A biologically inspired model of motor control of direction

A thesis presented
by

Ioana Domnica Marian

to
The Department of Computer Science
in partial fulfillment of the requirements
for the degree of
Master in Computer Science

Under the supervision of
Ronan G. Reilly, Ph.D

University College Dublin
Dublin, Ireland
October 2002

Thesis advisor
Ronan G. Reilly

Author
Ioana Domnica Marian

Abstract

Sensorimotor coordination has been an active research topic for both neuroscience and artificial intelligence over the last decade. For the visual guidance of movement to be efficiently implemented in artificial systems, it is essential to understand the computational mechanisms underpinning biological motor control. This thesis contributes to the understanding of this issue through the development of a biologically inspired computational framework based on spiking neural networks. The work described in this thesis focuses upon the modeling of two developmental processes.

Firstly, we address the development in a self–organizing map of neural directional selectivity. As a result of the learning process and of the input patterns used, the network learns to represent 12 directions of movement in a distributed code. A population coding of direction results, which is analyzed using the population vector scheme, and is compared with neurobiological data on motor cortex organization.

Secondly, we propose a computational mechanism based on spike-timing dependent learning, for the transfer of directional information between a visual and a motor map. Learning of the visuomotor mapping resides in the development of connection strengths that are dependent on the similarity between the preferred directions of neurons in the two maps. The computational mechanism obtained and the neural behaviors resulted are discussed with respect to their neurophysiological implications. We believe that biologically inspired modeling of motor control development can be highly beneficial to the understanding of brain computations underlying movement control.

Acknowledgments

I would like to thank everybody who has supported me whilst I have worked towards the completion of this thesis. Firstly, I would like to thank to my supervisor Ronan Reilly for support and guidance he has provided during my work. I would like to thank to the Enterprise Ireland for supporting financially my research during my work at University College Dublin.

I would like to thank all my colleagues who have helped me to continue my research and write this thesis. My special thanks to Nikolinka Collier and Corina Sas for their continuous support and help while I have worked for this thesis. I would also like to thank my parents for their understanding and support whenever I have needed. Finnally, I would like to thank to my friend Octavian Goga without whom I could have never been able to carry on this work.

Dedicated to my parents

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Major advances in science often consist in discovering how macro–scale phenomena reduce to their miscroscale constituents. The ’astonishing hypothesis’ is that our minds can be explained by understanding the detailed behavior of neurons in the brain and their interactions with each other (Crick, 1994). Brains are collections of billions of interconnected cells, each of them being an individual machinery which receives, process and transmits information (Kandel et al., 2000). The human brain generates complex patterns of behavior at different scales of organization based on the large amount of neural components that interact simultaneously in a rich number of parallel ways. In order to understand the inherent complexity of such a system many complementary research strategies are employed. Today, in this attempt, experimental and theoretical neuroscience studies are accompanied by mathematical theories of complex systems organization, evolutionary and developmental studies of the brain organization, all assisted by computational modeling means.

The work presented in this thesis lies at the intersection of three domains: neuroscience, artificial intelligence and developmental psychology. This work makes use of computer simulations with networks of neuron–like elements in order to understand how cognitive phenomena can be grounded at the neural level. In this introduction, we motivate our approach and we present the problem statement. In Section 1, a short overview of several research strategies of brain function is provided. We believe that our methods are best described as computational neuroscience. An introduction to this field is given, accompanied by a discussion of the advantages of computer modeling, as a methodology. The computational approach is based on information processing with spiking neurons, which are motivated in Section 1.1.2. In Section 2 we propose a number of challenging problems in motor control, which are investigated in this thesis through biologically inspired modeling. The specific objectives of our work are introduced in Section 3. Finally, Section 4 outlines the structure of the thesis.

1.1 What is Computational Neuroscience?

Computational neuroscience is an evolving approach that draws on neurobiological data, but uses computational modeling and computer simulations to investigate the principles of operation governing neurons and networks of neurons (see Bower and Beeman, 1998; De Schutter, 2001). The domain is rapidly growing, involving researchers with backgrounds ranging from psychology and cellular biology to artificial intelligence. What is new about this discipline and how does it relate to other research techniques of the nervous system?

Firstly, brain function can be investigated with dedicated tools from experimental neuroscience. Using imaging techniques (PET, fMRI), intra- and extra-cellular recording, patch clamp and neuron staining techniques promises to allow researchers to visualize the ’brain in action’ (Toga and Mazziotta, 1996; Frackowiak et al., 1997; Kandel et al., 2000). Today, this experimental technology has advanced to the point that biological information can be readily obtained, hence, this knowledge has accumulated and generated huge databases of neurophysiological information (Bower and Beeman, 1998). Supported by advances in computer software and hardware technology, experimental biologists oriented towards the test of mathematical models and the simulation of neurobiological details (see Koch and Segev, 1998). This research direction has impelled the emergence of the computational neuroscience field, where it advocates the use of structurally realistic neural models in the simulation of the brain phenomena (Bower and Beeman, 1998).

The main goal of neuroscience, since its early beginnings, has been to bridge the gap between the behavior of single neurons and complex behavior emerging from their cooperative function. The theory of complex systems is specifically aimed at understanding this issue. The dynamics of complex systems are characterized by local, spontaneous ordering tendencies, which can lead to global self–organized states. The brain is such a self-organizing system and the modeling of how it can learn by itself represents a successful paradigm in the brain research (Kelso, 1988; Kohonen, 1995).

Dynamical systems theory is also increasingly exploited as a means of understanding brain function, both at a neural and cognitive level. At the neural level, a number of researchers have argued that the dynamical properties of firing neurons may have a central role to play in explaining how the brain computes (i.e., synchronous oscillations may play a crucial role in cognitive binding, Singer, 1994). At the cognitive level, a recent view is that cognitive processes are behavioral patterns of non-linear dynamical systems and are best studied using the mathematics of dynamical modeling and dynamical systems theory (Kelso, 1988; Port and Van Gelder, 1995). This paradigm has recently challenged the computational perspective of the brain function (Van Gelder, 1995). Despite the current debate of whether the brain *computes* or *integrates* (Chauvet, 2002), the dynamical systems theory remains a valuable, generally accepted mathematical approach for analyzing patterns of behavior generated at different scales of organization.

To understand brain function it is important to consider that it is the end product of an evolutionary and developmental process. With regard to brain evolution from a simple to a more complex system, a viable approach is to consider that it has developed incrementally over evolutionary time (Crick, 1994). An incremental approach will tend to favor the addition of small specialists modules rather than a re–engineering of the entire system (Crick, 1994). For instance, taking this view, what evolution has done in constructing our sensory systems was to provide additional sources of constraint on the possible identity of an object (Reilly, 2001) or to facilitate the apparition of a new response to the environment (Goodale, 2000).

Compared to the evolutionary perspective, the developmental study of the brain is more within our grasp. It represents a primary methodology for the analysis and the understanding of the evolution of human cognition and behavior during a lifetime (see the developmental psychology and the epigenetic approach, Piaget, 1969). It also represents a fruitful paradigm in which artificial systems can be constructed to develop by their own means, complex behaviors based on simpler components (Zlatev and Balkenius, 2001; Weng et al., 2001). A developmental approach provides a structured decomposition of complex tasks. It divides high–level processes into computationally simple, and developmentally earlier, behaviors (Scassellati, 1998). From this perspective, language and cognition may be understood through the incremental development of more sophisticated structures on the foundations of preexisting low–level, sensory–motor programs (Reilly, in press).

To conclude, in order to understand the complexity of brain processes, several research disciplines are employed in a complementary manner. The new field of computational neuroscience is rooted in neuroscience, by drawing on biological data about information processing in the cells of the brain. In this attempt, the mathematical means of dynamical systems theory is valuable and can be employed for the analysis of behavioral patterns emerged. Regarding the developmental approach, in this thesis it is believed that only by proceeding in an incremental manner can we overcome the complexity issues facing any attempt to model the brain. The core methodology of computational neuroscience is represented by means of computer modeling. The advantages of cognitive modeling in general and those of simulation with functionally realistic neurons, in particular, are outlined below.

1.1.1 Why cognitive modeling?

Experimental investigations in neuroscience can provide a detailed characterization of the chemical processes that underlie cognitive processes, but what matters for most scholars in cognitive science are not the details themselves, but the principles that are embodied in these details (McClelland, 2000). What is essential about the computational models is that they enable researchers to explore the nature of these principles, by implementing on a computer the underlying mechanisms (Levine, 2000; Rolls and Treves, 1998; O'Reilly and Munakata, 2000).

Furthermore, it is important to view the computational modeling of the brain processes from the *reconstructionist paradigm*. This concentrates on the process of constructing human cognition from the action of a large number of interacting components (McClelland, 2000). The emphasis is on investigating the emergent phenomena which arise from these interactions, and which is not obviously present in the behavior of individual elements (Cleeremans and French, 1996; O'Reilly and Munakata, 2000).

Computational modeling allows manipulation and control of variables more precisely than can be done with a real system. This enables the researcher to explore the causal roles of different components (see the modeling of dendrites role in auditory coincidence detection in Agmon-Snir et al., 1998). Computer modeling allows testing of hypothesis, but it is also a powerful means of generating new and original hypothesis. A computational model can provide novel sources of insight into behavior by providing alternative explanations of a phenomena (see the new hypothesis on pattern recognition in a Purkinje cell in Steuber and De Schutter (in press)). Finally, when creating a computational model, one has to be explicit about assumptions and about exactly how the processes work (O'Reilly and Munakata, 2000).

Computational neuroscience differs from previous approaches to neural modeling (see the connectionist paradigm, Rumelhart and McClelland, 1986) in that, researchers in the area believe that understanding the way the brain computes is very closely dependent on the anatomical and physiological details of the neural components. Hence, it focuses upon the simulation of the cortical functions by using neural models that are in agreement with the structure and physiology of real neurons.

1.1.2 Why spiking neurons?

Cognitive models developed in the previous decade, dominated by the connectionist psychology paradigm (Rumelhart and McClelland, 1986) have neglected the spiking nature of the neurons. These models have been usually constructed based on a neural model with sigmoid activation function that gives a continuous, real-valued output. Networks of such neurons have proven very powerful computationally (O'Reilly, 2001). Moreover, they have been supported biologically by the long-standing belief in neuroscience that information in the brain is carried mainly in the neurons discharge rates (for a review see Recce, 1999).

Recent observations of how fast computations take place in the visual brain has questioned whether the firing rate interpretation alone can account for rapid neural information processing (Thorpe and Gautrais, 1997). Experimental evidence has been accumulated in the last years to indicate that biological neural systems use the timing of single action potentials to encode information (Abeles et al., 1993; Gerstner et al., 1996; Rieke et al., 1997). Consequently, learning that information can be encoded in the temporal pattern of neuron firing, the research on information processing in neural systems has focussed on investigating computations with *spiking neurons* (Gerstner and Van Hemmen, 1994; Maass, 1995; Rieke et al., 1997; Stevens and Zador, 1998).

This research stream has given rise to a new generation of neural networks, referred to as *pulsed neural networks* and has focused on providing a mathematical description of the computational properties of biological neurons (Gerstner, 1999; Maass, 1999). Several alternative theories to the rate–coding hypothesis have been proposed, which suggest different schemes for where the neural information may be contained, such as in the timing of the spikes, and in the correlated activity of neurons (Stevens and Zador, 1995; Recce, 1999).

The computational work of this thesis is based on a simplified spiking neural model that focuses upon several aspects of the biological neuron function. Compared to the classical rate–coding neuron, this model accounts for the spiking nature of real cells and allows different modes of computation and learning. On the other hand, in comparison with structurally realistic neural models, it accounts only for a limited number of computational aspects, with the advantage of being simple to analyze and to simulate in a large–scale network of neurons.

The development of the new generation of spiking neural networks has necessitated the design of dedicated simulation environments. Several simulation frameworks for the realistic modeling of biological neurons (Hines and Carnevale, 1995; Bower and Beeman, 1998) or for computation with simplified models (Delorme et al., 1999; Sougné, 1999) have been created in the last decade. In our case, the choice of a modeling environment was motivated primarily by the ability of the system to support the development of a family of models required by current and future work goals. What seemed to be the best solution was to use a general-purpose simulator, regardless of the neural model implemented, which was extended to allow computations and learning with spiking neurons. The simulator chosen is a classic neural networks simulator, SNNS: Stuttgart Neural Network Simulator (Zell et al., 1992), whose extension for modeling of spiking neurons is referred to, in this thesis, as SpikeNNS.

To conclude, this thesis advocates the use of the computer simulation methodology in order to investigate computational principles of the cortex. Our work is aimed at providing an illustration of how cognitive brain functions can be grounded at the neural level. In doing this, we employ simulations with biologically inspired neural models. The general aim is to explore within a developmental approach how emergent properties of the brain are high–level effects that depend on low–level computational properties of the basic constituents. This goal is investigated with a number of models of specific cognitive phenomena.

1.2 Identifying difficult cognitive problems

1.2.1 Neonatal imitation

Imitation is an essential behavior for cognitive development in infants, because it serves communication and rapid acquisition of adaptive behavior and is an alternative to expensive trial–and–error learning (Butterworth, 1999). Meltzoff and Moore (1977) have shown that infants as young as 12–days–old can imitate facial actions of caregivers, such as tongue protrusion, mouth opening and manual gestures. Their findings served as an essential testimony to the existence of imitation in newborns and suggested the possibility that immediate imitation is a fundamental mechanism of communication in humans (Nadel and Butterworth, 1999). Since the publication of their original results, a large number of studies have been issued to investigate the scale of the neonate imitation phenomena. A range of facial expressions and hand gestures, as well as deferred imitation capacity, have been described with respect to the age of onset and generalization capacity (Kugiumutzakis, 1999; Meltzoff and Moore, 1999).

The ’holy grail’ for the theories of neonatal imitation has been to elucidate the mechanisms whereby infants are able to connect the felt but unseen movements of the self with the seen but unfelt movements of the others (Butterworth, 1999). This process is considered to require an inter–modal mapping, that is, a transfer of information between different perceptive modalities (i.e., visual and somatosensory) in order to control the imitative acts. A common belief is that proprioceptive feedback on self-produced movements can be compared to the visually specified target in a supra-modal (Metlzoff and Moore, 1999) or amodal framework (Trevarthen et al., 1999).

To us, neonatal imitation behavior represents the challenging problem. At a deeper analysis, it appeared to be the ’tip of the iceberg’ of a more general problem of brain computation: sensory fusion or cross–modal matching of information. That is, the basic computational demand for imitation is met by the transfer of information between different modalities. In this way, it led us to the fundamental topics of perception–action coupling and sensorimotor coordination.

1.2.2 Visuomotor coordination

Most human and animal movements are under continual sensory guidance. Even a simple task such as reaching and grasping an object, requires the analysis of visual information to trigger a reaching movement and the integration of proprioceptive information and tactile sensation to tune the grip to the weight, friction, and shape of the object (Kandel et al., 2000). That is, purposeful action is possible through the integration of sensory signals coming from various sources (vision, hearing, touch) and their translation into a set of motor commands to the muscles (Massone, 1995; Kandel et al., 2000).

Consequently, significant efforts have been dedicated in the last decades to the understanding of the computational mechanisms that support sensorimotor coordination in neural systems. Particular attention has been given to the development of coordination between the eye and the hand for reaching movements (see Caminiti et al., 1992). One reason is represented by the huge applicability, that a mechanism which allows the correct transformation of visual signals into motor output, would have in the implementation of artificial systems capable of online, adaptive motor control (Zeller et al., 1995; Weng et al., 2001). Another motivation is that by understanding the brain computations that underlie reaching movements, it will be possible for humans to control a robotic arm solely through the power of thought (Nicolelis and Chapin, 2002).

Despite the efforts of classical artificial intelligence solely, it seems that the behavioral capabilities of biological organisms can be simulated only by closely reproducing neural computational mechanisms. Consequently, several recent proposals have been made for a more biologically inspired modeling of visuomotor coordination (Burnod et al., 1992; Bullock et al., 1995; Zeller et al., 1995). These attempts are facilitated by the development of a new conceptual scheme of cortical control of reaching movements.

Research on the visual guidance of arm–reaching movement has made significant progress in the recent years, in explaining control theory formalisms such as the 'coordinate transformation' (i.e., sensorimotor transformation), in terms of computational properties of single neurons and networks of neurons (Andersen et al., 1997; Kalaska et al., 1997; Caminiti et al., 1998; Burnod et al., 1999). Details regarding the manner in which the brain solves the sensorimotor mapping have cumulated and have given rise to an integrative framework that links neurophysiological and computational aspects and allows the ready implementation of the latter in terms of the former (Bullock et al., 1995; Burnod et al., 1999).

What is still needed, are models whose architecture is supported by anatomical evidence, composed of elements that correspond as closely as possible to known neural cell types, and whose functionality meet psychophysical criteria (Bullock et al., 1995). It is this challenge that motivates part of the work presented within this thesis. The advantage of modeling the neurophysiological processes involved in the cortical control of reaching is bi-directional. Firstly, more adaptive and flexible models can be obtained. Secondly, models can be used to test computational principles and eventually to reveal unknown mechanisms of visuomotor control of movement.

1.2.3 Cortical control of motion direction

The human motor system is organized in a functional hierarchy, with each level concerned with different decisions (Kandel et al., 2000). Voluntary movements are organized at the highest level of the frontal cortical lobes, in the premotor and primary motor cortex. These areas are involved in the preparation, execution and adaptation of movement. Movements of the arm, such as reaching or grasping, involve multiple joints and require precise activation of the skeletal muscles. This raises the question of whether cells at the cortical level control muscle activation or do they encode more global features of the movement, such as direction, amplitude, or speed (Kakei et al., 1999). Today, there is substantial evidence that the direction of movement is represented at the cortical level, in the activity of large populations of cells that are broadly selective to the direction of motion (Georgopoulos et al., 1984).

Feature detection is not only a characteristic of motor cells, but represents an essential property of neurons in the brain. It means that neurons respond to a particular feature of the stimulus (i.e., orientation, direction, frequency of signal) and specialize in detecting a range of values of that feature in the input space (Hubel and Wiesel, 1962). The dedication of neurons is very widely distributed in the brain. For instance, directional selectivity has been described for neurons in the visual, motor, and touch cortex (Kandel et al., 2000). The specialization of the neurons generally occurs by the way each cell is connected with other cells, from the input layers or within the same layer, and it results in the emergence of cortical feature maps (see orientation visual maps in Blasdel, 1992). The development of these maps can be modeled through a process of self-organization and topographical mapping of the input space into the network nodes (Kohonen, 1984).

Despite the substantial evidence indicating that directional tuning is an essential feature of the motor cortical neurons, it is not clear yet how the cortical control of motion direction is developed. This contrasts with the detailed anatomical and computational knowledge existent on the development of sensory, and in particular, visual maps (Obermayer et al., 1990; Douglas et al., 1991; Blasdel, 1992; Sirosh, 1995). Nevertheless, it is believed that the development of visual preferences may be based on a few design principles that in turn rely on very general mechanisms utilizing the input structure of the system (Niebur and Wörgötter, 1992).

Part of the work presented in this thesis concerns the simulation of the process whereby neurons in the motor areas develop directional selectivity. Firstly, modeling this process can offer insights into whether directional tuning of motor cells is innate or acquired. Secondly, the investigation of the functional principles of motor cortex can show in what degree the organizational mechanisms of the sensory cortices also operate in the motor areas. Thirdly, understanding the way the motor cortex organizes itself to control the direction of motion is highly important for explaining the development of sensorimotor coordination as a whole. Furthermore, current models of motor control which implement the control of end–effector direction, lack a developmental model of how neurons at the cortical level acquire directional selectivity (Bullock et al., 1993). Hence, we believe that the modeling of the formation of directional motor map may have important implications for neurophysiology and robotics.

1.3 The problem statement

Most generally, the research presented in this thesis is aimed at investigating how cognitive functions in the brain can emerge from the properties of basic components, when these interact and function cooperatively. This objective is narrowed down to the study of two topics: cortical control of movement direction and visuomotor mapping of directional information. Our detailed research objectives are:

- Modeling of the self–organization of motor cortical neurons for coding the direction of movement.
- Modeling of the alignment of visual and motor neural representations for the guidance of directional movements.

Note that these objectives are related. That is, by achieving the self–organization of the motor directional map, the organized network can be used in the next stage for the development of visuomotor mapping.

The original contribution of our models is represented primarily, by the investigation of the two topics described above within the biologically inspired computational framework, of spiking neural networks. Secondly, the motor directional map model represents a first attempt to simulate the emergence of cortical directional selectivity within the self–organization paradigm (Kohonen, 1984). The resulting representation of movement will be compared to the neurophysiological data that describes the coding of direction of movement in the motor cortex (Georgopoulos et al., 1984; Georgopoulos et al., 1993). If the model succeeds in developing neural directional selectivity with a similar profile to that of real motor cells, than we have a computational hypothesis for how the population coding emerges in the motor cortex. Learning the visuomotor mapping of directional information in the conditions of a population coding in the motor area represents another innovative feature of our modeling work. This has computational and neurophysiological relevance for the learning of visuomotor mapping.

In order to achieve these main objectives, they are preceded by the design of a modeling environment for networks of spiking neurons. When modeling large–scale pulsed neural networks with plastic synapses, the time efficiency of the simulation becomes an essential issues in the design of the simulator (Jahnke et al., 1999). A number of strategies are implemented and compared and an innovative event–driven mechanism is proposed to reduce the time of simulation.

1.4 Thesis outline

The structure of this thesis is organized as follows.

Chapter 2 provides a biological framework of cortical control of motion direction, for use in our modeling work. It introduces the self-organization paradigm for biological modeling of cortical feature map formation. Finally, it reviews several recent neural network models that address issues of cortical coding of motion direction.

Chapter 3 introduces a number of new theories on the biological and computational mechanisms of perception–action coupling. Next, it focuses upon the description of an integrative framework of how the visual guidance of arm reaching is implemented in the brain. A review of several biologically inspired models of learning of visuomotor coordination is presented.

Chapter 4 focuses upon the description of the implemented spiking neural model. It presents how information can be communicated in the timing of single spikes, what types of computations are implemented and at what level of detail. Learning with spiking neurons is also discussed.

Chapter 5 presents the implementation of the SpikeNNS simulator. Design issues are discussed, with particular attention given to the strategies used to increase the time–efficiency of the simulation. The configurable features of the simulator are outlined.

Chapter 6 describes in turn, the model of the motor cortex self-organization and that of visuomotor mapping learning. The results of the simulations are analyzed and discussed in comparison with current neurophysiological data and with previous modeling work.

Chapter 7 is devoted to a final discussion of the neurophysiological and theoretical implications of our models. It also proposes the potential integration of the models within artificial motor control systems and it proposes possible avenues of future work. The relevance of our visuomotor mapping model to the imitation problem is discussed in the end.

Part I

Neurobiology and models of motor control of direction

Chapter 2

Control of movement direction

The state of the art in biological motor control reflects a long–standing debate concerning, whether cortical neurons encode parameters of hand path rather than muscle activity, and whether the central nervous system uses spatial coordinates, rather than joint or muscle coordinates (Bizzi and Mussa-Ivaldi, 2000; Johnson et al., 2001). This is related to a shift in the main paradigm of motor control, from correlating primary motor cortical neuronal firing with movements of isolated muscles (Scott and Kalaska, 1995; Scott, 1997) to correlating neuronal firing with whole arm movements (Georgopoulos et al., 1984; Schwartz et al., 1988; Caminiti et al., 1991). Pivotal to this change of framework were the experimental studies of Georgopoulos and colleagues (1982, 1984), showing that the discharge rate of primary motor cortex neurons is tuned to the direction of arm movement.

Whether muscles or movements are represented in the motor cortex, is related to the distinction between planning and execution of movements (Bizzi and Mussa-Ivaldi, 2000). Furthermore, it represents an important aspect for any developmental approach of motor cortex self–organization. The purpose of this chapter is to provide a biological framework of motor control for use in our modeling work. Considering the existing controversies in the research on human motor control, we intend to marshal and evaluate neurophysiological and psychophysical evidence in support of the working hypotheses of the thesis.

Section 1 examines experimental evidence for the hypothesis that motor areas plan movement in spatial coordinates and represent abstract parameters of the hand path. It includes a discussion of the planning–execution distinction and of the cortical representation of motion direction and an introduction of the population coding scheme. Section 2 focuses upon the organization of cortical feature maps and introduces the means for the simulation of their development. One of the most successful approaches to modeling the formation of brain maps is the self–organizing feature map (Kohonen, 1984). Two adaptations of the original algorithm are outlined, based on learning in the lateral synapses and spiking neural models. Section 3 presents a review of a number of recent models that address the issue of spatial coding of directionality. In the absence of previous modeling work on motor directional maps formation, inspiration can be drawn from models of visual maps organization (Farkaš and Miikkulainen, 1999) or from simulations which address population coding of movement (Lukashin and Georgopoulos, 1994).

2.1 Biological motor control

Movement of primates is the result of information processing in a complex hierarchy of motor centers within the nervous system, which yields three levels of control: the spinal cord, brain stem, and motor cortex (Kandel et al., 2000). The highest levels of cortical motor control are often associated with the premotor regions, which are the lateral ventral cortex, the dorsal premotor cortex and the supplementary motor areas. The lowest cortical level is occupied by the primary motor cortex.

The premotor cortex has a major role in coordinating and planning complex sequences of movements. It integrates sensory information from the posterior parietal cortex with executive inputs from prefrontal lobes. It projects to the primary motor cortex, which directly controls simple movements of the limbs. Both premotor and primary motor cortex project to the brain stem and the spinal cord. The spinal cord is the lowest level of the hierarchical organization that is directly responsible for executing movements (Kandel et al., 2000).

2.1.1 Distinction between planning and execution

The dominant view of how biological motor control takes place is that the brain forms a neural representation of a movement before its execution, within which it encodes certain parameters of the movement itself (Schmidt, 1988; Wiesendanger et al., 1992; Kandel et al., 2000). The concept of *motor program* has been introduced by Bernstein in 1967 (Kandel et al., 2000) and elaborated by Schmidt (1988). He argued that people do not learn specific movements, instead they construct generalized motor programs and produce different movements by varying the parameters (e.g., duration, level of force) that determine the way in which movements are constructed.

The existence of some abstract plan of action that is actuator–independent and specifies the important parameters of the movement to be executed has been particularly useful for explaining the phenomena of motor equivalence. *Motor equivalence* states that individual motor actions share important characteristics even when performed in different ways. Handwriting is perhaps the most representative example (Kandel et al., 2000).

The motor programming paradigm has been recently challenged by the dynamical view of behavior, which considers that stable patterns of behavior and the transitions between them emerge naturally from the dynamics of complex systems and do not exist *a priori* in some independent form (Kelso, 1988; Van Gelder, 1995). Further proposals have been made to suggest that the dynamical and computational view are nevertheless compatible, and that a dynamical system (e.g., the motor system) can be analyzed in terms of its intrinsic computational components (see Crutchfield, 1998).

The motor planning paradigm supports the hypothesis that planning (or response selection) and execution of movement constitute two separate stages of information processing (Bizzi and Mussa-Ivaldi, 2000). The common view on how these processes take place is that response selection and implementation are realized through a sequence of transformations between three major levels of representation: (1) *extrinsic kinematics*, such as motion of the hand through space; (2) *intrinsic kinematics*, such as joint motions; (3) *dynamics*, such as the causal forces that produce movement (Kalaska, 1995).

Evidence for the distinction between planning and execution came from two sources:

- Psychophysical experiments showing that movement kinematics, namely the transformation from extrinsic to intrinsic kinematics, are planned independently of the dynamics of musculoskeletal system in which movement occurs. Morasso's experiments first suggested that motor goals as simple as reaching and pointing are planned by the brain in terms of extrinsic coordinates representing the motion of the hand in space (Morasso and Mussa-Ivaldi, 1982; Shadmer and Mussa-Ivaldi, 1994; Bizzi and Mussa-Ivaldi, 2000).

- Neurophysiological data suggesting that motor cortex plans movement in spatial coordinates (see Section 2.1.2 below).

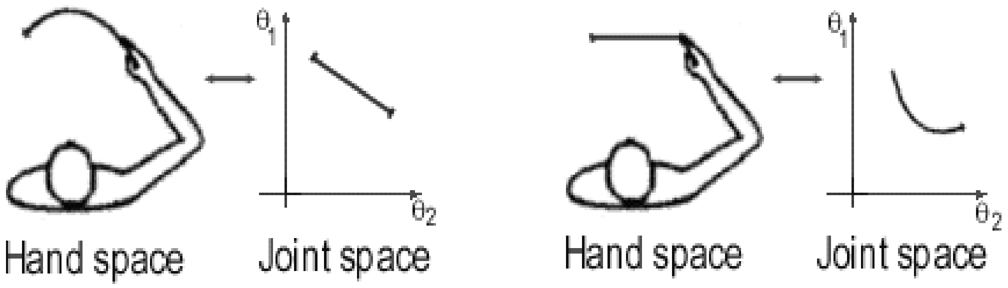
2.1.2 Motor planning in spatial coordinates

In the introduction to this chapter, it was mentioned that a long–standing controversy in biological motor control is the question about whether muscle dynamics or movement kinematics are represented in the motor cortex (Kalaska et al., 1992; Johnson et al., 2001; Flash and Sejnowski, 2001). The debate is directly related to the issue of which coordinate system the brain is using to encode movement: the *spatial* (extrinsic) coordinates frame, which represents movement in the Cartesian space or the *motor* (intrinsic) coordinates that represents motion in terms of the actuator dynamics, such as the joint and muscles coordinates. The answer to this question is highly relevant to understanding how visual to motor transformation takes place, and if the brain is directly involved in computing the inverse dynamics of movement, or only the kinematics transformations (see also discussion in Section 3.1.1).

If cortical planning were to take place in *motor* (joint or muscles) *coordinates*, it would have the advantage of taking into account the physical constraints but it may result in movements that are not adequate in the task space (Hildreth and Hollerbach, 1985). That is, because the joint–to–spatial coordinates transformation is a nonlinear mapping (see Section 3.1.1) and programming a straight line in joint coordinates would generate a complex curved end–effector trajectory (Figure 2.1a). On the other hand, planning in *hand coordinates* gives rise to straight hand trajectories, but requires more coordinate transformations and do not incorporate the limitations of the joints and muscles (Figure 2.1b).

Evidence for the motor cortex planning movement in spatial coordinates came from numerous neurophysiological experiments concerned with neural coding of the movement scalar parameters (i.e., direction, speed or amplitude) by recording the activity of single cells in the cortex (for a review see Georgopoulos et al., 1993; Johnson et al., 2001). The breakthrough work of Georgopoulos and colleagues (1982, 1984) demonstrates the existence in the motor cortex of the monkey, of a correlation between the cells' firing patterns and the direction of hand movement for reaching in two dimension. Further investigations demonstrate that *directional tuning* is a prominent feature of cortical neurons.

Directional selectivity has been described in the dorsal premotor cortex and primary motor cortex for movements in both 2-D and 3-D space (Schwartz et al., 1988), during manual tracking of visual targets (Johnson et al., 1999), for coding of finger and wrist movements (Georgopoulos et al., 1999), for drawing movements (Schwartz, 1994), and during instructed delay reaching tasks (Crammond and Kalaska, 1994; Johnson et al., 1999). Activity of motor



(a) Straight line in JOINT space

(b) Straight line in HAND space

Figure 2.1: Coding of movement in hand and joint coordinates. (a) A straight line in joint coordinates generates a complex curved hand trajectory. (b) A straight line in Cartesian coordinates requires a relatively complex elbow and shoulder joint movement (adapted after Hildreth and Hollerbach, 1985).

cortical neurons is also modulated by other parameters of movement kinematics, such as amplitude (Messier and Kalaska, 2000) and speed (Reina et al., 2001).

On the other hand, there is a large amount of experimental data suggesting that activity in the motor cortex reflects the onset and magnitude of muscle activity in single and multi–joint movements and implies that cortical motor neurons encode the dynamics of limb movements as well as a postural signal, instead of specifying only the kinematics of motion (Werner et al., 1991; Burnod et al., 1992; Scott and Kalaska, 1995; Scott, 1997).

Several attempts were made recently to explain these apparently contradictory findings (Wise et al., 1997; Kakei et al., 1999; Shadmehr and Moussavi, 2000). A unifying approach considers that experimental data actually reveals combinatorial properties of single neuron activity. In this view, the discharge rates of cells signal different scalar parameters of movement (direction, amplitude, or speed) as well as muscle forces and postural signals (Johnson et al., 2001). It was also suggested that motor cortical activity at the population level, may encode different types of information, which change from visual to motor aspects over time in a trial sequence (Zhang et al., 1997). Multiple coordinate systems and reference frames may also exist in the parietal and motor frontal lobes with the purpose of implementing a gradual mapping of visuo-motor information (Andersen et al., 1997; Johnson et al., 2001, see also Section 3.2).

To summarize, there is neurobiological and psychophysical evidence for a distinction at the cortical level between planning and execution of movement. The dominant view is that there is a cortical, abstract plan of action, which specifies the important parameters of movement, such as direction. It is also acknowledged that activity of motor cortical neurons co–varies with the dynamics of limb movements. However, it is believed that scalar parameters are encoded in the firing of a large number of the neurons that are active during the preparation and onset stages of movement.

2.1.3 Cortical encoding of movement direction

Apparently, the problem of coding the direction of movement has two possible solutions: the single cell representation or the neural population coding scheme. The simplest method would be to have single cells sharply tuned to one direction of motion, which would be activated only when movements in that particular direction are issued (such a model is implemented in Bullock et al., 1993). Instead, empirical studies of neural response show that in many brain systems, sensory information is distributed throughout a population of neurons (for a review see Abbott, 1994). In these ensembles, individual neurons are broadly tuned to a stimulus, in such a way that the individual firing rate reflects the information coded, however an accurate estimation of the stimulus can be realized only based on the spike trains coming from many neurons (Abbott, 1994; Rieke at al., 1997).

A *population coding* scheme has the advantages of suppressing the fluctuations in the signal of a single neuron and of being resistant to damage. Various methods have been developed to describe how spike trains from different cells are combined to reconstruct the properties of an external event (Abbott, 1994; Gielen, 2001; Gerstner and Kistler, 2002). A popular reconstruction method is the *population vector* scheme, a method to encode and decode stimuli in a distributed way by using the joint activities of a number of neurons (Georgopoulos et al., 1986; Seung and Sompolinsky, 1993).

Preferred directions of neurons and population coding

In their original research, Georgopoulos and his colleagues (1982, 1984) found that neurons in the premotor and primary motor cortex are selective to the direction of motion. Motor neurons seem to have a preferred direction, which is signaled by a change in the neural

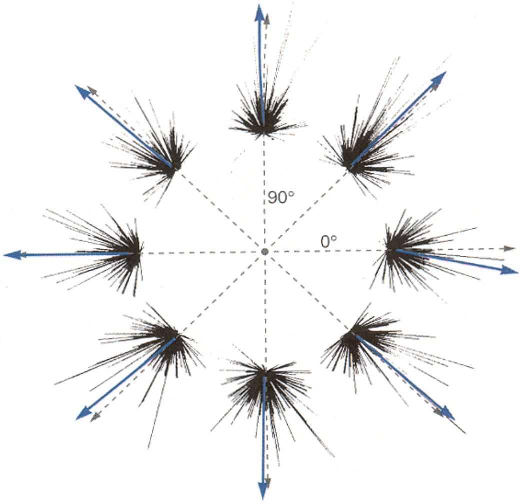


Figure 2.2: Cortical neurons with different preferred directions are active during movement in a particular direction. The eight clusters represent the activity of a population of neurons, during reaching movement in eight different directions. The population vectors (solid arrows) closely match the directions of movement of the limb (dashed lines) (adapted from Kandel et al., 2000 based on data from Georgopoulos et al., 1982).

activity with the direction of hand movement. The neurons fire most briskly for movements in a preferred direction and are almost silent for movements in the opposite direction.

The authors proposed that movement in a particular direction is determined not by the action of single neurons but by that of a broad population of neurons (Figure 2.2). The direction of movement at time t is estimated by a *population vector* $P(t)$ that yields the sum of each cell i preferred direction C_i, weighted by the cell activity V_i, given by the formula:

$$P(t) = \sum_i V_i(t)\ C_i. \tag{2.1}$$

The relation between the discharge rate of the neuron and the direction of motion was described by a *cosine tuning* function given by the formula $V_i(M) = b_i + a_i \cdot cos(\theta - \theta_i)$, where b_i and a_i are regression coefficients and $\theta - \theta_i$ is the angle formed by the cell's preferred direction and the direction of current movement M. More recent studies reveal that cells in the monkeys arm area can show bimodal tuning functions and that, their optimal tuning is actually narrower than the cosine function, namely with an average of 56° (Amirikian and Georgopoulos, 2000).

The population vector is a robust measure and a good predictor of motion direction during the preparatory stages of movement. It was shown that during an instructed delay period, the population vector gives a reliable signal concerning the direction of movement that is later executed (Georgopoulos et al., 1993). However, several studies recording neural activity in motor cortex during reaching described a 'rotation' of the neuron's preferred direction with the evolution of movement (Caminiti et al., 1991). At the population level, the average change was found similar to the change in the shoulder angle, suggesting that postural signals and muscles dynamics may also be reflected in the motor cells activities (Burnod et al., 1992). This illustrates that the interpretation of population activity depends on the neurons considered for averaging and on the proper characterization of neuronal response (Gielen, 2001).

Further studies of the cortical encoding of motion direction suggest that directional information can also be contained in the synchronous activity of the motor neurons (Hatsopoulos et al., 1998). Hatsopoulos and colleagues have shown that significant synchrony between directionally tuned neurons occurs clustered around the onset of movement (e.g., a time window of 400 ms) and may encode information distinct from that provided by firing rate modulations alone. The mechanism, which causes motor neurons to synchronize their activities, may depend on common input within the same area or from other areas, or may be due to network interactions among subsets of neurons coding for similar preferred directions (see discussion in Section 6.2.5).

With respect to how the central nervous system translates the cortical directional information into the desired limb movement, a simple neural networks mechanism was proposed by Georgopoulos and colleagues (Lukashin et al., 1996). This transformation was implemented with a three-layered feedforward network, which maps impulse activity recorded in the monkey's motor cortex into motor actions of a simulated actuator. Activities of units in the input layer are given by the spike trains recorded from 75 directionally tuned motor cells. The intermediate layer integrates these inputs by using a sigmoid activation function and maps them into an output layer that determines the contraction forces on the muscles by changing the rest lengths. The network connectivity is set so that the mapping from the output layer to the actuator determines a synergistic activation of muscles to generate the required motor output. As a result, the actuator responds to the motor cortical commands with a good fidelity. Another example of how everything can be combined, from plans to actions, is presented in the final chapter, where we discuss the possible applications of our work (Section 7.2).

2.1.4 Summary

The goal of the previous subsections was to define a theoretical framework for biological motor control to use in our simulation work. The modeling work described in the remainder of this thesis rests on three main theoretical assumptions derived from this framework:

- Brain centers form a representation of movement (i.e., plan) before its execution. Our working concept of motor plan considers the existence in the motor cortex, of a neural representation of movement, built in advance of its initiation. This concerns initial objectives of movement, without specifying details of implementation. We consider this hypothesis as being a weak assumption, completely supported by the existence in motor areas of preparatory activity that codes movement information prior to its onset (Johnson et al. 1999; Messier and Kalaska, 2000).
- This initial neural representation codes abstract parameters of hand-path. Based on experimental data on motor cells directional selectivity, we consider that activity occurring during movement preparation encodes the direction of whole limb movement. We also consider that neural activity in motor areas is correlated with the direction of movement immediately after movement onset (i.e., first 400 ms) (Georgopoulos et al., 1984).
- The direction of movement is coded in spatial coordinates and is represented in the activity of a large population of motor neurons. The former definition follows from the previous assumption and it states that the motor direction is represented in hand coordinates. The later assumption implies that directional information is provided by the population vector, which is considered an accurate predictor of movement (Georgopoulos et al., 1993).

In the remainder of this chapter, we focus on computational and modeling aspects of cortical directional selectivity. As such, we introduce in the next section cortical feature maps and the means to study their formation and development.

2.2 Self–organizing maps

Self–organization refers to spontaneous ordering tendencies observed in both artificial and natural complex systems that consist of a large number of components that interact simultaneously (Depew and Weber, 1999). The brain is such a self–organizing system that can learn by itself, by changing (adding, removing, strengthening) the connections between neurons (Kohonen, 1984; Kelso, 1995).

Sensory feature maps are an illustrative example of how the brain self–organizes to represent the external world by undergoing a process of abstraction, similar to the process of principal component analysis (Haykin, 1994). That is, it suppresses trivial details and maps the most important features of the input along the dimensions of the cortical map (Kohonen, 1984). A consequence of the tendency to compress the input space is the formation of *topographic maps*, in which the most important similarity relationships among the input signals are converted into spatial relationships among responding neurons (Kohonen, 1984; Ritter et al., 1992).

2.2.1 Principles of topographical maps organization

The topographic representation of input patterns is a key feature of brain design, for visual, tactile and auditory data analysis and also for motor control (Kandel et al., 2000). This general self–organization property of the cortex is supported by a set of functional and design principles.

Neural feature detectors. Feature detection is a basic principle of cortical processing. By the way each neuron is connected with other neurons within a network, it becomes specialized in representing certain features from the input space. The dedication of neurons is very widely distributed in the brain, characterizing the activity of neurons in all sensory (visual, auditory, somatosensory) and motor areas and employing different levels of the neural preference (e.g., from visual neurons sharply tuned to one attribute to the broadly tuned motor neurons) (Kandel et al., 2000).

It is important to note that the *optimal tuning width* of the cells that take part in the neural population code derives from trying to improve the quality of the code and to maximize information content and is influenced by many factors (i.e., noise, dimension of the encoded variable) (Eurich and Wilke, 2000; Gielen, 2001; Sompolinsky et al., 2001). Furthermore, the traditional concept of neurons as dedicated processors of a single parameter (i.e., either movement direction or force) has changed to reflect data showing that motor neurons, as well as neurons from the parietal and visual cortex, signal a mixture of scalars and vectorial parameters (Johnson et al., 2001; see Section 2.1.2).

Activity-dependent mechanisms. Map formation is an activity–dependent process, demonstrated by the existence of a critical period of development for the emergence of visual maps (Hubel and Wiesel, 1970). The mechanisms of plasticity on a medium time scale rely on activity dependence within intra–cortical circuits, with Hebbian preference to the most active inputs and competition between hidden nodes. The activity–dependent means are probably common to all cortical maps and play a role in normal functioning and the maintenance of topographic maps. Much support for these computational hypotheses has been brought by modeling studies simulating the formation and adaptation of sensory maps (for a review see Miikkulainen et al., 1998).

Organization. Topographical mapping leads to the formation of a map characterized by:

- Nearby cells react to stimuli with similar features.
- Whole input feature space be covered.
- No presence of global order.

Plasticity. For a long time, cortical maps were considered as being static in adult animals. The breakthrough research of Hubel and Wiesel first demonstrated that visual deprivation change physiological and anatomical monocular organization of afferents into visual cortex (Hubel and Wiesel, 1962; Hubel and Wiesel, 1970). Further evidence from studies on somatosensory deprivation in young animals (Merzenich et al., 1983) or nerve lesions (Sanes and Donoghue, 1992) have shown that re-organization is a common feature of adult cortical sensory maps. The experimental observations have been supported by computer simulations that illustrate how reorganization can take place after sensory deprivation (Ritter et al., 1992).

2.2.2 Characterization of sensory map layout

The primary visual cortex offers perhaps the clearest instance of feature map organization. This visual area is organized into narrow columns of cells, which have similar receptive fields and identical axes of orientation (Figure 2.3). Each orientation column also contains complex cells that respond to movement in a particular direction across their receptive field. The mapping of adjacent columns reveals a precise organization with an orderly shift in axis of orientation from one column to the next. Approximately each three-quarter of a

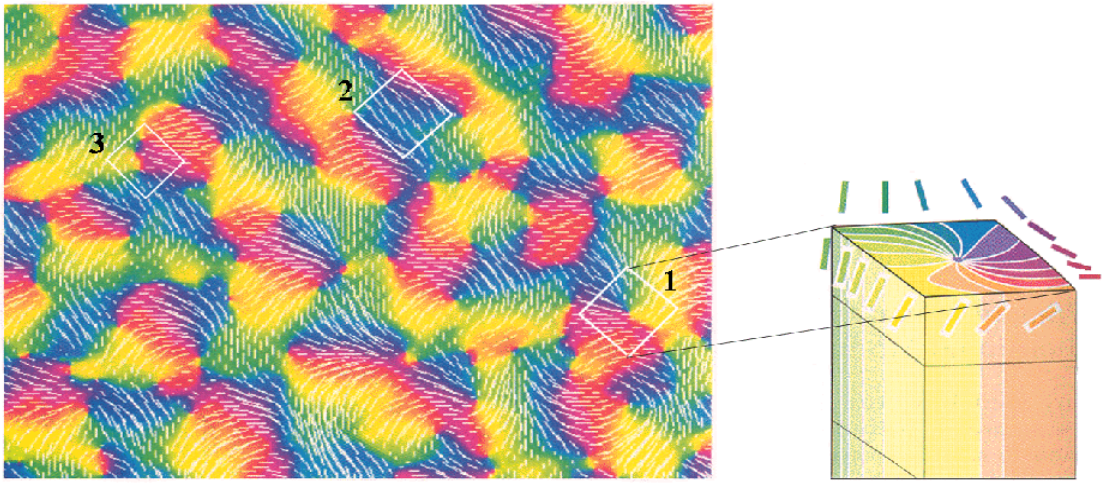


Figure 2.3: Orientation map in the macaque cortex. On the left, cortical areas that are most actives at the presentation of a particular orientation are indicated by the corresponding orientation color (bars on the right). The map shows features such as: (1) *pinwheel centers*, around which orientation preference changes through 180°; (2) *linear zones*, where orientation changes almost linearly; and (3) *fractures*, characterized by a discontinuous change of orientation preferences. On the right, the enlargement of a pinwheel-like area is presented (adapted after Kandel et al., 2000, based on data by Blasdel, 1992).

millimeter contains a complete cycle of orientation changes (e.g., pinwheel center). A complete sequence of ocular dominance columns and orientation columns forms a *hypercolumn*. This is the basic computational module of the visual cortex, which processes information on orientation, binocular interaction, color and motion (Kandel et al., 2000).

The visual orientation map in Figure 2.3 illustrates three general principles of cortical design: local correlation, homogeneity, and isotropy (Niebur and Wörgötter, 1992). Within a hypercolumn, isotropic subregions or linear zones are formed where neurons share the same orientation preferences (i.e., are locally correlated). The map is homogeneous, because there are no systematic differences between locations over distances much larger than a hypercolumn width.

Formation of visual feature maps has been extensively modeled, mainly by using variants of the self-organizing feature map (SOM, Kohonen, 1984) with receptive fields and plastic lateral connections (Obermayer et al., 1990; Sirosh, 1995; Miikkulainen et al., 1998; see also Sections 2.2.4, 2.3.1). Well-studied examples are also the formation of tonotopic maps in the auditory cortex of the bat, that is mapping of sound frequencies (Kohonen, 1984; Martinetz et al., 1988; Kandel et al., 2000) and organization of somatosensory maps for the mapping of touch (Ritter et al., 1992).

2.2.3 Motor map organization

In contrast to the detailed knowledge existent on the functional architecture of sensory cortical areas (visual and auditory), the organizational principles of the frontal motor regions are much less clear (Sanes and Donoghue, 1992). Without an explicit description of the architectural and functional principles, modeling of motor cortical computations is more difficult. Our view is that there are two milestones in modeling motor cortex development: (1) the characterization of connectivity in motor cortex and (2) understanding the nature of the training information which drive this process.

The latter requirement lies at the heart of a self–organizing process. That is, organization of a feature map follows from projecting the similarity relationships between input signals into spatial relations in the output map units. Whilst these relationships are easily recognized in the input space of the visual (retinal receptive fields), somatosensory (touch receptors signals), or auditory (sound frequencies) mapping, in the case of the motor cortex it is less obvious what sort of data is available for training and which are its characteristics.

With respect to the organization of directional neural maps in the motor cortex, several points are noteworthy: (1) neurons are broadly tuned to the direction of arm movement, with a unimodal tuning, close to the cosine-function; (2) cells in a motor cortical column tend to have similar preferred directions; (3) particular directions are multiply represented in the motor cortex; and (4) neural preferred directions cover a directional continuum (Georgopoulos et al., 1984; Lukashin and Georgopoulos, 1994). Experimental findings have also showed that the strength of temporal correlation between the firing of single cells decreases with inter–electrode distance (Hatsopoulos et al., 1998). This suggests the existence of local, short–range excitatory synapses. Concerning the functional characteristics of lateral connections in the motor cortex, Georgopoulos and colleagues (1993) estimated that the strength of connection between two motor neurons is negatively correlated with the difference between their preferred directions (see also our results in Section 6.1.5).

To summarize, cortical areas in the brain undergo a process of self–organization that is driven by activity–dependent mechanisms and consists of a topology preserving mapping. It leads to the specialization of neurons as feature detectors and the formation of topographical cortical maps, in both sensory and motor areas. The layout and development of topologically ordered maps was best described and modeled in the case of sensory (i.e., visual) maps, starting decades ago with the seminal work of Hubel and Wiesel (1962).

Nevertheless, it is believed that development of cortical preferences may be based on a few design principles, which in turn rely on very general developmental mechanisms utilizing the input structure of the system (Douglas and Martin, 1991; Niebur and Wörgötter, 1993). In this thesis, we adopt the view that developmental principles described for sensory areas reflect general laws of cortical organization. Accordingly, they form the basis of our motor cortex modeling work. Up to the present, this assumption was confirmed by most studies providing evidence on motor cortex connectivity and plasticity (Georgopoulos et al., 1993; Hess and Donoghue, 1994; Rioult-Pedotti et al., 1998; Xing and Andersen, 2000).

2.2.4 Role of lateral connections

The previous sections described the characteristics of cortical maps with respect to neuron preferences within a columnar organization. The vertically oriented computational units (i.e., neural columns) communicate with one another by means of *horizontal* or *lateral connections* (Kandel et al., 2000). Until recently, the lateral connections were thought to have a secondary role in shaping the cortex, with the primary role attributed to plasticity of thalamocortical afferents (see Miikkulainen and Sirosh, 1996).

New experimental and modeling studies suggest that lateral interactions might play a much larger role in modulating and controlling the cortical response, in the representation of information and development of cortical maps (Douglas and Martin, 1991; Sirosh et al., 1996). Several computational functions of lateral connections have been hypothesized:

- Recurrent lateral connections may provide a mechanism for activity normalization (Somers et al., 1996) and mediate competition and synchronization over large distances of cortex (Usher et al., 1996).
- Lateral connections may form the substrate for encoding memories as attractors in the cortical network (Taylor and Alavi, 1996).
- They could play a crucial role in the development of cortical columns representing orientation, ocular dominance and spatial frequency (Sirosh, 1995; Miikkulainen et al., 1996). The may also mediate reorganization of cortex in response to drastic changes

> in the input environment, such as retinal lesions and input deprivation (Gilbert and Wiesel, 1992).

As mentioned before, most of the knowledge about the development and functions of lateral connections has been based on studies of visual areas organization (Gilbert and Wiesel, 1992; Weliky and Katz, 1994; Douglas et al., 1991). Only recently, studies on the functional topography of the auditory cortex (Read et al., 2001) and on the organization of cortical maps on motor cortex (Sanes and Donoghue, 1992; Hess and Donoghue, 1994) brought essential information on the spatial arrangements and the functions of horizontal connections in other, than visual brain areas.

Lateral connections are found to be remarkably ordered. Horizontal connections wire together cells with similar response properties in different columns. In the visual cortex, horizontal projections link columns with common ocular dominance and orientation selectivity (Gilbert and Wiesel, 1992), as well as color selective cells with similar responses from different blobs (Kandel et al., 2000). In the auditory cortex dorsoventral connections link regions with matched characteristic frequencies (Read et al., 2001). In the motor cortex lateral connections may link neurons with similar proffered directions (Georgopoulos and Lukashin, 1996). This organization is neither genetically determined nor static. The development of lateral connections, like that of afferent connections, depends on cortical activity caused by external input and represents correlation in the input (Hubel and Wiesel, 1962; Hess and Donoghue, 1994).

Lateral connections are often reciprocal, and contact both inhibitory and excitatory cells. Roughly 15% to 20% of the neurons in the cortex are GABAergic inhibitory interneurons (Kandel et al., 2000). The importance of these neurons for controlling the positive feedback loops between excitatory cortical pyramidal neurons has been revealed, for example, in the epileptic-like effects of GABA antagonists (Grinvald et al., 1988).

Physiological studies on the cells of cat visual cortex suggest that individual axons inhibit isotropic nearest–neighbor cortical columns and outside this core region may provide a type of anisotropic lateral inhibition of cortical columns (Budd and Kisvárday, 2001). Mechanisms for long–range inhibition have also been described across the brain, possibly implemented by subcortical mechanisms (i.e., thalamus and basal-ganglia) (Taylor and Alavi, 1996). Long–range competition is an important requirement for self–organization to occur and, generally, realistic models of the cortex include both excitatory and inhibitory connections, with inhibition being spread further than excitation (Kohonen, 1984; Sirosh, 1995).

To this point, we have briefly described the neurobiology of cortical feature maps, the way the topology preserving maps form and adapt in the brain. In the remainder of this chapter, we focus upon the modeling of this process in general, and of cortical directional map development, in particular.

