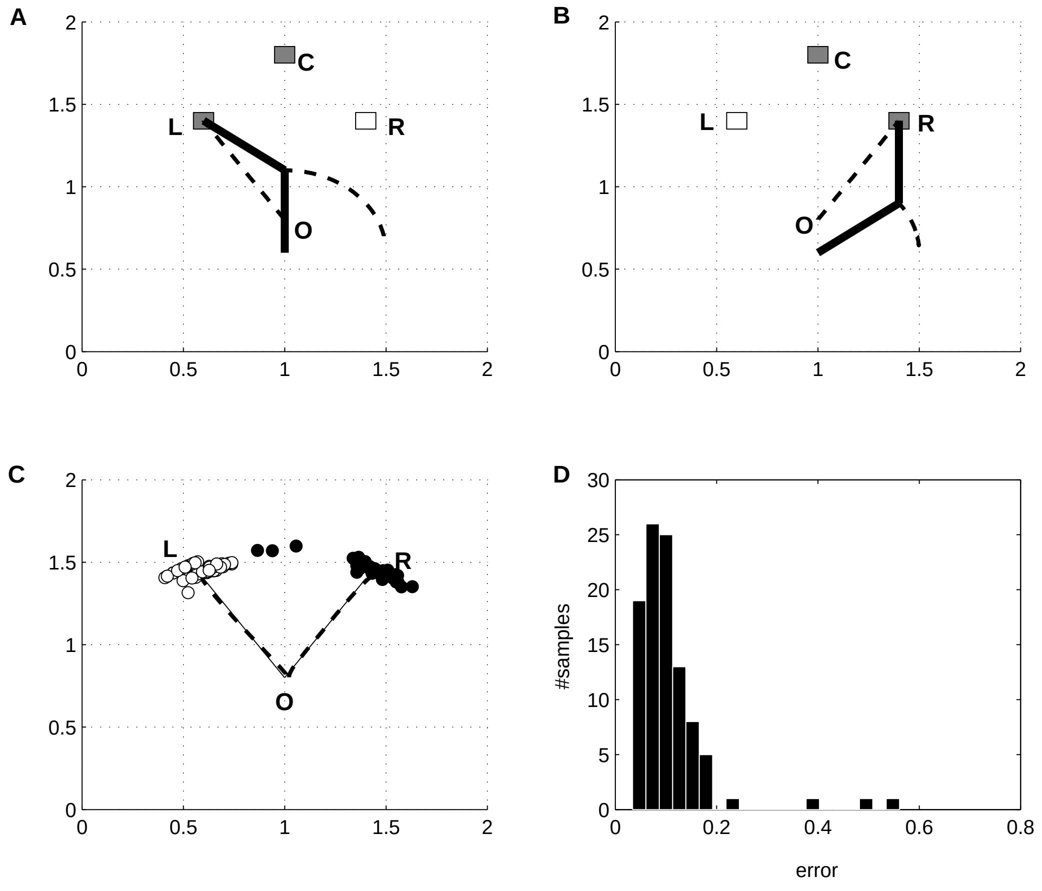

Figure 4.5: Typical movement performed by the arm when the cue stimulus matches the stimulus shown on the **(A)** left, or **(B)** right probe location. In both panels, O denotes the position of the end point of arm before movement initiation. L, R and C denote the position of the left, right, and cue stimulus. **(C)** Target trajectories of the tip of the arm during a movement to the left or right probe location (solid), and the actual trajectories for one validation run to the left (dashed) and one to the right (dashed). The open (closed) circles represent the location of the end-point for other trials at the end of movement when the target location was the center of the left (right) probe region. **(D)** Histogram of error values for the 100 trials, where the error was defined as the Euclidean distance between the target and observed end-point of the arm.

= 0.0018) readouts, the readouts that computed the joint torques ($\tau_1(.)$: mean = 0.9246, SD = 0.0247; $\tau_2(.)$: mean = 0.9571, SD = 0.0019), and the readouts that predicted the joint angles ($\hat{\theta_1}(.)$: mean = 0.6344, SD = 0.4411; $\hat{\theta_2}(.)$: mean = 0.9899, SD = 0.0047). The last two plots from the right in panel K also show the histogram of correlation for the actual joint angles with the target (shown in blue, for θ_1 : mean = 0.9733, SD = 0.0282, for θ_2 : mean = 0.9887, SD = 0.0035).

The panels A (B) of Fig. 4.5 shows typical movement performed by the arm when the cue stimulus matches the stimulus shown on the left (right) probe location. In both panels, O denotes the position of the end point of arm before movement initiation. L,R and C denote the position of the left, right, and cue stimulus. Panel C of Fig. 4.5 shows the target trajectories of the tip of the arm during a movement to the left or right probe location (solid), and the actual trajectories (dashed) for one validation run to the left and one to the right. The open (closed) circles represent the location of the end-point for other trials, at the end of movement, when the target end point was the center of the left (right) probe region. Panel D shows the histogram of error values for the 100 trials, where the error was defined as the Euclidean distance between the target and observed end-point of the arm, for each trial (mean = 0.1083 m, SD = 0.0784 m).

4.3.3 Memory and decision signals from untrained neurons in neural microcircuits

Recordings from randomly chosen task-related neurons in PFC show a strong correlation to memory and decision signals during a working memory task [118, 22]. The results shown in Fig. 4.6 show that randomly chosen neurons from generic neural microcircuits used in both the tasks discussed in this chapter, also contain strong correlation to the working memory and decision signals. Panel A of Fig. 4.6 shows the target (solid) and observed (dashed) performance of the "+" readout during one of the closed-loop validation runs[9] of the two-interval discrimination task. Panel B shows a mean peri-stimulus time histogram (PSTH) derived from responses of 50 randomly chosen neurons from the layer of 100 neurons that received this feedback signal in the neural microcircuit used in this experiment. It was observed that the PSTH signal has strong correlation[10] with the observed value of the "+" readout indicating that the neurons in the generic circuit contained information about the decision signal. Panel C and D show similar results for a closed-loop validation run during the delayed-match-to-sample task for the

[9]With the correlation between target and observed signal being 0.979545.

