Simulation of biological neuronal structures. Design and functional study of the cerebellum

Thesis · November 2011

Jesús Garrido
University of Granada

Simulation of biological neuronal structures. Design and functional study of the cerebellum

Jesús Alberto Garrido Alcázar

Department of Computer Architecture and Technology

University of Granada

A thesis submitted for the degree of

PhilosophiæDoctor (PhD) in Computer Science

October 2011

Declaración

Dr. Eduardo Ros Vidal, Catedrático de Universidad del Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada, y Dr. Richard Rafael Carrillo Sánchez, investigador contratado bajo el Programa Juan de la Cierva del Departamento de Arquitectura de Computadores y Electrónica de la Universidad de Almería,

CERTIFICAN:

Que la memoria tiulada "Simulation of biological neuronal structures. Design and functional study of the cerebellum", ha sido realizada por D. Jesús Alberto Garrido Alcázar bajo nuestra dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor por la Universidad de Granada.

Granada, 23 de Septiembre de 2011

Fdo. Eduardo Ros Vidal

Fdo. Richard Rafael Carrillo Sánchez

Abstract

In this work, an extensive simulation study of the cerebellum is presented. Our study required the further development of the EDLUT spiking neural network simulator. Thus we have addressed the development of a detailed cerebellar model from different levels of abstraction. Firstly, in a detailed model, the granular-layer network generated rebounds and oscillations in the β/γ-frequency band and filtered unsynchronized trains with millisecond precision. We found that weights at multiple synapses could play a crucial role to enhance coincidence detection (which allows the reduction of non-synchronous signals) and sensitivity rebound (which determines specific time windows for signal transmission). These results predict that the granular layer operates as a complex adaptable filter which can be controlled by weight changes at multiple synaptic sites. In a higher level of abstraction, a model of the whole cerebellum which can infer corrective models in the framework of a control task is presented. This work studies how a basic temporal-correlation kernel, including long-term depression (LTD) and long-term potentiation (LTP) at parallel fibers-Purkinje cell synapses, can effectively infer corrective models. Finally, we study how cerebellar input representations (context labels and sensorimotor signals) can efficiently support model abstraction towards delivering accurate corrective torque values for increasing precision during different-object manipulation.

This work has been partially supported by the EU projects SENSOPAC (FP6-IST-028056) and REALNET (FP7-IST-270434), and the Spanish National projects ARC-VISION (TEC2010-15396) and ITREBA (TIC-5060). Some of these projects covered the early stages of the PhD and the national FPU grant program covered the last stage.

Resumen

En este trabajo se presenta un extenso estudio de simulación del cerebelo. Dicho estudio requiere del desarrollo del simulador de redes neuronales de impulsos EDLUT, herramienta para la que se implementó un modelo realista del cerebelo desde distintos niveles de abstracción. Usando un modelo detallado, la capa granular fue capaz de generar rebotes de sensibilidad y oscilaciones en las bandas de frecuencia β/γ, así como de filtrar trenes de actividad desincronizados con precisión de milisegundos. Según este modelo, los pesos en múltiples sinapsis podrían desempeñar un papel crucial para mejorar la detección de coincidencias (que permite la reducción de señales no sincronizadas) y el rebote de sensibilidad (que determina las ventanas de tiempo preferidas para la transmisión de señales). Estos resultados predicen que la capa granular actúa como un filtro complejo adaptable, que puede ser controlado por los cambios de pesos sinápticos. Desde un nivel más alto de abstracción, se presenta un modelo completo de cerebelo capaz de inferir modelos de corrección en el marco de una tarea de control. Usando dicho modelo se muestra cómo un núcleo operacional básico de correlación temporal (incluyendo depresión —LTD— y potenciación a largo plazo —LTP— en la sinapsis fibras paralelas-células de Purkinje) puede inferir eficientemente modelos correctivos. Finalmente, se estudia cómo las representaciones de las entradas cerebelares (etiquetas de contexto y señales sensoriomotoras) pueden apoyar la abstracción de modelos con el fin de producir valores correctores de fuerza fiables para incrementar la precisión durante la manipulación de objetos.

Este trabajo ha sido financiado por los proyectos de la UE SENSOPAC (FP6-IST-028056) y REALNET (FP7-IST-270434), y los proyectos nacionales ARC-VISION (TEC2010-15396) e ITREBA (TIC-5060). El programa nacional FPU permitió la realización de las últimas etapas de esta tesis.

To my family

Acknowledgements

To begin with, I would like to thank my thesis directors, Eduardo, and Richard. They have introduced me to the fascinating world of research, providing both the knowledge and tools indispensable to become a researcher. From their respective viewpoints -i.e. either the proactive, and the critical ones- they have help me to achieve the right middle point to face any project and its inherent issues with chances of success. I would also like to express my sincere gratitude to some other colleagues around the world, such as those of the research group led by Egidio D'Angelo at the University of Pavia (Italy), and the group directed by Angelo Arleo at the Pierre & Marie Curie University in Paris (France), for their appraisable insight and fruitful discussions about neurophysiologic experimental evidences of different hypotheses and models presented in this work. They provided me with unthinkable approaches which might have been out of reach for a mere computer engineer who had never seen a biological system before.

I can not obviate my grateful acknowledgment to the person who has suffered firsthand the inconvenience and unpleasantness of having to deal with spikes day after day, Niceto. Without his hard work, I have no doubt that this thesis would not have been successfully accomplished. Our endless and helpful conversations about life, the upcoming future, and the development of a career as a researcher have been as profitable as those covering learning plots. At this point, I also want to remember all the comrade-in-arms —both at the Faculty of Sciences, at the CIE, and finally at the CITIC of the University of Granada— whom I have shared much more than a workplace with. Many interesting people showing a wide range of different views yielding infinite opportunities to learn by mingling and chatting with them.

In addition to my colleagues' support, I really appreciate the role played by those few close individuals who have helped me carry on problems that belong to my own career with relieving sympathy and trying to lessen their transcendence anytime they came up. Namely, my parents Matilde, and José Agustín, my brother José Agustín, and, last but not least, my girlfriend Laura. Since I have always turned to them in my bad moments, while feeling upset, frustrated or deceptive, they very much deserve to be recalled in my good moments as well. Thank you, my family, for being so supportive.

Moreover, I would like to thank all those friends of mine whom I have spent with my leisure time on tapas, barbecues, and enjoyable parties within the almost five years since I finished my degree. Even though I have never talked to them a single word about cerebellum, spikes, or robotic arms, they have somehow taken part of this work. Furthermore, all the time I have spent playing volleyball and beach-volley ought to be remembered. Memories necessarily attached to so many people that I have played with: from the former Computer Science College's team, to the current one called "C.V. Horizonte Absoluto", including beach-volley teammates, thank you all.

Finally, I would want to thank those entities and programs which have financially supported this work: the EU funded project SENSOPAC (IST-028056) for three years which allowed me to get off to my research career, and the FPU program -standing for Formación de Profesorado Universitario- for the last two and a half years of this work sponsored by the Spanish Ministry of Education.

Contents

List of Figures **xiii**

List of Tables **xvii**

List of Acronyms **xix**

1 Introduction **1**

1.1 Computational neuroscience 1

1.2 The importance of studying the cerebellum 5

1.3 General work motivation 7

1.4 Our Contribution 8

1.5 Objectives 9

1.6 Project Framework 10

1.7 Chapter Organization 12

2 Simulating the cerebellum: the EDLUT simulation environment **13**

2.1 Abstract 13

2.2 Introduction 14

2.2.1 Simulation strategies: event-driven vs time-driven 15

2.2.1.1 Time-driven simulation models 16

2.2.1.2 Event-driven simulation models 17

2.2.2 The state of the art: Actual simulation tools 19

2.2.2.1 NEURON 20

2.2.2.2 GENESIS 23

2.2.2.3 NEST 24

2.2.2.4 Other simulation tools 25

2.2.2.5 Discussion 27

2.3 The EDLUT simulation environment 30

2.3.1 EDLUT graphical user interface 32

2.3.2 Integrating EDLUT with the environment 38

2.3.2.1 Using plain-text files 39

2.3.2.2 Using TCP-IP sockets 39

2.3.2.3 Using the C/C++ API 40

2.3.2.4 Using the Simulink S-function module 44

2.4 Building more neuronal models 46

2.4.1 Time-driven models 47

2.4.1.1 Leaky Integrate and Fire time-driven model 49

2.4.1.2 Spike Response time-driven model 51

2.4.2 Stochastic SRM table-based models 53

2.5 Results 56

2.5.1 Hybrid simulation of time-driven and event-driven models 56

2.5.2 EDLUT released as open-source 58

2.6 Discussion 60

3 The impact of synaptic weights on spike timing in the cerebellar input layer 63

3.1 Abstract 63

3.2 Introduction 64

3.2.1 The state of the art: temporal processing in the cerebellum 65

3.2.1.1 The input stage: the granular layer 66

3.2.1.2 The Purkinje cells and the molecular layer 69

3.2.1.3 The cerebellar output stage 70

3.2.1.4 The importance of the oscillations in the granular layer 73

3.3 Materials and methods 74

3.3.1 The EDLUT Simulator 75

3.3.2 Neural Models 76

3.3.3 Network topology 79

3.3.4 Coincidence Detection Methods 83

3.3.5 Time-Slicing Methods 85

3.4 Results 88

3.4.1 Coincidence Detection 88

3.4.2 Time Slicing through the Permissive Window and Post-Transmission Sensitivity Rebound 90

3.4.3 Coincidence Detection and Time-Slicing Co-Optimization 96

3.4.4 Oscillations during Protracted Stimulation 98

3.5 Discussion 102

4 Using the cerebellum for correcting in manipulation tasks 107

4.1 Abstract 107

4.2 Introduction: The problem of manipulation tasks 108

4.2.1 The state of the art: functional models of the cerebellum 113

4.2.1.1 The cerebellum as an adaptive filter 116

4.2.1.2 The cerebellum as a synchronous system 119

4.2.1.3 The cerebellum embedding an internal clock 122

4.2.1.4 Discussion 125

4.3 Materials and methods 126

4.3.1 Training trajectories 127

4.3.2 Control loop. Interfacing the cerebellar model and the robot simulator 129

4.3.3 Spike-based cerebellar model 131

4.3.3.1 First Approach: Studying adaptation at parallel fibers . 132

4.3.3.2 Second Approach: Studying information convergence at the granular layer 136

4.3.3.3 Spike-Time dependant plasticity 141

4.3.3.4 The Teaching Signal of the Inferior Olive 142

4.3.4 Quantitative performance evaluation 145

4.4 Results 147

4.4.1 Influence of the learning mechanisms at the parallel fibbers . . . 147

4.4.1.1 LTD vs LTP trade-off 147

4.4.1.2 Corrective learning in presence of signal delays 152

4.4.1.3 Adaptation to several dynamical models 153

4.4.2 Influence of the information convergence at the granular layer . . 156

4.5 Discussion 162

5 Discussion 165

5.1 Future works 165

5.2 Publication of results 166

5.2.1 International Peer-review Journals 166

5.2.2 International Conferences 167

5.2.3 Other dissemination sources 168

5.3 General Scientific Framework 170

5.4 Main contributions 171

6 Introducción en castellano 173

6.1 La neurociencia computacional 173

6.2 La importancia de estudiar el cerebelo 177

6.3 Motivación general del trabajo 179

6.4 Nuestra contribución 180

6.5 Objetivos 181

6.6 Marco del proyecto 182

6.7 Organización de capítulos 184

7 Discusión en castellano 187

7.1 Trabajo futuro 187

7.2 Publicación de resultados 188

7.2.1 Revistas internacionales 189

7.2.2 Conferencias Internacionales 190

7.2.3 Otros medios de difusión 190

7.3 Marco científico general 192

7.4 Principales contribuciones 193

References 197

List of Figures

1.1 The structure of the cerebellum 6

1.2 Module structure of the SENSOPAC European Project 11

2.1 Time-Driven Simulation Scheme 17

2.2 Event-Driven Simulation Scheme 20

2.3 EDLUT simulation kernel architecture 31

2.4 EDLUT Interface main view 33

2.5 EDLUT Interface raster plot 35

2.6 EDLUT interfacing with a robotic arm 40

2.7 EDLUT interfacing with a robotic arm simulator using Simulink 46

2.8 Hybrid time-driven and event-driven simulation scheme 48

2.9 Simulated architecture of the granular layer 50

2.10 Several alternatives for the SRM potential function 54

2.11 Event-driven table-based simulation of a set of stochastic cells 55

2.12 Obtained error (MAE) after simulating 0.5 s of real activity 57

2.13 Simulation speed measured as the rate of processed spikes per second . . 59

3.1 The organization of a cerebellar module 66

3.2 The cellular basis for low-frequency oscillations and resonance in the cerebellar granular layer 67

3.3 Multiple loops involved in controlling cerebellar low-frequency activity . 71

3.4 Event-driven simulation of the included cell types 79

3.5 Structural organization of the simulated granular layer network 80

3.6 Network activity in response to an input train 84

3.7 Network activity in response to a repeated input train 86

3.8 Coincidence-detection experiments 89
3.9 Influential connections for coincidence detection 91
3.10 Sensitivity rebound curve obtained with different weights 93
3.11 Quantitative analysis of sensitivity rebound 95
3.12 Sensitivity rebound curve obtained with different electrical coupling coefficients 97
3.13 Network connection diagram and main weights involved in coincidence detection and sensitivity rebound 99
3.14 Synchronous activity oscillation during continuous stimulation 100
3.15 Frequency analysis of activity during continuous stimulation 101

4.1 Scheme of the cerebellum organization 110
4.2 The most relevant connections within a cerebellar module and in relation with an adaptive filter model 117
4.3 Three joint periodic trajectories describing 8-shape movements in joint coordinates 128
4.4 System control loop 130
4.5 First-approach cerebellum model diagram 134
4.6 RBF bank encoding cerebellar input signals 135
4.7 Second-approach cerebellum model diagram 138
4.8 Similarity indices for a spatiotemporal activity between two activity patterns using EC, IC, and IC&EC configurations 140
4.9 LTD integral kernel 143
4.10 Cerebellum state after 300 trajectory-learning iterations 148
4.11 Learning estimates 150
4.12 Learning characterization 151
4.13 Temporal-correlation kernel for different sensorimotor delays (delays from 25ms to 150ms have been tested) 153
4.14 Temporal-correlation kernel behavior with different deviations between sensorimotor delays and the kernel peak (deviation from-50ms to 70ms have been tested) 154
4.15 Learning the corrective models for the 8-like target trajectory when manipulating objects with different masses (2kg, 1.5kg, 1kg, and 0.5kg) . . 155

4.16 Learning Performance when manipulating different objects (0.5kg, 1kg, 1.5kg, and 2kg) during 400-trial learning processes 156
4.17 Multi-context simulation with changes in dynamics and kinematics using EC cerebellar input 157
4.18 Single-context simulation using EC and IC cerebellar input 158
4.19 Multi-context simulation with changes in kinematics and dynamics using IC and EC cerebellar input configurations 159
4.20 Multi-context simulation with changes in kinematics and dynamics using IC cerebellar input. Interpolation capability 160
4.21 Multi-context simulation with changes in kinematics and dynamics using EC, IC, and IC&EC cerebellar input 160
4.22 Multi-context simulation by interpolating changes in kinematics and dynamics using EC, IC, and IC&EC cerebellar input 161

List of Tables

3.1 Parameters of the different used cell types 77
3.2 Default configuration of synaptic weights 81

4.1 Neuron model parameters for the simulations 136
4.2 Performance improvement using several dynamical models 153

LIST OF ACRONYMS

.

List of Acronyms

AMPA 2-Amino-3-(5-Methyl-3-oxo-1,2-oxazol-4-yl) Propanoic Acid: compound that is a specific agonist for the AMPA receptor, where it mimics the effects of the neurotransmitter glutamate.

API Application Programming Interface: particular set of rules and specifications that software programs can follow to communicate with each other. It serves as an interface between different software programs and facilitates their interaction.

CFs Climbing Fibers: neuronal projections from the inferior olivary nucleus located in the medulla oblongata. These axons pass through the pons and enter the cerebellum via the inferior cerebellar peduncle where they form synapses with the deep cerebellar nuclei and Purkinje cells.

CNS Central Nervous System: the part of the nervous system that integrates the information that it receives from, and coordinates the activity of, all parts of the bodies of bilaterian animals. It consists of the brain and the spinal cord. Some classifications also include the retina and the cranial nerves in the CNS.

CPU Central Processing Unit: portion of a computer system that carries out the instructions of a computer program, to perform the basic arithmetical, logical, and input/output operations of the system.

CR Conditioned Response: The learned response to the previously neutral stimulus.

LIST OF ACRONYMS

CS Conditioned Stimulus: a previously neutral stimulus that, after becoming associated with the unconditioned stimulus, eventually comes to trigger a conditioned response.

CSIM Circuit SIMulator: re-usable general purpose discrete-event simulation environment for modeling complex systems of interacting elements. It contains hierarchical block diagram tools and extensive model libraries covering several domains.

DCN Deep Cerebellar Nuclei: clusters of gray matter lying within the white matter at the core of the cerebellum. They are, with the minor exception of the nearby vestibular nuclei, the sole sources of output from the cerebellum. These nuclei receive collateral projections from mossy fibers and climbing fibers, as well as inhibitory input from the Purkinje cells of the cerebellar cortex.

EC Explicit Context encoding approach: In this encoding approach, an external signal (related to the label or any captured property of the object, for instance assuming information captured through visual sensory system) feeds these dedicated fibers.

EDLUT Event-Driven Look-Up Table: computer simulation environment for simulating networks of detailed spiking neurons.

EPSP Excitatory Post-Synaptic Potential: temporary depolarization of postsynaptic membrane potential caused by the flow of positively charged ions into the postsynaptic cell as a result of opening of ligand-sensitive channels.

FMRI Functional Magnetic Resonance Imaging: type of specialized MRI scan used to measure the hemodynamic response (change in blood flow) related to neural activity in the brain or spinal cord of humans or other animals.

GABA γ-Aminobutyric Acid: main inhibitory neurotransmitter in the mammalian central nervous system. It plays a role in regulating neuronal excitability throughout the nervous system.

GENESIS GEneral NEural SImulation System: general purpose simulation platform that was developed to support the simulation of neural systems ranging from subcellular components and biochemical reactions to complex models of single neurons, simulations of large networks, and systems-level models.

GoCs Golgi Cells: inhibitory interneurons found within the granular layer of the cerebellum. These cells synapse onto the dendrite of granule cells and unipolar brush cells. They receive excitatory input from mossy fibers and parallel fibers, which are long granule cell axons, and inhibitory input from stellate cells.

GPLv3 General Public License v3: the most widely used free software license, originally written by Richard Stallman for the GNU Project. More information about the terms of this license can be found on the free software fundation site `http://gplv3.fsf.org/`.

GrCs Granule Cells: tiny neurons (around 10 micrometres in diameter) which are found within the granular layer of the cerebellum, the dentate gyrus of the hippocampus, the superficial layer of the dorsal cochlear nucleus, and in the olfactory bulb. In this work, we only refers to cerebellar granule cells.

GUI Graphical User Interface: type of user interface which represents the information and actions available to a user through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces.

HH Hodgkin-Huxley: mathematical model that describes how action potentials in neurons are initiated and propagated. It is a set of nonlinear ordinary differential equations that approximates the electrical characteristics of excitable cells such as neurons.

IC Implicit Context encoding approach: The implicit contextual information is conveyed using groups of fibers which encode the actual position and velocity. These deviations implicitly encode a context-like representation based on sensorimotor complexes.

IO Inferior Olive: the largest nucleus situated in the olivary body, part of the medulla oblongata. It is closely associated with the cerebellum, meaning that it is involved in control and coordination of movements, sensory processing and cognitive tasks likely by encoding the timing of sensory input independently of attention or awareness.

ISH In Situ Hybridization: a type of hybridization that uses a labeled complementary DNA or RNA strand to localize a specific DNA or RNA sequence in a portion or section of tissue (in situ).

ISI Inter-Stimulus Interval: temporal interval between the offset of one stimulus to the onset of another.

JCR Journal Citation Report: annual publication by the Healthcare & Science division of Thomson Reuters which provides information about academic journals in the sciences and social sciences, including impact factors.

LAN Local Area Network: computer network that interconnects computers in a limited area. The defining characteristics of LANs, in contrast to wide area networks, include their usually higher data-transfer rates, smaller geographic area, and lack of a need for leased telecommunication lines.

LCs Lugaro Cells: one class of interneurons in the cerebellum, which were first described by Lugaro in the early nineteenth century. These are found just inferior to the Purkinje cell bodies, between the molecular and granular layers of the cerebellum.

LIF Leaky Integrate-and Fire: neuronal model which is characterized by adding a *leak* term to the membrane potential, reflecting the diffusion of ions that occurs through the membrane when some equilibrium is not reached in the cell.

LSAM Large-scale analog model: bottom-up executable mathematical model of the cerebellum, developed at the Swedish Institute of Computer Science in Stockholm.

LTD Long-Term Depression: activity-dependent reduction in the efficacy of neuronal synapses lasting hours or longer.

LTP Long-Term Potentiation: long-lasting enhancement in signal transmission between two neurons that results from stimulating them synchronously.

MAE Mean Absolute Error: quantity used to measure how close forecasts or predictions are to the eventual outcomes.

MEG MagnetoEncephaloGraphy: technique for mapping brain activity by recording magnetic fields produced by electrical currents occurring naturally in the brain.

MFs Mossy Fibers: one of the major inputs to cerebellum. There are many sources of this pathway, the largest of which is the cerebral cortex, which sends input to the cerebellum via the pontocerebellar pathway. Other contributors include the vestibular nerve and nuclei, the spinal cord, the reticular formation, and feedback from deep cerebellar nuclei.

MPI Message-Passing Interface: standardized and portable message-passing system designed to function on a wide variety of parallel computers.

NEST Neural Simulation Technology: initiative (in the sense of project) founded in 2001 with the goal to collaborate in the development of simulation methods for biologically realistic neuronal networks. Concretely, the NEST simulator is developed within this initiative.

NMDA N-Methyl D-Aspartate: name of a selective agonist that binds to NMDA receptors but not to other glutamate receptors. Activation of NMDA receptors results in the opening of an ion channel that is nonselective to cations with an equilibrium potential near 0 mV.

NMODL Neuron MODeling Language: language used in Neuron simulator in order to specify the properties and mechanisms of the membrane channels of the neuron.

NR2B One particular NMDA subunit which is mainly present in immature neurons and in extrasynaptic locations, and contains the binding-site for the selective inhibitor ifenprodil.

ODE Ordinary Differential Equation: relation that contains functions of only one independent variable, and one or more of their derivatives with respect to that variable.

PCs Purkinje Cells: cells are found within the Purkinje layer in the cerebellum. Purkinje cells are aligned like dominos stacked one in front of the other. Parallel fibers make relatively weaker excitatory (glutamatergic) synapses to spines in the Purkinje cell dendrite, whereas climbing fibers originating from the inferior olivary nucleus in the medulla provide very powerful excitatory input to the proximal dendrites and cell soma.

PFs Parallel Fibers: synaptic connections that form excitatory synapses onto the dendrites of Purkinje cells (from the Granulle cells).

PVL Parallel Virtual Machine: software tool for parallel networking of computers which was designed to allow a network of heterogeneous Unix and/or Windows machines to be used as a single distributed parallel processor.

PyNN Python Neural Network: simulator-independent language for building neuronal network models.

RBF Radial Basis Function: real-valued function whose value depends only on the distance from the origin, so that sums of radial basis functions are typically used to approximate given functions (see Figure 4.6). This approximation process can also be interpreted as a simple kind of neural network.

SCs Stellate Cells: neurons with several dendrites radiating from the cell body giving them a star shaped appearance. The three most common stellate cells are the inhibitory interneurons found within the molecular layer of the cerebellum, excitatory spiny stellate interneurons and inhibitory aspiny stellate interneurons. Cerebellar stellate cells synapse onto the dendritic arbors of Purkinje cells.

SRM Spike Response Model: generalization of the leaky integrate-and-fire model in which action potentials are generated when the voltage passes a threshold from below, but adding a term of refractoriness.

STDP Spike-timing-dependent plasticity: biological process that adjusts the strength of connections between neurons in the brain based on the relative timing of a particular neuron's output and input action potentials.

TCP/IP Transmission Control Protocol/Internet Protocol: set of communications protocols used for the Internet and other similar networks. The two of the most important protocols in it are the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which were the first two networking protocols defined in this standard

UBCs Unipolar Brush Cells: glutamatergic interneuron found in the cerebellar cortex.

US Unconditioned Stimulus: stimulus that unconditionally, naturally, and automatically triggers a response.

VN Vestibular Nuclei: any of a group of four nuclei, designated lateral, medial, superior, and inferior, that are located in the lateral region of the rhombencephalon, receive the incoming fibers of the vestibular nerve, and project via the medial longitudinal fasciculus to the abducens, trochlear, and oculomotor nuclei and to the ventral horn of the spinal cord, on both the left and the right sides.

XML eXtensible Markup Language: set of rules for encoding documents in machine-readable form. The design goals of XML emphasize simplicity, generality, and usability over the Internet.

1

Introduction

The third edition of the book *Principles of Neural Science* (which could be considered a general reference of modern neuroscience) comments “Perhaps the last frontier of science —its ultimate challenge— is to understand the biological basis of consciousness and the mental processes by which we perceive, act, learn, and remember”. But this challenge remains still active, and we could consider it one of the main challenges for the current twenty-first century. A deep knowledge of the biological bases of neural systems can lead us to several benefits. Firstly, a further understanding of the brain could make possible the development of innovative treatments for pathologies related with the Central Nervous System. Secondly, the human brain could be considered the best information processing system due to its reliability, accuracy, performance and storage capacity. Being able to *emulate* the way of processing information in biological systems could allow the design of a new generation of processing architectures capable of exceeding the bounds of current systems.

1.1 Computational neuroscience

From that perspective, and largely due to the increasing capabilities of computational resources, computational neuroscience has emerged in the last years as a promising branch in neuroscience. Computational neuroscience refers to the use of mathematical and computational models in the study of neural systems. The design of mathematical and quantitative models has been a key component of research in neuroscience for many decades. Indeed, one of the most celebrated achievements in the field — the *Hodgkin-*

Huxley model for the generation of action potentials [126] in 1952 — was a triumph of the quantitative approach and meant a major boost for the development of new tools. Also, much of what is understood about the functionality of the visual, auditory and olfactory systems, as well as the neural basis of learning and memory, has been informed by mathematical and computational modeling. Nevertheless, it is fair to say that, until recently, computational modeling represented only a small part of the total research effort in neuroscience, which has traditionally been dominated by experimental studies (in fact, neuroscience research positions used to be taken by researchers with a strong background in biological systems). The recent move towards computational modeling has opened up new directions of research, and allowed investigation of issues beyond those that are accessible to direct experimental study. More importantly, it has brought new ideas from fields such as statistical physics, information theory, nonlinear systems theory and engineering into neuroscience, providing a richer conceptual framework for answering the most difficult fundamental questions in the field.

The primary motivation for using computational modeling is, of course, to understand the behavior of the system under study using mathematical analysis and computer simulation. This is certainly the case in neuroscience. Computational modeling has been traditionally used for many other physical systems (such as astronomical systems, fluid flows, mechanical devices, structures, etc.). However, and differently with this kind of systems, the application of computational modeling to living systems — and especially to the nervous system — is specially suitable because biological systems can be seen explicitly as processors of information. Thus, computational models in these systems are not just tools for calculation or prediction, but often elucidate essential functionality. In the case of neuroscience, this can be seen in terms of two related roles served by computational modeling. These are: 1) Determining WHAT the various parts of the nervous system do; and 2) Determining HOW they do it.

Experimental studies of the nervous system at all levels (sub-cellular, cellular and systemic) are critical for understanding the anatomical structures and physiological processes of the system, but these observations must then be organized into a coherent model of system functionality. This is only possible if the appropriate conceptual elements for such a functional description are available. Psychologists and neurologists have traditionally used performance (or its deficits) as the basis for assigning functionality to components of the nervous system, which has produced useful qualitative and phenomenological models. This is the case of the eye-blink conditioning or vestibulo-ocular reflex which have traditionally been closely related to the study of the cerebellum. These are often sufficient for clinical purposes, but provide only limited understanding of the system per se. An alternative (or complementary) approach is provided by viewing the nervous system as acquiring, transforming, storing and using information to control an extremely complex system — the body — embedded in a complex dynamic environment. In this view, the functionality of the system emerges from lower level phenomena such as membrane potential dynamics, dendritic current flows, channel kinetics, synaptic plasticity, etc., much as the functionality of a computer emerges from the currents and voltages in its components.

As with the computer, the emergent functionality of the nervous system depends on the underlying phenomena but cannot be described entirely in their terms. To truly understand this functionality, it is necessary to relate the concrete phenomena measured by experiments to the abstractions of information processing — and ultimately to the phenomena of cognition and behavior —. Computational modeling does this by providing a well-developed formalism relating signals and information. Through such modeling, mathematical and computational models can be applied directly to the nervous system, leading to a coherent quantitative and testable functional description of the brain rather than a qualitative model or a compendium of observations. This conclusion was already foreseen by Andrew Huxley (one of the authors of the Hodgkin & Huxley action potential model) when he commented in his Nobel award lecture in 1963: “an important lesson I learnt from these manual computations was the complete inadequacy of one’s intuition in trying to deal with a system of this degree of complexity”. This sentence gathers one of the principles of every model. Since any system can aspire to model absolutely all the levels of abstraction of the analogous real system, the first decision in developing a model will be the analysis of the factors which might influence the behavior, and choose the level of detail from which our model will work.

The same philosophy must be applied in order to model the central nervous system. It processes information at many scales, ranging from molecules to large networks comprising millions of neurons. For the information-based model of nervous system functionality to work, it is essential explaining how phenomena at each level arise from those at lower levels, e.g., how the recognition of objects in the visual field relates to signals generated by visual system neurons, or how the activity of individual motor neurons produces smooth limb trajectories. Unfortunately, experimental methods often do not provide the data necessary for this. In particular, the data needed to understand how networks of neurons process information collectively is very difficult to obtain. Current technology allows in vivo access to the nervous system mainly at the extremes: high resolution intracellular and extracellular data through single electrode recordings, and low resolution regional activity data through functional magnetic resonance imaging (fMRI) and magnetoencephalography (MEG). Though electrode arrays are now fairly widely used, they still provide extracellular access only to a few hundred neurons at best. However, most functional networks in areas such as the hippocampus and cerebellum (two of the better studied regions) comprise anywhere from a few hundred thousand to several million cells. Information processing in these networks occurs through self-organized dynamic patterns of activity spanning large parts of the system [12, 99, 107, 158, 174]. These emergent patterns can no more be understood by looking at the activity of individual cells (or even a few hundred cells) than the meaning of a book discerned by reading individual letters. Nor can large-scale data from fMRI studies supply the resolution necessary to see these patterns and relate them to interactions between cells.

Computational modeling provides a way out of this dilemma by allowing the study of network models — as large as desired — constructed using neuron models that are themselves based on cell-level data obtained from experiments [69, 100, 172, 226, 240]. These model networks can be simulated computationally under a variety of situations to give insight into how the corresponding networks in the brain might work. Specific issues such as the effect of synaptic modification, modulation by external signals, or the significance of particular connectivity patterns, can be studied, and hypotheses that cannot be tested directly, can be provisionally validated or rejected in simulation. In many cases, models are becoming an indispensable tool in the hypothesize-and-test loop of neuroscience. Computational models allow investigators to try out their “what-if” intuitions in simulation, leading to better hypotheses and better designed experiments with greater likelihood of success. Of course, the quality of the results depends on the quality of the models, but the models have become increasingly good with advances in numerical techniques, computational power and experimental methods [25, 37].

As the focus of interest in neuroscience moves from phenomena to functionality, computational modeling is also being used to address previously inaccessible problems such as the neural basis of cognition and even consciousness [93, 139, 163]. Issues related with representation, intention and executive control are being explored at the edge between neuroscience and artificial intelligence, and the understanding of the brain as an extremely sophisticated information processing system continues advancing in a multidisciplinary way.

1.2 The importance of studying the cerebellum

The cerebellum (whose name translates as little brain in Latin) is a region of the brain that plays a main role in motor control. However, it is also involved in some cognitive functions such as attention and language, and probably in some emotional functions such as regulating fear and pleasure responses [282]. Although the cerebellum does not plan movements, it contributes to coordination, precision, and accurate timing. It receives input from somatosensory systems and from other parts of the brain and spinal cord, and integrates these inputs to fine tune motor activity. Because of this fine-tuning function, illnesses in the cerebellum does not cause paralysis, but instead produces disorders in fine movement, equilibrium, posture, and motor learning.

The number of cells stimated in the cerebellum also make it one of the main parts of the brain. The granule cells are by far the most numerous neurons in the brain: in humans, estimates of their total number average around 50 billion, which means that about three-quarters of the total number of neurons in the brain are cerebellar granule cells. The cerebellum has the appearance of a separate structure attached to the bottom of the brain, tucked underneath the cerebral hemispheres. The surface of the cerebellum is covered with finely spaced parallel grooves, in striking contrast to the broad irregular convolutions of the cerebral cortex. These parallel grooves conceal the fact that the cerebellum is actually a continuous thin layer of neural tissue (the cerebellar cortex), tightly folded in the style of an accordion as shown in Figure 1.1. Within this thin layer there are several types of neurons with a highly regular arrangement, the most important being Purkinje cells and granule cells. This complex neural network gives rise to a massive signal-processing capability, but almost all of its output is directed to a set of small deep cerebellar nuclei lying in the interior of the cerebellum.

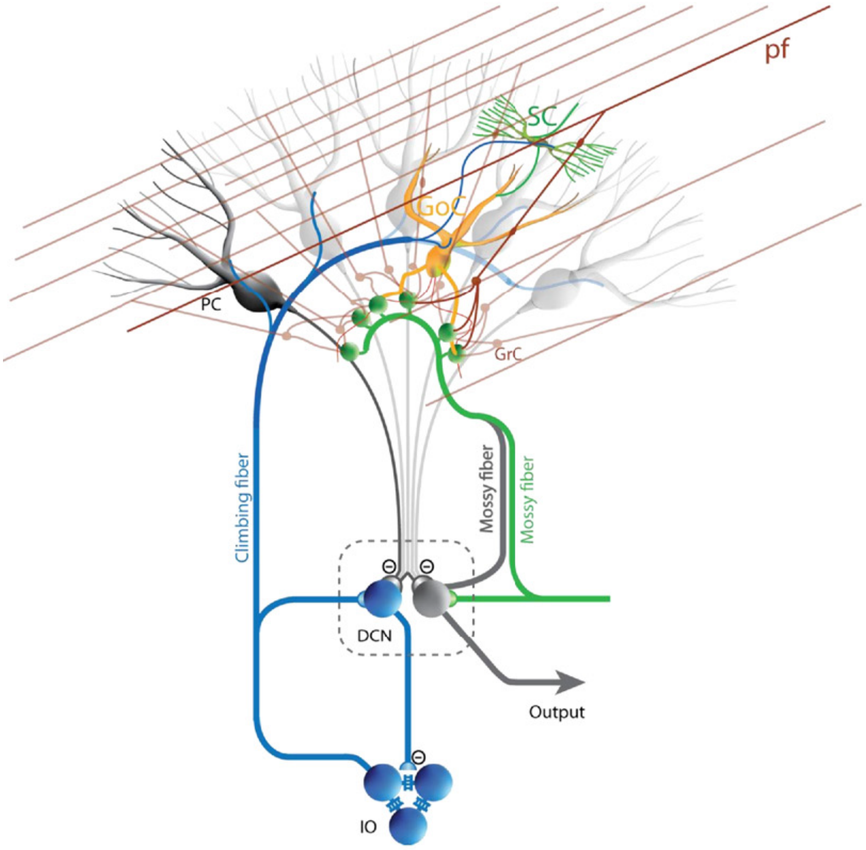


Figure 1.1: The structure of the cerebellum - Diagram showing the main cell types in the cerebellum. Extracted from [67].