2.2.5 The Kohonen map

The self–organizing feature map (SOFM) was developed by Kohonen (1984) as a means to explain the formation, in the sensory cortex, of ordered, two-dimensional representations of multidimensional external world signals. Kohonen defined the self–organization process running in the cortical maps as a *topology–preserving mapping*, which translates similarity relations existent in the input space in topological relations between the projecting images. Within this adaptive process, nodes in the output map converge to such values that every unit becomes sensitive to a particular domain of input signals, in a regular order (Kohonen, 1984).

A basic Kohonen map consists of an input layer and an output map (also termed as competitive layer). The input can be a multidimensional pattern represented by a vector, with each unit coding the values of a dimension from the input pattern. Every unit in the competitive layer receives a sum of weighted values from the input layer and is connected with a number of other competitive units, by excitatory and inhibitory lateral synapses (see Figure 6.1 in Section 6.1.1).

Kohonen maps are able to preserve the similarity relations in the input space, while realizing a *dimensionality reduction*, usually from a multidimensional input to a two–dimensional (planar) output. The striped ocular dominance patterns are perhaps the most illustrative example of mapping a three dimensional space (e.g., the visual world) into a two dimensional surface (e.g., the cortex) (Kohonen, 1995).

The core of learning relies on the formation of ’clusters’ or ’bubbles of activity’, which concentrate the neural response around some location in the output plane (e.g., around the maximally responding unit). The location c of the maximum is defined according to a *best match* criterion between the input vector x and the weights vector w of each output unit i:

$$|x - w_c| = \min_i |x - w_i|. \tag{2.2}$$

Next, around the maximally responding unit c, a topological neighborhood N_c is defined, and the weights of all units within this neighborhood are subject of adaptation. The modification of the selected weights w_i at the time $t+1$ is given by the formula:

$$w_i(t+1) = w_i(t) + \eta[x(t) - w_i(t)], \quad \text{for } i \in N_c \tag{2.3}$$

$$w_i(t+1) = w(t), \quad \text{otherwise} \tag{2.4}$$

where η is the learning rate. The topological neighborhood N_c starts fairly wide in the beginning to allow the initial formation of the correct order and than is shrinked with time until it comprises only one unit. In a similar manner, the learning rate η is slowly decreased, such as for the final convergence of the map a low value of it is used (Kohonen, 1995).

The learning process resides in a 'winner-take-all' strategy, where nodes in the output layer are competing each other to learn the input vectors. During the presentation of each input, the maps adapts in two ways. First, the weight vectors become a better approximation of the input vectors, and second, the neighboring weight vectors become more similar. Kohonen demonstrated that, combined, these two adaptation processes ensure the ordering of the output map, so that the afferent weights w_{ij} will become an ordered image of the input vector x. An essential role in the formation of a localized neural response of the network is played by the lateral feedback system, with short-range excitation and long-inhibition. This will be referred to, as a Mexican-Hat interaction profile (see Figure 5.8 in Section 5.3.2).

Variants of SOFM have been applied successfully to a huge number of domains, ranging from monitoring and control in industrial tasks and telecommunications, to medical applications and robotic-arm control. Currently, the SOFM appears to be the *de-facto* standard for biological modeling of cortical features map formation (for a review of self-organizing feature maps applications see Kohonen, 1995).

Spiking self–organizing feature maps

Simulation of cortical maps formation using competitive Hebbian models was most commonly implemented based on the classical model of continuous, rate coding neuron (Obermayer et al., 1992; Sirosh, 1995; Miikkulainen et al., 1998). In the last decade, alternative neural models, known as spiking neurons, have been developed in computational neuroscience (Maass, 1997; see also Section 4.2.1). Conversely, the research focus was on exploring the capacity of these neurons to perform fast computations in terms of single spiking events.

Much of the work done in this direction was oriented towards the implementation of spike–based Hebbian learning algorithms (Song et al., 2000) and less has been done with respect to computations in spiking Kohonen maps. The few attempts made in this direction (Choe and Miikkulainen, 1995; Ruf and Schmitt, 1998; Panchev and Wermter, 2001) propose different ways of adapting the plastic weights, dependent on where they consider that learning information is contained.

Choe and Miikkulainen (1995) adapted a well–known model of a laterally interconnected self–organizing map (LISSOM; Sirosh, 1995) to study the development of orientation selectivity in a network of spiking neurons. Their model owes much to the traditional firing rate coding scheme, since learning of both afferent and lateral connection strengths is applied in terms of neural discharge rates, after the network reaches a stable state of firing. The resulting learning scheme is slow and does not take into account the information possibly encoded in the timing and order of the spikes. Hence, it generally neglects the advantages of spiking neural models.

A different approach to self–organization based on the timing of *single firing events* was proposed by Ruf and Schmitt (1998). They developed a learning rule for a network of leaky integrate–and–fire neurons, connected by short–range excitatory synapses and long–range inhibitory connections. When an input vector s^l is applied to the competitive network, each output node receives a weighted sum of the afferent signals over all input units i, $\sum_i w_{ij} \cdot s_i^l$. The product $w \cdot s^l$ represents a measure of similarity between the two vectors with respect to the Euclidean distance (Kohonen, 1984). Hence, the earlier an output node fires, the more similar its weight vector is to the input vector. Therefore, the competitive node whose weights represent the best match of the input vector, namely the winner, will fire first.

Based on this logic, the authors defined learning of the afferent weights of output nodes in terms of first spike timings. The weight of the connection w_{ij} between the input neuron i and the competitive neuron j is updated by the rule:

$$\Delta w_{ij} = \eta \, \frac{T_{out} - t_j}{T_{out}} \, (s_i^l - w_{ij}), \tag{2.5}$$

where t_j is the firing time of the j neuron and T_{out} is a time out limit.

Instead of the classic topological neighborhood, a temporal neighborhood of the winner is implemented, based on the idea that the firing of a winner neuron drives the firing of topologically close neurons and postpones the firing of remote neurons (e.g., due to the Mexican–Hat–like shape of the connectivity pattern). Hence, learning applies only to the neurons which fired until T_{out}. Moreover, the adaptation of afferent weights is scaled by the distance between the firing time of the competitive neuron and the firing of the winner. This learning rule has been tested with one and two-dimensional input patterns, on small scale networks (10 and 25 neurons), where the lateral weights were slightly decreased during training (no learning rule was implemented for the lateral synapses) (Ruf and Schmitt, 1998). A variant of this algorithm was recently presented by Panchev and Wermter (2001), who applied to afferent weights a variant of spike–dependent learning, based on the relative timing of the pre- and post-synaptic spikes.

Regarding the biological plausibility of the learning algorithm described above, we can consider the studies on visual information processing by Thorpe et al., (1996). Authors suggested that during visual object recognition, the brain does not have time to evaluate more than one spike from each neuron per processing step. Accordingly, Thorpe and Gautrais (1998) proposed a coding scheme based on the *time–to–first–spike*, where only the first spike of each neuron counts. This idea was supported by other experimental studies, which have showed that most of the information about a new stimulus is conveyed during the first 20 or 50 ms after the onset of the neuronal response (Tovee et al., 1993; VanRullen and Thorpe, 2001).

Hebbian learning in the lateral synapses

Section 2.2.4 emphasized the importance of horizontal connections in modulating cortical neural activity and development of brain feature maps. However, the original self–organizing feature map (Kohonen, 1984) and many of its variants account only for static lateral feedback. This means a fixed connectivity, which has a role in shaping the network activity, but is not subject to learning.

Recently, significant efforts have been made in studying the computational role of *plastic lateral connections* within artificial neural network models of brain cognition (see Miikkulainen and Sirosh, 1996). Most learning rules for lateral weights are based on the hypothesis of Hebb, which observed that development in many parts of the brain appears to result due to correlation–based synaptic adaptation processes. He proposed that

> When an axon of cell A is near enough to excite cell B and repeatedly or consistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased (Hebb, 1949, page 62).

Sirosh (1995), Miikkulainen and Sirosh (1996) were among the first who considered the role of Hebbian adaptation on the lateral synapses during the simulation of a visual feature map formation. The LISSOM model (Laterally Interconnected Synergetically Self–Organizing Map, Sirosh, 1995) applies learning to both afferent and lateral connections. First, excitatory and inhibitory lateral connections strengthen by the correlated activity of pre- and post-synaptic neurons, computed as the neural activation after the map has settled in a stable bubble. Second, normalization is applied, so that the total synaptic strength of the lateral connections remains constant. Connection death is also allowed, by the fact that synapses whose weights become very small are pruned.

Their success in modeling the formation of visual feature maps have brought compelling evidence that lateral connections develop simultaneously with the afferent weights, depend on cortical activity caused by external input, and represent the input correlation. The role of learning in lateral synapses has been investigated within models studying the formation of orientation (Miikkulainen et al., 1998) and directional maps (Farkaš and Miikkulainen, 1999), dynamic receptive fields (Sirosh et al., 1996), synchronous oscillations (Wang, 1996).

More recently, experimental neurophysiological investigations have shown that the relative timing of the pre- and post-synaptic spikes plays an important role in determining whether a synapse is potentiated or depressed (Markram et al., 1997; Zhang et al., 1998). Newer formulations of the Hebbian rule have been explored for learning in pulsed neural networks (Roberts, 1999; Song et al., 2000; see also Section 4.4).

In summary, in this section we introduced the design and developmental principles of cortical feature maps. The self–organizing feature map was presented, as one of the most successful modeling paradigms of cortical map development. The role of lateral feedback system in the organization process was outlined, with a focus upon the plasticity of these synapses. A learning procedure for a spiking self-organizing map, in terms of the timing of single spiking events was described. The modeling work presented in this thesis explores the simulation of directional motor map formation based on self–organization with spiking neurons and plastic synapses.

2.3 Models of cortical coding of directional selectivity

This final section on biological motor control reviews a number of neural network models of cortical directional selectivity. Firstly, the simulation of a directional map development in the visual cortex is presented. It was argued throughout the previous section that the existence of neural feature detection and topographical maps in the brain might reflect the manifestation of general design and developmental principles. Consequently, in our work, the motor cortex modeling will be inspired by both neurobiological data and simulation results on sensory map development. It is the scope of modeling work to explore which of these principles and in what context, can they lead to the development of motor directional maps.

Secondly, in the absence of modeling work on directional selectivity development in motor cortex, inspiration can be drawn from two sources: simulations which address population coding of motion trajectory (Lukashin and Georgopoulos, 1994) and arm-reaching studies that require representation of directional information in the motor network (Bullock et al., 1993). By reviewing these studies, we hope to ascertain how motor directional selectivity might be implemented and whether it would be useful to have a model which develops such a feature by unsupervised means.

2.3.1 Visual directional map formation

Most of the models of visual directional selectivity address the processes by which single complex cells develop directional preferences due to the spatio-temporal properties of their receptive field (Maex and Orban, 1996; Blais et al., 2000). With respect to feature map formation, the development of orientation and ocular dominance columns most frequently studied was (Sirosh, 1995; Erwin et al., 1995; Miikkulainen et al., 1998). Nevertheless, orientation and directional maps in the visual cortex are functionally and anatomically related, reflected by the fact that many of the cells with orientation preferences show orthogonal directional selectivity (Kandel et al., 2000).

Recently, it was proposed to account for the emergence of directional selectivity in the primary visual cortex (V1) through self-organization, in a similar way with the development of orientation or ocular dominance preferences. Farkaš and Miikkulainen (1999) built a model of directional selectivity, based on a self–organizing map of leaky–integrator neurons with

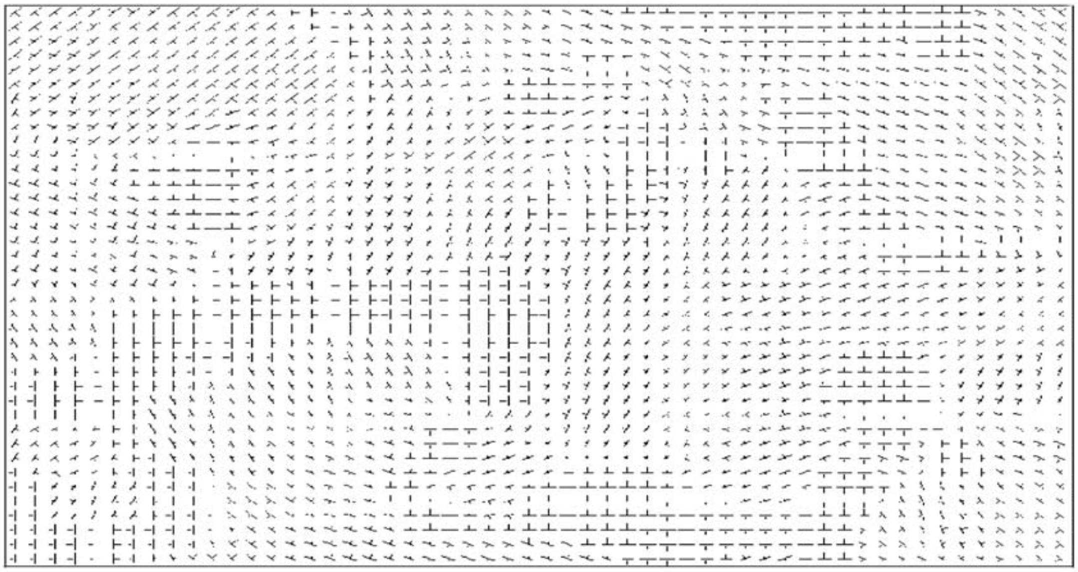


Figure 2.4: A fragment from the self–organized direction and orientation map in Farkaš and Miikkulainen (1999). Each neuron is marked with a line that identifies its orientation preference and perpendicular to it is represented the shorter line of directional preference. The length of lines is proportional to the neural selectivity. Note that most of the neurons become directional selective and only few of them show no selectivity.

sigmoid output functions. The input to the map consisted of 16 sequences of moving bars, whose direction of motion was perpendicular to their orientation. Only afferent weights were subject of learning, according to the standard Oja's rule. That is, the weight changes proportional to the difference between the current input and the back–propagated output (Oja, 1982).

After training, a neuron's *preferred direction* was defined as the direction for which the neural output was highest. Each neuron in the map is also assigned orientation preference and selectivity. A fragment of the self–organized directional and orientation map is shown in Figure 2.4. The trained map shows several major characteristics described in biological directional maps: (1) most of the orientation–selective neurons are also directional selective; (2) neurons' preferred directions are perpendicular to their preferred orientation; and (3) most of iso–orientation patches contain subregions with opposite direction selectivity.

In terms of the neuron response profiles, the model could not reproduce some aspects of the behavior of biological directionally tuned cells. Thus, the response of a model neuron to movement in the direction opposite to its preferred direction was found to be rather higher compared to the response measured in the brain maps. This excessively broad tuning of neurons might be due to the absence in the model, of the lateral feedback, and particularly of synaptic inhibition, which is known to provide the mechanism by which unwanted responses are suppressed (Livingstone, 1998). A more recent model of visual directional map formation has been proposed by Tversky and Miikkulainen (2001). The authors focus on implementing a delay–adaptation learning rule and explore its effectiveness for the development of directional selectivity in a laterally connected self–organizing network of integrate–and–fire neurons.

In a summary, only recently the formation of directional selectivity has been tackled within the self–organization paradigm, in a similar manner with the development of other neural response properties. Current models draw inspiration mainly from orientation maps simulation, but much remains to be specified in order to obtain directional maps (visual or motor) that are consistent with the biological observations.

2.3.2 Models of motor control of directionality

Our review of previous work on motor cortex modeling showed that only a limited number of simulation studies address issues related to the organization and development of motor directional selectivity. So far, we are not aware of any simulation work aimed at modeling self–organization of motor cortex, in a similar way with the models of visual map formation (see Section 2.3.1). Instead, we will review in the remainder of this chapter two categories of models with relevance to our goals: (1) neural network models concerned with cortical coding of movement trajectory in the directionally tuned motor populations of neurons (Lukashin and Georgopoulos, 1994; Lin et al., 1997); (2) simulation studies of visually guided reaching, which require processing of directional information by the motor cells (Bullock et al., 1993; Burnod et al., 1992).

Coding of trajectories by neural population vectors

It was pointed out in Section 2.2 that, compared to the experimental data available on the visual cortex, much remains to be described with respect to motor maps organization and functional principles. Recent modeling work by Lukashin and Georgopoulos (1994) produced compelling evidence that the strength of connection between neurons in a pair is correlated with the similarity of their preferred directions. This neural network simulation was aimed at studying how different trajectories are encoded by the time series of neuronal population vectors. At the initialization, each neuron in the network was assigned a preferred direction of motion. The neural trajectories coded by the network were obtained by adding together, head–to–tail, the neural population vectors computed at short successive intervals of time (see Equation 2.1). Next, the network was trained using the simulated annealing learning to find a set of afferent connection strengths, which minimize the error between the desired and actual movement trajectory.

After training, the organization of the network led to different trajectories realized by different neuronal subsets. The authors hypothesize that similarly, at the motor cortex level, trajectories may be encoded by different sets of synaptic weights that permanently store the information about the essential parts of hand path and do not change during the movement. Consequently, the realization of a particular trajectory needs only one motor command, that is a global activation of an appropriate neuronal subset. As already observed, their study also suggests that neurons with similar preferred directions tend to be mutually excitatory and those with opposite preferred directions tend to be inhibitory.

Lin et al. (1997) presented another study dealing with coding of directional information in the firing activities of neurons in motor cortex. Using data collected in the macaque cortex (Schwartz, 1994) the authors trained a self–organizing feature map to extract the directional patterns encoded in the discharge rates of neurons. First, the average discharge rates d_j of $n = 81$ motor neurons were computed during the execution of 20 ms of in-out finger movements. The inputs to the self–organizing map consisted of the discharge rate vectors $X_j = [d_{1j}, d_{2j}, ..., d_{nj}]$ formed from all n individual cells contributions.

After training, the topology–preserving mapping property of the SOM led to the formation of an output map that represents the similarity relationships between the inputs. Thus, discharge rate vectors coding for similar directions were mapped in neighboring nodes, while input vectors coding for opposite directions were classified according with their high degrees of dissimilarity. The findings of this study have partially inspired the way the training set of our model on motor cortex self–organization was generated (see Section 6.1).

To conclude, previous work on trajectory coding by population vectors provided important insights into the organization of network connectivity and on how directional information can be read out from neurons discharge rates. What is still missing is a developmental model of how directional selectivity and population coding emerge in the motor cortex.

Representation of directional information in models of motor control

An important stream of models which deal with motor directional selectivity are robotics studies implementing spatial to end–effector direction mapping strategies (Ritter et al., 1989; Bullock et al., 1993; Fiala, 1995). This approach focuses on solving a transformation from spatial trajectory to end–effector *directions*, as opposite to the mapping from spatial trajectory to end–effector *positions*. There are several computational advantages to spatial–to–direction mapping systems, such as continuity of trajectories and motor equivalence, appropriateness for tool use and robustness when unexpected events occur in the environment (Bullock et al., 1993).

The main application of direction mapping networks is for the visual control of arm–reaching. In the DIRECT model (Bullock et al., 1993), spatial directional information in body–centered coordinates is transformed into joint–rotations (i.e., directions) commands. For this purpose, the signals on the current position of the arm and the directional information on the desired trajectory are combined via a self–organizing *position–direction* map. Learning is conducted so that, each cell in the map becomes sensitive to a particular spatial direction in a particular position of joint space. It results that 10 k units are needed.

The sharp directional tuning of the neural response opposed to the broad tuning of real cells is motivated as a compromise of the model biological plausibility in favor of system performances. In accord with the experimental data is the fact that neural activity correlates with the arm movement direction, but not arm movement end-point and also that activity level may be primed prior to movement (i.e., preparation of movements).

Another group of researchers who have explored the use of directional information for the control of visually guided movements is represented by Burnod and colleagues (Burnod et al., 1992; Caminiti et al., 1992; Baraduc et al., 1999). Burnod et al. (1992) presented a model which was largely inspired by neurophysiological data on the mechanisms of visuo–motor transformation (see detailed description in Section 3.3.3). With respect to directional information coding, the behavior of the network is in accordance with some of the experimental results described in the motor cortex (Caminiti et al., 1992): (1) cells in the model are tuned to a preferred direction depending upon the initial position of the arm; (2) the population vector predicts the movement direction; and (3) the preferred direction of a single neuron rotates with the arm within a specific subspace.

More recently, Baraduc et al. (1999) proposed a model of motor cortical command to investigate the population coding of direction during reaching movements. The neural network architecture consists of three populations of neurons (visual, proprioceptive and motor) with directional and positional tuning properties that ensure a distributed coding of direction in both spatial and motor coordinates (see Section 3.3.3). Based on their findings, the authors suggest that the neural population vector resulted in motor area does not give a faithful image of cortical processing during arm reaching. However, we point out that motor population coding results as an effect of visuomotor mapping learning and occurs in a network with fixed lateral synapses.

To conclude, there are currently a number of models of visuomotor control that implement directional selectivity of motor cells. However, they are either based on sharply directionally tuned neurons (Bullock et al., 1993) or address the motor control of directionality by using supervised means and only within the process of visuomotor learning (Baraduc et al., 1999). Our simulation work of motor cortex organization is aimed at developing directional selectivity and population coding by using the spatio–temporal characteristics of information processing in a pulsed self-organizing feature map (Section 6.1).

Chapter 3

Visuomotor development

Sensorimotor coordination has been an active research topic for both neuroscience and artificial intelligence over the last decade. The integration of sensory information for movement guidance represents perhaps the most basic operation that a nervous (or artificial) system must solve (Churchland and Sejnowski, 1992). Despite exploring the same problem, studies carried out in these fields have arrived at different solutions. These differences arose mainly due to the divergent research goals.

Robotic models, rather than revealing how nervous systems attain sensorimotor control, demonstrate that many schemes are capable of implementing sensorimotor coordination and focus on the accuracy of controlled behavior, often without considering the biological plausibility of the resulting implementation (Kalaska, 1995). Nevertheless, their capabilities are still behind the adaptive motor abilities of biological organisms. Conversely, significant progress has been made in neuroscience towards understanding the biology of sensorimotor transformation in terms of integrative mechanisms that function at the single cell level or in small–networks of neurons (for a review see Kalaska et al., 1997; Snyder, 2000).

Several attempts were made recently to create an integrative framework that links neurophysiological and computational aspects of sensorimotor coordination. The aim is to provide a conceptual scheme that is readily implementable, to be used for the biological modeling of visually guided movements (Bullock et al., 1995; Burnod et al., 1999). The purpose of this chapter is to describe a general theoretical and modeling framework of sensorimotor development, inspired by these recent neurophysiological findings and computational theories.

Section 1 introduces the main research themes on biological sensorimotor coordination. First, a general description of the classical approach to solving the sensorimotor mapping problem is presented. Following this, we present a number of alternate hypotheses that have recently been proposed. Section 2 focuses upon the neurobiological mechanisms underpinning visuomotor mapping for arm reaching. Neurophysiological evidence is presented for the hypothesis of progressive match of visual to motor information, through intermediate spatial representations and supported by multimodal combinatorial properties of neurons. Finally, Section 3 reviews a number of biologically inspired models of visuomotor coordination development.

3.1 Sensorimotor coordination in biological systems

The general understanding of the sensorimotor coordination problem is that it represents a mapping of signals from various sensory modalities onto an appropriate set of efferent motor commands addressed to skeletal muscles or robotic actuators (Massone, 1995). A number of different aspects contribute to the difficulty of this task. First, investigations of the neural circuits involved in this process in the primate brain are faced with significant issues of complexity. This is because, it is not obvious how the perceptual areas are connected with the motor regions for sensory information transfer (Glickstein, 2000; Kandel et al., 2000). Second, in order to guide motor actions, the visual signals undergo a translation into motor outputs, which is commonly described as a non–linear coordinate systems transformation. Solving this mapping is again a non–trivial problem (Pellionisz and Llinas, 1980; Hildreth and Hollerbach, 1985; Jordan, 1996).

As a result, significant efforts in the last few decades have been devoted on one hand, to identifying the connections between sensory and motor areas in the brain (Tanne et al., 1995; Glickstein, 2000) and on the other hand, to developing various conceptual and modeling schemes that can solve the coordinate system transformation problem (see Kalaska et al., 1997; Snyder, 2000). These research directions are briefly discussed in the first part of this section, referred to, as the classical paradigm. Following this, we introduce three new theories, which refine the way researchers look at the visuomotor mapping problem.

3.1.1 The classical approach

Coordinate system transformation and the inverse kinematics problem

The concepts of reference frames and coordinate systems are used widely in the study of eye and limb movements. A *reference frame* is invoked when an experimental result is described: for example, describing the position of eyes relative to the head (a head-fixed frame of reference) (Soechting and Flanders, 1995). A *coordinate system* represents the set of axes fixed to the frame of reference, used to make the measurement (i.e., Cartesian coordinates). There are three main types of coordinate systems: *end-point coordinates* (e.g., retinal or Cartesian), *generalized coordinates*, that are independent variables describing the dynamics of a system (e.g., joint angles), and *actuator coordinates* dependent on how motor control is implemented (e.g., muscle forces) (Mussa-Ivaldi, 1995). For each coordinate system there are several possible reference frames: eye–centered, head–centered, limb–centered (Andersen et al., 2000).

The problem of visuomotor coordination was first described as a *coordinate system transformation* by Pellionisz and Llinas (1980). Given the fact that sensory input and the motor output vectors are represented in different coordinate systems, the transformation of sensory signals to motor commands can be described as matrix multiplication. And this represents the standard technique for solving the geometrical problem of going from one coordinate system to another (Pellionisz and Llinas, 1980).

The transformation of one class of coordinates to another is a nonlinear mapping. The translation from Cartesian coordinates to the joint angles required to move the arm to the target is referred as the *inverse kinematics problem*. The mapping from desired joint angles to muscle activities represents the *inverse dynamics problem*. The opposite process, which computes first the motor commands and translates them into hand movements, is referred as the *forward kinematics*. A biological / artificial system that has to perform the transformation from extrinsic to intrinsic coordinates must learn the spatial transformations (i.e., the inverse kinematics) and the muscles dynamics (i.e., inverse dynamics) (Ghez et al., 2000).

A common approach to sensorimotor learning in artificial systems is to learn first an inverse model, namely the transformation from desired movements to motor commands, which will further be used to control the end–effector (Massone, 1995; Jordan, 1996). There are various methods by which such an inverse model can be learned, mainly based on *error correction mechanisms*, which translate errors expressed in the Cartesian (visual) coordinate system into the motor command or other coordinate system errors (for a review see Jordan, 1996; see also Section 3.3.2).

Nevertheless, it is an open question whether biological processes for inverse kinematics or inverse dynamics exist, or whether the nervous system effectively computes the spatial error between hand and target (Hildreth and Hollerbach, 1985). Despite the aptness of the coordinate system transformation concept as a description of neurophysiological processes, the question has been raised whether or not it has a biological relevance (Robinson, 1992; Kalaska, 1995; Burnod et al., 1999). Robinson (1992) argues that coordinate systems are a human invention for measuring spatial relationships and they are irrelevant for neural networks performing visuomotor transformations:

> Mathematical descriptions of what a system is trying to do are of little help to the neurophysiologist trying to understand how real neurons do it (Robinson, 1992, page 48).

Recently, a number of researchers acknowledged that even though neurophysiological data suggests that the brain does not literally implement this formalism, the *metaphor* of coordinate transformation does have heuristic value (Kalaska, 1995; Soechting and Flanders, 1995; Kalaska et al., 1997).

The connectivity problem

From the neurophysiological perspective, the sensory–to–motor transformation represents (partially) a problem of identifying the anatomical ways which connect the visual and motor areas. Since the primary sensory and motor cortical areas had been identified by 1900, most researchers have assumed that a series of cortico–cortical fibers must exist to link the perceptive and motor cortex for the sensory guidance of movement (see Glickstein, 2000). As Kandel and colleagues puts it, there exists a separation between motor and sensory systems and integration needs to be done:

> Purposeful action is possible only because the parts of the brain that control movement have access to the ongoing stream of sensory information in the brain. The integrative action of the nervous system depends therefore on the interaction between motor and sensory systems (Kandel et al., 2000, page 651).

The way visuomotor coupling is attained is particularly difficult to observe in the primates brain, due to the complexity of inter–neural circuits that stand between the sensory and motor neurons:

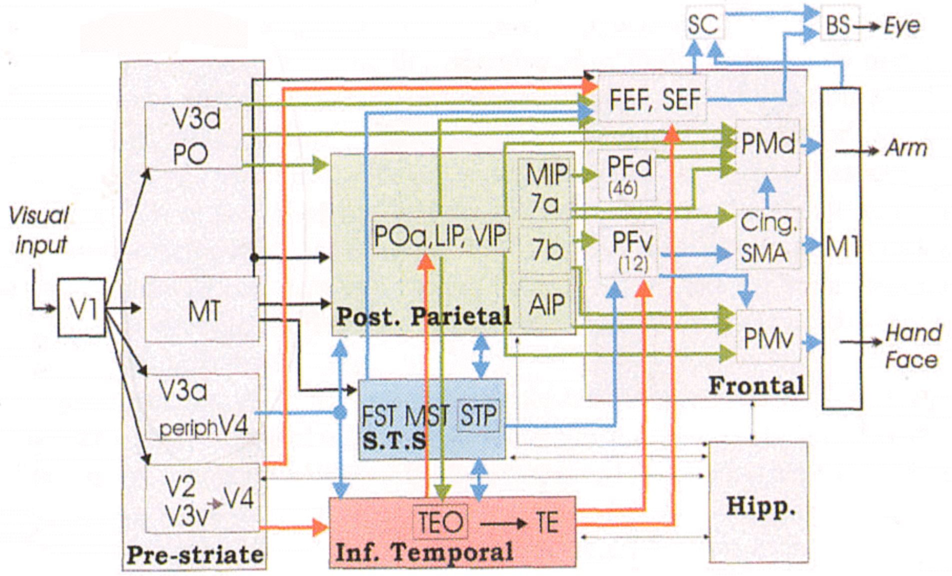


Figure 3.1: Cortical neural circuits allowing visual inputs from VI to be transformed into motor output in MI (adapted after Rossetti et al., 2000). The dorsal stream is represented in green, the ventral 'what' stream is shown in red and their possible interactions are drawn in blue. Legend: AIP: anterior parietal area; BS: brainstem; Cing: cingulate motor areas; d: dorsal; FEF: frontal eye field; Hipp: Hippocampus; LIP: lateral intraparietal area; M1: primary motor cortex; MIP: medial intraparietal area; MST: medial superior temporal area; MT: medio-temporal area; PM: premotor cortex ; SC: superior colliculus; SEF: supplementary eye field; SMA: supplementary motor area; STS: superior temporal sulcus; TE: temporal area; TEO: temporal-occipital area; v: ventral; V1: primary visual cortex; VIP: ventral parietal area.

> ...between the sensory receptors detecting signals and the motor neurons innervating the muscles are interneurons. Most of the answer to the problem of sensorimotor control resides in how these interneurons connect up to form circuits, and how these circuits are hooked up to produce behavior suited to the perceived situation (Churchland and Sejnowski, 1992, page 331).

Today, there is clear evidence that multiple-synaptic cortico-cortical pathways connect the primary visual and motor areas by way of the *dorsal stream* (Goodale and Milner, 1992; Tanne et al., 1995). Subcortical routes may also be involved in the visuomotor transformation process. An example is the route from visual to motor systems through the pons and cerebellum (Glickstein, 2000).

The dorsal pathway projecting from the primary visual cortex (V1) through the dorsal extrastriate visual area V3 and middle temporal area (V5/MT) to the superior temporal and parietal cortex is concerned with the perception of movement and spatial information (see the green stream in Figure 3.1). Within the dorsal stream there are several pathways that are responsible with the regulation of reaching movements, grasping of the objects, or movements of the eyes as they follow a moving object (Andersen et al., 2000).

The connectivity problem is illustrated in Figure 3.1, which points at the complexity of the dorsal stream cortical circuits involved in the transformation of visual inputs into motor output (figure adapted from Rossetti et al., 2000). The dorsal stream (in green) is pictured along with the ventral 'what' stream (in red) and the substrate of their interactions is also shown. Within the dissociation paradigm, the study of Rossetti and colleagues explores the possible support for interaction between the vision and action systems.

To conclude, the biological theories outlined above helped to create a paradigm of a neatly partitioned brain into areas for perception and action. This in turn, influenced the way sensorimotor mapping problem was defined and addressed (i.e., as a transformation problem). It is believed in this thesis that the description of biological visuomotor development in terms of the inverse kinematics problem and coordinate system transformation cannot be taken literally, but rather as a *metaphor* of how the motor system processes information (Kalaska, 1995). It is also believed that understanding of how basic visuomotor coupling is implemented in the brain does not necessarily involve finding the connecting pathways from the primary visual cortex to the primary motor cortex.

The progress achieved in understanding the various functions of the neural populations placed along the dorsal stream and involved in movement control has changed the way researchers think about the sensory to motor transformation (Caminiti et al., 1996; Burnod et al., 1999). For instance, while reviewing neural information processing in those brain areas involved in the generation of sensory guided movements, researchers observed that there is not such a distinct line between sensory and motor processing in the brain (Goodale, 2000; Rossetti et al., 2000). In the remainder of this chapter, we focus upon the biological and computational hypotheses brought forward by some new ideas about visuomotor coordination development.

3.1.2 New theories of visuomotor coupling

This section presents new approaches of visuomotor mapping problem from three perspectives: neurobiological, computational, and evolutionary. The first of these frameworks is build upon neurophysiological data from studies of visually guided arm–movements in monkeys. This research strand provided new insights into the organization of the cortical areas underlying reaching. Firstly, the cortical control of reaching is distributed along several areas in the parietal and frontal cortex (Johnson, 1992; Colby and Duhamel, 1996). Secondly, recruitment of neural populations from these areas for computation of motor commands involves both serial and parallel mechanisms (Caminiti et al., 1996; MacKay, 1996). Thirdly, the parieto–frontal network has a gradient architecture which favors the link of sensory and motor signals into a common hybrid reference frame (Johnson et al., 1996; Caminiti et al., 1998). Fourthly, the common frame is the eye–centered representation used in both ongoing and intended arm and eye movements (Andersen et al., 1997). Fifthly, neural cells from the parietal and frontal areas possess combinatorial properties by which they integrate information coming from various sources (Wise et al., 1997; Caminiti et al., 1998).

These findings have led to the reconsideration of the nature of visuo–to–motor transformation:

- The computational demand for reaching is met by operations that align distributed sensory and motor representations through unsupervised means. The nonlinear mapping is performed gradually, sustained by the combinatorial properties of individual neurons and the gradient architecture of the parieto–frontal network (see Section 3.2).

Another evolving research stream focuses on the use of computational modeling in an attempt to understand the mechanisms of adaptive behavior in autonomous agents (Arbib, 1987; Cliff, 1990). Research carried out on understanding simpler animals behavior indicated that all sensorimotor systems interact with their immediate surroundings by forming a closed loop with the environment (Cliff, 1995). It was suggested that:

- Sensorimotor pathways generating adaptive behavior might not be so precisely clustered into representation–transforming modules. Hence, one might try to model and understand *entire* sensorimotor pathways that are complete sequences of neural processing from the sensory input to the motor behavior (presented in Section 3.1.2).

Finally, whilst studying the interplay between eye and hand movement during reaching or manipulation of objects, researchers have realized that an essential collaboration may exist between oculomotor and limb motor control systems (Engel et al., 2000; Soechting et al., 2001). These behavioral and electrophysiological findings on eye–hand coordination suggested the hypothesis:

- Hand and eye movements are subject to similar control mechanisms and gaze provides the signal for targeted limb motion (presented in Section 3.1.2).

In a summary, the developing ideas outlined above promise to advance our knowledge of the biological and computational mechanisms of visuomotor transformation. They are rapidly maturing and we may soon witness the emergence of a new paradigm in biological visually guided motor control. The work presented in this thesis was partially motivated by the attempt to bring modeling support to these somewhat radical proposals. In the following, the latter two theories will be briefly described in subsections below, leaving the neurobiological mechanisms to be described in a dedicated section (Section 3.2).

The unitary nature of the sensorimotor cycle

One might assume, that being ”probably the most basic operation a nervous system evolved to solve” (Churchland and Sejnowski, 1992) the sensorimotor coupling is implemented by some basic, primitive mechanisms on which sophisticated structures, like a primates brain, have built sensory guidance of movement (Goodale, 1996; see also the modeling of the entire sensorimotor pathways in computational neuroethology, Cliff, 1995).

The concept of a unitary sensorimotor cycle as a motor primitive for the generation of adaptive behavior in animals (and humans) is not recent. For a long time in biology, *the reflex arc* was assumed to play a central role in the production of complex sequences of movements, which were understood to be formed by the combination of simple reflexes (Kandel et al., 2000). More recently, computational neuroscience scholars in the search of primitives for sensorimotor development have studied and emphasized the importance of primitive and postural reflexes for infant development of motor control (Kuniyoshi and Berthouze, 1998; Metta et al., 1999). Metta and colleagues (1999) studied the issue of sensorimotor development within an artificial system and suggested that simple initial behaviors, such as motor reflexes and sensory-triggered motions, can be seen as computational *building blocks*. That is, they can guide the learning of more sophisticate behaviors and act as a bootstrap procedure for the whole system (see Section 3.3.1).

Further evidence for the unitary nature of sensorimotor pathways came from experimental and modeling studies of visually guided behaviors in less intelligent animals (such as amphibians and arthropods) (Arbib, 1987; Liaw and Arbib, 1993) and flies (Franceschini et al., 1992). In simpler nervous systems it is more evident that perceptive capabilities of an organism have evolved in close relation with the achievement of specific motor goals. As Goodale puts it:

> Vision did not evolve to enable organisms to perceive. It evolved to provide distal control of their movements (Goodale 2000, page 365).

Neurophysiological data from simple vertebrate systems shows that different classes of visuomotor behavior rely on independent neural circuits specialized for generating an adaptive behavior to a certain sensory stimulus. For instance, in amphibians different circuits are involved for visually guided prey–catching and visually guided obstacle avoidance. Similarly, in rodents different subcortical pathways exist for control of orientation to food and novel stimuli and for the avoidance of obstacles during locomotion (Goodale, 1996). These observations suggest once more, that in a simple vertebrate's nervous system, vision and action are not so clearly partitioned in representation–action modules.

However, the complexity of mammals' lives has demanded more flexible organization of the circuitry, than that observed in simple vertebrates. According to Goodale (1996), this has been achieved by the development in the mammals' brain of a special system for perception, and particularly by the sophistication of the dorsal stream responsible for the perception–action loop. The ancient subcortical visumotor circuits that are shared by the monkey's brain with simpler vertebrates have become modulated by the more recently evolved layer of cortical control. This layer has evolved to make more adaptive motor behavior possible. The role of the inter–neural circuits in diversifying the motor behavior and allowing more complex response patterns has been explored also by modeling studies of the visuomotor pathways in the lamprey (Churchland and Sejnowski, 1992).

The classical view, as outlined in Section 3.1.1, defines the visuomotor mapping problem as the task of finding the path between the sensory and motor systems which implements the coordinate system transformation. The lesson from neuroethology is that perception–for–action is a crucial adaptive operation, which is implemented in less sophisticated nervous systems as unitary sensorimotor pathways (circuits). In this thesis, we consider an incremental approach to system construction. From this view, we believe that the development of new specialized modules or the increased sophistication of existing circuits preserves the unity of the perception–action cycle as a built–in, inherited mechanism.

Common computational programs for eye–hand movements

Within the visuo–motor mapping problem, special attention has always been given to eye–hand coordination. That is, because of its fundamental relevance to the organization of human motor skills and to the understanding of the perception–action relationship (Caminiti et al., 1992). Within this field, much consideration has been given to the possible interactions between the systems controlling the eye and arm. For a long time, the common assumption was that eye and arm movements may have little in common due to the fact that the eye is comparatively simple and predictable mechanical object, in contrast with the complexity of the arm system (Hildreth and Hollerbach, 1985). The traditional view of eye–hand coordination was that the central nervous system uses visual information (retinal or gaze angles) to build up representations of the environment and to guide limb movement (Kandel et al., 2000).

Recently, it was shown that for eye–hand coordination during manipulatory tasks, the gaze is well ahead of the hand and the object. This suggests that gaze supports hand movement planning (Johansson et al., 2001). Furthermore, Engel et al. (2000) demonstrated a similarity in the response of eye and manual tracking to a change in the direction of target motion. As a result, they proposed that hand and eye movements are subject to common control mechanisms and that gaze (i.e., extraretinal information) provides the signal for targeted limb motion. This hypothesis is also supported by neurophysiological studies which reveal that arm movements are coded and updated in eye-centered coordinates (Andersen et al., 2000; Snyder, 2000; see also Section 3.2).

These findings gave rise to a computational hypothesis about the way gaze signal might be ’re-used’ for arm movement control (Scassellati, 1998; Reilly and Marian, 2002). It is known that, in certain species, the auditory or visual signals produce an alignment of the head to the source of the signal. As a result, the motor head map is used to generate an appropriate motor behavior (Lazzaro et al., 1993; Kolesnik and Streich, 2002). Inspired by this idea, models of sensorimotor transformation have been built to simulate the sensory guided behavior of amphibians or flies (Franceschini et al., 1992; Liaw and Arbib, 1993).

The new hypothesis is that such a process might also operate in the case of motor control in primates, and to generate the neural program for arm movement the system might make use of the eye motor program (Metta et al., 1999; Reilly and Marian, 2002). The significance of this thesis resides in its capacity to support and eventually implement the development of visually–guided reaching on the priori achieved foundation of motor programs for eye–movements (see discussion in Section 7.1.4).

3.2 Neurobiological bases of visuomotor coordination

In Section 3.1.2 we made the case that sensorimotor coupling is a fundamental operation that a neural system has to implement. While the unitary nature of the perception–action cycle is a ubiquitous feature of simple nervous systems, the sophistication of cortical circuitry in the primate's brain poses the problem of where and how perception and action are integrated. In this thesis, we address the issue of direct sensory control of action, defined as the immediate motor response evoked by the visual activity that codes the task parameters. We will try to argue that despite the sophistication of the circuitry, this process relies on a series of basic computational mechanisms. In this section, the most recent experimental data on the neurobiology of reaching movements is reviewed in search of the basic computational building–blocks of perception–action coupling.

Experimental results on arm control indicate that no cortical area is uniquely responsible for reaching (Andersen et al., 1997; Kalaska et al., 1997). The *distributed representation* of sensory and motor information and the co–existence of several levels of representation suggest that multiple cortical and subcortical sites simultaneously interact to produce an effective reaching to visual targets (Kalaska et al., 1992; MacKay, 1995; Caminiti et al., 1998). This cooperative interaction favors the idea of a more *parallel organization* of visuomotor transformation. This contrasts with the classical view of a sequential cascade of anatomically segregated neural populations, each generating a serial representation to be integrated in the next stage (see Figure 3.1, Kalaska et al., 1992).

Based on the fact that visually derived information is available in motor areas (Johnson et al., 1999) and signals coding the intention of movement have been observed in visual areas (Andersen et al., 1997), the visuomotor mapping can be realized by the simultaneous (as opposed to serial) engagement of neural populations distributed along the frontal and parietal lobes. These cell populations combine in a nonlinear way information coming from different input sources and are able to provide an output in a different coordinate system (Salinas and Abbott, 1995). In other words, visuo–to–motor mapping can be realized at multiple stages, in a widely distributed manner, and based on the activity of integrative mechanisms that function at the level of single cell or small–networks of neurons. In the following, we shall see how these mechanisms are implemented in the brain for the control of arm–reaching movements.

3.2.1 Gradient architecture of parieto–frontal network

Visually guided reaching can be achieved through a combination of different sources of information, relating to target location, gaze direction, arm position and movement direction. Signals about the location of the target on the retina, and the position and movement of the eye and the arm appear to be distributed and co–exist in many reach–related areas from the parietal and frontal cortices (Johnson, 1992). The parietal cortex is known to contribute to the formation of multimodal spatial representations in different coordinate frames, while frontal activity is related to processes involved in the selection and execution of motor actions (Kalaska et al., 1997; Flash and Sejnowski, 2001).

Recent physiological recordings in the superior parietal and the frontal lobes revealed a gradient–like distribution of neural functional properties (Johnson et al., 1996; Marconi et al., 2001). These findings describe a gradual rostro–caudal change of response properties across the premotor and primary motor cortex. This change ranges within the frontal lobe from higher–order visuospatial processes (more rostrally) to lower–order actuator–specific signals (more caudally) (Johnson et al., 1996; Shen and Alexander, 1997). The visual–to–somatic gradient in the frontal lobe is represented in the left part of the brain in Figure 3.2.

Johnson and co–workers (1996) found a symmetrical gradient for the superior parietal cortex: arm–movement and posture related activity tended to occur more rostrally, whilst signal–related activity was found more caudally in the medial intermediate parietal area (MIP) (see representation of right part of the brain in Figure 3.2). Moreover, cell populations with similar response properties in the parietal and frontal areas are interconnected by cortico–cortical projections. These links are perhaps necessary for the match of retinal-, eye- and hand-related information during visually guided reaching (Caminiti et al., 1998; Marconi et al., 2001).

Based on these findings, Burnod and colleagues (1999) proposed that cortical control of reaching is distributed over a ’network of networks’ in the parieto–frontal lobes. These networks form a visual–to–somatic gradient presented in Figure 3.2 and schematized as follows: (1) target location in retinal and eye coordinates is signaled by neurons at the caudal and intermediate levels of superior parietal cortex and in the rostral part of the frontal lobe

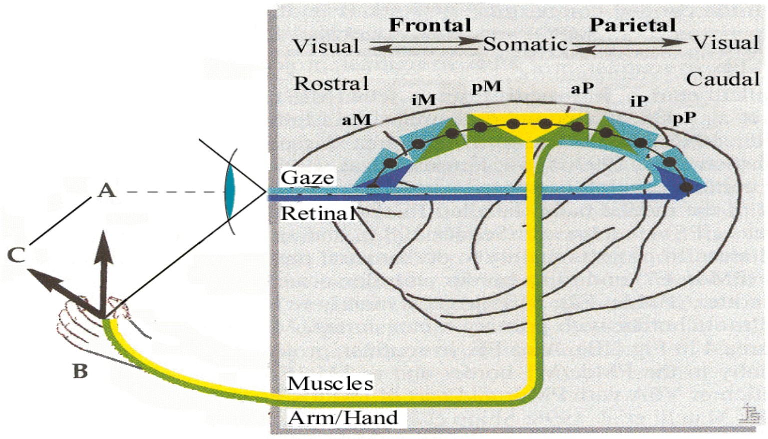


Figure 3.2: Parieto–frontal architecture with a visual–to–somatic gradient for visually guided reaching (adapted after Burnod et al., 1999). During reaching, the visuomotor transformation requires the combination of retinal (*dark blue*), gaze (*light blue*), arm / hand (*green*) and muscle (*yellow*) signals to move the hand from position B to the target in A (foveated) or in C (non-foveated). In right, cortical areas involved are represented. The authors grouped these areas approximately, in three parietal regions: anterior *aP*, intermediate *iP*, posterior *pP* and three frontal motor regions: anterior *aM*, intermediate *iM*, and posterior *pM* which are all reciprocally connected.

(see the dark blue lines and areas in figure); (2) eye–movement–related neurons (i.e., signal gaze direction) are distributed less caudally in the superior parietal cortex and less rostrally in the frontal lobe (indicated with light blue in figure); (3) neurons tuned to arm position extend more rostrally in the parietal lobe and more caudally in the frontal lobe (in green in figure); (4) arm movement-related activity predominate in the rostral bank of the central sulcus (in yellow in figure). At the border of these regions, where activities related to different signals overlap, combinatorial domains result with a role on integrating information from different sources (see Section 3.2.3).

3.2.2 Gain field mechanisms for coordinate transformation

The parieto–frontal network described above supports a gradual mapping of visual to somatic information through the involvement of neural populations coding different types of signals. In order to correctly guide motor action, the sensory signals coding retinal position and gaze direction must be combined, as noted in Section 3.1.1 by an operation similar to a coordinate system transformation.

The most accepted perspective on of how this nonlinear mapping is implemented in the brain is that it is realized in a distributed manner, with multiple coordinate systems and reference frames forming the parietal cortex for spatial representation. Mapping of visual signals to motor output is realized by the smooth transition of information from one reference frame (or combinatorial domain) to another (Salinas and Abbott, 1995; Andersen et al., 1997; Caminiti et al., 1998; Snyder, 2000).

Andersen and collaborators (1997, 2000) have provided important insights on how neural circuits may implement nonlinear functions similar to coordinate transformation. They have suggested that the transformation from retina–centered coordinates to head– or body–centered coordinates can be understood in terms of *gain fields* associated with neurons in area 7a of the posterior parietal cortex. Cells in this area receive a convergence of eye position signals and visual signals and the response of a neuron is a product of the receptive field and the linear gain field. This multiplication implements a nonlinear function that can be described as a coordinate system transformation.

The gain field mechanism represents space in a distributed format, allowing inputs from multiple sensory systems with different spatial frames and outputting signals for action in several motor coordinate frames. Several neural network models based on gain field mechanisms have been proposed for the conversion of eye and retinal position into head–centered coordinates (Zipser and Andersen, 1988; Salinas and Abbott, 1995), or retina (eye–centered) and auditory (head–centered) signals into motor error coordinates (Andersen et al., 1997).

The solution to the nonlinear mapping problem illustrated above, focuses on the formation of distributed representations in multiple coordinate frames, by using gain field modulation. Another modality to combine signals for arm reaching, for instance the target location and the limb starting position is to converge them onto single cells. These then have to perform a match of the stimulus attributes (target location) with attributes of the appropriate motor response (joint angles to reach the target) (Kalaska et al., 1997). This approach is based on neural combinatorial properties and it is presented in the section below.

3.2.3 Combinatorial properties of neurons

The combinatorial properties of neurons have been observed in many areas of the parietal and frontal lobes (for a review see Wise et al., 1997; Caminiti et al., 1998). For instance, discharge rates of neurons in premotor areas and in the 7m parietal area co-vary with the direction of both eye and arm movement (Kalaska et al., 1997). Parietal cells in V6a (dorso-medial part of the parieto–occipital area) exhibit complex interactions between visual inputs and motor functions (Battaglia-Mayer et al., 2000). Moreover, about 40% of neurons in the premotor ventral area are modulated by the direction of gaze (Mushiake et al., 1997).

Based on these findings, Burnod and coworkers (1999) attempted to explain the biology underpinning sensorimotor transformation in terms of *combinatorial domains* and *matching units*. The authors proposed that the gradient architecture of the parieto–frontal network (see Section 3.2) favors the functional overlap of the regions coding for different reach–related signals (i.e., retinal, gaze, arm position, and muscle output). Signals from different sources along the visual–to–somatic axis can be matched within three combinatorial domains (see Figure 3.2):

- The anterior parietal *aP* and posterior frontal *pM* domain combines information on *muscle dynamics* with *positional and directional arm* information (c-domain1 combines yellow and green signals in Figure 3.2).
- The intermediate parietal *iP* and frontal motor area *iM* domain relates information on *movement direction* with *gaze positional and directional* signals (c-domain2 combines green and light blue signals),
- The posterior parietal *pP* and anterior frontal motor *pM* domain relates *positional and directional information* of gaze and arm with *visual inputs on the retina* (c-domain3 combines light and dark blue signals).

Within each domain, information can be combined along two other axes: the *position–direction* axis and the *sensory–motor* axis. It is well known that directional information is encoded in the activity of neurons in all reach–related areas from the parietal and frontal lobes (Georgopoulos et al., 1993; see Section 2.1.3). Hence, on the position–direction axis, processing units are tuned to both position and direction. That is, they respond maximally to movement in a certain direction and at a certain position in space.

Finally, neurons sharing the same combinatorial domain and similar position and directional tuning properties may have different temporal relationships to the signals relevant to reaching. This results in the existence of four types of units on the sensory–motor axis: *sensory units* that are timed–locked to sensory signals in all domains; *motor units*, time–locked to motor events; *matching units*, which learn correlation between sensory and motor signals; and *condition units*, which store correlation between sensorimotor signals and reinforcement contingencies.

The progressive match framework for visuomotor mapping

In our view, one of the best-articulated proposals of unsupervised visuomotor learning for arm-reaching was described in Burnod and co–workers (1999). This model exploits the gradient nature of the parieto–frontal architecture together with the combinatorial properties of neurons. In particular, it proposes that sensorimotor mapping is accomplished in a progressive manner, by the gradual involvement of sets of matching units belonging to different combinatorial domains. Synaptic learning results from the increase of the connection strength between two units, due to their repeated co–activation (e.g., Hebbian learning).

Below, we describe what in their view, are the first four stages required for the alignment of hand direction (in motor coordinates) toward the target position (in visual coordinates). Each stage corresponds to learning in different sets of matching units (for the description of the complete scenario see Burnod et al., 1999):

- *Motor babbling*. When the monkey moves the hand, co–activations in the first combinatorial domain (c-domain 1, see Section 3.2.3 above) due to inputs from motor commands and re–afferent somato-motor signals result in reinforced connections. After learning, these will allow the matching units to generate adequate muscles commands in order to produce an expected sensory effect.
- *Control of gaze and attention*. When the gaze shifts toward a stimulus, co–activations in the second combinatorial domain (c-domain 2) due to gaze movement and retinal input reinforce connections between neurons coding these signals. This learning enables the matching units to shift gaze to any focus of attention.
- *Hand tracking*. When the eyes look at the moving hand, co–activations in the third combinatorial domain (c-domain 3), due to inputs signaling gaze movement and hand

movement, result in reinforced connections that relate equivalent hand and gaze pathways. This set of matching units learns to perform visual tracking of hand movement.

- *Reaching to foveated targets*. When the hand moves toward the foveated target, co–activation of previously reinforced hand–gaze and gaze–retinal connections leads to learning in the connections of matching units. After training, these units will be able to serve reaching to foveated targets, even if the hand is not in the field of view.