[10]With the correlation between the PSTH signal and the observed value of the "+" readout being 0.903421.

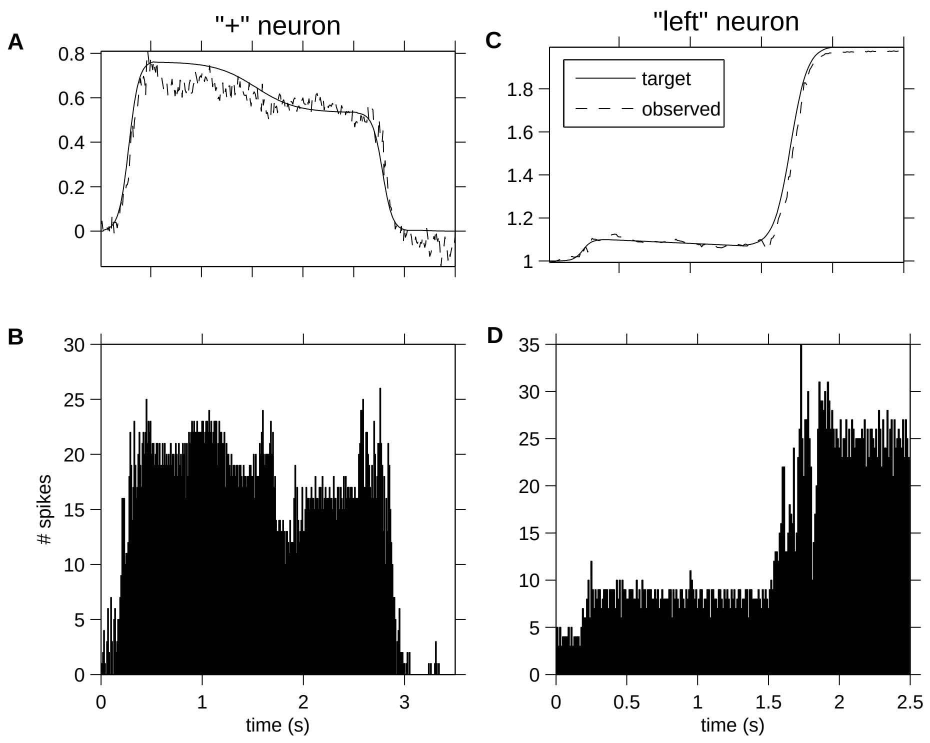

Figure 4.6: Memory and decision signals from untrained neurons in neural microcircuits. **(A)** The target (solid) and observed (dashed) performance of the "+" readout during one of the closed-loop validation runs for the two-interval discrimination task. **(B)** The PSTH obtained by recording the activity of 50 randomly chosen neurons from the layer of 100 neurons that received this feedback signal in the neural microcircuit used in this experiment. Strong correlation was observed between the PSTH signal and the observed value of the "+" readout indicating that the neurons in the generic circuit contained information about the decision signal. Panel C and D show similar results for the "left" readout, during a closed-loop validation run for the delayed-match-to-sample task.

"left" readout[11]. It is to be noted that unlike the response of trained readouts, the PSTH's shown in panel B and D show the recordings from neurons in the circuits which were not trained for any specific tasks.

4.4 Discussion

This chapter describes a new neurocomputational paradigm that uses synaptic learning mechanisms to present a unified model for decision making followed by action selection. The model is unified in the sense that identical learning algorithm (simple linear regression) is used to train readouts that make decisions based on the activity of model PFC circuit, and the readouts that generate motor commands or predict the joint angles based on the activity of the model M1 circuit. Biologically realistic neural microcircuit models composed of spiking neurons and dynamic synapses were used to generate models for two different interval discrimination paradigms. Initially, for comparison with other models of working memory, a model was presented in section 4.3.1 for the two-interval discrimination task that incorporated the L, M and D phases involved in working memory and decision making. Additionally to show that this paradigm can be extended to incorporate the A phase, is robust of how external inputs are encoded, and is task independent, another model was presented in section 4.3.2 for the delayed-match-to-sample task with a goal directed movement of a two-jointed robot arm during the A phase. Note that the A phase for the two-interval discrimination task could easily have been implemented in an identical manner to the delayed-match-to-sample task, but was avoided to prevent redundancy.

A prominent feature of this neural model is that synaptic learning only occurs at synapses that project the activity in the neural microcircuit models onto the linear readout neurons, leaving the circuit dynamics intact. A key advantage of this learning mechanism is that since only the readouts, and not the neural circuit itself, have to adapt to specific computational tasks, the same circuit can support completely different computations in parallel. 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 computational tasks. However this has turned out to be not necessary for the tasks described in this chapter, although it remains an interesting open research problem how unsupervised learning could optimize a circuit for some particular computational task.

It is to be noted however that although spiking neural circuits were used to model the interval-discrimination tasks for added biological realism, it is not a

[11]With the correlation value between the target and observed signal being 0.99789, and the correlation between the observed value and the PSTH being 0.909589.

requirement, and any recurrent circuit would give similar results, as long as it has the kernel property. Readouts make a binary decision by reaching one of the states corresponding to the decision made by them. The actual point of time when the readout makes a decision can be thought of as a threshold crossing event, i.e. the first time when the readout crosses a threshold after the presentation of the probe stimulus in the D phase.