In addition to its direct role in motor control, the cerebellum is also necessary for several types of motor learning, the most notable one being learning to adjust to changes in sensorimotor primitives. Several theoretical models have been developed to explain sensorimotor calibration in terms of synaptic plasticity within the cerebellum. Most of them derive from early models formulated by David Marr [192] and James Albus [9], which were motivated by the observation that each cerebellar Purkinje cell receives two dramatically different types of input: on one hand, thousands of inputs from parallel fibers, each individually very weak; on the other hand, input from one single climbing fiber, which is, however, so strong that a single climbing fiber action potential will reliably cause a target Purkinje cell to fire a complex action potential [11]. The basic concept of the Marr-Albus theory is that the climbing fiber serves as a "teaching signal", which induces a long-lasting change in the strength of synchronously activated parallel fiber inputs. Observations of long-term depression in parallel fiber inputs have provided support for theories of this type, but their validity remains controversial (see section 4.2.1 for an extensive review about functional models proposed in the literature).

1.3 General work motivation

The main aim of this work is the study and implementation of a cerebellar model. It is well-known that the cerebellum plays an important role specially in motor control but is also involved in other cognitive functions such as attention and language. The cerebellum does not initiate the movement, but it contributes to coordination, precision, and accurate timing. It receives inputs from sensory systems and from other parts of the brain and spinal cord, and integrates these inputs with motor commands to fine tune motor activity. However, the design of a medium-scale cerebellar model requires several stages before achieving the whole system working.

Firstly, an ultra-fast simulator of biologically realistic neural networks is needed. Indeed, if we expect to control a real robotic system (as opposed to a simulator), the performance requirements lead to a real-time simulator. EDLUT [98] can be considered the only simulator of biological neural networks able to fulfill these requirements as shown in previous literature [40, 227]. Thus, the usage of this biological architectures in order to control robotic systems means a challenge and a novelty itself. However, the efficient table-based simulation strategy (that EDLUT implements) needs an extremely careful validation to avoid potential errors derived from the behavior precompilation in look-up tables. Therefore, as a first stage, in chapter 2 we further develop the EDLUT simulator to accomplish with all the requirements that subsequent models might imply.

Secondly, we can take advantage of this simulation software to carry out an exhaustive study of the influence of different model parameters (and its interpretation in biological systems) at the granular layer. The granular layer can be considered the first processing layer in the cerebellum, and it carries the input signals to the parallel fibers. The importance of the granular layer is out of any doubt since the granular cells (the main component of the granular layer) are the most numerous neurons in the brain (in humans, estimates of their total number average around 50 billion, which means that about three-quarters of the brain's neurons are cerebellar granule cells). But further of this quantitative approach, experimental data have shown the existence of complex signal processing at this layer [58, 59, 64]. Thus, in chapter 3 we address the temporal filtering capabilities at the granular layer.

Finally, the simulation of a whole cerebellum model within a control loop in order to accurately control a robotic arm involves a demanding test bank for our biological system [40]. From an engineering point of view, the best system is that which achieves the best performance (in terms of accuracy, speed...). This point of view could also fit in neuroscience (or neurophysiology), where the biologically inspired system with the best performance might be a clue in order to achieve a deeper knowledge of the real biological system. Following this approach, in chapter 4 we implement a whole cerebelar model capable of accurately control a robotic arm as a learning system.

1.4 Our Contribution

In this work we developped and upgraded a spiking neural network simulation framework capable of simulating networks with a high level of detail in real-time environments: EDLUT [98]. This tool was specially suitable for interacting with external applications, robotic or visual processing systems based on realistic neural networks. A previous work in the literature [227] laid the foundations of the efficient event-driven simulation algorithm based on look-up tables. During this work, this algorithm evolved to a whole simulation environment due to its Graphical User Interface (GUI) and the implemented systems of communication with several tools (by means of files, TCP/IP sockets, application programming interface or the implementation as a Simulink® module). We also implemented in EDLUT the capability of running simulations by using hybrid schemes with time-driven and event-driven cell models and showed the advantages that this simulation strategy can provide in realistic networks such as the granular layer in the cerebellum. Finally, we implemented realistic cell models of most the cells in the cerebellum (such as granule, Golgi, stellate, and Purkinje cells) and Spike-response models of the cuneate nucleus.

In addition to this, we made extensive usage of these implemented tools in order to study the temporal filtering at the granular layer in the cerebellum. Previous studies showed the capability of this layer to produce oscillations in several frequency bands (specially β and γ between 12 and 80 Hz) and transmit these rhythms to other regions of the brain [58]. In this study we showed the influence of each synaptic connection on enhancing these filtering capabilities.

Finally, we developed a biologically plausible model of the scaled-down cerebellum and took full advantage of this in order to control a non-stiff-joint robotic arm while solving a manipulation task. We showed how this control loop including a cerebellum-like module achieves an accurate inference of the modified dynamical model when manipulating heavy tools which substantially affect the base model. Furthermore, we studied the influence of the learning parameters (LTD and LTP) and the way in which feedback information such as the current state can highly improve the learning capabilities (in terms of accuracy and performance).

1.5 Objectives

The main aim of this work is to contribute to the understanding of the central nervous system (and specifically of the cerebellum) from the point of view that simulation studies can provide. Therefore, a biollogically-plausible neural network model can be functionally studied and posteriorly exploited in order to solve practical problems (e.g. robot control). To achieve this objective, this thesis addresses the following separate goals:

- Development and upgrading of an efficient simulation scheme for medium scale spiking neural systems emphasizing the usability of the resulting tools for researching porpuses of neurophisiology groups.

- Implementation of realistic models of the main cerebellar neuron types in the proposed simulation software and analysis of the advantages of using each alternative implementation depending on the properties of the concrete network model.

- Study of temporal processing capabilities of the granular layer (coincidence detection and time slicing), the oscillatory behaviour and the functional role of these emerging properties in the framework of the whole cerebellar models.

- Study of how sensorimotor primitives can be efficiently infered through adaptation of weights (LTD and LTP) at the parallel fibers.

- Evaluation of kinematics and dynamics model inference capabilities at the cerebellum through long term adaptation at parallel fibers to purkinje cell connections.
- Evaluation of how contextual signals can be efficiently used for context switching in the framework of an accurate control task.

1.6 Project Framework

The work described in this document has been developed within two European projects "SENSOrimotor structuring of Perception and Action for emergent Cognition" (SENSOPAC) [2] and "Realistic Real-time Networks: computation dynamics in the cerebellum" (REALNET).

The SENSOPAC project (funded under the EU Framework 6 IST Cognitive Systems Initiative) extended from January, 2006 to July, 2010 in collaboration with 12 institutions from 9 different countries. The SENSOPAC project combined machine learning techniques and modelling of biological systems to develop a machine capable of abstracting cognitive notions from sensorimotor relationships during interactions with its environment, and of generalising this knowledge to novel situations. Through active sensing and exploratory actions the machine discovers the sensorimotor relationships and consequently learn the intrinsic structure of its interactions with the world and unravel predictive and causal relationships. The project has demonstrated how a robotic system can bootstrap its development by constructing generalisation and discovering sensor-based abstractions, based on neuroscience findings on tactile sensory data representation and processing. The research group at the University of Granada was mainly involved in the development of the spiking neuron computation environment (EDLUT) (see chapter 2 in this document) and the application of neuroscientist findings in order to design biologically inspired control systems capable of carrying out manipulating tasks. Figure 1.2 shows the module organization of the SENSOPAC project. University of Granada (and this work as a part) focused in the fourth module dealing between neurophisiologist (module 5) and more abstract and robotic systems (modules 1, 2, and 3).

As a continuation of this project, the REALNET project (funded under the EU Framework 7 Information and Communication Technologies work programme) started in Febrery, 2011 and will extend until Febrery, 2014.This project aims to understand

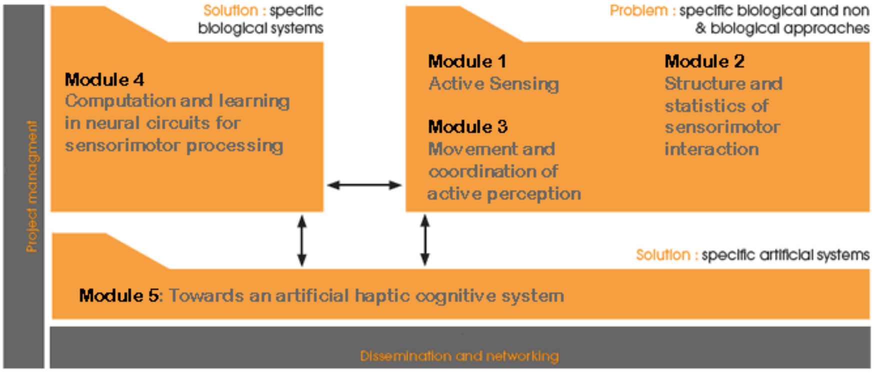

Solution : specific biological systems
Problem : specific biological and non & biological approaches
Module 4
Computation and learning in neural circuits for sensorimotor processing
Module 1
Active Sensing
Module 2
Structure and statistics of sensorimotor interaction
Module 3
Movement and coordination of active perception
Solution : specific artificial systems
Module 5: Towards an artificial haptic cognitive system
Project managment
Dissemination and networking

Figure 1.2: Module structure of the SENSOPAC European Project - Module diagram showing the main tasks developed in the framework of the project. The research group at the University of Granada was mainly involved in Module 4.

circuit computations by using a different approach: to elaborate realistic spiking networks and use them, together with experimental recordings of network activity, to investigate the theoretical basis of central network computation. As a benchmark this project will use the cerebellar circuit. Based on experimental data, this project will develop the first realistic real-time model of the cerebellum and connect it to robotic systems to evaluate circuit functioning under closed-loop conditions. The data deriving from recordings, large-scale simulations and robots will be used to explain circuit functioning through the adaptable filter theory [94]. Within this project, the research group at the University of Granada will be focused on evolving the EDLUT simulator environment in order to simulate more realistic biological structures and increasing the performance of the simulations (see chapter 2).

Since these kind of projects require a multidisciplinary approach, this work mainly presents the results from the biological systems point of view. This is specially relevant in the last part of this work (see chapter 4), where we use a cerebellum-like simulated architecture in order to manipulate tools with a robotic arm. In addition to this, this work implies dealing with robotic system (developing a robotic arm simulator, studying the biologically plausible control loops, or the conversion from/to spiking signals). All these other tasks have been mainly accomplished by Niceto Luque at the University of Granada.

1.7 Chapter Organization

This document has been organized in three main chapters according to the three main issues which have been addressed in this work. First, on chapter 2, we develop and upgrade an extremely efficient spiking neural network simulator (EDLUT). We have included a subsection as the state of the art, where the best known simulation tools are presented and compared (see section 2.2.2). The development in EDLUT has been focused in two main issues: facilitating the usage of EDLUT for neuroscientist researchers (such as the development of a graphical user interface, the new communicating methods or the releasing as a open-source tool), and the implementation of biologically detailed and efficient structures (such as time-driven cell models, stochastic cell models based on look-up tables or hybrid scheme network models). Thus, chapter 2 represents the base tool which will be used in the following works.

In chapter 3, we carry out a simulation study of the temporal filtering properties at the granular layer and the influence of the connections at the different synaptic weights. In this study, we take advantage of the efficient simulation by using EDLUT in order to iteratively simulate the granular layer response to the same stimulation, but using different parameters. The temporal processing at the granular layer is thought to strongly influence not only the cerebellum activity, but the whole brain rhythms too (see the state of the art at section 3.2.1).

Finally, in chapter 4, we study the influence of the synaptic plasticity at the cerebellar parallel fibers and the information convergence at the granular layer in a *real* problem (manipulation task). In order to achieve this learning system, we make use of one of the main functional model of the cerebellum (see section 4.2.1 where we discuss the main functional theories), implement this model by means of a reallistic network model over EDLUT and control fast non-stiff-joint robotic arms accurately.

To summarize, this work represents an evolution from the development of the simulation tool, to the implementation of an accurate robotic system. Each chapter deals with different levels of abstraction (cell models, network models, and system models respectively), and encloses an state of the art of the concrete problem.

2

Simulating the cerebellum: the EDLUT simulation environment

2.1 Abstract

Emerging research areas in neuroscience are requiring simulation of large and detailed spiking neural networks. In this chapter, a previously presented event-driven simulator based on look-up tables (EDLUT) to precompile the behavior of the neuronal models has been further upgraded in order to facilitate the user interaction by means of a Java-based graphical user interface (GUI) and connected with external research systems by several communication methods. In addition to this, to obtain the advantages of an event-driven simulation method and a traditional time-driven method, a hybrid simulation method has been integrated in EDLUT. This method efficiently simulates neural networks composed of several neuron models: highly active neurons or neurons defined by very-complex models are simulated using a time-driven method whereas other neurons are simulated using an event-driven method based on lookup tables. To perform a comparative study of this hybrid method in terms of speed and accuracy, a model of the cerebellar granular layer has been simulated. The performance results showed that a hybrid simulation can provide considerable advantages when the network is composed of neurons with different characteristics. Finally, both alternatives (time-driven and event-driven) of Leaky Integrate and Fire (LIF) and stochastic Spike Response Models (SRM) have been implemented and conveniently validated. All these developments and advances have been made available under GPLv3 license on the EDLUT official website

http://edlut.googlecode.com.

2.2 Introduction

In the last years, the spiking neural network simulation tools have strongly evolved, from the very first researchers who implemented their own simulators to reproduce a concrete feature of the central nervous system, to the current simulation environments, capable of modelling the morphology and the topology of thousands of cells in great detail using the processing facilities of the most powerful supercomputers.

However, the issue of simulating the central nervous system is far from being resolved. It must not be forgotten that only the human brain is composed of more than 100 billions of neurons [149]. This number is still unreachable for the last simulation systems. Thus, a very hard and imaginative work will have to be carried out during upcoming decades in order to achieve one of the major challenges in the twenty-first century.

In this chapter, we will start presenting the main simulation strategies and tools, paying special attention to the implementation details which make these tools efficient, and afterwards we will focus on the development of the EDLUT simulation environment [227], a simulation tool capable of running networks including thousands of detailed cells in real-time in an single processor.

In this work, the EDLUT event-driven simulation scheme has been successfully evolved from the basic simulation algorithm to a whole simulation environment. First, the original source code (presented in [38] was reimplemented in order to make it more readable and modularly scalable (in terms of functionality), documented, and freely released on its current website [98]. Posteriorly, and following the same line, a Java-based graphical user interface (GUI) was developed. In addition to this, several other functionalities have emerged from the daily usage of EDLUT: simulation of time-based cell models and hybrid (both time-based and event-driven) neural networks, design and implementation of new realistic cell models, such as stochastic SRM and Leaky Integrate-and-Fire (used in chapters 3 and 4), connectivity with many other tools by means of TCP/IP sockets, Matlab/Simulink® interface and its own C++ API. All these developments have been addressed, tested, and used during this thesis.

2.2.1 Simulation strategies: event-driven vs time-driven

Many research projects are studying and modeling nervous circuits of specific brain areas. To accomplish these tasks, well-known simulators such as GENESIS [25] and NEURON [124] have been traditionally used since they provide a good accuracy when simulating detailed biophysical models of neurons. Their main drawback is the low simulation speed, which is caused by the simulation method that they mainly employ: time-driven simulation with intensive numerical calculation. This simulation method divides the simulated time into short time steps and in each step the neural state variables are approximated and updated through a numerical analysis method [211]. This iterative processing involves an intensive computation which hinders the simulation of large scale neural networks.

The demand for fast simulations of neural networks has given rise to the application of another simulation method: event-driven simulation. This simulation method only computes and updates the neural state variables when a new event modifies the normal evolution of a neuron, that is, when an input is received or an output is produced. At the same time, it is known that most information transmission in biological neural circuits is carried out by the so-called spikes. These events are relatively infrequent and localized in time: less than 1% of neurons are simultaneously active [149] and the activity is extremely sparse in many nervous areas such as the granular layer in the cerebellum [49]. This makes event-driven simulation schemes particularly efficient.

Most common event-driven simulators [80, 278] use relatively-simple neural models described by equations which can be evaluated repeatedly at arbitrary times (e.g. the Spike-Response model). However, even the limited complexity of these models makes it difficult to predict the future behavior of a neuron, especially to detect the nearest threshold-crossing point that corresponds to the next firing time [81, 188]. To mitigate these two limitations (i.e. model-complexity restriction and firing-time prediction which allows a straightforward event-driven simulation) EDLUT (Event-Driven neural simulator based on LookUp tables) was implemented [227]. This application software is an open source project (`http://edlut.googlecode.com`) for efficient simulation of biological neural networks. It is of particular interest in the field of neurobotics and embedded neural computing in which real-time processing is required, for example, for experiments which include perception-action loops.

2.2.1.1 Time-driven simulation models

The time-driven simulation paradigm consists in creating an iterative process which divides the simulated time into short time steps and in each step the system updates all the state variables. However, the update of the state variables may imply a huge computational load (the number of equations to calculate is determined by the level of detail of the model, but a single compartment Hodgkin & Huxley cell model can include about 5 equations per cell). Algorithm 1 shows a generic time-driven pseudocode. Note that the *UpdateSimulationState*(...) function encapsulates the update of each cell state.

Algorithm 1 Generic time-driven simulation engine. A generic simulation algorithm is quite simple. Each simulation step (determined by the *dt* parameter), the simulation state is updated to the current time. The parameter *dt* determines the simulation accuracy.

```
program TimeDrivenModel ()
    InitializeSimulation(...);
    While t < FinalTime then
       NewState = UpdateSimulationState(dt);
       t = t + dt;
    EndWhile;
    GetSimulationStatistics();
end.
```

This kind of simulation model is specifically designed for biological models which include differential equations in their state definition, and can not be calculated by event-driven strategies. The accuracy of this kind of methods is determined by the *dt* parameter as shown in figure 2.1: a tiny value of this parameter will lead us to a high level of detail in the simulations, but also produce a dramatic fall in the simulation speed.

This is why NEURON developers introduced the concept of variable time step [125]. In this case, the simulation loop includes the estimation of the *dt* parameter depending on the current cell state and the previous received activity [37], as shown in algorithm 2. The usage of a variable time step benefits not only the simulation efficiency (when the

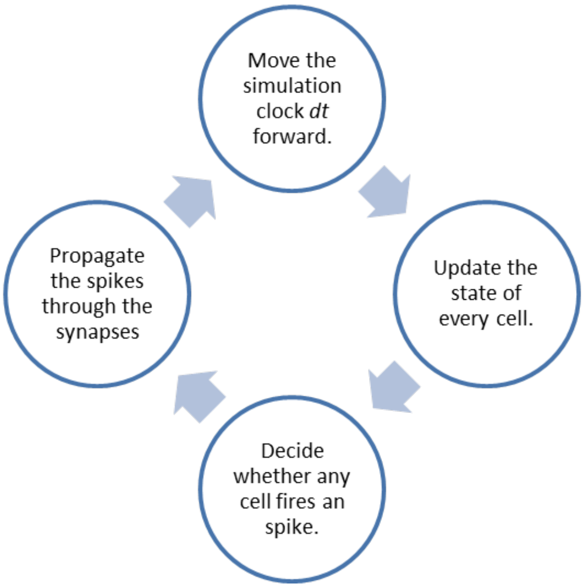

Move the simulation clock *dt* forward.
Update the state of every cell.
Decide whether any cell fires an spike.
Propagate the spikes through the synapses

Figure 2.1: Time-Driven Simulation Scheme - This figure shows the time-driven engine steps. Note that the clock moves forward by previously defined time steps dt.

state variables keep stable for a long time, the time-step will be progressively extended and in the opposite, when the state variables fast vary along the time, the time-step will be shorted to fulfil the accuracy condition) but also the accuracy, as far as using these techniques the error always keeps within an admissible range.

Therefore, this kind of simulation strategies are specially suitable for modelling systems where the state equations follows a continuous evolution (they generally are defined by means of differential equations) and can not be reformulated as event-driven equations (equations which can be calculated at arbitrary times). On the other hand, the efficiency of this model is strongly related to the number of equations to be integrated. As far as the number of equations might be proportional to the number of cells in the whole network, these algorithms are not recommended for large scale simulations.

2.2.1.2 Event-driven simulation models

Contrary to time-driven simulation models, the event-driven simulation schemes are characterized by calculating only at arbitrary times, when a new event occurs. This kind of scheme perfectly fits with the concept of spiking neural network, where the information transmission only takes place at discrete times when a spike is elicited.

Algorithm 2 Generic time-driven simulation engine with variable time-step. This simulation algorithm includes an additional step in which the next time step is calculated based on the estimated error and the minimum accuracy required.

```
program TimeDrivenModel ()
    InitializeSimulation(...);
    While t < FinalTime then
       NewState = UpdateSimulationState(dt);
       dt = StimateTimeStep(Accuracy);
       t = t + dt;
    EndWhile;
    GetSimulationStatistics();
end.
```

However, in order to implement an event-driven cell model, we have to overcome the problem of reformulating the state equations to be calculated only when event happens. Even though this issue usually is not so trivial, there are specific approximations to complex models which mimic the original time-driven models.

Algorithm 3 shows the pseudocode of a simple event-driven algorithm. In an event-driven algorithm, the time advance to the next time when some kind of event has occurred as shown in figure 2.2. In this way, these systems avoid the calculation of the neural state when the network is inactive, resulting in the increasing of the efficiency Therefore, these algorithms are suitable for neural networks where the network activity is sparse, such as the granular layer in the cerebellum [49].

However, event-driven methods sometimes have to use approximation methods such as precalculated look-up tables which allows precompiling the behaviour of the differential equations as a function of the input stimuli (excitatory and inhibitory conductance) and the current state (membrane potential) [227]. The usage of those approximation methods shall be very accurate in order to avoid possible appearance of simulation errors by accumulation.

In short, we can conclude that while time-driven schemes will be preferred to simulate small-sized networks with great detail or extremely complex models. On the other hand, event-driven methods will be suitable for problems where the performance is an important requirement (such as spiking neural networks integrated in control or

Algorithm 3 Generic event-driven simulation engine. In this algorithm, the state will be updated when a new event is received by the cell. After processing the spike, the next fired spikes (and the refractory period) will be predicted and inserted in the heap (sorted by the occurrence time).

```
program TimeDrivenModel ()
    InitializeSimulation(...);
    While heap.IsNotEmpty() or t < SimulationTime then
       CurrentEvent = heap.TopEvent();
       t = CurrentEvent.time();
       NewState = UpdateSimulationState(t-t0);
       NewEvents = PredictNextEvents(NewState);
       heap.insert(NewEvents);
       t0 = t;
    EndWhile;
    GetSimulationStatistics();
end.
```

real-time systems) or simulating medium-large neural networks. However, other factors should also be evaluated before choosing the appropriated simulator, such as the possibility of parallelization, the existence of the model previously tested, or the available simulation environment.

2.2.2 The state of the art: Actual simulation tools

Currently, a large quantity of spiking neural network simulation tools are available on the web and are continuously evolving to be more efficient and capable of simulating as much cells and as in much detail as possible. However, these two aspects traditionally have been considered as opposite: a system which implements very detailed cellular and synaptic models will require a considerable computational effort to simulate the whole system. In the same way, a system which is expected to simulate large networks and fast will have to reduce the level of detail in its cells and synapses.

In this section, we will review the best-known free available simulation tools and we will discuss the main advantages and drawbacks, or the kind of problems these tools are specially suitable. A deeper review about this topic can be found in [28], which is the

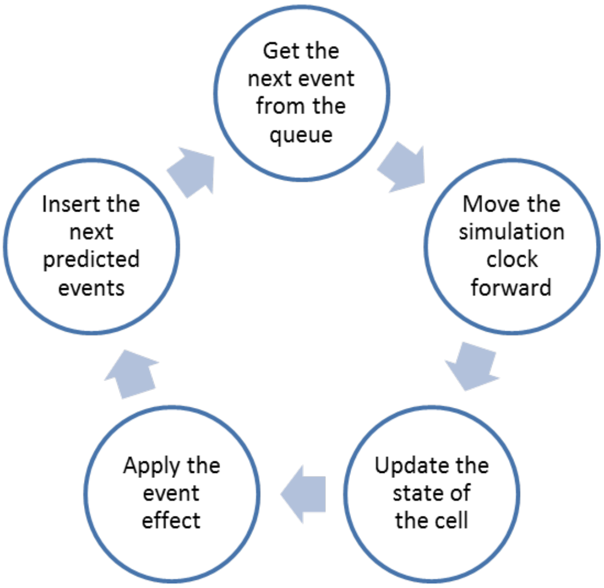

Get the next event from the queue
Move the simulation clock forward
Update the state of the cell
Apply the event effect
Insert the next predicted events

Figure 2.2: Event-Driven Simulation Scheme - This figure shows the event-driven engine steps. Note that the clock moves forward only to arbitrary times when events occur.

main information source of this subsection as well as our own experience in developping models for some of these environments.

2.2.2.1 NEURON

Neuron is a simulation environment for empirically-based simulations of neurons and networks of neurons. This software is specially designed for researchers with a strong neurophysiological background, although knowledge of computational basis of simulation can help to accelerate the model simulation. *Neuron* can be considered by far the most popular simulation tool between neurophisiology laboratories. Since it started its development in the early 1990s, and up to 2007, more than 600 scientific papers and books have described neuron models from a membrane patch to large scale networks with tens of thousands of artificial spiking cells (*Neuron* distinguishes between the cells which are defined based on their morphological properties and those implemented by means of behavioral equations such as the next-firing time or the current state, the artificial spiking cells) [36].

Neuron has several features that facilitate conceptual control of the experiments [28]. These features lean on the *native syntax* specification of model properties: that is, most key attributes of biological neurons and networks have direct counterparts in *Neuron*. For instance, the gating properties of voltage- and ligand-gated ion channels are specified with kinetic schemes or families of Hodgkin & Huxley style differential equations. Thus, *Neuron* users never have to deal directly with compartments. Instead, cells are represented by unbranched neurites, called sections, which can be assembled into branched architectures (the topology of a model cell). Each section has its own anatomical and biophysical properties (which have a real anatomical equivalent), plus a discretization parameter that specifies the local resolution of the spatial grid. The properties of a section can vary continuously along its length, and spatially inhomogeneous variables are accessed in terms of normalized distance along each section [37, 124]. Once the user has specified the cell topology, and the geometry, biophysical properties, and discretization parameter for each section, *Neuron* automatically sets up the internal data structures that correspond to a family of ODEs (Ordinary Differential Equations) for the model's discretized cable equation. This high level of detail allows the development of new models based on experimental measures of cell morphology.

In addition to this, several mechanisms have been developed in order to achieve accurate and efficient models. Firstly, *Neuron*'s spatial discretization of conductance-based model neurons uses a central difference approximation that is second order correct in space. The discretization parameter for each section can be specified by the user, or assigned automatically according to the *d_lambda* rule [37, 124]. Moreover, *Neuron*'s computational engine uses algorithms that are tailored to the model system equations [122, 123, 124]. In order to advance the simulation time, *Neuron* implements natively a time-driven simulation scheme for traditional cell/synapsys models (see section 2.2.1.1) by using fixed step backward Euler and Crank-Nicholson, but allowing the development of event-driven cell models (see section 2.2.1.2), such as simple integrate-and-fire models. Following this strategy, networks of artificial spiking cells are solved analytically by a discrete event method that is several orders of magnitude faster than continuous system simulation [36, 124]. However, the implementation of event-driven models (artificial spiking cells) only allows those models which can be analytically solved (the current state and the next firing time). This restriction limits the number of models which can be simulated by means of this extremely efficient strategy (for example, Hodgkin & Huxley models or leaky integrate-and-fire models with reversal potential are not suitable to be implemented in this way).

Regarding the interaction between users and simulator, models can be created by writing programs in an interpreted language based on hoc [124], which has been enhanced to simplify the task of representing the properties of biological neurons and networks. Users can extend *Neuron* by writing new function and biophysical mechanism specifications in the NMODL (Neuron MODeling Language) language, which is then compiled and dynamically linked. Even though there is also a GUI for conveniently building and using models and running simulations; the use of scripts and NMODL files is frequently preferred for the experiment reproducibility. However, this language has partially been replaced by a Python API (application programming interface) which easily integrate the simulation of biological models with *Neuron* and the power of scientific data analysis and representation tools implemented in Python.

Another feature which we should not obviate (specially due to the spread of using clusters and supercomputers in research) is the parallelization capability of the simulation. Several kinds of parallel processing are supported in *Neuron*.

- Multiple simulations distributed over multiple processors, each processor executing its own simulation.

- Distributed network models with gap junctions. This feature has allowed the simulation of thousands of cells as the realistic model of the granular layer at the cerebellum [251].

- Distributed models of individual cells (each processor handles part of the cell). This kind of parallelization is still in an early development stage, and therefore, setting up distributed models of individual cells requires considerable effort.

Currently, the *Neuron* project is directed by Michael Hines, who develops almost all the code. However, one of the strongest advantage of *Neuron* is the existence of a large community of users who have worked on specific algorithms, written or tested new code, implemented new synaptic, cell and network models and made all this code available. More information about this project, documentation, tutorials, courses and even the source code can be found in the official website at `http://www.neuron.yale.edu/neuron/`.

2.2.2.2 GENESIS

Genesis (the General Neural Simulation System) was given its name because it was designed, at the outset, to be an extensible general simulation system for the realistic modelling of neural and biological systems [25]. Typical simulations that have been performed with *Genesis* range from subcellular components and biochemical reactions to complex models of single neurons [70], simulations of large networks [207], and systems-level models [256]. However, the concept of *realistic models* are defined as those models that are based on the known anatomical and physiological organization of neurons, circuits and networks [25]. For example, realistic cell models typically include dendritic morphology and a large variety of ionic conductances, whereas realistic network models attempt to duplicate known axonal projection patterns. However, *Genesis* comes mainly equipped with mechanisms to easily create large scale network models made from single neuron models that have been previously implemented.

Typical *Genesis* neurons are multicompartmental models with a variety of Hodgkin & Huxley type voltage- and/or calcium-dependent conductances. However, users have added, for example, the Izhikevich [140] simplified spiking neuron model, and they could also add IF (Integrate-and-Fire) or other forms of abstract neuron models.

In *Genesis*, a simulation is constructed of basic building blocks (elements). These elements communicate by passing messages to each other, and each contains the knowledge of its own state variables (fields) and the methods (actions) used to perform its calculations or other duties during a simulation (in a similar way to the development with object-oriented programming languages). *Genesis* elements are created as instantiations of a particular precompiled object type that acts as a template. Model neurons are constructed from these basic components, such as neural compartments and variable conductance ion channels, linked with messages. Neurons may be linked together with synaptic connections to form neural circuits and networks. This object-oriented approach is central to the generality and flexibility of the system, as it allows modelers to easily exchange and reuse models or model components.

The simulation implementation is quite similar to the *Neuron* model. *Genesis* uses an interpreter and a high-level simulation language to construct neurons and networks. This use of an interpreter with pre-compiled object types, rather than a separate step to compile scripts into binary machine code, gives the advantage of allowing the user to interact with and modify a simulation while it is running, with no sacrifice in simulation speed. Commands may be issued either interactively to a command prompt, by use of simulation scripts, or through the graphical interface. In addition to this, there are also a number of *device objects* that may be interfaced to a simulation to provide various types of input to the simulation (pulse and spike generators, voltage clamp circuitry, etc.) or measurements (peristimulus and interspike interval histograms, spike frequency measurements, auto- and cross-correlation histograms, etc.). It is also remarkable that there is a parallel version for *Genesis* (called *Parallel Genesis*) implemented on the basis of MPI (Message Passing Interface) and/or PVM (Parallel Virtual Machine).

Finally, the *Genesis* development team is participating in the NeuroML [55, 106] project, along with the developers of *Neuron*. This will enable *Genesis 3* to export and import model descriptions in a common simulator-independent XML format. More information about new developments and future plans in *Genesis*, as well as documentation and further tutorials can be found on the official website at `http://genesis-sim.org/`.

2.2.2.3 NEST

The *Nest* simulator can be considered the third contender in the battle of the most-used neural network simulators. The *Nest initiative* was founded as a long term collaborative project to support the development of technology for neural systems simulations [82]. The *Nest* simulation tool is the reference implementation of this initiative.

The domain of *Nest* is large neuronal networks with biologically realistic connectivity. Using this software, network models have been developed including more than 10^5 neurons [202]. Typical neuron models in *Nest* have one or a small number of compartments. This simulator is specially though emphasicing the efficient representation and update of synapses. Furthermore, in many applications, network construction has the same computational costs as the integration of the dynamics. *Nest* is designed for large scale simulations where performance is a critical issue. Thus, *Nest* developers argued that when comparing different integration strategies, one should evaluate the efficiency, i.e. the simulation time required to achieve a given integration error, rather than the plain simulation time [203].

The primary user interface of *Nest* is a simulation language interpreter which processes a rather high level expressive language with an extremely simple syntax which incorporates heterogeneous arrays, dictionaries, and functions . There is no built-in graphical user interface as it would not be particularly helpful in *Nest*'s domain: network specification is procedural, and data analysis is generally performed off-line for reasons of convenience and efficiency. The simulation language is used for data pre- and post-processing, specification of parameters, and for the compact description of the network structure and the protocol of the virtual experiment. The neuron models and synapse types are implemented as derived classes on the C++ level such that all models provide the same minimal functionality. In addition to this, in the framework of the *Facets* project [1] a Python interface has been created.

The simulation kernel of *Nest* supports parallelization by multi-threading and message passing, which allows distribution of a simulation over multiple processors of an SMP (Symmetric MultiProcessing) machine or over multiple machines in a cluster. Communication bulk is minimized by storing synapses on the machine where the postsynaptic neuron is located [202]. The user only needs to provide a serial script, as the distribution is performed automatically.

The *Nest* simulator is provided to the scientific community under an open source license through the NEST initiative's website `http://www.nest-initiative.org`. At present NEST is used in teaching at international summer schools and in regular courses at the University of Freiburg.

2.2.2.4 Other simulation tools

Although there are many other simulation tools, we will only capture the essence of some of them to avoid the unnecessary extension of this document.

XPPAUT is a general numerical tool for simulating, animating, and analyzing dynamical systems. These can range from discrete finite state models (McCulloch-Pitts) to stochastic Markov models, to discretization of partial differential and integro-differential equations. *XPPAUT* was not specifically developed for neural simulations but due to its ability to provide a complete numerical analysis of the dependence of solutions on parameters (bifurcation diagrams) it is widely used by the community of computational and theoretical neuroscientists. While it can be used for modest sized networks, it is not specifically designed for this purpose and due to its history, there are limits on the size of problems which can be solved (about 2000 differential equations is the current limit). However, *XPPAUT* has become an useful tool for bifurcation analysis in order to describe the behavior of cell models under stimulation. More information about this simulation tool can be found on [89] and the official website `http://www.math.pitt.edu/~bard/xpp/xpp.html`.