Note that within the parieto–frontal network, gaze and arm directional and position signals play a unifying role. First, a subset of processing units is pre–selected based on gaze and arm position and these units then learn stable relationships between visually derived signals and somatomotor signals.

In summary, recent work on the coding of reaching movements in the brain has significantly advanced the knowledge of sensorimotor transformation on several fronts. Experimental evidence points to the fact that cortical control of reaching is distributed along a parieto–frontal network and it involves serial and parallel recruitment of neural populations distributed along a visual–to–somatic gradient. Coding of both arm and eye movements in an eye—centered reference frame, may represent the 'missing link' in understanding this visuomotor mapping.

Burnod and co–workers (1999) integrative framework was described in more detail, with the belief that it currently represents the most coherent proposal of how computational processes involved in reaching can be specified at the biological level. Up to the present, the simulation of a complete neural network to perform the neural operations of matching units has not been solved yet. In this thesis, we bring computational evidence from simulations with realistic spiking neural models for the learning mechanisms involved in the progressive match framework. In the remainder of this chapter, we focus upon discussing biologically inspired neural network models of visuomotor coordination.

3.3 Biologically inspired models of visuomotor mapping

At present, after several decades of applied research, the adaptive capabilities of motion control on biological organisms are still vastly superior to the capabilities of robotic systems. Evident sources of inspiration for the creation of more intelligent robots are the real nervous systems. In the recent years, numerous neural network models have been developed to apply biologically inspired control mechanisms to various robot control tasks (Bullock et al., 1995; Zeller et al., 1995). Accordingly, several recent proposals have been made for a more physiologically inspired modeling of visuomotor coordination development (Burnod et al., 1992; Salinas and Abbott, 1995). Efforts on this direction have been aimed at the construction of neural architectures that are completely founded upon anatomical evidence. This means, to use neural elements that correspond as closely as possible to known neural cell types and to apply biologically plausible learning (Bullock et al., 1995).

This research direction reflects a change in emphasis away from the more traditional approaches that model visuomotor development based on control theory formalisms (Massone, 1995; Jordan, 1996; Jordan and Wolpert, 2000). That is, it explores alternate learning principles to the error-correction mechanisms (see Mohl, 1993). The attention is oriented towards investigating the efficacy of the neocortex organizational principles, when they are applied to arm-reaching tasks. These principles are of an unsupervised nature and are the result of correlation–based associations (e.g., Hebbian learning).

At first, it was suggested that nonlinear sensorimotor transformations could be modeled as sets of linearized representations, and so become a simple linear problem (Bullock et al., 1993). Furthermore, Baraduc and Guigon (2002) demonstrated using rigorous mathematical apparatus, that linear transformations can be learned by Hebbian associations, so long as the training examples satisfy a regularity condition. For cases where the examples do not satisfy the regularity condition (e.g., non–uniform distribution of preferred attributes of neurons in a population) learning the transformation is possible only with a stochastic error–correcting learning rule (see discussion in Section 3.3.2).

This section first reviews the main developmental paradigm for learning visuomotor coordination. Then we briefly discuss the relation between self–organizing and error–correcting mechanisms and finally, we focus upon our main interest that of physiologically inspired models driven by means of unsupervised learning.

3.3.1 The developmental paradigm

A large number of models that learn inverse kinematics through experience use a *circular reaction learning* protocol. The circular reaction concept was introduced by Piaget (1969) and describes the following behavioral loop: as a child performs random, spontaneously generated movements of his arm, his eyes follow the arm's motion, thereby enabling the learning of a transformation from a visual representation of arm position to a motor representation of the same arm position. Inspired by this developmental process, researchers in robotics have implemented a similar adaptive process for learning inverse kinematics. This process is based on an autonomously controlled behavioral cycle with two components: production and perception, and is referred to as *motor babbling* (Kuperstein, 1988; Ritter et al., 1992).

During a motor babbling stage, the control system endogenously generates random movements, which displace the arm to different postures, bringing the end–effector into view. For each configuration of the arm, the vision system locates the end–effector and correlations between joint variables and eye variables are learned. After the correlations are stored in connection weights, the visual input passes through the weight network to generate a predicted arm activation vector. The difference between the predicted and externally generated activation vectors is used to modify the weights network and after many babbles the model will learn to accurately reach to the foveated objects in the workspace (Kuperstein, 1988; Gaudiano and Grossberg, 1991).

In the last decade, it has been pointed out that an algorithm based exclusively on random movement generation–and–observation for learning the inverse kinematics has a number of drawbacks (Jordan, 1996; see Section 3.3.2 below). Despite this limitation, the *action–perception* cycle remains the *de facto* behavioral framework for building developmental models of eye–hand coordination.

The *circular reaction learning* is not the only concept that computational researchers have borrowed from developmental psychology in an attempt to create artificial systems capable of adaptive, on–line control of goal directed reaching. Kuniyoshi and coworkers proposed a developmental framework based on the concepts of *assimilation* and *accommodation*, *imitation* and *entrainment dynamics* (Kuniyoshi and Berthouze, 1998; Kuniyoshi et al., 1999; see also Smith and Thelen, 1993). The implementation of these notions allows an artificial system to bootstrap itself towards a higher complexity through embodied interaction dynamics with the environment. Acquisition of eye–hand coordination and cross–modality mapping are attributed to the emergence of ordered structure from: interaction between many sensory-motor processes, embodiment, and basic pre–existing behavior patterns (i.e., reflexes) (Berthouze and Kuniyoshi, 1998; Kuniyoshi and Berthouze, 1998).

The importance of a pre–existing repertoire of motor primitives such as primitive reflexes, to support the development of coordinated movement has been also emphasized by Metta and co–workers (1999). The authors proposed a developmentally plausible solution for the emergence of eye–hand coordination. Instead of computing the kinematics required by the transformation of information in 3D visual coordinates to hand coordinates, they implemented a direct mapping between the eye–head motor plant and the arm motor plant. In this approach, the eye–head movement is controlled using force fields. Consequently, the position of the target point in space can be coded using motor commands, namely the commands that control the position of the head and that of the eye with respect to the head. This allows the arm's force–fields to be obtained through a *motor–to–motor* mapping, from eye–head force fields. On the human infants, this mapping is assembled through experience and it is possible thanks to the built–in motor reflexes and the sensory–triggered motions that provide an effective means for linking vision and proprioception.

3.3.2 Self–organizing and error–based mechanisms for direction mapping

The direct inverse modeling approach outlined in the previous section has a number of limitations and cannot properly control nonlinear redundant objects (Jordan, 1996). More effective algorithms have been proposed, which solve the inverse kinematics problem in two phases. During an initial *learning by doing* period, the forward kinematics from joint angles to arm posture are learned. In a second stage, the inverse kinematics from desired trajectory to joint–angle map are learned, by using error–correction mechanisms (Mel, 1991; Jordan, 1996).

The error–correction algorithms represent the most common approach in modeling sensory guided reaching. A more biologically plausible alternative is to use unsupervised learning based on Hebbian associations. It was pointed out that correlation–based association mechanisms can assure the alignment of sensorimotor maps for the correct transfer of information for reaching (Baraduc and Guigon, 2002). We illustrate here the application of these learning rules to the direction–mapping problem.

Direction–mapping learning is an effective approach for translating spatial trajectory to end–effector directions, which has received increased attention in recent years amongst researchers working on modeling of sensorimotor coordination (Bullock et al., 1993; Fiala, 1995; see also Section 2.3.2). Perhaps the most illustrative example of learning direction mapping for visually guided movements is the self–organizing DIRECT model proposed by Bullock and colleagues (1993).

The central operation in the DIRECT system is the mapping of spatial directions in body–

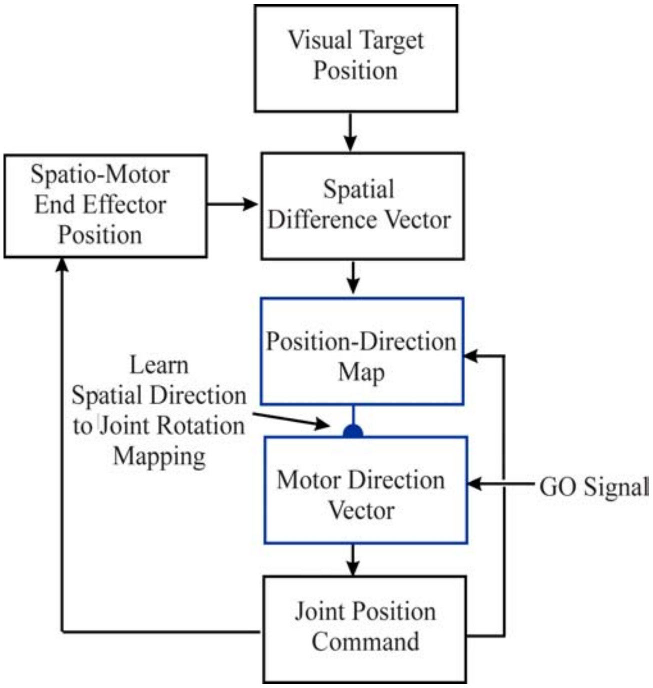


Figure 3.3: Processing stages of the self–organizing DIRECT model for control of movement (adapted after Bullock et al., 1993). Learning of the spatial direction to joint rotation mapping occurs during an action–perception cycle triggered by the GO signal, and is based on an improved Hebbian-like learning rule. See in text for details.

centered coordinates to joint directions in muscle–coordinates. This is accomplished in a number of processing stages as illustrated in Figure 3.3. First, the current position of the end effector is determined and the desired movement direction is computed as the difference between the visual representation of the target position and the actual representation of the end effector (first two stages in Figure 3.3). Second, the actual motor position signal is combined with the spatial direction vector using a self–organizing network. Learning at this stage results in the formation of a map where each cell is sensitive to a particular spatial direction in a particular position of joint space (the position–direction map in Figure 3.3).

Learning of spatial–to–motor direction mapping takes place in the third stage of the model (see the modifiable blue synapses in Figure 3.3). Here, translation of position–direction information into joint-directions is learned during a motor babbling period, when the GO signal generates random movements. The adaptation of the plastic synapses is based on a variant of normalized Hebbian learning mechanism, known as the outstar rule (Grossberg, 1998). After learning, the model is capable of visual control of reaching with tools and clamped joints, in the presence of visual input distortions or in the absence of visual signals (i.e., blind reaching).

Learning of the differential transformation between the motor and spatial directions can also be implemented by using a gradient descent algorithm (Ritter et al., 1989). Fiala (1995) proposed an error–correction based algorithm that combines the DIRECT and VITE models (Bullock and Grossberg, 1989) in order to obtain bell–shaped velocity profiles of trajectories. In this architecture, the spatial directional vector is mapped to a motor direction vector through an intervening field of cells, referred to as direction mapping cells. The difference in activity between the spatial and motor direction vectors is computed and minimized with a gradient descent algorithm. Feedback information from the motor output layer is also taken into account during learning.

The model learns a kind of inverse Jacobian in a manner similar to that obtained by Ritter et al. (1989) and succeeds in reproducing straight–line motions and unimodal, bell–shaped velocity profiles. Based on the model's good performance, Fiala (1995) has tried to motivate the use of error–correction learning mechanisms with behavioral and biological evidence. Thus, behavioral results on spiraling–in movements during reaching suggest that the visual error between the hand and target is computed during movement (Roby-Brami and Burnod, 1995). However, these experimental findings also indicate that movement usually begins as a straight line along the transformed direction and a spiraling movement is often observed during the later part of the reaching. This observation suggests the existence of different stages in movement control, which, accordingly, may rely on different mechanisms.

According to Doya (1999), Hebbian synaptic mechanisms characterize the self–organization of the cerebral cortex, while error–correction learning is implemented in the cerebellum. Given the separate localization of these adaptation mechanisms a plausible hypothesis is that learning of movement control implies synaptic changes driven by different rules, in both structures. At the cortical level, correlation–based associations may be involved in the formation of stable connections for the alignment of visual and motor neural representations. This process allows the correct transfer of information to initiate the movement, by specifying the directional information. The involvement of error–correction mechanisms through the cerebellum may be necessary for the adjustment of the trajectory during the later stages of movement, to obtain an adequate reaching of the target (see also Fiala, 1995; Doya, 1999).

To conclude, there is currently a biologically motivated research direction in modeling of arm movement control. Much attention within this direction is given to the use of self–

organizing algorithms for learning of inverse kinematics. It has been argued that Hebbian association mechanisms can learn linear transformations and they can implement synaptic adaptation for development of visuomotor mapping. Furthermore, despite the limitations of direct inverse modeling, there is a growing research stream which builds–upon the developmental approach of visuomotor acquisition. In this thesis, modeling work of visuomotor mapping is based exclusively on unsupervised learning and it is placed within the developmental paradigm.

3.3.3 Models inspired by physiological data

A significant part of this chapter has been devoted to the review of recent experimental data on neurobiology of visuomotor transformation (Section 3.2). In comparison, the survey of biologically inspired modeling of sensorimotor development is less extensive. This is because, while much detail regarding the manner in which the brain solves the visuomotor transformation has been accumulated, the implementation of these concepts within artificial systems has only just begun. We believe that most importantly, the current progress towards the understanding of arm reaching neurobiology has given rise to a conceptual framework that explicates the neural basis of computation, allowing its readily implementation (see Section 3.2.3 for instance).

The advantages of constructing artificial systems inspired by real living systems are clear (Bullock et al., 1995; Cliff, 1995). Note that we are referring to an implementation fully grounded on physiological data, with architecture, neural models and learning rules all designed to be biologically plausible. We believe that an important step forward in this endeavor has been represented by the proposal of Burnod and colleagues (1999) (see Section 3.2.3). Even if the implementation of this model is not yet complete, it represents the result of a series of precursor models, which explore the alignment of hand, gaze and visual signal representations by unsupervised means. We review this previous work here.

Salinas and Abbott models

The biological modeling of the visuomotor transformation has been significantly advanced by the studies of Salinas and Abbott (1995, 1996). Their work addresses two major problems. The first problem considers how to find the mechanisms that ensure the alignment of neural representations, in such a way that sensory activity representing target location evokes an appropriate motor response toward the target (the *alignment problem*). The second issue addresses the *coordinate transformation* problem. That is, if the target location in body–coordinates is determined by a combination of retinal position and gaze direction, then what set of weight connections assure that the correct combination is transferred to the motor network?

Their solution to the alignment problem consists of an unsupervised mechanism that transfers information between two networks composed of broadly selective neurons. Sensory neurons can encode retinal position solely, or retinal and gaze direction information, while motor neurons are assigned with preferred directions of movement. Both motor and sensory neurons firing rates are characterized by maximal cosine tuning curves (see Section 2.1.3 on direction coding).

The networks are coupled through modifiable synapses and learning occurs during the observation of the random movements generated by the motor network (e.g., motor babbling phase). The training procedure is based on a general Hebbian learning mechanism that modifies the synaptic weight w_{ij} between a j sensory neuron and an i motor neuron according with the correlation between their firing rates R_i^m and R_j^s:

$$w_{ij} =< R_j^s R_i^m > -k \tag{3.1}$$

where k is an arbitrary constant set to optimize the performance of the system. This process leads to an accurate alignment of the visual and motor representations, which ensures that sensory activity coding the target position is correctly used to guide the movement generated in the motor map. The authors derive a condition for the network alignment to occur. The condition is that the strength of the connection between a sensory neuron j and a motor neuron i depends on the magnitude of the difference between the neurons preferred locations.

Furthermore, the authors address the problem of linear (and nonlinear) mapping of the target location in retinal coordinates into movement direction in head–centered coordinates. It is shown that a sensory–motor network with synaptic weights that depends on the difference between the preferred direction of the motor neuron and the sum of the preferred retinal position and gaze direction for the sensory neuron in a pair can provide a general mechanism for linear coordinate transformation. The nonlinear coordinate transformation problem was further implemented with computational mechanisms at the population level, which are shown to realize neural multiplication (Salinas and Abbott, 1996).

A more general computational framework, which deals in detail with the distributed representation of linear transformations and discusses the limitations of Hebbian mechanisms when compared to supervised means, in learning of nonlinear coordinate transformations can be found in Baraduc and Guigon (2002).

Burnod and colleagues work

A significant contribution to the understanding of computational mechanisms involved in the control of arm–reaching was brought by the work of Burnod and co–workers during the last decade. In comparison with Salinas and Abbott mathematical model, those of Burnod and colleagues are more tightly linked to the physiological and anatomical aspects of sensorimotor learning. Models developed by this research group incorporate somatic information beside the visually derived signals, in order to compute the appropriate motor command for reaching out to a visual target (Burnod et al., 1992; Baraduc et al., 1999).

A precursor to matching units' concept and neurons combinatorial properties was first described by Burnod et al. (1992). The primary goal of this study was to examine how the central nervous system might learn to reach towards a target position from any initial arm posture. The proposed architecture has three layers: (1) a map of *matching* units, which combine visual directional input and somatic arm signals and project to the next layer of synergy units; (2) a laterally connected map of *synergy* neurons which integrate the input from the matching layer and periphery feedback and which projects to a third (3) layer of *motor output* units that displace the hand in space. The appropriate combination of the visual information about movement trajectory with the kinesthetic information concerning the orientation of the arm in space is learned by the network from spontaneous movements.

Learning of the inverse kinematics from hand space to actuator space takes place in the matching units layer, according to correlation–based adaptation rules applied to sensory and motor co–activated events. The neural response in each layer is given by a combination of the inputs coming from the afferent source and the lateral signals from other units in the layer. The computation learned by the units can be approximated by a bilinear operation. This is interpreted by the authors as a projection of the visual information on a motor reference frame, that simultaneously rotates with the arm (Burnod et al., 1992). Once the network is trained, the signal on the direction of intended hand movement is projected onto the matching units and activates the appropriate synergy units as a function of the starting arm posture.

A simpler, but more readily implementable variant of the above model was proposed recently by Baraduc et al. (1999). They consider a neural network model which combines broadly tuned (muscular) proprioceptive and visually derived directional input to calculate the angular motor commands for the initial part of the movement of a two–link arm. The arm position is represented by a set of 20 proprioceptive neurons coding muscle length. Direction of the desired movement is coded by a population of 50 visual neurons with cosine tuning functions. Motor commands are specified by a population of 50 motor neurons with optimal tuning property, which command motor synergies (direction in joint space). Compared to Burnod et al. (1992) model, this architecture is simplified by considering only two layers and learning only at one level. The advantage is that, in this case, the motor synergy layer encodes information directly in joint coordinates.

Information concerning the position of the arm and the visual desired direction is combined in two steps. First, an intermediate representation of arm position is formed in a somatic layer, by the combination of feed–forward proprioceptive signals and lateral intralayer feedback. Second, activity in the somatic layer is combined with the visual directional activity. Only weights between the proprioceptive input layer and the somatic map are subject of learning. Training is performed by motor babbling in 5 positions and takes place according to a variant of the delta rule (i.e., supervised learning). After learning, the network generates arm commands (in joint coordinates) which move in the same direction as the visual input, over a large part of the visual inputs (16 directions of movements were tested).

Compared to previous models this approach has several advantages. In contrast to the DIRECT model (Bullock et al., 1993), Baraduc and co–workers use broadly tuned motor neurons for learning of visuomotor mapping of direction. Compared with Salinas and Abbott approach, generation of movement is modeled with respect to the arm position (proprioceptive feedback). However, the somatic map organization produces a motor population vector, which largely deviates from the desired direction of movement. We believe this result might be improved by separating the motor map organization from learning the direction mapping task. A further discussion of these models can be found in the final chapter, along with an evaluation of the results of our work.

Part II

SpikeNNS: a simulator for pulsed neural networks

Chapter 4

Spiking neural model

In the previous decade, the majority of artificial neural network models were based on a computational paradigm involving the propagation of continuous variables from one unit to another (Rumelhart and McClelland, 1996). However, in recent years experimental evidence has been accumulating to suggest that biological neural networks, which communicate through spikes, use the timing of these spikes to encode and compute information (Abeles, 1991; Thorpe et al., 1996; Rieke et al., 1997). Several alternative schemes to the rate–coding hypothesis have been proposed, suggesting that information may be coded in synchronization and oscillations of populations of neurons, or in the precise timing of the neural pulses (Recce, 1999; Gerstner and Kistler, 2002).

A new generation of *pulsed* neural networks has emerged, which focuses upon the mathematical formalization of the computational properties of biological neurons (Gerstner, 1995; Maass, 1997; Stevens and Zador, 1998). The neural models created capture the spiking nature of the neurons and retain the essentials of the behavior to be modeled, while trying to simplify the description (Gerstner, 1999; Izhikevich, 2001). This chapter focuses upon the description of a simplified spiking neural model and of the type of computations that it can account for.

In order to understand how a biologically inspired neural model behaves it is necessary to offer a brief introduction in the physiology of the real cell (Section 1). Neural activity of biological cells may be described at several levels of abstraction. In Section 2, the main tendencies in neural modeling, i.e., detailed vs. simplified, are reviewed, outlining the advantages and drawbacks of each direction. This description is followed by the presentation

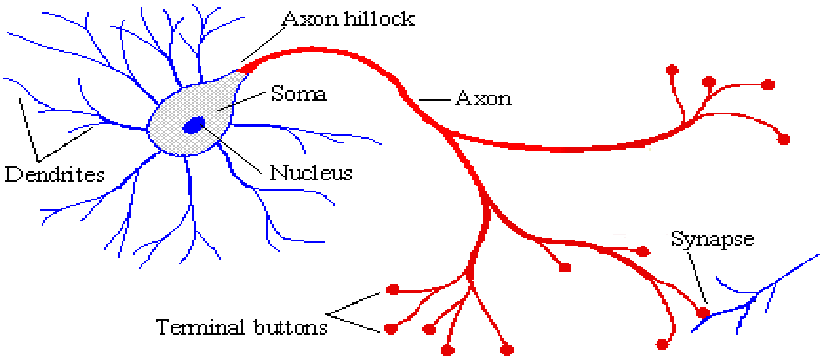


Figure 4.1: Schematic of a biological neuron. See in text for the description of the functionality of each component.

of a number of time-coding schemes and computation modes that are implemented in our simulator. In Section 3 the neural model implemented in SpikeNNS is described (e.g., equations, parameters, input patterns). The last section offers a brief review of a number of learning rules for spiking neurons.

4.1 Biological neuron physiology

The brain is a collection of about 10 billion interconnected neurons. Each neuron is a cell that uses biochemical reactions to receive, process and transmit information. The neural unit represents the basic information processing mechanism that is the foundation of human cognition.

4.1.1 Neural components

A typical neuron can be divided into three functionally distinct parts, namely the *dendrites*, the *soma*, and the *axon* (see Figure 4.1). A neuron receives connections from thousands other neurons. Most of these contacts take place on the neuron dendritic tree, however they can also exist on the soma or the axon of the neuron. The morphology of the dendritic tree plays an important role in the integration of the synaptic inputs and it influences the way the neuron processes information and computes (Mel, 1994). The strengths of the charges received by a neuron on its dendrites are added together through a nonlinear process of spatial and temporal summation (Koch, 1999). The resulting input flows to the soma and suffers voltage attenuation, so that about half of the total charge injected into the distal dendritic site reaches the soma.

Two parameters play a critical role in the spatio–temporal integration of synaptic inputs: the *space constant* λ that gives the voltage attenuation with distance and the *membrane time constant* τ_m, which describes how fast the voltage decays. The input transmission from a dendritic site to the soma is characterized by a *propagation delay*. In the case of slowly varying inputs this is dictated by the membrane time constant value, but for fast synaptic inputs it outpaces τ_m, giving the possibility of submilisecond coincidence detection in dendrites (Koch, 1999).

The primary function of the soma is to perform the continuous maintenance required to keep the neuron functional (Kandel et al., 2000). The part of the soma that performs the important non–linear processing of the input is the *axon hillock*. If the total input produces a depolarization up to the neural *threshold* (i.e., of about -55 mV), the axon hillock fires an action potential. The output signal is transmitted down the axon, which delivers it to other neurons.

4.1.2 The action potential

The firing of a neuron, referred to as the *action potential*, is an all or none response. This means that, incoming stimuli either produce action potentials, if they exceed the neuron's threshold value, or they do not. A spike or action potential is a stereotyped impulse of fixed magnitude generated by the neuron. After the firing of an action potential, the neuron enters a *refractory period* when no further action potentials can be generated. Even with very strong input, it is impossible to excite a second spike during or immediately after a first one. This causes that action potentials in a spike train are usually well separated. The minimal distance between two spikes defines the *absolute refractory period* of the neuron. The absolute refractory period is followed by a phase of *relative refractoriness* where it is difficult, but not impossible, to generate an action potential (Kandel et al., 2000; Gerstner and Kistler, 2002).

4.1.3 Synapses

The site where the axon of a presynaptic neuron makes contact with the dendrite (or the soma) of a postsynaptic cell is the *synapse*. The most common type of synapse in the vertebrate brain is a chemical synapse (Shepherd and Koch, 1990). When an action potential arrives at a synapse, it triggers a complex set of biochemical processes that lead to a release of neurotransmitters from the presynaptic terminal into the synaptic gap. The voltage response of the postsynaptic neuron to a presynaptic action potential is referred to as the *postsynaptic potential*. A single synaptic input is rarely sufficient to determine the generation of an action potential. This is usually triggered as a consequence of the nonlinear interaction of several excitatory and inhibitory synaptic inputs (Mel, 1994; Poirazi et al., in press). At the end of the synaptic action, the postsynaptic potential decays exponentially towards the resting value with a rate of decay given by the time constant τ_m. The physical and neurochemical characteristics of each synapse determine the strength and polarity of the new input signal. Synapses are believed to be the locus of learning and memory in the brain (Squire and Kandel, 1999). This is where the brain is the most flexible and the most vulnerable.

4.2 Computational and modeling aspects

The idea that brains are computational in nature has captured the attention of many researchers in the last decade (Churchland and Sejnowski, 1992; Maass, 1995; Koch, 1999; O'Reilly and Munakata, 2000). In a strict sense, a physical system computes a particular function if there is an appropriate mapping between the physical states of the systems and the elements of the function and if the mapping of the input space into the output space performed by the function can be described in terms of some rule (McCarthy, 1963; Churchland and Sejnowksi, 1992). For instance, one can describe the stochastic behavior of a neuron that maps an activation value V to a binary response that is a *spike* in 25% of the cases and *no spike* in 75% cases with a probability distribution function: $f : V -> (g : \{spike, no\,spike\} -> [0, 1])$ (after Touretzky, 2001).

The studies concerned with the way the brain represents and computes information based on the activity of neural components and using as main methodology the computer simulations, are unified under the computational neuroscience discipline (see the introduction to the field in Section 1.1). This research stream differs from other approaches of neural modeling in that it believes that understanding the way the brain computes is very closely dependent on the knowledge of the anatomical and physiological details of neural elements (Bower and Beeman, 1998). Hence, we can talk about a general tendency in computational neuroscience towards realistic simulation of the structure and physiology of the biological neurons.

4.2.1 Levels of detail in neural modeling

Neural activity may be described at several levels of abstraction. We can distinguish between two major modeling directions in computational neuroscience, depending on how much of the neuron details the modeler wants to take into account. That is, because the more details are taken into account, the greater are the computational demands of the model.

Detailed neural models

One trend in computational neuroscience is oriented towards creating ever more detailed and complex models (De Schutter and Bower, 1994 a,b; Bower and Beeman, 1998; Segev and Burke, 1998; Steuber and De Schutter, in press). The detailed modeling of the neurons aims to describe the chemical processes undergone at the subcellular level, including the biophysics of ionic channels and dendritic trees, the synaptic interactions between excitation and inhibition, the voltage-dependent events in the active dendrites (Destexhe et al., 1998; Mainen and Sejnowski, 1998). In this effort, the neurobiologists argue that one has to consider the anatomical and physiological details of the neurons if he / she pursues a full understanding of the nervous system (Bower and Beeman, 1998).

The model proposed by Hodgkin and Huxley in 1952 offers perhaps the clearest instance of a detailed model of the processes involved in action potential generation. The model provides a detailed description of the biophysics of ionic mechanisms underlying the initiation and propagation of the neural spike. By doing this, it offers an accurate *quantitative* model of the physiological data. However, complex frameworks like the Hudgkin-Huxley model, which account for numerous ions channels and different types of synapses are difficult to construct and to analyze. An important conceptual drawback of this family of models is that their numerical complexity (e.g., solve a large number of nonlinear differential equations) can prevent one from understanding which features are responsible for a particular phenomenon and which are irrelevant (Koch, 1998).

Hodgkin-Huxley-like models can be applied to point neurons, that is, neurons without any spatial structure. However, the morphology and architecture of the dendritic tree plays an important role on the nonlinear functioning of the neurons (Mel, 1994; Vetter et al., 2001). For the construction of detailed neuronal models that consider all of the cells complexities, including the branched cable structure, the standard approach is to divide the neuron into a finite number of isopotential connected compartments (Rall and Agmon-Snir, 1998; Segev and Burke, 1998). Each compartment acts like a capacitance-resistance circuit, modeled by a system of ordinary differential equations. It can represent somatic, dendritic or axonal membrane and contain a variety of synaptic inputs (Segev and Burke, 1998).

Compartmental models are implemented by powerful, biologically realistic simulators, such as Genesis (Bower and Beeman, 1998) and Neuron (Hines and Carnevale, 1998). Nevertheless, there is an important computational cost in memory and speed that detailed simulations based on compartmental models must pay. For instance, modeling of calcium dynamics in a *single* Purkinje cell, using 4550 compartments and 8021 ionic channels required 8 Sun workstations running one hour for the simulation of 550 ms of Purkinje cell activity (De Schutter and Bower, 1994a, b).

A major argument in favor of detailed modeling is that the close replication of the nervous system structure as a basis of exploring its functions increases the chances to discover its unknown (or unsuspected) organizational principles (Bower, 1997). The drawbacks arrive from the complexity of these models. That is, they are difficult to construct and analyze; they scale poorly with the network size and activity; they represent a quantitative, rather than a qualitative description of the neural behavior, which can prevent one from understanding the crucial features of the system. Despite the recent efforts oriented for the creation of simulation frameworks that allow modeling of large-scale networks of biologically realistic neurons (see Parallel GENESIS, Bower and Beeman, 1998; Goddard et al., 2001), the detailed models are generally considered more suitable for the study of single neurons or small networks behavior.

Formal spiking neuron models

The second stream of research in computational neuroscience is oriented towards modeling the spiking nature of the neurons and retaining the essential elements of the behavior being modeled, while trying to simplify the complexity of the resulting description (Gerstner, 1991; Maass, 1995; Maass, 1997; Rieke et al., 1997). The principal motivation for the creation of simplified models is that they allow studying more easily the computational and functional principles of neural systems (Koch, 1999).

The reduction of the detailed neuron models to formal models requires simplifications in at least two respects. First, the non–linear dynamics of spike generation must be reduced to a single ordinary differential equation and second, the spatial structure of the neuron (i.e., the dendritic tree) is neglected and reduced to an input (Gerstner and Kistler, 2002). To support the validity of the former simplification, Kistler et al. (1997) demonstrated that spike generation in the Hodgkin–Huxley model can be reproduced to a high degree of accuracy (i.e., up to 90%) by a single variable model. The authors pointed out that the Hodgkin–Huxley model shows a sharp, threshold–like transition between an action potential for a strong stimulus and graded response (no spike) for slightly weaker stimuli. This suggests that the emission of an action potential can be described by a threshold process (see also Section 4.3.1).

Several simplified neural models have been proposed in the last decades. The *leaky integrate–and–fire* neuron is probably the best–known example of a formal neural model (Tuckwell, 1988; Bugmann, 1991; Stevens and Zador, 1998). It simulates the dynamics of the neuron membrane potential in response to a synaptic current by implementing an equivalent electrical circuit. The function of the integrate–and–fire circuit is to accumulate the input currents and, when the membrane potential reaches the threshold value, to generate a spike. Immediately after emitting a pulse, the potential is reset and maintained there for an absolute refractory period.

The simplified mathematical models for spiking neurons cannot account for the entire range of computational functions of the biological neuron. Rather, they try to abstract a number of essential computational aspects of the real cell function. The essential features implemented can differ between models, as a function of what the modeler considers to be relevant and crucial for its domain study. Thus, the *integrate–and–fire model* focuses upon the temporal summation function of the neuron (Bugmann and Taylor, 1997). The *spike response model* proposed by Gerstner (1999) simplifies the action potential generation to a threshold process. The *resonate–and–fire model* (Izhikevich, 2001) focuses upon the operation of the neuron in a resonating regime. By contrast with the detailed neural models, the computational strength of the spiking neurons arises from the way they interact with each other, when they work cooperatively in large networks.

In summary, this section presented a brief description of the main modeling directions in computational neuroscience. The aim of this was to outline the advantages and drawbacks of each research stream. The review indicates that there is a trade–off problem that a researcher has to solve, when choosing the appropriate level of detail for the task to be simulated. Another aspect, which differentiates the two approaches, concerns the time scale at which they work. Detailed neural models are more appropriate to investigate neural patterns of activity that emerge within a short–time scale (i.e., hundred of ms of seconds). Conversely, networks of spiking neurons can be used to model learning in developmental processes (see Koch, 1999).

As stated in the introduction to this thesis, the general aim of our work is to investigate how cognitive phenomena such as visuomotor coordination, can emerge through development, from the properties of basic elements when they interact and function cooperatively. Consequently, our simulation work will be built using a simplified neural model, considered to be best suited to our modeling purposes. In the remainder of this section, we describe what types of computations with spiking neurons are accounted for in our model.

4.2.2 Neural communication with spikes

Neurons communicate by producing sequences of fixed size electrical impulses called action potentials or spikes (Adrian, 1926). As Rieke and colleagues puts it:

> Spike sequences are the language for which the brain is listening, the language the brain uses for its internal musings, and the language it speaks as it talks to the outside world (Rieke et al., 1997, page 1).

In the theory of neural information processing, there are two main hypotheses with respect to where in the spike train the neural information is encoded: in the neural firing rate or in the precise timing of the spikes. These hypotheses are introduced in turn, below.

Rate coding

Adrian (1926) introduced the concept of *rate coding*, by which the number of spikes in a fixed time window following the onset of a static stimulus code for the intensity of the stimulus. Since Adrian's studies, the rate coding hypothesis has been dominant in the neural computational field (see for a review Recce, 1999). The definition of the rate has been applied to the discovery of the properties of many types of neurons in the sensory, motor, and central nervous system, by searching for those stimuli that make neurons fire maximally (Kandel et al., 2000).

Recent observations on the behavior of cortical visual neurons demonstrated a temporal precision in brain function that is higher than would be predicted from frequency coding (Abeles et al., 1993; Thorpe et al., 1996; Abeles and Gat, 2001). This suggests that firing rate alone cannot account for all of the encoding of information in spike trains. Consequently, in the last decade, the focus of attention in experimental and computational neuroscience has shifted towards the exploration of how the timing of single spikes is used by the nervous system (Gerstner and Kistler, 2002; see Section 4.2.2).

It is important to understand that the pulse coding represents an extension of the way neurons code information, rather than a replacement of the firing rate code. Panzeri and Schultz (2001) proposed such a unified approach to the study of temporal, correlation and rate coding. They suggest that a spike count coding phase exists for narrow time windows (i.e., shorter than the timescale of the stimulus-induced response fluctuations), while for time windows much longer than the stimulus characteristic timescale, there is additional timing information, leading to a temporal coding phase.

Temporal coding by relative latencies

In a temporal code, information can be contained in the *temporal pattern* of spikes (inter–spike interval codes) or in the *time–of–arrival* of the spike (relative spike timings) (Cariani, 1997). In the following, we discuss the later coding scheme, which is implemented in SpikeNNS.

Neurobiological studies of sensory coding of stimuli in the auditory and visual systems revealed that *latency of transmission* is a potential candidate for coding the stimulus features (Bugmann, 1991; Heil and Irvine, 1996; Heil, 1997). An example is the study by Gawne et al. (1996) who showed that the latency of neurons response in the striate cortex is a function of the stimulus contrast and that synchronization based on spike latencies can make an important contribution to binding contrast related information. The coding scheme which represents analog information through differences in the firing times of different neurons is referred to as *delay coding* or *latency coding* (Hopfield, 1995; Gawne et al., 1996; Maass, 1997; Thorpe and Gautrais, 1998).

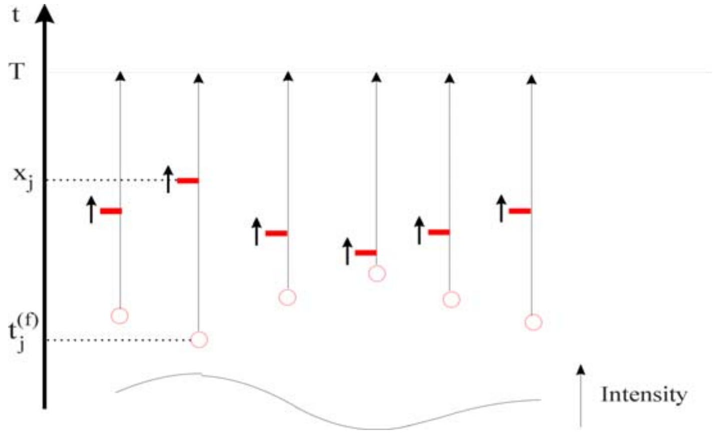


Figure 4.2: Coding by relative delay. The neurons in figure emit spikes at different moments in time $t_j^{(f)}$. The most strongly activated neuron fires first (i.e., second from left). Its spike travels a considerable distance along the axon, until last neuron fires (i.e., the fourth from left). The latencies x_j are computed with respect to a reference time T (adapted after Thorpe et al., 2001).

According to Hopfield (1995) and Maass (1997), a vector of real numbers $(x_1, ..., x_n)$ with $x_j \in [0, 1]$ can be encoded in the firing times t_j of n neurons, such as $t_j^{(f)} = T - c \cdot x_j$, where T is some reference time and $c \cdot x_j$ represent the transmission delays. The timing can be defined relatively to some other spike produced by the same neuron or to the onset of a stimulus. If for each neuron, we consider only the latency of the first spike after the stimulus onset, then we obtain a coding scheme based on the *time–to–first–spike*.

According to Van Rullen and Thorpe (2001) cells can act as ’analog–to–delay converters’. That is, the most strongly activated cells will tend to fire first and will signal a strong stimulation, whereas more weakly activated units will fire later and signal a weak stimulation. This coding scheme was proposed by Thorpe and Gautrais (1998), who argued that during visual object recognition the brain does not have time to evaluate more than one spike from each neuron per processing step. The idea is supported by other experimental studies (Tovee et al., 1993) and was used to implement learning in a number of neural network models, based on the timing or the order of single spike events (Ruff and Schmitt, 1998; Van Rullen et al., 1998, see also Section 6.1).

4.2.3 Computational properties of spiking neurons

Spiking neural models can account for different types of computations, ranging from linear temporal summation of inputs and coincidence detection to multiplexing, nonlinear operations and preferential resonance (Koch, 1999; Maass, 1999). Several recent studies employing rigorous mathematical tools have demonstrated that through the use of temporal coding, a pulsed neural network may gain more computational power than a traditional network (i.e., consisting of rate coding neurons) of comparable size (Maass and Schmitt, 1996; Maass, 1997).

A simple spiking neural model can carry out computations over the input spike trains under several different modes (Maass, 1999). Thus, spiking neurons compute when the input is encoded in temporal patterns, firing rates, firing rates and temporal corellations, and space–rate codes. An essential feature of the spiking neurons is that they can act as coincidence detectors for the incoming pulses, by detecting if they arrive in almost the same time (Abeles, 1982; Softky and Koch, 1993; Kempter et al., 1998).

When operating in the integration mode (see the integrate-and-fire model in Section 4.2.1), the output rate changes as a function of the mean input rate and is independent of the fine structure of input spike trains (Gerstner, 1999). By contrast, when the neuron is functioning as a coincidence detector, the output firing rate is higher if the spikes arrive simultaneously, as opposed to random spike arrival. More precisely, the neuron fires (e.g., signals a detection) if any two presynaptic neurons have fired in a temporal distance smaller than an arbitrary constant $c1$, and do not fire if all presynaptic neurons fire in a time interval larger than another constant $c2$ (Maass, 1999).

For a neuron to work as a coincidence detector, two constraints have to be satisfied: (1) the postsynaptic potential has to evolve in time according to an exponential decay function and (2) the transmission delays must have similar values, so that the simultaneous arrival of the postsynaptic potentials which cause the neuron to fire will reflect the coincidence of presynaptic spikes (Maass, 1999). Note that not any spiking neural model can detect coincidence. For instance, the resonator neuron fires if the input train of spikes has the same phase with its own oscillation, but has low chances to spike if the inputs arrive coincidentally (Izhikevich, 2000).

In SpikeNNS, neurons can compute in two regimes: coincidence detection and threshold–and–fire. Acting as coincidence detectors is more likely for hidden units, when they compute over pulses coming from the input layers. That is, because in our implementation, the input spikes arriving on the afferent connections are affected by similar delays with a small noise factor. The latency of the spikes is given by the firing times of the input nodes. The operation in this computing domain depends also on the neural threshold and on the value of the membrane time constant that describes how fast decays the posystynaptic potentials (see Section 4.3.2).

In the threshold–and–fire mode, neurons perform a linear summation of the inputs in a similar manner with the integrate–and–fire model. The integration of pulses over a larger time interval is particularly required in the case of spikes arriving on the lateral synapses, which are affected by a large range of delays (e.g., from 1 to 10 ms). The shift between computing modes, i.e., coincidence vs. integration, and between different ways of integrating the input signals is illustrated by our simulations in Sections 6.1 and 6.2.1 and discussed in Section 7.1.3.

4.3 Neural model in SpikeNNS

The neural model implemented in SpikeNNS is a simplified version of the Spike Response Model (Gerstner, 1991; Gerstner et al., 1993; Gerstner, 1999) referred to as SRM_0. The Spike Response Model (SRM) represents an alternative formulation to the well–known integrate–and–fire model. Instead of defining the evolution of the neuron membrane potential by a differential equation, SRM uses a kernel-based method. By doing this, the Spike Response Model is slightly more general than the integrate–and–fire models because the response kernels can be chosen arbitrarily, whereas for the integrate–and–fire model they are fixed (Gerstner, 1999). According to Kistler et al. (1997) the Spike Response Model can reproduce correctly up to 90% of the spike times of the Hodgkin-Huxley model. The model can also be used to simulate the dynamics of linear dendritic trees, as well as non–linear effects at the synapses (Gerstner, 1999). The Spike Response Model offers us a powerful computational framework that captures the essential effects during spiking and has the advantages of a simple and elegant mathematical formalization.

4.3.1 Spike Response Model

The Spike Response Model describes the state of a neuron by a single variable, the membrane potential V_i. Figure 4.3a shows the time evolution of the membrane potential of neuron i as a function of time t. Before any input spike has arrived at the postsynaptic neuron i, the variable $V_i(t)$ has the value 0. The firing of a presynaptic neuron j at time $t_j^{(f)}$ evokes a postsynaptic potential in the neuron i modeled by the kernel response ϵ_{ij}. Each incoming spike will perturb the value of V_i and if, after the summation of the inputs, the membrane potential V_i reaches the threshold θ then an output spike is generated. The firing time is given by the condition $V_i(t_i^{(f)}) = \theta$. After the neuron has fired the membrane potential returns to a low value which is described by the refractory period function η. After firing, the evolution of V_i is given by the equation:

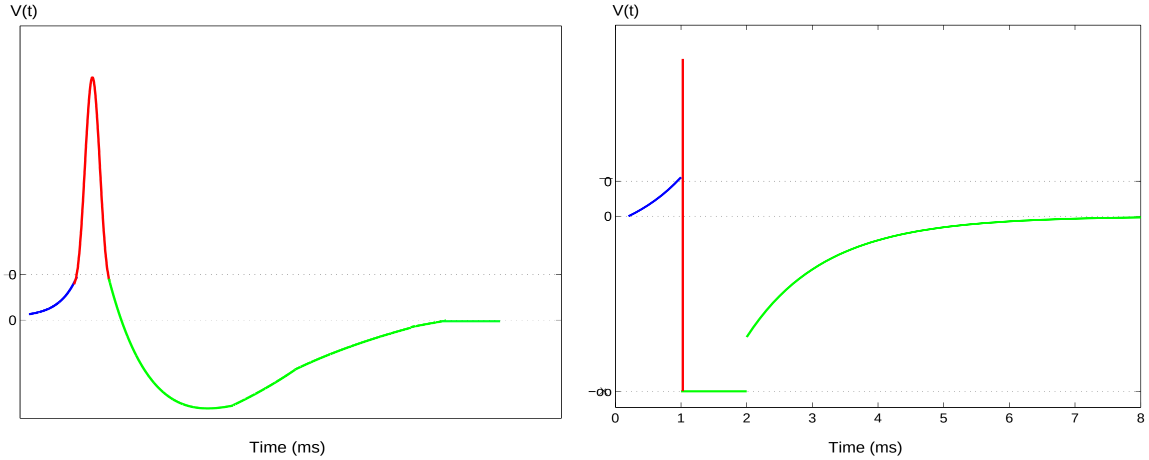
$$V_i(t) = \eta_i(t - t_i) + \sum_{j \in \Gamma_i} w_{ij} \sum_{t_j^{(f)} \in F_j} \epsilon_{ij}(t - t_i, t - t_j^{(f)}) + \int_0^{\infty} \tilde{\epsilon}(t - t_i, s)\mathcal{I}^{ext}(t - s)ds, \qquad (4.1)$$

with $s = t - t_j^{(f)}$. The first term in the equation (i.e., the kernel η_i) accounts for refractoriness in the neuron behavior. The second term represents the contribution of all previous spikes $t_j^{(f)}$ of presynaptic neurons j on the membrane potential of neuron i. Γ_i denotes the set of neurons presynaptic to i, F_j is the set of all firing times of neuron j and w_{ij} are the synaptic strengths between cells (see Figure 4.5).

The kernel ϵ_{ij}, as a function of $t - t_j^{(f)}$, represents the time course of the postsynaptic potential evoked by the firing of the presynaptic neuron j at time $t_j^{(f)}$ (see Figure 4.4a). The shape of the postsynaptic potential depends also on the time $t - t_i$ that has passed since the last spike of the postsynaptic neuron. That is, because if the neuron is in a refractory period, its response to an input spike is smaller than if the neuron is fully responsive. The last term represents the effect on the neuron of an external driving current $\mathcal{I}^{ext}$ and the kernel $\tilde{\epsilon}(t - t_i, s)$ is the linear response of the membrane potential to the input current and depends on the time that has passed since the last output spike was emitted at t_i (Gerstner, 1999).

SRM$_0$

A simpler version of the Spike Response Model can be obtained by neglecting the dependence of the ϵ_{ij} kernel on the term $t - t_i$ (i.e., the effect of the neuron's last spike on the postsynaptic potential function) and by considering a null external current. Consequently,



(a) Membrane potential in SRM

(b) Membrane potential in SpikeNNS

Figure 4.3: (a) Spike Response Model, the membrane potential V of neuron i as a function of time t. (b) SpikeNNS neural model, the time course of the membrane potential V_i. Θ is the neural threshold. See in text for more details on the time evolution of the membrane potential.

equation 4.1 can be rewritten:

$$V_i(t) = \eta_i(t - t_i) + \sum_{j \in \Gamma_i} w_{ij} \sum_{t_j^{(f)} \in F_j} \epsilon_{ij}(t - t_j^{(f)}). \tag{4.2}$$

This version of the Spike Response Model has been entitled SRM$_0$ and has been applied for the analysis of computations with spiking neurons by Maass (1999).

The neural model implemented in SpikeNNS is completely specified by the set of Equations 4.2, 4.3, 4.5, which account for several important aspects of neural behavior: the spiking nature of the neuron, the attenuation of the response at the soma resulting from synaptic input, the absolute and relative refractory periods. The model also accounts for spike latency and noise in the neural response (see description in sections below).

Figure 4.3b shows the time evolution of the membrane potential V_i in the simplified neural model implemented in SpikeNNS. Compared with the membrane potential in the SRM model represented in Figure 4.3a, the shape of the action potential is reduced to a formal event, captured by a δ pulse (the vertical line). After the spike emission, the membrane voltage is reset to a negative value and is kept there for 1 ms (see Section 4.3.3). The ascending curve of the ϵ function also reduces to a pulse, followed by an exponential decay of the post-

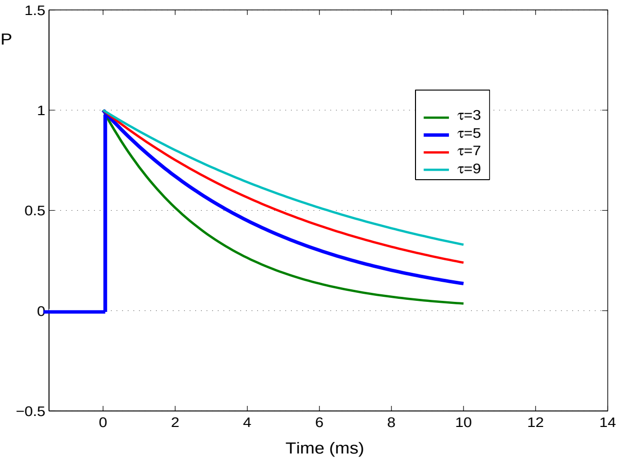


Figure 4.4: Postsynaptic potential function in SpikeNNS. The curve decay with time is plotted for different values of the decay rate, given by the membrane time constant τ_m. Note that the slope of the curve is negatively correlated with the value of the membrane time constant.

synaptic potential (see Section 4.3.2 below). In SpikeNNS, Equation 4.2 is implemented by the activation function ACT_Spike (Section 5.1), which expresses how the membrane potential V of a node i is calculated at a given time t.

Let us now consider the mathematical description of the two kernels ϵ and η required in the Equation 4.2.

4.3.2 Postsynaptic potential function

In SpikeNNS, each hidden neuron is connected to a number of other neurons either from the same layer or from an input or another hidden layer. The firing of any node, i.e., input or hidden, is transmitted to all its postsynaptic units, where it evokes a postsynaptic potential of some standard form (see Figure 4.4). The spike transmission is affected by a noisy delay d, which in our implementation is proportional with the Euclidian distance between the presynaptic and the postsynaptic node (see Section 5.3.2). This delay corresponds to the axonal and dendritic transmission delay of real neurons (Koch, 1999).

When the presynaptic spike reaches the postsynaptic unit, the postsynaptic potential (PSP) jumps to a maximum value, i.e., in our simulation this value is set to 1. Afterwards, it decays exponentially towards the resting value, with a rate being given by the time constant τ_m. In our model, the postsynaptic potential ϵ_{ij} is described as a function of the difference

$s = t - t_j^{(f)} - d$:

$$\epsilon_{ij}(s) = exp(-\frac{s}{\tau_m})\,\mathcal{H}(s) \tag{4.3}$$

with

$$\mathcal{H}(s) = \begin{cases} 1, \text{ if } s \geq 0, \\ 0, \text{ otherwise,} \end{cases} \tag{4.4}$$

where t is the time of consideration, $t_j^{(f)}$ is the time of the presynaptic node firing and d is the delay on the connection. The Heaviside function sets the postsynaptic potential to a null value, for any time moment t that precedes the arrival of the presynaptic spike, that is $t < t_j^{(f)} + d$. Note that in our implementation the postsynaptic potential can only take positive values. Negative values of the postsynaptic potential, corresponding to the inhibitory synapses, are obtained through the multiplication of the PSP value by a negative weight value.

The decay of the postsynaptic potential with a rate given by the membrane time constant τ_m reflects the attenuation with time of the synaptic inputs in the biological neuron (see Section 4.1). The choice of this parameter has a significant effect on the way multiple synaptic inputs are integrated in time (see below).

Temporal summation of postsynaptic potentials

A single synaptic input is rarely sufficient to generate an action potential. The response of a neural cell is usually determined by the way it integrates multiple synaptic inputs. The basic arithmetic that dendrites of a real neuron compute is still a matter of controversy (Poirazi and Mel, 2000). Both linear and nonlinear interactions between synaptic inputs in the brain have been described by neurophysiological experiments (Cash and Yuste, 1999; Koch, 1999) and explored computationally with different formalisms (Rall, 1977; Mel, 1992). In SpikeNNS, we consider that both excitatory and inhibitory inputs accumulate linearly in time.

The total synaptic input to a hidden neuron i at some moment t is given by the contribution of all previous spikes of the presynaptic neurons (see Figure 4.5). The set of presynaptic neurons to the node i is $\Gamma_i = \{j \,|\, j \text{ is presynaptic to } i\}$. The set of all firing times of the presynaptic node j is given by $F_j = \{\, t_j^{(f)} \,|\, V_j(t_j^{(f)}) = \Theta\}$. In SpikeNNS, a limited number

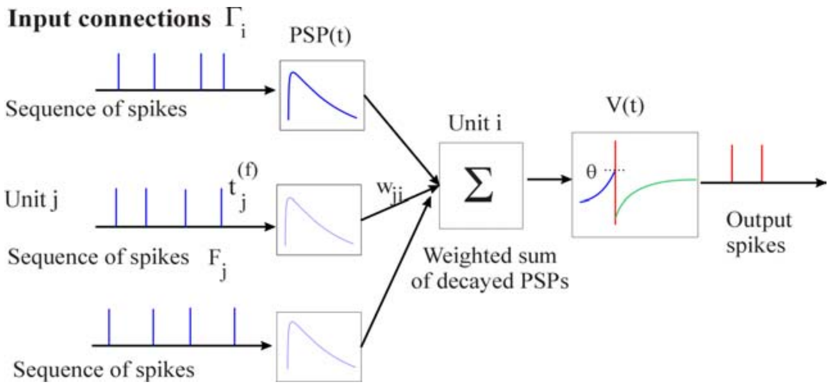


Figure 4.5: The presynaptic contribution to the membrane voltage $V(t)$ of neuron i. Each presynaptic neuron j in Γ_i emits a series of spikes F_j with firing times $t_j^{(f)}$. Neuron i computes the weighted sum of the decayed PSPs. If the sum exceeds the threshold then an output spike is generated.

of spikes per neuron are stored (e.g., a maximum of 10 spikes/neuron were stored for the simulation run in Section 6.1).