It should be noted that in principle, the task described in section 4.3.2 can also be performed by a single large neural microcircuit. Instead 2 separate neural microcircuits were used to model the functionality of PFC and M1 individually for added biological realism, with the synaptic input from PFC to M1 being optimized by linear regression. Note that the sparse, long-range synaptic projections from M1 to PFC were not modeled, as they were not required for this task. More precisely, these recurrent projections were not modeled since the decision making process is not influenced by the current action performed, so this feedback had no computational relevance. The author is confident that adding these synapses would not have changed the performance, but this was not tested.

The feedback from the trained readouts played an apparently important role in the neural model described above. Although the generic neural microcircuits used to model the PFC and M1 circuits are endowed with fading memory due to the kernel property of the circuits, this is not sufficient for holding information in working memory for longer timespans ranging in the order of seconds. Apparently the feedback from trained readouts provides the circuit with additional needed information that falls outside the window of fading memory, hence enhancing the information present in the circuit dynamics.

Obviously closed-loop applications of generic neural microcircuit models like the ones discussed in this chapter present a harder computational challenge than open-loop sensory processing tasks, since small imprecisions in their output are likely to be amplified by the plant (e.g. the arm model) to yield even larger deviations in the feedback, which is likely to further enhance the imprecision of subsequent outputs. This problem can be solved by teaching the readouts from the neural microcircuit during training to ignore smaller recent deviations reported by feedback, thereby making the target trajectory of output torques an attractor in the resulting closed-loop dynamical system. It is not claimed that such teacher-signals are biologically realistic, but however they can be interpreted for example in context of movement generation, as the movements shown by an instructor during learning of a new motor-skill ("imitation learning").

One of the potential drawbacks of earlier models of working memory [22, 28] is their inability to capture the temporal dynamics of real PFC neurons during the delay period [21]. While the real PFC neurons show a monotonic increase or decrease in their firing rate during the D phase, the neurons in these models have a stationary firing rate in this phase. The neural algorithm presented in this chapter is able to overcome this challenge, as the readouts can successfully "learn" to decode the value of the original cue stimuli from its temporally decayed value.

According to the "internal model" hypothesis [119, 120], there exists an internal model for each of the tasks that we have learned throughout our lives. One outcome of such a hypothesis would be that a given neuron may participate with a different synaptic weight in a number of neural assemblies, each supporting a different internal model. Interestingly, this is reflected in our setup too, as neurons in the generic neural circuit make synaptic projections to the set of readouts with different synaptic weights assigned for each task.

This study also demonstrates the ability of generic neural microcircuit models to hold "partial attractor" states in their circuit dynamics for significantly longer and biologically relevant time-scales ranging in the order of a couple of seconds, even in the presence of noise. The role of feedback in enhancing the inherent fading memory of a neural circuit is also demonstrated. Further it also shows the potential of generic neural circuits to integrate the mechanisms involved in the sense-think-decide-act sequence involved in decision making followed by action selection, that happen at significantly longer time-scales. The idea of an uniform modus-operandi across cortical regions is not only attractive, but also intuitively plausible. Further work is needed to explore the ideas presented here in detail.

4.5 Materials and methods

The model for the two-joint robot arm. The model can be described using the well known Lagrangian equation in classical dynamics. The dynamic equations for this arm model are given by equation 4.1:

$$\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -h\dot{\theta}_2 & -h(\dot{\theta}_1 + \dot{\theta}_2) \\ h\dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{4.1}$$

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 to the two joints, and

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

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

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

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

Equation 4.1 can be compactly written 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. I_1, I_2 are the moments of inertia of the two joints. The values of the parameters that were used in 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 minimum-jerk model. For a given start point $\langle x_{start}, y_{start}\rangle$ and target end point $\langle x_{dest}, y_{dest}\rangle$ of a movement (both given in Cartesian coordinates), an interpolating trajectory of the tip of the arm was generated according to the following equation given in [91]

$$x(t) = x_{start} + (x_{start} - x_{dest}) \cdot (15\zeta^4 - 6\zeta^5 - 10\zeta^3)$$

$$y(t) = y_{start} + (y_{start} - y_{dest}) \cdot (15\zeta^4 - 6\zeta^5 - 10\zeta^3)$$

where $\zeta = t/MT$ and MT is the target movement time (in this case $MT = 500$ ms). From this target trajectory for the endpoint of the robot arm, the target trajectories of the joint angles θ_1, θ_2 of the robot arm were generated by applying standard equations from geometry (see e.g. [121]). From these the target trajectories of the torques were generated according to equ. (4.1).

The target values for the "+" and "-" neurons. The target values for these readouts were 0 before the onset of L phase. Following this the target values for these readouts during L, M, and D phases were modeled as simple scaled sigmoids. More precisely, for a pair of input frequencies, $\langle f_1, f_2\rangle$, the target value for the "+" readout, $f_+(t)$ is given by:

$$f_+(t) = \begin{cases} 0 & 0 \le t < 0.03 \text{ s} \\ \dfrac{(f_1+20)}{1+e^{-(-5+10\cdot\frac{t-0.03}{0.5})}} & 0.03 < t \le 0.53 \text{ s} \\ 0.7\cdot(f_1+20) + \left(\dfrac{0.3\cdot(f_1+20)}{1+e^{-(5-10\cdot\frac{t-0.53}{2})}}\right) & 0.53 < t \le 2.53 \text{ s} \\ 0.7\cdot(f_1+20) + \left(\dfrac{60-0.7\cdot(f_1+20))}{1+e^{-(-5+10\cdot\frac{t-2.53}{0.97})}}\right) & 2.53 < t \le 3.5 \text{ s},\ f_1 > f_2 \\ \dfrac{0.7\cdot(f_1+20)}{1+e^{-(5-10\cdot\frac{t-2.53}{0.97})}} & 2.53 < t \le 3.5 \text{ s},\ f_1 < f_2 \end{cases} \tag{4.2}$$