The *circuit simulator* (*CSIM*) is a tool for simulating heterogeneous networks composed of (spike emitting) point neurons. *CSIM* is intended to simulate networks containing a few neurons, up to networks with a few thousand neurons and on the order of 100000 synapses. It was written to do modeling at the network level in order to analyze the computational effects which can not be observed at the single cell level. The core of CSIM is written in C++ which is controlled by means of Matlab (there is no standalone version of *CSIM*). *CSIM* adopts an object oriented design for *CSIM* which is similar to the approaches taken in *Genesis* and *Neuron*. That is there are objects (e.g. a LifNeuron object implements the standard LIF model) which are interconnected by means of well defined signal channels. From the site `http://www.lsm.tugraz.at/csm` precompiled versions for Linux and Windows are available.

SPLIT is a tool specialized for efficiently simulating large-scale multicompartmental models based on HH formalism. It should be regarded as experimental software for demonstrating the possibility and usefulness of very large scale biophysically detailed neuronal network simulations. Recently, this tool was used for one of the largest cortex simulations ever performed [85]. It supports massive parallelism on cluster computers using MPI. The model is specified by a C++ program written by the SPLIT user. This program is then linked with the SPLIT library to obtain the simulator executable. Currently, there is no supported graphical interface, although an experimental Java/QT-based graphical interface has been developed. *SPLIT* should be regarded as a pure, generic, neural simulation kernel with the user program adapting it into a simulator specific to a certain model.

In the last years, the *Brian* simulator [108] has been released under the CeCILL license. *Brian* is a simulator for spiking neural networks available on almost all platforms. The motivation for this project is that a simulator should not only save the time of processors, but also the time of scientists. Brian is easy to learn and use, highly flexible and easily extensible. The *Brian* package itself and simulations using it are all written in the Python programming language. In Brian, cell models are defined directly by their equations and some parameters such as the threshold and reset value (for integrate-and-fire models) can be customized. Brian uses vector-based operations (using NumPy and SciPy) to simulate neural populations very efficiently. For large networks, the cost of interpretation is small and the speed is comparable to C code. *Brian* can also be used with the Parallel Python package to run independent simulations on a cluster or on different processors (e.g. running a simulation with different parameter values). However, due to the easy and fast design of new models, *Brian* simulator is mainly used with didactic purposes.

2.2.2.5 Discussion

Even though we are sure that further simulation environments exist, the above commented tools provide an wide view of the state of the art in this discipline and allow us analyzing the advantages and weaknesses of each one of them.

First, there is not a single tool which can be considered the best choice. Even whether *Neuron* could be considered the worthiest choice when simulating small/medium sized systems where the cell model is expected to influence the whole system result, the same simulator should not be chosen when large sized systems are required to study the influence of the connectivity parameters (such as the convergence/divergence rates). In the same way, *Genesis* is specially valuable for providing a generic environment capable of simulating systems with very different levels of detail, but this feature has the great disadvantage of being less efficient than more specific simulators. Thus, a detailed analyses of the suitable tool and a wide knowledge of the features of each simulator will be fully recommended in order to achieve works both efficient and accurate.

Second, the existence of so many different simulators makes the reusability of the experiments becomes an important issue. As we have previously commented, using several simulators generally implies learning different definition or scripting languages, different ways of stimulating the cells, distinct levels of abstraction (and defining different characteristics in the cell/network/simulation)... Thus, some years ago, the thought of reusing *Neuron* models for reproducing the simulations in *Genesis* could be considered Utopian. However, in the last years, several initiatives have emerged related to this issue. In this sense, *PyNN* [68] is a simulator-independent language for building neuronal network models. This implies that a user can write the code for a model once, using the *PyNN* API and the Python programming language, and then run it without modification on any simulator that *PyNN* supports (currently *Neuron*, *Nest*, *PCSIM* and *Brian*). The *PyNN* API aims to support modelling at a high-level of abstraction (populations of neurons, layers, columns and the connections between them) while still allowing access to the details of individual neurons and synapses when required. *PyNN* provides a library of standard neuron, synapse and synaptic plasticity models, which have been verified to work the same on the different supported simulators. *PyNN* also provides a set of commonly-used connectivity algorithms (e.g. all-to-all, random, distance-dependent, small-world) but makes it easy to provide your own connectivity in a simulator-independent way.

However, the use of non-standard cell models is still an open issue in this framework. *PyNN* translates standard cell-model names and parameter names into simulator-specific names, e.g. standard model IF_curr_alpha is iaf_neuron in *NEST* and StandardIF in *NEURON*, while SpikeSourcePoisson is a poisson_generator in *Nest* and a NetStim in *Neuron*. Only a few cell models have been implemented so far. Thus, *PyNN* will have to provide any alternative to use platform-dependant code in order to use non-standard cell models.

The second trend in this issue is *NeuroML* [105]. *NeuroML* is an international, collaborative initiative to develop a language for describing detailed models of neural systems. The aims of the *NeuroML* initiative are:

- To create specifications for a language in XML to describe the biophysics, anatomy and network architecture of neuronal systems at multiple scales.

- To facilitate the exchange of complex neuronal models between researchers, allowing for greater transparency and accessibility of models.

- To promote software tools which support *NeuroML* and support the development of new software and databases.

- To encourage researchers with models within the scope of *NeuroML* to exchange and publish their models in this format.

The *NeuroML* project focuses on the development of an XML (eXtensible Markup Language) based description language that provides a common data format for defining and exchanging descriptions of neuronal cell and network models. The current approach in the project uses XML schemas to define the model specifications. In order to achieve a complete compatibility with simulators which define different levels of abstraction, the *NeuroML* model description language is being developed in Levels, where each Level concentrates on a particular biophysical scale:

- Level 1 focuses on the anatomical aspects of cells and consists of a schema for Metadata and the main *MorphML* schema. This Level is most suitable for tools (such as NeuronLand) which focus solely on detailed neuronal morphologies. Note that the Metadata elements are used at this and higher Levels.

- Level 2 adds the ability to include information about the biophysical properties of cells using the Biophysics schema and also includes the properties of channel and synaptic mechanisms using *ChannelML*. This Level of model description is useful for applications (such as *Neuron*) which can be used to simulate neuronal spiking behaviour.

- Level 3 adds the ability to specify cell placement and network connectivity using *NetworkML*. Files containing positions and synaptic connections in *NetworkML* can be used by applications to exchange details on (generated) network structure. Full Level 3 files containing cell structure and 3D connectivity can be used by applications such as *neuroConstruct* [104] for building and analyzing complex network of biophysically detailed neurons.

Therefore, there is no doubt that the current trend is the integration of the existing tools by means of common definition languages and generic levels of abstraction. Even when these commented projects are in a very early stage of development, the integration of the most widely used simulation tools (mainly *Neuron* and *Genesis*) foretell the generalization in a short/medium term.

Finally, but also related with the second point, the last weakness of these simulation tools is the lack of integration with different systems. Even though spiking neural networks have been used for different simulation experiments, such as visual systems (vestibulo occular reflex and eye-blink conditioning experiments) or robotic arm controlling, this kind of experiments seems to be forgotten by the studied general simulators. The integration of these tools with general propose languages (such as C/C++ or Python) by means of APIs would be a first stage, but the adaptation to run within more industrial environments (such as Simulink) could help to the general use of this kind of systems for many different purposes.

2.3 The EDLUT simulation environment

Due to the lack of a spiking neural network simulator which accomplish with the near real-time requirements using detailed cell models, the development of a new simulation tool was needed. Thus, based on the previous works by Ros et al. [38, 227], a further development has been carried out. In this work, an event-driven simulation strategy was proposed, but an innovative concept was introduced: the cell behavior precalculation in look-up tables. In this way, the complexity of the cell models do not represent a problem for the efficiency of the simulation: every simulation state value can be retrieved just by querying the cell model table. In this way, this simulator can make efficient use of large memory resources as ”computing resources” since their use in terms of look-up-tables of precompiled estimations allow fast simulations without intensive numerical calculation (but rather memory accesses). Thus, EDLUT allows splitting spiking neural network simulations in two stages:

1. Cell behaviour characterization. In this first stage each cell model is simulated reiteratively with different input stimuli and from different cell states. This allows scanning the cell behaviour (for example the state variable values, the next firing time,...) which is compiled into lookup-tables. Usually the cell model dynamics are defined with several differential equations that define the cell behaviour. Therefore at this stage the tool uses conventional numerical methods (such as Runge-Kutta method) to approximate the cell states after receiving a specific stimulus. This represents a massive simulation load for each cell type but, but it needs to be done only once. The results are stored in well structured tables and in this way, numerical calculation during network simulations can be avoided. Furthermore, simulations can be done adopting an event-driven simulation scheme which accelerates significantly the simulation speed.

2. Network simulation towards system behaviour characterization. At this stage, multiple network simulations are run with different weight configuration or even including Spike-Time Dependent (STD) learning rules. This stage does not require intense numerical calculation; the cell states at different times are retrieved from the lookup-tables, allowing massive simulations with little computational expenditure.

This two-stage simulation process (see figure 2.3 for further detail) entails a further advantage. When the generic event-driven strategy was above commented, an special mention was remarked to the cell models which could be simulated using this scheme. In general, event-driven tools only allow the simulation of neuron models whose modelling equations could be calculated at arbitrary times. However, the EDLUT simulation strategy allows the characterization of the differential equations in the first stage by means of classical integration methods, and the subsequently usage of these values in the network simulation stage. Thus, the cell model equations does not mean a limitation for this simulator.

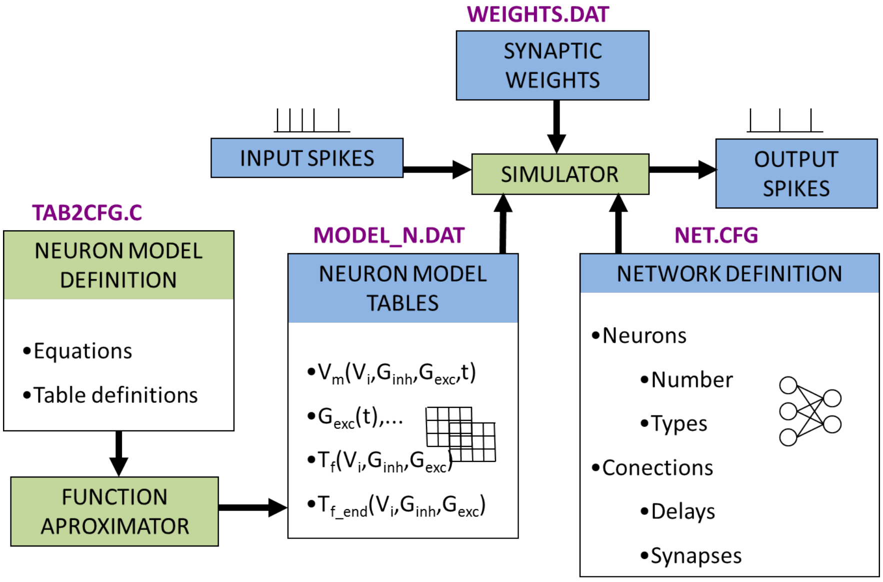


Figure 2.3: EDLUT simulation kernel architecture - This figure shows the original simulation scheme and the files which are needed to simulate an spiking neural network.

The EDLUT simulation kernel was originally implemented in C to obtain higher performances in very complex simulations. In order to keep these high performances, but gain code legibility and make EDLUT kernel easier to extend with new features, it has been fully reimplemented. Therefore, the event treatment in EDLUT has been done as homogeneously as possible. All the events which can be processed inherit from a base class event. In this way, adding new functionality will be as simple as define another inherited class and its functions which specify the treatment of this kind of event. The original types of events that EDLUT considered were the following:

- The input spike type. Following the same methodology proposed in EDLUT, external inputs reach the network through *stub* neurons. Thus, an incoming spike does not modify the *stub* neuron state, it only generates the corresponding input spikes (propagated spike events [227]) for every neuron which is connected to the output of this neuron. In this way, input spike processing is really simple, it only inserts the first (the earliest) output spikes produced by the input neuron.

- The internal spike. These are directly fired by the dynamics inherent to the neurons (as a response to input conductance or as a consequence of the self-generated activity of the neurons). In these events, a different processing is carried out depending on whether the neuron model is time-driven or event-driven. The last one updates the neural state and after that, checks if other spike is scheduled for later, and in that case it inserts it in the simulation heap. However, this state-update stage is not necessary when simulating event-driven neural models due to the periodical evolution of the state variables. Finally, the first propagated spike is inserted in the simulation heap.

- The last implemented stage of the spike processing is the propagated-spike event which corresponds to one spike being transmitted between neurons. Thus, the input spike will be processed and only if the neural model is event-driven the state will be updated. After that, the next propagated spike (generated by the same source neuron) will be inserted in the simulation heap.

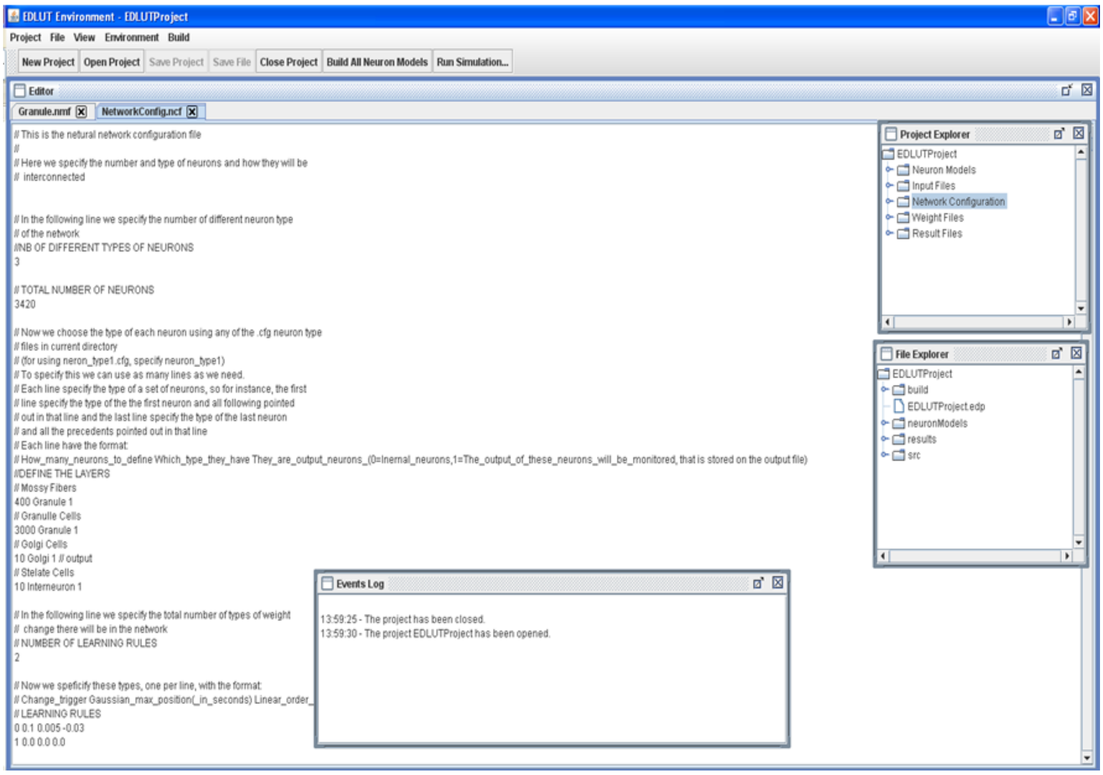
In addition to this, some other different types of events have been implemented in order to handle different situations which could happen during the simulation (such as communication with external systems following an simulation loop, saving the synaptic weights periodically to explore learning processes,etc.).

In conclusion, a extensive work has been accomplish with the aim of making EDLUT easier to utilize for potential users and adding the necessary functionality to address the experiments which are detailed in subsequent sections.

2.3.1 EDLUT graphical user interface

In order to facilitate the simulation of small-size networks and help beginners to understand the working methodology with EDLUT, a Java-based graphical user interface

(GUI) has been developed. The choice of Java is based on the idea of making EDLUT portable to different platforms and easy to run by non-expert users. This GUI pursues automatize the needed tasks before extracting simulation results. Figure 2.4 shows the view of the EDLUT interface design.



```
// This is the netural network configuration file
//
// Here we specify the number and type of neurons and how they will be
//  interconnected


// In the following line we specify the number of different neuron type
// of the network
//NB OF DIFFERENT TYPES OF NEURONS
3

// TOTAL NUMBER OF NEURONS
3420

// Now we choose the type of each neuron using any of the .cfg neuron type
// files in current directory
// (for using neron_type1.cfg, specify neuron_type1)
// To specify this we can use as many lines as we need.
// Each line specify the type of a set of neurons, so for instance, the first
// line specify the type of the the first neuron and all following pointed
// out in that line and the last line specify the type of the last neuron
// and all the precedents pointed out in that line
// Each line have the format:
// How_many_neurons_to_define Which_type_they_have They_are_output_neurons_(0=Inernal_neurons,1=The_output_of_these_neurons_will_be_monitored, that is stored on the output file)
//DEFINE THE LAYERS
// Mossy Fibers
400 Granule 1
// Granulle Cells
3000 Granule 1
// Golgi Cells
10 Golgi 1 // output
// Stelate Cells
10 Interneuron 1

// In the following line we specify the total number of types of weight
//  change there will be in the network
// NUMBER OF LEARNING RULES
2

// Now we speficify these types, one per line, with the format:
// Change_trigger Gaussian_max_position(_in_seconds) Linear_order_
// LEARNING RULES
0 0.1 0.005 -0.03
1 0.0 0.0 0.0
```

```
13:59:25 - The project has been closed.
13:59:30 - The project EDLUTProject has been opened.
```

Figure 2.4: EDLUT Interface main view - Main view of the EDLUT java-based graphical interface. Note the presence of the different sections: the file editor, the project explorer (top-right corner), the file explorer (down-right corner), and the event log (down-centered).

In the EDLUT Simulation Environment, we can see the next components:

- Project Explorer: It shows the different configuration and result files existing in the current simulation project.
- File Explorer: It shows all the files and directories existing in the current project directory.
- Event Log: It shows the events happening in the environment (such as neuron model building or simulation finished).

- Editor: In the text editor we can see and edit the project files, and see the graphical components (such as result file plots).

The EDLUT Simulation Environment allows working manually with the different files to define the simulation characteristics:

- The neuronal models: The *.nmf file defines the behavior of the cell models, the state variables, the initial values, the equations which determine the evolution and the tables which are precalculated. On the other side, the *.ndf files define the way in which the cell model tables will be loaded, the number of dimensions, the usage of several interpolation methods,etc.

- The network topology: The *.ncf files define the network topology: the cell layers, the synaptic connections between them, and the plasticity mechanisms.

- The initial connection weights: The *.wcf files define the initial weights which will be applied at the beginning of the simulation.

- The input activity: The *.idf files allow the definition of input stimuli. Although, there are many other ways of providing input activity to the network, which will be commented in the next section.

- The output activity: the *.srf files register the output activity of the simulated network. In the same way as input activity, other connectivity mechanisms have been defined in the next section.

We are developing different ways to easily generate these files and to plot them. But now, in this version, we generate the files manually with the text editor. Although, a first approach to the process of plotting the results is shown in figure 2.5.

In the next paragraphs, the format of the files used in EDLUT is in great detail explained.

Neuron model files define the internal behaviour of each cell. The file extension is *.nmf, and the format of this file is the specified in the listing 2.1:

Listing 2.1: Description of a neuron model file (*.nmf). As shown in the code, the first half of the file represents standard C code which will be compiled to precalculate the description tables. The second half includes the values describing the tables and dimensions to be created.

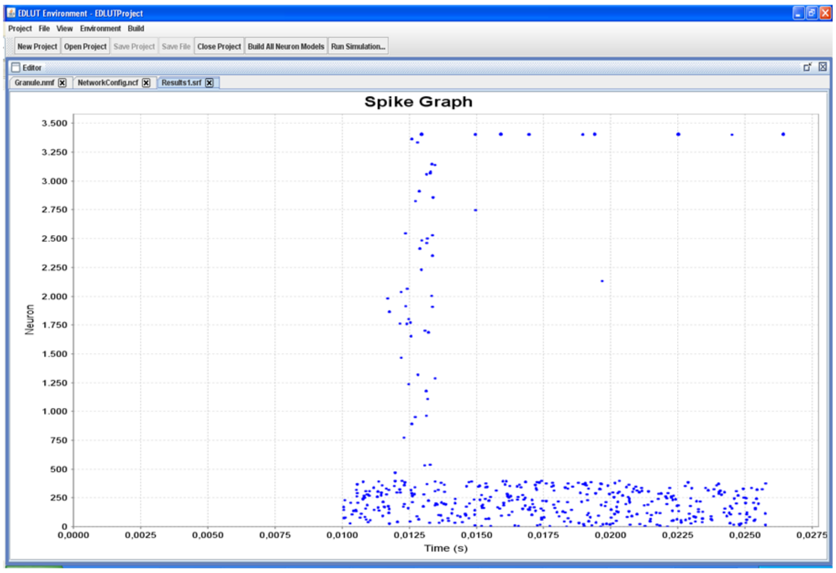


Figure 2.5: EDLUT Interface raster plot - The figure shows the representation of the output spikes after simulating a network. The spike time is shown in the x-axis, while the number of the cell spiking is shown in the y-axis.

```
// Declaration of equations and neural variables
#define NUM_EQS Number_of_Equations
struct consts {
    float Constants_of_Equations;
};

union vars {
    float list[NUM_EQS+1];
    struct {
        float t;
        float Neural_Variables;
    } named;
};

// Definition of functions and selectors
inline float Name_of_the_Function (union vars *v, struct consts *c, float
   h){
    // Calculate and return the neural variable
    return(value);
}

struct teq_sys {
    float (*eq)(union vars *, struct consts *, float);
    int diff;
}

Eq_sys[][NUM_EQS]={
    {{Name_of_Function_Asociated_To_Var1, Function_diferential},...}
};

inline int Name_of_the_Selection_Function(union vars *v, struct consts
   *c){
    return(0);
}

int (*(Eq_sel[]))(union vars *, struct consts *)=
    {Name_of_the_Selection_Function1,...};


// Definition of files and tables to generate
Nr_of_Files_To_Generate

    // Definition of each File
    Name_of_File

    Nr_of_Tables_in_the_File

        // For each table
        Values_of_the_Declared_Constants

        Initialization_of_the_State_Variables

        Nr_of_Equations_To_Simulate

        List_of_Equations

        Nr_of_Table_Dimensions

            // For each table dimension
            Nr_of_the_dimension_variable

            Nr_of_intervals

            Init_Val Final_Val Nr_of_Coordinates Lin|Log Selector
```

In the same way, the neuron description file describes the precompiled tables and defines how to retrieve the data. The file extension is *.ndf, and the format of this file is shown in listing 2.2:

Listing 2.2: Description of a neuron description file (*.ndf). This file includes a description of the tables stored in the precompiled file.

```
Nr_of_state_variables

List_of_the_asociated_equations

State_Variables_Initialization

Nr_of_spike_prediction_table

Nr_of_end_spike_prediction_table

Nr_of_synaptic_state_variables

Nr_of_used_tables

Nr_of_dimensions_of_the_table  Nr_of_variable  Interpolation ...
```

Input description files define the external spikes introduced to the network in the simulation. The file extension is *.idf, and the format of this file is as shown in listing 2.3. In this file, we can add as many spike lines as we need to stimulate the input cells.

Listing 2.3: Description of an input description file (*.idf). This file describes the input spikes which stimulate the neural network. In this way, we can define periodic input patterns starting at the $firstspiketime$ an repeating each *Period*

```
Total_Nr_Of_Spikes

{First_Spike_Time   Nr_Spikes  Period  First_Cell  Nr_Of_Cells}
```

Network configuration files define the network topology in the simulation. The file extension is *.ncf, and the format of this file is as shown in listing 2.4:

Listing 2.4: Description of a network configuration file (*.ncf). This file describes the network topology, including the number and the type of the cells, the synaptic connections and their type (excitatory, inhibitory, electrical coupling,...) and finally, the synaptic plasticity.

```
Nr_of_Different_Cell_Models

// Define the cells of the network
Total_Nr_of_Cells

{Nr_of_Cells Name_of_Cell_Model Monitored}

// Define the weight change rules
Nr_of_Weight_Change_Rules

{Trigger Peak(in_secs) Independent_Change_Coef Dependent_Change_Coef}

// Define the connections of the network
Total_Nr_of_Connections

{1st_Src_Cell_Nr Src_Cell_Nr 1st_Target_Cell Target_Cell_Nr Nr_of_Repeat
    Delay(Sec) Delay_Increment Synapses_Type Max_Conductance(nS)
    Weight_Change_Rule}
```

The next file format is the weight configuration file, which defines the initial weights in the network. The file extension is *.wcf, and it is composed of lines with pairs of number of connections and the initial synaptic weights. Using this format, all the synapses which have been declared in the *.ncf must be specified.

Finally, the simulation results (the spikes which have been fired during the simulation) are stored in files with the extension *.srf. These files are composed of pairs containing the time and the number of cell which fired each spike. This type of file can be visualized with the plot editor and with a text editor.

More information and details about the EDLUT kernel, the source code, the above commented graphical interface and some simple examples which can be used as beginning tutorials can be found and freely downloaded on the official website `http://edlut.googlecode.com`.

2.3.2 Integrating EDLUT with the environment

One of the most important features that encouraged us to create the EDLUT simulator was the lack of tools which allowed the integration of spiking neural networks with more complex external system. Therefore, this is one of the main feature which make EDLUT a suitable tool for control tasks such as the robotic arm control.

Currently, EDLUT can be integrated with the environment using one (or several) of four strategies: running the stand-alone application and communication by means of plain-text files, running the stand-alone application and communication by means of TCP/IP sockets, using the C/C++ API to access the EDLUT kernel, and finally, running EDLUT as a Simulink s-function module.

2.3.2.1 Using plain-text files

The communication by means of input/output plain-text files was the original communication method implemented in EDLUT, and it means the simplest way to use the simulator. After compiling the simulation kernel the user only has to indicate the input and the output file names as two of the command-line arguments, and the simulator loads the input file, processes these input spikes, and finally, stores the output spikes in the selected file. In order to get the easiest usage of EDLUT, this is the communication method implemented in the above commented EDLUT graphical user interface.

The main advantage of this integration method is the simplicity. An external tool which integrates the EDLUT functionality in this way only should write the neural network input activity following the above commented *.idf format, run the EDLUT simulator as a single process and finally retrieve the EDLUT output activity by means of reading the *.srf file format. Thus, this method specially fits with simulation problems where there is no feedback between systems and the input activity is known in advance or debugging tasks. On the other hand, this model is not recommended for online simulations where the performance is a requirement due to the lag produced by the reading/writing in the files.

2.3.2.2 Using TCP-IP sockets

The communication by means of TCP-IP sockets emerged with the idea of solving the necessity of interconnecting the EDLUT simulator (running a cerebellar model for example) with an external system with real-time restriction (for example a robotic arm). Within this system architecture, the communication using plain-text files is not suitable, and a connection through sockets matches much better. Thus, the main computer shall run the cerebellum model and send/receive the outputs/inputs by TCP-IP sockets to other computers or systems which convert the signals to robot understandable signals/commands.

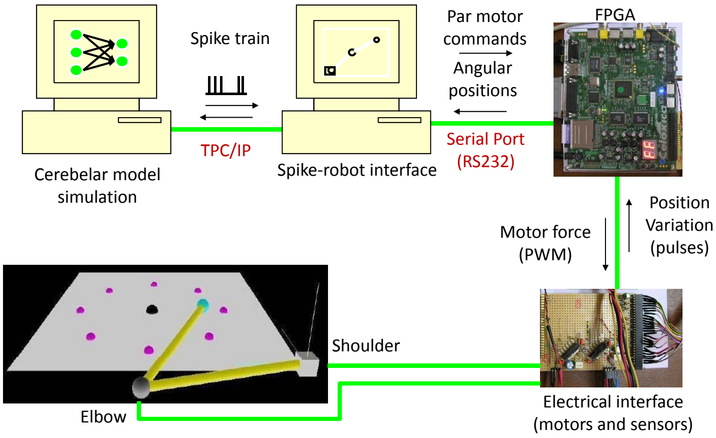


Figure 2.6: EDLUT interfacing with a robotic arm - The TCP-IP socket interface is specially suitable for communication with external feedback systems, such as a robotic arm. Note the EDLUT kernel could run in a computer, and the robot interface could be placed in a different one due to the fast LAN communication.

An example of EDLUT integrating with external robotic systems can be observed in figure 2.6. This home-made robotic arm was completely built in the Department of Computer Architecture and Technology at the University of Granada and was used in target-reaching experiments [41].

An important advantage of this system is the possibility of parallelism by using different processors/computers/cores to simulate parts of a large network model. However, this model should minimize the communication between several processes by distributing correctly the network along the processing unities.

2.3.2.3 Using the C/C++ API

The most efficient way to integrate two C++ systems is by linking their source-code libraries. Following this simple idea, we decided to release the source code of EDLUT and build an extensive documentation of the source code. In this way, the C++ users of EDLUT can obtain full access to the whole set of classes which implement EDLUT. Even though EDLUT implements 69 classes, most of them can be abstracted by using the high-level abstracted types, as shown in listing 2.5.

Listing 2.5: Example of the C++ API usage. This example shows the usage of the C++ API to simulate a small-sized network stored in the file 'NetUPMC.dat' and the synaptic weights stored in 'weightsInit.dat'. Using this API, the program generates the input activity to the network in each simulation step and gets the output activity after simulating.

```
/*****************************************************
 *                    StepByStep.cpp                 *
 *                 -------------------               *
 * copyright            : (C) 2010 by Jesus Garrido  *
 * email                : jgarrido@atc.ugr.es        *
 *****************************************************/

#include <cstdlib>
#include <cstdio>
#include <cstring>
#include <time.h>
#include <math.h>

#include <iostream>

#include ”../include/simulation/Simulation.h”
#include ”../include/communication/ArrayInputSpikeDriver.h”
#include ”../include/communication/ArrayOutputSpikeDriver.h”

#include ”../include/spike/EDLUTFileException.h”
#include ”../include/spike/EDLUTException.h”

using namespace std;

/*!
 *
 *
 * \note This software is only an example of how to run step-by-step
     simulations.
 *
 * This simulation runs with a 10-ms step and generates 10 random spikes
     each step
 * in input cells.
 */
int main(int ac, char *av[]) {

        int result = 0;
        clock_t startt, endt;
```

```
const char * NetworkFile = "NetUPMC.dat";
const char * WeightsFile = "weightsInit.dat";
double SimulationTime = 1;
double StepTime = 0.10;

const int NumberInputCells = 42;

// Create the new simulation object (and load the network and
    weight definition file)
Simulation * Simul = new Simulation(NetworkFile, WeightsFile,
    SimulationTime);

// Create a new input object to add input spikes
ArrayInputSpikeDriver * InputDriver = new ArrayInputSpikeDriver();
Simul->AddInputSpikeDriver(InputDriver);

// Create a new output object to get output spikes
ArrayOutputSpikeDriver * OutputDriver = new
    ArrayOutputSpikeDriver();
Simul->AddOutputSpikeDriver(OutputDriver);

// Get the external initial inputs (none in this simulation)
Simul->InitSimulation();


startt = clock();          // Simulate network and catch errors

double InputSpikeTimes [NumberInputCells];
long int InputSpikeCells [NumberInputCells];

double * OutputSpikeTimes;
long int * OutputSpikeCells;

// Simulate step by step.
for (double CurrentTime = 0; CurrentTime<SimulationTime;
    CurrentTime+=StepTime){

        cout << "Simulation at time " << CurrentTime << endl;

        // Generate input spikes (we generate one spike at random
            time for each input cell)
        for (int i=0; i<NumberInputCells; ++i){
                InputSpikeTimes[i] = rand() * StepTime /
                RAND_MAX+CurrentTime;
            InputSpikeCells[i] = i;
    }

    // Load inputs
    InputDriver->LoadInputs(Simul->GetQueue(),
       Simul->GetNetwork(), 10, InputSpikeTimes,
       InputSpikeCells);

    // Simulate until CurrentTime+StepTime
    Simul->RunSimulationSlot(CurrentTime + StepTime);

    // Get outputs and print them
    int OutputNumber =
       OutputDriver->GetBufferedSpikes(OutputSpikeTimes,
       OutputSpikeCells);

    if (OutputNumber>0){
        for (int i=0; i< OutputNumber; ++i){
            cout << ”Output spike at time ” <<
               OutputSpikeTimes[i] << ” from cell ”
               << OutputSpikeCells[i] << endl;
        }

        delete [] OutputSpikeTimes;
        delete [] OutputSpikeCells;
    }
}

endt = clock();

cout << ”Oky doky” << endl;

cout << ”Elapsed time: ” << (endt-startt) / (float)CLOCKS_PER_SEC
   << ” sec” << endl;
cout << ”Number of updates: ” << Simul->GetSimulationUpdates() <<
   endl;
cout << ”Mean number of spikes in heap: ” <<
   Simul->GetHeapAcumSize() /
   (float)Simul->GetSimulationUpdates() << endl;
cout << ”Updates per second: ” << Simul->GetSimulationUpdates() /
   ((endt-startt) / (float)CLOCKS_PER_SEC) << endl;
cout << ”Total spikes handled: ” << Simul->GetTotalSpikeCounter()
   << endl;
```

```
        delete Simul;
        delete InputDriver;
        delete OutputDriver;

        return result;
}
```

As this work has been carried out in the framework of two European Projects (SENSOPAC and REALNET), we had the opportunity of integrating EDLUT with the following models:

- Large-scale analog model (LSAM) is a bottom-up executable mathematical model of the cerebellum, developed at the Swedish Institute of Computer Science in Stockholm. LSAM is primarily a biomimetic model, based on detailed cerebellar measurements and neuronal connectivity data collected at Lund University, Sweden, and the University of Pavia, Italy. However, as we uncover algorithms evolved by nature, we find that they may also be useful in human-designed, artificial systems. In order to provide LSAM with the ability of spike processing, it has been integrated by means of a restricted set of only 4 standard-C functions, which are enough to simulate spiking neural networks and supply the input/output data.

- EDLUT has been used to simulate a Cuneate Nucleus developed at the Pierre and Marie Curie University in Paris (France) [26]. This model has been integrated with a biologically inspired robotic arm in the framework of a Braille reading experiment in real-time as the final demo in the European Project SENSOPAC.

All these examples show that EDLUT can be easily and efficiently integrated using the C++ API, and this integration is suitable to the development of high performance systems.

2.3.2.4 Using the Simulink S-function module

Simulink® is an environment for multidomain simulation and Model-Based Design for dynamic and embedded systems. It provides an interactive graphical environment and a customizable set of block libraries that allows designing, simulating, implementing and testing a variety of time-varying systems, including communications, controls, signal processing, video processing, and image processing. Even though is not freely available, Simulink represents one of the most common frameworks for commercial control systems. This is the reason that motivated the implementation of this interface towards linking EDLUT simulator with traditional control research groups.