In our model, the slope of the postsynaptic potential curve is negatively correlated with the value of the membrane time constant τ_m (see Figure 4.4a). That is, for large values of τ_m, the postsynaptic potential persists longer and it allows the temporal summation of inputs that produce in this way, an aggregate PSP larger than would be elicited by an individual input. The neural threshold Θ and the membrane time constant τ_m represent the principal parameters in determining how many excitatory inputs are needed for a neuron to fire. For example, the choice of a membrane time constant $\tau_m = 5$ (the blue graph in Figure 4.4a) causes an exponential decay of the postsynaptic potential from the maximum value to 0, in about 10 ms. This relatively slow decay of the postsynaptic potential curve favors the significant summation of the synaptic inputs which arrive in a time window no larger than $4-5$ ms. For instance, given the above value of the τ_m, in the simulations run in Chapter 6, the threshold values were set so that, at least three synaptic inputs (e.g, most commonly 4 or 5 inputs) were necessary for a postsynaptic spike to be emitted.

4.3.3 Refractoriness

After emitting the spike a node enters a refractory period. In the SRM$_0$ model the neuron behavior in the refractory period depends only on the last firing moment t_i. This means that the effects of any previous own spikes are discarded. Gerstner (1999) referred to this as

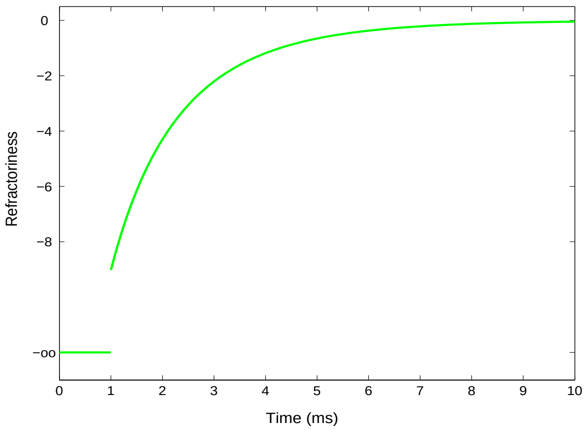


Figure 4.6: Refractory function in SpikeNNS. The Parameters $m = 0.8$ and $n = 3$. See in text for the meaning of these values.

the 'short-term memory' approximation, that is, the neuron remembers only its most recent firing. The approximation can be considered correct if the intervals between two spikes are much longer than τ_m. This would make the influence of previous own spikes on the current refractory behavior, negligible. The refractory period is described as a function of the difference $u = t - t_i$ and is modeled by the equation:

$$\eta_i(u) = -\ exp[-u^m + n]\, \mathcal{H}'(u)\, \Theta. \tag{4.5}$$

with

$$\mathcal{H}'(u) = \begin{cases} \infty, & \text{if}\, 0 \leq u < 1, \\ 1, & \text{otherwise} \end{cases} \tag{4.6}$$

where Θ is the neural threshold and m and n are arbitrary parameters which adjust the slope of the decay curve (see Figure 4.6) (formula adapted after Sougné, 1999).

After a spike emission, the biological neuron behavior is characterised by the existence of two periods. First, there is an absolute refractory stage, while no further action potentials can fire. This is followed by a relative refractory period when it is difficult, but not impossible to generate an action potential (Section 4.1). In the neural model described here, the absolute refractory period lasts 1 ms, during which the Heaviside function (Equation 4.6) maintains the value of the η kernel at $-\infty$ (see Figure 4.6). The duration of the relative refractory period depends on the values of the parameters m and n. In Figure 4.6 these value are set so that the average inter–spike value is 10 ms.

4.3.4 Coding of the input patterns

When computing with temporal patterns in a network of spiking neurons, the input and output to the model is represented by vectors of *time series*, instead of vectors of analogical numbers, as in conventional neural network models (Maass, 1999). In our implementation, an input pattern consists of a series of firing times F_j of the input units j that encode the training values, as explained below.

The latency code described in Section 4.2.2 represents analog information through differences in the firing times of the neurons. Here, we define a coding scheme, where the analog values x_j can be encoded by the time advances of the input spikes to a reference time T_{int} (see Hopfield's formulation in Section 4.2.2). Figure 4.7 shows an input pattern represented by the spikes on five input units. An input spike has a large contribution on a postsynaptic node potential if it has just reached the node and a smaller effect if its arrival is less recent. Given equal transmission delays d, a vector of real values $(x_1, ..., x_n)$ with $x_j \in [0, 1]$ can be encoded in the postsynaptic potentials of the input spikes as a function of the difference between an arbitrary value T_{int} and the input units firing times $t_j^{(f)}$:

$$x_j = \epsilon(T_{int} - t_j^{(f)} - d) = exp\,(-\frac{T_{int} - t_j^{(f)} - d)}{\tau_m}). \tag{4.7}$$

Let us explain how a network can be trained with input values coded by this scheme. For instance, if the firing time of an input unit is $t_j^{(f)} = 2$ ms, the delay value is set to $d = 2$ms, $T_{int} = 10$ ms, and $\tau_m = 5$ ms, then a hidden unit receives a signal $x_j = 0.4$, given by the postsynaptic potential in Equation 4.7 (see first PSP value in Figure 4.7a). This value can be further used for the training of the hidden layer weights. For instance, in a self-organization process, weights are adapted as a difference between the input ($\epsilon = 0.4$) and the current value of the weight. Accordingly, the weights of a winner for this input will learn the value of 0.4. Note that, learning on the synapses of spiking neurons depends also on their output values, in a different way that it is the case for the continuous, rate–coding neurons. That is, learning is applied only to the excitatory synapses of those neurons, which fire. The later a hidden neuron fires, the smaller is the contribution of the input values on its afferent synapses change (see also Section 5.2.3).

In SpikeNNS, two coding schemes are used: *time coding* and *coding by synchrony*. Which scheme is used depends on the way the input patterns are specified and on the operation mode of the neurons (see Section 4.2.3). That is, neurons can either operate as integrators,

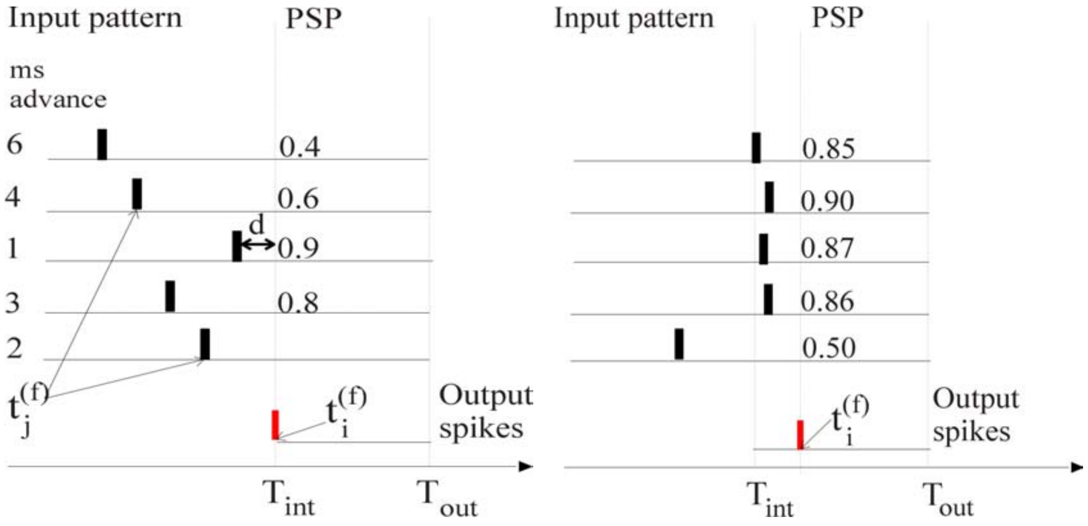


Figure 4.7: An input pattern consists of a train of spikes emitted at the times $t_j^{(f)}$. (a) A time coding scheme, where values are encoded in the spikes relative latencies to the T_{int} moment. In left the time advances of the input spikes are shown, computed as $T_{int} - t_j^{(f)} - d$. The PSP values represent the real values seen by the hidden units and which are used in training the synaptic weights. (b) Coding by synchrony, where the hidden neurons spike by detecting the coincidence in the firing of the input units.

by summing up the input values that arrive in a large time window, or they can function as coincidence detectors, by detecting the synchrony in the input spikes emission. Accordingly, the temporal coded inputs can be specified in two ways.

Firstly, real values can be encoded as described above, by using the spike time advances to a fixed reference time T_{int}. This method offers complete control over what values are encoded and received by the hidden units. Secondly, information can be encoded in the synchronous emission of the input spikes. This means that the hidden units can detect the coincident firing of a subset of input units. It represents a means to generate hidden units selectivity for a neural event that is characterised by the correlated firing of a set of presynaptic units.

With regard to how the time–coded input patterns are effectively implemented and applied in SpikeNNS, it consists of the following scenario. The series of firing times F_j corresponding to one pattern read from the input file, are converted in series of $1'$s and $0'$s that are applied to the hidden network at the times $t_j^{(f)}$. All the 1 values representing input spikes with the same timing $t_j^{(f)}$ are collected in a *subpattern* with that time stamp. By doing this, a single time–coded pattern generates as many subpatterns as different firing times are in the original input pattern.

4.4 Learning with spiking neurons

Part of the work carried out on this thesis has focused upon the implementation of an extension of the SNNS simulator (Stuttgart Neural Networks Simulator) to support simulations with spiking neurons (see Chapter 5). The new simulator version, SpikeNNS, implements a general training framework adapted to the characteristics of learning with spiking neurons and temporal coded input patterns (see Section 5.2.3). By creating a general learning frame, the simulator is not limited to the learning rules currently existent in SpikeNNS, but it can be easily extended with any learning functions.

The current version of the simulator implements several learning rules for a self–organization process with spiking neurons. Issues of learning with spiking self–organizing maps have been discussed in Section 2.2.5. The adaptation rules implemented in SpikeNNS for the self-organization of a spiking neural map with plastic lateral synapses are described in Sections 6.1 and 6.2 along with the presentation of the simulations performed.

In Section 2.2.5, we emphasized the crucial role played by the horizontal connectivity in development in the real and artificial networks. Accordingly, significant modeling efforts have been dedicated to the integration of learning in lateral synapses in the models of cortical functions. Most of these attempts focused on exploring Hebbian–like association rules (see Section 2.2.5). More recently, experimental investigations shown that the relative timing of the pre– and post–synaptic neurons plays an important role in determining whether a synapse is potentiated or depressed (Markram et al., 1997; Zhang et al., 1998). Newer formulations of the Hebbian rule take into account the temporal order of spiking, so that potentiation occurs only if the postsynaptic excitatory potential precedes the firing of the postsynaptic neuron by at most 20 ms. Depression is the rule for the reverse order (Song et al., 2000). In the following, we describe briefly a number of learning rules that illustrate different ways to compute and learn with spiking neurons. Some of them will be implemented in future work.

Learning on neural firing rates. Choe and Miikkulainen (2000) proposed an adaptation of the lateral weights according to the Hebbian rule and based on the spiking rates of leaky integrate-and-fire neurons:

$$w_{ij}(t) = \frac{w_{ij}(t-1) + \alpha \cdot V_i X_j}{\mathcal{N}}, \tag{4.8}$$

where $w_{ij}(t)$ is the connection weight between neurons i, j, $w_{ij}(t-1)$ is the previous value of the weight, α is the learning rate. V_i and X_j are the average spiking rates of the neurons computed after the network has reached a stable rate of firing. $\mathcal{N}$ is a normalization factor that prevents the excessive increasing of the weights. This learning algorithm was used successfully for image segmentation and contour binding by the synchronization of neural group activity (Choe and Miikkulainen, 2000).

Learning on the spike times synchronization. Sougné (1999) proposed a neurocomputational model for binding and inference that solves the binding problem by means of oscillation synchrony. Learning of the connection weight between two nodes i and j is defined by Equation 4.9. This expresses the fact that with every synchronous firing of the nodes, the weights w_{ij} as well as w_{ji} are increased by a constant value c:

$$\Delta w_{ij}^{+} = \begin{cases} c, \text{ if } t_j^{(f)} = t_i^{(f)} \\ 0, \text{ otherwise.} \end{cases} \tag{4.9}$$

The delays of the connections also undergo learning, so that at each synchronous firing of the nodes i and j, the delay d_{ij} as well as d_{ji} are set to a fixed value, which favors the synchronization of neurons at a certain frequency of oscillation. A decrease of the connection weights (i.e., forgetting of the binding) occurs if the two nodes do not fire inside a certain time cycle (i.e., up to 250 ms). This type of learning is successfully used to implement a short time memory, binding on multiple instantiation and with multiple predicates.

Spike–timing dependent plasticity (STDP). Song et al. (2000) proposed an adaptation of the Hebbian learning rule, which takes into account the relative timing of the pre- and post-synaptic action potentials. The synaptic modification arising from a pair of pre- and post-synaptic spikes separated by a time interval Δt is given by the function $F(\Delta t)$:

$$F(\Delta t) = \begin{cases} A_{+} exp(\Delta t/\tau_{+}), \text{ if } \Delta t < 0 \\ -A_{-} exp(-\Delta t/\tau_{-}), \text{ if } \Delta t > 0 \end{cases} \tag{4.10}$$

where A_{+} and A_{-} determine the maximum amounts of synaptic modification and the parameters τ_{+} and τ_{-} determine the ranges of pre–to–post synaptic interspike intervals over which synaptic change occur. Strengthening of a synapse occurs for pre–synaptic action potentials which precedes postsynaptic firing by no more than 50 ms. Weakening of the synapse is produced by a presynaptic action potential that follows the postsynaptic spike. The main role of this form of STDP is to strongly weaken causally ineffective inputs, hence, synaptic weakening dominates over synaptic strengthening.

Spike–timing learning has been mostly implemented for excitatory synapses, due to the rich experimental data showing how potentiation and depression of these synapses occurs in the brain. Conversely, there is much less neurobiological data from which to conceptualize the modification rules for inhibitory synapses, and consequently fewer cognitive models account for the plasticity of inhibitory synapses (see Roberts, 2000; Soto_Trevino et al., 2001).

Learning in the inhibitory synapses. In SpikeNNS learning in the inhibitory synapses is implemented according to the rule proposed by Levy and Desmond (1985). The authors proposed that presynaptic activity paired with postsynaptic inactivity leads to the potentiation of the active inhibitory synapse, while postsynaptic activity is required (indifferent of the presynaptic activity) for a loss of strength of the inhibitory synapse (see also Section 6.1). More recently, experiments in the mormyrid electric fish have suggested that pairing delays between pre- and post-synaptic spikes that cause excitatory synapses to decrease cause the inhibitory synapses to increase. Thus, if the postsynaptic inhibitory potential follows the postsynaptic spike within a time window of about 50 ms, than the inhibitory synapses is potentiated (Roberts, 2000).

Chapter 5

SpikeNNS simulator: design and implementation

The development in the last decade of a third generation of neural networks (Maass, 1997) has required the design of dedicated simulation environments. Simulation platforms either for the detailed modeling of individual neurons, such as Neuron (Hines and Carnevale, 1995) and Genesis (Bower and Beeman, 1998) or dedicated to large–scale neural simulations, such as SpikeNet (Delorme et al., 1999), Infernet (Sougné, 1999), NSL (Weitzenfeld et al., 1999) have been created in recent years. Due to the large attention given to realistic modeling of the structure of biological neurons, the most powerful and flexible simulators in existence today implement detailed neural models. As a result, these simulators become more appropriate for the modeling of single neuron behavior and less suitable for large-scale modeling of cognitive phenomena.

The aim of the modeling work presented in this thesis is to provide an example of how cortical processes, such as movement planning or visuomotor mapping, are grounded in the neural substrate. In Section 4.3 it was argued that a simplified neural model, such as the Spike Response Model (SRM) (Gerstner, 1999) corresponds best to our modeling goals. To date, SRM has been implemented only as part of simulations that are dedicated to specific tasks (Gerstner, 1999; Lerchner, 2001). Therefore, in the absence of a flexible simulation platform that can be adapted to our modeling goals, the approach taken in this thesis was to create a library of functions that model networks of SRM neurons. This library was implemented as an extension of the general purpose–simulator Stuttgart Neural Network

Simulator (SNNS) (Zell et al., 1992).

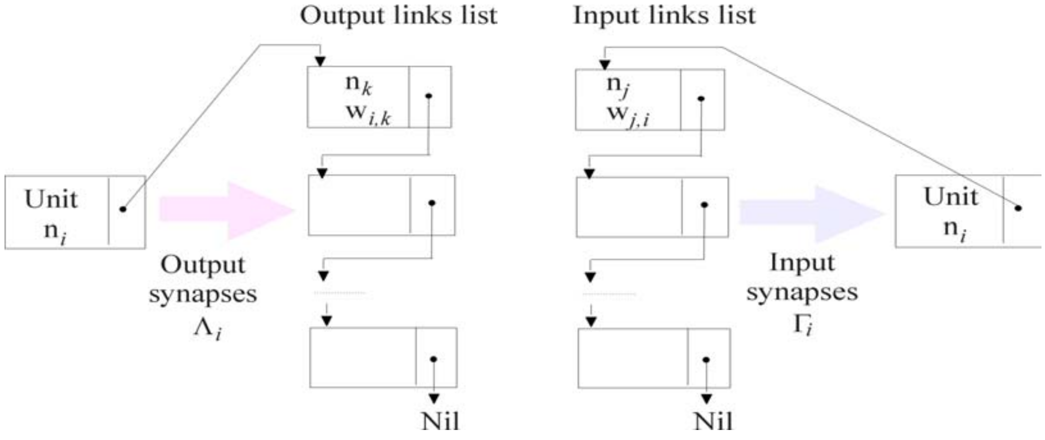
This chapter presents the implemented system, entitled SpikeNNS. We begin in Section 1 by describing the implementation of the neural model introduced in Chapter 4, Section **??**. When modeling large–scale networks of spiking neurons with plastic synapses, the time and memory efficiency of the simulation became essential issues in the design of the simulator (Jahnke et al., 1999; Mattia and Del Giudice, 2000). In the first section, we present several versions of the neural model implementation, aimed at decreasing the computational workload per neural unit. Section 2 focuses upon how learning with networks of spiking neurons is implemented. In this part, we deal with efficiency issues at the network level, partially solved by implementing a discrete event–driven strategy. The issue of high neural activity patterns is addressed, when spike events on the order of 10^5 are generated in the network. Several strategies are proposed that lead to a highly efficient event–driven algorithm that reduces the simulation time up to 20 times compared to the baseline algorithm. Finally, Section 3 outlines the configurable features of the SpikeNNS simulator.

5.1 Implementation of SpikeNNS neural model

The neural model implemented in SpikeNNS is a simplified version of the Spike Response Model (Gerstner 1999), described by the set of Equations 4.2, 4.3, and 4.5 (Section 4.3). This section presents the effective implementation of the neural model and focuses upon the time–efficiency of the simulation.

5.1.1 Design considerations

The implementation of a neural model requires consideration of several design issues. One issue concerns the type of the synaptic connectivity employed. This falls into two categories: (1) the *regular connectivity* follows some deterministic rules, giving rise to patterns such as receptive fields or topographical projections; (2) the *non-regular connectivity* consists of sparse, probabilistic connections. In SpikeNNS both categories of connection patterns are implemented. However, while regular connections are relatively simple to configure and manage efficiently with respect to the time and memory load, sparse connections are more difficult to deal with. The difficulty concerns both the generation of the probabilistic pattern of synapses, and the efficient management of such a pattern (see Section 5.3.2).



(a) Sender oriented scheme (b) Receiver oriented scheme

Figure 5.1: Data structures used to store the connections list. (a) A sender-oriented connectivity stores the weights and addresses of the output connections Λ_i (b) In the receiver-oriented scheme the weights and addresses of input links Γ_i are stored.

A commonly used method in representing the sparse connectivity pattern is to create a list of connections, one for each neuron n_i. Each item in the list contains the weight of the connection and the address n_j of the pair unit. If the address stored denotes the target neuron n_j to which n_i sends spikes, then the scheme is called *sender–oriented* (Figure 5.1a). When the list stores the addresses of the input nodes, that is, those neurons from which n_i receives spikes, the scheme is referred to as *receiver–oriented* (Figure 5.1b). Each of these schemes demands different processing of spike generation and distribution.

In a receiver–oriented scheme it is the job of the receiver unit to handle the incoming spikes and to set the transmission delays. Whenever a unit computes its activation it checks all its input synapses to see whether a spike has been emitted on any of them. If this is the case, then it sets the delay of the spike transmission (see lines 3-6 in Algorithm 1). It is important that the algorithm ensures that the delay for each spike is computed only once. Alternatively, in the sender-oriented scheme, the distribution of spikes and the generation of synaptic delays are managed by the emitting neuron.

A number of simulators implementing the traditional rate–coding neural model use the receiver-oriented scheme (i.e., SNNS). The reason for this is that scheme can be advantageous in the case of a continuous neural output function, which causes constant firing activity on synapses. However, this is not the case with spiking neurons activity. In Sections

5.1.2 and 5.1.3 the appropriateness of each connectivity scheme for the implementation of the spiking neural model is discussed.

Another design issue which has to be considered at the neural level is the method by which the integration of the membrane potential is realized (Equation 4.2, Section 4.3). A straightforward way to compute the sum term in Equation 4.2 consists of the following. Whenever an update of the unit activation is requested, the values of all previous postsynaptic potentials coming from all presynaptic nodes are computed, integrated and added to the refractory kernel. The optimization of this method is possible if we consider that the depolarization of the postsynaptic potential ϵ follows a deterministic function (see Equation 4.3). Therefore, its value at time t can be expressed as a function of its previous value at $t - \Delta t$. This idea is explored in Algorithm 3 in Section 5.1.3.

Besides considering *how* we integrate a neuron activity, we have to decide *when* we do this integration. Solutions to this problem reside in using either a synchronous, time-driven integration of unit activity or an asynchronous integration. Both strategies are implemented and compared with respect to their efficiency for simulation of pulsed neural networks.

5.1.2 Activation and output functions - version 1

The first version of the Spike Response Model (Section 4.3) in SpikeNNS was created using the SNNS receiver-oriented connectivity scheme. Equations 4.2, 4.3 and 4.5 have been implemented within the body of two functions: the *Activation* function (ACT_Spike) and the *Output* function (OUT_Spike). A straightforward (non-optimal) implementation of these functions is shown in Algorithm 1. Note that the Output function could have been incorporated in the Activation function, but is kept separate to ensure compatibility with the implementation of these functions in SNNS.

To get a quantitative measure of the complexity of the algorithms presented in this section, we estimate the scaling of the number of operations required per algorithm, with the variables of the simulation process. The main variables are: T, the time of the simulation; N, the number of neurons in the network; S, the synapses per neuron; F_j, the set of non-negligible firing times on each synapse. Other variables that affect the number of the operations executed will be introduced in the following subsections. Note that our measure NO is given by the number of operations in the performance critical parts (i.e., such as loops) and does not reflect the exact number of low-level operations required in the implementation. In the

Algorithm 1 ACT_Spike v1 -receiver oriented

```
Require: node $i$, current time $t$
Ensure: activation value $V_i(t)$
 1: $V_i(t) = 0$
 2: for all $j \in \Gamma_i$ do {presynaptic nodes to $i$}
 3:     $t_j^l$ = get_spike($j$) {receiver unit checks for new spike from this unit}
 4:     if $t_j^l \notin \mathrm{F}_j$ then
 5:         $d_j^l$ = noisy_delay(length($w_{ji}$)) {generate delay}
 6:         insert_spike ($t_j^l, F_j$) {insert the spike in the set of firing times}
 7:     end if
 8:     for all $t_j^k \in F_j$ do {all previous firing times of node $j$}
 9:         $PSP_j^k(t) = \epsilon(t - t_j^k - d_j^k)$ {compute $k$th postsynaptic potential}
10:         $V_i(t) = V_i(t) + PSP_j^k(t) \cdot w_{ji}$ { compute weighted sum of PSPs}
11:     end for
12: end for
13: $REF(t) = \eta(t - t_i^{(f)})$ {calculate refractory period}
14: $V_i(t) = V_i(t) - REF(t)$ {subtract REF from the PSPs aggregate}
15: if $V_i(t) \geq \theta$ then {if the neuron fires}
16:     $t_i^{(f)} = t$ {record firing time}
17: end if
    {No propagation of emitted spike is done by the sender unit}
```

Algorithm 2 OUT_Spike

```
Require: activation $V_i(t)$
Ensure: output value $O_i(t)$
if $V_i(t) \geq \theta$ then {the neuron fires}
    $O_i(t) = 1$ {emit a signal, set output to 1}
else
    $O_i(t) = 0$ {no signal}
end if
```

case of Algorithm 1, the critical computational effort is represented by the computation of the ϵ kernel (i.e., given by an exponential function) for F_j times, for each input synapse in S (lines 8-11). This entails $NO \approx S \cdot |F_j|$. In addition, checking the firing activity of each input synapse (lines 3-7, Algorithm 1) needs a supplementary effort that scales with S. Then, the estimated number of operations for Algorithm 1 is:

$$NO_1 \approx S \cdot (|F_j| + 1). \tag{5.1}$$

5.1.3 Acceleration of the activation function - version 2

Let us now consider the sender-oriented connectivity scheme. In Algorithm 1, the receiver unit checks the activity of *all* incoming synapses at each integration step (lines 2-7). By contrast, in a sender–oriented strategy there is no need for this additional operation. Whenever a spike is generated, it becomes the task of the emitting unit (or of the main simulation engine) to compute the propagation delays for all output synapses. Because the propagation is performed only once for each spike, the uniqueness of the delay value per spike is ensured. The sender–oriented scheme is implemented in Algorithm 3.

Algorithm 3 ACT_Spike v2 - sender oriented

```
Require: node $i$, current time $t$, previous integration time $t_{int}$ and previous weighted sum
    $PSP_i(t_{int})$
Ensure: activation value $V_i(t)$
 1: $x = t - t_{int}$ {elapsed time from last integration}
 2: $PSP_i(t) = PSP_i(t_{int}) \cdot exp(-x/\tau_m)$ {decay the sum of past PSPs}
 3: for all $w_{ji}^k$ with $t_j^k > t_{int}$ do {only spikes that occurred since last integration}
 4:     $PSP_j^k(t) = \epsilon(t - t_j^k - d_j^k)$
 5:     $PSP_i(t) = PSP_i(t) + PSP_j^k(t) \cdot w_{jj}$ {add new weighted PSP to the sum}
 6: end for
 7: $V_i(t) = PSP_i(t)$ {set the voltage to the aggregate of inputs}
 8: $REF(t) = \eta(t - t_i^{(f)})$ {compute refractory period}
 9: $V_i(t) = V_i(t) + REF(t)$ {subtract refractoriness }
10: if $V_i(t) \geq \theta$ then {if the neuron fires}
11:     $t_i^{(f)} = t$ {record firing time}
12: end if
    {Note that propagation of spikes is handled by the main simulation engine (lines 14-17
    in Algorithm 4)}
```

Algorithm 1 can be further improved through the judicious choice of the method used to integrate the membrane potential. The new version of the algorithm exploits the fact that between any two firing moments, a neuron depolarization has a deterministic evolution. Rather than computing the sum over all the postsynaptic potentials at each time step, the contribution of the previous spikes effects on the node i is stored and it is decayed every time when a new potential has to be added. Thus, at the current time moment t the previous sum computed at time t_{int} is decayed by the formula $PSP_i(t) = PSP_i(t_{int}) \cdot exp[-(t - t_{int})/\tau_m]$, where τ_m is the time membrane constant (line 2 in Algorithm 3).

Apart from decreasing the simulation time this method has the advantage of minimizing the memory load. Thus, for the receiver–oriented scheme the transmission delays d_j^k of all spikes in F_j emitted on each synapse j in the network have to be stored. For instance, in the case of a network with $1k$ units and 10% connectivity, where the last 10 spikes are stored for each synapse, the additional memory load is on the order of 10^5 floats. Instead, in the sender-oriented scheme only two values are kept within each neuron structure: the last integration moment t_{int} and the corresponding decayed sum $PSP_i(t_{int})$.

As a result of the optimizations implemented, the computational effort of the Algorithm 1 is reduced in Algorithm 3 at $NO \approx aS \cdot bF_j$, where a and b are two variables which range between 0 and 1. The term bF_j represents the number of spikes released on a single synapse between two integration moments. The value of the parameter b depends on the time resolution of the simulation. By considering an algorithm that increments simulation time in fine clock steps (e.g., 1 ms) (see Section 5.2.1) or which integrates the neuron's activity for each new spike received on a synapse (see Section 5.2.1) than bF_j is 1. This means that no more than one new spike per synapse has to be processed at any call of the activation function. The term aS represents the percent of synapses on which new spikes are emitted between two integration moments. Hence, the estimated number of operations for computing a single unit in a time bin reduces to

$$NO_2 \approx aS. \tag{5.2}$$

with a a variable dependent on the network activity. The differences in performance between the two versions of the *Activation* function (Algorithm 1 vs. 3) becomes more obvious when these algorithms are employed for the integration of a large number of units over a long time interval (see Section 5.2.1).

The algorithms presented above illustrate in turn, how receiver– and sender–oriented schemes may be implemented separately. However, by using the SNNS libraries, it has been possible to construct a mixture of these algorithms that uses both sets of data structures. This is, because SNNS implements a receiver-oriented scheme, to which we added, rather than replacing (i.e., for compatibility reasons the default connectivity could not be deleted) structures for the sender–oriented connections.

By doing this, we found that the most time efficient method for integrating spikes may be a composite version of the two algorithms. That is, during a first time period when the input patterns are applied, values associated with these spikes are stored in the input synapses, using the receiver–oriented structures. These spikes are integrated only once at an arbitrary time moment T_{int}, by using the first algorithm (see description of time coded input patterns in Section 4.3.4). After first integration is done at T_{int}, further spikes are integrated one at a time, using the second algorithm. The advantage of using the first algorithm to store and process the values of the input patterns spikes, is that it avoids the generation and handling of a spike event for each firing of an input unit. In this way it is possible to significantly reduce the number of events the algorithm has to deal with. Nevertheless, this is not a portable solution, because it makes use of two lists of connections that can introduces a significant memory load. For spiking neuron simulations the sender-oriented connectivity is the best choice from the two schemes.

What remains to be explained for Algorithm 3 is how it computes the values of the incoming postsynaptic potentials (lines 3 and 4). This is because, by implementing a sender–oriented connectivity solely, a unit does not have access to the values of the input synapses (i.e., the weights and spikes delays). Consequently, it does not know how to calculate the postsynaptic potentials. A possible solution is to parse the output connection structures of all units in the network to find a neuron input synapses. However, this is an exhaustive, time consuming solution. An alternative is to create a data structure for each spike emitted on a synapse that stores the necessary values: the addresses of the emitting and target units n_i, n_j, the firing presynaptic time t_j^k, the transmission delay d_j^k and the synaptic weight w_{ji}. This solution, of a spike-oriented nature, in contrast with the synapse-oriented scheme, leads us naturally into an event-driven approach to the simulation (see Section 5.2.1 below). The complete description of how these spikes are created and delivered is described in Section 5.2.2.

5.2 Simulation of networks of spiking neurons

Up to this point, we focussed upon the simulation of one time step of neural behavior. This consists of computing the neuron's membrane potential by integrating the effects of the incoming spikes (i.e., the postsynaptic potentials) with the effect of the neuron's own spikes (i.e., the refractoriness). In this section, we address the issue of information processing and learning within a network of spiking neurons during a certain time interval.

An essential implementation aspect in building discrete simulations concerns the way the simulation time is progressed. Performing a simulation means to mimic the occurrence of events (i.e., spikes) as they evolve in time and recognizing their effects as represented by states (i.e., network activity) (Ferscha and Tripathi, 1994). The simulation of one basic time step for a whole network is called a *time slice* (Jahnke et al., 1995). Most commonly each time slice involves four operations:

- **Activity propagation.** Spikes from input or hidden units are propagated through the hidden layers of the network.
- **Integration.** Each unit in the network calls the *Activation* function and integrates its inputs.
- **Output.** If a unit fires, the spike time is recorded and the signal is managed according with the implemented strategy (see Sections 5.1.2, 5.1.3).
- **Learning.** Adaptation of the synaptic weights can follow each pattern application or wait until all input patterns are applied. Information concerning learning (e.g., spikes timings, frequency of spiking) may be recorded for each time slice.

In this section, we present a number of algorithms for the simulation during a time period T, of the four operations described above. The first three of them, namely the activity propagation, the integration and the output are implemented in the function PROPAGATE_Spike. Learning is applied after a time coded pattern is presented to the network and the activity is propagated through the hidden layers for a certain time interval. A general training framework is described in the LEARN_Spike function. Note that when computing with spiking neurons, a single time coded pattern actually consists of a set of input signals, each applied as a distinct sub-pattern at a certain time moment (see description of input patterns application in Section 4.3.4). Hence, we bear in mind that the PROPAGATE_Spike function applies only a time coded pattern, consisting of several sub-patterns, while the entire training set is learned using the LEARN_Spike function.

5.2.1 Continuous vs. event–driven protocols

Currently, two kinds of discrete simulation are distinguished with respect to the way simulation time is progressed: *time driven* simulation and *event driven* simulation. In a *time driven* simulation, time is advanced in steps of a constant size Δt (Ferscha and Tripathi, 1994). The choice of Δt influences the simulation accuracy. That is, ticks short enough to guarantee the required precision generally imply longer time simulation. The continuous, time–driven protocol appears to be the *de facto* standard for detailed neural modeling (Bower and Beeman, 1998). This is because, it ensures the time resolution needed for the integration of the differential equations describing the model.

On the other hand, large–scale simulations that involve learning in plastic synapses and long training procedures are faced with important time and memory efficiency issues. These are usually tackled through the use of simplified neural models and event–driven strategies (Watts, 1994; Mattia and Del Giudice, 2000; Claverol et al., 2002). Efforts have also been made to implement simulations on parallel computers (Fujimoto et al., 1992; Brettle and Niebur, 1994; Jahnke et al., 1995) or to create dedicated hardware (Schoenauer et al., 1998; Elias and Northmore, 1999).

Continuous time algorithm

A straightforward implementation of a continuous–time algorithm for the simulation of the network activity during a training step is presented in Algorithm 4.

Algorithm 4 PROPAGATE_Spike - time driven

```
Require: current pattern number $p$, simulation time $st$, time out $t_{out}$
 1: $ct = st$ {set the time clock}
 2: while $ct < t_{out}$ do {while current time less than time out}
 3:   for all $l \in Input(p)$ do {all input units}
 4:     if $l(ct) = 1$ then {if there is an input value at the current moment}
 5:       $out^l$ = OUT_Identity($l(ct)$) {send a pulse}
 6:       for all $i \in \Lambda_l$ do {output synapses to hidden nodes}
 7:         $d_i^l$ = noisy_delay(input_delay) {set delay for input spike from $l$ to $i$}
 8:       end for
 9:     end if
10:   end for
11:   for all $i \in$ Net do
12:     $act^i$ = ACT_Spike($i, ct$) {call activation function}
13:     $out^i$ = OUT_Spike ($act^i$)
14:     if $out^i$ then {propagate spike to postsynaptic nodes}
15:       for all k $\in \Lambda_i$ do {all postsynaptic nodes to $i$}
16:         $d_i^k$ = noisy_delay(length($w_{ik}$)) {compute spike delay}
17:       end for
18:       insert($i$, LearningNodes) {synapses of firing units are subject to learning}
19:     end if
20:   end for
21:   $ct+ = \Delta t$ {increment the current time contor}
22: end while
```

Let us consider the computational effort demanded by the execution of the Algorithm 4, when the activation functions described in Section 5.1 are used in turn (in line 12). Thus, by using the first version of the activity function from Algorithm 1, to compute a network with

N neurons, S synapses per neuron, during a time interval T with a time step Δt, a number of operations are required:

$$NO_3 \approx \frac{T}{\Delta t} \cdot N \cdot S \cdot (|F_j| + a)). \tag{5.3}$$

The first term in the sum accounts for the computational effort needed to integrate the activities of all units (lines 11-13, Algorithm 4). The second term estimates the operations required for the distribution of spikes that occur in a time slice, at a network activity a (lines 14-17, Algorithm 4). The value of a is defined as the average number of spikes in a time slice divided by the number of neurons and takes values in the range of $[0, 1]$. When using the second version of the activity function (Algorithm 3), both computational complexities scale with the network activity a. The number of operations reduces to:

$$NO_4 \approx \frac{T}{\Delta t} \cdot a \cdot N \cdot S \cdot 2. \tag{5.4}$$

These two estimations of the number of operations, neglect the computational effort required for the distribution of the input spikes (lines 3-10, Algorithm 4). This is mainly because, input units have an identity activation function that does not require the computation of a response kernel and the total number of input patterns is small compared to the number of times the network activity during a training step.

The number of operations implemented by Algorithm 4 is higher when using the first version of the activation function. This is, because at each time step, the activity of every unit is updated by computing the ϵ values for all presynaptic firing times in $|F_j| \approx 10$. Conversely, the second version of the algorithm decreases the number of operations, by reducing the computational effort per neuron to the integration of one spike from each active synapse aS in the network.

Both implementations of Algorithm 4 operate in a synchronous way. They integrate all neurons N at each time step $T/\Delta t$ that determines their poor scaling with the network size and the simulation time. Moreover, the global integration of the network activity may not be necessary, due to the fact that at each time moment only a small number of neurons receive a new spike. An observation can be made here, with respect to the differences between the simulation of spiking neurons and rate–coding neurons. Learning based on a continuous neural model makes use of the integration of network activity at each time step. This is because, the output synapses are constantly active due to the continuous output function of the neurons. By contrast, the output of spiking neurons are discrete, rare events, whose transmission is affected by axonal latencies and which reach the target units at certain time moments. A simulation which discretizes the computation of unit states (e.g, activities) at spike-event occurrences seems more appropriate for a pulsed neural network simulation.

Event driven algorithm

An alternative to time driven simulation is to implement an event driven strategy (Ferscha and Tripathi, 1994). Instead of advancing the simulation time in a continuous way and processing events synchronously at each clock tick, the integration of a unit activity can be performed in an asynchronous way, triggered by the reception of one or several spike-events (Watts, 1994; Grassmann and Anlauf, 1998; Mattia and Del Giudice, 2000). In this section we describe Algorithm 5 that implements a basic event-driven strategy for the computation of the network activity during a training step.

Algorithm 5 PROPAGATE_Spike - event driven

```
Require: current pattern number $p$, simulation time $st$, time out $t_{out}$
 1: $ct = st$ {set the time clock}
 2: $SL = nul$ {init spike list}
 3: while $ct < t_{out}$ do {while current time less than time out}
 4:   if $SL = nul \lor ct \geq$ time(next_subpattern(p)) then
 5:     apply_inputs (next_subpattern(p)) {call lines 3-10 Algorithm 4}
 6:   end if
 7:   $e$ = first_event($SL$)
 8:   $ct = e.time$ {simulation time becomes the time of the current spike }
 9:   if not nul_event($e$) then
10:     $i = e.target$ {get the spike's target unit}
11:     $act^i$ = ACT_Spike($i$, $ct$) {integrate activation for target unit $i$}
12:     $out^i$ = OUT_Spike($act^i$)
13:     $SL$ = remove_first_event($e, SL$) {delete event $e$ from the list}
14:     if $out^i$ then {propagate spike to postsynaptic nodes}
15:       for all k $\in \Lambda_i$ do {all postsynaptic nodes to $i$}
16:         $d_i^k$ = noisy_delay(length($w_{ik}$))
17:         $e$ = create_spike_event($k, ct + d_i^k$)
18:         $SL$ = insert_event_order($e, SL$)
19:       end for
20:       insert($i$, LearningNodes) {synapses of firing units are subject to learning}
21:     end if
22:   end if
23: end while
```

The core of Algorithm 5 consists in processing spiking events from a chronologically ordered list SL (lines 7- 13). Whenever a unit fires a spike, a new event e is created and inserted in order in the spike list SL (lines 17-18). Each new spike is fully characterized by a time stamp representing the delivery moment and the index of the target unit. The input patterns are applied when the event list becomes empty or when the current time of the simulation exceeds next sub-pattern time stamp (lines 4-5).

The *simulation engine* of the event–driven algorithm is presented in Figure 5.2. It continuously takes the first event from the event list (i.e., the one with the lowest timestamp) and delivers it to the target unit. This process continues until a time out moment is reached or no further events have occurred. The most important task of the event-driven engine is to allow an asynchronous processing of the events, while preserving the time order of the events. Doing this, makes the parallelization of the simulation possible, because it permits events to be processed by different processors, while the time order is ensured (Mohraz et al., 1997; Jahnke et al., 1999). The most common procedure for achieving this goal is to maintain a global ordered list of events and whenever a new spike is generated, it is inserted in order in the list. Alternative methods have been proposed, mainly by creating and updating several lists instead of one, in order to reduce the excessive increase of a global list length (Mattia and Del Giudice, 1999; Claverol et al., 2002). In Section 5.2.2 it is presented our solution to this problem.

Evaluation of performances

The event-driven algorithm presented above leads to a significant decrease in the number of operations required computing the network activity during a training cycle. By using the efficient version of the activation function ACT_Spike2 in Algorithm 5, the simulation of the network activity in a time period T requires a number of operations:

$$NO_5 \approx \frac{T}{\sigma t} \cdot a \cdot N \cdot S \cdot (a + 1 + \log(length(SL))) \tag{5.5}$$

Here, σt represents the time resolution used in the generation of noisy delays and input signals (i.e., note that is different from Δt step). The first term represents the computational effort employed by the integration of all units that receive spikes at a certain moment of the simulation (lines 7-12, Algorithm 5). Note that it scales with a^2. Second term is the computational effort for the spike distribution (lines 14-17, Algorithm 5). Compared to the previous versions, this algorithm brings an additional computational effort required by the insertion in order of the new spikes in the event list (line 18, Algorithm 5). The length of the list scales with $T/\sigma t \cdot a \cdot N \cdot S$.

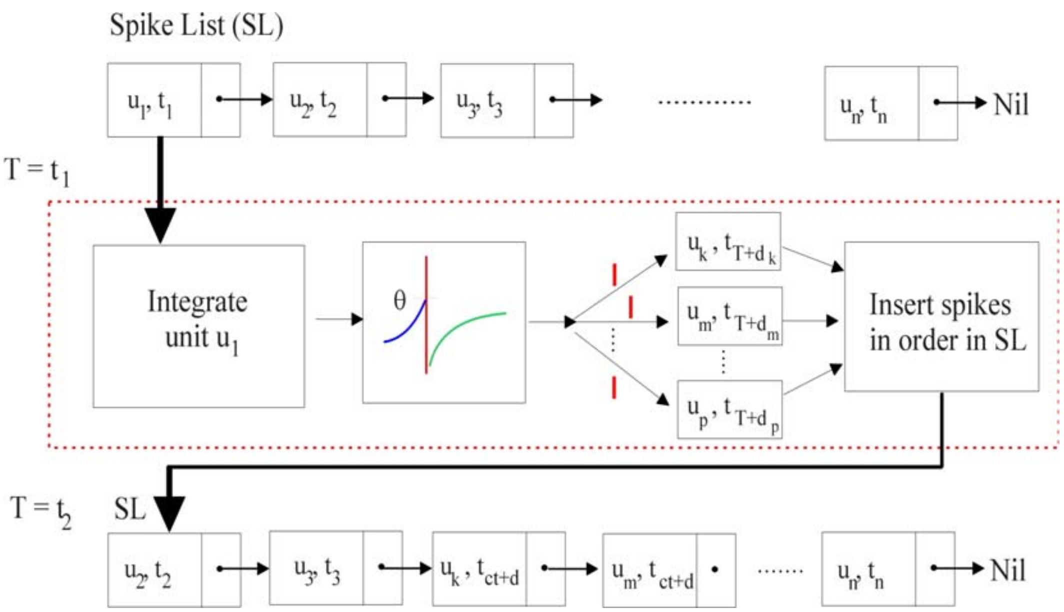


Figure 5.2: Event-driven simulation engine. At $t1$ the spike list SL contains events sorted in a chronological order. The engine takes the first spike in the list (i.e., the oldest) and delivers it to the target unit. This integrates its activity and possibly, emits a spike. The emitting unit computes the delays and creates the corresponding spike structures that are further inserted in order in SL. The process is repeated at time $t2$ with the new configuration of the spike list.

One can see that compared with Algorithm 4 (Eq 5.4) the number of operations on Algorithm 5 is reduced by two factors. First, in each time bin only a percent a of the total number of units in the network are computed. This causes a decrease in the number of operations in the performance critical part that concerns the integration of units activities, by a factor of a^2. Second, the occurrence of the integration time instants is not fixed to a time clock, but is given by the time resolution σt and the network activity a. The number of operations in Equation 5.5 can be further decreased in two ways. Firstly, by decreasing the percent of spikes which are computed in a time slice and secondly, by increasing the σt value. Both issues are discussed in Section 5.2.2.

In order to compare the performance of the event-driven and time-driven policy, the Algorithms 4 and 5 have been implemented to train a pulsed self-organizing network with plastic lateral synapses. Note that, in the remainder of this chapter, the continuous time Algorithm 4 implemented with the activation function ACT_Spike1 will be used as the basic-line measure. The network architecture and the learning procedure are described in detail in Section 6.1. In a nutshell, the organization process consists in the training a self-organizing

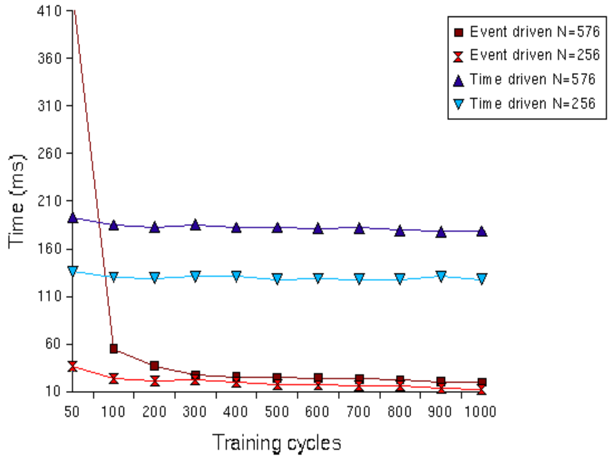


Figure 5.3: Computational times per training cycle for the event and time driven algorithms, for two network sizes N = 256, 576 units.

feature map of spiking neurons with a Mexican–Hat shape of lateral connectivity, to encode 12 directions of movement. Two networks have been used, of dimensions $N = 256$ and $N = 576$ units. The size of the test maps was limited by the low dimension of the set of input patterns (i.e., 12x20 patterns) which would cause the failure of the self-organization process in larger maps (see Section 6.1).

In a self–organization process, the lateral feedback system is used as a basic mechanism to modify over time the form of the emergent activity pattern. Given an untrained map, the neural activity starts out spreading over a large part of the network. That is in our case, up to 30%-50% of the network. In a few hundred iterations of the learning procedure, the network response to one stimulus converges to a stable activity bubble that includes a relatively small set of firing units. It becomes evident that an event-driven algorithm can benefit from the localization of the network activity, by updating units only within the active area. In contrast, a continuous time procedure computes the entire network activity in a time-stepped fashion.

Figure 5.3 shows the computational times per training cycle when the Algorithms 4 and 5 are used for the task described above. Results illustrate the good scaling of the event-driven algorithm with the simulation time, when the level of activity in the network (i.e., a value) decreases. In opposite, the time spent in the synchronous simulation (first two graphics in Figure 5.3) scales poorly with the reduction in the network activity, but it is strongly correlated with the network dimension. The scaling of event-driven algorithm is

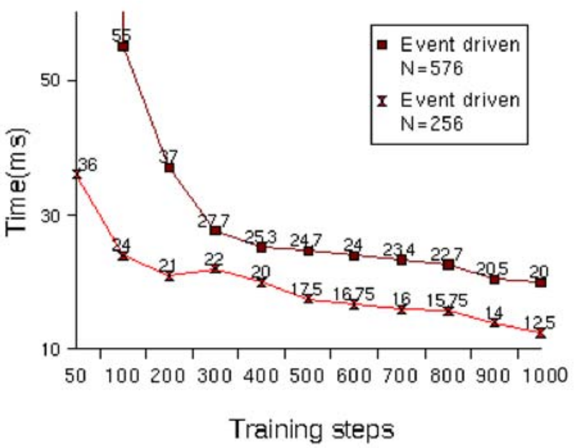


Figure 5.4: Scaling of the event-driven algorithm with the change in the network activity. Computational times per cycle are presented for first 1000 training steps, when activity decreases from 100 HZ (on first 100 steps) to 33 Hz at the end.

better illustrated with Figure 5.4. Note that the network activity decreases from 100 Hz on the first 100 cycles to 30 Hz after 1000 cycles.

These results suggest that the event-driven algorithm represents for a self-organization process with spiking neurons, a more efficient simulation framework than the time-driven approach. Time-efficiency represents a crucial issue in the simulation of self-organizing networks. That is, because the mapping accuracy of a Kohonen network depends upon the dimension of the training sample and the number of learning steps performed (Kohonen, 1995). Similarly, the simulation time of a pulsed SOM can increase significantly, unless time-efficient strategies are employed.

5.2.2 Optimization of the event–driven algorithm

On Figure 5.3 (left upper side) one notes that the event-driven algorithm applied to the training of the larger network (N= 576 units) performs more poorly than the continuous approach. This behavior is recorded during the first hundred learning cycles when the network exhibits neural activity with frequency up to 100 Hz. The inefficiency of the event-driven approach in the conditions of high neural activity arises from a poor management of the spike–event list. The main factor is represented by the insertion operation that scales with the number of events in the list. The observation about the limitations of the event-driven strategy is consistent with those of other authors (Schoenauer et al., 1998; Mattia and Del

Giudice, 2000). They have suggested that an event-driven simulation is suitable only for spiking neural networks with low activity. Most generally, the real neurons fire at low frequencies, however it is also common that they exhibit bursts of spikes with a high frequency of oscillation (Kandel et al., 2000). Therefore, we need to be able to simulate this type of neural behavior, without a tremendous increase of the simulation time. This section proposes an event-driven algorithm that scales well with the increase in the network activity.

In this section, two mechanisms are proposed for the efficient simulation of networks that exhibit activity patterns with high frequencies. Both of them address the most time-expensive process in the event-driven policy, namely the management of the event list.

Multiple spikes. A straightforward implementation of an asynchronous algorithm generates for aN firing neurons with S synapses per neuron a maximum number of $aN \cdot S$ action potentials in each time slice (Equation 5.5). Our idea was that, instead of creating a distinct event for each spike, to accumulate in a single structure all presynaptic spikes that would reach the target unit at the same time moment and to deliver them together. We define the concept of a $multiple\ spike$, which is a data structure that stores in a list, all synaptic weights m that deliver a spike to a certain neuron i at the time moment t. A similar concept was previously formulated by Schoenauer et al. (1998) and defined as weight caching.

By implementing the multiple spike concept, the overall computational load per time slice reduces by $1/m$. This means that we have fewer spikes to integrate less to distribute and less to insert in the spike list. In the most favorable scenario, m can equal S. Consequently, the time of the simulation scales well with the increase in the network activity. On the worst case, m can represent just a low percentage from S. In this case, the method does not bring a significant improvement on the time performance of the algorithm. The main parameters which affect the value of m are: the topology of local connections and the time resolution used in the generation of input signals and delay values σt. A possible trade-off might be to use a large time resolution (e.g., $\sigma t = 1$ ms). This will increase the probability of spikes to be delivered at the same time, hence it will favor their aggregation in multiple spikes and will decrease the length of the event list.

The ordered-insertion problem. The use of multiple spikes for the aggregation of pulses to be delivered at the same time to a target unit is aimed at reducing the length of the event list. However, there are cases when trying to aggregate spikes does not make any difference and there is no significant reduction of the length of the list. This faces us with the problem of how large dynamic data structures can be managed efficiently. It was mentioned above, that the most expensive operation in the management of the event structure is represented by the ordered insertion (line 18 in Algorithm 5). That is, because the complexity of this operation scales with the length of the list. As the list grows (i.e., in the range of 10^5 events for a frequency of 100 Hz), so does the search time required by the ordered insertion. A basic improvement, which results in speeding the simulation up to four times, can be achieved if the list is searched selectively from the start or from the end, depending on the new spike time stamp.

Several implementations of event-driven strategies for the simulation of pulsed neural networks have been developed in recent years (Watts, 1994; Mattia and Del Giudice, 2000; Claverol et al., 2002; Delorme and Thorpe, in press). Perhaps one of the most efficient (and ingenious) solutions for the management of the event structure was proposed in Mattia and Del Giudice (2000). Authors suggested the use of not only one event list, but of several FIFO queues, each of them associated to a fixed axonal delay value. The neural model accounts for the existence of a discrete set D of ordered delays for spike transmission (e.g., a maximum number of 16 delays have been implemented). Synapses of the neurons in the network are organized in matrix-structured layers, each layer corresponding to one delay value. When an event is generated it is not inserted in a single global list, but it is directed to the synaptic queue corresponding to its transmission delay. Because in the same queue all spikes share the same transmission delay, the spike generated first will be the oldest (top of the queue) and the latest generated spike will be the last in the queue (end of the queue). Accordingly, the data structure needs no sorting operation and the insertion is done in $O(1)$ complexity.