And in similar fashion, the target value for the "-" readout, $f_{-}(t)$ is given by:

$$f_{-}(t) = \begin{cases} 0 & 0 \leq t < 0.03 \text{ s} \\ \dfrac{(64 - f_1)}{1 + e^{-(-5+10\cdot\frac{t-0.03}{0.5})}} & 0.03 < t \leq 0.53 \text{ s} \\ 0.7 \cdot (64 - f_1) + \left(\dfrac{0.3 \cdot (64 - f_1)}{1 + e^{-(5-10\cdot\frac{t-0.53}{2})}} \right) & 0.53 < t \leq 2.53 \text{ s} \\ \dfrac{0.7 \cdot (64 - f_1)}{1 + e^{-(5-10\cdot\frac{t-2.53}{0.97})}} & 2.53 < t \leq 3.5 \text{ s}, f_1 > f_2 \\ 0.7 \cdot (64 - f_1) + \left(\dfrac{60 - 0.7 \cdot (64 - f_1))}{1 + e^{-(-5+10\cdot\frac{t-2.53}{0.97})}} \right) & 2.53 < t \leq 3.5 \text{ s } f_1 < f_2 \end{cases} \tag{4.3}$$

The target functions for the "left" and "right" neurons. The target values for these readouts were 0 before the onset of L phase. Following this the target values for these readouts during L and D phases were modeled as simple scaled sigmoids. The target values during the M phase was modeled as a linear decay. The target value during the A phase was obtained by extrapolating the value reached during the D phase. The sample and probe colors were encoded by their hue values that were scaled in the range $[0, 1]$[12]. More precisely, for a color triplet $\langle C_{cue}, C_{left}, C_{right} \rangle$, the target value for the "left" readout, $f_{left}(t)$ is given by:

$$f_{left}(t) = \begin{cases} 0 & 0 \leq t < 0.08 \text{ s} \\ \dfrac{(C_{cue} + 0.1)}{1 + e^{-(-5+10\cdot\frac{t-0.08}{0.32})}} & 0.08 < t \leq 0.40 \text{ s} \\ (C_{cue} + 0.1) \cdot (1 - 0.3 \cdot (t - 0.40)) & 0.40 < t \leq 1.40 \text{ s} \\ 0.7 \cdot (C_{cue} + 0.1) + \left(\dfrac{1 - 0.7 \cdot (C_{cue} + 0.1))}{1 + e^{-(-5+10\cdot\frac{t-1.40}{0.60})}} \right) & 1.40 < t \leq 2.0 \text{ s}, C_{cue} = C_{left} \\ \dfrac{0.7 \cdot (C_{cue} + 0.1)}{1 + e^{-(5-10\cdot\frac{t-1.40}{0.60})}} & 1.40 < t \leq 2.0 \text{ s}, C_{cue} = C_{right} \\ f_{left}(2.0) & t > 2.0 \end{cases} \tag{4.4}$$

And in similar fashion, the target value for the "right" readout, $f_{right}(t)$ is given by:

[12]Keeping the saturation and intensity values at 100%

$$f_{right}(t) = \begin{cases} 0 & 0 \leq t < 0.08 \text{ s} \\ \dfrac{(1 - C_{cue})}{1 + e^{-(-5+10\cdot\frac{t-0.08}{0.32})}} & 0.08 < t \leq 0.40 \text{ s} \\ (1 - C_{cue}) \cdot (1 - 0.3 \cdot (t - 0.40)) & 0.40 < t \leq 1.40 \text{ s} \\ \dfrac{0.7 \cdot (1 - C_{cue})}{1 + e^{-(5-10\cdot\frac{t-1.40}{0.60})}} & 1.40 < t \leq 2.0 \text{ s}, C_{cue} = C_{left} \\ 0.7 \cdot (1 - C_{cue}) + \left(\dfrac{1 - 0.7 \cdot (1 - C_{cue}))}{1 + e^{-(-5+10\cdot\frac{t-1.40}{0.60})}}\right) & 1.40 < t \leq 2.0 \text{ s}, C_{cue} = C_{right} \\ f_{right}(2.0) & t > 2.0 \end{cases} \tag{4.5}$$

Chapter 5

Conclusions

Chapter 2 demonstrated that persistent memory and online switching of real-time processing can be implemented in generic cortical microcircuit models by training a few neurons (within or outside of the circuit) through very simple learning processes (linear regression, or alternatively – with some loss in performance – perceptron learning). The resulting high-dimensional attractors can be made noise-robust through training, thereby overcoming the inherent brittleness of constructed attractors. The high dimensionality of these attractors, which is caused by the small number of synaptic weights that are fixed for their creation, allows the circuit state to move in or out of other attractors, and to absorb new information from online inputs, while staying within such high-dimensional attractor. The resulting virtually unlimited computational capability of fading memory circuits with feedback can be explained on the basis of the theoretical results.

Chapter 3 demonstrated 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. 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 are well-suited for real-time computational tasks ([17], [18],[19], [94]). Whereas related work had so far focused on sensory processing tasks such as speech recognition or visual movement analysis ([17],[18], [95]) I 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. 3.7). 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 280 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 50 to 280 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. 3.8 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 3.2) as implemented for the experiments reported in this chapter. 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 [97], 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 chapter is somewhat related to preceding work [89], where abstract systems of differential equations were used, and to the melody-generation with artificial neural networks in discrete time of [20]. 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 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 chapter 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. I believe that the control framework presented in this chapter, 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 chapter 3 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 [19], 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 [122]). 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.

Chapter 4 presents a new neurocomputational paradigm that uses synaptic learning mechanisms to present a unified model for working memory and decision making using biologically realistic neural microcircuit models composed of spiking neurons and dynamic synapses. Additionally results for the two-interval-discrimination task show that the neural algorithm is task independent. It is to be noted however that although spiking neural circuits were used to model the interval-discrimination tasks for added biological realism, it is not a requirement, and any recurrent circuit would give similar results, as long as it has the kernel property.