Figure 2.7 shows the usage of the EDLUT for Simulink version in a further control system where a planner and a crude inverse model has been included in the simulation loop. It has been compiled as an C++ S-function after implementing all the needed modules to interface both systems. The static parameters that the final module receives are the following (in this order):

1. The name of the network configuration file (*.ncf).

2. The name of the weight configuration file (*.wcf).

3. The name of the output log file. This file stores all the events which are processed during the simulation and mainly has a debugging function.

4. The time-driven model step time. This is the time-step of the time-driven cell models which may be included in the network (if any).

5. This parameter is an input map. This array matches each module input line with an input cell. Thus, the number of elements in this array will match with the number of input lines in the signal.

6. This parameter is the output map. In the same way with the previous item, this array matches each module output line with an output cell. Thus, the number of elements in this array will match the number of output lines in the signal.

7. The name of the file where the synaptic weights will be stored after the learning process. This file will be useful specially when studying neural networks which include synaptic plasticity.

8. This parameter is the time step (in seconds) between successive weight storages.

In the module initialization, the system checks all the parameters and sets the number of input and output lines. Both, the input and the output array, are boolean values representing the spikes which stimulate the input cells of the networks or the output cells which fire spikes. The communication step time will be determined by the inherited time-step (in this example it was fixed to 2ms). Thus, a true value in an

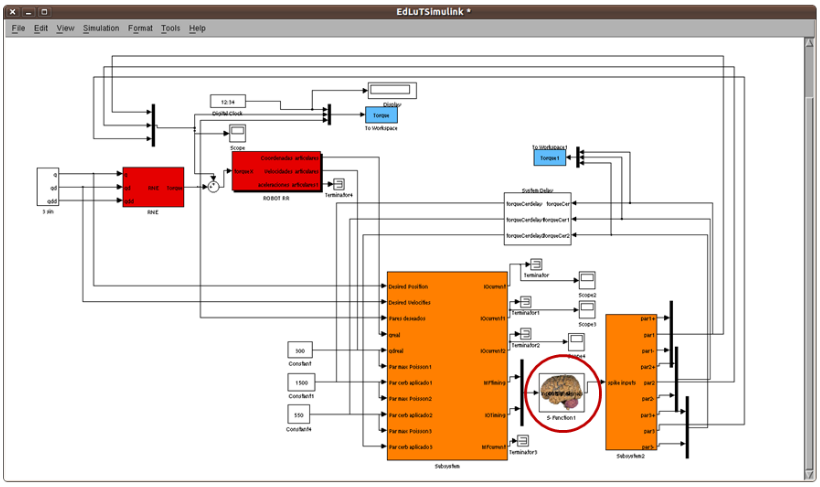


Figure 2.7: EDLUT interfacing with a robotic arm simulator using Simulink - This model simulates the whole loop including a red surrounded cerebellar model (this box includes EDLUT modelling an artificial simplified cerebellum defined as explained in chapter 4) and a robot arm simulator. EDLUT receives input spikes by means of boolean input signals and sends output spikes in the same way.

input line in a time-step will represent an input spike reaching the target cell at the beginning of the interval.

This EDLUT interface is specially useful because it avoids the necessity of using other communication mechanisms such as the TCP-IP sockets and therefore, the performance will be highly increased.

2.4 Building more neuronal models

The availability of preimplemented and validated cell models is an important feature in every neural simulator and allows a faster design of network models for beginner users. EDLUT is distributed with event-driven versions of Leaky Integrate and Fire and Hodgkin & Huxley models (see [227]). However, the development of new event-driven models is not always a trivial task and the model should be widely validated. This has motivated EDLUT to be able to integrate time-driven models in network simulations. These models are easier and faster to implement using inherited C++ classes from the *TimeDrivenNeuronModel* base class.

Therefore, Leaky Integrate and Fire (LIF) models and stochastic Spike Response Models (SRM) have been implemented using both alternatives (time-driven and event-driven schemes) and have been tested in order to validate their behavior and performance (in terms of computing speed).

2.4.1 Time-driven models

In order to achieve a simulation including both event-driven and time-driven methods, the previously-developed Event-Driven LookUp-Table simulator (EDLUT [227]) has been upgraded. The previous architecture was mainly composed of a simulation engine capable of processing spikes. Now, it handles a spike as a concrete type of event, so that, different classes of events can be processed homogeneously. Thus, the simulation loop could be seen as a simple event handler as shown in algorithm 4.

Algorithm 4 EDLUT Simulation Engine. The simulation algorithm mainly follows an event-driven method. However, the homogeneous event treatment allows the implementation of a time-driven method without modifying this loop. Many new features can be added by means of adding a new class of events which implements the ProcessEvent method.

```
program NeuralSimulator ()
    InitializeSimulation(...);
    repeat
       Event := GetTheNextSimulationEvent();
       SimulationTime := Event.Time;
       Event.ProcessEvent();
       Event.RemoveFromHeap();
    until Heap.IsEmpty();
end.
```

The time-driven events interface with the time-driven simulation loop. Each time that one of this kind of event is caught, all the time-driven neurons are processed and their state is updated. The processing algorithm is mainly composed of 3 different parts:

1. The neural state of each time-driven neuron is updated. This evolution is computed according to the neural model associated to that neuron and the elapsed time from the last state update.

2. If one of those updated neurons fires an spike, this is inserted on the top of the event heap as an internal spike and it will be processed according to its time stamp.

3. Finally, a new time-driven update event is inserted in the event heap. This upcoming event will be handled after the shortest time step of all neurons has been processed (in case of simulating with several time steps).

Figure 2.8 shows the two different processing pathways. Note the difference with figures 2.1 and 2.2. Based on the event-driven engine, we have implemented a new kind of event which on being caught, the simulator will update the state variables of each time-driven cell.

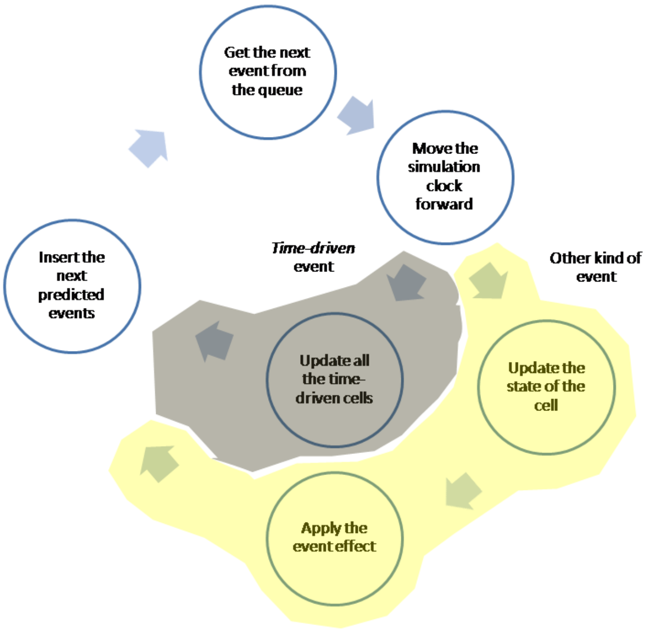

Get the next event from the queue
Move the simulation clock forward
Time-driven event
Other kind of event
Insert the next predicted events
Update all the time-driven cells
Update the state of the cell
Apply the event effect

Figure 2.8: Hybrid time-driven and event-driven simulation scheme - This figure shows the implemented algorithm capable of simultaneously processing event-driven and time-driven cell models.

Also the input spike type has been added to the original EDLUT data structures. Following the same methodology proposed in EDLUT, external inputs reach the network through *stub* neurons. Thus, an incomming spike does not modify the *stub* neuron state, it only generates the corresponding input spikes (propagated spike events [227]) for every neuron which is connected to the output of this neuron. In this way, input spike processing is really simple, it only inserts the first (the earliest) output spikes produced by the input neuron.

Another kind of implemented event is the internal spike. These are directly fired by the dynamics inherent to the neurons (as a response to input conductance or as a consequence of the autonomous activity of the neurons). In these events, a different processing is carried out depending on whether the neuron model is time-driven or event-driven. The last one updates the neural state and after that, checks if other spike is scheduled for later, and in that case it inserts it in the simulation heap. However, this state-update stage is not necessary when simulating event-driven neural models due to the periodical evolution of the state variables. Finally, the first propagated spike is inserted in the simulation heap.

The last implemented stage of the spike processing is the propagated-spike event which corresponds to one spike being transmitted between neurons. Thus, the input spike will be processed and only if the neural model is event-driven the state will be updated. After that, the next propagated spike (generated by the same source neuron) will be inserted in the simulation heap.

Finally, some other different types of events have been implemented in order to handle different situations which could happen during the simulation (such as communication with external systems following a simulation loop, saving the synaptic weights periodically to explore learning processes, etc.).

In order to test the utility of hybrid simulation systems, two different alternatives of several neuron models (Leaky Integrate and Fire -LIF- and Spike Response Model -SRM-) have been implemented and subsequently tested: time-driven and event-driven based in look-up tables.

2.4.1.1 Leaky Integrate and Fire time-driven model

Using the LIF models, an abstraction of the granular layer of the cerebellum has been built. This network is composed of partially overlapped neurons including as a whole

4877 neurons of two different types (see Figure 2.9).

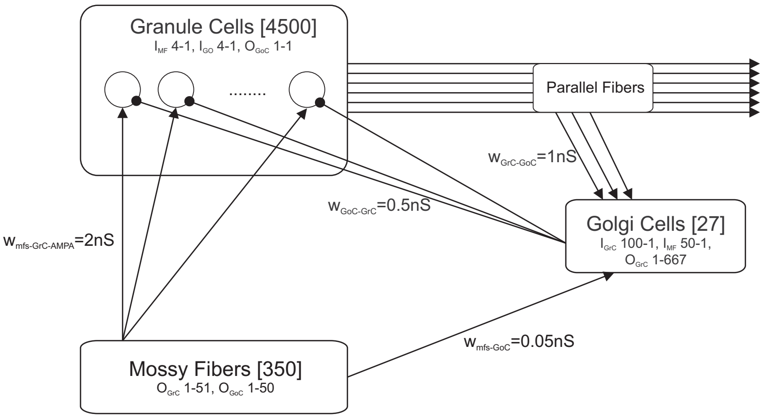


Figure 2.9: Simulated architecture of the granular layer - The whole network is composed of 350 mossy fibers (MFs), 4500 granule cells (GrCs) and 27 Golgi cells (GoCs). The convergence/divergence ratios are represented into each layer (e.g. each GrC receives activity from 4 MFs and 4 GoCs, and each GoC receives activity from 100 GrCs and 50 MFs).

Both neuronal types (Granule Cells -GrCs- and Golgi Cells -GoCs-) have been implemented using a Leaky Integrate-and-Fire model [100]. However, different constants have been defined in order to get realistic dynamics according to each cell type. Firstly, this model was strictly implemented following a time-driven scheme. Afterwards, an adaptation process was carried out in order to develop a more efficient event-driven model based in look-up tables.

The state of the neuron is characterised by membrane potential (V_{m-c}) which is expressed by equation (2.1).

$$\begin{aligned} C_m \frac{dV_{m-c}}{dt} &= g_{AMPA}(t)(E_{AMPA} - V_{m-c}) + g_{GABA}(t)(E_{GABA} - V_{m-c}) + \\ &\quad G_{rest}(E_{rest} - V_{m-c}) \end{aligned} \tag{2.1}$$

where C_m denotes the membrane capacitance, E_{AMPA} and E_{GABA} represent the reversal potential of each synaptic conductance and E_{rest} is the resting potential. g_{AMPA} and g_{GABA} input driven conductances that integrate all the contributions received through individual synapses and are defined as decaying exponential functions. The parameters of the neuronal model and a more detailed description can be found in [100] and in the third chapter of this document (see granular layer cell models).

These neuron models account for synaptic conductance changes rather than simply for fixed current flows, providing an improved description over common I&F models. The version of these neural models implemented for EDLUT simulator can be found and downloaded at the EDLUT project official site (see `http://edlut.googlecode.com`).

LIF models can be directly implemented following event-driven schemes based on lookup tables with only regarding the correctly adjusting the table dimensions. However, once both time-driven and event-driven schemes were implemented, we used the first of them to validate and test the performance of the second one.

2.4.1.2 Spike Response time-driven model

The Spike Response Model (SRM) is well known in the field of neuroscience [100]. The SRM formalism provides a linear probabilistic neuronal model, as opposed to the more classical integrate-and-fire model which is non-linear and deterministic. Compared to the Hodgkin-Huxley formalism, the SRM permits a higher transparency and controllability of all free parameters (e.g. synaptic integration time constant, amplitude and shape of excitatory post-synaptic potentials, and so on). This cell model has been used in [26] in order to implement a cuneate nucleus model.

Let V denotes the membrane potential of a SRM unit. If an input spike arrives at time t, the membrane potential undergoes a depolarization $\Delta V(t)$ whose time-course is stereotyped and taken as:

$$\Delta V(t) \propto \sqrt{t} \cdot e^{\frac{-t}{\tau}} \tag{2.2}$$

where the free parameter τ determines the decay time constant of the EPSP (excitatory post-synaptic potential) of the neuron. We took $\tau = 7ms$ in our simulations.

If several afferent spikes excite the neuron within a short time window, then the EPSPs add linearly:

$$V(t) = V_r + \sum_{i,j} w_i \Delta V(t - \hat{t}_i^j) \tag{2.3}$$

where i denotes the pre-synaptic neurons, j indexes the spikes emitted by a pre-synaptic neuron i at times t_i^j , $V_r = -65mV$ is the resting potential, and w_i indicates the synaptic weight of the projection from the pre-synaptic unit i, and it is defined as:

$$w_i = W \cdot w_i^{0,1} \tag{2.4}$$

where the factor W is the upper bound of the synaptic efficacy, and $w_i^{0,1}$ is constraint within the range $[0, 1]$. W was set to $20mV$ in our simulations.

At each time step, a function $g(t)$ that can be thought of as the instantaneous firing rate of the cell and is defined according to:

$$g(t) = r_0 \cdot \log\left(1 + e^{\left(\frac{V(t)-V_0}{V_f}\right)}\right) \tag{2.5}$$

where the constants $r_0 = 1Hz$, $V_0 = -60mV$, $V_f = 0.5mV$ are the spontaneous firing rate, the probabilistic threshold potential, and a gain factor, respectively.

The refractoriness property of the cell is modelled as a function $A(t)$:

$$A(t) = \frac{(t - \hat{t} - \tau_{abs})^2}{\tau_{rel}^2 + (t - \hat{t} - \tau_{abs})^2} \cdot \theta(t - \hat{t} - \tau_{abs}) \tag{2.6}$$

where τ_{abs} and τ_{rel} are the absolute and relative refractory periods, respectively, $\hat{t}$ is the time of the last spike emitted, and $\theta(t)$ is the Heaviside function. We used $\tau_{abs} = 6ms$ and $\tau_{rel} = 1ms$. The functions $g(t)$ and $A(t)$ permit the probability of firing $p(t)$ to be calculated:

$$p(t) = 1 - e^{-g(t)\cdot A(t)} \tag{2.7}$$

The time-driven implementation of this cell model was quite simple. A new C++ class redefined the neuron model characteristics methods, such as $UpdateState(...)$

(where the state variables are recalculated) and *ProcessInputSpike*(...) (where the input conductances are incremented after a spike reaches the cell). For more information about the C++ implementation, the doxygen documentation of the *NeuronModel* and *TimeDrivenNeuronModel* classes (available at `http://edlut.googlecode.com` is recommended.

2.4.2 Stochastic SRM table-based models

The accuracy of the time-driven implementation of the stochastic SRM table-based model can be increased by shorting the simulation time-step. However, the shorter the simulation step is, the more time the simulation will take. Therefore, an alternative version based on look-up tables could accelerate the simulation in order to run models with thousands of cells.

However, the traditional cell models implemented in EDLUT with the table-based model generator did not had the ability to stochasticity in the precompiled tables. Thus, the EDLUT simulation kernel had to be redesigned to include this feature. In addition to this, the function in equation 2.2 could not be included in table calculation without having to keep a register of the previous activity in the cell. Thus, this function was approximated by the equation 2.8:

$$\Delta V(t) \propto a \cdot t \cdot e^{\frac{-t \cdot b}{\tau}} \tag{2.8}$$

where a and b are both constants to fit the function to the originally proposed in equation 2.2. Figure 2.10 shows the difference between the original function (black line), the implemented alternative (blue line) and another proposed function (red line). Even though the last function was suitable as well, the linear function was finally selected due to the simpler implementation and calculation.

Using the equation 2.8, the membrane potential can be calculated in an arbitrary time by only expanding the next equation:

$$\begin{aligned} V(t+\Delta t) &= \sum_{i,j} \left(\left(t + \Delta t - t_i^j \right) \cdot e^{-(t+\Delta t - t_i^j)} \right) = \\ &= e^{-\Delta t} \cdot \sum_{i,j} \left(\left(t - t_i^j \right) \cdot e^{-(t-t_i^j)} + \Delta t \cdot e^{-(t-t_i^j)} \right) \end{aligned} \tag{2.9}$$

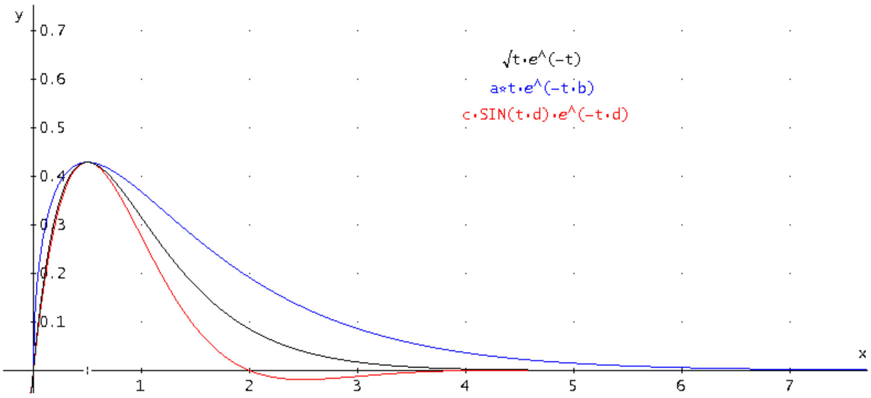


Figure 2.10: Several alternatives for the SRM potential function - The originally proposed function (black line) could not be implemented following an event-driven scheme without activity buffers. The linearly proportional function (blue line) was finally implemented, and the sinusoidal function (red line) was also considered. Both functions are quite similar to the original one.

This function can be rewritten as following:

$$V(t + \Delta t) = e^{-\Delta t} \cdot (V(t) + \Delta t \cdot A(t)) = f(V(t), \Delta t, A(t)) \qquad (2.10)$$

where $A(t)$ is an auxiliary function which can be calculated from the previous value an the time-step Δt, i.e.:

$$A(t) = \sum_{i,j} e^{-(t-t_i^j)} A(t + \Delta t) = \sum_{i,j} e^{-(t+\Delta t-t_i^j)} = e^{-\Delta t} \cdot A(t) = f(A(t), \Delta t) \qquad (2.11)$$

Therefore, the membrane potential can be calculated in any arbitrary time without saving the pre-synaptic activity just by means of two state variables $A(t)$ and $V(t)$. In the next stage, the look-up tables have to be designed as following:

1. Membrane potential table: This table precalculates the state variable $V(t)$ as a function depending on the variables $V(t)$, Δt, and $A(t)$.

2. Auxiliary variable table: This table implements the behavior of the $A(t)$ state variable as a function of $A(t)$, and t.

3. The next firing firing spike time table: This table calculates the time when the next spike will be fired by means of calculating the membrane potential (eq. 2.9), the refractoriness (eq. 2.6), the firing rate (eq. 2.5) and the firing probability (eq. 2.7) for each time-step (Δt). Using this firing probability and an additional seed dimension which initializes the rand number generator, the table calculation loop will check the next spike time.

This table design requires the update of the random generator (the seed state variable) previously to each estimation of the next fired spike. This update can be done by using the current system clock. Thus, each time the system need to calculate the next firing time the seed table dimension will be accessed with a different value, including in this time the random component of the behavior.

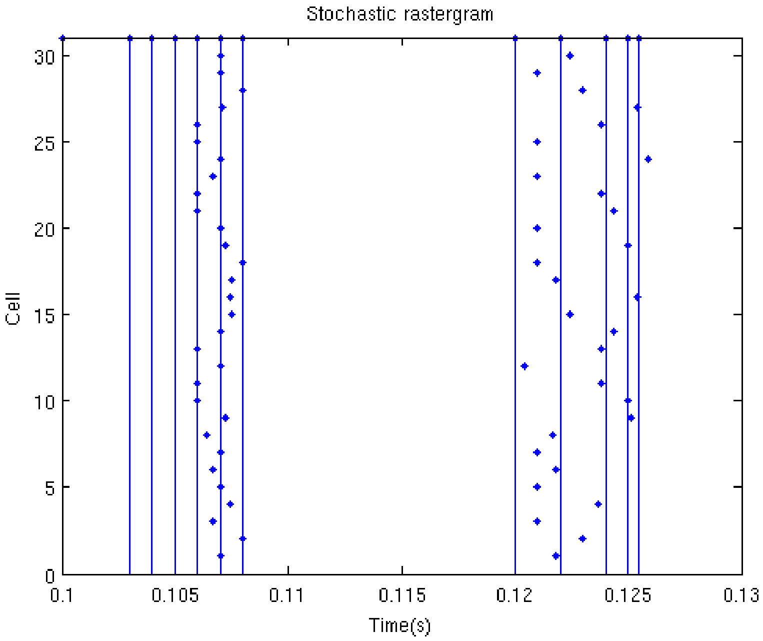


Figure 2.11: Event-driven table-based simulation of a set of stochastic cells - Spike raster of 30 SRM stochastic cells. Input spikes (vertical lines) stimulated all the target cells using the same synaptic weigth ($w_i^{0,1} = 0.1$ in eq. 2.4. However, the post-synaptic spikes (blue dots in the plot) happened at different times due to the stochasticity introduced in the tables.

Figure 2.11 shows the result of simulating 30 stochastic SRM cells connected to the same input cell. Even though all of them are input responsive, and the output spike rate is related to the input firing rate, the output activity presents delays in their responsiveness depending on the randomness in the simulation. However, other model features, such as the 6-ms absolute refractory period and the 1-ms relative refractory period are kept intact.

In addition to this, the stochastic cell model has been used in order to implement a reading-Braille robotic hand which implements the cuneate nucleus layer to separate the input patterns in real time. This system was shown in the final demo of the European Project SENSOPAC and is currently under publication.

2.5 Results

This section briefly comments the results (in terms of accuracy and efficiency) of the advanced features which have been implemented in EDLUT. However, the following chapters of this work can be considered EDLUT results as well, because this tool has been used to implement all the showed models.

Finally, we will conclude evaluating the consequences of the releasing of EDLUT as an open source tool and its availability on the source code google repository `http://edlut.googlecode.com`.

2.5.1 Hybrid simulation of time-driven and event-driven models

A hybrid network including a time-driven model (the GoC model) and an event-driven model (the GrC model) has been built. This hybrid simulation method includes the advantages of the well-controlled accuracy of a time-driven model in GoCs and a fast simulation of GrCs with the event-driven model. Moreover, these advantages are more acute if we take into account the number of GrCs (4500 in our model and more than 50 billions in the human cerebellum) and the small number of GoCs (only 27 in our experiment and also a much smaller number than GrCs in the human brain) which diverge into a wider area in GrCs. These characteristics make the cerebellum more suitable to be simulated using this method.

In order to test the accuracy of the whole system, a complete-time-driven granular layer has been also simulated using different time steps and has been compared with a hybrid simulation scheme (event-driven model in GrCs and time-driven model in GoCs). The mean-absolute error (MAE) of the GrC activity histogram (taking 1-ms bins) has been used as accuracy measure, following a population code strategy. The reference model was a complete time-driven model implementing 4th order Runge-Kutta methods with 0.1 ms time step. The network was stimulated using random inputs at different frequencies, and the MAE has been calculated after simulating the network with each input frequency.

Figure 2.12 shows the accuracy of the tested schemes. As expected, the error in relation to the 0.1ms-step time-driven model increased as we simulated higher input frequencies. However, this enhancement seems to be more marked in a hybrid scheme (simple line), due to the progressive accumulation of inputs arriving at very near times. However, a hybrid system presented similar precision to a time-driven scheme with 1 ms time step (triangle line). Thus, the time-driven methods with the shortest time steps obtained the better accuracy, while the hybrid model presented reasonably good results (similar to 1ms-step time-driven method, and better than 10ms-step time-driven method -cross line-).

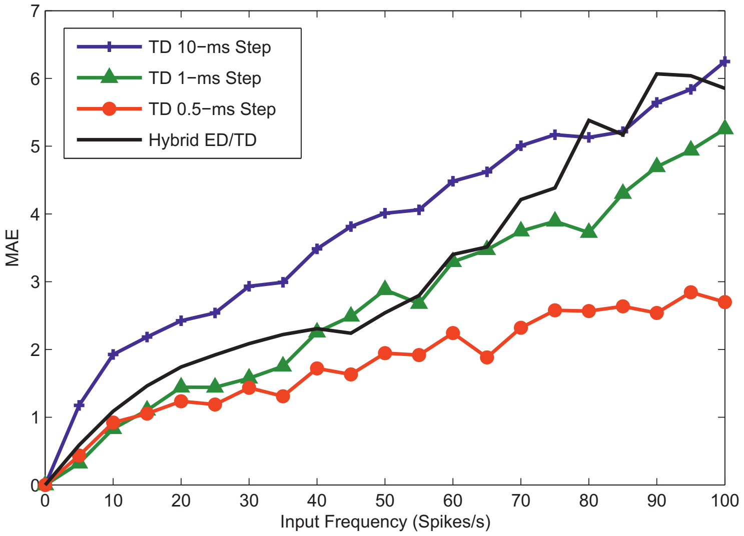


Figure 2.12: Obtained error (MAE) after simulating 0.5 s of real activity - Different simulation methods are compared with the reference model (a whole time-driven network using 4th-order Runge-Kutta methods with 0.1 ms step).

However, an event-driven simulation method based on look-up tables supposed an improvement on the simulation speed specially when processing sparse activity, as it is believed to occur in some layers of the cerebellum (e.g. the parallel fibers) [49]. The performance of the system was measured as the rate of processed spikes per seconds. Using this estimator, we obtained a fair comparison between methods with different levels of activity (due to the accumulation of the error along the simulation).

Figure 2.13 shows the performance of all the studied systems. The hybrid scheme (simple line), using both time-driven (0.1ms time step) and event-driven models, kept a nearly constant rate of processed spikes per second due to the intrinsic characteristics of the simulation method. On the contrary, full time-driven networks (square, circle and triangle line) improved their performances as the input frequency increased (more activity). However, this improvement was not completely lineal (as expected in a time-driven model) due to the implementation of the method (using an event-driven method as the base of time-driven models). This aspect became more evident when we simulated the time-driven network with 10ms time step (cross line). In this case, the performance is extremely close to the hybrid network because the computational load of the simulation is mainly caused by the event processing (using long time steps remarkably reduces the number of evaluations of the neural-model differential equations).

The enhancement of the performance when using hybrid networks can be noticed if we compare the rate of processed spikes between the hybrid system (simple line) and the 0.1ms-time-step time-driven model (square line). In both models, GoCs are implemented using a 0.1ms time-driven model. However, the difference lies in the model of GrCs. In this case, using an event-driven model (hybrid system) the number of spikes per second approximately doubled the pure time-driven model (this rate of processed spikes is specially higher with low input activity rates -between 0 and 20Hz-, as expected in event-driven simulation schemes). Thus, using event-driven models in layers with a huge quantity of neurons markedly accelerates the network simulation.

2.5.2 EDLUT released as open-source

EDLUT simulator was released under GPLv3 license on April 2009. This license establishes several freedoms to the potential users, such as using the software for any purpose, changing the software to suit their needs, and freely sharing the software and the changes the developers may make.

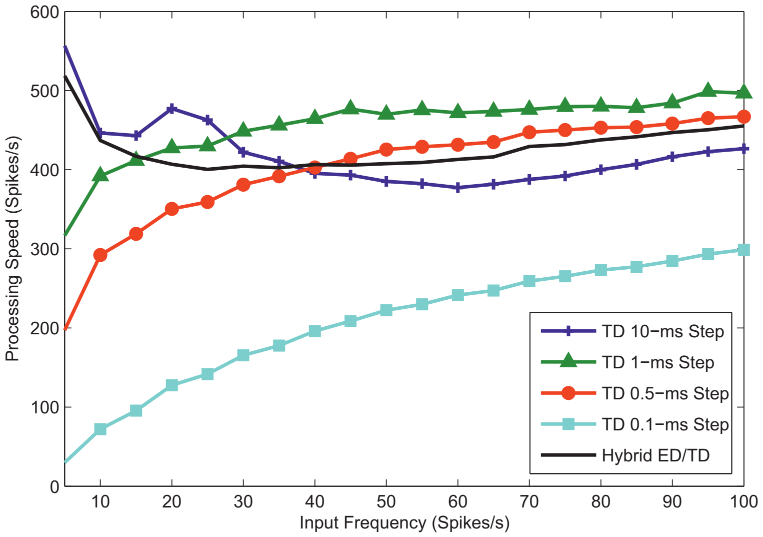


Figure 2.13: Simulation speed measured as the rate of processed spikes per second - While hybrid scheme achieves a nearly-constant processing rate due to the intrinsic characteristics of event-driven schemes, time-driven schemes increased their performances as the input frequency increased.

The simulator is compatible with the best known operating systems. Concretely, EDLUT for windows can be compiled using Visual Studio 2008 or using the Cygwin tools just by typing *make* in the root directory. Linux and MacOS versions can be compiled from the source code with the *makefile* tool as well. In addition to this, other alternatives can be built using different parameters in the *makefile* tool, such as building the doxygen documentation in several formats (html, latex, pdf,...), building the EDLUT kernel as a library, or the Simulink version. More information about the EDLUT compiling process can be found on the README file supplied with the EDLUT download.

Since EDLUT was released, the website has received more than 11000 visits, and more than 25000 pages have been generated, from all over the world (specially from Spain, Brazil, and United States). The visiting statistics reveals that EDLUT has been favourably received, and specially within the open-source community.

At the EDLUT website, users can find the last versions of the EDLUT kernel, the table generator, and some examples of usage (including the above commented neuronal models, and network models). In addition to this, some other documents, presentations, and all the modifications on the source code are available at the source code repository of the website.

2.6 Discussion

In this work, the base of an event-driven simulator based on look-up tables has been widely upgraded to accomplish the main requirements of a real-time neural network simulator capable of interfacing with a huge quantity of external systems using different ways of communication: the software API, the Simulink module, the plain-text files or the TCP/IP socket interface. A Java-based graphical user interface has been developed as well, which facilitates the usage of this simulation tools for beginner researchers. All these developments have been widely documented and are available on the project repository.

In addition to this, the event-driven simulation environment has been widely improved to make it capable of natively simulating time-driven methods (alone or in conjunction with event-driven methods). This allowed the simulation of hybrid networks where some layers were using time-driven-based models while others were running event-driven-based models.

A correct decision on what simulation method should implement each neuron model could lead to an impressive enhancement of the simulation performance. This decision should take into account some of above commented factors such as the activity rates in which the neural model will be working, the number of neurons to simulate, the influence of each neuron on the rest of the network activity, the possibility of implementing an event-driven alternative, the complexity of the neuronal dynamics, etc...

Event-driven methods (or more generally hybrid networks) showed good performance and accuracy when working with low rates of activity. In the opposite, time-driven model would be preferred when the characteristics of the experiment produce high neural activity. However, as long as different layers in a biological system could present different levels of activity, a hybrid system including both time-driven and event-driven methods can be convenient.

The availability of a hybrid simulation environment also allows the validation of new developed neural models for the event-driven method including this model in the network and comparing its behaviour with the previously implemented time-driven model. In this way, faster and more precise table-based neural models could be designed and subsequently tested in a realistic simulation.

As future work, the time-driven implementation will include a variable and independent time step for each neuron, speeding up and enhancing the precision of this method. Moreover, different approaches will be evaluated in order to parallelize the simulation of large scale neural networks on clusters and supercomputers. Finally, the integration with generic tools to describe the model in different abstraction levels (cell, neural network, system,...) such as PyNN or NeuroML propose could make easier the learning curve to potential EDLUT users and would allow the simulation models being reusable from well-known tools such as NEURON or GENESIS.

3

The impact of synaptic weights on spike timing in the cerebellar input layer

3.1 Abstract

Regulation of spike timing is particularly relevant for the cerebellum, which plays a central role in processing temporal sequences. In order to understand the potential impact of plasticity at different sites in the network, we have used a real-time event-driven simulator (EDLUT) to carry out massive simulations on the whole synaptic weight parameter space. In response to repeated burst inputs, the granular-layer network generated rebounds and oscillations in the β/γ-frequency band and filtered unsynchronized trains with millisecond precision. We found that synaptic weights at multiple synapses could play a crucial role to enhance coincidence detection (which allows the reduction of non-synchronous signals) and time slicing (which determines specific time windows for signal transmission). Interestingly, coincidence detection was especially sensitive to weights in the feed-forward inhibitory pathway (mossy fiber→Golgi cell→granule cell), whereas time slicing was especially sensitive to weights in the feed-back dis-inhibitory loop (granule cell→stellate cell→Golgi cell→granule cell). These results predict that the granular layer operates as a complex adaptable filter which can be controlled by weight changes at multiple synaptic sites.

3.2 Introduction

The computations elaborated by central neuronal networks depend on multiple interacting factors [12, 205]. These mainly comprise: (i) the properties of different neuronal types, (ii) how neurons are connected, (iii) and how their synaptic weights are organized. Although invaluable knowledge can be gained from experimental results, what usually remains elusive are the dynamics emerging from multiple interactions between network elements and even more, the impact of distributed synaptic plasticity in the network [12]. Computational modeling can be used to explore this complex parameter space.

The granular layer processes incoming signals that have to be relayed to Purkinje cells for further pattern recognition [79, 192]. Recently, investigations into granular layer processing have been promoted by several relevant discoveries. First, the properties of granule cells (GrC) and Golgi cells (GoC) have been clarified to a considerable extent [61, 64, 83, 92, 208, 249, 250]. These neurons are able to generate very fast and robust burst discharges in response to inputs. Secondly, circuit connectivity has been extensively investigated, showing the functional importance of two main loops implementing feed-forward inhibition (mossy fiber→GoC→GrC) [150] and feed-back inhibition (GrC→GoC→GrC). Moreover, recent works suggest that the stellate cell (SC) axons arborize within the molecular layer, and their varicosities make GABAergic synaptic contacts with Purkinje cells and seem to contact also other GABAergic neurons, such as Golgi cells [87, 101, 260]. These findings may prove the plausibility of a feed-back dis-inhibition loop (GrCs→SC→GoC→GrC) [19]. Thirdly, synaptic plasticity has emerged as an important property of the network and is probably distributed over numerous synaptic contacts [114, 218].