However, the picture is completely different when a neural model accounts for noisy delays. That is, because a recently generated spike with an associated high transmission delay must be delivered to the target unit *later* than an older spike that has a short synaptic delay. In this case, the time stamp of an event is dependent on both the spike time and the noisy delay. The algorithm proposed by Mattia and Del Giudice (2000) has a very low complexity when used for learning with plastic synapses and theoretically, the solution suggested is very ingenious and highly efficient. Its drawback is that it employs a crucial simplification of the neural model, consisting in the limitation of the synaptic delay to a fixed number of values.

Formal neural models apply several simplifications to the detailed structure of the biological neuron (see Section 4.2.1), but noise in the synaptic transmission is preserved by almost all simplified models (Gerstner, 1999; Mass, 1999). This is, because noisy delays are an essential factor in the optimal tuning of the neural response to the stimulus attributes, hence they play a crucial role in the development of neural selectivity and in the self-organization of the cortex (see Section 6.1). We consider that keeping the noisy delays is a *must* for a simplified neural model and the solution proposed in Mattia and Del Giudice is acceptable only for a limited set of applications, where noisiness can be discarded.

Quick sorting of an unordered pool. Our solution to an efficient management of the spike list structure consists in eliminating the insertion overhead. Instead of performing the ordered insertion, we just add the spike to an unordered pool of spikes, an operation of complexity $O(1)$. Since the events have to be processed in chronological order, at constant time intervals, the simulation engine stops processing spikes, and selects from the pool and sorts chronologically, those events which will be computed in the next interval. The time window during which processing of events takes place continuously is referred to as T_{window}. It is similar to the *safe window* concept used in parallel simulations to guarantee the temporal correctness of the algorithm (Ferscha and Tripathi, 1994). Figure 5.5 shows the simulation engine of the algorithm.

The selection of spikes to be processed in the next time window is performed using a *quick sort* algorithm. Most importantly, the sorting algorithm is run only over a small percentage q from the elements in the pool, namely those whose time mark is within the next processing interval (see Figure 5.5). This is realized by setting the first pivot point of the quick sort procedure to $t + T_{\text{window}}$. By doing this, after the first iteration of the algorithm the pool is ”shuffled” and only the elements with a time stamp less than $t+T_{\text{window}}$ are sorted (lines 23-27 in Algorithm 6). As a consequence, instead of dealing with the total number of insertion operations proportional to

$$NO_6 \approx \frac{T}{\sigma t} \cdot a \cdot N \cdot S \cdot \log(\frac{T}{\sigma t} \cdot a \cdot N \cdot S) \tag{5.6}$$

we have an insertion operation of complexity $O(1)$ and an additional computational effort to sort the pool given by

$$NO_7 \approx \frac{T}{T_{\text{window}}} \cdot q \cdot N_p \cdot \log(q \cdot N_p). \tag{5.7}$$

where N_p is the number of spikes in the pool. The value of the interval T_{window} is choose by considering two aspects. On one hand, the value of this variable is set so that it ensures a correct chronological processing of the spike-events from the pool. To obtain this, each processing interval cannot be larger than the minimum synaptic transmission delay. As the value T_{window} grows, so does the risk of integrating more recent spikes before older ones.

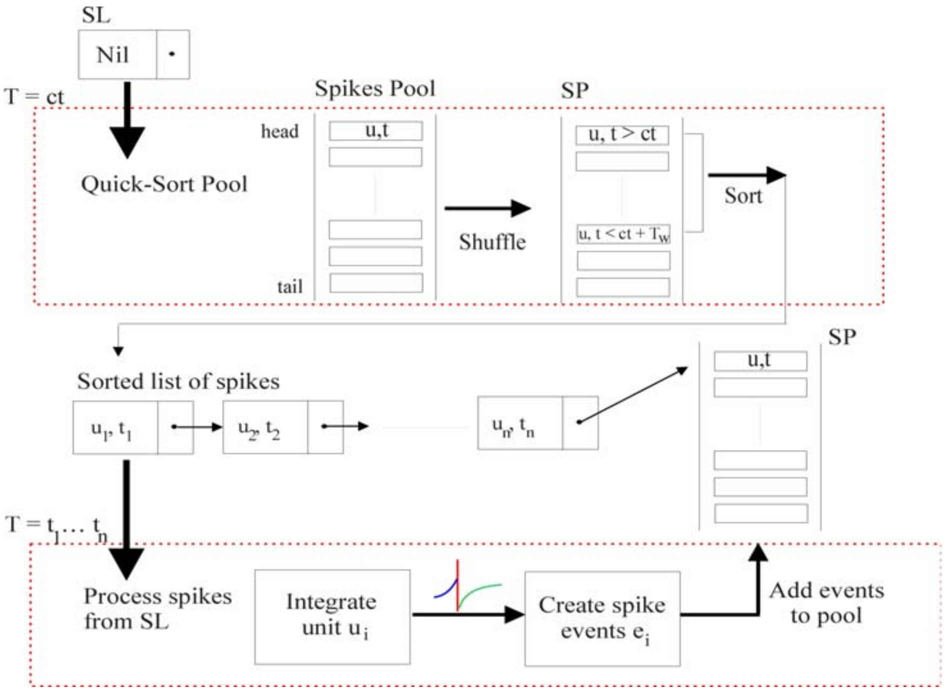


Figure 5.5: The event-driven simulation engine based on the quick-sort strategy of an unordered pool. Whenever the spike list SL becomes null, that is, when all spikes in the current time window T_{window} have been processed, the quick-sort algorithm is run. Note that it is run only upon a fraction of the elements in the spikes pool, namely those with a time stamp between the current time t and $t + T_{\text{window}}$. Sorted spikes are stored in SL and are processed by the simulation engine (i.e., integration, propagation, output). Any new spikes generated are simply added at the end of the pool.

On the other hand, too fine a value of T_{window} increases the chances that the sort algorithm will fail. That is, if the sorting interval is too small, the algorithm does not find not even a single event or it returns a very small number of events with time stamps within this interval (see lines 23-27 in Algorithm 6). Moreover, a large time window is desirable because it decreases the number of times when sorting is required. The implementation of the quick-sorting strategy and of the multiple spike concept, give the final version of the PROPAGATE_Spike function in Algorithm 6.

The implementation of Algorithm 6 for the simulation of a number of cognitive models indicates that it represents an efficient solution for networks with high frequencies of neural activity and when the synaptic delays have equally distributed values. In our simulations (Sections 6.1, 6.2), the T_{window} interval was set to 2 ms and σt to 0.1 ms. This means that the insertion operation complexity in Equation 5.6 is 20 times higher in comparison with the sorting effort from Equation 5.7. Moreover, the value of the percent p is given by the number

Algorithm 6 PROPAGATE_Spike - quick sorting strategy

```
Require: current pattern number $p$, simulation time $st$, time out $t_{out}$
$ct = st$ {set the time clock}
$PL = nul, SL = head(PL)$ {SL points to the head of the spike pool}
while $ct < t_{out}$ do {while current time less than time out}
  if $SL$ = $nul$ $\vee$ $ct \geq$ time(next_subpattern(p)) then
    apply_inputs (next_subpattern(p)) {call lines 3-10 Algorithm 4}
  end if
  $e$ = first_event($SL$)
  $ct = e.time$ {simulation time becomes the time of the current spike }
  if not nul_event($e$) then
    $i = e.target$ {get the spike's target unit}
    $act^i$ = ACT_Spike($i, ct$) {integrate activation of unit $i$}
    $out^i$ = OUT_Spike($act^i$)
    $SL$ = remove_first_event($e, SL$)
    if $out^i$ then {propagate spike to postsynaptic nodes}
      for all k $\in \Lambda_i$ do {all nodes postsynaptic to $i$}
        $d_i^k$ = noisy_delay(length($w_{ik}$))
        $e^m$ = multiple_spike($k, ct + d_i^k$) {create or update a multiple spike}
        $PL$ = add_event_pool($e^m, PL$)
      end for
      insert ($i$, LearningNodes) {synapses of firing units are subject to learning}
    end if
  else if $PL \neq$ nil then {if SL is nul and spike pool PL is not empty}
    $T_{\text{pivot}} = ct + T_{\text{window}}$ {get the next processing window}
    $PL\,[center] = T_{\text{pivot}}$ {set the first pivot point for quick sorting}
    while $SL = nul \wedge T_{\text{window}} < time\_out$ do
      $SL$ = sort_pool($PL, T_{\text{pivot}}$) {sort spikes with time less than $T_{\text{pivot}}$}
      $T_{\text{pivot}} = ct + T_{\text{window}}$ {increase the window}
    end while
  end if
end while
```

of spikes to be sorted within a 2 ms interval, hence it is low compared to the total number of spikes in the pool. Together, these factors make the sorting computational effort to remain low and almost independent of the pool size, N_p.

Moreover, we can complete now the implementation of the activation function ACT_Spike2 (Section 5.1.3) by specifying the way a neuron integrates the presynaptic spikes. It was pointed out there, that without a receiver-oriented connectivity each spike must store the complete information needed for the computation of the weighted postsynaptic potentials by the target unit. In the event-driven Algorithm 6, the *Activation* function is called whenever a neuron receives a multiple spike. At that very moment, when the spike reaches the unit, the postsynaptic potential (PSP) is set to the maximum value (e.g., 1 in SpikeNNS). Therefore, no other computations are needed, excepting to multiply the PSP by the connection weight, which is stored in the multiple spike structure.

With these clarifications, lines $3-6$ in function ACT_Spike2 become:

```
for all $w_{ji}^k \in$ MultipleSpikeStructure do
    $PSP^k(t) = 1$
    $dPSP_i(t) += PSP^k(t) \cdot w_{ij}^k$
end for
```

Evaluation

We conclude the description of the event-driven algorithm with the evaluation of its time-efficiency. First, we want to illustrate the gradual improvement in the performance of a basic event-driven implementation when each of the above methods is incorporated. We compare the version presented in Algorithm 4 with three improved algorithms: (1) the implementation of multiple-spikes at a time resolution $\sigma t = 0.1$ms; (2) multiple-spikes at time resolution $\sigma t = 1$ms; and (3) the quick-sort algorithm with multiple-spikes at $\sigma t = 0.1$ms (i.e., Algorithm 6). The performances of the algorithms were compared with respect to how well each of them realize the management of the spike-events structure. Our measure of the computational effort is defined by the time required to compute (e.g., integrate activities, output spikes, propagate pulses and handle the spike-list structure) a certain number of units, in this case $N_k = 1000$ units.

The findings presented in Figure 5.6 reveal a gradual increase in the algorithm performance of up to 20 times when the above strategies are added one by one to the basic event-driven implementation. Note that if the multiple spike strategy is applied to a series of events generated with a fine time resolution (i.e., 0.1 ms) the probability of spikes to accumulate is low. Consequently, it leads to an average improvement in performance by 30%. Only when the time resolution is increased to 1 ms does the method prove really efficient. The best performing algorithm consists of a combination of multiple spike strategy and quick-sorting of the unordered pool. It shows an almost linear scaling with the increase in the network activity and most importantly, this is obtained at a time resolution of 0.1 ms. The complexity of Algorithm 6 is shown in Table 5.1.

In the second part of the evaluation, we intend to compare the time performance of Algo-

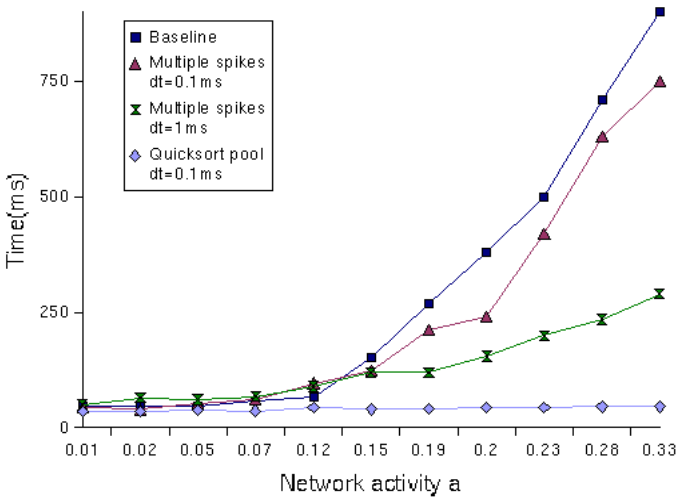


Figure 5.6: Computation times for updating 1000 units vs. levels of network activity, when different event handling methods are applied. The network activity is measured as the average number of spikes in one ms divided by the total number of neurons for N=576.

rithm 6 with the times reported in Mattia and Del Giudice (2000) (i.e., further referred in the text as the layered-delay algorithm). This comparison is facilitated by the fact that both algorithms have been applied to spike-driven learning processes of similar complexities. The quick-sorting strategy has been implemented to apply learning in a self-organization feature map of spiking neurons, with plastic excitatory and inhibitory lateral synapses (see the description of the learning procedure in Section 6.1). The learning procedure was run only for a small number of cycles while the network activity was high (i.e., 100 Hz).

Results reported for the layered-delay algorithm are the time per neuron needed to complete the simulation of one second of neural time in networks with different sizes (i.e., up to 15k units). The algorithm is applied for learning with integrate-and-fire neurons, when the spiking frequencies are maintained constant. Thus, the excitatory neurons frequency is $F_e = 2$ Hz and inhibitory neurons frequency $F_i = 4$ Hz. The connectivity rate, when the network size is varied, represents a constant percent 10% from the total number of units.

To obtain a similar measure of the computational effort required to integrate one neuron activity in our algorithm we proceed as follows. First, we calculate the number of neurons which fire during 1s of simulated time. This is given by $N_f = N_e \cdot F_e + N_i \cdot F_i$, where N_e, N_i are the number of excitatory, and respectively inhibitory neurons, in a network of a given size. Next, we run the Algorithm 6 on a set of networks with similar sizes and connectivities

(1) Continuous time v1.	$NO \approx \frac{T}{\Delta t} \cdot N \cdot S \cdot (\lvert F_j \rvert + a)$, with a $\in [0,1]$
(2) Continuous time v2	$NO \approx \frac{T}{\Delta t} \cdot N \cdot S \cdot a \cdot 2$
(3) Event driven - basic	$NO \approx \frac{T}{\sigma t} \cdot N \cdot S \cdot a \cdot (a + 1 + \log(\frac{T}{\sigma t} \cdot N \cdot S \cdot a))$
(4) Event driven - optimal	$NO \approx \frac{T}{\sigma t} \cdot N \cdot \frac{S}{m} \cdot a \cdot (a+1) + \frac{T}{T_{\text{window}}} \cdot q \cdot N_p \cdot \log(q \cdot N_p))$ with $m \in [1, S]$, $q \in [0.05, 0.25]$ and $N_p = \frac{T}{\sigma t} \cdot N \cdot \frac{S}{m} \cdot a$

Table 5.1: Computational effort for four algorithms, estimated by the number of operations required in the implementation of the performance critical parts: (1) continuous time algorithm (Algorithm 4, Section 5.2.1) with activation function version 1 (Algorithm 1, Section 5.1.2); (2) continuous time algorithm (Algorithm 4) with improved activation function version 2 (Algorithm 3, Section 5.1.3); (3) basic event-driven algorithm (Algorithm 5, Section 5.2.1); (4) time-efficient version of the event driven algorithm, with multiple spikes and quick-sorting of the pool (Algorithm 6, Section 5.2.2).

as the one tested in Mattia and Del Giudice (2000). During training of these networks, we compute the time per neuron required for the simulation of all operations entailed by the firing of same number of neurons, N_f. The reason for this measure of evaluation is that the two algorithms run on networks that show activities with significantly different frequencies, hence the computational efforts per time slice cannot be compared. The results of the comparison are presented in Table 5.2.

The main strength of the Mattia and Del Giudice algorithm resides in the layered structure of 4 up to 16 delay values. This was designed particularly for computational simplicity, even if it was done at the expense of the biological plausibility. Moreover, the simulation time per neuron obtained with the layered-delay algorithm (Table 5.2) is based on an average spiking frequency of 2.5 Hz. This means that in one second of simulation time, each neuron fires less than three times. In the case of the quick-sorting algorithm, time per neuron was computed when a number of N_f neurons fired at a frequency of 100 Hz. The difference in the frequencies is essential when comparing the algorithms. For instance, in the case of a network with $N = 4k$ units a spiking frequency of 100 Hz generates a pool of events of the order of $220,000$ elements. Hence, our algorithm manages to keep the simulation time approximately twice as long as the layered-delay algorithm with 30 times higher frequency. And most importantly, it still manages to preserve the essentials of neural behavior (i.e.,

N (1k units)	**N$_f$** (1k units)	**Layered delays** (ms) 2.5 Hz	**QuickSort pool** (ms) 100 Hz
0.5	1.0	0.45	1.0
1	2.4	0.8	2.5
2	5.0	1.4	3.1
3	7.0	2	4.3
4	9.5	2.7	5.2
5	12.0	3.2	6.3

Table 5.2: Execution times per neuron vs. size of the network N, when N$_f$ neurons fire. For the layered–delays algorithm we refer to the Mattia and Del Giudice (2000) algorithm. Note that the average firing rate in the layered delays simulation is 2.5 Hz, whereas the execution times for the Quick-Sorting algorithm are recorded for a neural activity of approx. 100 Hz.

noisy synaptic transmission).

5.2.3 Learning framework

So far in this section, we described the implementation of the PROPAGATE_ACT function that is used to apply and propagate a time-coded input pattern through the network. The final task is to implement the training function LEARN_Spike that specifies how the network learns a complete set of input patterns.

The training procedure consists of a loop that executes a series of actions. It takes a new input pattern from the set (lines 2, 12 Algorithm 7) and it applies it to the network. It propagates the network activity until a reference time $time_out$ is reached or until there are no more spikes to process (see PROPAGATE_ACT in Section 5.2.1). Then, it updates the synapses of those neurons selected for learning during the training step (i.e., the firing neurons). Learning is applied with a frequency L_f, that can be 1 or any number up to the training set dimension. Before the beginning of a new training step, the network activity can be reset or preserved, as a function of the parameter a_f. In the simulations described in chapter 6, the network activity was reset during training after the presentation of each input pattern, and it was preserved in the testing phase.

The Learning_Function (line 7 in Algorithm 7) can implement any type of synaptic adaptation rule. Several rules for learning with spiking neural networks have been described

Algorithm 7 LEARN_Spike

```
Require: pattern set $P$, learning frequency $L_f$
setup_simulation_times ($t_{start}, t_{out}, t_{int}, ct$)
$cp$ = first_pattern($P$) {get first pattern}
while not_nul ($cp$) do {for all input patterns}
   PROPAGATE_ACT($cp$)
   if $number(cp) \% L_f = 0$ then {time to apply learning }
      for all $i \in$ LearningNodes do
         Learning_Function($i$) {apply learning to the synapses of neuron $i$ }
      end for
   end if
   reset_network_activity($a_f$) {reset or keep neurons activities}
   update_simulation_times ($t_{out}, t_{int}, ct$)
   $cp$ = next_pattern($P$) {get next pattern}
end while
```

in Section 4.4. At the present, the existent learning rules in SpikeNNS implement a self-organization learning framework used for the simulation of the cognitive models described in Sections 6.1 and 6.2. Hebbian and anti-Hebbian plasticity for excitatory synapses and a type of correlation-based learning for the inhibitory synapses have been also implemented (see Section 6.2). Future work is aimed at extending the number of learning functions with supervised rules and other forms of spike-timing dependent synaptic plasticity.

5.3 Description of simulator features

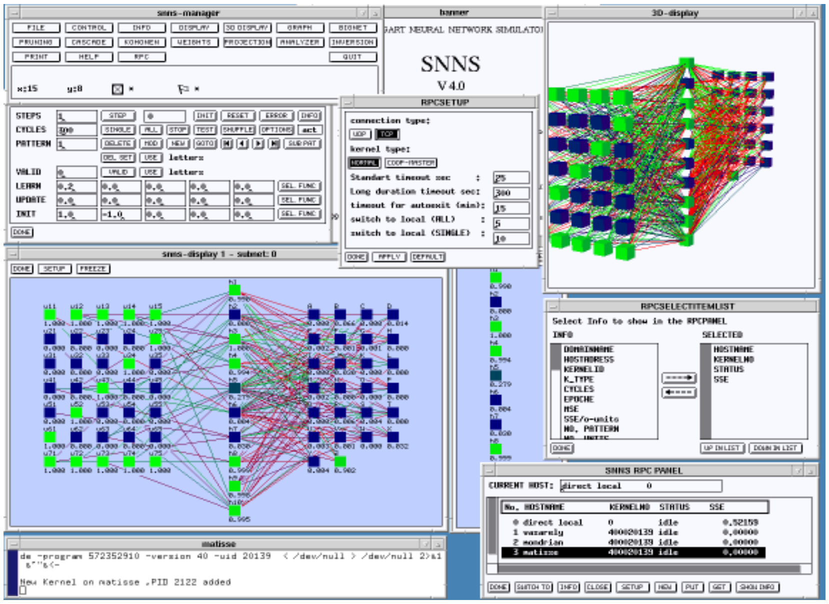
5.3.1 Stuttgart Neural Network Simulator

The scope of the modeling work carried out in this thesis is to provide an illustration of how a cognitive phenomena, such as visuomotor mapping or movement planning, can be grounded at the neural level. In Section 4.3 it was argued that a simple, rather than a detailed neural model, is more appropriate for our modeling goals. We have described the implementation of a neural model, namely the Spike Response Model (Gerstner, 1999). The reasons we did not use a preexisting simulator for pulsed neural networks and choose to build our own implementation are twofold. Firstly, the most powerful and user-friendly simulators for spiking neurons existent today are based on detailed models of the neuron structure: Neuron (Hines and Carnevale, 1995), GENESIS (Bower and Beeman, 1998). This makes them less suitable for the simulation of large-scale cognitive phenomena. Secondly, freely available simulators of simplified neural models are restricted as applicability, since they have been created for specific modeling issues and run on certain hardware platforms. For instance, INFERNET (Sougné, 1999) is a simulation framework created particularly to explore cognitive binding and inference, and which works on Macintosh machines.

Therefore, it is rather difficult to adapt a simulation environment created to support the development of a family of models in one domain to another domain study. What seemed to be the best solution was to use a general-purpose simulator that can provide good visualization facilities and the possibility of code re-use. It was decided to take a simulation environment designed for the traditional, continuous valued neurons and to extend it to deal with the more biologically realistic spiking neurons. The simulator chosen was the Stuttgart Neural Network Simulator (available at http://www-ra.informatik.uni-tuebingen.de/SNNS/).

Stuttgart Neural Network Simulator (SNNS) is a software simulator, currently available for Unix and Windows platforms, developed since 1990 at the Institute for Parallel and Distributed High Performance Systems (IPVR) at the University of Stuttgart (Zell et al., 1992). It supports arbitrary network topologies, it is highly configurable and includes a relatively large number of learning procedures, starting with backpropagation algorithms, ART maps, Kohonen networks and ending up with time-delay and recurrent networks. The graphical user interface (Figure 5.7) offers a 2D/3D representation of the neural networks and allows an user-friendly control of the kernel during the simulation run. The sources of the implementation in C for Unix platform are freely available, and can be easily extended with user-defined libraries. SNNS is a widely distributed neural network simulator and its use and extension is technically supported by the SNNS team and by the SNNS discussion mailing list.

The advantages of using the SNNS framework for the development of our pulsed neural network libraries were twofold. First, we benefit from the substantial functionality of the simulator and second, we can make the modeling extensions publicly available to the substantial SNNS user community. However, there are also disadvantages when adapting a simulator built for processing with rate coding neurons for the simulation of spiking neurons behavior. Some of the aspects of how an efficient simulation of pulsed neural network differs from a traditional neural network simulation have been emphasized in Section 5.2.1. The main design considerations regard the type of the connectivity scheme (i.e., sender-oriented vs. receiver-oriented) and the type of simulation engine (i.e., synchronous vs. asynchronous). The first implementation of the Spike Response Model using the SNNS data structures in-



x:15 y:8

STEPS 1
CYCLES 100
PATTERN 1
letters
VALID 0
letters
LEARN 0.2 0.0 0.0 0.0 0.0
UPDATE 0.0 0.0 0.0 0.0 0.0
INIT 1.0 -1.0 0.0 0.0 0.0

connection type: TCP
kernel type: NORMAL
Standart timeout sec : 25
Long duration timeout sec: 300
timeout for autoexit (min): 15
switch to local (ALL) : 5
switch to local (SINGLE) : 10

Select Info to show in the RPCPANEL
INFO: DOMAINNAME, HOSTADRESS, KERNELID, K_TYPE, CYCLES, EPOCHE, MSE, SSE/o-units, NO. PATTERN
SELECTED: HOSTNAME, KERNELNO, STATUS, SSE

CURRENT HOST: direct local 0

```
No. HOSTNAME        KERNELNO  STATUS    SSE
 0 direct local     0         idle      0.52159
 1 vasarely         400020139 idle      0.00000
 2 mondrian         400020139 idle      0.00000
 3 matisse          400020139 idle      0.00000
```

```
de -program 572352910 -version 40 -uid 20139  < /dev/null > /dev/null 2>&1
 &"'&<-

New Kernel on matisse .PID 2122 added
```

Figure 5.7: User interface (XGUI) of the Stuttgart Neural Network Simulator.

dicated that the original kernel libraries are not very appropriate for the simulation of spiking neurons (see Sections 5.1.2, 5.2.1). Consequently, we had to re-engineer a number of functions in the SNNS kernel and create dedicated data structures suited to the specifics of computing with spiking neurons. Another aspect that we had to consider in the simulator design, was the compatibility of the spiking neural functions with the SNNS own functions. Thus, the names, parameters, return codes or functionality of the new procedures have been designed in accord with SNNS conventions. At some point, this aspect constrained the way functions were built.

The extension of the SNNS environment to support simulations with spiking neurons was named SpikeNNS. This extension needed the re-engineering of some parts of the kernel, as well as the addition of dedicated functions, in order to allow computation with spiking neurons. The library of spiking neurons functions consists of: the implementation of the Spike Response Model (SRM), the event-driven simulation engine, functions for the application of time-coded input patterns, routines for learning with spiking neurons. Additional functions are required for initialization, testing and gathering firing statistics. The sources of SpikeNNS simulator will be freely available at http://cortex.cs.may.ie/tools/SpikeNNS.html.

5.3.2 Setting up a simulation with SpikeNNS

The process of setting up a simulation in SpikeNNS involves a number of steps:

- First, the specification of a prototype of the basic computational unit is required. The parameters of the neural model are specified from the SNNS configuration file. They are discussed below.
- Next, the connectivity of the network is created. This can be done either using the graphical interface available in SNNS for building networks, or by generating it with SpikeNNS functions. Parameters for the configuration of three types of a connectivity are given in Table 5.4 and described below.
- Finally, an initialization procedure is run to set the specific values of the spiking neurons simulation.

Initialization of neural parameters

The behavior of a spiking neuron is described by Equations 4.2, 4.3, and 4.5 (Section 4.3) and depends on a set of parameters, which can be defined by the user. These parameters are enumerated in Table 5.3 and have been described in detail in Section 4.3. The initialization of these values and other spike-specific parameters is done with the procedure INIT_Spike. At initialization, each hidden neuron in the network is assigned a noisy threshold computed by the formula:

$$\Theta = max\Theta - R(\Delta\Theta), \tag{5.8}$$

where R is a function that returns a random number uniformly distributed between 0 and $\Delta\Theta$. When a spike is emitted on a synapse, a noisy delay is generated as a function of the synapse length l:

$$d = l + G(\Delta d), \tag{5.9}$$

where l is given by the Euclidean distance between the connected neurons and G is a Gaussian distribution with mean 0 and standard deviation Δd. The firing time of a neuron is affected by noise with Gaussian distribution with mean 0 and Δf standard deviation, according with the formula:

$$t^{f'} = t^f + G(\Delta f). \tag{5.10}$$

Neural parameters
$max\Theta$ - maximum neural threshold
$\Delta\Theta$ - threshold spread value
τ_m - membrane time constant
u, v - refractory period parameters
F_j - number of spikes stored / neuron
Δd - noise factor in the generation of synaptic delays
Δf - noise in the firing time value
N_e - percent of excitatory neurons
c_e, c_i - connectivity rates of excitatory and inhibitory neurons
Δc - random factor in connections number

Table 5.3: The configurable parameters of the neural model.

At initialization, each unit is probabilistically defined as either excitatory or inhibitory, so that an average number N_e of excitatory neurons will exist in the network. If the users chooses to implement a sparse connectivity pattern (see Section 5.3.2), than the number of lateral connections per neuron is specified by the formula

$$c = C \cdot N + G(\Delta c), \tag{5.11}$$

where N is the total number of neurons in the network and $C = c_e \vee c_i$ is the rate of connectivity per excitatory, or inhibitory neuron respectively. The Gaussian distribution has the mean 0 and standard deviation Δc.

Creation of a connectivity pattern

SNNS is a simulation environment designed for the modeling of classical neural networks. Hence, it does not address in particular, the problem of creating biologically realistic connectivity patterns. Most readily, the network topology can be set up to full connectivity between any two layers or between any clusters of units, defined by their coordinates (x, y) in the network. By contrast, the topologies of biologically inspired networks usually are based on sparse probabilistic connections or regular synaptic patterns (see below). That is, because in the cerebral cortex each neuron is coupled to a reduced number of other neurons, in a non-random fashion (Braitenberg and Schuz, 1998). Cortical synapses within the same layer most commonly connect cells with similar neural response, while projections between

Type	**Connectivity parameters**
Sparse intra-layer connectivity	n - target layer $max\, w$, Δw - maximum weight and deviance value σ_e, σ_i - connectivity decay parameters
Topographical inter-layers projections	s, t - source and target layer $max\, w$, Δw - maximum weight and deviance value L_p - sets the size of projection for topographical connections bi - sets on the bidirectional connectivity between s and t
Receptive afferent fields	i, t - input and target layer $max\, w$, Δw - maximum weight and deviance value $d_i, \Delta d$ - delay and deviance value for input connections R_f - sets the radius of the receptive field m - sets the value of the magnification factor

Table 5.4: The configurable parameters for three types of connectivity: intra–layer, inter–layers, and receptive fields.

layers convey information from one stage to another in a topographical manner (Kandel et al., 2000; see also Section 2.2.4).

SpikeNNS offers a number of configurable functions, which can be used to generate three types of connectivity patterns, independently of the neural model used (see parameters in Table 5.4):

- *Intra-layer sparse connectivity*, with a short-range distribution of the excitatory connections and long-range distribution of inhibitory synapses. There is experimental evidence that a Mexican-Hat shape of connectivity with inhibition surrounding excitation is biologically plausible and it favors the process of lateral and afferent weights self-organization (see Section 2.2.4).
- *Inter-layers topographical projections*. In the cerebral cortex sets of neurons which convey information from one processing stage to another are usually projecting in a topographical manner (see the retina-LGN-V1 projections, Kandel et al., 2000).
- *Afferent receptive fields*, which convey signals from a distinct area of the input space to a cortical (e.g., hidden) neuron (Kandel et al., 2000).

In SpikeNNS, connections are defined by their weight w and the synaptic length l that determines the value of the spike transmission delay (see Equation 5.9). The synapse weight is computed by the formula:

$$w = max\, w - R(\Delta w), \tag{5.12}$$

where R is a function that returns a random number uniformly distributed between 0 and Δw.

Sparse connectivity. To create within a target layer n, a sparse, centered-surround connectivity pattern, a number of steps are involved (see parameters in Table 5.4). First, for every neuron i in n, the probability of being connected with any other neuron j in the same layer is computed. The probability to create a synapse depends on the sign of the source neuron (i.e., excitatory or inhibitory neuron), the distance to the target neuron and the connectivity decay parameter. Thus, an excitatory synapse from neuron i to neuron j has the creation probability of:

$$p = e^{\frac{-d[u_i, u_j]}{\sigma_e}}, \tag{5.13}$$

where $d[u_i, u_j]$ is the Euclidean distance between the units and σ_e is a constant which controls how fast the connectivity decays with the distance in the layer (formula adapted after Ström, 1997). An inhibitory connection is added with a probability of

$$p = 1 - e^{\frac{-\sigma_i}{d[u_i, u_j]}}. \tag{5.14}$$

After the probabilities of all possible output connections of the neuron i have been generated, these values are sorted in a descending order in an array. The array is then parsed starting from the highest values and an output synapse is created if the associated probability is less than a random number r. The value of this number is uniformly distributed within an interval, whose limits are set differently, depending on the sign of the synapse $r = base_{i,e} - std.dev_{i,e}$. Synapses are added to the network topology until the number of connections per neuron c (Equation 5.11) is reached.

Figure 5.8 shows for two target units at locations (4,12) and (13,6) in the network, the presynaptic units in the layer. The excitatory units (shown in blue) are distributed (probabilistically) within an area of dimension four surrounding the target unit. The long-range inhibitory synapses are received from units located at distances greater then four, up to the network margins. The percent of excitatory units in the network is of 80%. Parameters used in the generation of this pattern, were $\sigma_e = 4$, $\sigma_i = 3.5$. Random values were generated using $base_e = 1$ and $std.dev_e = 0.3$ for excitatory synapses, and $base_i = 0.7$ and $std.dev_i = 0.4$ for inhibitory synapses.

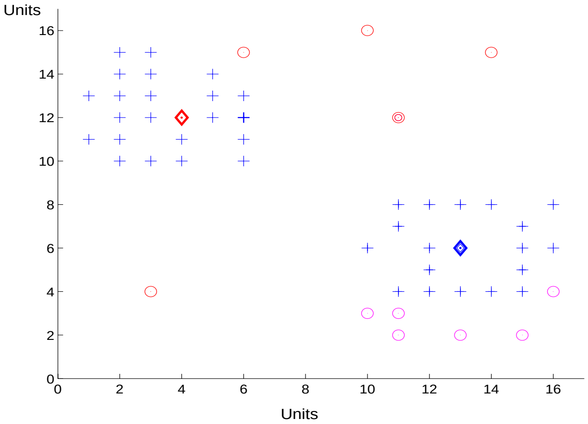


Figure 5.8: Sparse probabilistic connectivity pattern in SpikeNNS, which a short-range distribution of excitation and long-range distribution of inhibition. For two neurons, at locations (4,12) and (13,6) the input units are shown. The blue signs designate excitatory presynaptic units. The inhibitory units are shown in red (or magenta). The connectivity decay parameter is $\sigma = 4$.

Topographical projections. In SpikeNNS, topographical projections can be created only between hidden layers (see parameters in Table 5.4). Each neuron in the source layer s projects in the target layer t in a sub-array of units of size L_p. Note that current implementation of the function works only for dimensions of the target layer that are equals or larger than the source layer dimension. If bi parameter is set on (i..e, $bi = 1$), than reciprocal projections between layers s, t are created.

Receptive fields. Receptive fields can be created from an input layer i to any hidden layer t (see Table 5.4 for parameters). In such a connectivity scheme, a hidden unit of coordinates (x, y) receives stimulation from a square area of size R_f that is centered in the input layer (approximately) at $(\frac{x}{\sqrt{m}}, \frac{y}{\sqrt{m}})$. The value of the magnification factor m gives the ratio of hidden units per one input unit. That is, because the hidden layer has a higher dimension then that of the input layer and this allows the formation of several receptive fields for each location in the input array.

Implementation of the functions described above is independent of the type of the neural model specified (i.e., rate-coding or spiking neuron). Hence, they can be used to create network topologies, when any activation or output functions from SNNS are employed. Note that each of the above functions deletes any previously existing connectivity between the layers given as parameters. Therefore, they cannot be used in conjunction with the SNNS graphical interface for setting intra- or inter-layer connectivity. All parameters for the initialization of the neural model the setting of the connectivity patterns and for learning can specified in the SNNS configuration file. The SNNS control panel allows on-line modification of at most five parameters. When a particular function is loaded, such as ACT_Spike, LEARN_Spike or INIT_Spike, all its parameters are initialized from the configuration file. Only the first five values are displayed and can be updated through the control panel.

5.3.3 Learning parameters

As noted above, the parameters for learning are set in the configuration file and initialized during the call of the function INIT_Spike. Training of a pulsed neural network in SpikeNNS is performed using Algorithm 6 (Section 5.2.2), which represents our most efficient version of an event-driven strategy. The general learning framework was described in Section 5.2.3. Three timing parameters have to be set for the LEARN_Spike function: an absolute value of the simulation start time t_{start}, an arbitrary value of the integration time moment t_{int}, and the time out limit t_{out}, until which the network activity is computed during a training step. For a description of these parameters significance see Section 4.3.4. Other parameters which have to be set for the training of a pulsed neural network concerns: the look-ahead value for the event-driven algorithm T_{window}, the flag a_f, which tells if the network activity is reset after each pattern propagation, and the learning frequency L_f (see description of parameters in Section 5.2.3).

During training and testing of a network, a number of statistics with respect to the neurons firing behavior can be collected and saved in a results file. This information concerns mainly the spike timings for each neuron in the hidden layers and the discharge rates during each input pattern application. The extension of the SNNS simulator to support modeling of spiking neural networks is only at its beginning and more work will be required in order to provide an integrated new version of SNNS.

Part III

Applications

Chapter 6

Models and results

During visually guided reaching for objects, our brain transforms a visual stimulus on the retina into a finely coordinated motor act. This process employs the activity of neural populations distributed along several areas in parietal and frontal cortex (Section 3.2). An ubiquitous feature of neurons in the parieto–frontal region is their broad selectivity to movement direction (Sections 2.1.2). This directional tuning of the neuronal responses has two consequences. First, the generation of motor output is based on a population coding of movement direction (Sections 2.1.3). Second, co–activation of neurons with similar tuning properties can establish a mechanism for the transfer of visual information into the motor command required for reaching.

In this chapter, we address two basic questions on the cortical control of the direction of movement. First, we investigate by means of computational modeling the mechanisms whereby motor neurons develop directional selectivity. Section 1 describes the formation of a directional motor map based on a self–organizing process involving spiking neurons and using input patterns analogous to proprioceptive feedback. The neural responses of individual units are characterized and the resulting population code is analyzed and compared to experimental findings of motor coding of movement.

Second, the self–organized motor map is re-used for the simulation of a developmental process of eye–hand coordination acquisition. The model proposed in Section 2 explores the means by which visual directional activity can be correctly used to guide the generation of the desired motor action. Visual and motor network alignment is discussed in the light of the connectivity pattern organization and of the individual unit's contribution to the mapping process.

6.1 Self-organization of neurons in motor cortex for coding the direction of movement

Modeling of the cortical coding of motion direction is relevant to understanding how control of movement is implemented in the central nervous system. It also represents a crucial obstacle to be overcome if the goal is the *understanding–for–reproducing* of the mechanisms involved in visual guidance of reaching movements. Given the importance of the subject, it is surprising that only a very small number of studies have been aimed so far at exploring the organization of directional motor maps (see Sections 2.2.3, 2.3). Previous work on coding of movement trajectories by neural population vectors (Lukashin and Georgopoulos, 1994) and on visuomotor mapping of direction (Burnod et al., 1992) represents only a starting point in modeling the organization of motor networks. What is still needed is a self–organizing neural network model to investigate how directional selectivity and population coding emerge in the motor cortex.

In this section such a neural network model is proposed. The simulation work is grounded in the experimental data showing that directional tuning is a prominent feature of motor neurons (Georgopoulos et al., 1984). The main findings of the experimental studies are: (1) neurons in the motor cortex are broadly tuned to preferred directions of movement (Section 2.1.2); (2) motion in a certain direction is determined by the activity of a large population of neurons; (3) the coordinated action of neurons can be characterized by using a neural population vector, which proves to be a good predictor of motion direction (Section 2.1.3); (4) particular directions are multiply represented in the motor cortex and cover a directional continuum; (5) the strength of the connection between two neurons tends to be negatively correlated with the angle between their preferred directions (Section 2.2.3).

A common approach in building neural networks that code motion directionality is to assign neurons with preferred directions from a directional continuum (see Section 2.3.2). This is done without considering the development of the neural selectivity or of the way neurons relate with each other as a consequence of having certain preferences. Instead of taking this view, we are interested in investigating the processes whereby directional selectivity emerges in a neural network. We want to explore how and why individual neurons learn to respond maximally for a particular movement direction, and not for another. Furthermore,

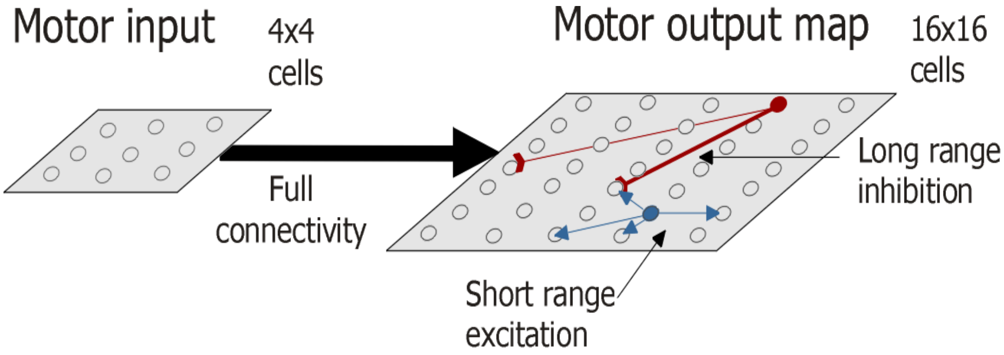


Figure 6.1: The motor system consists of a two–dimensional self–organizing feature map of spiking neurons. The input layer is fully connected to the output map. This has a heterogeneous composition of $N_e = 70\%$ excitatory neurons and $N_i = 30\%$ inhibitory neurons. The connectivity has a Mexican–Hat profile, represented by short-range excitatory synapses and long–range inhibition. The neural and connectivity parameters are shown in Table 6.1.

we want to discover the factors that influence the development of preferred attributes and the locus of acquisition of directional selectivity.

With respect to the representation of movement at the network level, the goal is to explore the possibility of obtaining in a stable self–organizing map, a distributed coding of a set of movement directions. The computational demand for the organizational process is to ensure the generation of distinct directional commands.

The model and the learning scenario proposed here are inspired by the simulation work on the self-organization of visual feature maps (Section 2.2.2, 2.3.1). This procedural transfer is grounded in the general idea that developmental principles described for sensory areas may reflect general mechanisms of cortical computation (Sections 2.2.3, 2.3). If this model proves successful in simulating the formation of directional motor maps, than it provides evidence for the generality of the mechanisms employed, while also helping to develop new hypothesis on the functional principles of the motor cortex.

6.1.1 Structure of the model

The motor system architecture is presented in Figure 6.1. It consists of a two dimensional heterogeneous self–organizing map with N_e excitatory and N_i inhibitory spiking neurons. Each neuron in the output map is fully connected to the input layer and is linked by probabilistic connections with other competitive neurons. The input unit activity is characterized

Model parameters
$max\,\Theta = 3$, $\Delta\Theta = 0.2$ - threshold parameters
$\tau_m = 5$ - membrane time constant
$u = 0.8$, $v = 3.5$ - refractory period parameters
$F_j = 10$ - number of spikes stored / neuron
$d_i = 2$ - input delay value
$\Delta d = 0.3$ - delay spread parameter
$\Delta f = 0.2$ - firing time noise
$N_e = 0.70$ - percent of excitation
$c_e = 10\%$, $c_i = 12\%$ - rates of connectivities
$\Delta c = 15\%$ - randomness on connections number
$\sigma_e = 4$, $\sigma_i = 3.5$ - connectivity decay parameters
$max\_w_a = 0.5$, $max\_w_l = 0.4$ - afferent and lateral weights parameters
$\Delta w = 0.1$ - weight spread parameter

Table 6.1: Neural and connectivity parameters for the self–organization of the motor directional map. The neural model is described by Equations 4.2, 4.3, 4.5 in Section 4.3. Detailed explanations of the connectivity parameters have been given in Section 5.3.2.

by a simple identity transfer function. The activation function of the output neurons is given by the membrane potential Equations 4.2 in SRM$_0$ model (Section 4.3). The neural response of an output unit evolves over time by combining the afferent signals with lateral excitatory and inhibitory feedback. The parameters used for setting the neural model are shown in Table 6.1 and their meanings have been described in Sections 4.3 and 5.3.2.

The self–organizing feature map (SOFM) has proven highly effective in modeling the formation of orientation and directional maps in visual cortex (Sections 2.2, 2.3). The basic principle of an SOFM is represented by the topology–preserving mapping of the input space to clusters of nodes in the output map. Learning of the afferent weights is assisted by a lateral feedback system, which leads to a cooperative organization of afferent and lateral connections (for the role of lateral connections in the development of cortical maps see also Section 2.2.4). In a network with a center–surround connectivity pattern (i.e., Mexican–Hat profile) the repeated exchange of localized excitation and long–range inhibition has an essential role in focusing the network activity, by enhancing the center and suppressing the activity at a remove from it.

It is also important to point at the sparseness of the connections between the competitive neurons. This is an important property of the network architecture, which is in agreement with biological data showing that in the cerebral cortex each neuron is coupled to a reduced number of other neurons (Braitenberg and Schuz, 1998). In our network, the probability of a connection existing between two excitatory neurons i and j decays with the increase in the Euclidean distance $d(i, j)$. Conversely, for two inhibitory neurons the probability of connection increases with the distance between them. How fast the connectivity decays is given by the parameters σ_e and σ_i. Details on the procedure used to create the probabilistic connectivity pattern have been presented in Section 5.3.2. For this simulation, parameters are tuned such as each excitatory neuron connects probabilistically with neighbors within distance $d = 4$, whilst an inhibitory neuron acts upon a fraction of cells placed from distance $d = 4$ to the margins of the network.

6.1.2 Learning procedure

Normally, learning in a self–organizing map consists of three steps. First, the activation of the hidden layer is computed with respect to the input vector. Following this, a best–matching unit is selected and designated as *winner* of the learning step. Finally, the weight vectors are adapted for all units in the winner neighborhood. One way to modify the weights is as a function of the cells discharge rates, after the network has reached a stable firing state (Section 4.4). When computing with spiking neurons, a more appropriate alternative is to apply learning as a function of timing of single firing events (Section 2.2.5).

In the algorithm proposed here, the winner is randomly selected from the subpopulation of units that fire the quickest in one simulation step. After choosing a winner, learning is applied as follows. The afferent weights of a competitive neuron i are adapted in such a way as to maximize their similarity with the current input pattern j. A measure of the similarity is the difference between the postsynaptic potential ϵ_{ij} that encodes the input stimulus and the connection weight w_{ij}. Furthermore, a spatial and a temporal neighborhood of the winner are created, such that only the neurons inside the N_c area and which have fired up until a reference time T_{out} are subject to learning. The learning rule is adapted from Ruf and Schmitt (1998) and is given by the formula:

$$\Delta w_{ij} = \eta \left(\epsilon_{ij} - w_{ij} \right) \frac{T_{out} - T_i}{T_{out}}, \quad \text{any } i \in N_c \tag{6.1}$$

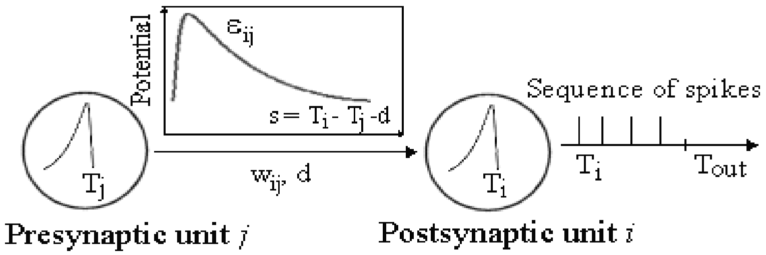


Figure 6.2: Modification of the synaptic weight w_{ij} is applied as a function of the parameters: T_j - time of the presynaptic spike, T_i - time of the postsynaptic spike, d - delay of transmission, and τ_m - time membrane constant.

with

$$\epsilon_{ij} = \begin{cases} exp(-\frac{s}{\tau_m}), & \text{if } s = T_i - T_j - d \geq 0, \\ 0, & \text{if } s < 0. \end{cases} \tag{6.2}$$

In equations above, η is the learning rate, τ_m the membrane time constant, ϵ_{ij} is the postsynaptic potential from presynaptic unit j, and T_i and T_j are the times of the first spikes of neurons i, respectively j (see Figure 6.2). Note that the temporal neighborhood $T_{out} - T_i$ is larger for the winning neurons and decreases for units that fire later. Hence, the largest change in the afferent weights occurs in the case of the winning neuron.

A similar rule is applied for learning in the lateral weights. The synaptic efficacy of a lateral connection is modified depending on the activity of the connected neurons and upon the arrival time of the presynaptic spike. The weight of an excitatory synapse is updated only if both neurons i and j are in N_c, if they both have fired and if the presynaptic spike has been emitted before the postsynaptic spike. For example, if the postsynaptic neuron i fired at T_i, than for all excitatory presynaptic neurons j that fired before T_i and fall inside a certain spatial neighborhood, the connection strength w_{ij} is potentiated. Note that strengthening of excitatory connections predominates over synaptic weakening. The later occurs only when the weight exceeds the postsynaptic potential value. Weights are also kept from excessive increase through normalization.

The learning algorithm for the inhibitory synapses was adapted after the rules proposed by Levy and Desmond (1985). If a presynaptic activity occurs before the firing of the postsynaptic neuron than we have a loss of the strength of inhibitory synapse with the value given by Equations 6.1 and 6.2, where $T_i - T_j - d > 0$ and $T_j, T_i \notin 0$. If we have a presynaptic activity paired with a postsynaptic inactivity than we have a potentiation of the inhibitory synapse with

$$\Delta w_{ij} = \eta \left[-(\epsilon_{ij} - u_i \cdot w_{ij}) \right] \frac{T_{out} - T_j}{T_{out}}, \quad \text{any } j \in N_c \text{ and } T_j > 0 \tag{6.3}$$

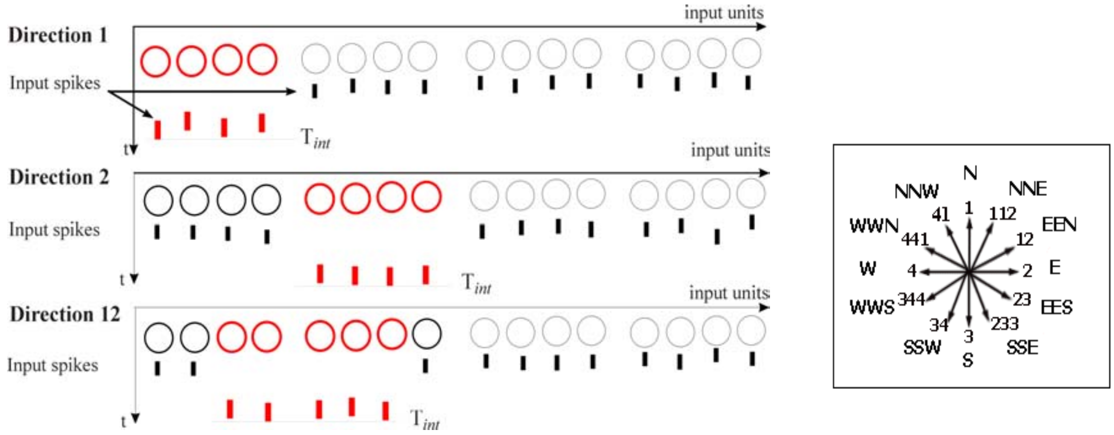
where u_i is the activation of the postsynaptic silent node, averaged over the T_{out} period. With the advancement of the simulation time, the neural response becomes more selective, hence more neurons are silent for other directions than the preferred direction. In this case, the rules above tend to favor an excessive increase of the inhibitory synapses, which is avoided through normalization by a fixed N factor.

6.1.3 Training patterns

In the theoretical introduction of this work it was suggested that a significant advance in the modeling of motor cortex development will be made by identifying the input signals that drive the organization process (Section 2.2.3). Hence, the milestone of our simulation work was the formation of a training set, which encodes the directional information needed for learning and moreover, is biologically plausible.

A source of inspiration in creating the input patterns was represented by the study of Lin et al. (1997) on neural trajectory computation (Section 2.3.2). Their work investigates how directional information can be extracted from the firing rates of motor cortex neurons by using a self–organizing feature map (SOFM). The SOFM is used to establish a correspondence between the discharge firing patterns of a number of $n = 81$ neurons computed at different moments in time, and the resulting direction of movement. Besides reading the directional information encoded, the self–organizing map allows a visualization of the similarity relationships between the input vectors. Thus, it indicates that neural discharge vectors for similar directions share a higher degree of similarity than vectors coding for opposite directions. This is a consequence of the fact that for similar directions, neurons fire at approximately the same rates.