Readouts make a binary decision by reaching one of the states corresponding to the decision made by them. The actual point of time when the readout makes a decision can be thought of as a threshold crossing event, i.e. the first time when the readout crosses a threshold after the presentation of the probe stimulus in the D phase.

It was found that using population coding to encode the inputs projecting on to model PFC circuits was essential to obtain the demonstrated results. Using a population of neurons to encode each analog input stream is not unrealistic, as sufficient evidence exists in current literature of its existence. It is however not claimed here that the precise mechanism of population coding used in this chapter is the one used in cortical circuitry of PFC.

The feedback from the trained readouts played an apparently important role in the neural model described above. Although the generic neural microcircuits used to simulate the model PFC circuit are endowed with fading memory due to the kernel property of the circuits, this is not sufficient for holding information in working memory for longer timespans ranging in the order of seconds. Apparently the feedback from trained readouts provides the circuit with additional needed information that falls outside the window of fading memory, hence enhancing the information present in the circuit dynamics.

This study also demonstrates the ability of generic neural microcircuit models to hold "partial attractor" states in their circuit dynamics for significantly longer and biologically relevant time-scales ranging in the order of a couple of seconds, in presence of noise. Also a point of interest is the robustness of this neurocomputational model to factors such as synaptic pruning, and feedback noise.

According to the "internal model" hypothesis [119, 120], there exists an internal model for each of the tasks that we have learned throughout our lives. One outcome of such a hypothesis would be that a given neuron may participate with a different synaptic weight in a number of neural assemblies, each supporting a different internal model. Interestingly, this is reflected in our setup too, as neurons in the generic neural circuit make synaptic projections to the set of readouts with different synaptic weights assigned for each task.

The results presented in chapter 4 demonstrate the role of feedback in enhancing the inherent fading memory of a neural circuit. Further it also shows the ability of generic neural circuits to model working memory and decision making, which happens at significantly longer time-scales.

Bibliography

[1] E. L. White. *Cortical Circuits.* Birkhaeuser, Boston, 1989.

[2] O. Sporns and R. Kötter. Motifs in brain networks. *PLOS Biology*, 2:1910–1918, 2004.

[3] R. A. W. Galuske, K. E. Schmidt, R. Goebel, S. G. Lomber, and B. R. Payne. The role of feedback in shaping neural representations in cat visual cortex. *PNAS*, 99(26):17083–17088, 2002.

[4] K. N. Darrow, S. F. Maison, and M. C. Liberman. Cochlear efferent feedback balances interaural sensitivity. *Nature, Neuroscience*, 2006. In Press.

[5] K. N. Darrow, S. F. Maison, and M. C. Liberman. Selective removal of lateral olivocochlear efferents increases vulnerability to acute acoustic injury. *J. Neurophysiol.*, 2006. In Press.

[6] G. Major, R. Baker, E. Aksay, B. Mensh, H. S. Seung, and D. W. Tank. Plasticity and tuning by visual feedback of the stability of a neural integrator. *Proc Natl Acad Sci*, 101(20):7739–7744, 2004.

[7] W. Maass, P. Joshi, and E. D. Sontag. Principles of real-time computing with feedback applied to cortical microcircuit models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 835–842. MIT Press, 2006.

[8] W. Maass, P. Joshi, and E. D. Sontag. Computational aspects of feedback in neural circuits. *PLOS Computational Biology*, 2006. in press.

[9] P. Joshi and W. Maass. Movement generation and control with generic neural microcircuits. In A. J. Ijspeert, M. Murata, and N. Wakamiya, editors, *Biologically Inspired Approaches to Advanced Information Technology. First International Workshop, BioADIT 2004, Lausanne, Switzerland, January 2004, Revised Selected Papers*, volume 3141 of *Lecture Notes in Computer Science*, pages 258–273. Springer Verlag, 2004.

[10] P. Joshi and W. Maass. Movement generation with circuits of spiking neurons. *Neural Computation*, 17(8):1715–1738, 2005.

[11] P. Joshi. Modeling working memory and decision making using generic neural microcircuits. In Stefanos Kollias, Andreas Stafylopatis, Wlodzislaw Duch, and Erkki Oja, editors, *Artificial Neural Networks – ICANN 2006*, volume 4131 of *Lecture Notes in Computer Science*, pages 515–524. Springer, 2006.

[12] P. Joshi. From memory based decisions to decision based movements: A model of interval discrimination followed by action selection. *submitted for publication*, 2006.

[13] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[14] H. Jäger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304:78–80, 2004.

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

[16] J.-J. E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, 1991.

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

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

[19] W. Maass, T. Natschläger, and H. Markram. Computational models for generic cortical microcircuits. In J. Feng, editor, *Computational Neuroscience: A Comprehensive Approach*, chapter 18, pages 575–605. Chapman & Hall/CRC, Boca Raton, 2004.

[20] H. Jäger. Short term memory in echo state networks. GMD Report 152, German National Research Center for Information Technology, 2002.

[21] P. E. Latham and P. Dayan. Touché: the feeling of choice. *Nature Neurosci.*, 8:408–409, 2005.

[22] C. K. Machens, R. Romo, and C. D. Brody. Flexible control of mutual inhibition: A neural model of two-interval discrimination. *Science*, 307:1121–1124, 2005.

[23] G. Rainer, H. Lee, and N. K. Logothetis. The effect of learning on the function of monkey extrastriate visual cortex. *PLoS Biology*, 2(2):275–284, 2004.

[24] B. D. Mensh, E. Aksay, D. D. Lee, H. S. Seung, and D. W. Tank. Spontaneous eye movements in goldfish: oculomotor integrator performance, plasticity, and dependence on visual feedback. *Vision Research*, 44:711–726, 2004.