The granular layer has recently been proposed to perform two operations, threshold detection [148] and spatio-temporal transformations of incoming signals [190, 191, 251]. Threshold detection, under double control of mossy fibers (MF) and Golgi cells, can filter uncorrelated signals (noise) allowing only transmission of well-correlated spikes in groups of neurons (i.e. synchronized spike trains). At the same time, the feed-forward inhibitory loop can generate time windows through which spikes may or may not pass, thereby filtering out-of-time signals [58, 59]. This filter is under the control of circuit inhibition [162, 251] and also probably of multiple forms of long-term synaptic plasticity [237], which can optimize its performance.

By exploiting a real-time spiking network simulator (EDLUT) to perform massive explorations of the parameter space [227], we have investigated the function of the multiple granular layer loops and the impact of synaptic weights. The cerebellar granular layer filtered unsynchronized (temporally scattered) trains, comprising both threshold detection and time slicing, with millisecond precision. This operation could be optimally achieved by adjusting synaptic weights at the various synapses of the three loops of the granular layer. Interestingly, given the specific time-constants of neurons and synapses, the network generated rebounds and oscillations in the β/γ-frequency band. By these means, the granular layer privileged coherent burst transmission over a specific operative band while filtering out weak uncorrelated input activity. These observations suggest that the granular layer operates as a complex adaptable filter [79], which can be controlled by weight changes at multiple synaptic sites.

3.2.1 The state of the art: temporal processing in the cerebellum

The cerebellum is organized in modules including cortical microcomplexes [30, 73, 217, 272]. This structured organization has always helped in the study of the cerebellum, and has made this one of the best known systems in the brain. The understanding of circuit mechanisms can be conceived by addressing the connectivities within an individual module and the relation between the various modules. Each module receives two major kinds of inputs, one from the mossy fibers and another from the climbing fibers. This inputs ultimately converge onto Purkinje cells, which eventually inhibit the deep cerebellar nuclei (the main output of the circuit). Although connectivities among neurons and interneurons in the cerebellar cortex occur within individual modules, the intracortical connections between modules occur prominently via the parallel fibers, apart from the Lugaro cell axons running along the parallel fibers and contacting different inhibitory neurons (including Purkinje cells, Golgi cells and stellate cells [76, 84, 169]). Moreover, at the cerebellar input, common mossy fibers can activate more lobules and a single olivary neuron also usually reaches different modules even at a considerable distance (for review see [74, 272]). In this review we disectionate the cerebelar organization by studying each stage separately.

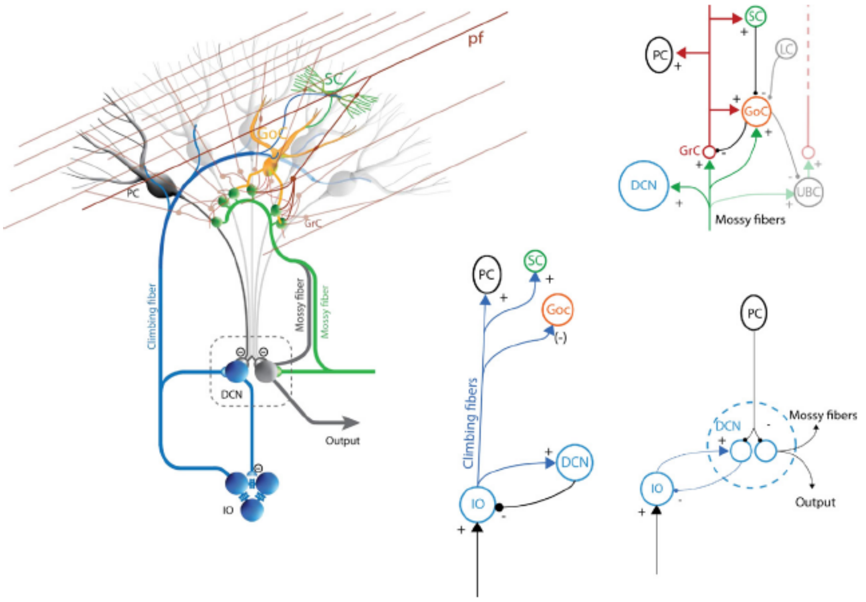


Figure 3.1: The organization of a cerebellar module - This schematic drawing shows the most relevant connections within a cerebellar module. The cerebellar module is made of a series of connections, in which different circuit elements communicate in closed loops and are contacted by the same afferent fiber set. The mossy fibers reach granule cells (GrC) and DCN cells which, in turn, receive inhibition from the same common set of Purkinje cells (PC). Moreover, the IO cells emit climbing fibers that contact DCN and PC, which also project to the same DCN cells. For convenience, the circuit can be divided in three sections illustrated schematically in the insets. The top-right inset also shows a Lugaro cell (LC) and a Unipolar Brush cell (UBC) (typical of the vestibulocerebellum).

3.2.1.1 The input stage: the granular layer

The mossy fibers provide one of the major inputs to the cerebellum and mediate sensorimotor and higher cognitive inputs via dedicated pathways running through the spinal cord, brainstem and cerebral cortex [135]. The properties of the mossy fiber firing pattern appear to depend on the specific characteristics of the particular input source and the actual stimulus status. For example, during slow head rotations, the vestibular input is represented through a linear encoding of mossy fiber spike rates, typically in the 0–40 Hz range [13, 18]; the trigeminal input tends to generate spike bursts in response to transient stimuli causing corresponding bursts in granule cells [46, 223]; and the oculomotor eyeball input as well as joint input appears to produce both bursts and tonic discharges related to changes in position [152, 270]. Since many of the sensory systems nuclei, pontine nuclei and cortical efferents also include neurons capable of both phasic and tonic discharge (e.g. see [149, 200, 235]), the combined capacity is probably rather common in mossy fiber signaling, even though some spiking patterns may only become apparent during a particular status of the stimulus.

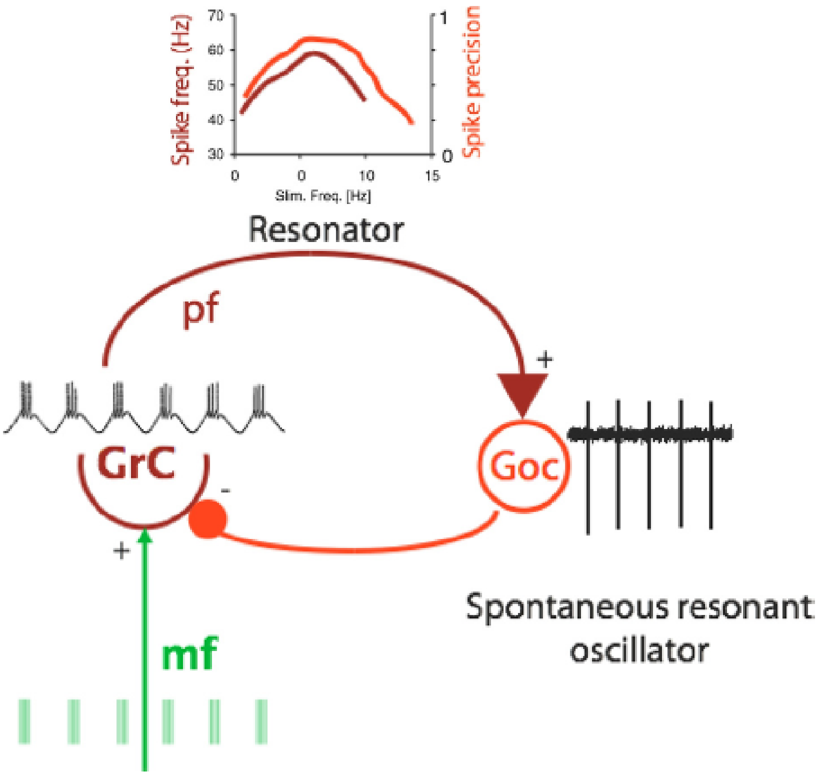


Figure 3.2: The cellular basis for low-frequency oscillations and resonance in the cerebellar granular layer - The granule cell (GrC) and the Golgi cell (GoC) are endowed with ionic currents that allow the emergence of theta-frequency resonance. Granule cell intrinsic excitability is enhanced in the 4–10 Hz frequency range, and Golgi cells show enhanced responses and precision at the same frequencies. Moreover, Golgi cells show pacemaker activity and phase reset with the corresponding characteristic period. This circuit is therefore appropiately designed to generate enhanced responses when, for example, the mossy fiber input would occure in theta-burst patterns. Experimental traces have been redrawn from [64, 92, 250].

Signals coming into the cerebellum through the mossy fibers are processed in the granular layer network, which includes a feed-forward inhibitory loop (mossy fiber→Golgi cell→granule cell), a feedback inhibitory loop (mossy fiber→granule cell→Golgi cell→granule cell), and a feed-forward dishinibitory loop (mossy fiber→granule cell→stellate cell→Golgi cell→granule cell). Here, with the intervention of the inhibitory circuits and synaptic plasticity, mossy fiber spikes are recoded into new spatiotemporally organized sequences by granule cells and Golgi cells exploiting their specific electroresponsive properties, which are specialized for sustaining bursting and repetitive activity on specific frequency bands ([58, 64, 190, 249, 250]) (see Figure 3.2). Four relevant aspects of this processing are:

1. Granular layer processing is fast and precise since this earl response need to be as fast as possible (even though it might lead to inaccurate signals); output spikes are emitted within milliseconds exploiting fast synaptic and excitable mechanisms [45, 61, 242].

2. Specific input patterns, under the guidance of inhibitory circuits, can induce bidirectional NMDA receptor-dependent long-term synaptic plasticity at the mossy fiber-granule cell synapse [14, 63, 96, 186, 190, 230, 247]. Long-term potentiation (LTP) and probably also long-term depression (LTD) are expressed presynaptically [247] and (D'Errico, Prestori and D'Angelo, unpublished observations), and as such they may have a prominent impact on timing through their control of repetitive neurotransmitter dynamics, i.e. short-term facilitation and depression [208]. However, the functional explanation of this synaptic plasticity is still an open issue which will have to be address in order to achieve a fully functional cerebelar model.

3. By controlling first spike delay, LTP would allow spikes to fall inside the window set by Golgi cells feed-forward inhibition, while LTD would drive the granule cells response beyond the window limit ("window-matching" effect; [58, 67]). By doing so, the granular layer operates a spatiotemporal filtering of signals and a spatiotemporal redistribution of activity, which can eventually lead to computational operations involving coincidence detection and pattern separation (Mapelli J, Gandolfi D and D'Angelo E, unpublished observations). How this complex response can be achieved by adjusting the synaptic weights in the granular layer circuit is the main aim of this chapter.

4. The granule cells are resonant and the Golgi cells are pacemaking and resonant at low frequency (< 10 Hz in vitro, but probably higher in vivo [64, 249, 250, 274]). The granular layer can be entrained in repetitive synchronous discharges in the 7–25 Hz range [52, 53, 115, 215]. Thus together, these four elementary aspects of granular layer processing show that this layer is in principle well equipped to control the absolute timing and phase of oscillations and resonance (Fig. 3.2). This control is of fundamental importance since every subsequent computation in the cerebellum will depend on it.

3.2.1.2 The Purkinje cells and the molecular layer

A second major input to the cerebellar cortex comes from the inferior olive through the climbing fiber system. The inferior olive itself receives inputs from many brain regions that form, in fact, directly or indirectly a source for one of the mossy fiber inputs (for review see [74]). Although the inferior olive has traditionally be proposed as an error signal generator, recent experimental studies have shown the olivary neurons have a propensity to oscillate [48, 161, 176, 268], and their climbing fiber activities can produce theta-frequency patterns in the cerebellar cortex by directly innervating the dendritic arbors of Purkinje cells and inhibitory interneurons, including stellate cells [19] and possibly Golgi cells [286]. In fact, in Purkinje cells, climbing fiber activities are able to exert a very powerful phasic excitation through the complex spike [198]. The complex spike signal may carry an error in motor performance and as such it might be used as an instruction for generating synaptic plasticity at the parallel fiber to Purkinje cell synapse [50, 138].

The Purkinje cells have their own processing mechanisms, which also rely on intrinsic electroresponsive properties and synaptic plasticity. Their most relevant computational aspects are:

1. Purkinje cells are spontaneously active (30–50 Hz) and their discharge is modulated by inputs from the olivary neurons, granule cells, and molecular layer interneurons. Following the original observations by [3], it was recently shown that the molecular layer can sustain synchronous high-frequency (100–200 Hz) oscillations entraining the Purkinje cells [71, 199]. Thus, the granular layer patterns need to be precisely synchronized in order to efficiently affect Purkinje cells activity.

2. Purkinje cell synapses are sites of plasticity, including for example LTD and LTP at the parallel fiber to Purkinje cell synapse [50] (this plasticity has been proposed in traditional theories as the main site of motor learning at the cerebellum), LTD at the climbing fiber to Purkinje cell synapse [114], and LTP at the interneuron to Purkinje cell synapse [151]. For alternative interpretations see [177].
3. Purkinje cells may act as perceptrons exploiting their plasticity capabilities for pattern recognition [31].
4. Purkinje cells may communicate through spike pauses modulating the interspike intervals over milliseconds [127, 241, 255]. In this respect, it is relevant to note that under particular forms of anesthesia, but less so in the awake state, Purkinje cells show extensive bistable up-and down states lasting over much longer time periods of hundreds of milliseconds to seconds [141, 179, 233].

Thus, apart from a direct inhibitory feed-forward control imposed by the molecular layer interneurons (stellate and basket cells), the dynamic firing properties of the Purkinje cells are presumably tightly controlled by both the climbing fiber and the parallel fiber system, which have probably impact on all four aspects described above.

3.2.1.3 The cerebellar output stage

The Purkinje cells form the only output of the cerebellar cortex and they inhibit the cells of the vestibular nuclei (VN) and deep-cerebellar nuclei (DCN), which ultimately convert the activities of the microzones and those of the mossy fiber and climbing fiber collaterals into the final cerebellar output (Fig. 3.1) (and therefore, they produce the final corrective signals in motor control tasks). The VN and DCN are thus at a key location within the cerebellar network. Their projection neurons can be divided into at least two main groups: those that inhibit the inferior olivary (IO) cells presumably regulating their coupling and oscillations [72, 141] and those that exert a more direct control on the ultimate motor output (Fig. 3.3). In fact, while the role of inhibitory interneurons has not been demonstrated convincingly in freely behaving animals yet, principal neurons can be divided into types A and B, which modulate their firing in relation to activation of agonist or antagonist muscles [111, 269] (see chapter 4 for a further explanation on the agonist/antagonist cerebelar theory).

The most relevant properties of the DCN neurons are the following:

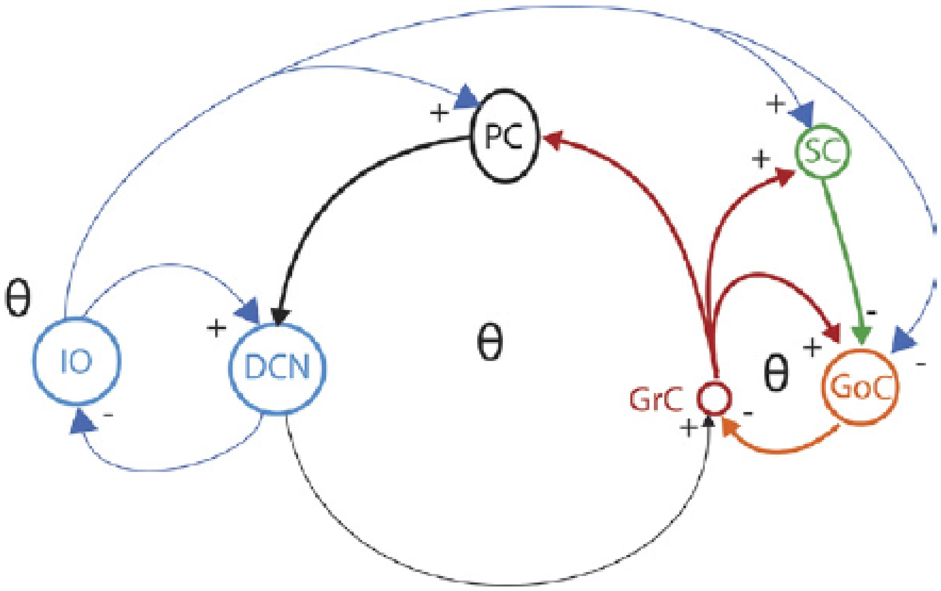


Figure 3.3: Multiple loops involved in controlling cerebellar low-frequency activity - Due to the intrinsic resonance of neurons in the granular layer loop (GrC-GoC), to signal reentry from the DCN, and to oscillatory activity in the IO complex, the olivo-cerebellar circuit demonstrates a design suitable to operate within a dominant frequency band of <10 Hz. GrC, GoC, PC, mf and cf. indicate granule cells, Golgi cells, Purkinje cells, mossy fibers and climbing fibers, respectively. While the backbone of the circuit summarizes classic knowledge on the cerebellum (e.g. see [88]), some less known connections like those from climbing fibers to the SC and GoC [19] and from the DCN to GrC [32, 265] may also play an important role for the overall network synchronization and phase-locking.

1. DCN neurons are intrinsically active at frequencies ranging from a few Hz to tens of Hz [267]. In general, the intrinsic dynamics of the cells generate silent pauses and often rebound excitation, producing alternating phases of activity depending on the strength and length of the inhibition induced by the Purkinje cells [267]. The projecting GABAergic and non-GABAergic DCN cells can be distinguished based on their synaptic currents; the synaptic currents in the GABAergic cells have lower amplitude, lower frequency and slower kinetics than those of the non-GABAergic cells [266]. Therefore, the GABAergic cells appear better designed for conveying phasic spike rate information, whereas the larger non-GABAergic cells relay more faithfully tonic spike rate.

2. The DCN and VN neurons may act as one of the main substrates of downstream motor memory storage [137, 175, 275]. This hypothesis is supported by the fact that the synaptic strength of their inputs as well as their active membrane properties can be readily modified [6, 7, 261]. Interestingly, as predicted by a recent model of the cerebellar nuclei neurons and their Purkinje cell and mossy fiber collateral inputs [75], Pugh and Raman [220] showed that the extent of plasticity varies with the relative timing of synaptic excitation evoked by the mossy fiber collaterals and the hyperpolarization induced by the Purkinje cells activity.

Thus, one can hypothesize that the synchronous oscillations in the Purkinje cell activities together with plasticity at the mossy fiber-DCN and the Purkinje cell-DCN synapses form the main mechanistic tools to control the activity in the DCN output neurons, and that different sets of neurons in the DCN are sensitive for oscillations at different frequency ranges (for details about hypothesis see [75]). But in addition to this, there are several connections between these three main subcircuits. Activity of the inferior olive can be conveyed through the climbing fiber→stellate cell→Golgi cell circuit [19, 87, 259]. Moreover, Golgi cells may also be inhibited directly through metabotropic receptor activation by the climbing fibers, as proposed by [286]. Finally, some mossy fibers can originate from the DCN [265]. Thus, activity of the IO and DCN can be reverberated in the granular layer.

As a whole, one can conclude that all circuit subsections make their own contribution to oscillatory activity in the cerebellum and eventually interact through several internal connection loops. Importantly, the granular layer is the starting point for the activities generated in several of the other circuit sections.

3.2.1.4 The importance of the oscillations in the granular layer

From the above commented, it emerges that granular layer oscillations may play a critical role in cerebellar activity. Low-frequency oscillations are fundamental for several neurophysiological processes, including motor control, the formation of memories and sleep (for review see [35]). Low-frequency activity was shown to correlate with that in the cerebral cortex, and may therefore represent a suitable band for communication between cerebellum and the thalamo-cortical system [210]. Moreover, it may provide a binding element between the two main functional sections of the cerebellar cortex, i.e. mossy fiber and the climbing fiber input systems. The disruption of appropriate control mechanisms in the olive and DCN allows low-frequency oscillations to prevail at the DCN output stage causing muscle tremor, as it occurs with harmaline application and in essential tremor in humans [178]. Muscle tremor occurs at <10 Hz for larger muscles, and is also species-specific ranging from about 7 Hz to 25–30 Hz [111, 164]. Therefore, low-frequency patterns may have important yet incompletely understood roles in cerebellar control, opening new fields for future research.

Low-frequency oscillations are essential for signal processing at high rate (for review see [35]). Since the afferent inputs are largely encoded with 5-ms precision in the 1st spike delay [147], the same accuracy in the time-window matching process seems needed for efficient elaboration of incoming information. The repetition of these time-windows during protracted stimulation is predicted to generate high-frequency oscillations in the granular layer, providing a coherent framework for data processing over large granular layer fields. This periodic output may then be sampled by Purkinje cells, which also have a high-frequency regimen of activity and provide precise timing of Purkinje cells simple spike activity over the same scale [127, 241]. The high-frequency sampling based on oscillating background could be important if the Purkinje cell works as a perceptron, allowing signal sampling over very short time windows and improving pattern recognition [31].

The repetition of spikes emitted by granule cells in the gamma frequency band may also be important to implement other physiological processes. First, parallel fiber-Purkinje cell release probability is usually low (except for ascending axon synapses, [134, 245]), so that short high-frequency bursts can ensure efficient transmission through short-term parallel fiber-Purkinje cell facilitation. Secondly, there are forms of parallel fiber-Purkinje cell LTD which require doublets [43, 44], so that persistent changes could be induced only at those synapse that receive high-frequency inputs. The demonstration of high-frequency oscillations in the granular layer remains an interesting challenge for future cerebellar investigations.

In conclusion, available evidence suggests that both slow and fast granular layer oscillations could have specific roles in cerebellar signal processing. While high-frequency oscillations may support millisecond-scale timing in granular layer activities preparing signals for Purkinje cells and allowing fast and precise elaboration of single motor acts, low-frequency oscillations may support repetition of complex motor sequences. Indeed, the granular layer demonstrates a theta-frequency preference that is indicative of the existence of such higher-order dynamics, and anatomical and functional evidence suggests that these could involve entire cerebellar modules. This low frequency activity may be important for coordinating cerebellar communication with the sensorimotor cortex correlating with processes like learning, arousal and attention. Thus, experimental works have shown the existence of oscillations at several frequency bands and different sites at the cerebellum, but an functional explanation and the way in which these oscillations can efficiently support the execution of motor control tasks. Computational neuroscientist studies probably play an important role in the understanding of these new theories.

3.3 Materials and methods

Brain computations employ spikes and information is transmitted between neurons at the synapses, which are modifiable thereby storing memory of past activity. It is therefore desirable to investigate brain circuit functions by means of realistic spiking networks [100, 184, 218]. The main objective of the work described in this chapter is to study the impact of synaptic weights on temporal spike train detection in the granular layer of the cerebellum. Thus, we have widely tested the influence of different synaptic weights using a very common methodology in neuroscience. We have firstly defined a control configuration, with default synaptic weights. After that, we have explored the synaptic weight space in perpendicular axes (one synapse type is changed in each experiment). Moreover, we have carried out some other experiments modifying two different axes (e.g. changing the synaptic weights in GrC-GoC and GrC-SC simultaneously), obtaining very similar results to the ones shown in the following section, but 3-dimensional graphs have not been included for the sake of clarity.

3.3.1 The EDLUT Simulator

For extensive spiking network simulations, we have developed and used an advanced Event-Driven simulator based on LookUp Tables (EDLUT) [40, 227]. EDLUT (released as free software under GPLv3 license) allows the behavior of a pre-defined cell model (whose dynamics are defined by differential equations) to be compiled into lookup tables. Then, complex-network simulations can be performed without requiring an intense numerical analysis. With EDLUT, it has been possible to split spiking neural network [100] simulations in the two following stages.

- Cell behavior characterization. In this first stage each cell model is simulated reiteratively with different input stimuli and from different cell states. This allows to scan the cell behavior which is compiled into lookup tables. Therefore, at this stage, the tool uses conventional numerical methods (such as the Runge-Kutta method) to approximate the cell state variables after receiving a specific stimulus. This represents a massive simulation load for each cell type but, once this is carried out, the results are stored in well structured tables. In this way, numerical calculation during network simulations can be avoided, significantly accelerating the simulation.

- Network simulation towards system behavior characterization. At this second stage, multiple network simulations are run with different weight configurations. This stage does not require an intense numerical calculation; the cell state at different times is retrieved from the lookup tables, allowing massive simulations with little computational expenditure. In order to explore the large dimensional space of connection weights of the two experiments addressed in this work, we have performed more than 1 million simulations (of 1 second each) (consuming roughly a total of about 11 days of a CPU at 2.4 GHz) with a network composed by 5177 cells.

Further information about this advanced software for spiking neural networks simulation can be found in section 2.3 of this document.

3.3.2 Neural Models

The simulated network consists of three different cell types. These cell types are modeled by a modified version of the Leaky Integrate-and-Fire model (LIF) [100] whose dynamics are defined by Expressions 3.1, 3.2 and 3.3. These neuron models account for synaptic conductance changes rather than simply for fixed current flows, providing an improved description over common I&F models. The version of these cell models implemented for EDLUT simulator can be found and downloaded at the EDLUT project official site (see `http://edlut.googlecode.com`).

Each network neuron is simulated using a specific model parameterization and includes a subset of the following synaptic receptors:

$$g_{AMPA}(t) = \begin{cases} 0 & \text{when } t < t_0 \\ g_{AMPA}(t_0) \cdot e^{-\frac{t-t_0}{\tau_{AMPA}}} & \text{when } t \geq t_0 \end{cases} \tag{3.1a}$$

$$g_{NMDA}(t) = \begin{cases} 0 & \text{when } t < t_0 \\ g_{NMDA}(t_0) \cdot e^{-\frac{t-t_0}{\tau_{NMDA}}} & \text{when } t \geq t_0 \end{cases} \tag{3.1b}$$

$$g_{GABA}(t) = \begin{cases} 0 & \text{when } t < t_0 \\ g_{GABA}(t_0) \cdot e^{-\frac{t-t_0}{\tau_{GABA}}} & \text{when } t \geq t_0 \end{cases} \tag{3.1c}$$

$$\nu_{EC}(t) = \begin{cases} 0 & \text{when } t < t_0 \\ \nu_{EC}(t_0) \cdot e^{-\frac{t-t_0}{\tau_{EC}}} & \text{when } t \geq t_0 \end{cases} \tag{3.1d}$$

where t denotes the current simulation time and t_0 denotes the time when an input spike is received. g_{AMPA} and g_{NMDA} represent AMPA and NMDA receptor-mediated conductance respectively, which provide excitation and g_{GABA} represents the GABA receptor-mediated conductance, which provides inhibition. τ_{AMPA}, τ_{NMDA} and τ_{GABA} are the decaying time constants of each receptor type. ν_{EC} does not represent a conductance but is computed in the same way and accounts for the effect of the electrical coupling. τ_{EC} denotes the time constant of the spikelets. These synaptic conductance responses were modeled as simple decaying exponential functions which provide several advantages. Firstly, it is an effective representation of realistic synaptic conductance. Thus, the improvement in accuracy as compared to the next most complex representation, a double-exponential function, is hardly worthwhile when considering the membrane potential waveform [227]. Secondly, each exponential conductance type requires only a single state variable, since synaptic inputs through several synapses can simply be recursively summed when updating the total conductance if they have the same time constants:

$$g_{AMPA_{postspike}}(t) = G_{AMPA,j} + g_{AMPA_{prespike}}(t) \tag{3.2}$$

$G_{AMPA,j}$ is the AMPA receptor synaptic weight of synapse j; a similar relation holds for the other synaptic receptors. Most other representations would require additional state variables and/or storage of spike time lists, so the exponential representation is particularly efficient in terms of memory usage and computation time.

The parameters (time constants) of each synaptic receptor (Equations 3.1a, 3.1b, 3.1c and 3.1d have been chosen to model three cerebellar cell types: granule, stellate, and Golgi cell (see Table 3.1) [209, 229, 243, 262]. Note that different synapses of a cell might include different synaptic receptors.

Parameter	**GrC**	**SC**	**GoC**
Membrane capacitance (C_m)	2pF	4pF	50pF
Total AMPA-receptor peak conductance (g_{AMPA})	3.9nS	12nS	136nS
Total NMDA-receptor peak conductance (g_{NMDA})	4.5nS	-	-
Total GABA-receptor peak conductance (g_{GABA})	30nS	14nS	100nS
Firing threshold (θ_{V_m})	-40mV	-40mV	-50mV
Resting potential (E_{rest})	-65mV	-56mV	-65mV
Excitatory reversal potential (E_{AMPA},E_{NMDA})	0mV	0mV	0mV
Inhibitory reversal potential (E_{GABA})	-65mV	-58mV	-65mV
Resting conductance (G_{rest})	0.2nS	0.2nS	3nS
Resting time constant (τ_m)	10ms	20ms	16.7ms
AMPA receptor time constant (τ_{AMPA})	0.5ms	0.64ms	0.5
NMDA receptor time constant (τ_{NMDA})	40ms	-	-
GABA receptor time constant (τ_{GABA})	10ms	2ms	10ms

Table 3.1: Parameters of the different used cell types - These parameters have been obtained from the following papers: Granule Cell (GrC) [60, 61, 62, 64, 95, 208], Stellate Cell (SC) [42, 47, 120, 167, 257], and Golgi Cell (GoC) [92, 249, 250]

.

The membrane potential is partially computed (V_{m-c}) by differential Equation (3A), which accounts for the effect of chemical synapses (including AMPA, NMDA, and GABA receptors) and resting conductance (G_{rest}),

$$\begin{aligned} C_m \frac{dV_{m-c}}{dt} &= g_{AMPA}(t) \cdot 1.273 \cdot (E_{AMPA} - V_{m-c}) + \\ &+ g_{NMDA}(t) \cdot 1.358 \cdot g_{\infty,NMDA}(V_{m-c})(E_{NMDA} - V_{m-c}) + \\ &+ g_{GABA}(t) \cdot (E_{GABA} - V_{m-c}) + G_{rest}(E_{rest} - V_{m-c}) \end{aligned} \tag{3.3a}$$

$$g_{\infty,NMDA}(V_{m-c}) = \frac{1}{1 + e^{-\alpha V_{m-c}}[Mg^{2+}]/\beta} \tag{3.3b}$$

where C_m denotes the membrane capacitance, E_{AMPA}, E_{NMDA}, and E_{GABA} represent the reversal potential of each synaptic conductance and E_{rest} is the resting potential. g_{AMPA}, g_{NMDA}, and g_{GABA} conductances integrate all the contributions received through individual synapses. $g_{\infty,NMDA}(V_{m-c})$ is a gating function which accounts for the voltage-dependent magnesium block [145], with $\alpha = 62V^{-1}$, $[Mg^{2+}] = 1.2mM$ and $\beta = 3.57mM$ (adapted from [95]).

To include the electrical coupling effect, the total membrane potential (V_m) is defined by Equation 3.4.

$$V_m = V_{m-c} + f_{EC}\nu_{EC}(t) \tag{3.4}$$

where the potential $\nu_{EC}(t)$ integrates the contribution of the current spikelets and f_{EC} is a factor (0.044) to adapt the amplitude of the spikelet for the current coupling coefficient [39].

Since we are using an event-driven simulation scheme, each time a neuron receives a spike, all the neural state variables (V_m, g_{AMPA}, g_{NMDA}, g_{GABA}, and v_{EC}) are updated (Equations 3.1 and 3.3) and then, the corresponding synapse state variables are increased following Expression 3.2. Equation 3.2 is amenable to numerical analysis. In this way, we can calculate V_m, and therefore, the time when it will reach the firing threshold (θV_m) can be predicted. An example of membrane potential computed for neurons of this network is shown in Figure 3.4.

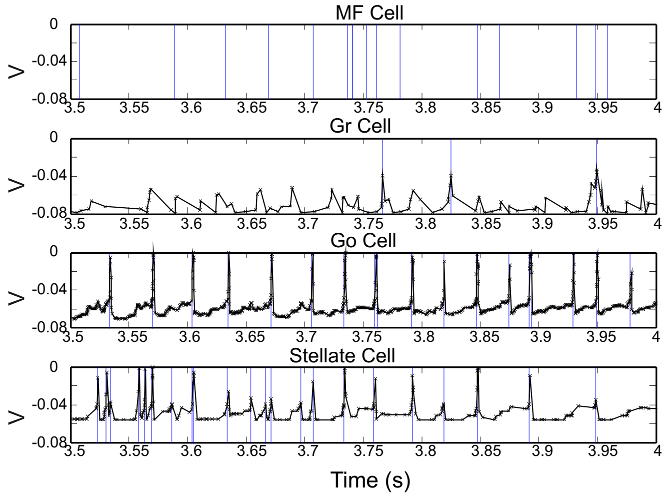
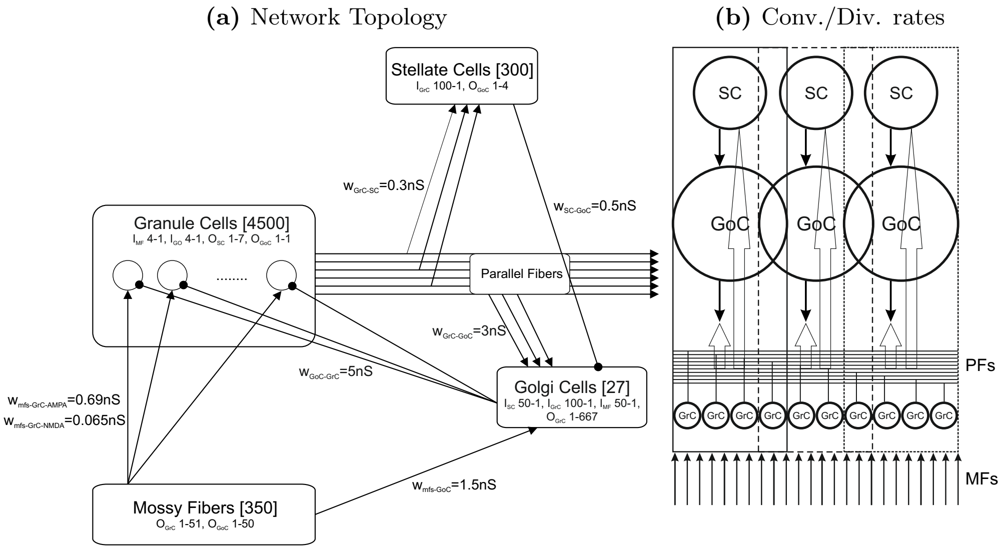


Figure 3.4: Event-driven simulation of the included cell types - The response (membrane potential and output spikes) of the cell types (GrC, GoC, and SC) to an MF input train is shown. Spikes are indicated by vertical gray lines. Since we have used an event-driven simulation, the membrane potential is only computed when a spike is received or emitted, that is, at the points indicated by a cross. Therefore, the lines which join these points do not represent actual intermediate values.

3.3.3 Network topology

The model shown in Figure 3.5a represents an abstraction of the granular layer including as a whole 5177 cells of different types. The generation of the network topology was developed in two different steps: in the first one, the number of constitutive elements was calculated following anatomical studies of cell densities in the granular layer [165]. In a second stage, we have connected the constitutive elements respecting convergence-divergence ratios and connectivity rules found in biology [88, 117, 118]. Overlapping of multiple innervation fields allowed the mixing of contributions of single neurons within the network 3.5b. The selection of source and target cells in every synapse has been randomly developed avoiding the temporal characteristics being influenced by a certain connection pattern.