Our hypothesis was that input vectors, which code directional information in the way described above, may be used for the self–organization of a directional motor map. The main assumption we made was that input vectors, which represent opposite directions, are highly dissimilar. Conversely, topologically close to each other directions are encoded by vectors



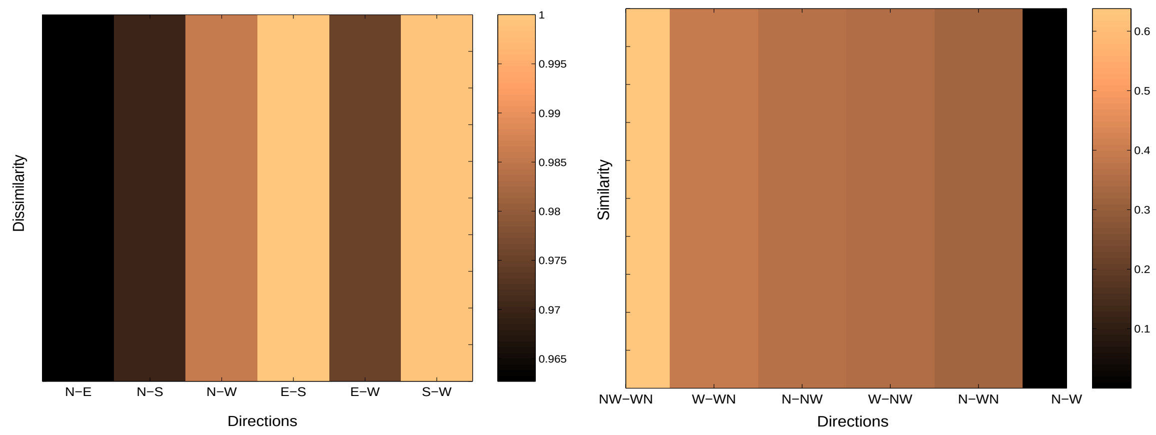
(a) Coding of directions

(b) Directions of motion

Figure 6.3: (a) Coding of three movement directions by the input vectors. The red units are referred to as principal units, since they code relevant information within a pattern. The rest of the units transmit noisy values. Orthogonal directions such as direction 1 and direction 2, are encoded by different subsets of input units. Direction 12 is topologically located between the two directions. Hence, it is encoded by a subset of units shared with the other two directions' representations. (b) Directions of movement which the self-organizing map learns to stably represent.

with similar values. A recent review of the literature on muscular control of movement has provided compelling support for our hypothesis. It indicated that the training inputs of the type we have used can be provided by proprioceptive signals arriving from the muscles involved in a movement (Theeuwen et al., 1996; Bolhuis et al., 1998). This is, because movements in opposite directions entrain activation of different sets of muscles (i.e., agonist / antagonist) that in turn provide feedback signals with distinct representations. An important future direction of our work is to use experimental data collected from arm muscles during directional movements to train the motor network (see also Section 7.2.2).

For simplicity we have defined only 12 directions of movement as represented in Figure 6.3b. Each direction is encoded by an input vector of dimension 16. The input values represent the firing times of the units (see Section 4.3.4). The values are generated in such a way that directions which form an angle larger than 90° are coded by vectors that are almost orthogonal (i.e., have a low degree of similarity). Examples are: **N**orth and **E**ast, or **NNE**ast and **SSW**est. Directions that are situated inside an arc circle of 90° are encoded by input vectors with higher degrees of similarities.



(a) Dissimilarity measure

(b) Similarity between patterns

Figure 6.4: (a). Dissimilarity values computed between input patterns representing four orthogonal directions: **N**orth, **E**ast, **S**outh, **W**est. (b). Similarity values between directions situated within a arc circle of 90^0: **N**orth, **N**orth-**W**est, **W**est-**N**orth, **W**est. The similarity (or dissimilarity) measure is given by the Euclidian distance between the vectorial representations of the patterns.

The degree of similarity between two vectors is given by the number of units shared in the representation (Figure 6.3a). For example, if direction **N**orth (direction 1 in Figure 6.3a) is encoded by the firing times of four input units u_n, than a direction similar with **N**, such as **NNE** (direction 12 in figure) will be encoded by a set of units u_k, so that $u_k \wedge u_n = 2 \vee 3$. Conversely, an opposite direction such as **S**outh (direction 2 in figure) is encoded by a set of units u_s, with $u_s \wedge u_n = nil$. Note that information on each pattern is distributed over all 16 input units. Thus, a number of four up to five principal units encode the training information provided by the input pattern and the rest of units code noisy values. The similarity and dissimilarity values between four directions of movement are shown in Figure 6.4. The similarity between two patterns is computed by calculating the Euclidian distance between the input values on each of the 16 pairs of neurons in the vectors.

After creating a prototype pattern for each of the 12 directions of movement we generate a training set of 12x20 patterns by adding noise to each value in the original vector by the formula: $u = u_i + R(m)$, where $R(m)$ is a random number generated uniformly in the interval $[0, 0.5]$ for a principal unit, and $[0, 2]$ for the other units. A small training set was sufficient for our simulation and it favored multiple representations in the network of each directional pattern.

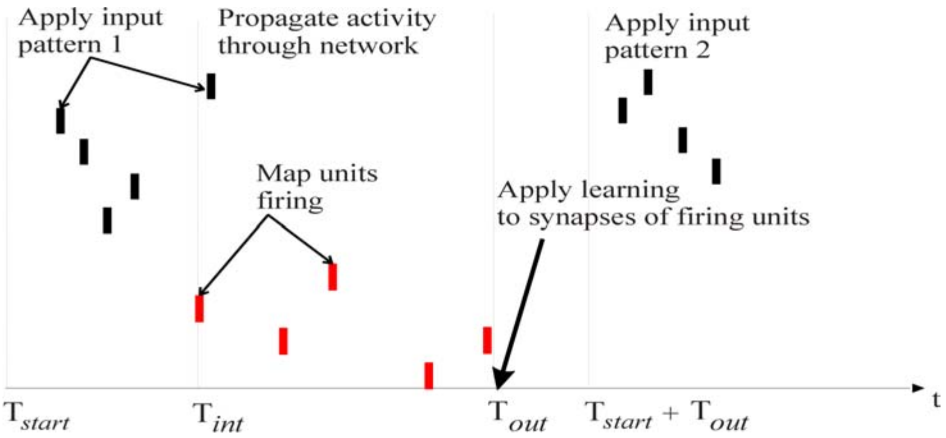


Figure 6.5: Phases of learning in a pulsed neural network: apply a time-coded input pattern starting from T_{start}; integrate competitive units activity beginning with the time T_{int}; propagate activity through the output layer until T_{out}; and update connections weights according with the learning rules. See also Section 5.2 for more details on the implementation of this scenario.

6.1.4 Simulation

The main phases of a simulation with spiking neural networks and time-coded input patterns have been described in detail in Section 5.2. Figure 6.5 summarizes these stages as input pattern application, propagation of the network activity and adaptation of learning synapses. One learning step lasts for $T_{out} = 30$ ms and is divided into the following stages (Figure 6.5). Input patterns are applied starting with the moment $T_{start} = 0$. First integration of network activity is done at $T_{int} = 9$ ms. Network activity is propagated into the output layer and neurons are allowed to fire for approximately 20 ms, until T_{out} is reached. Afterwards, modifiable synapses are adapted according with the learning rule, network activity is reset and the process continues with another input pattern.

For this simulation, the weights of afferent and lateral connections are initialized around the midpoint of the input pattern values (e.g., 0.5). The learning rate starts with a value of 0.5 and is decreased by a fixed percentage $\eta = 0.999\,\eta$. An initial radius of N_c is set to 5 and is reduced over time to the minimum value of 2. Training was performed for 1000 cycles, equivalent to applying 240,000 directional sequences. After a coarse organization of the network weights occurred, its characteristics were analyzed, as presented below. The network training was discontinued for two reasons. First, learning with spiking neurons is faster that with the traditional, rate-coding model, hence self-organization may occur earlier (see also

Farkaš and Miikkulainen, 1999). Second, activity in a self–organizing map decreases rather fast with the advance of the simulation time. This is due to the increase of selectivity in the neural response. Consequently, the advancement of network training would favor the formation of narrow tuning responses in a small set of neurons, associated with a complete silence in the others.

6.1.5 Results

Self-organization of afferent weights

Training of a pulsed neural network start with neurons being equally responsive to all input patterns. This leads initially, to the activation of the entire network for the presentation of any input pattern. With the advance of simulation time, the number of neurons that respond to a particular input vector slowly decreases, as selectivity of the neural response increases. In the end, in the trained self–organized map, only a small number of patches of activity occur for each directional command. These are usually organized around the winners of the corresponding direction. Hence, as a result of the self–organization process aimed at increasing the similarity between input patterns and weight vectors (Eq. 6.1), the afferent weights of a neuron that wins mostly for a particular direction resemble closely the corresponding input vector.

The labeling of the map is performed by marking each output unit with the inputs for which it is the most representative. Figure 6.6 shows the neurons' preferred directions labeled according with their winning behaviors while a test is run. The trained network was tested with the same pattern set used for learning. This was presented for a number of 10 times, while the winning behavior of each neuron was recorded. If for instance, neuron i wins mostly for a particular input pattern, than it is labeled with the corresponding direction. For the correspondence between directions and numbering labels see Figure 6.3.

The trained network bears a set of features, which have been induced by the self-organizing network property of a topology-preserving mapping. That is, the similarity relationships from the input space are translated into neighborhood relations in the output map. We may say that the network found a way to map directions according to their similarity relations (see similarity measures in Figure 6.4b). It resulted in most similar patterns being mapped to neighboring units. Note the succession of neurons labeled with $1, 12, 2, 23, 3, 34, 4, 41$ in

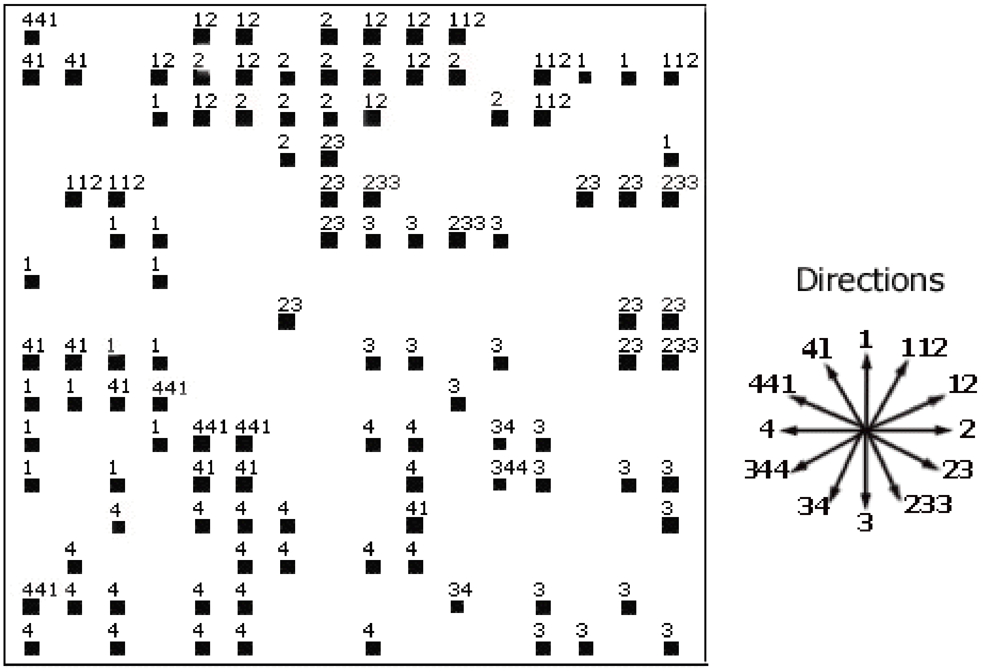


Figure 6.6: Self–organizing map labeled with the winning neurons' preferred directions. The preferred direction is given by the input pattern for which the unit is the best representative, that is, for which it wins most of the times after training has ended.

Figure 6.6.

A relatively high percentage of neurons developed selectivity for the input vectors, while approximately 45% of the total number of neurons did not respond directly to the presented patterns. With respect to individual neurons response, we note that 85% of the winners win for more than one direction of movement (not shown in the picture). Conversely, we also found about 15% of winner neurons whose first spike is constantly correlated with only one direction of movement. Together, these observations suggest that even if the most probable directional information is not read out from the timing of single spikes (i.e., due to a high number of broadly tuned winners), the second type of unit may also play a role in the representation of direction (see more in Sections 6.1.6 and in final discussions Section 7.1.3).

Figure 6.7 illustrates the self-organization of the afferent weights for seven units placed at consecutive locations in the output map. The rough pattern of the initial afferent weights (left) evolved during training into a smooth profile (right figure), which peaks over different subsets of input units corresponding to the pattern best learned by that output node. For instance, output unit 1 is unselective, node 2 is broadly tuned to direction **N**orth (encoded

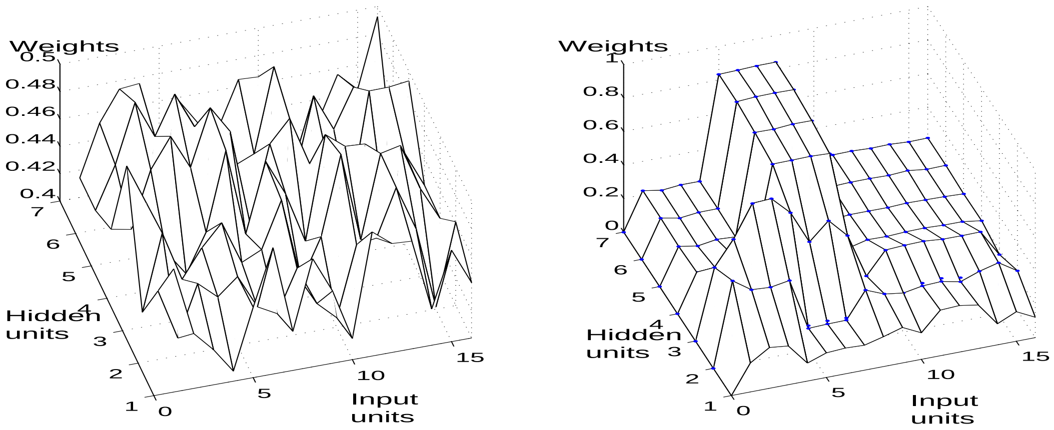


Figure 6.7: The afferent weights of seven neurons in the output map. (a) Initial random weights with an average value of 0.45. (b) As self–organization progresses, the weights organize into smooth hill–shaped profiles. Each weight vector has a peak (up to 0.9 value) for the input units which code the pattern best learned.

in first 4 input units), node 3 is broadly tuned to direction **NE**ast (input units 2 to 6) and last 4 nodes are more sharply tuned to direction **E**ast (inputs 4 to 8).

Rate coding of directional information

Figure 6.6 shows the calibration of the output map with respect to the winning behavior of neurons. That is, it reflects the organization of afferent connection weights. Apart from this type of labeling, normally performed for a SOM, we want to characterize the selectivity of each neuron as a function of its discharge rate during a time interval. Thus, for each of the 12 directions we run a validation procedure consisting of the following: while 30 input patterns are applied with a frequency of 60 Hz, the activity of the network is allowed to oscillate and the firing rates of neurons are recorded. The preferred direction of the unit is given by the input, which evokes the highest discharge rate.

The network labeled according with the firing rate behavior is presented in Figure 6.8. Only neurons with tuning levels above 30% are represented and the tuning value (normalized) is indicated by the length of the thick line. Note that compared to Figure 6.6 the number of unselective neurons has decreased to approximately 25% of the total number of neurons. The existence of approximately 15% neurons which are not directionally selective has also been described experimentally in the motor cortex (Georgopoulos et al., 1984).

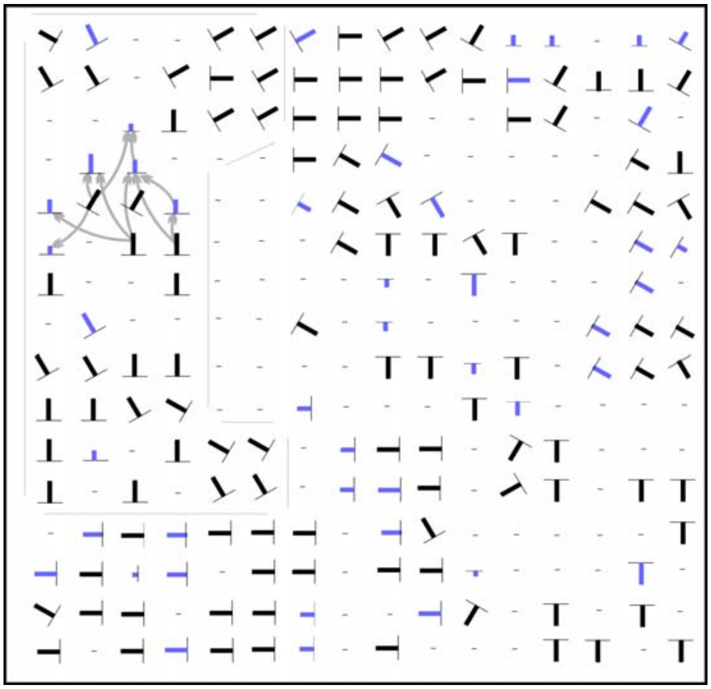


Figure 6.8: Self–organizing map labeled with the neurons' preferred directions, computed as the movement direction for which a neuron's discharge rate is highest. The level of neural tuning (normalized) is represented by the thick line. Two types of neural behavior are shown: *winning* neurons (in black) and *lateral* neurons (in blue). See the text for details on the functions of these neural behaviors. In the left side, an assembly of neurons is delimited, composed of winning and lateral neurons whose collective firing encodes the movement direction towards *N*orth. The gray arrows indicate the excitatory connections from the neurons which fire first (winners) to the neurons which fire later (lateral units).

The increase in the number of selective neurons compared to the previous calibration suggests that neurons other than winners have also developed directional preference. Thus, the analysis of the discharge rate behaviors revealed a set of neurons (drawn in blue in Figure 6.8) that had remained invisible during the first calibration of the map. These neurons have small values on the afferent weights and never won during the testing procedure for any of the directions involved. However, when recording their firing behavior in time it occurs that they spike later than the winners and their discharge rates are tuned to the direction of movement.

We refer to the latter units as *lateral* neurons, due to the fact that their spiking activity is mainly caused by the integration of the lateral excitation in addition to the afferent input. The lateral synapses from the winners (which fire quickest) to the later firing neurons are illustrated for a small assembly of neurons in the left upper corner of Figure 6.8. Because neither the lateral nor the afferent simulation solely can cause firing, the lateral neurons need to integrate both sources of excitation before they emit a spike. This constraint leads to a more selective directional response. Their tuning curves are less broadly than those of the winner neurons, with a mean width of $30°$. Later in this chapter, a third category of neurons is described that need to integrate three types of inputs in order to spike.

The joint activation of the two types of neurons give rise to a sort of collaborative cell assembly (Reilly, 2001), which enhances the strength of excitation between neurons tuned to similar directions of movement and suppresses the response in opposite directions. On the left side of Figure 6.8 the approximate boundaries of the cell assembly coding for movement direction *N*orth are indicated. The most important effect of collaborative cells assembly formation is the emergence of a population coding (see Section 6.1.5). Both the emergence of collaboration between neurons and the formation of a population code are supported by the horizontal feedback system. That is why, the organization of the lateral weights deserves a particular analysis.

Self–organization of lateral weights

The formation of the directional map shown in Figure 6.8 was driven by two principles. First, the SOM property to preserve the topological relations between input patterns led to the formation of an ordered representation of the directional space. Second, the lateral connectivity pattern with a Mexican-hat profile played an essential role in shaping the network activity. Thus, it favored the formation of distributed, stable representations for each direction of movement.

The lateral connection strengths are not static, but they evolve together with the afferents. For each firing neuron the lateral weights are adapted by a Hebbian learning rule (Equation 6.1) according to how well the neuron's activity correlates with the activities of other neurons. As the afferent weights organize into a smooth profile (see Figure 6.7), lateral correlations between neurons fall off with distance and become stronger only between neighbors with similar directional selectivity. Thus, the excitatory weights between neurons which develop preferences to opposite directions decrease through normalization and end up being pruned. Conversely, inhibition (decorrelation) increases between neurons that become selective to orthogonal input patterns.

To extract the common features of the lateral synaptic strengths we calculated the mean

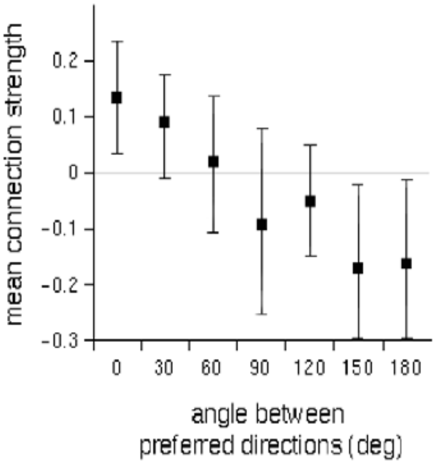


Figure 6.9: The dependence of the mean value of the lateral connection strength on the angle between the preferred directions of the neurons in a pair. The mean value of the weight was calculated by averaging over all weights, which connect neurons, whose preferred directions form the same angle.

value of the connection weight as a function of the angle between the preferred directions of two neurons in a pair. Figure 6.9 shows the mean values calculated considering all horizontal synapses in the network. We found that the mean connection strength is negatively correlated with the difference between preferred directions of the neurons in a pair. Our findings are in good agreement with the experimental estimation of lateral synaptic weights between motor neurons, as being a function of the difference between preferred directions (see Section 2.2.3).

Different profiles of afferent and horizontal strength vectors develop in winners as opposed to lateral neurons. One has to remember that during learning the change in all synaptic weights is constantly scaled by the difference between the time of the neuron’s first spike and a fixed arbitrary time out (see Equation 6.1). Hence, the later the lateral neurons fire, the smaller the adaptation of their afferent weights. Instead, their firing is triggered by the receipt of a lateral spike, therefore, the more the lateral connections will be strengthened. Conversely, because in winner units the first spike is caused by the afferent stimulation, they develop an opposite profile to the weight vector, characterized by strong afferent connections and small values of the lateral weights.

The strength of inhibitory connections also evolves over time. Inhibition increases whenever the firing of a presynaptic inhibitory neuron is correlated with postsynaptic inactivity (see

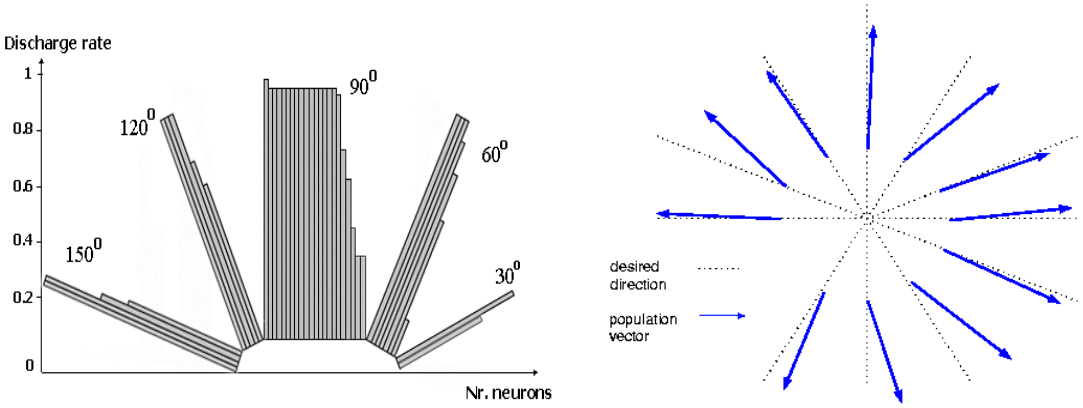
Section 6.1.2). Because inhibitory synapses run over large distances, the firing of an assembly of neurons for one direction of movement will result in lateral inhibition spread all over the network. Consequently, neurons belonging to assemblies that are not involved in coding the current direction receive a high proportion of inhibition. Because lateral neurons rely mainly on lateral excitation to fire, they end up by remaining silent. Instead, winner neuron activity is due to afferent stimulation, hence they fire (even accidentally) in more cases and for more directions than the lateral neurons. Therefore, the responses of lateral neurons are more suppressed (inhibited) than those of the winners. Consequently, it enhances their narrow tuning to the input stimuli (see Section 6.1.5).

It is important to explore with future models the influence on the lateral connection organization of variants of learning procedure and of different rates of connectivity. For instance, a learning algorithm based on the timings of a spike series would allow the winner neurons to integrate the effects of lateral feedback. This would cause a significant strengthening of the entire lateral connectivity system. The increase in the number of lateral neurons can be obtained by implementing higher connectivity rates, for instance excitatory neurons fully connected with neighbors within a certain area. A strong recurrent feedback is essential for the generation of neural oscillations in the network that have a role in maintaining the movement representation in time (see also Section 7.1.3).

Population coding

To this point, results have been presented in terms of selectivity of individual cells to the direction of movement. With few exceptions, our findings indicate that neurons (both winners and laterals) are broadly tuned to several directions of movement. Moreover, the command in each movement direction elicits in the output map the activity of an entire population of neurons (i.e., the collaborative cell assembly). These observations suggest two ideas. Firstly, directional information cannot be read out from the individual response of the majority of neurons (e.g., 80%). Secondly, it is possible that directional information is represented in the collective activity of subsets of neurons broadly tuned to similar directions. The later hypothesis is illustrated in Figure 6.10(a) where the joint activity of neurons with various preferrences, coding one direction of movement, is shown. That is, more than 40 neurons are firing at different rates while representing direction **N**orth.

In order to test the population coding hypothesis we computed for each of the 12 direc-



(a) Population coding of direction **N**

(b) Population vectors for 12 directions

Figure 6.10: (a). Discharge rates (normalized) of the population of neurons coding direction **N**orth. Each bar represents the contribution of single neurons and its orientation corresponds to the preferred direction of the neuron. (b) Population vectors for 12 directions of movement. Each vector is a resultant of individual neuron contributions (only the neural population for direction **N** is shown).

tions of movement the neural vectors, by using the formula (adapted after Lukashin and Georgopoulos, 1994):

$$P_x(t) = \sum V_i(t) \cdot cosPD_i, \;\; P_y(t) = \sum V_i(t) \cdot sinPD_i. \tag{6.4}$$

where V_i is the discharge rate of the neuron i during a testing interval of 600 ms. In Equation 6.4 we consider that the preferred direction of the neuron PD_i is given by its discharge behavior, as described in Section 6.1.5 (see also Figure 6.8). The populations vectors yielded by the vectorial summation $P_x + P_y$ are shown in Figure 6.10(b). Note that the neural vectors resulted point very closely to the desired directions of movement.

The population code is an effect of the distributed representation of direction, caused by the broad selectivity of output map cells. Each cell's activity is highest for a movement in a particular direction and decreases with movements further away from that direction. Figure 6.11 shows the discharge rates (normalized) of four neurons for different directions of movement. In the absence of a directional continuum, we cannot present an exact estimation of the tuning curves. However, we can approximate the shape and the width of the neural response. Neurons have unimodal tuning curves that differ from cells with the width of the curve of $120°$ (neuron 3 responds to maximum 5 directions of movement) to cells with a

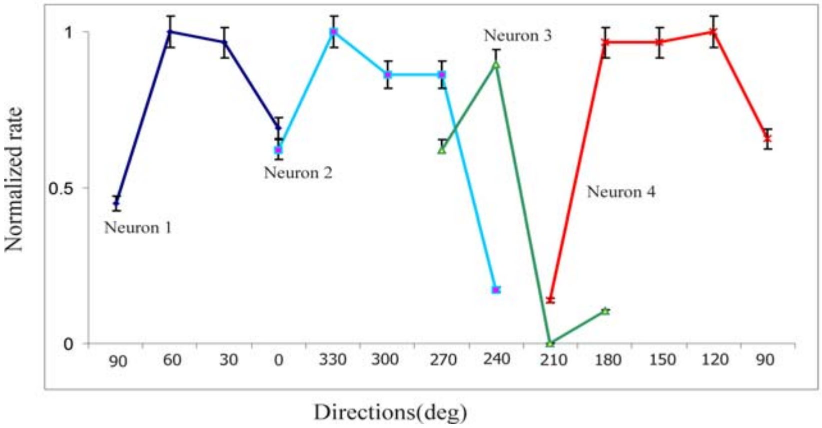


Figure 6.11: Discharge rates (normalized) of four directional selective neurons plotted as a function of the movement directions. The preferred directions of neurons of the neurons are located at approximately $90°$ from each other. Note that neurons show different shapes and widths of the tuning curves.

curve width of $30°$ (not shown in figure). The median of the curve width is at $50°$. A given neuron participates in coding of movements in directions which form an angle no larger that $90°$ with its preferred direction. Since directions forming an angle larger than $90°$ are encoded by orthogonal input vectors, the participation of the neuron in coding movement in such directions is due mainly to the successive propagation of excitation in the lateral feedback system.

The width of the neural tuning found in our model is in partial agreement with the experimental findings of Amirikian and Georgopoulos (2000), which describe real neurons' tuning curves with a half-width from 30 to $90°$ and the median $56°$. Their results indicate that motor cortical cells are more sharply tuned than previously thought (i.e., do not fit the cosine function). Our simulation findings point in the same direction (note that in our case the average of the midpoint of the curve is at $25°$), but more modeling work is needed for a better fit of the results to the experimental data.

To this end, we can try to put everything together into a hypothetical scenario of how the map response evolves in time from the moment it is presented with a move command. First, winner neurons are activated very rapidly. By doing this, the first spikes occurring in the network offer a quick motor output available to initiate the movement (e.g, if this is desired). But, since winners' tuning to the intended direction of movement is rather broad, this initial specification of the direction is very rough. In our case it restricts movement in a sector of

30° around the desired direction. One can also compute the instantaneous population code contributed by the activated winners, but this information is not more accurate than the readout of single spike timings from a few sharply tuned winning neurons.

Following the winners' activation, the lateral neurons get involved due to the horizontal excitation spread. During successive propagations of activation in the neural assembly, the lateral neurons contribute to creating a steady state for the network activity. This state represents the attractor of the desired direction and has a completely distinct position in the space for each of the learned directions. At these immediately successive stages of movement initiation (or movement preparation) the directional information can be read out correctly to drive the movement accurately to the target.

6.1.6 Discussion

In this section, a neural network model was proposed for the emergence of directional selectivity in motor cortex, based on acquired experience (i.e., proprioceptive feedback). The self–organized network exhibits properties that are consistent with the experimental findings on biological motor control. Furthermore, we believe that the modeling process can provide us with valuable knowledge about the organization and developmental principles of the motor cortex. The main conclusions are summarized as follows.

First, the model shows spontaneous emergence of a feature map during unsupervised learning and starting from random afferent and lateral connections. The self–organization of the network leads to a stable, ordered representation of 12 directions of movement. Most of the neurons in the self–organized network develop directional selectivity. All directionally selective neurons have a preferred direction, computed as the movement direction for which the neuron's discharge rate is highest. Furthermore, directionally selective units can be divided into two categories. Winners are units whose afferent weights are tuned to the input vectors and this is reflected in a rapid response to the movement command. Lateral neurons are units whose firing is determined by the integration of lateral excitation with afferent stimulation.

With respect to winner neurons' directional tuning, we found that the vast majority of them are broadly tuned to several directions of movement. However, in the case of a small percent of winner neurons (about 15% of all units) directional information is accurately encoded in the timing of the first spike. It is noteworthy, that the original experiments of Georgopoulos et al. (1982) also described the existence in the motor area of fast responding neurons, which are significantly tuned to one direction of movement. It is possible that the central nervous system uses this directional information for the initiation of fast movements, in a similar way the visual brain uses single spikes to take rapid decisions (Thorpe et al., 2001; see further discussion in Section 7.1.1).

To summarize these findings, we believe that the main outcome of our model is the emergence of directional selectivity in a motor self–organizing map of spiking neurons. Even if our model is a very simplified image of cortical motor computations and needs further refinements, we believe it represents a progress in the simulation of motor cortex organization. It was motivated by the absence in the motor control field of computational studies of how motor directional selectivity emerges (see Section 2.3).

The results obtained emphasize the advantages of modeling the self-organization process, as opposed to assigning neurons with preferred attributes (Lukashin and Georgopoulos, 1994). Thus, in our model, neurons can effectively 'gain' a preferred direction, as a function of the neural parameters, noise and location within the network. It shows that preferred attributes are not developed solely based on the neural features but also as function of its interactions with the network. Conversely, directional knowledge is encoded at the network level, that is in the horizontal connectivity, as well as in single neurons responses. We believe that by modeling the developmental processes, we can help to reveal the unknown functional principles of motor cortex organization.

An important issue concerns the function of the lateral connectivity system in the formation of the directional map. The role of horizontal connections in the organization of sensory feature maps have been emphasized recently by many studies on the visual cortex (see Section 2.2.2). However, much less is known about the profile and the function of lateral connectivity in the motor cortex. Our results indicate that neurons that develop similar directional selectivity become functionally correlated. In the trained map it was found that the strength of connection between neurons in a pair becomes negatively correlated with the difference between their preferred directions. This is an important modeling finding, which is in complete agreement with experimental estimations and previous simulation work (Georgopoulos et al., 1993; Lukashin and Georgopoulos, 1994. See further discussion in Section 7.1.2).

As a result of the distributed representation in the network of the directional information, we can observe the emergence in the model of a population coding. This result is even more interesting, as it was obtained with a self–organization process based on the timing of single spike events. Note that during training any information regarding the discharge rate behavior of neurons was discarded. Furthermore, by its nature the self-organization mapping focuses upon learning winner neurons. Hence, the emergence of a distributed representation of direction is determined by the organization of afferent and lateral connections. Firstly, as the afferent weights of a winner become more similar to an input vector, the unit increases its sensitivity not only to the best matching input, but also to all similar inputs (see the similarity measure in Figure 6.4). Secondly, the network response to one direction is amplified by the lateral feedback system, which activates the lateral neurons that are not directly responsive to the input stimulation. We may conclude that the population coding is a function of both the properties of the input space and the built–in network's constraints which allow the formation of collaborative cell assemblies (see also discussion in Section 7.1.1).

In our population codes, the individual contributions of single neurons can be summed up using the population vector scheme (Lukashin and Georgopoulos, 1994). Even if the neural vectors slightly deviate from the intended direction of movement, the main achievement of the model is that it is able to provide distinct commands for each direction (Figure 6.10). This means that the self–organizing map managed, within a distributed representation, to form stable attractors for each of the directions involved. Future work is aimed at tuning the neural parameters and the network design constraints in order to increase the accuracy of where the population vector points.

6.2 Development of visuomotor alignment of directional codes

A fundamental operation of animal brains and robot controllers is the integration of visual information with motor commands. Visually guided reaching in primates is considered to involve a series of neuronal events that transform retinal information about target location into the metrics of arm movement (Section 3.1). Most traditional modeling efforts using neural networks for robotic motor control have focused upon developing various formalisms capable of doing the coordinate system transformation (Section 3.1.1). Nevertheless, these implementations are neither relevant to the biological motor control, nor are they very adaptive or capable of learning and development (see Kalaska, 1995). Conversely, it was pointed out that inspiration could be drawn from the way motor control is realized in living systems.

Thus, recent behavioral and electrophysiological findings gave rise to a conceptual framework for the understanding, and possibly implementing, the computational mechanisms underpinning visuomotor coordination (Section 3.2). The work presented is in agreement with these latter theories.

In this section, an artificial neural network model is proposed to address a basic issue of visuomotor coordination: what are the computational mechanisms that allow visual information on the direction of movement to evoke an appropriate motor response in the same direction? We believe that this operation meets the basic computational demand for visuomotor mapping and represents a building block of the perception–action cycle (see Section 3.1.2).

Our approach to the visuomotor coordination problem was inspired by experimental data showing that neural selectivity to movement direction is a ubiquitous feature of the parieto–frontal networks involved in reaching (Section 2.1.3). During visual analysis of movement and at all stages of motor control the activity of a substantial percentage of movement–related neurons depends upon the direction of movement (i.e., involves direction selective cells) (Kandel et al., 2000; see also Section 2.1.2). The idea was to use the *motion selective cell* as the basic information-processing element from which neural networks capable of visuomotor control are built. Moreover, it is currently believed that visuomotor mapping of information (including coordinate system transformation) is realized progressively, by the gradual involvement of several populations of neurons (Section 3.2). We propose that the correlated activity of motion selective cells in the parieto–frontal areas meets the basic computational demand in the chain of operations required for visually guidance of movement.

The learning algorithm proposed here consists of a sequence of processing steps. First, the visual perception of a moving stimulus is translated into the firing pattern of directionally selective visual neurons. Next, this activity is transmitted to the motor areas that are involved in the control of movement direction. Simply, by the correlated firing of visual and motor neurons with similar neural response, a communication pathway for the directional information is established. In our simulation, the visual system corresponds to some early stages of visual information processing (e.g., retina and primary visual cortex). We believe that an equivalent computational mechanism can operate for the coupling of two or more networks of directionally selective cells located at any stages along the parieto–frontal networks involved in reaching.

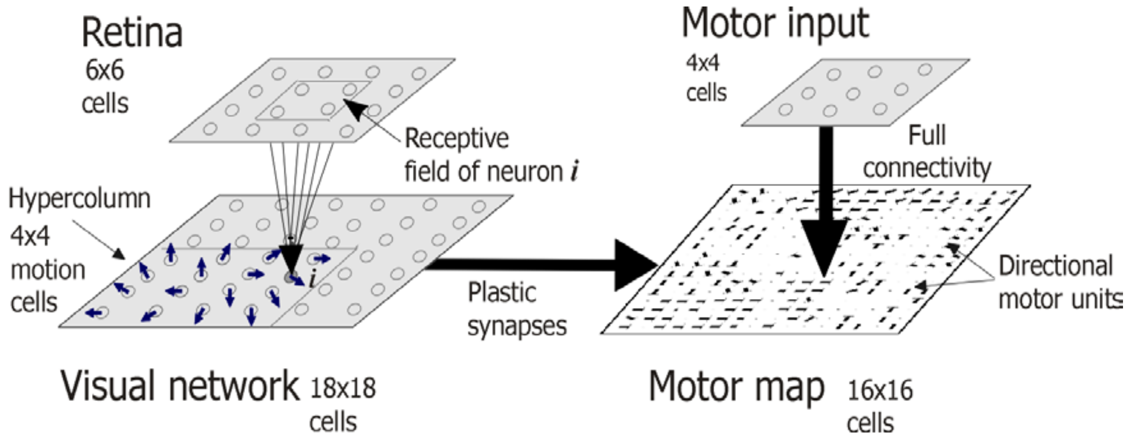


Figure 6.12: Neural architecture for the alignment of visual and motor representations of direction. The left part represents the visual system, consisting of a retina and a cortical visual layer. On the right is shown the spiking self–organized motor map trained to encode 12 directions of movement. The visual cortical layer is connected to the motor map through a synaptic pathway subject to spike–timing dependent plasticity.

6.2.1 Architecture of the model

In order to simulate the computational mechanism described above, we have implemented a very simple architecture consisting of a visual system connected to a motor map. There are two input layers and two 'cortical' layers: the *motor area* and the *visual area*. The visual area is represented by a layer of directionally selective neurons which have receptive fields from an input, retina–like layer (see description in Section 6.2.2 below). The motor area is a self-organizing feature map trained to represent 12 directions of movement. The organization process has been presented in Section6.1. All parameters describing the spiking neural model (i.e., postsynaptic potential, refractory period, transmission delays) are kept constant from previous simulation (Section 6.1, Table 6.1).

Each cortical neuron receives afferent connections from one input layer and lateral connections from other neurons in the same area. The visual cortical layer is fully connected to the motor map. The implementation of full cortico–cortical connectivity, rather than a coarse topographic ordering is motivated by the topology and the small scale of the networks. Due to the reduced dimension of the networks one can consider them as being tiny patches of cortical areas, large enough to comprise a complete representation of all features of input space (i.e., directions) at several positions. Full connectivity is required at this level to ensure that any population of motor neurons has access to all visual directions of movement.

In our view, the development of visuomotor coordination is achieved in two phases. During an initial organizational stage, the motor and visual systems develop independently, neural selectivity to the directional information. This stage ends up with the formation of two cortical maps, which stably encode direction of movement. Next developmental stage corresponds to the effective coupling of the two systems, which are allowed to interact and to learn, and eventually to coordinate. Experimental data concerning infants visual development indicate a prior emergence of cortical orientation mechanisms, followed at three month of age by the development of selectivity to motion (Atkinson, 2000). The later enables the cortical control of eye and hand movements. From this age until the age of $5-6$ months the cortical mechanisms for reaching (i.e., for eye-hand coordination) are developed. These observations indicate a possible sequencing of the processes involved in the perception and control of motion and in the achievement of visuomotor coordination.

The separation of the developmental stages is usually implemented by models of visuomotor coordination, but most commonly the directional maps are obtained by assigning (rather than developing) preferred directions to the motor (and / or visual) neurons (see Bullock and Grossberg, 1993; Salinas and Abbott, 1995). An original feature of our visuomotor mapping model is that it is based on a motor network, which has self–organized to represent directionality. Hence, it allows natural interactions between neurons and a self–developed way to encode the direction of movement.

In the next subsections, we focus upon describing the visual coding of movement direction and we proceed to the modeling of visuomotor coordination development. The reader should refer to the description of the motor map organization from the Section 6.1.

6.2.2 Visual coding of movement direction

After several decades of research on visual processing of information, it is now well established that the cortical analysis of visual space relies on the functioning of a fundamental neural machinery referred to as the *hypercolumn* (Section 2.2.2). A hypercolumn represents a set of columns which are responsive to lines of all orientations from a particular region in space via both eyes, and to movements in directions orthogonal to the orientation axes (Kandel et al., 2000). Experimental results have described a precise organization with an orderly shift in axis of orientation (or direction) from one column to the next (Section 2.2.2).

The formation of orientation and directional selectivity in the visual cortex has been extensively modeled using the self-organization paradigm (Kohonen, 1984). Most commonly, directional selectivity is developed in a map of cells with orientation preferences (Section 2.3.1). When the self-organized visual map is presented through the retina with a moving stimulus of a certain orientation, its response is represented by the firing of a certain set of neurons. These are the cells in the network which receive activation from retinal units located along the trajectory of the moving stimulus and whose preferred orientations (i.e., directions) are similar to the stimulus attributes. For an illustration of this process the reader can refer to a demonstration of how a moving oriented line is processed in a visual feature map (Bednar and Miikkulainen, in press; http://www.cs.utexas.edu/users/jbednar/sweeping_small.html).

Based on these findings, our visual area was implemented with a built–in, basic capacity of signaling motion direction. We have simulated a hypercolumnar organization based on directionally tuned neurons. One hypercolumn consists of 4x4 neurons, each firing for only one preferred direction and being silent for movements in different directions. Excitatory lateral synapses connect each neuron with the first order neighbors that have the same preferred direction.

The cortical map of area 18x18 is connected to a retinal layer formed from 6x6 neurons (left part of Figure 6.12). Each cortical neuron receives input from a fixed-size receptive field of dimension 3x3 units. The receptive field of one neuron (i, j) was centered at $(i/3, j/3)$ and the afferent weights were initialized within the range $[0.5, 1]$. The magnification factor, which is the ratio of cortical neurons per retinal neurons, is 9. In this way, it allows different receptive fields for each location in the retina (see description of implementation in Section 5.3.2). The lateral weights were initialized within the range $[0.3, 0.4]$. Parameters of the visual model were inspired by the simulations of visual cortex organization performed by Miikkulainen and colleagues (1996, 1998).

The scale of the hypercolumn dimension and the range of lateral connectivity are clearly a severe simplification. The built–in capacity for signaling motion was also implemented for simplicity reasons, as a substitute for the self–organization of the visual map. It was motivated by our intention to focus rather on the motor network organization and the visuomotor mapping process, than on the organization of a visual directional map. The main reason was that the formation of visual cortical maps has been extensively modeled compared with that of the motor cortex organization (see Sections 2.3). Nevertheless, this simple mechanism allows us to investigate the problem of directional information mapping, while retaining the principal characteristics of visual feature maps (i.e., narrowly tuned directional

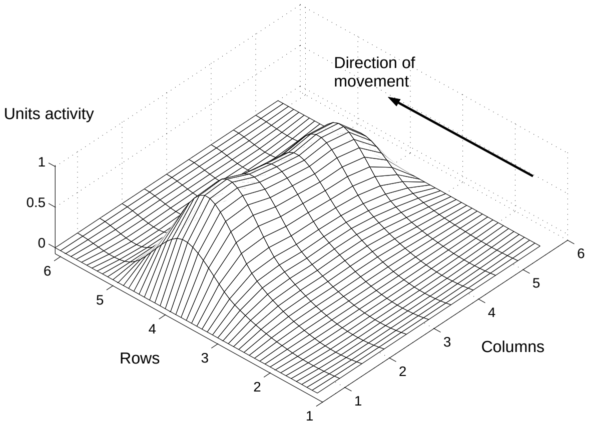


Figure 6.13: The visual input to the model consists of moving bars of a fixed length. There are 12 input patterns, corresponding to bars moving in the 12 directions of motion that can be controlled by the motor network. Movement of a bar through the retina elicits the activity of the cells located along the direction of motion.

neurons, lateral connections, receptive fields).

The retinal input to the system consists of directionally oriented moving bars of a fixed length (see Figure 6.13). A set of 12 input patterns was created corresponding to the 12 directions of movement controlled by the motor network. A bar moving in a certain direction in the retina determines the retinal neurons located along its pathway, to spike at a fixed continuous rate. The retinal activation reaches the visual layer after an average delay of 60 ms. This value was implemented in accordance with the delay existent in the brain between retina and primary visual cortex (Thorpe and Gautrais, 1997).

6.2.3 Learning procedure

The learning scenario was inspired by the sensorimotor stages that an infant progresses through to develop eye–hand coordination. When executing movements during the early motor–babbling period, infants perceive and learn contingencies between the motor activity and the visual image of their actions. By doing this, the simultaneous moving and tracking of the arm can provide the behavioral context for the development of visuomotor coordination. This becomes functional from about four months onwards, when infants start to make visually directed arm movements (Atkinson, 2000).

The behavioral framework described above is simulated by using the paradigm of direct

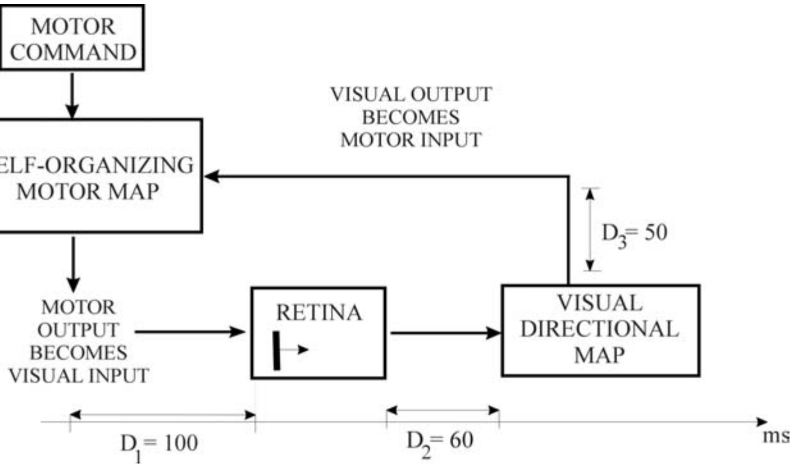


Figure 6.14: Schema of the training procedure used for the alignment of visual and motor directional representations. Note that the motor output becomes visual input and the visual activation is fed back into the motor network. The delay $D1 = 100$ ms corresponds to the interval between the time moment when the motor command was issued and the eye starts seeing the arm movement. The delay $D2 = 60$ ms represents the time required by the visual input to get from the retina to the visual network. The third value corresponds to the transfer delay from visual area to the motor area of $D3 = 50$ ms.

inverse modeling (Kuperstein, 1988), also known as the *motor babbling* approach. This has been discussed in detail in Section 3.3. The general algorithm consists of three steps: (1) a movement in a random direction is generated, (2) the visual image of the arm moving is recorded, and (3) the systems learns the correlation between the motor command and the visual effect of the movement. Most of the previous models of visually guided arm reaching have been based on different forms of error-correction mechanisms (Ritter et al., 1989; Fiala, 1995; Jordan, 1996). Only recently, it was pointed out that unsupervised learning means can also lead to an accurate mapping of sensory information (Salinas and Abbott, 1995; Burnod et al., 1999; see also Section 3.3).

Prior modeling work using correlation-activity associations was based on rate coding neurons, where learning is applied as a function of neural discharge rates (Salinas and Abbott, 1995; Baraduc et al., 1999). In this thesis, the focus is upon learning and computation with spiking neurons, based on the timing of single firing events (Section 4.2.3, see also Section 6.1). We believe this represents a simpler and more adequate framework for the study of visuomotor mapping through unsupervised means. That is, because detecting temporal correlations between populations of neurons is a natural computation with spiking neurons which has no equivalent in the rate coding networks.

In our algorithm, one learning step consists of a sequence of actions as follows (see Figure 6.14):

1. The motor input units generate a command for movement in a random direction d. This input elicits, in the motor map, the activation of the neural assembly involved in the representation of the direction d. The activity in the motor cortical population is maintained for a time interval of 400 ms, by the activation of the input command with a constant firing rate of 30 Hz.

2. Following the movement onset (i.e., at $D1 = 100$ ms after the motor command is issued) the retina is presented with a bar moving in direction d. This motion persists until the end of the learning cycle. After an average delay of $D2 = 60$ ms, the retinal information reaches the visual network, where the motion selective cells fire and signal the direction.

3. The cortical visuomotor circuit transmits the neural activity from the visual to the motor area with a delay of $D3 = 50$ ms.

4. During a time window of approximately 200 ms while both networks are active, spike–timing dependent learning is applied to the inter–cortical connections. The connection between a visual presynaptic neuron j and a motor postsynaptic neuron i is increased as a function of the time difference between the arrival of the postsynaptic potential t_j and the firing moment of postsynaptic neuron t_i:

$$\Delta w_{ij} = \begin{cases} \eta\,(\sum_{t_j} \epsilon_{ij} \; - w_{ij}), & \text{if } (\exists)\, \epsilon_j \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{6.5}$$

with

$$\epsilon_{ij} = \begin{cases} exp(-\frac{t_i - t_j}{\tau_m}), & \text{if } t_i - t_j \geq 0. \\ 0, & \text{if } t_i - t_j < 0. \end{cases} \tag{6.6}$$

In Equation 6.5, η represents the learning rate. The sum is calculated over the decayed values of postsynaptic potentials ϵ_{ij} coming from the visual neuron j. Each potential is computed by using the Equation 6.6, at the moment t_i when neuron i fires. The summation represents one way to ensure that the last presynaptic spike that has reached the neuron i is considered for learning. Difficulty arises from the large delay ($D = 50$ ms) between the visual presynaptic and the motor postsynaptic neurons, correlated with the exponential decay of the ϵ values.

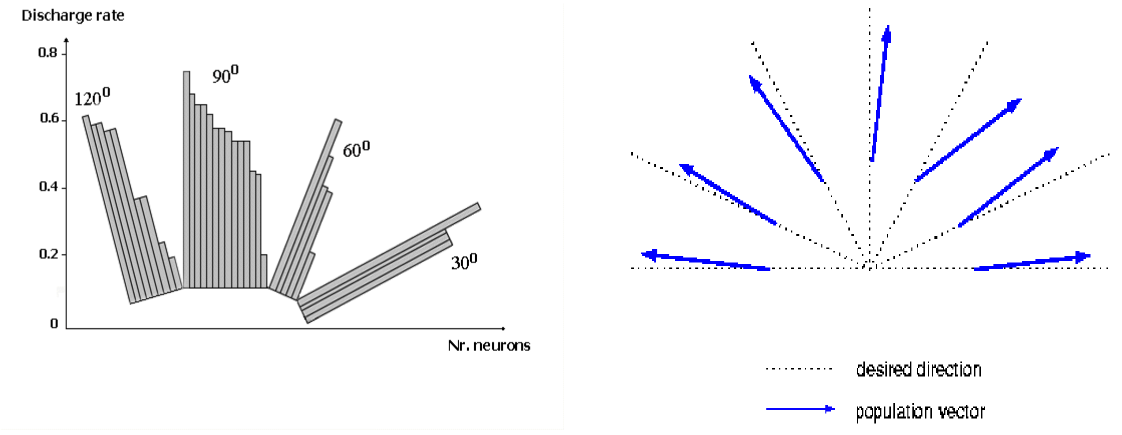
Furthermore, Equation 6.5 states that no modification of the synaptic weight w_{ij} is performed for time steps prior to the arrival of at least one visual spike to the motor neuron. That is, because about half of the time in a learning step, the motor neurons do not receive any activity coming from the visual network, due to the cumulated delay value $D1 + D2 + D3$. Hence, by applying anti–Hebbian learning a rapid pruning of all connections would occur. Our simplification is meant to speed up the learning process. More realistic scenarios that lead to similar results can be implemented, with the condition that the overall time when the visual synapses are active is much larger than the initial time interval when they are silent. In our case, a reduction of the strength of connection occurs only if the weight value increases over the value of presynaptic spikes effects.

Learning with the above algorithm happens quickly, mainly because the inter–cortical connection weights are initialized with low values between [0, 0.1] and strengthening of synapses dominates over synaptic weakening. The learning rate η was set to 0.1. After 500 cycles a pattern of strong connections develops from the visual to the motor area, causing the alignment of visual and motor neural representations in such a way that permits visual information to drive motor movement.

6.2.4 Results

To test the learning of visuomotor mapping of direction, the retina is presented with a bar moving in a constant direction for a time period of 400 ms, while the activity evoked in the motor map is recorded. For each of the 12 possible direction of motion, this motor activity is analyzed with respect to the shape of the firing patterns elicited (i.e., neurons involved) and to the population vector resulting. Figure 6.15a shows the discharge rates of the motor neurons activated by the visual information coding the movement of a bar in direction **N**orth. An initial qualitative analysis reveals that the firing patterns elicited in the motor network in the visual condition (i.e., when the motor map is exclusively driven by the visual stimuli) and in the motor condition (i.e., when a movement command is issued, see Figure 6.10a) are very similar.