[25] H. S. Seung, D. D. Lee, B. Y. Reis, and D. W. Tank. Stability of the memory of eye position in a recurrent network of conductance-based model neurons. *Neuron*, 26(1):259–271, 2000.

[26] R. Singh and C. Eliasmith. Higher-dimensional neurons explain the tuning and dynamics of working memory cells. *Journal of Neuroscience*, 26(14):3667–3678, 2006.

[27] D. Durstewitz, J. Seamans, and T. Sejnowski. Neurocomputational models of working memory. *Nature, Neuroscience*, 3, 2000.

[28] P. Miller and X-J. Wang. Inhibitory control by an integral feedback signal in prefrontal cortex: A model of discrimination between sequential stimuli. *PNAS*, 103(1):201–206, January 2006.

[29] E. Todorov. Direct cortical control of muscle activation in voluntary arm movements: a model. *Nature Neuroscience*, 3(4):391–398, 2000.

[30] P. Miller, C. D. Brody, R. Romo, and X-J. Wang. A recurrent network model of somatosensory parametric working memory in the prefrontal cortex. *Cerebral Cortex*, 13:1208–1218, 2003.

[31] R. J. Douglas, C. Koch, M. Mahowald, K. Martin, and H. Suarez. Recurrent excitation in neocortical circuits. *Science*, 269(5226):981–985, 1995.

[32] S. Grossberg. How does the cerebral cortex work? development, learning, attention, and 3D vision by laminar circuits of visual cortex. *Behavioral and Cognitive Neuroscience Reviews*, 2:47–76, 2003.

[33] W. Maass and E. D. Sontag. Neural systems as nonlinear filters. *Neural Computation*, 12(8):1743–1772, 2000.

[34] S. Häusler and W. Maass. A statistical analysis of information processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, 2007. [Epub ahead of print].

[35] A. Destexhe and E. Marder. Plasticity in single neuron and circuit computations. *Nature*, 431:789–795, 2004.

[36] W. Maass, T. Natschläger, and H. Markram. On the computational power of circuits of spiking neurons. *J. of Physiology (Paris)*, 2005. in press.

[37] M. I. Leon and M. N. Shadlen. Representation of time by neurons in the posterior parietal cortex of the macaque. *Neuron*, 38(2):317–322, 2003.

[38] K. Hikosaka and M. Watanabe. Delay activity of orbital and lateral prefrontal neurons of the monkey varying with different rewards. *Cerebral Cortex*, 10(3):263–267, 2000.

[39] L. Tremblay and W. Schultz. Modifications of reward expectation-related neuronal activity during learning in primate orbitofrontal cortex. *J Neurophysiol*, 83(4):1877–1885, 2000.

[40] W. Schultz, L. Tremblay, and J. R. Hollerman. Changes in behavior-related neuronal activity in the striatum during learning. *Trends Neurosci*, 26(6):321–328, 2003.

[41] X. J. Wang. Synaptic reverberation underlying mnemonic persistent activity. *Trends Neurosci.*, 24(8):455–463, 2001.

[42] M. E. Mazurek, J. D. Roitman, J. Ditterich, and M. N. Shadlen. A role for neural integrators in perceptual decision making. *Cerebral Cortex*, 13(11):1257–1269, 2003.

[43] M.N. Shadlen and J.I. Gold. The neurophysiology of decision-making as a window on cognition. In M. S. Gazzaniga, editor, *The Cognitive Neurosciences*, pages 1229–1241. MIT Press, 3^{rd} edition, 2005.

[44] F. Rieke, D. Warland, R. R. D. van Steveninck, and W. Bialek. *SPIKES: Exploring the Neural Code*. MIT Press, Cambridge, MA, 1997.

[45] J. D. Cowan. Statistical mechanics of neural nets. In E. R. Caianiello, editor, *Neural Networks*, pages 181–188. Springer, Berlin, 1968.

[46] M. A. Cohen and S. Grossberg. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Trans. Systems, Man, and Cybernetics*, 13:815–826, 1983.

[47] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. Nat. Acad. Sci. USA*, 81:3088–3092, 1984.

[48] P. Dayan and L. F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems.* MIT-Press, 2001.

[49] R. A. Legenstein and W. Maass. Edge of chaos and prediction of computational power for neural microcircuit models. *submitted for publication*, 2006.

[50] J. E. Savage. *Models of Computation: Exploring the Power of Computing.* Addison-Wesley (Reading, MA), 1998.

[51] W. Maass and H. Markram. Theory of the computational function of microcircuit dynamics. In S. Grillner and A. M. Graybiel, editors, *The Interface between Neurons and Global Brain Function*, Dahlem Workshop Report 93, pages 371–390. MIT Press, 2006.

[52] M. S. Branicky. Universal computation and other capabilities of hybrid and continuous dynamical systems. *Theoretical Computer Science*, 138:67–100, 1995.

[53] H. Siegelmann and E. D. Sontag. Analog computation via neural networks. *Theoretical Computer Science*, 131(2):331–360, 1994.

[54] H. Siegelmann and E. D. Sontag. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50:132–150, 1995.

[55] P. Orponen. A survey of continuous-time computation theory. In D.-Z. Du and K.-I. Ko, editors, *Advances in Algorithms, Languages, and Complexity*, pages 209–224. Kluwer Academic Publishers, 1997.

[56] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, New Jersey, 2nd edition, 1999.

[57] W. Maass and P. Orponen. On the effect of analog noise in discrete-time analog computations. *Neural Computation*, 10:1071–1095, 1998.

[58] W. Maass and E. Sontag. Analog neural nets with Gaussian or other common noise distribution cannot recognize arbitrary regular languages. *Neural Computation*, 11:771–782, 1999.

[59] W. Maass and H. Markram. On the computational power of recurrent circuits of spiking neurons. *Journal of Computer and System Sciences*, 69(4):593–616, 2004.