However, neither cerebellar glomeruli nor lateral inhibition [35, 88, 190] of GrCs have been explicitly implemented in order to focus on temporal rather than spatial dynamics. Previous works demonstrated that the evoked granular layer response to local stimuli appeared fragmented and partially surrounded by regions of lateral inhibition [190]. When local spike trains reach the glomeruli, connected GrCs and GoCs



(a) Network Topology

(b) Conv./Div. rates

Stellate Cells [300]
I_{GrC} 100-1, O_{GoC} 1-4
$w_{GrC\text{-}SC}$=0.3nS
$w_{SC\text{-}GoC}$=0.5nS
Granule Cells [4500]
I_{MF} 4-1, I_{GO} 4-1, O_{SC} 1-7, O_{GoC} 1-1
....
Parallel Fibers
$w_{GrC\text{-}GoC}$=3nS
$w_{GoC\text{-}GrC}$=5nS
Golgi Cells [27]
I_{SC} 50-1, I_{GrC} 100-1, I_{MF} 50-1, O_{GrC} 1-667
$w_{mfs\text{-}GrC\text{-}AMPA}$=0.69nS
$w_{mfs\text{-}GrC\text{-}NMDA}$=0.065nS
$w_{mfs\text{-}GoC}$=1.5nS
Mossy Fibers [350]
O_{GrC} 1-51, O_{GoC} 1-50
SC
SC
SC
GoC
GoC
GoC
PFs
GrC
MFs

Figure 3.5: Structural organization of the simulated granular layer network - (a) Network topology of the cerebellar granular-layer model. Excitatory connections are represented by arrows and inhibitory connections, by circles. Input spike trains are received through MFs which excite GrCs and GoCs. Each GrC receives approximately 4 MFs and each MF targets divergently a wide set of GrCs. Each GoC gathers input activity from a set of 50 MFs and from 100 GrCs. The output activity of the GoC inhibits a set of 667 randomly-chosen GrCs. Finally, 100 GrCs excite a SC and 50 SCs inhibit each GoC. Note that I represents the convergence rate and O, the divergence rate of the input/output connections. (b) Partial overlapping scheme of connection fields allows redistribution of activity in the network. The keys for this are (i) random connections of every cell (GrC, GoC, and SC), (ii) 75% overlapping of MF to GoC connections, (iii) redistribution of activity through parallel fibers.

are excited. Due to the long axons observed in GoCs, they are able to inhibit a larger area of GrCs than those previously excited and a center-surround organization (with excitation-inhibition respectively) will be caused at GrCs. However, this property (not explicitly implemented in our model) is not expected to influence the temporal filtering properties which are discussed in this chapter. The granular layer model which has been simulated in this work is composed of the following cell layers:

1. Mossy Fibers (MF) (350): They convey the input spikes and activate GrCs and GoCs.

2. Granule cells (GrC) (4500): The population of GrCs has a number of connections

per cell (from MF) that follows a Gaussian distribution with a mean of 4 connections and a standard deviation $\sigma = 1$ connection (each source MF has been randomly selected but avoiding a same MF to excite the same GrC more than once). Each GrC is inhibited by four different GoC.

3. Golgi cells (GoC) (27): Each GoC receives excitatory connections from 100 GrCs and 50 MFs, and inhibition from 50 SCs. The output of each GoC inhibits 667 GrCs (on average) which contributed to its excitation.

4. Stellate cells (SC) (300): Each SC receives excitation from 100 GrCs and inhibits the activity of 4.5 GoCs (on average). MF-GoC, GrC-GoC, and GrC-SC connections are converging and overlapping, whereas GoC-GrC connections are mainly diverging and overlapping.

Synapse	**Receptor**	**Default Weight Value**
$MF \rightarrow GrC$	AMPA	0.69nS
$MF \rightarrow GrC$	NMDA	0.065nS
$MF \rightarrow GoC$	AMPA	1.5nS
$GrC \rightarrow GoC$	AMPA	3nS
$GoC \rightarrow GrC$	GABA	5nS
$GrC \rightarrow SC$	AMPA	0.3nS
$SC \rightarrow GoC$	GABA	0.5nS

Table 3.2: Default configuration of synaptic weights - $MF \rightarrow GrC$ weights have been obtained from [95], $MF \rightarrow GoC$, $GrC \rightarrow GoC$, and $GoC \rightarrow GrC$ have been adapted from [185] and $GrC \rightarrow SC$ and $SC \rightarrow GoC$ have been calculated in order to preserve the activity frequencies proposed in [185].

Table 3.2 shows the different kinds of receptors which have been included in each synaptic connection. While both AMPA and NMDA receptors have been included in $MF \rightarrow GrC$ synapses, only AMPA transmitters have been implemented at the remaining excitatory synapses ($MF \rightarrow GoC$, $GrC \rightarrow GoC$, and $GrC \rightarrow SC$). The available evidence suggests, through in situ hybridization studies (ISH), that GoCs of the developing rat possess mRNA for NR1 and NR2D NMDA-receptor subunit [8]. Stellate and basket cells also transcribe these genes [277]. However, patch-clamp recording experiments suggest that both diheteromeric and triheteromeric NR2D-containing receptors are expressed only extrasynaptically on these cells, and neither receptor type participates in parallel fiber (PF) to GoC synaptic transmission [29, 56]. Although these experiments also indicate the presence and effect of NR2B-containing receptors on synaptic transmission in developing GoCs, ISH experiments suggest that adult GoCs do not express NR2B [212]. Therefore, in the absence of direct evidence for the presence of functional synaptic NMDA receptors in adult cerebellar GoCs, they have not been included in the presented study.

Firstly, all the synaptic weights within a certain connection category were set to the same value (as indicated in Table 3.2). The synaptic values were tuned following previous neuro-physiological studies based on the global peak conductance values (the maximum conductance resulting from the simultaneous activation of all the synapses reaching a single cell) [185]. The global peak conductance of the AMPA receptor channel in a GrC was 2.76nS [95], so every synaptic weight was fixed to $2.76/4 = 0.69nS$ (depending on the number of MFs -between 3 and 5- reaching the cell). The same methodology allowed to fix the synaptic weight of NMDA receptor channel to $0.26nS/4 = 0.065nS$ on average [95]. This apparently low weight in relation to AMPA channel can be explained taking into account the longer time constant associated to NMDA channels which produces a sustained activity.

Once the input channels of the network were established, the inhibitory loops were adjusted in order to get stability and synchronism. In this way, the global peak conductance of the AMPA channel reaching GoC was set to 375nS and the parallel fiber global peak conductance was 4 times the MF global peak conductance [185], so the final synaptic weight was 1.5nS in every $MF \rightarrow GoC$ and 3nS in every $GrC \rightarrow GoC$ connection. Following the same reference, the $GoC \rightarrow GrC$ synaptic weight was set to 5nS. Finally, the dis-inhibitory loop was experimentally configured in order to adjust the SC frequency and not completely inhibiting the GoCs. The $GrC \rightarrow SC$ synaptic weight was set to 0.3nS and $SC \rightarrow GoC$ to 0.5nS.

After simulating the network with the initial synaptic weights (Table 3.2), these were then modified to different values in order to explore the weight space. Moreover, we have studied the electrical coupling among GoCs and also among SCs, by multiplying the default coupling coefficient by a value between 0 (no coupling) and 0.1 (high coupling). An extensive study of the electrical coupling impact would require the evaluation of different topologies with different convergence ratios of multiple GoCs onto the same GrC. Nevertheless, this is somewhat equivalent to different weights between GoCs and

GrCs already covered by previous experiments. We have evaluated separately the potential impact of coupling among GoCs and SCs.

This network incorporated the essential structural and functional requirements needed to investigate the basic working hypotheses of coincidence detection and time slicing in cortical microzones. A further refinement of connectivity and dynamic setting of circuit weights through an appropriate learning rule are outside the scope of this work, but will be considered in the future.

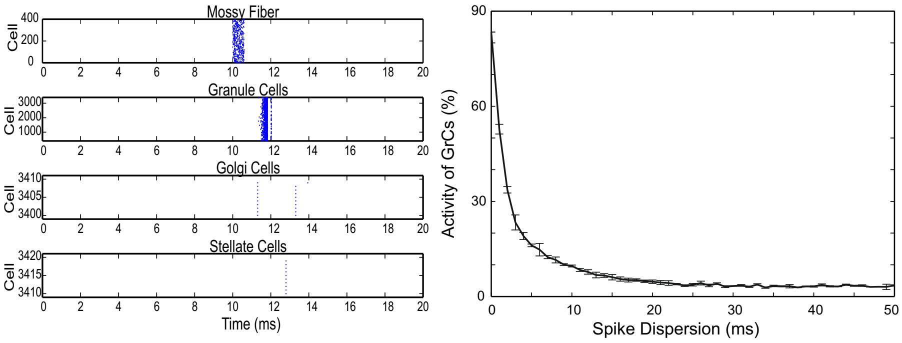
3.3.4 Coincidence Detection Methods

Coincidence detection is predicted to play an important function in the granular layer [192] and GrCs are indeed capable of sustaining such operation [61, 148]. Thus, we have considered how the granular layer can reduce temporally scattered inputs allowing only the transmission of synchronized groups of spikes. With a specific network configuration (dominant weights: $MF \rightarrow GoC$ and $GoC \rightarrow GrC$), the system is mainly maintained at a low output (inhibited) and most parallel fibers are silent. Only synchronous input spike trains overcome this bias inhibition and reach the parallel fibers. The strength of inhibition through the GoC will determine the level of synchronism that a train requires to reach the parallel fibers.

In order to test this hypothesis, we have used input spike trains coming through the MFs with different dispersion levels. Then, the number of GrCs that fired (called response rate) was taken as an estimation of how input transmission efficacy depended upon input spike synchronization.

One spike per MF was located, inside a given time window, at a time drawn from a uniform random distribution. Collectively, spikes across the various fibers formed a distributed train whose synchrony level was determined by the window duration (shorter windows generate more synchrony than longer windows). This implied that each distributed train or stimulus pattern was composed by a fixed number of 350 spikes randomly distributed in time windows with different lengths. We have assessed the impact of different trains using time windows from 0 (completely synchronized activity) to 100ms according to Equation 3.5:

$$t_i(\Delta_j) = u(T, T + \Delta_j) \tag{3.5}$$



(a) Rastergram

(b) Coincidence Detection

Figure 3.6: Network activity in response to an input train - (a) Exemplar simulation executed with a spike train arriving at 10 ms and a duration of 0.5ms. Spikes generated by the GrCs have a delay of at least 1ms with respect to the MF input. Most GrCs fired before they were affected by the inhibition coming from the GoC, which takes about 3 ms to arise. Some GoCs show a second spike and SCs fire due to excitation generated by GrCs and transmitted through parallel fibers. The activity in each layer is plotted with a dot per spike. (b) Granular layer coincidence detection plot: GrC response rate vs. input synchronism level (time elapsed between the first and last spike of the input train). It also includes the standard deviation of the measurements (15 simulations per point). The synaptic weights are set to default values (see Table 3.2 for details).

where $t_i(\Delta_j)$ is the time of the spike fired by the cell i in the time window Δ_j and u is a realization of the uniform random distribution between T (initial time of the spikes) and $T + \Delta_j$.

This approach allowed the determination of the amount of synchronization that is required to allow transmission through the granular layer. In fact, we measured the activity of GrCs (percentage of firing cells) versus the input spike dispersion (seconds) as shown in Figs. 3.6a and 3.6b. In order to perform a quantitative analysis of this property, we have calculated the T_{20} value. This value is defined as the minimum time-window length (i.e. input train dispersion) which generates an output rate under 20% with respect to the output activity when a completely synchronous train stimulates the MFs. Low values of T_{20} indicate a high required input synchrony to be transmitted, thus a highly restrictive coincidence detection filtering.

For each network configuration and time window length (from 0ms to 100ms), we have simulated the network 3 times with different input trains chosen from the expression shown in Equation 3.5. In this way, we have shown the average of 3 measures per estimate in the next sections. We have measured the response rate of GrCs (number of GrCs firing divided by the total number of GrCs) generated after the input stimuli and finally, we have measured the T_{20} value.

3.3.5 Time-Slicing Methods

Feed-forward inhibition ($MF \rightarrow GoC \rightarrow GrC$ loop) [150] confines the maximal response of GrCs within a short (about 5ms) permissive time window [58, 190]. Then, once a spike train has been transmitted through the parallel fibers, the GoC is excited and therefore, the activity of the GrCs is even more strongly decreased ($GrC \rightarrow GoC \rightarrow GrC$ loop). The action of the GoC is terminated by the SC, which are excited by the parallel fiber activity ($GrC \rightarrow SC \rightarrow GoC \rightarrow GrC$ loop) [19]. In this way, transmitted trains would occur through a permissive time window followed by a post-transmission silence and a post-inhibitory rebound (see Figure 3.7b).

Thus, after a first conditioning train was delivered, the response to a test spike train was evaluated with different inter-spike distances (between 2 and 200ms after the first spike train). In this way, we could follow the evolution of network response sensitivity after a first spike train has been received and transmitted. The network (Figure 3.7a) was activated with well structured (synchronous) trains arriving on top of a 20Hz background base activity.

The strength and the length of these different excitability phases will depend on the time constants of the cells and synapses and on the synaptic weights at the various synapses in the circuit. As well as on the temporal noise filtering, time slicing is expected to depend on the strength of the involved synapses. Therefore, we have investigated the influence of synaptic weights that can modify the behavior of the network as a temporal filter (Figure 3.7b).

Each MF was stimulated with two spike trains (only one spike per train stimulated each MF) to investigate the sensitivity rebound. The two spike trains were separated by different time intervals (from 2 to 200ms) and were individually placed within a narrow time window of 1ms (corresponding to well-synchronized activity across trials).

$$t_{i,j}(\theta_k) = T + j \cdot \theta_k + u(0,1) \tag{3.6}$$

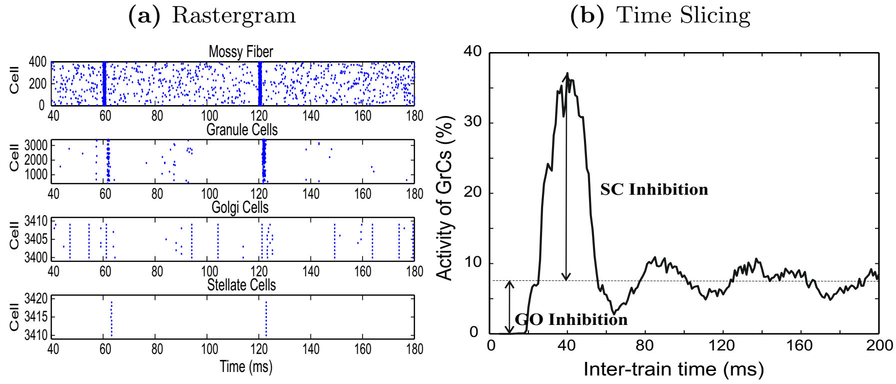


Figure 3.7: Network activity in response to a repeated input train - The 20Hz random activity distributed among the 350 MFs was generated as a background activity during the simulation time (500 ms). The network received two trains with inter-train intervals between 2ms and 200ms and an intra-train dispersion of 1ms. Default synaptic weights were used as indicated in Table 2. (a) Exemplar simulation with two input trains separated by 60 ms. Trains are superimposed over a background noisy activity. GrCs have their spikes concentrated just after the stimuli, and do not show remarkable activity in between them due to GoC inhibition. GoCs show pacemaking, are phase reset by the stimulus, fire around 2 spikes, and then are silenced by SC feed-back. Note that the activity in the GrC, GoC, and SC populations is highly synchronous. (b) Response plot for different inter-train time intervals. In this plot (and the following similar ones), we measured the activity (number of spikes) fired by GrCs after the second spike train reaches the GrCs (from 121ms to 123ms in this particular GrC raster plot in (a)). After the first train has been transferred by the granular layer, during the next 25ms, the sensitivity of the network to a new input is below the reference level. However, after another 30ms, the network shows a sensitivity enhancement. This is caused by the action of SCs, which inhibit the GoC and therefore reduce background inhibition. After 55ms more, smaller sensitivity rebound happens until achieving a stable state.

where $t_{i,j}(\theta_k)$ is the time (in ms) of the $j-th$ spike fired by the cell i with an inter-spike interval θ_k, u is a realization of the uniform random distribution between 0 and 1ms, and T is the time at which the first spike train starts (60ms). Moreover, low basal noise has been added to the main signal as 20Hz activity among the 350 MFs during the simulation time (300 ms). A single simulation is illustrated in Figure 3.7a.

For each synaptic weight configuration and inter-spike interval, we have simulated the network 5 times with different input trains chosen from Expression 3.6. In this way, all the results shown in the next sections correspond to the average of 5 measures per estimate. We have measured the GrC response rate (number of GrC spikes divided by the total number of GrCs) generated by the second synchronous spike train.

Finally, we have extracted two measures in order to quantitatively describe the efficiency of this sensitivity rebound. We have defined the rebound period as the inter-spike interval maximizing the response rate in GrCs. This value has been calculated by means of the response rate gradient, which allows to obtain the local maximum of the GrC activity rate.

The second measure was the Sensitivity Rebound Index (SRI), which is defined in the following equation 3.7:

$$SRI = \frac{Act_{Max} - Act_{Min}}{Act_{Av}} \tag{3.7}$$

where Act_{Max} denotes the maximum rate of activity of the sensitivity rebound (and which corresponds to the second spike train reaching the GrCs when the SCs are inhibiting the GoCs), Act_{Min} denotes the minimum rate of activity obtained before the rebound (and which matches with the second spike train reaching GrCs when the GoCs are inhibiting them), and Act_{Av} represents the average response rate of activity when there is no influence between spike trains (that is, inter-spike time is large enough and the network is in a stationary state).

These measures allowed to obtain a quantitative analysis of the sensitivity rebound (time and strength) and regardless of the variation on the average frequencies at different network layers.

3.4 Results

3.4.1 Coincidence Detection

An exemplar simulation is illustrated in Figure 3.6a. When most of the MF spikes are synchronous, the response rate in the granular layer is high (see Figure 3.6b). However, when the input is time-dispersed, the response rate decreases considerably, with values around 50% using a window of 1ms and around 10% using a window longer than 10ms. This observation suggests that only trains with millisecond precision, and therefore, well structured in time, will be transmitted through the granular layer.

Figure 3.8 shows the response rates simulating different dispersed input trains (from 0 to 100ms) with a reasonable range of synaptic conductance values. The most influential weights were those in the feed-forward inhibitory loop ($MF \rightarrow GoC \rightarrow GrC$) (Figures 3.8b and 3.8e) and in the $MF \rightarrow GrC$ synapse (see Figure 3.8a). A reduced coincidence detection capability was apparent as an increase in granular layer activity (as a consequence of high $MF \rightarrow GrC$, low $MF \rightarrow GoC$, or low $GoC \rightarrow GrC$ synaptic weights), thus input spikes are transferred to GrCs even in spike trains with large time dispersions, whereas an increased GrC response rate is observed when the granular layer activity becomes restricted within a narrow time window (in response to a highly synchronous input spike train).

This feature is a consequence of the time-window process observed in this circuit [67](See below). $MF \rightarrow GrC$ weights markedly influenced the general amount of activity and, when becoming dominant over the $MF \rightarrow GoC$ and $GrC \rightarrow GoC$ pathways, reduced the filtering action of the feed-forward inhibitory loop (Figure 3.8a). An unexpected high activity plateau is shown in Figure 3.8A when using high global peak conductance at $MF \rightarrow GrC$ synapses. This effect can be explained taking into account the following factors:

- When the input activity is reaching the MFs in a scattered way, the GoCs rarely get active, so the GrCs do not receive inhibition and will be very sensitive to the input activity.

- If the $MF \rightarrow GrC$ conductance is strong enough to make GrCs fire with only one or two near-simultaneous MF inputs, then the GrC activity is increased as the total input from all MFs is spread over a longer time window.

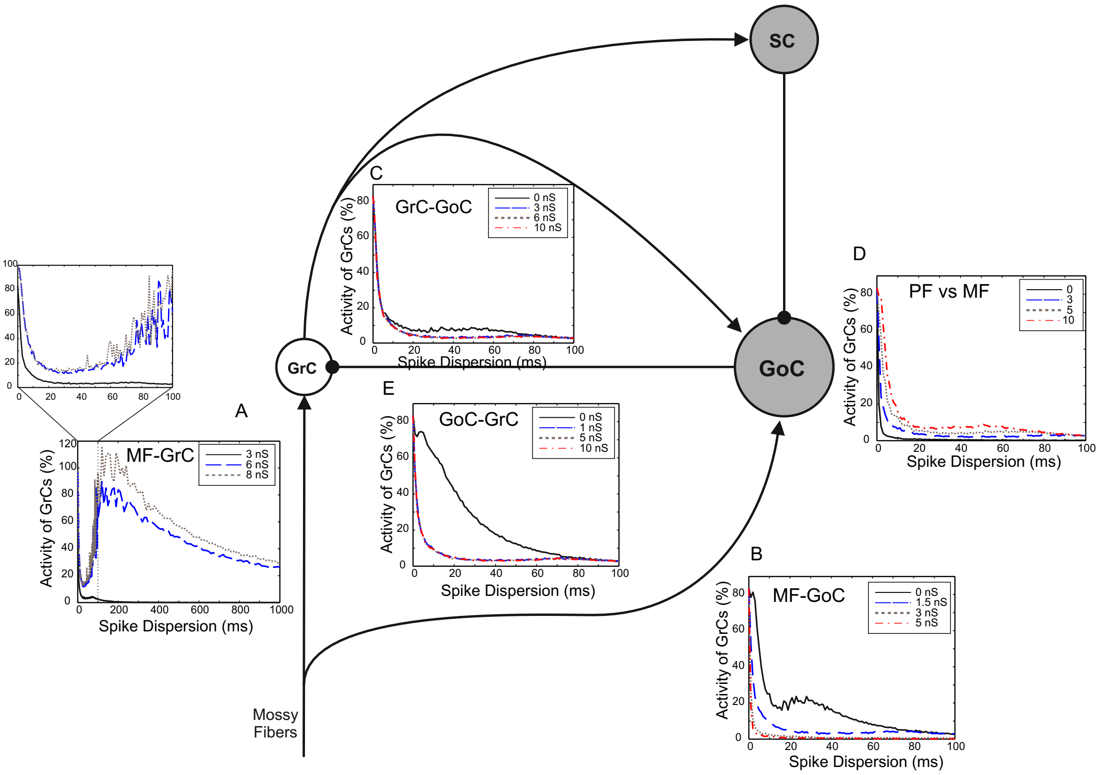


Figure 3.8: Coincidence-detection experiments - The effect of several weights of different synapses on the GrC activity is evaluated. For each synapse site in this network diagram, the obtained GrC activity is represented versus the input spike train dispersion. (a) GrC activity obtained with different $MF \rightarrow GrC$ global peak conductance values (including both AMPA and NMDA receptors). Note the activity enhancement when the input activity in comprised in a time windows of approximately 200ms and the influence of using different conductance values. A detailed simulation including only input activity in time-window lengths from 0 to 100ms has been presented in order to show the coincidence detection capabilities. (b) GrC activity obtained with different $MF \rightarrow GoC$ synaptic weights. Activity is also very sensitive to the weight of this connection. (c) GrC activity obtained with different $GrC \rightarrow GoC$ synaptic weights. No appreciable impact of this synapse conductance on the coincidence detection feature has been observed. (d) GrC activity obtained with different $MF \rightarrow GoC$ and $GrC \rightarrow GoC$ weights. The global peak conductance of both GoC input connections is kept fixed, but the global peak conductance ratio between them (all parallel fibers/all MFs) has been modified. (e) GrC activity obtained with different $GoC \rightarrow GrC$ synaptic weights. Activity is also very sensitive to the weight of these connections.

- When the total input from MFs is spread over long time windows, each GrC can fire more than once after their refractory period.

When the feed-forward inhibitory loop was acting, the feed-back loops did not have a significant effect, regardless of their different weight configurations (Figure 3.8c), probably because the activity through $MF \rightarrow GoC$ synapses had inhibited the GrC activity before GrCs excited GoCs. However, when the $MF \rightarrow GoC$ synapse was disabled (0nS), the $GrC \rightarrow GoC$ synapse weight could regulate coincidence detection, suggesting that the relative balance between MF and parallel-fiber inputs to GoCs is important to control this function. In figure 3.8d, we have plotted the activity of GrCs with respect to different global peak conductance values of the parallel-fiber-activated AMPA receptor in relation to the global peak conductance of the MF-activated AMPA receptor of GoCs. A network with a strong feed-forward loop (lower values in the defined conductance ratio) requires more synchronized MF inputs in order to produce GrC activity.

These characteristics are summarized in Figure 3.9 by means of the T_{20} term defined in Section 3.3.4. Whereas $MF \rightarrow GrC$ synapses get saturated when the global peak conductance goes beyond 3.6nS (Figure 3.9a) (this is, 0.82nS for the AMPA receptor and 0.08nS for the NMDA receptor in each synapse), an increment in the $MF \rightarrow GoC$ weight (Figure 3.9b) leads to lower T_{20} values indicating high input synchrony restrictions for spike trains in order to be transmitted through the granular layer. The $GoC \rightarrow GrC$ connection shows a different behavior where the influence is only evident with extremely low synaptic weights (under 0.8nS). While lower weights nearly disabled coincidence-detection capability, higher ones did not influence this feature (Figure 3.9c). Figure 3.9d represents the influence of the $GrC \rightarrow GoC$ (PFs) connection weight in relation to the $MF \rightarrow GoC$ connection (MFs) with a fixed global peak input conductance in each GoC synapse. Coincidence detection becomes more restrictive (higher input synchrony required for the input to be transmitted) for lower PF vs. MF weight ratios.

3.4.2 Time Slicing through the Permissive Window and Post-Transmission Sensitivity Rebound

The network showed the typical GrC, GoC, SC activation sequence already commented in Figure 3.5a but here, the effect of noise was more evident. The plot in Figure 3.7b

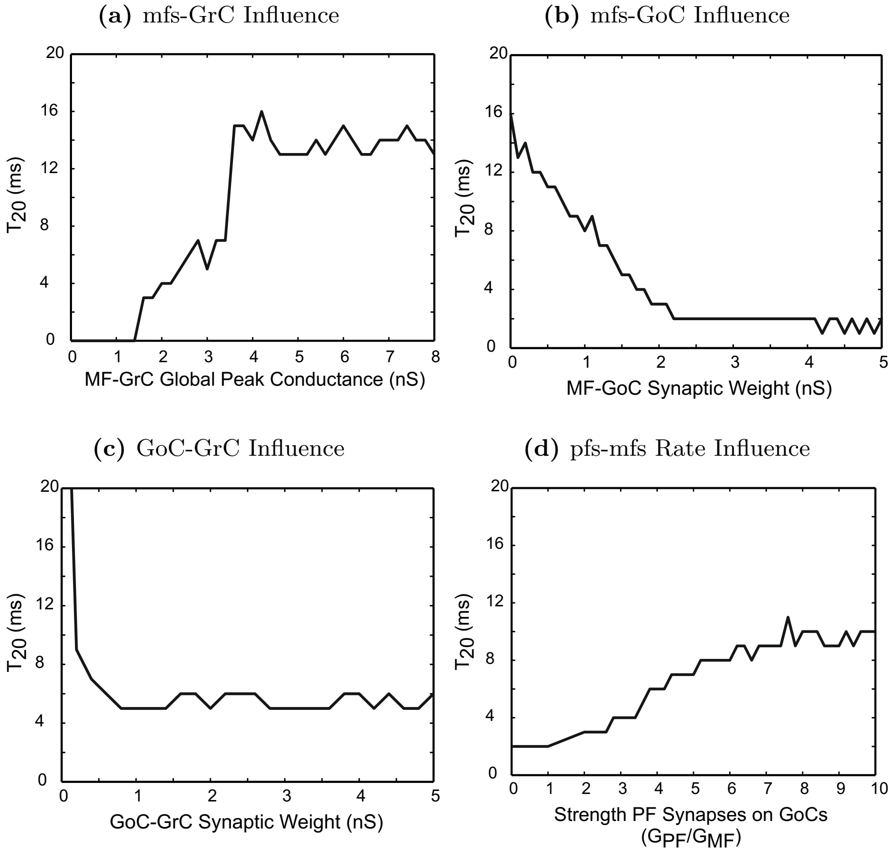


Figure 3.9: Influential connections for coincidence detection - The effect of the weight values on the GrC activity is systematically quantified. For this purpose, the T_{20} value (minimum time-window length which contains the input activity that reduces a GrC output activity under 20% of a reference GrC activity) has been used (the reference activity is defined as the GrC activity obtained when using a completely synchronous input train. See section 3.3.4). (a) T_{20} value using different $MF \rightarrow GrC$ global peak conductance values (including both AMPA and NMDA receptors). (b) T_{20} value using different $MF \rightarrow GoC$ synaptic weights. (c) T_{20} value using different $GoC \rightarrow GrC$ synaptic weights. (d) T_{20} value using different $MF \rightarrow GoC$ and $GrC \rightarrow GoC$ weights. The global peak conductance of all GoC input connections is kept constant and the ratio between the global peak conductance of all parallel fibers and all MFs has been modified.

shows that, in different time intervals, the activity of GrCs in response to the second test train is affected by different factors:

1. If the test spike train is received shortly after (less than 4ms) the conditioning spike train, the GoC and the SC are not yet active, but due to the refractory period, GrCs do not fire more than once. Thus, GrCs are almost inactive.

2. If the interval between the conditioning and the test spike trains is between 4ms and 25ms, the second train arrives during the period that we have called the *activity pause*. During this period, the activity of GoCs leads to a high reduction in the activity of the GrCs (close to 0%). This helps to have *clean* trains transmitted to the parallel fibers avoiding non-desired activity during this interval.

3. If the interval between the conditioning and test spike trains is between 25ms and 55ms, the network sensitivity to the test train is enhanced. In this period, the SCs inhibit the GoCs reducing their background activity. This facilitates the train transmission in the granular layer, leading the response activity to the test train to reach around 35%. This is what we call *sensitivity rebound period* (after the activity pause). It should be noted that, in real life, this mechanism would also depend on intrinsic pacemaking and pause/rebound properties of the GoC [249, 250] as well as from climbing fiber and Purkinje cell activity [19]. Although if input spike trains are strong enough, they will dominate the network behavior.

4. If the interval between the conditioning and the test spike train is between 55ms and 200ms, weaker sensitivity rebounds happen due to the influence of the $MF \rightarrow GrC \rightarrow GoC$ loop.

5. From 200ms on, the conditioning spike train arrival time does not significantly affect the sensitivity of the network to new incoming activity. Two trains with separation intervals above 200ms can thus be considered independent in terms of transmission rates in GrCs.

The $MF \rightarrow GrC$ (Figure 3.10a) and $MF \rightarrow GoC$ (Figure 3.10b) synaptic weights were the most influential ones. A high $MF \rightarrow GrC$ weight produced a higher basal activity, a deeper activity pause, and a larger sensitivity rebound. This clearly reflected the increased activation state of all network components, in particular, of the feed-back

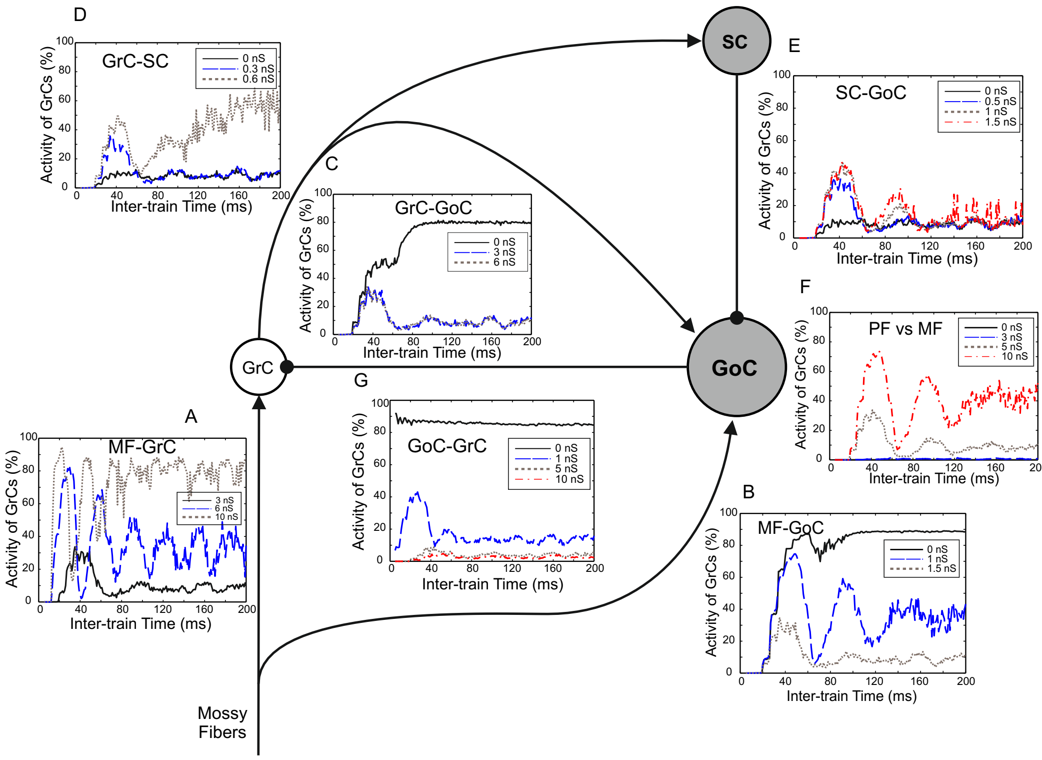


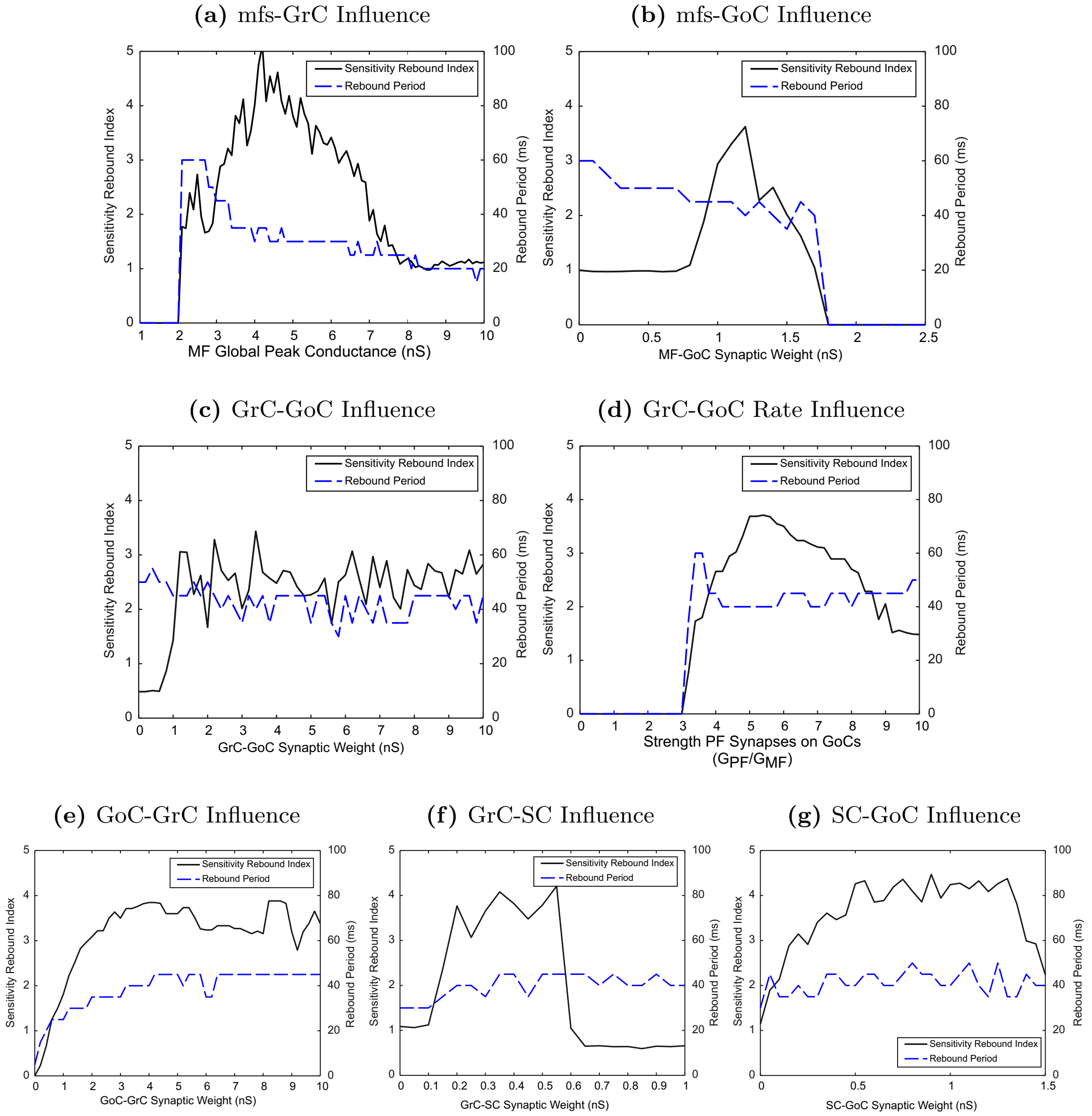
Figure 3.10: Sensitivity rebound curve obtained with different weights - The network diagram shows the effect of different synaptic weights at different sites in the circuit. For each synapse site in this diagram, the obtained GrC activity is represented versus the inter-input-train time. Note that input-activity sensitivity curve is particularly sensitive to weight changes of synapses in the feed-back dis-inhibitory loop ($MF \rightarrow GrC \rightarrow SC \rightarrow GoC$) as well as at the $MF \rightarrow GoC$ synapse. (a) GrC activity obtained with different $MF \rightarrow GrC$ global peak conductance (including both AMPA and NMDA receptors). Note that sensitivity curve is particularly sensitive to weight changes at this synapse. (b) GrC activity obtained with different $MF \rightarrow GoC$ synaptic weights. Activity is also very sensitive to the weight values of this connection. (c) GrC activity obtained with different $GrC \rightarrow GoC$ synaptic weights. No detectable sensitivity to this synaptic weights is evident (unless the connection is completely disabled, i.e. 0 nS). (d) GrC activity obtained with different $GrC \rightarrow SC$ synaptic weights. Activity is very sensitive to weight values of this connection and the network activity could become unstable using too high weights (0.6nS). (e) GrC activity obtained with different $SC \rightarrow GoC$ synaptic weights. Activity is especially sensitive to the lack of this connection. (f) GrC activity obtained with different $MF \rightarrow GoC$ and $GrC \rightarrow GoC$ weights. The global peak conductance of all GoC input connections is kept constant and the ratio between the global peak conductance of all parallel fibers and all MFs is modified. (g) GrC activity obtained with different $GoC \rightarrow GrC$ synaptic weights. Activity is also very sensitive to weight values of this connection.

loops passing through the GoC (affecting the activity pause) and the SC (affecting rebound excitation). Figure 3.11a clearly shows the effect of $MF \rightarrow GrC$ synaptic weights in the sensitivity rebound parameters presented above. The rebound period decreases as the global peak conductance rises (from near 60ms when the global peak conductance is 1nS to 20ms when the global peak conductance is higher than 8nS). However, the Sensitivity Rebound Index (SRI) shows that the sensitivity rebound regarding amplitude is higher when the global peak conductance is around 4nS. Higher synaptic weights saturate the GrCs and increase the average activity of this layer. On the other hand, lower synaptic weights cause very low activity in the GrCs, which implies low activity at both feed-back inhibitory and dis-inhibitory loops.