Moreover, if we compute the population vectors (PV) in the visual condition, we find that the motor PVs driven by visual information point very closely to the desired direction of movement (Figure 6.15b). This means that the visuomotor system learned how to map correctly the visual information into directional movement. Alignment of the activity in



(a) Motor activity in visual condition

(b) Motor population vectors in visual condition

Figure 6.15: (a) Population activity occurring in the motor network in visual condition, while a movement in direction **N**orth is perceived. The network activity elicited by the visual input resembles very closely the population activity which controls movement in the same direction (see Figure 6.10a). (b) The motor population vectors computed during the visual condition for 7 directions of movement, by using Equation 6.4.

the two networks is due to the selective strengthening of the cortico–cortical connections associating visual and motor neural populations with similar directional selectivity. In the following section, we try to explain how the alignment of the maps has occurred and what are the characteristics of the inter–cortical connectivity which allow it.

Organization of inter–cortical connectivity

The reader is reminded that one of the main results of the simulation on motor cortex self–organization was that the strength of lateral connections is negatively correlated with the difference between neurons' preferred directions. With respect to the visuomotor model, we want to perform a similar analysis of the cortico–cortical connections. Hence, we compute the cumulated weight value for all synapses between neurons whose preferred directions form the same angle. Results are shown in Figure 6.16. To illustrate the result the cumulated (normalized) value was used, instead of the average measure, because of the large number of weights with very small values (< 0.1) resulting from initialization. However, both measures indicate the same finding: the negative correlation of inters–cortical connection weights with the difference between the preferred directions of the visual and motor

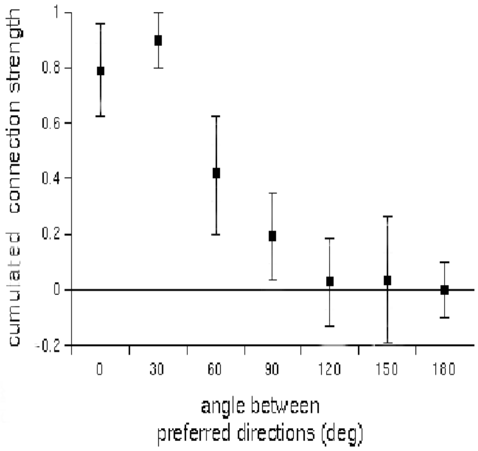


Figure 6.16: Dependence of the cumulated (normalized) weight value on the angle between preferred directions of neurons. Note that maximal connectivity strengths correspond to an angle between preferred directions of 30°.

neuron pairs.

In comparison to the organization of lateral weights in the motor cortex (Figure 6.9) the visuomotor connectivity is characterized by a larger distribution of weights values. In the latter case, non-zero weights exist between neurons with PDs forming an angle up to 150° and the peak of the synaptic strength corresponds to a difference between PDs of 30° (i.e., compared with 0 for the motor map). Both features suggest a broader coupling of neurons, which has an effect on the accuracy on the generation of the desired direction of movement.

The findings with respect to the strengths of the mapping connectivity are relevant to those inferred mathematically by Salinas and Abbott (1994). The authors proposed that for map alignment to occur, the strength of the synaptic connection between neurons has to depend only on the magnitude of the difference between their preferred locations (i.e., directions) (see Section 3.3.3). In our simulation we found that the function between the strength of connection and the angle is a negative correlation.

Analysis of neural properties

The organization of the inter–cortical connections as described above, indicates one of the causes of the networks alignment: the visual and motor neurons with similar directional preferences become selectively coupled. Strengthening of these synapses instead of others occurs due to the time–correlated activity of those visual and motor neurons involved in coding the same direction of movement.

The above analysis is quantitative. It shows that at the population level, neurons coding similar directions become selectively coupled, but does not tell us how many and which of the motor neurons respond to visual information. To answer these questions, we begin by examining the profiles of motor activity patterns in the visual and motor conditions. The analysis reveals differences with respect to which neurons are activated in each condition and in the tuning properties of those neurons involved.

First, in the visual condition a larger distribution of the preferred directions of neurons involved in the generation of movement is observed. Compared to the motor condition, where neurons participate in coding directions which form an angle no larger than 90° with their PDs, in the case of visually guided movement neurons with optimal tuning up to 150° from the current direction are activated. As we mentioned above, this broad activation is due to the development of inter–cortical connectivity that links more than only neurons with similar preferred directions.

The formation of this visuomotor coupling can be explained by considering the characteristics of the learning procedure and the specifics of the motor neural responses. In our learning algorithm, for a synapse between two neurons to be strengthened, the presynaptic visual spike which travels through the axon for about 60 ms (i.e., time interval whilst the postsynaptic neuron can fire up to 8 times) has to reach the postsynaptic motor neuron in a short time window before this one fires. Obviously, this large delay makes the synchronization of the presynaptic and postsynaptic activities more difficult. Accordingly, motor neurons, which fire at high discharge rates during the whole movement, rather than neurons which fire precisely timed, but fewer spikes, will be favored in establishing functional connections with the visual neurons.

The analysis of the dynamics of motor neurons reveals two things. First, the neurons that become responsive to the visual stimulation are indeed those neurons (lateral or winners) that exhibit high discharge rates in the motor condition. By firing during both the execution and the perception of movement direction, these units give rise to a new type of *visuomotor* neural behavior (see discussion in Section 7.1.3). Second, for each direction of movement we found that about 20% of the motor neurons remain silent during the visual condition. That is, they participate in the action execution in the motor condition, but not when the stimulus is visual.

Third, an interesting finding is the activation during the visuomotor mapping of a particular subset of motor neurons, from those, which are not very active for movement execution. These are motor cells that hardly reach the threshold when a movement is generated in the absence of visual input. Hence, during the motor condition, they emit very few spikes due to random factors (i.e., noise) and usually they are in a sub–threshold state. A slight increase in the inter-cortical weights determines that the visual signal is integrated together with the local lateral excitation and the motor input and causes their constant firing. Note that in order to fire, these neurons have to receive intra–lateral excitation, hence they are situated inside the assembly currently involved in the execution of movement. Furthermore, because the visual spikes trigger their activity the inter–cortical synapses of these units are significantly increased. This leads to the fact that, eventually, the visual signals independently can activate them.

Results show indeed, that these signals related to neurons firing at high rates during the visually guided actions. By contrast, the motor neurons involved in the command of movement spend much of their time in the refractory period. Hence, they have less chances to synchronize with the arrival of the visual spikes and to increase significantly their inter–cortical weights. Furthermore, it is not a coincidence that the signal related neurons are placed in the center of a cell assembly, similarly to the way lateral neurons are situated. That is, because only in such positions characterized by a strong surrounding excitation, can these neurons accumulate enough stimulation to fire. In a reciprocal manner, the visual related neurons send excitation to the other cells situated in the same neighborhood.

Based on these observations, we propose a hypothetical scenario for how visuomotor mapping of direction takes place in our model. In the first place, a percent of the motor neurons are activated directly by the excitation coming from the visual neurons. Second, the visual signals trigger activity in the visually–related motor neurons, which in turn spread activation in their neighborhood and determine the population to fire. Note that the visuomotor neurons firing are placed inside the cell assembly, which we want to activate. This hypothesis is consistent with the neurobiological findings describing the existence in the primary motor cortex of different types of units, including sensory–related, motor, and sensorimotor neurons (Zhang et al., 2000; see also discussion in Section 7.1.3).

Further work will be aimed at exploring the means by which we can improve the visuomotor mapping accuracy. These preliminary results suggest that a correct alignment of the two maps is favored by a large, distributed representation of directions, coded in the discharge rates of neurons, rather than in the precise timing of the spikes. We believe that a more accurate visuomotor mapping can be learned using a larger motor network, where each direction of movement is multiply represented in the discharge rates of an extensive population of neurons. We also aim to explore in more detail the role of visually–related motor neurons in the formation of the motor response.

6.2.5 Discussion

In this section a computational model for the visuomotor mapping of directional information was proposed. From our view, learning of the visual guidance of movement is a developmental process, which takes place by unsupervised means. Moreover, it occurs between two systems that have already developed specific capabilities for representing and generating directional movements. To elicit an appropriate motor response, the directional information must be transferred from the visually related areas to the arm-control areas. We believe that the basic operation in this process is carried out through the correlated activity of directionally selective cells placed along the reach–related areas of the parieto–frontal network.

Our model shows that the visuomotor transfer of coded information is supported by the development of inter-cortical connection weights negatively correlated with the difference between the preferred directions of the visual and motor neurons in a pair. This result confirms the estimation made by Salinas and Abbott (1994). Furthermore, we found out that not all motor neurons in the map become responsive to the visual input. Those which do, are represented mainly by neurons that are significantly tuned to directional information (i.e., fire with high discharge rates) during the movement command. Besides the visuomotor neural behavior, a type of signal-related neurons develop in the motor area, from those, which were not very active during the generation of movement. These neurons may play an important role in relaying information from the visual cortex to the motor areas (see also Section 7.1.3).

The idea of using the correlated activity of motion selective cells for visuomotor mapping has already been modeled by Burnod et al. (1982), Salinas and Abbott (1995), and more recently has been made the basis of a theoretical framework proposed by Burnod et al. (1999). Compared to this earlier work, the merit of the current study is to have obtained an alignment of the motor to visual information in conditions of realistically implemented motor coding of direction. In our scenario, by contrast with previous models, directional selectivity and population coding emerge in the motor network, as a result of a developmental process. Moreover, self–organization in our model occurs on networks of spiking neurons. In this respect, the simulation work described in this chapter revealed interesting findings on where directional information is coded. On the one hand, for the fast control of movements, directional information might be read out, if necessary, from the timing of the first spike of a subset of spontaneously activated neurons. On the other hand, the visuomotor mapping accuracy depends on the way the directional information is coded in the motor cortex. In this case, rate coding over a large population of neurons supports the correlation of activity in the two networks and allows learning. More work is needed in future to analysis in detail the synchronization of activity at the level of single spiking events.

Compared to the previous models of sensorimotor mapping based on the correlated firing of motion selective neurons our model has modest aims and attempts to address fewer issues. We do not account for coordinate system transformation (see Salinas and Abbott, 1995) and our simulations are not effectively implemented for motor control (Bullock and Grossberg, 1993). Rather, our model constitutes an illustration of how a *complex* problem, such as the visuomotor mapping can be implemented in a very simple manner by two networks of directionally selective spiking neurons. This model argues for the importance of the individual neurons low–level properties (i.e., as directional selectivity) in implementing large–scale phenomena. Our hypothesis that the neural selectivity for stimulus attributes lies at the foundation of visuomotor mapping is very similar to the theoretical concept of combinatorial domains proposed by Burnod et al. (1999). We consider that our simulation findings have computational relevance for the experimental framework developed by these authors (see discussion in Section 7.1.4).

Moreover, the model proposed supports the thesis that gaze (i.e., eye) movement neural activity can be *re–used* to control the movement of a limb. In our view, this means that the directional gaze information becomes correlated, through a mechanism similar to the one described here, with the arm movement activity, allowing the eye–hand coordination through a motor–to–motor program re–use (see Section 7.1.4). In the final chapter, the relevance of our findings to the current theses of visuomotor development will be discussed (Section 7.1.4, along with the possible applications of our models (Section 7.2).

Chapter 7

Discussion and future directions

The final chapter of this thesis includes a discussion of the simulation results obtained and of the theoretical and practical significance of our work. In Section 1 the neurophysiological implications of the models on motor cortex self-organization and visuomotor mapping are discussed. In Section 2 we discuss the psychological relevance of the visuomotor model to the imitation issue and we present future extensions and possible applications of our work.

7.1 Neurophysiological implications

7.1.1 Emergent vs. innate directional selectivity of motor cortical neurons

Despite the vast body of knowledge currently relating to the motor control of the arm, there are conflicting explanations of how the neocortex participates in motor control (see the debate on muscles vs. movements encoded in primary motor cortex in Section 2.1.2). One impediment to a complete explanation of the function of the M1 is that the fundamental organizational principles of the cortical motor areas are yet not clear (Sanes and Donoghue, 1992). For instance, there is a large body of evidence indicating that directional tuning is an essential feature of motor cortical neurons (Section 2.1.3). However, it is not yet known whether this neural characteristic is acquired by experience or genetically encoded.

Our modeling work on the self–organization of motor cortex represents a first attempt to provide a learning scenario for how motor cortical neurons develop directional selectivity.

Put generally, it demonstrates that a self–organizing map can learn to distinctively represent and command 12 directions of movement, by extracting the similarity relationships from the input space. The success of the self–organization process is dependent on two factors: the input patterns and the feedback connectivity system.

A self–organizing feature map is a means of visualizing in a reduced dimensional space (usually two) the spatial relations existing in a multidimensional space. Hence, if we aim to approach the formation of the motor directional maps from the self–organization perspective, the crucial aspect consists of the characterization of the input signals that are available to the training process. From this view, the population coding operating in the motor cortex and the functional connectivity that has formed, are the result of a self–organization process and reflect up to a point, the peculiarities of the input space. However, as it was pointed out elsewhere in this thesis, in the case of the motor cortex as opposed to the visual cortex, it is less clear what precisely might be the input (i.e., training) data to the self–organization process. Our hypothesis is that the formation of the directional motor map is driven by proprioceptive feedback from those muscles involved in movement. This idea will be the starting point of our future work on modeling motor cortex organization (see details in Section 7.2.2).

In our model, a directional feature map emerges through unsupervised learning from a random initialization of the afferent and lateral connection weights. There is one built–in constraint in the shape of the network connectivity: the short–range distribution of excitation and the long–range inhibition. The connectivity with a Mexican–Hat profile is a general feature of a self–organizing feature map. That is, because short–range excitation is needed to focus the activity in the immediate neighborhood of the winning neuron, while the long-range inhibition helps to suppress the network response in the contralateral direction of the movement.

There is experimental evidence for the existence in the motor cortex of adult animals of excitatory connections which link nearby neurons with similar neural responses and of distal inhibitory connections between neurons with different tuning curves (Georgopoulos et al., 1993; Hatsopoulos et al., 1998). However, it is not clear if this is a result of a developmental process taking place in the motor areas or represents a built-in feature. We believe that these kinds of questions can be easily explored through modeling work, with promising and valid results. Part of our future work is aimed at exploring the influence on the map organization of the variation of connectivity pattern parameters (i.e., the rate of the connectivity and the spatial distribution of the lateral feedback).

Can be the motor cortex modeled like the visual cortex?

Another observation with respect to the theoretical assumptions of our model, concerns the fact that it was mainly inspired by models of the visual cortex (see also discussion in Section 7.1.2 below). The power of the visual model was twofold. Firstly, at its heart, the modeling of visual cortex organization challenges the idea of innate cortical features (Hubel and Wiesel, 1962; Gilbert and Wiesel, 1992). We believe that if it is possible that the visual directional maps are shaped by experience, then it may also be feasible to model the developmental processes of motor cortex. This hypothesis is supported by experimental studies that indicate that there is a self–organization capacity in the cortex of adult animals, which is perhaps part of the original developmental organizing processes (Merzenich et al., 1983; Hess and Donoghue, 1994; Rioult-Pedotti et al., 1998).

Secondly, the modeling studies of the visual cortex put forward in the last decade a core of hypothesis on the computational and design principles of the brain. First, the self–organizing feature map (SOM, Kohonen, 1994) has been very successful in modeling the development of sensorial maps. It has become almost a *de facto* standard in the biological modeling of brain self-organizing processes. Secondly, computational studies pointed out the essential role played by the horizontal connectivity in the formation of orientation, binocular, or directional maps (Section 2.2.4). Placed in this context, our simulation work has the advantage of a bi-directional knowledge transfer. On one hand, our study has been largely inspired by existing models and data from the sensory cortices. On the other hand, if our model proves successful in simulating the formation of motor directional maps, then it provides computational evidence of the learning mechanisms and the functional principles of the motor cortex.

Our findings (Section 6.1) indicate that the self-organizing feature map represents an appropriate modeling framework for the developmental processes taking place in the motor cortex. Furthermore, we found that the lateral feedback system plays an essential role in the organization process, in a similar way to the role it plays in visual cortex development. Plasticity of both excitatory and inhibitory connections is essential for self–organization to occur, by finely adjusting cells tuning level to the input space features. Exploring the effects of learning in terms of single spike timing represents our original contribution in the area of self–organization with spiking neurons. Our model indicates that directional information may also be read out from the timing of the first spike of fast responding neurons. Surprisingly, this observation comes out as a possible common feature of information processing in the visual and motor brain.

Until very recently, the common belief in computational neuroscience was that information in the brain was carried mainly by the neuron's discharge rate (see Section 6.1.5). Recent experiments on visual categorization revealed the existence of a very fast processing of information in the visual cortex, possibly based on the order or timing of a single spike per neuron (Thorpe et al., 1996; Thorpe and Gautrais, 1998). In the case of the motor system, the influential work of Georgopoulos and his co–workers (1984, 1986) proposed the population coding scheme as the main paradigm used to interpret and predict movement based on the motor cells' discharge rates. Based on our modeling results, we suggest that a fast response of the motor cortical areas, read out from the timing of the first spike of optimally tuned neurons is certainly advantageous and quite likely implemented by the motor system. The only restriction is that such an answer has a very limited precision and only further processing of the directional information by a large population of cells can give rise to an accurate reach movement.

It is beyond the scope of this thesis to offer an answer to the question of whether directional selectivity is a genetically encoded feature of the motor cortex or is acquired by experience. The model proposed here provides only a number of computational ideas on what it takes to develop neural selectivity and population coding in a biologically plausible system, by unsupervised means. We believe that by developing optimal responses in its elements, rather than having them pre–wired, a system can show a flexible and plastic architecture that adapts to the resources available and to the particularities of the input space.

7.1.2 More evidence for the importance of horizontal connections

It was pointed out that in our model, an essential role in the organization of the motor map was played by plastic lateral connections (see Section 6.1.5). Our findings show the formation of functional connectivity in the motor area with a similar profile to the patterns of connections described in other brain areas. Thus, experimental data from primary visual cortex shows that horizontal projections link columns with common ocular dominance and orientation selectivity's (Gilbert and Wiesel, 1992). In the auditory cortex dorsoventral connections link regions with matched characteristic frequencies (Read et al., 2001). Recent modeling of eye–saccades planning in the lateral intraparietal area (Xing and Andersen, 2000) have shown that in order to hold memory activity for a saccades, the neural population develops excitatory connections between units with similar preferred saccade directions and inhibitory connections between units with dissimilar directions.

Previous modeling results similar with ours have been obtained by Lukashin and Georgopoulos (1994). They found that during a supervised learning process, the strength of connection between directionally tuned motor neurons becomes negatively correlated with the difference between their preferred attributes. This sort of experimental data and computational work, suggest the manifestation in the brain of a general principle for horizontal connections organization. It is generally believed that this is reflected in the correlation between the strength of interaction and similarity among units' preferences.

With respect to the computational function of the lateral feedback system, our model of motor cortex organization and visuomotor mapping, indicate a crucial role of the horizontal connections in shaping the activity of the network and in favoring the formation of stable attractors of motion directions (see Section 6.1.5 and 6.2.4). On the short scale, the lateral excitation increases the collaboration within a cell assembly, while the lateral inhibition suppresses the answer in the opposite direction. On the large scale, the plastic connections implemented in the visuomotor system, mediate the transfer of information and synchronization over a large distance (i.e., 50 ms delay). Our findings suggest that correlated activity in motor and visual networks is a result of both organization of long–range connections and collaboration mediated by the local lateral pattern.

Similar observations have been made by Usher et al. (1996), who studied the role of long–range connections for visual binding and line completion. They used a network of leaky integrate–and–fire neurons with long–range connections implemented only between cells with similar orientation preference. Their findings revealed a clear tendency for synchronization between cells with same orientation preference separated by large distances. In their model, if two distal cells placed in the range of clustered connections receive the same stimulus (even if they are not optimally tuned themselves), they indirectly synchronize via the intermediate synchronization with their optimally tuned neighboring cells.

Compared to Usher and co–workers work, our simulation has the advantage of developing the profile of the long–range connections. In our case, the network coupling consists of a full connectivity from the visual to the motor network, initialized with weight values nearby 0. By doing this, we do not arbitrarily restrict which is the visual directional information perceived by the populations of motor neurons that encode all movements directions (i.e., in our case, whole motor network). The coupling of neurons according to their preferences, should be an emergent feature, rather than a built–in property that limits the network plasticity.

In our model, as a result of learning, clustering of connections occurs in a similar way to the pattern implemented by Usher and colleagues. The strength of long–range connections cluster in spatial neighborhoods that correspond to the directional cell assembles in the motor area. Accordingly, correlated activity between visual and motor neurons is induced not only via the long-range synapses, but also through the mediation of the visually-related motor neurons optimally tuned to the direction of movement (see discussion in Section 7.1.3 below). Future work will take into account a more realistic scenario in which the cortico-cortical projections start with some initial non-zero (i.e., biased) values and weakening of synapses, besides strengthening, will be allowed.

7.1.3 Dynamics of single neuron activity in the motor cortex

Up to this point, the discussion has focused upon describing the main requirements for self–organization of the motor map and the alignment of visual and motor neural representations. However, an important co–lateral effect of modeling these developmental processes was to observe the emergence in the motor network of different patterns of neural behavior. These may reflect various functions in the preparation and execution of movement, which are discussed below.

During initialization, the neural spiking model is set up in such a way that all motor cells begin the simulation equally selective to all motion directions. However, learning in a self–organization map (SOM) is a competitive process and takes place by amplifying any small differences in the neural response. If one neuron wins for a certain direction, its synaptic strengths are modified to increase its chances of winning again for that pattern. In this tuning process, the variability of the neural response is an essential factor and is given by the level of noise (i.e., in the threshold value, the firing time, the spike transmission delay) and the pattern of connectivity. Due to the fact that input signals can arrive in a synchronous or asynchronous way, this neural variability causes the possible operation of neurons in two modes. Thus, a neuron is capable of switching between computational modes, from the integration of firing rate input received from a large number of neurons, to the detection of coincident spike arrivals (see also experimental evidence for neurons acting in different computational modes in Destexhe and Pare, 1999). Hence, we can describe the existence of two main classes of neurons.

The first class, that of coincidence detectors, is mainly formed by the *winner neurons*, which represent about 40% of the neurons in the motor area (i.e., $110/264$) (see Section 6.1.5). These neurons respond very rapidly to the input signals that are emitted synchronously and affected by similar values of noisy delays. Hence, their afferent weights become highly tuned to one input pattern. Note an important difference between learning with a SOM of continuous, rate-coding neurons and a SOM consisting of spiking neurons. While in the former an input pattern is mapped onto a single neuron that has the maximum activation, in the later, any pattern similar to the best–matching pattern will determine the firing of the winner neuron. Hence, if there is no increase of the neural threshold, what we have obtained in our spiking SOM, were neurons highly responsive to several (i.e., maximum three) directions of movement.

It is noteworthy that the preferred directions of each neuron, when represented on a circle, occupy an arc of maximum 60°. Similar results have been described experimentally by Battaglia-Mayer and colleagues (2000) on a study of early coding of reaching in parieto-occipital cortex. The authors have found that reach-related cells in this area have about three preferred directions of movement. Consequently, they characterized the neural response through a ’field of global tuning’, defined as the sector of the directional continuum within which all its preferred directions lie (e.g., approx. a quarter of a circle).

A second class of behavior is represented by the integrators, which are neurons that are commonly needed to integrate a large number of inputs in order to fire. If a neuron did not spike at the coincident arrival of the input signals, then due to the exponential decay of postsynaptic potentials, the accumulation of several excitatory potentials will be required before the postsynaptic spike will occur. Hence, these neurons’ activity strongly depends on the strength and the number of their lateral excitatory connections. In the motor condition, about 20% (i.e., $50/264$) of all neurons show a significant directional tuning while operating in the integration domain (see Section 6.1.5). These cells, referred to as *lateral neurons*, need to integrate both motor input and local lateral excitation in order to become active.

Another subclass of integrators is the neurons, which need to sum excitation from three sources: afferent, local, and long–range connections. Directionally tuned activity occurs in these motor neurons only during the visual condition. Accordingly, they were named *visually–related* neurons and they constitute about 10% of all motor units (i.e., $25/264$). A remark here is important with respect to the degree of directional tuning of these neurons. It was shown in Sections 6.1.5 and and 6.1.5 that most of the winners are broadly tuned to several directions of movement, while lateral neurons responses are less broader. In the light of the above discussion, we can say that the selectiveness or tuning of the neural response increases with the number of inputs integrated. Thus, the lateral neurons are significantly more directionally tuned than the winner neurons, and the visually–related neurons are optimally tuned (i.e., most selective) compared to both previous categories.

Even if our simulations represent a drastic simplification of the mechanisms involved in neural control of reaching, several hypotheses are presented here, with respect to the functional roles of neurons. Studies of visuomotor processing in the parieto–frontal network involved in reaching demonstrated the existence of various types of neural activity. During an instructed delay task followed by a pursuit tracking task, Johnson and colleagues (1999) have analyzed the directional discharge of neurons in monkey's premotor and primary motor cortex. From 240 neurons, in 132 cases, significant directional tuning was found for both the cue and track periods. In 26 neurons, directional tuning was found only during the cue period, and in another 54 the directional tuning was significant only in the track period. These neural behaviors can be classified as: (1) *visuomotor neurons*, whose activity show the co–existence of visual and movement control signals; (2) *signal neurons*, defined as motor neurons with visual properties, which respond transiently to the onset of the visual cue; (3) *movement-related neurons* that fire only for movement control.

We have obtained similar dynamics for cell activities in the motor network as a consequence of learning the visuomotor mapping task. From 264 motor neurons, about 60% developed visual properties, from both the winner and lateral neurons. In the absence in our model, of a behavioral task comparable to the instructed-delayed task, the *visuomotor neurons* are represented by those neurons which show directional activity during movement execution and under visual stimulation. The *signal neurons* correspond to our visually–related motor units (10%), which fire only during the visual stimulation period and are almost silent during movement execution. Finally, we have also found about 8% of motor neurons that are involved in the control of directional movement, but remain silent when stimulated by visual signals. These correspond to the *movement-related neurons* described experimentally.

Our results suggest that the formation of the motor network's response under visual guidance take place in a few steps. First, the *visuomotor neurons* are activated via the long-range inter-cortical connections by the visual directional signals. Note that in our model, the strength of these connections reaches a peak for a difference between the distal visual and motor neuron preferred directions of 30°. Therefore, the motor network response evoked in this way is broadly tuned to the desired direction of movement. Instead, the *signal* (i.e., visually–related) neurons that occur in the motor network are optimally tuned to the visual direction of movement. Hence, they can play an essential role in finely adjusting the motor population response (see Section 6.2.4). The next stages in the formation of the network response involve a successive propagation of activation, started by the visuomotor and signal neurons and mediated by the motor lateral neurons. The joint activity of all these neurons leads to the formation of the desired direction attractor.

Up to this point, we focused upon discussing the immediate implications of our modeling results. In the remainder of this section and along the next section we will outline the theoretical relevance of our models and their possible application to real systems.

7.1.4 Theoretical significance of the visuomotor mapping model

In the theoretical background of this thesis we reviewed a number of recent neurobiological theories of visuomotor control of movement (Sections 3.1.2, 3.2). At that point, we introduced four main hypotheses:

- The sensorimotor cycle has a unitary nature;
- The visuomotor transformation is achieved gradually, supported by the combinatorial properties of the neurons;
- The existence of common motor programs for eye and hand movements can reflect the operation of cortical computational principle of ’program re–use’
- The alignment of motor and visual networks for the correct transfer of information can be learned through a simple Hebbian learning principle.

While the last principle has been directly addressed in our simulations, the implications of our work for the other points might not be immediately clear. Therefore, we propose below an integrative view, which presents the theoretical relevance of our computational results.

The progressive match framework

Despite the simplification of the visuomotor transformation process in our model, we believe it illustrates a number of basic computational principles of this process. In particular we consider that our modeling work is relevant to the theoretical framework proposed by Burnod and colleagues (1999), even more so because, at the time we implemented our simulations (2000) we were not aware of their work.

The basic computational demand for reaching is met by the alignment of the visual and motor neural representations. This was achieved in our model by implementing a Hebbian–like learning mechanism, that correlates activity in spiking neurons with some feature selectivity, that is, in our case, directionality. Burnod's et al. (1999) model is based on the operation of an equivalent mechanism. The visuomotor transformation is described in terms of a progressive match of different sets of sensory information by neurons with tuning properties. Matching takes place gradually, in several combinatorial domains. In each domain, an identical computational mechanism operates, through the co-activation of matching neurons tuned to the same preferred attribute (position or direction).

The contribution of our model resides mainly in the fact that it is based on computations with spiking neurons and implements a realistic population coding of motor directionality. The operation of the computational mechanism implemented is not restricted to any particular area. It can align neural representation coding for any type of stimulus features (orientation, direction, pitch, etc.). Moreover, it implements learning on two of the four combinatorial domains described by the authors (Section 3.2.3). If we consider the training input to the self–organizing motor cortex as proprioceptive afferent feedback coming from activated muscles, than we have in the motor network organization, the learning of the first domain, which relates muscle dynamics and arm command. By relating the gaze direction and hand movement direction in the visuomotor mapping process, the system learns hand-tracking in the third combinatorial domain. Note that in our model, and the Burnod et al. (1999) framework, motor control, i.e., referred to as a motor babbling stage in the progressive match framework, is developed prior to visuomotor mapping learning.

The relevance of our model is even more significant, it we consider that in the Burnod and coworkers proposal most concepts were inspired by neurophysiological data. Instead, our model started out with a minimum set of architectural assumptions and a number of equivalent concepts emerged in the network, through development. For instance, in the progressive match model, a key element is the set of units, which integrate information on the sensorimotor axis: sensory units, motor units, and matching units. It is clear that the type and functionality of these units has been implemented according to the experimental data (Johnson et al., 1996; Caminiti et al., 1998; see data described in Section 7.1.3). Conversely, in our model these types of behavior have simply emerged during the self–organization process.

Moreover, in the absence of a working model, Burnod and colleagues focus upon the role of the matching units within the learning process of sensorimotor correlations. Instead, we have seen that correlated activity in two networks is a result of synchronization via long–range connections but also through intermediate synchronization with neurons in the same cell assembly. That is, learning takes place in a more distributed manner and involves matching (i.e., visuomotor) units as well as signal (i.e., sensory) units and movement-related units. We believe that an important further step into the realm of biologically inspired modeling of arm–reaching will be represented by the complete implementation of the progressive match architecture. Our efforts will be aimed at implementing more conceptual elements of this framework. A particular goal will be to obtain the formation of *condition* or set units, which are involved in maintaining the neural representation during delayed tasks (Johnson et al., 1999; Burnod et al., 1999).

Cortical Software Re–Use

Another theoretical framework within which we can discuss our results is the cortical software re-use theory (CSRU, Reilly, 1997; Reilly, 2001). Put simply, CSRU states that a general principle of creative cognition is the appropriation of computational programs from one domain and their application to another. For instance, CSRU proposes that perceptual binding can be seen as a collaborative process between cell assemblies that are equally well developed (Reilly, 2001). The style of computation is best viewed as a process of dynamical entrainment, involving the synchronization of firing patterns in reciprocally connected cortical areas. We believe that the neural mechanism for ’binding’ visual input to the relevant motor output for visuomotor mapping implements a similar type of computation. In CSRU terms, the visual neural activity is *re–used* to control the movement of the limb.

A stronger claim (i.e., hypothesis) is that the limb movement may make use of the eye motor activity, in a so called *motor–to–motor program re–use* (see Section 3.1.2, Metta et al., 1999;

Reilly and Marian, 2002). Note that this hypothesis states that visual (i.e., retinal) information is not necessarily required for the guidance of movement. The alternative to this process is that the motor program for eye movement is re-used to control limb movement in the same direction. This re-use has the advantage that eye-movement related signals can be read out at any processing stage, from various cortical and subcortical areas, and they are in head-coordinates, compared with the retinal information in Cartesian coordinates.

Some computational support for this hypothesis already exists. Metta and co–workers (1999) have implemented a model of visually guided reaching based on the alignment of the head map with the arm network (see Section 3.3.1). Similar results have been obtained by Marjanovic et al. (1996), who constructed a system that first learns to foveate a visual target and then re-use the saccade map to achieve ballistic reaching. Such modeling work provides a compelling example of how motor programs for eye movements or heading (see also Kolesnik and Streich, 2002) can support the development of visually guided reaching. From a developmental perspective, the program re–use makes much sense, as the 'software' for heading, eye movements and gaze focus develop priori to the control of reaching. Consequently, this hypothesis has a great potential in the robotics field.

The perception–action cycle

A final thought in this theoretical section, will be given to the unitary nature of the sensorimotor coupling. With respect to this rather abstract issue, much less can be inferred from our simple model of visuomotor mapping. One might say that our assumptions are rather a personal choice than scientifically proven facts. We have developed them while trying to find ways to implement the sensorimotor transformations.

The personal belief of the author of this thesis is that sensorimotor mapping is a fundamental, built–in property of any living nervous system. This means, that as a general principle of any nervous system, sensing–for–acting is implemented as one unitary computational operation. Hence, to characterize the task of transforming the sensory information from one modality to the motor output in another modality as the sensory–motor transformation 'problem', is perhaps to view it from the wrong angle. Moreover, because it is a general and ancient design principle of the nervous system, it is implemented at the lowest–level of the system and it is preserved in more complex variants of the system (i.e., primates or human brains). We consider that the apparent modularity of the human brain and the high degree of sophistication of its circuitry conceal the functioning of some low–level, built–in mechanisms which implement fundamental computational operations.

With respect to the transformation concept, we believe that it owes much to Cartesian dualism (i.e., external vs. internal world). We are probably on the same line with the critique made by a roboticist of the general tendency to assume that description and implementation of a system must be equivalent:

> We believe that classical and neo-classical AI make a fundamental error: both approaches make the mistake of assuming that because a description of reasoning / behavior / learning is possible at some level, then that *description must be made explicit and internal to any system* that carries out the reasoning / behavior / learning (Brooks et al., 1998, page 961, our emphasis).

Insights from our modeling work, and other more sophisticated models by Salinas and Abbott (1995), Burnod, Baraduc and colleagues (1999), showed us that a global complex operation, such as information transfer for the visual guidance of movement, may rely on the simple mechanism of correlated activity of single cells. The core of our model is based entirely on the ubiquitous feature of neurons to be directionally selective. A more general solution to the problem of sensorimotor transformation based on similar basic computational mechanisms was given by Salinas and Abbott (1995). Furthermore, it was discussed that combinatorial properties of directionally and positional selective neurons are the key element in the progressive match architecture for visually guided reaching (Burnod et al., 1999).

First, at the level of a single cell, several sources of information can be integrated along the sensorimotor axis (see Section 3.2.3). Second, the correlated activity of cells with the same preferred attributes (direction and position) can allow the correct transfer of information. Third, the coordinate transformation can be understood in terms of neural gain field, where the response of a neuron is a product of the receptive field and the linear gain field (see Section **??**). The point we want to make here, is that neural information processing systems rely heavily, on the computational features of single units.

In the computational neuroscience field the ideas outlined above, are well known. However, in the field of artificial intelligence and robotic applications, almost no attention is given to the properties of the neural model. For instance, a very succinct comparison between types of neural models existing reveals the following. A continuous rate-coding neuron, that represents the computational unit of the classical neural networks, can compute a temporal linear summation of inputs. A simplified model of the spiking neuron can in addition detect coincidence, can do multiplexing, and can compute in a temporal domain using delay codes (Maass, 1999). A compartmental model, which includes the dendritic tree, can perform spatial summation, nonlinear operations (division), can increase its discrimination and memory power up to thousand times that of the linear neuron, and can detect movement direction and binaural stimuli (Koch, 1999; Poirazi and Mel, 2000).

The above comparison represents a twofold argument. First, artificial systems may benefit enormously from paying more attention to the neurobiology of the living systems and to the way these implement perception and control of action. Second, the single neuron is indeed a very powerful computational device. Hence, we believe that by connecting these neurons in small size circuits, primitive operations such as the perception–action cycle can be implemented as an intrinsic feature of the system.

7.2 Applications and future directions

The central goal of this thesis was to offer a computational model that helps to bridge the gap between cognitive description and neural implementation of mental phenomena. That is, to understand the link between what a single computational element does and what many of them do when they function cooperatively. It was also pointed out, that understanding the way the brain organizes the control of movement can be largely beneficial to the design of artificial control systems. In general, the research dedicated to the understanding of computations in real nervous systems shares the same motivation: to apply what is learned from nature into the design of adaptive, intelligent, and eventually self-developing artificial systems. We will try to discuss, in this section, possible integration and future extensions of our models to systems of motor control. Up to the present these ideas are only at the stage of proposals. It remains future work to show to what degree their implementation can be beneficial.

7.2.1 Transforming plans in actions

A possible integration of the motor cortex organization model is within control systems based on a direction-mapping strategy. In this case the system implements a transformation from spatial trajectory to end-effector directions or rotations as opposed to end-effector positions (Bullock et al., 1993; Fiala, 1995; Ritter et al., 1989). The idea of mapping spatial into motor directions is supported by experimental evidence on the directional selectivity of cells in premotor and primary motor cortex (Caminiti et al., 1991; Georgopoulos et al., 1986; see Section 2.1.3). There is also psychophysical evidence for a direction-based rather than a position-based transformation coming from studies on blind reaching. These experiments suggest that the magnitude of the error is dependent on movement amplitude, rather than on the end–point alone (Fiala, 1995).

One of the most efficient implementations of direction mapping for visually guided reaching is the DIRECT model proposed by Bullock et al. (1993) (see Section 3.3.2). To control arm movements, the system first performs an integration of position and visual directional information into a position-direction map. In our model this corresponds to the motor network which learns to align motor and visual directional information. Furthermore, the DIRECT model focuses on learning the mapping from motor directions in body–centered coordinates into joint-rotations in joint coordinates. The accuracy of a three joint arm movement in 2D and 3D space strongly relies on the way the visual directional and positional information are correlated in the motor map. The authors argue that only a sharp tuning of each cell in the map, to a visual direction in a particular joint position, can ensure the accuracy of reaching movements. Even if they acknowledge that this is a significant deviation from the neurophysiological evidence (see population coding of directionality in motor cortex Section 2.1.3) they justify it by the fact that in the case of broadly tuned cells, the model fails to generate correct reaching movements.

The accuracy of visually guided movements is not an easy task for our model either, nor for any model grounded on neurophysiological data (see also Baraduc et al., 1999). Our analysis of the network organization leads us to believe that mapping accuracy is strongly influenced by two factors: the parameters of the horizontal connectivity pattern and the quality of the motor population codes for directions. By the parameters of lateral connections we mean the rate of connectivity, the profile of excitation and inhibition, and the plasticity rules. By the quality of the motor coding of direction we mean the stability and the distributed nature of the neural representation of each direction. In other words, the accuracy of visually guided reaching not only depends on how well the visuomotor coupling is learned, but also on how precise the motor control of the movement is itself. That is why we consider that a separation of the visuomotor development process into two stages may be beneficial. First, a motor babbling or motor learning phase is required, to ensure the motor cortex organizes for the precise control of movements. Learning the visuomotor correlations follows this.

To conclude, we believe that the advantages of integrating a developmental model similar to ours in a motor control system are many:

- It allows the emergence of population coding of directionality based on broadly tuned cells. This gives biological plausibility to the model, besides bringing all the benefits of a distributed representation (as opposed to a localist representation): flexibility, plasticity, reduced size.
- By exploring the way learning evolves in the lateral connections, our model allows the formation of stable attractors of movement directions, which in turn contributes to the accuracy of reaching.
- Only by modeling the developmental process, can one observe the emergence of different patterns of neural behaviors, with different functions in integrating and combining information, matching, conditioning or delaying the response.

Motor primitives and the equilibrium point hypothesis

Another direction to follow in order to translate our motor plans into actions, is to control the arm movement in conformity with the spring–like properties of muscles and reflex loops (Bizzi et al., 1992). This idea involves putting together the concept of motor primitives and the equilibrium point hypothesis, as an alternative to the inverse dynamics problem (i.e., the DIRECT model).

The motor primitives represent an appealing, rather theoretical concept, used by researchers on both artificial and biological motor control to reduce the complexity of movement generation to elementary units of action. They are defined as a set of *basis behaviors*, which are not further reducible to each other and which can be composed to produce the complete behavioral repertoire for the system (Brooks, 1986; Mataric, 1997; van Essen et al., 1996). On the other hand, the equilibrium point hypothesis is an experimentally–derived theory, according to which movement arises from shifts in the equilibrium positions of the joints. An equilibrium position is a consequence of the interaction of central neural commands, reflex mechanisms, muscle properties and external loads.

A recent extension of the equilibrium point theory, developed by Bizzi et al. (1991) and

Bizzi and Mussa-Ivaldi (1995) proposes that the muscles generate *convergent force fields* (i.e., equivalent to motor primitives), which direct the limb toward an equilibrium point in space. The vectorial superposition of these independent force fields can generate a vast repertoire of motor behaviors. The simulation studies of Mussa-Ivaldi (1999) have shown that by combining a small number of convergent force fields it is possible to reproduce the kinematics features of reaching arm movements.

The motor primitives proposed by Bizzi and colleagues suggest that spinal mechanisms can serve as substrate for the operation of motor cortical activity, in order to produce a directed movement of the limb. Georgopoulos (1996) proposed an integrative account of how directionally tuned motor cortical commands can be translated in the activation of muscles. In his view, this mapping can be accomplished by connecting a population of central cortical neurons through a set of motor inter–neurons, with a number of spinal populations associated with different motor primitives. Then, the preferred direction of a cortical cell emerges as the vectorial, weighted sum of the force fields that act on the hand at a certain position in space.

For our simplified version of motor control, this idea can be more beneficial and easier to implement than dealing with the complexity of a multi–staged architecture, such as the one implemented in the DIRECT system (i.e., with nine layers and learning at four different stages). It also allows a bi-directional transfer of information in the system: an efferent path, from the cortical motor network to the muscles and a re–afferent path, which brings feedback on the muscles activation to the motor cortex. We believe that this can be the appropriate framework for our future modeling work of motor directional map organization based on training input coming from muscles (see Section 7.2.2 below).

7.2.2 Future model

Our future model of visuomotor learning will be developed with a specific goal. That is, it will represent the neural controller of an avatar, endowed with simple vision, action upon objects, and proprioceptive feedback on the effect of its movements. The first step in this extension of the actual version of the model is to provide the motor network with proprioceptive feedback. In our view, the information available for the formation of the motor directional map is represented by afferent signals from those muscles that are involved in movement.

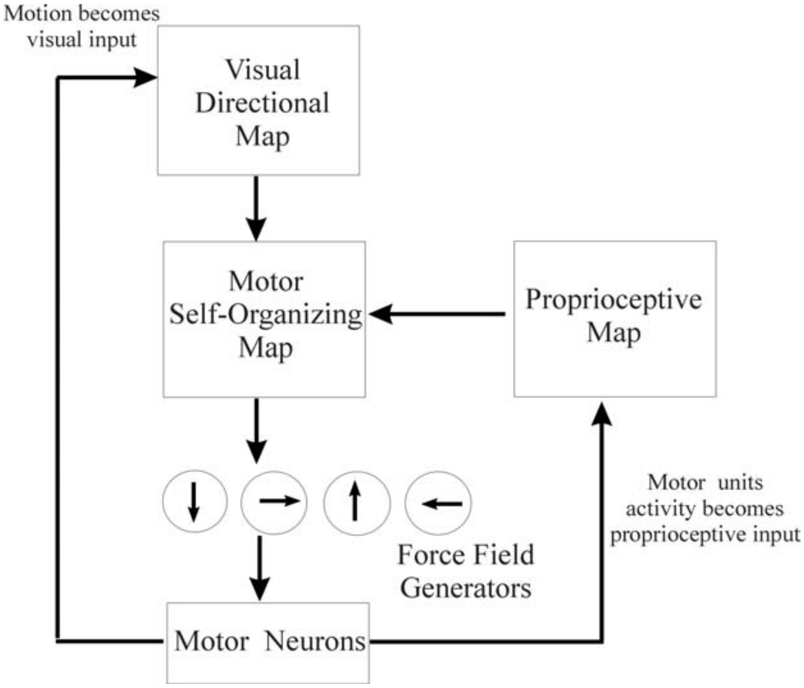


Figure 7.1: General architecture of the future work model. It includes several modules. A visual network of directionally tuned cells, with a retina-like input layer. A motor network which self-organizes for the control of movement direction. Its output is send to a set of force fields generators, which control the motor units in the muscles. Its input is provided by a proprioceptive network that receives afferent signals from the motor units involved in movement. These signals contain directional information, derived from the preferred directions of the muscles.

Recent research on the contribution of muscles to joint torque indicated that mono- and bi-articular muscles have different functional roles in the control of multi-joint movements (Bolhuis et al., 1998). Experimental data demonstrated that the activation of bi-articular muscles vary with the direction of force exerted, while mono-articular muscles show significant direction-dependent activation. Furthermore, it was shown that the mono-articular muscles have preferred movement directions, which cluster over subjects for both force direction and arm posture. To us, this data suggests that the motor unit activity may provide the directional information required for the organization of the cortical directional map.

Further, it is known that motor neurons in M1 make use of feedback information via afferent sensory pathways. At the contraction of muscles, information on which muscle contracts and how much tension it generates is fed–back to the motor cortex through the primary somatosensory cortex. Previous modeling work on this area was done by Chen and Reggia (1996) who studied the relation between the formation of motor and somatosensory feature maps for arm-muscle control. They have shown that an alignment of the neural representations of muscle activation in the two areas occurs through correlated-activity means. We intend to explore this alignment in the case of the motor and somatosensory networks encoding directional information, derived from muscle tensions and contractions. The proprioceptive network will be mainly used in order to transmit the motor units' activations as input to the motor cortical network, in the hope that a directional motor map will form. The proposed architecture is outlined in Figure 7.2.2.

7.2.3 The imitation challenge

At the end of this chapter, we want to turn our attention back to the initial motivation of this thesis, that is, the neonatal imitation phenomena. Here, we discuss this issue within the more general context of imitative behaviors, which represent one of our future modeling goals. This is because imitation plays a central role in human development and is currently being explored as a powerful, alternative mechanism for teaching robots (Schaal, 1999; Dautenhahn, 2000; Billard and Mataric, 2001). Hence, we would like to abstract some relevant ideas to the imitation modeling, from the work presented in this thesis.

The challenge posed by neonatal imitation is to understand the capacity of infants as young as few hours, to imitate facial expressions, such as tongue protrusion and mouth opening (Meltzoff and Moore, 1977). Meltzoff and Moore (1999) proposed that a key element in explaining the mechanisms of this behavior is that the imitative act can be differentiated into the *body part* and the *movement performed*. They consider that evidence suggests that neonates select what body part to use before they have determined what to do with it. Further, finding the correct action on the organ involves more effortful behavior preceded by a series of searching movements.

Certainly, being capable of organ identification is probably the most astonishing hypothesis regarding the newborns innate capabilities. Interestingly, this problem is less controversial when it comes to implementation in artificial systems. Generally, artificial systems avoid this problem, either by dealing with the imitation of a single body part (i.e., arm) or by implying a similar physical morphology between the demonstrator and imitator (Billard and Mataric, 2001). Such a built–in capacity is face detection, based on a direct mapping between the organ representation and the corresponding motor control area (Breazal and Scassellati, 2002).

With respect to limb action identification, the recent discovery of the *mirror neural system* in the monkey's premotor cortex (Rizzolatti et al., 1999) has been proposed as the system responsible for the linkage of self–generated and observed actions (Arbib, 2002). The interpretation is that mirror neurons can allow the matching of the neural command for an action with the neural code for the recognition of the same action executed by another individual. Mirror neurons have been observed for reaching and grasping actions, and they are highly specialized for certain types of movements. It is possible that these neuron properties are innate, similar to face detection capacity, hence they may explain, up to a point, the neonatal imitation of gestures. Thus, they can facilitate recognition of hand manipulation and may be involved in mapping the hand sight into the hand self–motion.

However, even if mirror neurons functionality has been recently incorporated in several imitation modeling proposals (Billard and Mataric, 2001; Maistros and Hayes, 2002; Metta and Fitzpatrick, 2002), few attempts have been made, so far, to understand the way they develop such a highly specialized *matching* property (Arbib, 2001). We believe that exploring the way mirror neurons' functionality emerges can provide insights into their 'true' role in imitation and language development. We consider that an improved version of our visuomotor model, which is already capable of showing emergence of multi–modal neural behaviors and to give rise to different dynamics of the neural response, is in a good position to explore this topic.

With respect to the *movement component* of the imitative act, most researchers agree that it is not innately specified, but up to the present it is not known yet how this mapping is achieved with such specificity. Here, it is important to delimit the existence of two developmental stages in imitation. First is neonatal imitation, which mainly consists of facial gesture imitation and is probably supported by a subcortical system (Atkinson, 2000). This is followed by the emergence of a true imitative behavior, which occurs after a few months of post-natal life and is marked by the acquisition of eye-hand coordination (i.e., at about 3-4 months) (Butterworth, 1999; Atkinson, 2000).

Our view is that imitation in the former case can be best explained in the terms of dynamical systems theory. From this perspective, development is self-organizing around various attractors on which the configurations of the system tends to converge (Butterworth, 1999). From this view, the prenatal movements provide experience to link muscles activations to body configurations. In our terms (i.e., the terms of the progressive match architecture), this stage corresponds to learning in the first combinatorial domain, where co–activation of motor commands and organ configuration (i.e., muscles activation) establish the foundation of proprioceptive control of movement. What this stage does, it creates a set of attractors in the movement–somatosensory space.

Further, seeing the protrusion of the tongue between lips means, in terms of object identification, that both lips and tongues visual areas are activated. These are further mapped to the corresponding motor areas. Here, the oscillatory activity in the two areas can only evolve towards one of the existent attractors. One attractor, which comprises neural activity for both lips and tongue movement, is the one that places the tongue between lips. Note that, the other imitated behaviors, such as mouth opening or eye–movements are even simpler, as the activity here involves only one area and elicits a limited number of possible actions. The search in the space of possible behaviors (i.e., see above the effortful search of the correct action) is equivalent to the formation of the desired action attractor. We can compare this process, with the formation in the motor network of the directional response in the presence of visual stimulation. The task in the case of neonatal imitation is more difficult, because it does not appear to involve any learning of the visuomotor connection, but is actually learnt on–line, resulting in the convergent correction of the movement. This is possible, we believe, due to the combinatorial properties of the neurons, which allow proprioceptive, motor and visual information to be matched.

In the latter case (i.e., after 3 month of postnatal life), imitation represents the result of a self-organizing learning process. The beginning of true imitative behavior accompanies the emergence of eye-hand coordination. This suggests that both processes require the development of equivalent neurobiological mechanisms. Hand–eye coordination begins to develop between 2-4 months, inaugurating a period of trial–and–error practice at sighting objects and grabbing at them. When executing actions, infants perceive and learn contingencies between the motor activity and the visual image of the movements. Our work focused upon the modeling of this behavioral scenario in order to develop visuomotor coordination. We believe that the operation of the developed computational mechanism can establish the foundations for imitative behavior.

References

Abbott, L. (1994). Decoding neuronal firing and modelling neural networks. *Quarterly Review of Biophysics*, *27*, 291–331.

Abeles, M. (1982). Role of cortical neuron: integrator or coincidence detector? *Isr. Journal Medcine Science*, *18*, 83–92.

Abeles, M., Bergman, H., Margalit, E., & Vaadia, E. (1993). Spatiotemporal firing patterns in the frontal cortex of behaving monkeys. *Journal of Neurophysiology*, *70*(4).

Abeles, M., & Gat, I. (2001). Detecting precise firing sequences in experimental data. *Journal of Neuroscience Methods*, *107*(1–2), 141–154.

Adrian, E. (1926). The impulses produced by sensory nerve endings. *Journal of Physiology (London)*, *61*, 49–72.

Agmon-Snir, H., Carr, C., & Rinzel, J. (1998). The role of dendrites in auditory coincidence detection. *Nature*, *393*, 268–272.

Amirikian, B., & Georgopulos, G. (2000). Directional tuning profiles of motor cortical cells. *Neuroscience Research*, *36*(1), 73–79.

Andersen, R., Batista, A., Snyder, L., & Cohen, C. B. Y. (2000). Programming to look and reach in the posterior parietal cortex. In M. Gazzaniga (Ed.), *The cognitive neuroscience* (Second ed.). MIT Press.

Andersen, R., Snyder, L., Bradley, D., & Xing, J. (1997). Multimodal representation of space in the posterior parietal conex and its use in planning movements. *Annual Review of Neuroscience*, *20*, 303–330.

Arbib, M. (1987). Levels of modeling of mechanisms of visually guided behaviour. *Behavioural and Brain Sciences*, *10*, 407–465.

Arbib, M. (2001). *Mirror neurons and the neural basis for action recognition: A computational model.*

Arbib, M. (2002). The mirror system, imitation, and the evolution of language. In C. Nehaniv & K. Dautenhahn (Eds.), *Imitation in animals and artifacts.* MIT Press.

Atkinson, J. (2000). *The developing visual brain.* Oxford University Press.

Balkenius, C., Zlatev, J., Kozima, H., Dautenhahn, K., & Breazeal, C. (Eds.). (2001). *Proceedings of the first international workshop on epigenetic robotics.* Lund, Sweden.

Baraduc, P., & Guigon, E. (2002). Population computation of vectorial transformations. *Neural Computation*, *14*(4), 845–871.

Baraduc, P., Guigon, E., & Burnod, Y. (1999). Where does the population vector of motor cortical cells point during arm reaching movements? In M. Jordan, M. Kearns, & S. Solla (Eds.), *Advances in neural information processing systems* (Vol. 10, pp. 83–89). MIT Press.

Battaglia-Mayer, A., Ferraina, S., Mitsuda, T., Marconi, B., Genovesio, A., Onorati, P., Lacquaniti, F., & Caminiti, R. (2000). Early coding of reaching in the parieto-occipital cortex. *Journal of Neurophysiology*, *83*, 2374–2391.