[60] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[61] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA*, 79:2554–2558, 1982.

[62] D. J. Amit and N. Brunel. Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cerebral Cortex*, 7(3):237–252, 1997.

[63] C. D. Brody, R. Romo, and A. Kepecs. Basic mechanisms for graded persistent activity: discrete attractors, continuous attractors, and dynamic representations. *Curr Opin Neurobiol*, 13(2):204–211, 2003.

[64] A. Destexhe, M. Rudolph, and D. Pare. The high-conductance state of neocortical neurons in vivo. *Nat. Rev. Neurosci.*, 4(9):739–751, 2003.

[65] H. Markram, Y. Wang, and M. Tsodyks. Differential signaling via the same axon of neocortical pyramidal neurons. *PNAS*, 95:5323–5328, 1998.

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

[67] G. Major, R. Baker, E. Aksay, H. S. Seung, and D. W. Tank. Plasticity and tuning of the time course of analog persistent firing in a neural integrator. *Proc Natl Acad Sci*, 101(20):7745–7750, 2004.

[68] R. A. Legenstein, C. Näger, and W. Maass. What can a neuron learn with spike-timing-dependent plasticity? *Neural Computation*, 17(11):2337–2382, 2005.

[69] J. Wickens and R. Kötter. Cellular models of reinforcement. In J. C. Houk, J. L. Davis, and D. G. Beiser, editors, *Models of Information Processing in the Basal Ganglia*. MIT Press, Cambridge, 1998.

[70] A. Destexhe, M. Rudolph, J. M. Fellous, and T. J. Sejnowski. Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons. *Neuroscience*, 107(1):13–24, 2001.

[71] A. Destexhe and D. Pare. Impact of network activity on the integrative properties of neocortical pyramidal neurons in vivo. *J. Neurophysiol.*, 81(4):1531–1547, 1999.

[72] D. A. Hoffman, J. C. Magee, C. M. Colbert, and D. Johnston. K+ channel regulation of signal propagation in dendrites of hippocampal pyramidal neurons. *Nature*, 387(6636):869–875, 1997.

[73] J. C. Magee and D. Johnston. Characterization of single voltage-gated Na+ and Ca2+ channels in apical dendrites of rat CA1 pyramidal neurons. *J. Physiol.*, 487 (Pt 1):67–90, 1995.

[74] J. Magee, D. Hoffman, C. Colbert, and D. Johnston. Electrical and calcium signaling in dendrites of hippocampal pyramidal neurons. *Annu. Rev. Physiol.*, 60:327–346, 1998.

[75] G. J. Stuart and B. Sakmann. Active propagation of somatic action potentials into neocortical pyramidal cell dendrites. *Nature*, 367(6458):69–72, 1994.

[76] R. D. Traub and R. Miles. *Neuronal Networks of the Hippocampus*. Cambridge Univ. Press, Cambridge, UK, 1991.

[77] Z. T. Mainen, J. Joerges, J. R. Huguenard, and T. J. Sejnowski. A model of spike initiation in neocortical pyramidal neurons. *Neuron*, 15(6):1427–1439, 1995.

[78] J. R. Huguenard, O. P. Hamill, and D. A. Prince. Developmental changes in Na^+ conductances in rat neocortical neurons: appearance of a slowly inactivating component. *J. Neurophysiol.*, 59:778–795, 1988.

[79] J. Anderson, I. Lampl, I. Reichova, M. Carandini, and D. Ferster. Stimulus dependence of two-state fluctuations of membrane potential in cat visual cortex. *Nature Neuroscience*, 3(6):617–621, 2000.

[80] L. J. Borg-Graham, C. Monier, and Y. Fregnac. Visual input evokes transient and strong shunting inhibition in visual cortical neurons. *Nature*, 393(6683):369–373, 1998.

[81] J. A. Hirsch, J. M. Alonso, R. C. Reid, and L. M. Martinez. Synaptic integration in striate cortical simple cells. *J. Neurosci.*, 18(22):9517–9528, 1998.

[82] W. Maass and H. Markram. Synapses as dynamic memory buffers. *Neural Networks*, 15:155–161, 2002.

[83] T. Natschläger, H. Markram, and W. Maass. Computer models and analysis tools for neural microcircuits. In R. Kötter, editor, *Neuroscience Databases. A Practical Guide*, chapter 9, pages 123–138. Kluwer Academic Publishers (Boston), 2003.

[84] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan, and M. Nicolelis. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. *Nature*, 408(6810):361–365, 2000.

[85] W. Maass, T. Natschläger, and H. Markram. Computational models for generic cortical microcircuits. In J. Feng, editor, *Computational Neuroscience: A Comprehensive Approach*, chapter 18, pages 575–605. Chapman & Hall/CRC, Boca Raton, 2004. Online available as #149 from http://www.igi.tugraz.at/maass/publications.html.

[86] W. Maass, R. A. Legenstein, and N. Bertschinger. Methods for estimating the computational power and generalization capability of neural microcircuits. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 865–872. MIT Press, 2005.

[87] E. Todorov. On the role of primary motor cortex in arm movement control. In M. L. Latash and M. Levin, editors, *Progress in Motor Control III*, pages 125–166. Human Kinetics, 2003.

[88] E. Todorov. Direct control of muscle activation in voluntary arm movements: a model. *Nature Neuroscience*, 3(4):391–398, 2000.

[89] A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, and K. Obermayer, editors, *Proc. of NIPS 2002, Advances in Neural Information Processing Systems*, volume 15, pages 1547–1554. MIT Press, 2003.

[90] A. Pouget and P. E. Latham. Population codes. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 893–897. MIT Press, 2003.

[91] T. Flash and N. Hogan. The coordination of arm movements: An experimentally confirmed mathematical model. *The Journal of Neuroscience*, 5(7):1699–1703, 1965.