A low $MF \rightarrow GoC$ weight increased the sensitivity rebound (the maximum activity rate) and delayed the rebound peak time (Figures 6B and 7B). In these conditions, the only remaining (though less significant) inhibition is that produced by $GrC \rightarrow GoC$ activity. It should also be noted that low weights in these connections produced instability in granular layer activity, because the SCs inhibit the GoCs and noise reaches the parallel fibers (cf. Figure 3.8b) preventing an efficient noise filtering. On the other hand, an extremely high $MF \rightarrow GoC$ weight caused an inhibition in GrCs which prevented these cells from firing (and obviously, the sensitivity rebound). Medium values (around 1.2nS) obtained higher sensitivity rebound indexes (Figure 3.11b).

As well as with $MF \rightarrow GoC$ weights, small $GrC \rightarrow GoC$ weights enhanced rebound activity (Figure 3.10c), but also in this case, the response became unstable because of the propagation of basal noise in the network. However, different values of this weight did not considerably modify the sensitivity rebound index or the rebound period (Figure 3.11c). Finally, if we maintain a constant global peak conductance of GoC input (parallel-fiber plus MF conductance), the ratio between parallel fiber global peak conductance and MF global peak conductance played an important role in terms of sensitivity rebound (Figures 3.10f and 3.11d). When the parallel fiber peak conductance was 4 times higher than MF peak conductance, the best performance in terms of sensitivity rebound index was observed.

The influence of $GoC \rightarrow GrC$ and $MF \rightarrow GrC$ connections is similar, since both of them adjust the average level of activity (Figures 3.10g and 3.11e). However, they affect the GrC activity in an opposite manner: obviously, low weights in $GoC \rightarrow GrC$


(a) mfs-GrC Influence

(b) mfs-GoC Influence

(c) GrC-GoC Influence

(d) GrC-GoC Rate Influence

(e) GoC-GrC Influence

(f) GrC-SC Influence

(g) SC-GoC Influence

Figure 3.11: Quantitative analysis of sensitivity rebound - The effect of the weight values on the input-activity sensitivity rebound is systematically quantified. For this purpose, the sensitivity rebound index (left axis) and the rebound period (right axis) (see Methods section) are evaluated for each of the following weight modifications: (a) $MF \rightarrow GrC$ global peak conductance (including both AMPA and NMDA receptor). (b) $MF \rightarrow GoC$ synaptic weights. (c) $GrC \rightarrow GoC$ weights. Note the lack of dependence of both measures on the weight values. Only the lack of this connection markedly reduces the sensitivity rebound index. (d) $MF \rightarrow GoC$ and $GrC \rightarrow GoC$ weights. The global peak conductance of all GoC input connections is kept constant and the global peak conductance ratio (between all parallel fibers and MFs) is modified. (e) $GoC \rightarrow GrC$ weights. (f) $GrC \rightarrow SC$ weights. (g) $SC \rightarrow GoC$ weights.

generate high activity levels in GrC, whereas low weights in $MF \rightarrow GrC$ generate low activity levels in GrCs.

The $GrC \rightarrow SC$ synapses are important for the sensitivity rebound (Figure 3.10d), in the sense that when synaptic weights were set to 0nS (no connection), the activity pause ended smoothly in around 60ms (without sensitivity enhancement). However, too high weights in this connection led to an unstable state when the GoC remained constantly inhibited due to the strong excitation of the SCs. This is corroborated in Figure 3.11f, which shows that medium weights enhanced the sensitivity rebound and lightly increased the rebound period. Similar conclusions can be extracted from the study of the $SC \rightarrow GoC$ connection (Figure 3.10e and 3.11g), which is also important for the sensitivity rebound. Medium synaptic weights produced large sensitivity rebounds but extremely high weights led to instability in the granular layer.

The synchronization of the network activity was facilitated by the partial overlapping of MF and parallel fiber innervation areas (see Figure 3.5b). A further mechanism that would enhance synchronization is electrical coupling. This was reported among SCs in the guinea pig's cerebellum [189] and among GoCs in the rat's cerebellum [86]. In the presented experiments, the whole network achieved a nearly perfect input-driven synchronism, so the electrical coupling (EC) among GoCs and among SCs does not seem to influence the sensitivity rebound (Figures 3.12a and 3.12b). In our experiments, the activity of the GoCs and SCs was already highly synchronous due to GrC outputs and the GoCs feed-back to GrCs, corroborating in this way previous works about granular layer synchronism [185].

3.4.3 Coincidence Detection and Time-Slicing Co-Optimization

Figure 3.13 summarizes the general architecture and main weights involved in coincidence detection and time-slicing. The inhibitory loop ($MF \rightarrow GoC \rightarrow GrC$) plays a crucial role for coincidence detection. At the same time, the excitatory loop ($MF \rightarrow GrC \rightarrow SC$) plays an important role for amplitude and temporal aspects of time slicing (Figures 3.10 and 3.11). In summary:

1. The main MF input pathway. A range of values (global peak conductance between 2 and 3.5nS) at the $MF \rightarrow GrC$ synapse is compatible with both coincidence detection and time slicing provided that saturation of GrC activity and too low

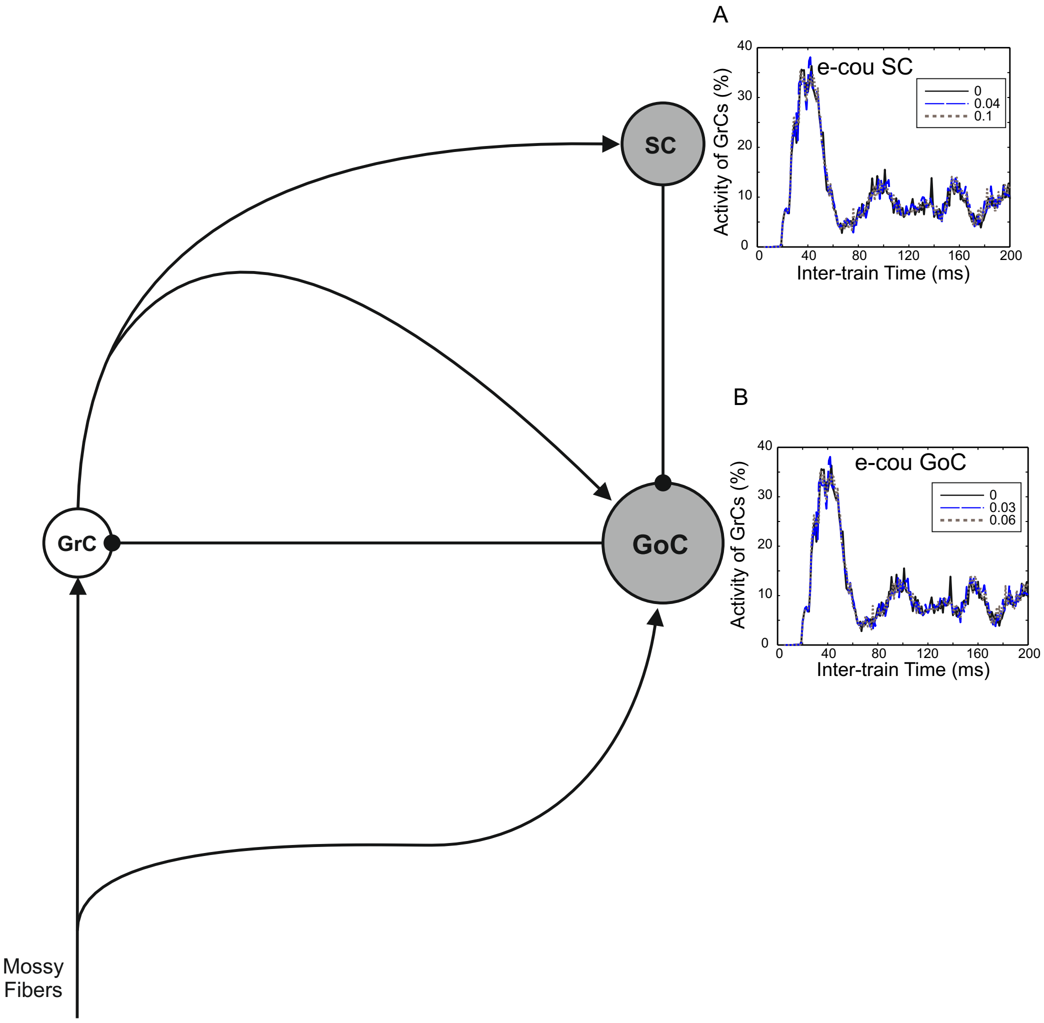


Figure 3.12: Sensitivity rebound curve obtained with different electrical coupling coefficients - The effect of several electrical-coupling coefficients among SCs and GoCs is evaluated. For each coupling site in this network diagram, the obtained GrC activity is represented versus the inter-input-train time. (a) Electrical coupling between SCs. (b) Electrical coupling between GoCs.

activity of GrCs is avoided (see Figures 3.9a and 3.11a). This may be actually achieved through long-term synaptic plasticity in the form of LTP and LTD revealed experimentally at the $MF \rightarrow GrC$ relay [14, 96, 247].

2. The feed-forward inhibitory loop. High weights in the $MF \rightarrow GoC$ (2-5nS) and $GoC \rightarrow GrC$ (0.5 to 5nS) connection facilitate coincidence detection (Figures 3.9b and 3.9c). On the other hand, lower synaptic weights in $MF \rightarrow GoC$ (0.75 to 1.75nS) and $GoC \rightarrow GrC$ (2 to 10nS) are essential for generating effective sensitivity rebound (Figures 7B and 7E). Although there is a trade-off between coincidence detection and time slicing, moderately-low weights allow both properties to co-exist.

3. The feed-back inhibitory loop. The $GrC \rightarrow GoC$ connection, or parallel fiber input to GoCs, allows a large range of weights (1 to 10nS) without substantially affecting coincidence detection and time slicing (Figures 3.8c and 3.11c). Apparently, the feed-back inhibitory loop is poorly effective in this configuration; due to its delay, it operates when the feed-forward pathway has already caused intense inhibition.

4. The feed-back dis-inhibitory loop. The $GrC \rightarrow SC$ and $SC \rightarrow GoC$ connections control the sensitivity rebound and are only important for time slicing (Figures 3.11f and 3.11g). Over a range of weight values ($GrC \rightarrow SC$ 0.2 to 0.6nS and $SC \rightarrow GoC$ 0.2 to 1.5nS), the feed-back dis-inhibitory loop allows effective coincidence detection. This loop, as well as the $MF \rightarrow GrC$ synapse, may therefore undergo long-term synaptic plasticity without compromising other temporal aspects of network activity.

3.4.4 Oscillations during Protracted Stimulation

In previous works, such as [185], the granular layer was proposed as a completely synchronous system where both GrCs and GoCs showed periodic and correlated activity. This behavior has been reproduced using our network model. In that previous work, different factors were proven to be very influential in order to achieve activity oscillations:

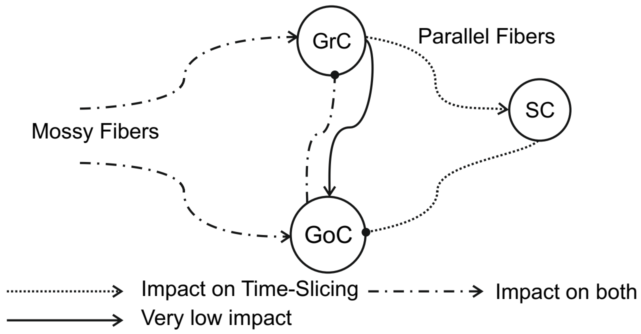


Figure 3.13: Network connection diagram and main weights involved in coincidence detection and sensitivity rebound - The most influential connection weights in coincidence detection ($MF \rightarrow GoC \rightarrow GrC$) and the connections which are significant to support the time-slicing capability ($GrC \rightarrow SC \rightarrow GoC \rightarrow GrC$) are indicated.

(i) high weights in $GoC \rightarrow GrC$ connections, (ii) medium weights in $GrC \rightarrow GoC$ connections, (iii) strong parallel fiber connections to GoC in relation to MF connections, and (iv) high MF firing rates.

If the input is repeated or maintained, the activity pauses (caused by the feedback inhibitory loop $MF \rightarrow GrC \rightarrow GoC \rightarrow GrC$) may lead to oscillations at a certain preferred resonant frequency whose period corresponds approximately to the sum of the permissive time window and the post-transmission silent-pause: this is what we call resonance working hypothesis (Figures 3.7b). The resonant period should be compatible with the reaction time of the circuit, which is determined by the cell time constants (Table 3.1: GrC 10ms, GoC 20ms, SC 16ms) and by the inhibitory synaptic time constant (Table 3.1: GoC-GrC 10ms). Rebound excitation peaks occur at 40ms and are up to 55ms approximately, accounting for frequencies in the β/γ range (Figure 3.11).

In order to prove the resonance working hypothesis, we have carried out a set of simulations in which the network was driven by random MF activity at different frequencies from 1Hz to 120Hz per cell. Following this stimulation through the MFs, the network activity showed coherent oscillations. These oscillations have never been easily measured in single GrCs because of their sparse activity, but clearly emerged as a population activity by means of histograms (Figure 3.14a). These oscillations were sustained by the $GrC \rightarrow GoC \rightarrow GrC$ feed-back loop. However, some synaptic weights

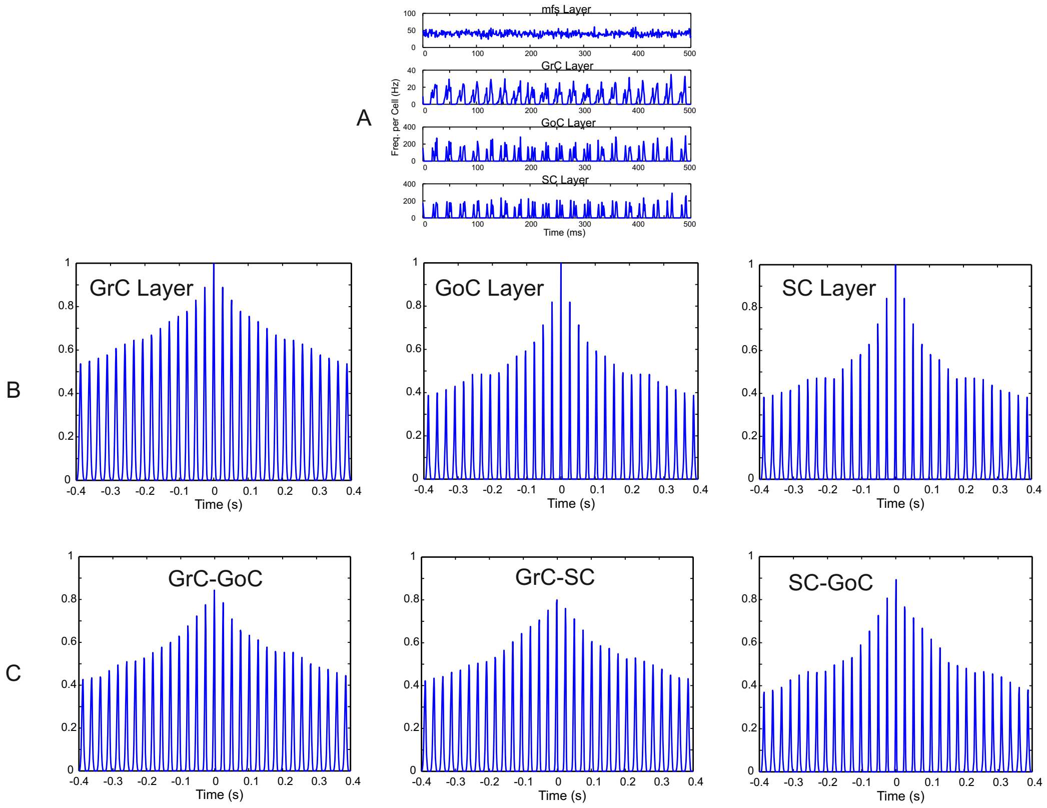


Figure 3.14: Synchronous activity oscillation during continuous stimulation - (a) Activity histogram of MF, GrC, GoC, and SC during a maintained random stimulus with an average frequency of 40 Hz. (b) Autocorrelograms of GrC, GoC, and SC activity showing coherent oscillations in population activity. (c) Cross-correlograms between GrC, GoC, and SC responses revealing the driving influence of the GrC activity on the other cortical neurons.

have been adjusted in order to accomplish those factors indicated in previous works as good candidates to enhance activity oscillations:

1. The $MF \rightarrow GrC$ global peak conductance was increased to 6.04nS, and the relation between NMDA and AMPA transmitter conductance was set to 1.61 [185]. Whereas in sensitivity rebound experiments synchronized trains prevailed over 20Hz input activity (which was considered basal activity), in resonance working hypothesis experiments, 20Hz activity (or higher-frequency activities) was the predominant activity (due to the lack of more synchronous trains). Because of that, a sensitivity enhancement was needed to produce activity oscillations.

2. The relation between the $GrC \rightarrow GoC$ and the $MF \rightarrow GoC$ global peak conductance was increased to 100. In this way, the feed-back loop ($MF \rightarrow GrC \rightarrow GoC \rightarrow GrC$) has been strengthened.

Network stimulations were run with different input frequencies in order to quantitatively evaluate the emerging oscillations. Autocorrelograms in Figure 3.14b show the high correlation index in all cell types in the granular layer (GrCs, GoCs, and SCs). Moreover, different cell types correlate each other as shown in cross-correlograms in Figure 3.14c.

For inputs with an average frequency ranging from 1Hz to 120Hz, the spectrogram analysis revealed output oscillations ranging from about 30Hz to 43Hz (see Figure 3.15a). Another oscillation term emerges from this spectrogram from 65Hz to 90Hz. This represents harmonic resonant frequencies. A more detailed spectrogram can be observed in Figure 3.15b, where only 20Hz, 40Hz, and 80Hz-input-activity experiments have been shown. Both frequency peaks (fundamental and second harmonic) are shown for each input frequency.

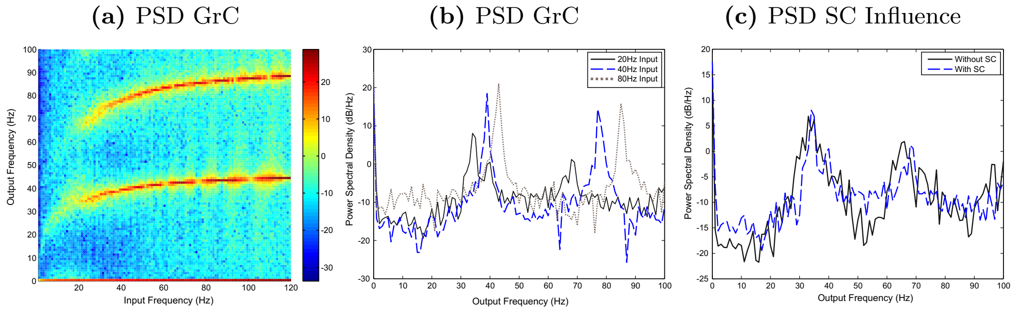

(a) PSD GrC **(b)** PSD GrC **(c)** PSD SC Influence

Figure 3.15: Frequency analysis of activity during continuous stimulation - (a) Power Spectogram Density (PSD) of GrC activity using different input frequencies (X-axis). Color code represents the power spectral density (dB/Hz). Note that the lowest resonance band (from 30 to 43Hz) shows the resonance fundamental frequency and the highest (from 60 to 86Hz) corresponds to the second harmonic for each input frequency. (b) Frequency-domain graph using only 20Hz, 40Hz and 80Hz input. Note the different resonance frequencies (peak values). (c) Frequency-domain graph of two different simulations using 20Hz random input and enabling/disabling (0nS synaptic weights) the $SC \rightarrow GoC$ connection.

Finally, Figure 3.15c shows the influence of SCs in GrC oscillations when the network is stimulated with 20Hz random inputs. Although it is not very remarkable, when using SCs, the resonance frequencies are slightly higher (from 32Hz without to 33Hz with SC). This is caused by the inhibition of GoCs, which makes GrCs fire before the activity pause. However, this factor is nearly negligible in relation with those explored by [185]. In fact, this influence becomes insignificant when the network is simulated with higher-frequency inputs.

3.5 Discussion

This chapter shows how synaptic weights at multiple synaptic connections may affect temporal signal processing in the cerebellar granular layer. This effect on the processing was demonstrated on a small network scale (4500 GrCs, 27 GoCs, and 300 SCs) and was visible on the millisecond time scale. Synaptic weights in specific network loops influenced coincidence detection and time slicing regulating the efficiency of temporal noise filtering and temporal separation of transmitted spike trains. Due to the specific time-constants in these loops, the circuit developed β/γ oscillations in response to continuous inputs.

Coincidence detection is one of the supposed functions of the granular layer [192] and has recently received experimental support [61, 148]. Our results showed that coincidence detection was critically regulated by the $GoC \rightarrow GrC$ inhibitory connection. When the network was deprived of GoCs, the coincidence detection capability vanished. Conversely, high synaptic weights in the feed-forward inhibitory loop ($MF \rightarrow GoC \rightarrow GrC$) enhanced coincidence detection restricting GrC activity to a permissive time window with a duration of a few milliseconds. The feed-forward inhibitory loop was dominant with respect to the feed-back inhibitory loop, whose effect was subordinated to extremely low $MF \rightarrow GoC$ weights. This can be explained by the delay in the action of the feed-back inhibitory loop, which operated after feed-forward inhibition had already occurred; thus, parallel-fiber connections and SCs do not seem to play a relevant role in controlling coincidence. It should also be noted that electrical coupling between GoCs or SCs had almost no impact on coincidence detection, although it may be important for low-frequency granular layer activity synchronization [86]. The $MF \rightarrow GrC$ weights were crucial for coincidence detection, in that the network required more-synchronized input trains when low values of these weights were used.

Time slicing in the granular layer network has been predicted on the basis of various experimental data [59] and has been recently supported by the observation of the effects of feed-forward inhibition in vitro [150, 190] and in vivo [130]. Our results indicated that time slicing was regulated by connection weights at granular layer synapses. The $MF \rightarrow GrC \rightarrow SC \rightarrow GoC$ dis-inhibitory loop proved to be the most important one by affecting both the sensitivity rebound amplitude and the inter-train interval which maximizes network activity. Clearly, the action of this feed-forward loop depended on the GoC activity caused by the feed-forward inhibitory loop; therefore, the two loops were not totally independent. It should be noted that both GoC electrical coupling (reported experimentally [86] and SC electrical coupling (observed in the molecular layer in [189]) were ineffective in enhancing the time slicing observed in our simulations. However, it could become effective if GoC and SC were less synchronized by the GrC input.

Although the dynamics of granular layer excitation were consistent with those of previous models [185, 251], by implementing the SC loop, we have supported a theoretical prediction suggesting a new relevant property of the system [59]. Once certain activity enters into the granular layer, it coactivates the GrC and the GoC. Then, the inhibitory feed-forward loop can limit the GrC activity within a short time window. Signals passing though this time window are sent back to the GoC through the parallel fibers, providing the basis for sustained β/γ oscillations in the presence of continuous inputs [274]. Interestingly, the SC loop blocks the GoC soon thereafter enhancing GrC responsiveness over the resting level maintained by background activity in the $MF \rightarrow GoC \rightarrow GrC$ loop. It should also be noted that rebound excitability occurred about 20ms after stimulation, a time at which electrophysiological data show that GoCs generate a secondary peak response [273]. Therefore, dis-inhibition would not prevent the GoC from making a second spike, but would rather modulate its probability and precise time of occurrence.

Oscillations of electrical activity have been observed in the granular layer of the cerebellum in vivo and in vitro [57, 194, 199, 263]. The emergence of high-frequency β/γ oscillations in our simulations (Figures 11A and 11B) resembles results obtained in a previous simulation study of the granular layer network [185]. In actual cells, this mechanism would be further influenced by intrinsic excitable properties of the GrCs [64] and GoCs [249, 250]. Moreover, a second preferential oscillation/resonance regime may occur in the theta band [116, 166, 215] under the influence of input patterns and intrinsic neuron properties [251] which have not been considered here. However, it should be taken into account that some of these groups were recorded their data from different animal species (rats and monkeys), which could explain the discrepancy in frequency.

Regulation of synaptic weights along the main MF input pathway ($MF \rightarrow GrC$) was compatible with both coincidence detection and time slicing, and regulation of synaptic weights in the feed-back dis-inhibitory loop ($GrC \rightarrow SC \rightarrow GoC$) was mostly relevant for time slicing. Thus, the $MF \rightarrow GrC$ and $GrC \rightarrow SC \rightarrow GoC$ synapses could be primary sites of plasticity calling for subsequent adaptation in the rest of the circuit. This adaptation could lead to different working modes depending on the kind of input activity (synchronized trains over basal activity getting sensitivity rebounds or only random activity to generate output oscillations). Conversely, the feed-forward inhibitory loop showed a conflicting regulation of coincidence detection and time slicing and is therefore suited to counter-balance primary changes occurring along the main input pathway and the feed-back dis-inhibitory loop. A balance between plasticity towards enhacing certain processing properties and homeostasis was envisaged in a previous model, in which weight changes at the $MF \rightarrow GrC$ synapse enhanced information transfer but had to be counter balanced by changes at the $GoC \rightarrow GrC$ synapse to preserve sparseness [237]. Finally, the feed-back inhibitory loop was poorly effective in coincidence detection or time slicing and potential forms of plasticity in this loop may be involved in fine-tuning the basal activity state of the GoC [234]. Therefore, learning rules in actual cells may be adapted to satisfy these different network requirements opening an issue for future biological research.

MF-GrC long-term synaptic plasticity (LTP and LTD) can be readily generated by specific MF patterns reflecting the rate and duration of afferent bursts [14, 63, 96]. However, the same patterns do not seem to be effective for generating LTP or LTD at the $MF \rightarrow GoC$ synapse (Cesana and D'Angelo, unpublished results), which may therefore take part in different processes of network regulation. Likewise, at the $GoC \rightarrow GrC$ synapse, intense bursts induce a reduction of GrCs input resistance by down-regulating the inward rectifier channels, suggesting a form of homeostatic control [231]. No extensive studies have been performed so far at other synapses in the circuit (e.g. see [114]), except for a preliminary report showing LTD at the $GrC \rightarrow GoC$ synapse

[225]. Therefore, at present, MF-GrC long-term synaptic plasticity (LTP and LTD) appears as the critical regulatory element in the whole circuit, which may be assisted by various other forms of plasticity providing the homeostatic balance of the network. Future experimental investigations will be needed to substantiate this hypothesis.

In previous models [185], some proposed functionalities and features such as the oscillations of electrical activity were already demonstrated to happen at the cerebellum. In the presented work, more detailed neuron models have been included and the network topology has been refined including the SC loop which seems to highly influence the temporal capabilities in the granular layer (but not in the oscillatory state). Thus, some other expected functionalities of the granular layer (coincidence detection and time slicing) have been observed.

Other detailed neuron and network models have been recently developed and studied [251]. In these models, several spatio-temporal properties which have been observed (center-surround organization, time-windowing, high-pass filtering in response to bursts of spikes, and coherent oscillations in response to input random activity) corroborate the initial hypothesis about granular layer processing capabilities. Those spatio-temporal properties complement the proposed ones on this work.

Our experiments do not corroborate the hypothesis of the cerebellum as a liquid state machine [196, 288] which were based on statistical properties of the granular layer connectivity. However, those hypotheses might be able to coexist with the one proposed in this work as long as spatio-temporal properties of the granular layer could improve the generation of different states (variable sets of GrCs) at parallel fibers based on having different input stimuli reaching the MFs.

Essential aspects of the system have been captured by the studied network model. In this way, permissive GrC time windows compare well to those measured experimentally (4-5 ms in real life [150, 190]). Accordingly, the characteristic period of network response to persistent random stimuli was in the β/γ frequency range, as predicted by [185]. Nonetheless, the potential effect of some properties of other neural models on the granular layer behavior (coincidence detection and time slicing and activity states induced by patterned theta-frequency inputs) should be assessed in the future [67, 75]. For instance, several voltage-dependent currents are known to enhance bursting while others can regulate the pattern of discharge during complex input patterns and lead to forms of cellular oscillation and resonance [64, 187, 249, 250]. In particular, spike frequency adaptation in GoCs may allow GrCs to restore a certain excitability favoring long-burst transmission (e.g. like those seen in vivo: [46, 148, 223]), while specific channels could enhance post-inhibitory rebounds and activity pauses [250]. Spontaneous beating in GoC could increase the inhibitory background state and post-inhibitory rebound excitation could favor the sensitivity rebound period. The neurotransmitter spillover can protract GABA-A, AMPA, and NMDA currents maintaining synaptic response and regulating temporal integration of impulses. Finally, local regulation of weights in extended networks could create differential processing in neighboring regions [191, 251].

An additional capability of our system should be taken into account. EDLUT-implemented models are able to run in real-time [227]. In this way, all these temporal properties of the granular layer could be directly implemented in a whole functional system to execute advanced control tasks (or any other tasks involving the cerebellum) in the framework of neurobotics experiments.

In conclusion, this simulation suggests that coincidence detection and time slicing, two major features of granular layer function, are markedly and differentially sensitive to local synaptic weights prompting to experimentally search for long-term plasticity at various synapses in the circuit. Thus, the cerebellum granular layer, possibly including SCs, appears to behave as a complex adaptable filter providing the basis for spike timing and pattern detection in Purkinje cells [59, 79]. It is also possible that β/γ-band dynamics revealed in vitro and in vivo [57, 110, 199] are generated, at least in part, in the granular layer. To further investigate these aspects, networks with spike-timing-dependent plasticity optimizing information transfer [23] and neurons with voltage-dependent dynamics [40, 208, 251] will be employed using the studied model as the fundamental building block for a whole EDLUT cerebellar simulator.

4

Using the cerebellum for correcting in manipulation tasks

4.1 Abstract

This chapter presents how a simple cerebellum-like architecture can infer corrective models in the framework of a control task when manipulating objects that significantly affect the dynamic model of the system. The main motivation of this work is to evaluate a simplified bio-mimetic approach in the framework of a manipulation task. More concretely, this work focuses on how the model inference process takes place within a feedforward control loop based on the cerebellar structure and on how these internal models are built up by means of biologically plausible synaptic adaptation mechanisms. This kind of investigations may provide clues about how biology achieves accurate control of non-stiff-joint robot with low-power actuators which involve controlling systems with high inertial components. This work also studies how a basic temporal-correlation kernel including specific long-term depression (LTD) and a non-specific manteined long-term potentiation (LTP) at parallel fiber-Purkinje cell synapses can effectively infer corrective models. We evaluate how this spike-timing-dependent plasticity correlates sensorimotor activity arriving through the parallel fibers with teaching signals (dependent on error estimates) arriving through the climbing fibers from the inferior olive. Therefore this chapter addresses the study of how these LTD and LTP components need to be well balanced with each other to achieve accurate learning. This is of interest to evaluate the relevant role of homeostatic mechanisms in biological systems

where adaptation occurs in a distributed manner. Furthermore, we illustrate how the temporal-correlation kernel can also work in the presence of transmission delays in sensorimotor pathways. We use a cerebellum-like spiking neural network which stores the corrective models as well-structured weight patterns distributed among the parallel fibers to Purkinje cell connections.