Bednar, J. A., & Miikkulainen, R. (2003). Self-organization of spatiotemporal receptive fields and laterally connected direction and orientation maps. In *Computational neuroscience: Trends in research 2003 (proceedings of cns*2002).*

Berthouze, L., & Kuniyoshi, Y. (1998). Emergence and categorization of coordinated visual behavior through embodied interaction. *Machine Learning*, *31*, 187–200.

Bialek, W., Rieke, F., Steveninck, R. V., & Warland, R. (1991). Reading a neural code. *Science*, *252*(5014), 1854–1857.

Billard, A., & Mataric, M. (2001). Learning human arm movements by imitation: Evaluation of biologicallyinspired connectionist architecture. *Robotics and Autonomous Systems*, *37*(2–3), 145–160.

Bizzi, E., Hogan, N., Mussa-Ivaldi, F., & Giszter, S. (1992). Does the nervous system use equilibrium-point control to guide single and multiple joint movements? *Behavioral and Brain Sciences*, *15*, 601–613.

Bizzi, E., & Mussa-Ivaldi, F. (2000). Toward a neurobiology of coordinate transformations. In *The cognitive neuroscience* (Second ed.). Cambridge, Massachusetts: MIT Press.

Bizzi, E., Mussa-Ivaldi, F., & Giszter, S. (1991). Computations underlying the execution of movement: A biological perspective. *Science*, *253*, 287–291.

Blais, B., Cooper, L., & Shouval, H. (2000). Formation of direction selectivity in natural scene environments. *Neural Computation*, *12*(5), 1057–1066.

Blasdel, G. (1992). Orientation selectivity, preference, and continuity in monkey striate cortex. *Journal of Neuroscience*, *12*, 3139–3161.

Bolhuis, B., Gielen, C., & Schenau, G. V. I. (1998). Activation patterns of mono- and bi-articular arm muscles as a function of force and movement direction of the wrist in humans. *Journal of Physiology*, *508*(1), 313–324.

Bower, J. (1997). Is the cerebellum sensory for motor's sake, or motor for sensory's sake: the view from the whiskers of a rat. *Progress in Brain Research*, *4*, 463–496.

Bower, J., & Beeman, D. (1998). *The book of GENESIS: Exploring realistic neural models with the GEneral NEural SImulation System.* Springer-Verlag.

Braitenberg, V., & Schuz, A. (1998). *Cortex: Statistics and geometry of neuronal connectivity.* Berlin: Springer-Verlag.

Breazeal, C., & Scassellati, B. (2002). Issues in building robots that imitate people, imitation in animal and artifacts. In K. Dautenhahn & C. Nehaniv (Eds.), *Imitation in animals and artifacts.* Cambridge, Massachusetts: MIT Press.

Brettle, D., & Niebur, E. (1994). Detailed parallel simulation of a biological neuronal network. *Computational Science & Engineering*, *1*(4), 31–43.

Brooks, R. A., Ferrell, C., Irie, R., Kemp, C., Marjanovic, M., Scassellati, B., & Williamson, M. (1998). Alternative essences of intelligence. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)* (pp. 961–968). AAAI Press.

Budd, J., & Kisvárday, Z. (2001). Local lateral connectivity of inhibitory clutch cells in layer 4 of cat visual cortex (area 17). *Experimental Brain Research*, *140*(2), 245–250.

Bugmann, G. (1991). Summation and multiplication: two distinct operation domains of leaky integrate-and-fire neurons. *Network*, *2*, 489–509.

Bugmann, G., Christodoulou, C., & Taylor, J. (1997). Role of temporal integration and fluctuation detection in the highly irregular firing of a leaky integrator neuron model with partial reset. *Neural Computation*, *9*, 985–1000.

Bugmann, G., & Taylor, J. (1993). A stochastic short-term memory using a pram neuron and its potential applications. In *Proceedings British Neural Network Society Meeting (BNNS'93)*. Birmingham.

Bullock, D., Cisek, P., & Grossberg, S. (1995). *Cortical networks for control of voluntary arm movements under variable force conditions* (Tech. Rep. No. TR-95-019). Boston University, Department of Cognitive and Neural Systems.

Bullock, D., & Grossberg, S. (1989). VITE and FLETE: Neural modules for trajectory formation and postural control. In W. Hershberger (Ed.), *Volitional action* (pp. 253–298). Amsterdam: North-Holland / Elsevier.

Bullock, D., Grossberg, S., & Guenther, F. (1993). A self-organizing neural model of motor equivalent reaching and tool use by a multijoint arm. *Journal of Cognitive Neuroscience*, *5*, 408–435.

Burnod, Y., Baraduc, P., Battaglia-Mayer, A., Guigon, E., Koechlin, E., Ferraina, S., Lacquaniti, F., & Caminiti, R. (1999). Parieto-frontal coding of reaching: an integrated framework. *Experimental Brain Research*, *129*, 325–346.

Burnod, Y., Grandguillaume, P., Otto, I., Ferraina, S., Johnson, P., & Caminiti, R. (1992). Visuomotor transformations underlying arm movements toward visual targets: a neural network model of cerebral cortical operations. *Journal of Neuroscience*, *12*(4), 1435–1453.

Butterworth, G. (1999). Neonatal imitation: existence, mechanisms and motives. In J. Nadel & G. Butterworth (Eds.), *Imitation in infancy*. Cambridge: Cambridge University Press.

Caminiti, R., Ferraina, S., & Johnson, P. (1996). The sources of visual information to the primate frontal lobe: a novel role for the superior parietal lobule. *Cerebral Cortex*, *6*(3), 319–328.

Caminiti, R., Ferraina, S., & Mayer, A. (1998). Visuomotor transformations: early cortical mechanisms of reaching. *Current Opinion in Neurobiology*, *8*, 753–761.

Caminiti, R., Johnson, P., & Burnod, Y. (Eds.). (1992). *Control of arm movement in space: Neurophysiological and computational approaches*. Springer-Verlag.

Caminiti, R., Johnson, P., Ferraina, S., & Burnod, Y. (1992). Reaching towards visual targets. i. neurophysiological studies. In R. Caminiti, P. Johnson, & Y. Burnod (Eds.), *Control of arm movement in space: Neurophysiological and computational approaches*. Springer-Verlag.

Caminiti, R., Johnson, P., Galli, C., & Burnod, S. F. andY. (1991). Making arm movements within different parts of space: The premotor and motor cortical representations of a coordinate system for reaching to visual targets. *Journal of Neuroscience*, *11*, 1182–1197.

Cariani, P. (1997). Temporal coding of sensory information. In J. M. Bower (Ed.), *Computational neuroscience: Trends in research* (pp. 591–598). New York: Plenum.

Cash, S., & Yuste, R. (1999). Linear summation of excitatory inputs by ca1 pyramidal neurons. *Neuron*, *22*(2), 383–394.

Chauvet, G. A. (2002). Editorial. *Journal of Integrative Neuroscience*, *1*(1).

Chen, Y., & Reggia, J. (1996). Alignment of coexisting cortical maps in a motor control model. *Neural Computation*, *8*(4), 731–755.

Choe, Y., & Miikkulainen, R. (1998). Self-organization and segmentation in a laterally connected orientation map of spiking neurons. *Neurocomputing*, *21*, 51–60.

Churchland, P. S., & Sejnowski, T. J. (1992). *The computational brain, computational neuroscience.* Cambridge, Massachusetts: MIT Press.

Claverol, E., Brown, A., & Chad, J. (2002). Discrete simulation of large aggregates of neurons. *Neurocomputing*, *47*, 277–297.

Cleeremans, A., & French, R. (1996). From chicken squawking to cognition: Levels of description and the computational approach in psychology. *Psychologica Belgica*, *36*.

Cliff, D. (1991). Computational neuroethology: A provisional manifesto. In J.-A. Meyer & S. Wilson (Eds.), *From animals to animats: Proceedings of the First International Conference on the Simulation of Adaptive Behaviour (SAB90)* (pp. 29–39). MIT Press Bradford Books.

Cliff, D. (1995). Neuroethology, computational. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks* (pp. 626–630). MIT Press.

Colby, C., & Duhamel, J. (1996). Spatial representations for action in parietal cortex. *Cognitive Brain Research*, *5*, 105–115.

Crammond, D., & Kalaska, J. (1994). Modulation of preparatory neuronal activity in dorsal premotor cortex due to stimulus-response compatibility. *Journal of Neurophsyiology*, *71*(3), 1281–1284.

Crick, F. (1994). *The astonishing hypothesis: the scientific search for the soul, Charles Scribner's sons.* New York: Macmillan Publishing Company.

de Schutter, E. (2001). *Computational neuroscience: Realistic modeling for experimentalists.* Boca Raton, Florida: CRC Press.

de Schutter, E., & Bower, J. (1994a). An active membrane model of the cerebellar purkinje cell. I. simulation of current clamps in slice. *Journal of Neurophysiology*, *71*, 375–400.

de Schutter, E., & Bower, J. (1994b). An active membrane model of the cerebellar purkinje cell: II. simulation of synaptic responses. *Journal of Neurophysiology*, *71*, 401–419.

Delorme, A., Gautrais, J., VanRullen, R., & Thorpe, S. (1999). SpikeNET: A simulator for modeling large networks of integrate and fire neurons. *Neurocomputing*, *26–27*, 989–996.

Delorme, A., & Thorpe, S. (2001). *Event-driven simulation of large networks of spiking neurons.* submitted.

Depew, D., & Weber, B. (1999). Self-organization. In *MIT encyclopedia of cognitive science* (pp. 737–739). MIT Press.

Destexhe, A., Mainen, Z., & Sejnowski, T. (1998). Kinetic models of synaptic transmission. In C. Koch & I. Segev (Eds.), *Methods in neuronal modeling* (Second ed., pp. 1–26). Cambridge, Massachusetts: MIT Press.

Diesmann, M., Gewaltig, M.-O., & Aertsen, A. (1999). Stable propagation of synchronous spiking in cortical neural networks. *Nature*, *402*, 529–533.

Douglas, R., Koch, C., Mahowald, M., Martin, K., , & Suarez, H. H. (1995). Recurrent excitation in neocortical circuits. *Science*, *269*, 981–985.

Douglas, R. J., & Martin, K. A. C. (1991). A functional microcircuit for cat visual cortex. *Journal of Physiology*, *440*, 735–769.

Douglas, R. J., Martin, K. A. C., & Whitteridge, D. (1991). An intracellular analysis of the visual responses of neurones in cat visual cortex. *Journal of Physiology*, *440*, 659–696.

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

Eckhorn, R., Bauer, R., Jordon, W., Brosch, M., Kruse, W., Munk, M., & Reitboeck, H. (1988). Coherent oscillations: a mechanism of feature linking in the visual cortex? *Biological Cybernetics*, *60*, 121–130.

Elias, J., & Northmore, D. (1999). Building silicon nervous systems with dendritic tree neuromorphs. In W. Maass & M. Bishop (Eds.), *Pulsed neural networks* (pp. 3–53). MIT Press.

Engel, A., Konig, P., Kreiter, A., & Singer, W. (1991). Interhemispheric synchronization of oscillatory neuronal responses in cat visual cortex. *Science*, *252*, 1177–1179.

Engel, K., Anderson, J., & Soechting, J. (2000). Similarity in the response of smooth pursuit and manual tracking to a change in the direction of target motion. *Journal of Neurophysiology*, *84*(3), 1149–1156.

Erwin, E., Obermayer, K., & Schulten, K. (1995). Models of orientation and ocular dominance columns in the visual cortex: A critical comparison. *Neural Computation*, *7*, 425–468.

Eurich, C. W., & Wilke, S. D. (2000). Multi-dimensional encoding strategy of spiking neurons. *Neural Computation*, *12*(7), 1519–1529.

Farkas, I., & Miikkulaine, R. (1999). Modeling the self-organization of directional selectivity in the primary visual cortex. In *Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN'99, edinburgh)* (pp. 251–256). Berlin: Springer.

Feldman, A. G. (1986). Once more on the equilibrium-point hypothesis (lambda model) for motor contro. *Journal of Motor Behavior*, *18*, 17–54.

Ferscha, A., & Tripathi, S. (1994). *Parallel and distributed simulation of discrete event systems* (Tech. Rep. No. CS-TR-3336). University of Maryland.

Fiala, J. (1995). *Neural network models of motor timing and coordination.* Unpublished doctoral dissertation, Department of Cognitive and Neural Systems, Boston University.

Flash, T., & Sejnowsky, T. (2001). Computational approaches to motor control. *Current Opinion in Neurobiology*, *11*, 655–662.

Fohlmeister, C., Gerstner, W., Ritz, R., & Hemmen, J. van. (1995). Spontaneous excitations in the visual cortex: stripes, spirals, rings and collective bursts. *Neural Computation*, *7*, 905–914.

Frackowiak, R., Friston, K., Frith, C., Dolan, R., & Mazziotta, J. (1997). *Human brain function.* San Diego, California: Academic Press.

Fujimoto, Y., Fukuda, N., & Akabane, T. (1992). Massively parallel architectures for large scale neural network simulations. *IEEE Transactions on Neural Networks*, *3*(6), 876–888.

Gaudiano, P., & Grossberg, S. (1991). Vector associative maps: Unsupervised real-time error-based learning and control of movement trajectories. *Neural Networks*, *4*, 147–183.

Gawne, T., Kjaer, T., & Richmond, B. (1996). Latency: another potential code for feature binding in striate cortex. *Journal of Neurophysiology*, *76*(2), 1356–1360.

Georgopoulos, A. (1996). On the translation of directional motor cortical commands to activation of muscles via spinal interneuronal systems. *Cognitive Brain Research*, *3*, 151–155.

Georgopoulos, A., Kalaska, J., Caminiti, R., & Massey, J. (1982). On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *Journal of Neuroscience*, *2*(11), 1527–1537.

Georgopoulos, A., Kalaska, J., Crutcher, M., & Massey, J. (1984). The representation of movement direction in the motor cortex: Single cell and population studies. In G. Edelman, W. Goll, & W. Cowan (Eds.), *Dynamic aspects of neocortical function* (pp. 501–524). MIT Press.

Georgopoulos, A., Pellizzer, G., Poliakov, A., & Schieber, M. (1999). Neural coding of finger and wrist movements. *Journal of Computational Neuroscience*, *6*(3), 279–288.

Georgopoulos, A., Schwartz, A., & Kettner, R. (1986). Neuronal population coding of movement direction. *Science*, *233*, 1416–1419.

Georgopoulos, A., Taira, M., & Lukashin, A. (1993). Cognitive neurophysiology of the motor cortex. *Science*, *260*, 47-52.

Gerstner, W. (1991). Associative memory in a network of 'biological' neurons. In R. Lippmann, J. Moody, & D. Touretzky (Eds.), *Advances in neural information processing systems* (pp. 84–90). San Mateo, California: Morgan Kaufmann Publishers.

Gerstner, W. (1995). Time structure of the activity in neural network models. *Physical Review E*, *51*, 738–758.

Gerstner, W. (1999). Spiking neurons. In W. Maas & C.M.Bishop (Eds.), *Pulsed neural networks* (pp. 3–53). MIT Press.

Gerstner, W., Kempter, R., van Hemmen, J., & Wagner, H. (1996). A neuronal learning rule for sub-millisecond temporal coding. *Nature*, *383*, 76–78.

Gerstner, W., Kempter, R., van Hemmen, J., & Wagner, H. (1997). A developmental learning rule for coincidence tuning in the barn owl auditory system. In *Computational neuroscience: Trends in research 1997* (pp. 665–669). New York: Plenum Press.

Gerstner, W., & Kistler, W. (2002). *Spiking neuron models. single neurons, populations, plasticity*. Cambridge University Press.

Gerstner, W., Ritz, R., & Hemmen, J. van. (1993). A biologically motivated and analytically soluble model of collective oscillations in the cortex. I. theory of weak locking. *Biological Cybernetics*, *68*, 363–374.

Gerstner, W., & van Hemmen, J. (1994). How to describe neural activity – spikes, rates, or assemblies? In J. Cowan, G. T. G, & J. Alspector (Eds.), *Advances in neural information processing systems* (pp. 463–470). San Francisco: Morgan Kaufmann Publishers.

Ghez, C., Krakauer, J., Sainburg, R., & Ghilardi, M. (2000). Spatial representations and internal models of limb dynamics in motor learning. In M. Gazzaniga (Ed.), *The cognitive neuroscience* (Second ed.). MIT Press.

Gielen, C. (2001). Population coding: Efficiency and interpretation of neuronal activity. In F. Moss & S. Gielen (Eds.), *Handbook of biological physics* (Vol. 4, pp. 853–886). Amsterdam: Elsevier.

Gilbert, C., & Wiesel, T. (1992). Receptive field dynamics in adult primary visual cortex. *Nature*, *356*, 150–152.

Glickstein, M. (2000). How are visual areas of the brain connected to motor areas for the sensory guidance of movement? *Trends in Neuroscience*, *23*, 613–617.

Goddard, N., Hood, G., Howell, F., M.Hines, & de Schutter, E. (2001). NEOSIM: Portable large-scale plug and play modelling. *Neurocomputing*, *38–40*, 1657–1661.

Goodale, M., & Milner, A. (1992). Separate visual pathways for perception and action. *Trends in Neuroscience*, *15*(1), 20–25.

Goodale, M. A. (1992). Visuomotor modules in the vertebrate brain. *Canadian Journal of Physiology and Pharmacology*, *74*, 390–400.

Goodale, M. A. (2000). Perception and action in human visual system. In M. Gazzaniga (Ed.), *The cognitive neuroscience* (Second ed.). MIT Press.

Grassmann, C., & Anlauf, K. (1998). Event driven simulation of spiking neural networks. In *NC'98* (pp. 100–105).

Grinvald, A., Frostig, R., & Lieke, E. (1988). Optical imaging of neuronal activity. *Physiological Reviews*, *68*, 1285–1366.

Grossberg, S. (1998). Birth of a learning law. *INNS/ENNS/JNNS Newsletter*, *21*, 1–4.

Hatsopoulos, N., Ojakangas, C., Paninski, L., & Donoghue, J. (1998). Information about movement direction obtained by synchronous activity of motor cortical neurons. In *Proceedings of the national academy of sciences usa* (Vol. 95, pp. 15706–15711).

Haykin, S. (1994). *Neural networks.* Macmillan College Publishing Company Inc.

Hebb, D. (1949). *The organization of human behavior.* New York: Wiley.

Heil, P. (1997). Auditory cortical onset responses revisited. I. first-spike timing. *Journal of Neurophysiology*, *77*, 2616–2641.

Heil, P., & Irvine, D. (1996). On determinants of first-spike latency in auditory cortex. *NeuroReport*, *7*, 2616–2641.

Heil, P., & Irvine, D. (1997). First-spike timing of auditory-nerve fibers and comparison with auditory cortex. *Journal of Neurophysiology*, *78*, 2438–2454.

Hess, G., & Donoghue, J. (1994). Long-term potentiation of horizontal connections provides a mechanism to reorganize cortical maps. *Journal of Neurophysiology*, *71*, 2543–2547.

Hildreth, E. C., & Hollerbach, J. (1985). *The computational approach to vision and motor control* (AI Memo 846). Massachusetts Institute of Technology.

Hines, M. (1993). NEURON–a program for simulation of nerve equations. In F. Eeckman (Ed.), *Neural systems: Analysis and modeling* (pp. 127–136). Norwell, Massachusetts: Kluwer.

Hines, M. (1998). The neurosimulator NEURON. In C. Koch & I. Segev (Eds.), *Methods in neuronal modeling* (pp. 129–136). MIT Press.

Hines, M., & Carnevale, N. (1995). Computer modeling methods for neurons. In M. Arbib (Ed.), *The handbook of brain theory and neural networks* (pp. 226–230). MIT Press.

Hines, M., & Carnevale, N. (2001). NEURON: a tool for neuroscientists. *The Neuroscientist*, *7*, 123–135.

Hodgkin, A., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, *117*, 500–544.

Hopfield, J. (1995). Pattern recognition computation using action potential timing for stimulus representation. *Nature*, *376*, 33–36.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, *160*, 106–154.

Hubel, D. H., & Wiesel, T. N. (1970). The period of susceptibility to the physiological effects of unilateral eye closure in kittens. *Journal of Physiology*, *206*, 419–436.

Izhikevich, E. (2001). Resonate-and-fire neurons. *Neural Networks*, *14*, 883–894.

Jahnke, A., Roth, U., & Klar, H. (1995). Towards efficient hardware for spike-processing neural networks. In *Proceedings WCNN'95, washington, usa.*

Jahnke, A., Roth, U., & Schoenauer, T. (1999). Digital simulation of spiking neural networks. In W. Maas & C. Bishop (Eds.), *Pulsed neural networks.* MIT Press.

Jahnke, A., Schonauer, T., Roth, U., Mohraz, K., & Klar, H. (1998). Simulation of spiking neural networks on different hardware platforms. In W. Gerstner, A. Germond, M. Hasler, & J.-D. Nicoud (Eds.), *Artificial neural networks - ICANN-97* (pp. 1187–1192). Springer-Verlag.

Johansson, R., Westling, G., Backstrom, A., & Flanagan, J. (2001). Eye-hand coordination in object manipulation. *Journal of Neuroscience*, *21*, 6917–6932.

Johnson, M., Coltz, J., Hagen, M., & Ebner, T. (1999). Visuomotor processing as reflected in the directional discharge of premotor and primary motoneurons. *Journal of Neurophysiology*, *81*, 875–894.

Johnson, M., Mason, C., & Ebner, T. (2001). Central processes for the multiparametric control of arm movements in primates. *Current Opinion in Neurobiology*, *11*, 684–688.

Johnson, P., Ferraina, S., Bianchi, L., & Caminiti, R. (1996). Cortical networks for visual reaching: physiological and anatomical organization of frontal and parietal lobe arm regions. *Cerebral Cortex*, *6*, 102–119.

Jordan, M. (1993). Computational aspects of motor control and motor learning. In H. Heuer & S. Keele (Eds.), *Handbook of perception and action: Motor skills.* New York: Academic Press.

Jordan, M., & Wolpert, D. (2000). Computational motor control. In M. Gazzaniga (Ed.), *The cognitive neuroscience* (Second ed.). A Bradford Book, MIT Press.

Kakei, S., Hoffman, D., & Strick, P. (1999). Muscle and movement representation in the primary motor cortex. *Science*, *285*, 2136–2139.

Kalaska, J., Scott, S., Cisek, P., & Sergio, L. (1997). Cortical control of reaching movements. *Current Opinions in Neurobiology*, *7*, 849–859.

Kalaska, J. F. (1995). Reaching movements: Implications of connectionist models. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks.* A Bradford Book, The MIT Press.

Kalaska, J. F., Crammond, D., Cohen, D., Prud'homme, M., & M.L.Hyde. (1992). Comparison of cell discharge in motor, premotor, and parietal cortex during reaching. In R. Caminiti, P. Johnson, & Y. Burnod (Eds.), *Control of arm movement in space* (pp. 103–127). Springer.

Kandel, E., JH, J. S., & Jessell, T. (Eds.). (2000). *Principles of neural science* (Fourth ed.). McGraw Hill.

Kelso, J. (1988). Dynamic patterns. In J. Kelso, A. Mandell, & M. Shlesinger (Eds.), *Dynamic patterns in complex systems.* Singapore: World Scientific.

Kelso, J. (1995). *Dynamic patterns: The self-organization of brain and behavior.* The MIT Press.

Kempter, R., Gerstner, W., & van Hemmen, J. L. (1997). Reduction of Hodgkin-Huxley equations to a threshold model. *Neural Computation*, *48*, 1069–100.

Kempter, R., Gerstner, W., & van Hemmen, J. L. (1998). How the threshold of a neuron determines its capacity for coincidence detection. *BioSystems*, *48*, 105–112.

Koch, C. (1999). *Biophysics of computation: Information processing in single neurons.* New York: Oxford University Press.

Koch, C., & Segev, I. (Eds.). (1998). *Methods in neuronal modeling: From ions to networks* (Second ed.). MIT Press.

Kohonen, T. (1984). *Self-organisation and associative memory.* Berlin: Springer-Verlag.

Kohonen, T. (1995). *Self-organizing maps.* Berlin: Springer.

Kolesnik, M., & Streich, H. (2002). Visual orientation and motion control of MAKRO- adaptation to the sewer environment. In B. Hallam, D. Floreano, J. Hallam, & J.-A. Meyer (Eds.), *From animals to animats 7. proceedings of the 7th International conference on Simulation of Adaptive Behavior* (pp. 62–69). MIT Press.

Kugiumutzakis, G. (1999). Genesis and development of early infant mimesis to facial and vocal models. In J. Nadel & G. Butterworth (Eds.), *Imitation in infancy.* Cambridge University Press.

Kuniyoshi, Y., & Berthouze, L. (1998). Neural learning of embodied interaction dynamics. *Neural Networks*, *11*, 1259–1276.

Kuniyoshi, Y., Nagakubo, A., Berthouze, L., & Cheng, G. (1999). Interfacing agents through boundaries of interaction dynamics. In *Proceedings of 9th International Symposium on Robotics Research(ISRR'99), snowbird, usa, oct. 11, 1999* (pp. 232–240).

Kuperstein, M. (1988). Neural model of adaptive hand-eye coordination for single postures. *Science*, *239*, 1308–1311.

Kupfermann, I., & Weiss, K. (2001). Motor program selection in simple model systems. *Current Opinion in Neurobiology*, *11*, 673–677.

Latash, M. L. (1993). What muscle parameters are controlled by the nervous system? In M. L. Latash (Ed.), *Control of human movement* (pp. 1–48). Champaign, Illinois: Human Kinetics.

Lazzaro, J., Wawrzynek, J., Mahowald, M., Sivilotti, M., & Gillespie, D. (1993). Silicon auditory processors as computer peripherals. *IEEE Journal of Neural Networks*, *4*, 523–528.

Levine, D. S. (2000). *Introduction to neural and cognitive modeling.* New Jersey: Lawrence Erlbaum Associates.

Levy, W., & Desmond, N. (1985). The rules of elemental synaptic plasticity. In W. Levy & J. Anderson (Eds.), *Synaptic modifications, neuron selectivity and nervous system organization.* Hillsdale, New Jersey: Lawrence Erlbaum.

Liaw, J., & Arbib, M. (1993). Neural mechanisms underlying direction-sensitive avoidance behavior. *Adaptive Behavior*, *1*, 227–261.

Lin, S., Si, J., & Schwartz, A. (1997). Self-organization of firing activities in monkey's motor cortex: trajectory computation from spike signals. *Neural Computation*, *9*, 607–621.

Livingstone, M. (1998). Mechanisms of direction selectivity in macaque V1. *Neuron*, *20*, 509–526.

Lukashin, A., Amirikian, B., & Georgopoulos, A. (1996). A simulated actuator driven by motor cortical signals. *Neuroreport*, *7*, 2597–2601.

Lukashin, A., & Georgopoulos, A. (1994). A neural network for coding of trajectories by time series of neuronal population vectors. *Neural Computation*, *6*, 19-28.

Maass, W. (1995). On the computational complexity of networks of spiking neurons. In D. S. T. G. Tesauro & T. K. Leen (Eds.), *Advances in neural information processing systems* (Vol. 7, pp. 183–190). MIT Press.

Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, *10*, 1659–1671.

Maass, W. (1999). Computing with spiking neurons. In W. Maass & C. Bishop (Eds.), *Pulsed neural networks.* MIT Press.

Maass, W., & Bishop, C. (Eds.). (1999). *Pulsed neural networks.* MIT Press.

Maass, W., & Schmitt, M. (1999). On the complexity of learning for spiking neurons with temporal coding. *Information and Computation*, *153*, 26–46.

MacKay, W. (1996). Cooperative structures for visually guided reach. *Canadian Journal of Physiology and Pharmacology*, *74*(4), 463–468.

Maex, R., & Orban, G. (1996). Model circuit of spiking neurons generating directional selectivity in simple cells. *Journal of Neurophysiology*, *75*, 1515–1545.

Mainen, Z., & Sejnowski, T. (1998). Modeling active dendritic processes in pyramidal neurons. In C. Koch & I. Segev (Eds.), *Methods in neuronal modeling* (Second ed., pp. 171–210). MIT Press.

Maistros, G., & Hayes, G. (2001). *An imitation mechanism inspired from neurophysiology* (Tech. Rep. No. EDI-INF-RR0031). Institute of Perception, Action and Behaviour, School of Informatics, University of Edinburgh.

Marconi, B., Genovesio, A., Battaglia-Mayer, A., Ferraina, S., Squatrito, S., Molinari, M., Lacquaniti, F., & Caminiti, R. (2001). Eye-hand coordination during reaching. I. anatomical relationships between parietal and frontal cortex. *Cerebral Cortex*, *11*, 513–527.

Marian, I., & Reilly, R. (2001). SpikeSNNS - a simulator for spike processing networks. In B. Katalinic (Ed.), *Proceedings of the 12th International DAAAM Symposium.* Vienna, Austria.

Markram, H., Lubke, J., Frotscher, M., & Sakmann, B. (1997). Regulation of synaptic efficacy by coindence of postsynaptic APs and EPSPs. *Science*, *275*, 213–215.

Martinetz, R., Ritter, H., & Schulten, K. (1988). Kohonen's self-organizing map for modeling the formation of the auditory cortex of a bat. In *SGAICO proceedings, connectionism in perspective* (pp. 403–412). Zürich.

Massone, L. (1995). Sensory-motor learning. in the handbook of brain theory and neural network. In M. A. Arbib (Ed.), *The handbook of brain theory and neural network.* MIT Press.

Mattia, M., & Del Giudice, P. (2000). Efficient event-driven simulation of large networks of spiking neurons and dynamical synapses. *Neural Computation*, *12*, 2305–2329.

McCarthy, J. (1963). A basis for a mathematical theory of computation. In P. Braffort & D. Hirshberg (Eds.), *Computer programming and formal systems.* North-Holland.

McClelland, J. (2000). Introduction. In R. C. O'Reilly & Y. Munakata (Eds.), *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain.* MIT Press.

McClelland, J., McNaughton, B., & O'Reilly, R. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, *102*, 419–457.

Mel, B. (1991). Vision-based robot motion planning. In I. W.T. Miller & R. Sutton (Eds.), *Neural networks for control* (pp. 229–254). MIT Press.

Mel, B. (1992). NMDA-based pattern discrimination in a modeled cortical neuron. *Neural Computation*, *4*, 502–516.

Mel, B. (1994). Information processing in dendritic trees. *Neural Computation*, *6*, 1031–1085.

Meltzoff, A., & Moore, K. (1977). Imitation of facial and manual gestures by human neonates. *Science*, *198*, 75–78.

Meltzoff, A. N., & Moore, M. (1999). Persons and representation: why infant imitation is important for theories of human development. In J. Nadel & G. Butterworth (Eds.), *Imitation in infancy.* Cambridge University Press.

Merzenich, M., Kaas, J., Wall, J., Nelson, R., Sur, M., & Felleman, D. (1983). Topographic reorganization of somatosensory cortical areas 3b and 1 in adult monkeys following restricted deafferentation. *Neuroscience*, *8*, 33–55.

Messier, J., & Kalaska, J. (2000). Covariation of primate dorsal premotor cell activity with direction and amplitude during a memorized-delay reaching task. *Journal of Neurophysiology*, *84*(1), 152–165.

Metta, G., & Fitzpatrick, P. (2002). Better vision through manipulation. In Y. M. H. K. . C. B. In C. G. Prince, Y. Demiris (Ed.), *Proceedings of the second international workshop on epigenetic robotics: Modeling cognitive development in robotic systems.* Lund, Sweden: Lund University Cognitive Studies Volume 94.

Metta, G., Sandini, G., & Konczak, J. (1999). A developmental approach to visually-guided reaching in artificial systems. *Neural Networks*, *12*, 1413–1427.

Miikkulainen, R., Bednar, J. A., Choe, Y., & Sirosh, J. (1998). A self-organizing neural network model of the primary visual cortex. In S. Usui & T. Omori (Eds.), *Proceedings of the Fifth International Conference on Neural Information Processing* (Vol. 2, pp. 815–818). Tokyo; Amsterdam: IOS Press.

Miikkulainen, R., & Sirosh, J. (1996). Introduction: The emerging understanding of lateral interactions in the cortex. In J. Sirosh, R. Miikkulainen, & Y. Choe (Eds.), *Lateral interactions in the cortex: Structure and function.* online. (Hypertext Book, http://www.cs.utexas.edu/users/nn/web-ubs/htmlbook96/)

Möhl, B. (1993). The role of proprioception for motor learning in locust flight. *Journal of Comparative Physiology*, *172*, 325–332.

Mohraz, K., Schott, U., & Pauly, M. (1997). Parallel simulation of pulse coded neural networks. In *Proceedings of the 15th IMACS World Congress* (Vol. 6, pp. 523–528).

Morasso, P., & Mussa-Ivaldi, F. (1982). Trajectory formation and handwriting: A computational model. *Biological Cybernetics*, *45*, 131–142.

Mushiake, H., Tanatsugu, Y., & Tanji, J. (1997). Neuronal activity in the ventral part of premotor cortex during target-reach movement is modulated by direction of gaze. *Journal of Neurophysiology*, *78*, 567–571.

Mussa-Ivaldi, F. (1999). Modular features of motor control and learning. *Current Opinions in Neurobiology*, *9*, 713–717.

M. Wiesendanger, B. H. A. P. V. M. U. W., M. Corboz, & Rouiller, E. (1992). Bimanual synergies in primates. In R. Caminiti, P. Johnson, & Y. Burnod (Eds.), *Control of arm movement in space* (pp. 45–64). Berlin: Springer Verlag.

Nadel, J., & Butterworth, G. (1999). Introduction: Immediate imitation rehabilitated at last. In J. Nadel & G. Butterworth (Eds.), *Imitation in infancy.* Cambridge: Cambridge University Press.

N. Franceschini, C. B., J. M. Pichon. (1992). From insect vision to robot vision. *Philisophical Transactions of the Royal Society, London (series B)*, *337*, 283–294.

Nicolelis, M., & Chapin, J. (2002). Controlling robots with the mind. *Scientific American*, *287*, 46–53.

Niebur, E., & Wörgötter, F. (1993). Orientation columns from first principles. In F. Eeckman (Ed.), *Computation and neural systems 92* (pp. 409–414). Kluver Academic.

Obermayer, K., Blasdel, G., & Schulten, K. (1992). A statistical mechanical analysis of self-organization and pattern formation during the development of visual maps. *Physical Review A*, *45*, 7568–7589.

Obermayer, K., Ritter, H., & Schulten, K. (1990). A neural network model for the formation of topographic maps in the cns: Development of receptive fields. In *Proceedings of IJCNN-90* (Vol. 2, pp. 423–429).

Oja, E. (1982). A simplified neural model as a principal component analyzer. *Journal of Mathematics and Biology*, *15*, 267–273.

O'Reilly, R. C. (2001). Generalization in interactive networks: The benefits of inhibitory competition and Hebbian learning. *Neural Computation*, *13*, 1199–1242.

O'Reilly, R. C., & Munakata, Y. (2000). *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain.* A Bradford Book, The MIT Press.

Panchev, C., & Wermter, S. (2001). Hebbian spike-timing dependent self-organization in pulse neural networks. In *Proceedings of World Congress on Neuroinformatics, vienna austria.*

Panzeri, S., & Schultz, S. R. (2001). A unified approach to the study of temporal, correlational and rate coding. *Neural Computation*, *13*, 1311–1349.

Pellionisz, A., & Llinas, R. (1980). Tensorial approach to the geometry of brain function. cerebellar coordination via a metric tensor. *Neuroscience*, *5*, 1761–1770.

Piaget, J. (1969). *The mechanisms of perception.* London: Rutledge & Kegan Pau.

Poirazi, P., Brannon, T., & Mel, B. (in press). Arithmetic of synaptic summation in a model ca1 pyramidal cell.

Poirazi, P., & Mel, B. (1999). Towards the memory capacity of neurons with active dendrites. *Neurocomputing*, *26–27*, 237–245.

Poirazi, P., & Mel, B. (2000a). Impact of active dendrites and structural plasticity on the memory capacity of neural tissue. *Neuron*, *29*, 779–796.

Poirazi, P., & Mel, B. (2000b). Sublinear vs. superlinear synaptic integration? tales of a dublicitous active current. In *Proceedings of the 7th Joint Symposium on Neural Computation* (Vol. 10, pp. 88–95).

Port, R. F., & van Gelder, T. (Eds.). (1995). *Mind as motion: Explorations in the dynamics of cognition.* MIT Press.

Rall, W. (1977). Core conductor theory and cable properties of neurons. In E. Kandel (Ed.), *The nervous system: Cell biology of neurons. section 1: Handbook of physiology* (Vol. 1, pp. 39–97). Bethesda: American Physiological Society.

Rall, W., & Agmon-Snir, H. (1998). Cable theory for dendritic neurons. In C. Koch & I. Segev (Eds.), *Methods in neuronal modeling: From ions to networks* (Second ed., pp. 63–96). MIT Press.

Read, H., Winer, J., & Schreiner, C. (2001). Modular organization of intrinsic connections associated with spectral tuning in cat auditory cortex. In *Proceedings of the national academy of sciences usa* (Vol. 98, pp. 8042–8047).

Recce, M. (1999). Encoding information in neuronal activity. In W. Maass & C. Bishop (Eds.), *Pulsed neural networks.* MIT Press.

Reilly, R. (1997). Cortical software re-use: A neural basis for creative cognition. In T. Veale (Ed.), *Computational models of creative computation* (pp. 36–42).

Reilly, R. (2000). Collaborative cell assemblies: Towards a cortical computational building block. In *Proceedings of EmerNet 2001.*

Reilly, R. (in press). On the relationship between object assembly and language production: A connectionist simulation of Greenfield's hypothesis. *Behavioral and Brain Sciences*.

Reilly, R., & Marian, I. (2002). Cortical Software Re-Use: A computational principle for cognitive development in robots. In *Proceedings of International Conference on Development and Learning (ICDL'02).*

Reina, G., Moran, D., & Schwartz, A. (2001). On the relationship between joint angular velocity and motor cortical discharge during reaching. *Journal of Neurophysiology*, *85*, 2576–2589.

Rieke, F., Warland, D., van Steveninck, R. R., & Bialek, W. (1997). *Spikes: exploring the neural code.* Cambridge, Ma, London: MIT Press.

Rioult-Pedotti, M., Friedman, D., & Donoghue, J. (1998). Motor skill learning strengthens horizontal connections in adult cortex. *Nature Neuroscience*, *1*(3), 220.

Ritter, H., Martinez, T., & Schulten, K. (1989). Topology-conserving maps for learning visuo-motor coordination. *Neural Networks*, *2*, 159–168.

Ritter, H., Martinez, T., & Schulten, K. (1992). *Neural computation and self-organizing maps.* Addison-Wesley.

Rizzolatti, G., Fadiga, L., Fogassi, L., & Gallese, V. (1999). Resonance behaviors and mirror neurons. *Archives Italiennes de Biologie*, *137*, 85–100.

Roberts, P. (1999). Computational consequences of temporally asymmetric learning rules: I.differential hebbian learning. *Journal of Computational Neuroscience*, *7*, 235–246.

Roberts, P. (2000). Modeling inhibitory plasticity in the electrosensory system of mormyrid electric fish. *Journal of Neurophsyiology*, *84*(4), 2035–2047.

Robinson, D. A. (1992). Implications of neural networks for how we think about brain function. *Behavioral and Brain Sciences*, *15*(4), 644–655.

Roby-Brami, A., & Burnod, Y. (1995). Learning a new visuomotor transformation: error correction and generalization. *Cognitive Brain Research*, *2*, 229–242.

Rolls, E. T., & Treves, A. (1998). *Neural networks and brain function.* Oxford, New York: Oxford University Press.

Rossetti, Y., Pisella, L., & Pélisson, D. (2000). Eye blindness and hand sight: temporal aspects of visuo-motor processing. *Visual Cognition*, *7*, 785–809.

Ruf, B., & Schmitt, M. (1998). Self-organization of spiking neurons using action potential timing. *IEEE Transactions on Neural Networks*, *9*, 575–578.

Rumelhart, D., McClelland, J., & the PDP Research Group. (1986a). *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 1). MIT Press.

Rumelhart, D., McClelland, J., & the PDP Research Group. (1986b). *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 2). MIT Press.

Salinas, E., & Abbott, L. (1995a). Transfer of coded information from sensory to motor networks. *Journal of Neuroscience*, *15*, 6461–6474.

Salinas, E., & Abbott, L. (1995b). Transfer of coded information from sensory to motor networks. In *Proceedings of the national academy of sciences* (Vol. 93, pp. 6461–6474).

Salinas, E., & Abbott, L. (1996). A model of multiplicative neural responses in parietal cortex. In *Proceedings of the national academy of sciences* (Vol. 93, pp. 11956–11961).

Sanes, J., & Donoghue, J. (1992). Organization and adaptability of muscle representations in primary motor cortex. In R. Caminiti, P. Johnson, & Y. Burnod (Eds.), *Control of arm movement in space* (pp. 103–127). Springer.

Scassellati, B. (1998). Building behaviors developmentally: A new formalism. In *Proceedings of the 1998 AAAI Spring Symposium on Integrating Robotics Research.*

Scassellati, B. (2000). How robotics and developmental psychology complement each other. In *NSF/DARPA Workshop on Development and Learning.* Lansing, Michigan.

Schaal, S. (1999). Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, *3*(6), 233–242.

Schmidt, R. (1988). *Motor control and learning: A behavioral emphasis* (Second ed.). Champaign, Illinois: Human Kinetics.

Schoenauer, T., Atasoy, S., Mehrtash, N., & Klar, H. (1998). MASPINN: Novel concepts for a neuro-accelerator for spiking neural networks. In *Proceedings of VIDYNN '98.* Stockholm.

Schwartz, A. (1994). Direct cortical representation of drawing. *Science*, *265*, 540–542.

Schwartz, A., Kettner, R., & Georgopoulos, A. (1988). Primate motor cortex and free arm movement to visual targets in three-dimensional space. I. relations between single cell discharge and direction of movement. *Journal of Neuroscience*, *8*, 2913–2927.

Scott, S. (1997). Comparison of onset time and magnitude of activity for proximal arm muscles and motor cortical cells prior to reaching movements. *Journal of Neurophysiology*, *77*.

Scott, S., & Kalaska, J. (1995). Changes in motor cortex activity during reaching movements with similar hand paths but different arm postures. *Journal of Neurophysiology*, *73*, 2563–2567.

Scott, S., & Kalaska, J. (1997). Reaching movements with similar hand paths but different arm orientations. I. activity of individual cells in motor cortex. *Journal of Neurophysiology*, *77*, 826–852.

Segev, I., & Burke, R. (1998). Compartmental models of complex neurons. In C. Koch & I. Segev (Eds.), *Methods in neuronal modeling: From ions to networks* (Second ed.). MIT Press.

Seung, H. S., & Sompolinsky, H. (1993). Simple models for reading neuronal population codes. In *Proceedings of the national academy of sciences usa* (Vol. 90, pp. 10749–10753).

Shadmehr, R., & Moussavi, Z. (2000). Spatial generalization from local learning of reaching movements in force fields. *Journal of Neuroscience*, *20*, 7807–7815.

Shadmehr, R., & Mussa-Ivaldi, F. (1994). Adaptive representation of dynamics during learning of a motor task. *Journal of Neuroscience*, *14*, 3208–3224.

Shen, L., & Alexander, G. (1997). Neural correlates of a spatial sensory-to-motor transformation in primary motor cortex. *Journal of Neurophysiology*, *77*(3), 1171–1194.

Shepherd, G., & Koch, C. (1998). Introduction to synaptic circuits. In G. Shepherd (Ed.), *The synaptic organization of the brain* (Fourth ed., pp. 1–36). Oxford University Press.

Singer, W. (1994). Putative functions of temporal correlations in neocortical processing. In C. Koch & J. Davis (Eds.), *Large-scale neuronal theories of the brain*. Cambridge, Massachusetts: MIT Press.

Sirosh, J. (1995). *A self-organizing neural network model of the primary visual cortex*. Unpublished doctoral dissertation, Department of Computer Sciences, University of Texas at Austin. (Technical Report AI95-237)

Sirosh, J., & Miikkulainen, R. (1996). Self-organization and functional role of lateral connections and multisize receptive fields in the primary visual cortex. *Neural Processing Letters*, *3*, 39–48.

Sirosh, J., Miikkulainen, R., & Bednar, J. (1996). Self-organization of orientation maps, lateral connections, and dynamic receptive fields in the primary visual cortex. In J. Sirosh, R. Miikkulainen, & Y. Choe (Eds.), *Lateral interactions in the cortex: Structure and function*. Electronic book, ISBN 0-9647060-0-8. (http://www.cs.utexas.edu/users/nn/web-pubs/htmlbook96)

Sirosh, J., Miikkulainen, R., & Choe, Y. (Eds.). (1996). *Lateral interactions in the cortex: Structure and function*. Electronic book, ISBN 0-9647060-0-8. (http://www.cs.utexas.edu/users/nn/web-pubs/htmlbook96)

Smith, L., & Thelen, E. E. (1993). *A dynamic systems approach to development: Applications*. Cambridge, Massachusetts: MIT Press.

Snyder, S. (2000). Coordinate transformations for eye and arm movements in the brain. *Current Opinion in Neurobiology*, *10*, 747–754.

Soechting, J., Engel, K., & Flanders, M. (2001). The Duncker illusion and eye-hand coordination. *Journal of Neurophysiology*, *85*, 843–854.

Soechting, J., & Flanders, M. (1995). Psychophysical approaches to motor control. *Current Opinion in Neurobiology*, *5*, 742–748.

Softky, W., & Koch, C. (1993). The highly irregular firing of cortical cells in inconsistent with temporal integration of random ESPS. *Journal of Neuroscience*, *13*, 334–350.

Somers, D. C., Toth, L. J., Todorov, E., Rao, S. C., Kim, D., Nelson, S. B., Siapas, A. G., & Sur, M. (1996). Variable gain control in local cortical circuitry supports context-dependent modulation by long-range connections. In J. Sirosh, R. Miikkulainen, & Y. Choe (Eds.), *Lateral interactions in the cortex: structure and function.* Electronic book, ISBN 0-9647060-0-8. (http://www.cs.utexas.edu/users/nn/web-pubs/htmlbook96)

Song, S., Miller, K., & Abbott, L. (2000). Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, *3*(9), 919–926.

Soto-Trevino, C., Thoroughman, K., Marder, E., & Abbott, L. (2001). Activity-dependent modification of inhibitory synapses in models of rhythmic neural networks. *Nature Neuroscience*, *4*, 297–303.

Sougné, J. (1999). *INFERNET: A neurocomputational model of binding and inference.* Unpublished doctoral dissertation, Université de Liege.

Squire, L., & Kandel, E. (1999). *Memory: From mind to molecules.* New York: Scientific American Library.

Steuber, V., & De Schutter, E. (in press). Long-term depression and recognition of parallel fibre patterns in a multi-compartmental model of a cerebellar purkinje cell. *Neurocomputing*.

Stevens, C. F., & Zador, A. (1995). Neural coding: The enigma of the brain. *Current Biology*, *5*, 1370–1371.

Stevens, C. F., & Zador, A. M. (1998). Novel integrate-and-fire-like model of repetitive firing in cortical neurons. In *Proceedings of the 5th Joint Symposium on Neural Computation.* La Jolla, California.

Tanne, J., Boussaoud, D., Boyer-Zeller, N., & Rouiller, E. (1995). Direct visual pathways for reaching movements in the macaque monkey. *Neuroreport*, *7*, 267–272.

Taylor, J. G., & Alavi, F. N. (1996). A basis for long-range inhibition across cortex. In J. Sirosh, R. Miikkulainen, & Y. Choe (Eds.), *Lateral interactions in the cortex: structure and function.* Electronic book, ISBN 0-9647060-0-8. (http://www.cs.utexas.edu/users/nn/web-pubs/htmlbook96)

Theeuwen, M., Gielen, C., & van Bolhuis, B. (1996). Estimating the contribution of muscles to joint torque based on motor-unit activity. *Journal of Biomechanics*, *29*, 881–889.

Thorpe, S., Delorme, A., & VanRullen, R. (in press). Spike-based strategies for rapid processing. *Neural Networks*.

Thorpe, S., Fize, D., & Marlot, C. (1996). Speed of processing in the human visual system. *Nature*, *381*, 520–522.

Thorpe, S., & Gautrais, J. (1997). Rapid visual processing using spike asynchrony. In M. C. Mozer, M. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems* (Vol. 9, pp. 901–907). MIT Press.

Thorpe, S., & Gautrais, J. (1998). Rank order coding. In *Computational neuroscience: Trends in research 1998* (pp. 113–118). New York: Plenum Press.

Toga, A., & Mazziotta, J. (1996). *Brain mapping: The methods.* San Diego, California: Academic Press.

Touretzky, D. S. (2001). *Computational models of neural systems.* http://www.cs.cmu.edu/afs/cs/academic.

Tovee, M., & Rolls, E. (1995). Information encoding in short firing rate epochs by single neurons in the primate temporal visual cortex. *Visual Cognition*, *2*(1), 35–58.

Tovee, M., Rolls, E., Treves, A., & Belles, R. (1993). Information encoding and the responses of single neurons in the primate visual cortex. *Journal of Neurophysiology*, *70*, 640–654.

Trevarthen, C., Kokkinaki, T., & Fiamenghi Jr., G. (1999). What infant's imitations communicate: with mothers, with fathers and with peers. In J. Nadel & G. Butterworth (Eds.), *Imitation in infancy*. Cambridge: Cambridge University Press.

Tuckwell, H. (1988). *Introduction to theoretical neurobiology*. Cambridge University Press.

Usher, M., Stemmler, M., & Niebur, E. (1996). The role of lateral connections in visual cortex: Dynamics and information processing. In J. Sirosh, R. Miikkulainen, & Y. Choe (Eds.), *Lateral interactions in the cortex: structure and function*. Electronic book, ISBN 0-9647060-0-8. (http://www.cs.utexas.edu/users/nn/web-pubs/htmlbook96)

Van Gelder, T. (1998). The dynamical hypothesis in cognitive science. *Behavioral and Brain Sciences*, *21*, 1–14.

Van Gelder, T., & Port, R. (1995). It's about time: An overview of the dynamical approach to cognition. In R. F. Port & T. van Gelder (Eds.), *Mind as motion: Explorations in the dynamics of cognition*. MIT Press.

Van Rullen, R., , Gautrais, J., Delorme, A., & Thorpe, S. (1998). Face processing using one spike per neurone. *Biosystems*, *48*(229–239).

Van Rullen, R., & Thorpe, S. (2001). Rate coding versus temporal order coding: what the retinal ganglion cells tell the visual cortex. *Neural Computation*, *13*, 1255–1283.

Vetter, P., Roth, A., & Häusser, M. (2001). Propagation of action potentials in dendrites depends on dendritic morphology. *Journal of Neurophysiology*, *85*, 926–937.

Wang, D. (1996). *Object selection based on oscillatory correlation* (Tech. Rep. No. 67). Ohio State University.

Watts, L. (1994). Event-driven simulation of networks of spiking neurons. In J. D. Cowan, G. Tesauro, & J. Alspector (Eds.), *Advances in neural information processing systems* (pp. 927–934). Morgan Kaufmann Publishers.

Weitzenfeld, A., Arbib, M., & Alexander, A. (1999). *The neural simulation language*. MIT Press.

Weliky, M., & Katz, L. (1994). Functional mapping of horizontal connections in developing ferret visual cortex: experiments and modeling. *Journal of Neuroscience*, *14*, 7291–7305.

Weng, J., McClelland, J., Pentland, A., Sporns, O., Stockman, I., Sur, M., & Thelen, E. (2001). Autonomous mental development by robots and animals. *Science*, *291*, 599–600.

Werner, W., Bauswein, E., & Fromm, C. (1991). Static firing rates of premotor and primary motor cortical neurons associated with torque and joint position. *Experimental Brain Research*, *86*, 293–302.

Wise, S., Boussaoud, D., Johnson, P., & Caminiti, R. (1997). Premotor and parietal cortex: corticocortical connectivity and combinatorial computations. *Annual Review of Neuroscience*, *20*, 25–42.

Xing, J., & Andersen, R. (2000). The memory activity of lip neurons in sequential eye movements simulated with neural networks. *Journal of Neurophysiology*, *84*, 651–665.

Zeller, M., Wallace, K. R., & Schulten, K. (1995). Biological visuo-motor control of a pneumatic robot arm. In A. Dagli, F. Chen, & J. Gosh (Eds.), *Intelligent engineering systems through artificial neural networks* (Vol. 5, pp. 645–650). New York: ASME Press.

Zhang, J., Riehle, A., Requin, J., & Kornblum, S. (1997). Dynamics of single neuron activity in monkey primary motor cortex related to sensorimotor transformation. *Journal of Neuroscience*, *17*, 2227–2246.

Zhang, K., & Sejnowski, T. J. (1999). Neuronal tuning: To sharpen or broaden? *Neural Computation*, *11*, 75–84.

Zhang, L. I., Tao, H. W., Holt, C. E., Harris, W. A., & Poo, M. M. (1998). A critical window for cooperation and competition among developing retinotectal synapses. *Nature*, *395*, 37–44.

Zipser, D., & Andersen, R. (1988). A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature*, *331*, 679–684.

Zlatev, J., & Balkenius, C. (2001). Introduction: Why ”epigenetic robotics”? In C. Balkenius, J. Zlatev, H. Kozima, K. Dautenhahn, & C. Breazeal (Eds.), *Proceedings of the first international workshop on epigenetic robotics.*