[92] J. J. Craig. *Introduction to Robotics: Mechanics and Control.* Addison-Wesley, Reading, (MA), 2nd edition, 1955.

[93] R. J. van Beers, P. Baraduc, and D. M. Wolpert. Role of uncertainty in motor control. *Phil. Trans. R. Soc. Lond. B*, 357:1137–1145, 2002.

[94] T. Natschläger and W. Maass. Information dynamics and emergent computation in recurrent circuits of spiking neurons. In S. Thrun, L. Saul, and

B. Schölkopf, editors, *Proc. of NIPS 2003, Advances in Neural Information Processing Systems*, volume 16, pages 1255–1262, Cambridge, 2004. MIT Press.

[95] R. A. Legenstein, H. Markram, and W. Maass. 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, 2003.

[96] A. d'Avella, P. Saltiel, and E. Bizzi. Combination of muscle synergies in the construction of a natural motor behavior. *Nature Neuroscience*, 6(3):300–308, 2003.

[97] R. Pfeifer. 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, 2002. See http://www.ifi.unizh.ch/groups/ailab/.

[98] H. S. Seung. How the brain keeps the eyes still. *Proc Natl Acad Sci*, 93(23):13339–13344, 1996.

[99] H. S. Seung, D. D. Lee, B. Y. Reis, and D. W. Tank. The autapse: A simple illustration of short-term analog memory storage by tuned synaptic feedback. *Journal of Computational Neuroscience*, 9:171–185, 2000.

[100] A. Pouget and P. E. Latham. Population codes. In M. A. Arbib, editor, *The handbook of brain theory and neural networks*, pages 893–897. MIT Press, 2003.

[101] J. M. Fuster. Unit activity in prefrontal cortex during delayed-response performance: neuronal correlates of transient memory. *J. Neurophysiol.*, 36:61–78, 1973.

[102] E. K. Miller, C. A. Erickson, and R. Desimone. Neural mechanisms of visual working memory in prefrontal cortex of the macaque. *J. Neurosci.*, 16:5154–5167, 1996.

[103] J. M. Fuster, M. Bodner, and J. K. Kroger. Cross-modal and cross-temporal association in neurons of frontal cortex. *Nature*, 405:347–351, 2000.

[104] A. P. Georgopoulos, R. Caminiti, J. F. Kalaska, and J. T. Massey. Spatial coding of movement: a hypothesis concerning the coding of movement direction by motor cortical populations. *Exp. Brain Res.*, 7 (Suppl.):327–336, 1983.

[105] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner. Neuronal population coding of movement direction. *Science*, 233:1416–1419, 1986.

[106] A. P. Georgopoulos, R. E. Kettner, and A. B. Schwartz. Primate motor cortex and free arm movements to visual targets in three-dimensional space. II. coding of the direction of movement by a neuronal population. *J. Neurosci.*, 8:2928–2937, 1988.

[107] R. E. Kettner, A. B. Schwartz, and A. P. Georgopoulos. Primate motor cortex and free arm movements to visual targets in three-dimensional space. III. positional gradients and population coding of movement direction from various movement origins. *J. Neurosci.*, 8:2938–2947, 1988.

[108] A. B. Schwartz, R. E. Kettner, and A. P. Georgopoulos. Primate motor cortex and free arm movements to visual targets in three-dimensional space. i. relations between single cell discharge and direction of movement. *J. Neurosci.*, 8:2913–2927, 1988.

[109] A. B. Schwartz. Motor cortical activity during drawing movements: population response during sinusoid tracing. *J. Neurophysiol.*, 70:28–36, 1993.

[110] A. B. Schwartz. Direct cortical representation of drawing. *Science*, 265:540–542, 1994.

[111] A. B. Schwartz and D. W. Moran. Motor cortical activity during drawing movements: population representation during lemniscate tracing. *J. Neurophysiol.*, 82:2705–2718, 1999.

[112] A. P. Georgopoulos, J. T. Lurito, M. Petrides, A. B. Schwartz, and J. T. Massey. Mental rotation of the neuronal population vector. *Science*, 243:234–237, 1989.

[113] N. Smyrnis, R. E. Kettner, and A. P. Georgopoulos. Motor cortical activity in a meorized delay task. *Exp. Brain Res.*, 92:139–151, 1992.

[114] J. Ashe, M. Taira, N. Smyrnis, G. Pellizzer, T. Georgakopoulos, J. T. Lurito, and A. P. Georgopoulos. Motor cortical activity preceding a memorized movement trajectory with an orthogonal bend. *Exp. Brain Res.*, 95:118–130, 1993.

[115] W. Kruse, S. Dannenberg, R. Kleiser, and K. P. Hoffmann. Temporal relation of population activity in visual areas MT/MST and in primary motor cortex during visually guided tracking movements. *Cereb. Cortex*, 12:466–476, 2002.

[116] R. Romo, A. Hernandez, L. Lemus, A. Zainos, and C. D. Brody. Neuronal correlates of decision-making in secondary somatosensory cortex. *Nature Neurosci.*, 5:1217–1225, 2002.

[117] E. Salinas, A. Hernandez, A. Zainos, and R. Romo. Periodicity and firing rate as candidate neural codes for the frequency of vibrotactile stimuli. *Journal of neuroscience*, 20(14):5503–5515, 2000.

[118] C. D. Brody, A. Hernandez, A. Zainos, and R. Romo. Timing and neural encoding of somatosensory parametric working memory in macaque prefrontal cortex. *Cerebral Cortex*, 13:1196–1207, 2003.

[119] Kenneth J. W. Craik. *The Nature of Explanation*. Cambridge University Press, 1943.

[120] E. Bizzi and F. A. Mussa-Ivaldi. Neural basis of motor control and its cognitive implications. *Trends in Cognitive Sciences*, 2(3):97–102, 1998.

[121] J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Publishing Company, 1989.

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