In addition to this, we have evaluated a way in which a cerebellar-like structure can store a model in the granular and molecular layers. Furthermore, we study how its microstructure and input representations (context labels and sensorimotor signals) can efficiently support model abstraction towards delivering accurate corrective torque values for increasing precision during different-object manipulation. We also describe how the explicit (object-related input labels) and implicit state input representations (sensorimotor signals) complement each other to better handle different models and allow interpolation between two already stored models. This facilitates accurate corrections during manipulations of new objects taking advantage of already stored models.

4.2 Introduction: The problem of manipulation tasks

Controlling fast non-stiff-joint robots accurately with low power actuators is a difficult task which involves high inertia. Biological systems are, in fact, non-stiff-joint *plants* driven with relatively low-power actuators. However, in this case, control schemes require building accurate kinematic and dynamic models (dynamic models would not be required in the case of very-stiff-joint robots with inappreciable inertia). Even if the basic dynamic model is very accurate, manipulating tools and objects will affect this base model. This will lead to significant distortions along the desired movements, affecting the final accuracy. Therefore, these systems require adaptive modules for tuning the corrective models to specific object or tool manipulation.

This challenge has been smartly solved by the biological systems by using the cerebellum as a force, stiffness, and timing control machine in every human movement. The cerebellar cortex performs a broad role in different key cognitive functions [170]. Three different layers constitute the cerebellar cortex: the granular layer, the molecular layer, and finally, the Purkinje layer. The cerebellar cortex seems to be well structured into microzones [79] related to a specific somatotopic organization in sensor and actuator areas. The human cerebellum involves about 10 million Purkinje cells receiving excitatory inputs from parallel fibers (150,000 excitatory synapses at each Purkinje cell). Each parallel fiber synapses on about 200 Purkinje cells; these parallel fibers are granule cell axons. These granule cells are excited by mossy fibers (with afferent connections from the spinal cord, with sensory and motor estimates). Each Purkinje cell receives further excitatory synapses from one single climbing fiber. This connection is so strong that the activity from a single climbing fiber can drive the Purkinje cell to fire [149]. These spikes from the Purkinje cells generated by climbing fibers are called complex spikes, while the Purkinje cell spikes generated by the activity received from the parallel fibers are called simple spikes. Basket cells and stellate cells, being activated by parallel fiber afferents, can inhibit Purkinje cells. Finally, Golgi cells receive excitatory input from parallel fibers, mossy fibers, and climbing fibers, and inhibit granule cells. The output of a Purkinje cell is an inhibitory signal to the deep cerebellar nuclei [149](Fig. 4.1). Granule cells and Purkinje cells play an important role in pattern recognition [216]. We can assume that the granular layer adaptation mechanism is essentially unsupervised [238] towards enhancing information transmission. In this layer, an efficient recoding of mossy fiber activity takes place improving the learning capability in subsequent stages (granular cell-Purkinje cell synapse). The cerebellum seems to play a crucial role in model inference within manipulation tasks but the way this is supported by actual network topologies, cells, and adaptation properties is an open issue.

We have addressed the study of how this model inference task can be achieved in a local and distributed manner within a basic cerebellum-like architecture based on spiking neurons. Furthermore, we evaluate how spike-timing-dependent plasticity (STDP) provides an efficient learning rule for this task. We do this by using a simple specific temporal-correlation kernel (LTD) and a non-specific maintained compensating LTP as the adaptation mechanism at the parallel fiber (PF)-Purkinje cell (PC) synapses. We explore how the LTD and LTP components of this learning rule need to be well balanced to achieve an acceptable performance. Although different systems that potentially compensate transmission delays have been proposed [193, 254], in this chapter, we explicitly avoid compensating them. The correlation kernel is able to correlate sensorimotor activity with error estimates without explicitly taking into account the transmission delays. This inferred model is therefore trajectory specific. By means of a certain correlation kernel the effect of several input spikes on plasticity is accumulated in a reduced number of variables, without the necessity of storing spike

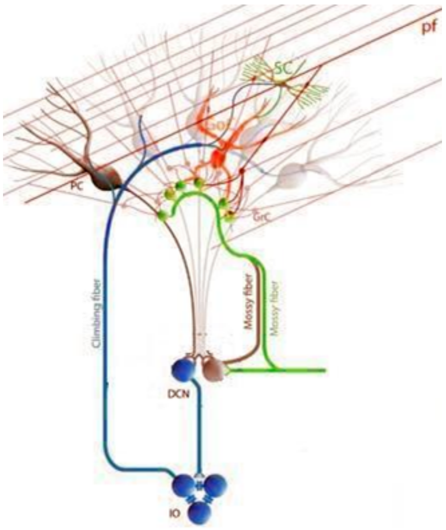


Figure 4.1: Scheme of the cerebellum organization - This scheme shows the most relevant connections within a cerebellar module. The cerebellar module presents different connections communicating different circuit elements in closed loops. Mossy fibers contact granule cells (GrC) and DCN cells which, in turn, receive inhibition from the same common set of Purkinje cells (PC). Moreover, the IO cells project climbing fibers that contact PC which also are projected to DCN cells [67].

times. This makes this correlation kernel computationally efficient for event-driven processing engines, as the one used in this chapter, EDLUT [227]. In this work, we explicitly evaluate how these corrective models are structured in a distributed manner among different synapses in the PF-PC connection space. The possibility of monitoring this spatio-temporal learned weight pattern represents a powerful tool to interpret how models are inferred to enhance the accuracy in a control task. We evaluate how this learning engine with non-specific (fixed gain) LTP and correlation-based LTD components can infer different corrective dynamic models corresponding to the manipulation of objects of different masses.

In this chapter, we describe how a spiking neural network mimicking a cerebellar micro-structure allows an internal corrective model abstraction. By adopting a cerebellar-like network, we explore how different sensor representations can be efficiently used for a corrective model abstraction corresponding to different manipulated objects. When a new object is manipulated and the system detects that significant trajectory errors are being obtained, the abstracted internal model adapts itself to match the new model (kinematic and dynamic modifications of a base arm plant model). The stored models to be used can be selected by explicit object-related input signals (as specific input patterns generated for instance from the visual sensory pathway) or implicit signals (such as a haptic feedback). This can be seen as a *cognitive engine* that abstracts the inherent object features through perception-action loops and relates them with other incidental properties such as color, shape, etc. The cognition process that relates both properties is important because it allows the inference of inherent properties just by activating explicit perceived primitives making possible to build up models of the environment that describe how it will *react* when interacting with it. To better illustrate this issue, we have simplified the cerebellum architecture. Through this simple cerebellar structure, we have monitored how the weight's space adapts to a distributed stable model that depends on the basic network topology, the target trajectory, and model deviations.

Control schemes of biological systems must cope with significant sensorimotor delays (100 ms approximately) [40, 224, 271]. Furthermore, actuators are very efficient but have a limited power and have to deal with viscoelastic elements. In order to deal with all these issues, biology has evolved efficient *model inference engines* to facilitate adaptive and accurate control of arms and hands [132, 153]. A wide range of studies have proven the crucial role of the cerebellum in delivering accurate correcting motor actions to achieve high precision movements even when manipulating tools or objects (whose mass or moment of inertia significantly affects the base dynamic models of the arm-hand) [160, 197]. For this purpose, the cerebellum structure needs to infer the dynamics model of the tool or object under manipulation [284] and store it in a structured way that allows an efficient retrieval of corrective actions when manipulating this item. There are scientific evidences of synaptic plasticity at different sites of the cerebellum and the sensorimotor pathway. The synaptic connection between PFs and PCs seems to have a significant impact on the role of inferring models of sensorimotor correlations for delivering accurate corrective commands during control tasks in most cerebellar models [9, 77, 137]. Furthermore, the adaptation at this site seems to be driven by the activity coming from the inferior olive (IO) and by the way this activity correlates with the activity received through the PFs. Within a cerebellar-like cell-based structure, the corrective model is inferred in a distributed way among synapses.

Furthermore, this scheme based on distributed cell populations allows several models to be inferred in a non destructive way by selecting a specific population each time.

The IO is an important paracerebellar center whose functional role is still an open issue [67, 74, 141, 142, 195, 238]. Different research groups have studied its potential role in delivering a teaching signal during accurate movements [136, 138, 236, 244].The IO is the only source of cerebellar climbing fibers (CFs) which target the Purkinje cells (PC). Each PC receives a single CF which massively connects with this single neuron strongly driving its activity. When a spike of the IO reaches its target PC, the Purkinje cell fires a complex spike. Each CF connects approximately with ten PCs. Nevertheless, the IO fires at a very low frequency (between 1–10 Hz, average 1 Hz) and therefore, the amount of spikes coming from the CFs is almost negligible compared to the activity of the PCs generated by the parallel fibers (simple spikes, firing at frequencies over 40 Hz)) [21, 97, 222, 246].

Neurophysiologic studies have revealed that there are many adaptation mechanisms at the cerebellum. Each of them may have a specific purpose (segmentation, maximization of information transference, correlation of sensorimotor signals, etc.) [65, 154]. In particular, the activity of the IO has a strong impact on the PF-PC synaptic adaptation [114]. The adaptation of these synapses mediated by this activity seems to play a crucial role in correlating the sensorimotor activity with a *teaching signal* (arriving from the IO) [9, 103, 137, 192]. This teaching signal can be seen as an *intentional signal* that highlights, in time domain, the accuracy of the movement that is being performed. As proposed in [41, 252], this signal may be related to the error during a movement. But since the IO is only capable of very low frequency output spikes (typically, output activity between 1 and 10 Hz), it does not encode the error quantity accurately in only one movement repetition, but rather provides a progressive estimate. Therefore, during repetitions of movements, its statistical representation may reproduce the error evolution more accurately [168, 236, 283] and thus, it can be a useful guide towards efficient error reduction to achieve accurate movements.

In the last decades, neuronal network simulators have markedly evolved, achieving to play a major role in neurophysiologic studies. New systems (such as the EDLUT simulation environment commented in chapter 2) allow the simulation of detailed cell models in the framework of large-scale networks. One of the main advantages of having this kind of tools is the the capability of testing functional hypotheses of biological structures having a full control of the parameters involved in the experiment. Thus, jointly with behavioral studies with animal species (rats, monkeys or even humans), simulation works can help to further understand the factors which influence the accurate realization of complex tasks. Concretely, in this chapter, we present a cerebellar-like architecture which is able to accurately carry out fast trajectories by using non-stiff-joint robots and we study the influence of the synaptic plasticity parameters at the parallel fibers and the sensor information convergence at the granular layer.

The working hypothesis and methodology of this work can be briefly described as follows:

- We address a biologically relevant task which consists in an accurate manipulation of objects which affect a base (kinematic and dynamic) model of the base plant using low power actuators.
- We define and implement a spiking-neuron based cerebellum model to evaluate how different properties of the cerebellar model affect the functional performance of the system.

It is important to remark that my personal contribution has been focused on the implementation of the spiking neural cerebellar-like network (neuron models, network topology, activity rates, parameter tuning, etc), while the actual control loop and interfaces with the robot (sensorimotor signals and errors) have been worked out by Niceto Luque.

4.2.1 The state of the art: functional models of the cerebellum

Since the earliest studies about the Central Nervous System, the cerebellum has received an important part of the neuroscientist community attention. Traditionally, the cerebellum has been reported to play an important role in motor control tasks [214]. However, in the last years, several studies have revealed that the cerebellum could be involved in some other cognitive functionalities, such as attention [109] or language [119], and probably some emotional ones, such as regulating fear or pleasure responses [282].

Although due to the symptoms of different diseases related to cerebellum this influence has become evident, the way in which the cerebellum can perform these tasks remains unclear. Several working models have been proposed in the last years with the target of explaining how the human motor system can achieve such high performances. Different works at the literature focus on specific properties of the cerebellum such as internal-clock-like behaviour, signal processing capabilities at different layers, functional abstractions as adaptive filters within the system, etc. This section tries to relate these different models and integrate them into a functional study of the cerebellum model. It evaluates whether these different properties can co-exist or are supported by complementary cell or network features.

The biological systems have not been engineered, they have evolved through millions of years. This makes the system analysis difficult since many functional properties may be distributed and supported by cell, network or adaptation mechanisms complementary. One of the challenges in these years is the integration of different functional models or hypothesis, and their contrasted study against realistic models of the biological systems. This will reveal how different cell and network features support complementary and in a distributed way functional properties. This review represents an integration effort which relates specific functional hypotheses such as internal clock and system adaptation (by means of adaptive filters) with a biologically realistic cerebellar model. Though this has been usually discussed in the original works in which the different working hypothesis are proposed, this review allows to evaluate them conjointly in an integrative manner.

It is particularly interesting to integrate hypothesis related with spatio-temporal processing (such as adaptive filters based models) and temporal properties of the network (such as resonance frequencies, time window sensitivity, internal clocks) and how they take place in a biologically realistic model. Some of these models such as the internal clock based on random activity sequences (at the granular layer) may seem conflicting with the adaptive filter models in which the granular layer delivers well structured information about sensorimotor complexes. In this section, we discuss how these different models and working hypothesis may co-exist and be supported by biologically realistic properties of cells and system topology.

Furthermore, there has also been an intense debate in the scientific community about the neural code itself, whether it is based on firing rates (encoding analog signals) or if it is related with single events (spike timing encoding) or hybrid models (such as rank order coding, in which the first spikes encode a major amount of information) or even models in which single spike times and cell stochasticity are used to sample analog signals [168, 238]. Interestingly enough, the different functional properties discussed in this review are supported by different neural codes, that seem to be compatible among themselves.

In this section, we will separately analyze three concrete models which represent the different perspectives from which neural systems can emerge:

- The model first proposed by Fujita [94] and the subsequent studies by Dean and Porrill [79]. This general model explains the cerebellum as an adaptive filter capable of correlating input signals and corresponding desired output signals by means of the signal decomposition into its frequency components at the granular layer. This model can correspond to an engineering point of view (signal filtering), although it has been widely based on neuro-physiological evidences.

- The second studied model is strongly based in neuro-physiological studies. This functional model, firstly proposed by Maex and De Schutter [185], develops a granular layer which generates synchronous output activity at the parallel fibers. Later, this hypothesis has been further developed including some others experimentally inspired capabilities such as center-surround organization, time-windowing, high-pass filtering in response to spike bursts and coherent oscillations in response to diffuse random activity [251] (as we have deeply discussed in chapter 3).

- Thirdly, we explain the functional model proposed by Yamazaki and Tanaka [288] focused on how the cerebellum can inherently embed internal clock signals. Following this model, the granular layer generates very long sequences of random active neuron populations but without recurrence after the occurrence of an initial input event. This long activity series can represent the passage of time (POT). On this way, the granular layer can be seen as a liquid state machine [183]. This model emerged from the attempt of implementing a functional model which could successfully perform the eyelid conditioning task.

Finally, we evaluate which parts of each model can coexist based on different complementary neural or network topologies or which ones are conflicting with each other.

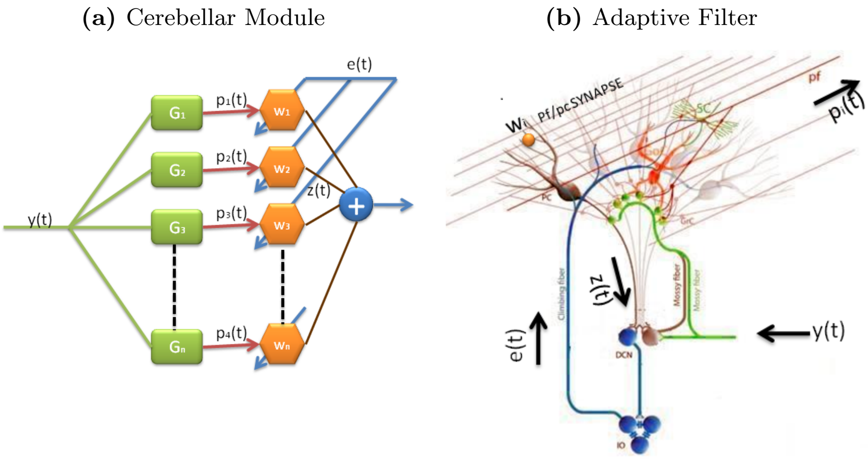
4.2.1.1 The cerebellum as an adaptive filter

In order to understand the cerebellum as an adaptive filter, firstly we may focus on what is meant with the terms *adaptive* and *filter*. The concept of filter involves a transformation between an input and its corresponding output according to some rules to achieve a specific goal. In our daily routine a common activity, such as running, generates tons of information from different sources (very high dimension sensor and motor signals). This amount of information is collected by the central nervous system (CNS). The CNS deals with incoming information from the visual cortex, auditory signals (we can hear the footsteps), the surrounding environment sounds, even some traffic claxons, and propioceptive information from muscles, joints, and tendons. The CNS filters and structures all this information allowing only the most relevant information related to the task under execution (running) to be considered. Once we learn to run, we are unconscientiously able to filter the information to keep on running in a flat surface but, what happen when the surface is plenty of rocks and holes in a hill?. The previous running acquired experience is not applicable directly in an autonomous way, we need to modify the way we run according to the newest sensorial incoming information, we have to adapt ourselves.

Adaptive filters as cerebellar functional basis was firstly proposed by Maar-Albus [9, 192]. It was a response coming from control theory towards a better understanding of the unclear biological control mechanims taking place in a distributed way at the CNS. It was a successful attempt to apply a very well known field (such as control theories and adaptive filters) to as reverse engineering effort towards understading the functional properties of the cerebellum [94].

Adaptive filters present parameters which can be self-adjusted to modify the output form. The input signal is decomposed into component signals by means of a set of filters with different spatio-temporal properties. Filter set outputs are weighted and summed up to obtain the desired output. In the work of Marr-Albus it was proposed how the self-adjustment can be driven by a learning rule consistent with the well known Hebbian covariance rule [239]. The learning rule modifies the weights according to the relation between the corresponding component signal and the teaching signal. This is called the Analysis-Synthesis filter model proposed by Dean and Porril [77, 79]. The system analyses and separates the frequency components of the input signal, correlates these individual components and the system error and is able to synthesize (through local adaptation at the parallel fibers) the appropriate filter responses towards a desired output.

To establish a comparison between this Analysis-Synthesis filter model and the cerebellar microcircuitry the reader is referred to Figure 4.2.


(a) Cerebellar Module

(b) Adaptive Filter

Figure 4.2: The most relevant connections within a cerebellar module and in relation with an adaptive filter model - (a)The cerebellar module presents different connections communicating different circuit elements in closed loops. Mossy fibers (MFs) contact granule cells (GrC) and Deep Cerebellar Nuclei (DCN) cells which, in turn, receive inhibition from the same common set of Purkinje cells (PC). Moreover, the IO cells project climbing fibers that contact PC which also are projected to DCN cells. (b)Filter inputs are assumed to be delivered as firing rate through the MFs. These incoming inputs are conveyed by GrC into parallel fibers. The component weights act as the synapses made by parallel fibers on PCs. PCs add the component weights signals obtaining the output. The correlation between climbing-fiber inputs and parallel fiber climbing is used by the correlation learning law, the strength of the synapse is decreased (Long Term Depression, LTD) if the correlation (with the error signal) is negative and it is increased (Long Term Potentiation, LTP) when the correlation is positive.

Climbing fiber and teaching signal It is well known that climbing fibers carry information from various sources (spinal cord, vestibular system, red nucleus, sensory and motor cortices,etc.). Within the adaptive filter, the climbing fiber activation is thought to work as a motor error signal sent to the cerebellum. Although recent studies revealed this is an important signal for motor timing, the way in which the error is translated into a climbing fiber signal is still an open issue [22, 74, 102, 281]. In the very first approach to cerebellum as an adaptive filter made by Fujita it was assumed that the climbing fiber response acted as a teacher signal in which its variable impulse rate was given by the difference between the output signal and a desired response. Other related works, such as the contributions of Schweighoffer [236, 237, 238] further study how this error estimation can be sampled with the low rate from the Inferior Olive (1–10 Hz) studying the viability of chaotic sampling approach. The weights are adjusted increasing or decreasing iteratively their values in order to obtain the desired output signal, if the correlation between the *error signal* (climbing fiber) and the component weights (mimicking the synapses made by parallel fibers on PCs) was positive the weight was increased; otherwise it was decreased. Fujita adopts a similar adaptation mechanism towards the reduction of the error. Although this approach does not discuss how biologically plausible it is, taking into account specific issues such as climbing fibers low rate, it is an inspiring source which is used in different applications [24, 181, 213]. Fujita's approach has been evolved along last decade [77, 78, 79] but not much effort has been done to relate the model with actual biological plausibility of different cell and network properties observed in biological systems.

Granular layer scheme In order to understand a functional scheme of this theory different general assumptions need to be done:

- Golgi cells act as leaky integrators. According to discharge pattern observations [121] it is assumed that Golgi cells act as temporal filters, with a transfer function given by $G(s) = \frac{K}{1+T_s}$ a low pass filter in which $K > 0$ represents the Laplace transform of complex frequency and T represents the cut-off period (the inverse of the cut-off frequency). That means Golgi cells, in this model, work as leaky integrator modules with a time constant T on the order of several seconds. Due to the feedback inhibitory loop (Granule cells→Golgi cells→Granule cells), a similar activity pattern can be expected at the corresponding parallel fiber. That means that we obtain banks of low pass filters that respond to different ranges of frequencies. Their output is convoluted with different weights (parallel fiber→Purkinje cell synaptic weights are adapted according to a teaching signal driven by the error). In this way, the global response is a weighted sum of different frequency components of the system input signal.

- Adaptive filter breaks down the input signal into different components (supposed to be done at the granular layer [33, 201]), ensuring a diversification in parallel fiber signals. This diversification is a phase diversification; this cerebellum model uses spikes as simple activity estimators (rate coding).
- Identical mossy fiber input signals should hit onto identical Golgi-Granule cell areas. This is a controversial point. Although in [78] this fact is justified by means of biological records, until nowadays, multiple recording of Granule activity has been a troublesome task, and the issue of neural code at the granular layer is still open. There is no a consensus along adaptive filter cerebellum theory [77, 78, 79, 94] on how the mossy fiber codification is done. Mossy fiber inputs are treated as signals given by *transducers.* Thus, transmission information spike theory [27, 90] is not taken into account; interspike/intraspike distance [26], population neural coding [90, 285] or first spike information [26] are not included in the general adaptive filter model.

Purkinje cells Finally, the output signals from granular layer are conveyed to Purkinje cells. The connection pathways to purkinje cells are simplified, only the direct synaptic contact PF/PC is represented as an excitatory connection and an inhibitory interneuron per Purkinje is responsible of the negative contribution to PCs. Both contributions are summed up into the final system output.

4.2.1.2 The cerebellum as a synchronous system

In 1998, Reinoud Maex and Erik De Schutter published their granular layer simulation model [185]. This model was composed of Golgi cells, granule cells and mossy fibers following an structure based in physiological findings. Their simulations concluded that the populations of both Golgi and granule cells became entrained in a single synchronous oscillation as the response to random mossy fiber stimulation. These oscillations showed frequencies ranged from 10 to 40Hz depending on the average input frequency, and were robust over different parameters such as synaptic connection strengths, mossy fiber firing rates or the spiking propagation speed in parallel fibers. However, following this model, desynchronization could happen in the network if one of the next conditions occurred:

- Very low mossy fiber input activity. The synchronization of activity at parallel fibers is mainly based on the convergence-divergence ratios at GoC. Thus, a very low activity rate at the mossy fibers (and therefore at the GrCs) generates almost no excitation at the GoCs, and in this way, the synchronization path (GrC $\rightarrow$ GoC $\rightarrow$ GrC) is disabled.
- Strong dominant excitation of Golgi cells through mossy fiber synapses (in relation to parallel fiber synapses). The convergence ratio in GrC $\rightarrow$ GoC is lower than the MF $\rightarrow$ GoC one due to the huge amount of GrCs existing in the cerebellum. Thus, MF $\rightarrow$ GoC activity destabilizes the synchronism in the GrC.
- Tonic activation of granule cell inhibition. The tonic inhibition of granule cells (which emerges from the Cl^- current recordings in GrCs of adult rats in vitro [276]) decremented the level of average activity in the granule cells. Thus, the synchronization elements (the Golgi cells) remain nearly inactive and do not produce such synchronism.

This model shows some weak points, most of them from the restrictions of the simulation capabilities in 1998. That is, the network topology oversimplifies some of the main aspects of the granular layer, such as the glomeruli [146] or the presence of interneurons [19]. Moreover, another point (which could be critical following Yamazaki's hypothesis [288]) is the absence of NMDA-receptor channels. In that model [288], the NMDA receptors plays a major role in the temporal integration of activity specially in Golgi cells due to the long time constant which is associated to these channels.

All the commented weaknesses were solved in a more detailed model presented by Solinas et al. in 2010 [251]. This model simulates a realistic large-scale model of the cerebellar granular layer and verifies some spatio-temporal filtering properties which previously were found in physiological studies. These filtering properties are the following ones:

- *Center-surround organization.*The response to a burst presents a central area (closer to the stimulated mossy fibers) where the excitation dominates, while in the surrounding areas, inhibition dominates because Golgi cell axons reach a wider area than the glomerulus ramifications. Thus, the excitation-inhibition diagram presents a mexican-hat shape.

- *Time-windowing.* This model also presents a variable-in-time response to a burst. After the stimulation of mossy fibers, the feedback inhibitory loop inhibits the previously stimulated granule cells. Thus, time-windows when the granule cells are mainly inhibited follows each excitation time. This effect has been previously predicted [67] and more detailed studies have been carried out in chapter 3 where the influence of the synaptic weights has been quantified.
- *High-pass filtering in response to spike bursts.* In response to different frequency stimulation, the granular layer shows a high-pass filter behavior with a rapid growth of the response between 50 and 100Hz.
- *Coherent oscillations in response to random input activity.* Finally, when the granular layer is stimulated with continuous random activity, the granule cells (and the Golgi cells as well) show synchronous oscillations in the theta band. This hypothesis is the same proposed by Maex and De Schutter in their previously commented paper [185].

Although this model further develops the previous hypothesis by Maex and De Schutter, the extensive work in [251] includes the implementation of the glomerulus topology following previously observed patterns, the usage of very detailed models (Hodking-Huxley gating schemes and synaptic transmission using synaptic vesicle cycling schemes) and the inclusion of non-previously used receptors such as NMDA, GABA-A (both a1 and a6) or kainite and the implementation of gap junctions confer a strong biological base to this model.

However, the main weakness of this model remains in the lack of a functional model of the whole system. Although this model shows the main granular layer properties which make the cerebellum being such an important part of the nervous system, it remains unclear how these properties can be efficiently used to get an efficient way of signal processing and furthermore, how this processing capabilities can efficiently resolve specific problems in the framework of relevant and well studied tasks, such as eyelink conditioning or the movement correction. An important point addressed in these characterization works is the study of how the information is encoded at the granular layer. The encoding mechanism proposed by Brasselet et al. [27] can profit from the spatio-temporal properties of the granular layer. However, a more detailed analysis and simulation evidences are still required.

4.2.1.3 The cerebellum embedding an internal clock

The hypothesis of the cerebellum as a liquid state machine emerges from a top-down strategy as described in [196]. This model (first proposed by Yamazaki and Tanaka [287]) confers the capability of generating activity patterns without temporal recurrence to the cerebellum and, in this way, representing the passage of time (POT). Following this hypothesis, the granular layer may work as an event-driven internal clock triggered by an initial activity pattern, and in which each time step would be represented with combinations of active granule cells.

Different performance has been shown in this model [287], such as:

- The sequences of activity are stable against noise, both with respect to input signals and the connection matrix.
- The activation pattern is reproducible. The internal clock can be reset by only stimulating the granule cells with a strong transient input signal.
- The sequence of activity becomes faster or slower by only changing different parameters of the network (such as the Golgi cell's time constant or the strength of the inhibition from Golgi cells to granule cells). However, these modifications in the speed of the clock caused a marked decrease in the accuracy of the model.

Following this model, the cerebellum can be seen as a liquid state machine [183]. As proposed by Yamazaki and Tanaka [288], the presentation of finite sequences of active neuron populations without recurrence as the response to different combinations of binary inputs can be compared to a liquid state machine with high power of information processing.

Although it seems clear that accurate mechanisms of passage of time exist in the cerebellum, different implementations have been recently proposed and analysed [290]. However, neuro-physiological findings mostly corroborate the hypothesis that the neural clock can be carried out by the inhibitory loop composed of granule cells and Golgi cells

[289]. In this way, the granule cells would excitate the Golgi cells (through the parallel fibers) as a result of the input stimuli, and subsequently would inhibit some of the granule cells, and would obtain the transition between active and inactive states which represents the passage of time.

Finally, a more detailed implementation was presented in [131]. This new version of the model of the internal clock takes advantage of the inhibitory path composed by granule cells, stellate cells and purkinje cells and the LTP suggested in the stellate cells→purkinje cells connection. Moreover, this model incorporates more detailed cell models (single-compartment Hodgkin-Huxley units in GrC, PC and SC and multi-compartment Hodgkin-Huxley units in GoC), but the results seem to minimize the role of the neuron model leaving to the network topology the key properties of the model.

This model especially fits with the problem of the Pavlovian eyelid conditioning. In this experiment, a conditional stimulus (CS) is presented a certain time before an unconditional stimulus (US). Typically, a tone is used as a CS, and an air puff directed to one eye as the US. After repeating this training for some iterations, the subject learns to close its eye (called conditioned response, CR) some time after the CS and just before the air puff reaches the eye. The time between the CS and the US usually is called the interstimulus interval (ISI). The resolution of the problem of the Pavlovian eyelid conditioning implies the ability of accurately estimating the passage-of-time from the CS to the US. Thus, this system, in conjunction with the US signal (which reaches the Purkinje cells through the climbing fibers) and the presence of long-term depression (LTD) of parallel fiber terminals at Purkinje cells [135], is able to successfully correlate the US occurrence with the elapsed time since the CS onset.

Even though this simple model successfully resolves the problem of the eyelid conditioning, some points remain unclear. First, this model has never been put into practice with more difficult problems such as coordination of locomotor movements in which the cerebellum is involved according to experimental and clinical observations [91, 128, 129, 180]. The eyelid conditioning solution implies the correlation of the inter-stimulus interval and the US, which is carried out by the LTD in the parallel fiber connections. However, in this problem the output of the cerebellum (deep cerebellar nuclei cells) could be considered bistable, because only two different states can be presented (active DCN cells representing closed eyelid or inactive representing open eyelid). In other motor tasks, such as target reaching or fast ballistic movements, a continuous in time and value correction is required in order to complete the desired trajectory. Furthermore, different sensory and contextual information feedback (current position, position error, information related to the physical characteristics of the system, etc.) may help to improve the performance [40, 182] (further details about these developments can be found in chapter 4 of this document). The way in which the proposed model of POT at the granular layer may take advantage of all this information needs to be investigated.

The model proposed by Yamazaki and Tanaka concludes a cerebellar circuit composed of two different subcircuits [288]. The first one (where granule cells, Golgi cells and mossy fibers are located) represents a liquid state machine, and generates non-recurrent sequences of activity. The second one (where mossy fibers and cerebellar nucleus cells are located) represents a simple perceptron. Both circuits receive a teaching signal from the inferior olive. This signal carries the activity generated by the US (the air puff hitting in the open eye) and supervises the input-output relationship. However, there is no agreement in the neuroscientific community around the kind of signal which the inferior olive carries. On one hand, several authors defend that the climbing fibbers could carry teaching signals which encode the error of the movement or conditioned response [41, 288]. On the other hand, physiological studies have showed very low firing rates; in both anesthetized and awake animals, olivary neurons discharge either a single spike or a burst of spikes (2–5 spikes with an interspike interval of 2–3 milliseconds) about once or twice per second [15, 16, 54]. Thus, it seems unlikely that the climbing fibers signals would encode any kind of error-related information. Nevertheless, there are also models [168, 238] related to how low rate sampling of error signals may take place at the Inferior Olive. Furthermore, physiological recordings showing synchronous and rhythmic firing in climbing fibers seem to reject the representation of the error signal.

Another point which has not been included in the Yamazaki's model is the synchronism in the parallel fiber activity. As discussed in section 4.2.1.2, several simulation and physiological studies have shown that the activity at parallel fibers (and GoCs as well) could follow rhythmic patterns due to the convergence-divergence ratios at granular layer [185]. Although the author of this model points out the lack of NMDA (N-methyl D-aspartate) channels as the determinant cause [289], recent simulations have reported synchronism in both granule cells and Golgi cells even by using NMDA receptors [251].

4.2.1.4 Discussion

In this section, we have analyzed three of the most important functional hypotheses of the granular layer. Every model has been related with different points of view and experimental data.

The first model, proposed by Fujita [94], and further developed by Dean and Porrill [77, 78, 79] comes from a engineering point of view. In this way, if we see the cerebellum as an whole analog system which receives input signals with different frequency components, the granular layer would separate the signal components in order to correlate those components and the error signal coming from the climbing fibers. Therefore, the cerebellum is able to send a modified analog corrective signal to the deep cerebellar nuclei. This model can solve most of the tasks where the cerebellum is involved, such as the eyelid conditioning or the movement correction, but in the other hand, the lack of realistic implementations of this system, and the suppositions that this model implies (input signal with frequency components, error correlated with these components,and the time constants of the Golgi cells as long as several seconds) makes this model still not well established among the neurophysiology community.

The second model, first proposed by Maex and De Schutter [185], and implemented in more detail by Solinas et al. [251], is based on the experimental observations in mammalians. These models build an artificial granular layer following morphologic characteristics and reproduce some of the previously studied effects (such as the center-surround activity organization, time-windowing, high-pass filtering in response to spike bursts and coherent oscillations in response to random input activity). The glomeruli topology deserves an special mention due to the major role in the spatial organization of the activity in the granular layer. Although this model fully fits from a neurophysiological point of view, yet its application in the context of a whole cerebellum model and a problem to be solved remains unclear.

The third model, proposed by Yamazaki and Tanaka [287, 288, 290], presents a third perspective. This model is created and evaluated in the framework of a defined problem (the Pavlovian eyelid conditioning), but taking into account the restrictions which the experimental studies impose. Therefore, following this model, the granular layer can behave as an internal clock by means of the generation of activity patterns without temporal recurrence at the parallel fibers. In the next stage, the teaching signal (the unconditioned stimulus) can correlate the stimulus with the parallel fiber activity (the combination of active fibers at the same time than the US). In this way, this model can be considered a primary application of the biological findings to the resolution of a *real* problem, but even though this is efficiently solved, some points have not been solved yet, such as the previously mentioned (the type of the signals reaching the cerebellum through the climbing fibers, the random input activity in mossy fibers, or the applicability of this model to a more complex task as the target reaching or movement correction).

But could this three models be compatible? Or are they necessarily opposed? In a first impression, the three models are not fully compatible, but some features might be able to work together in an integrative manner. For example, we could design a system where the passage of time could be represented by activity patterns without temporal recurrence at the parallel fibers, but this system should also improve the separation of the states by means of alternation of mainly active and mainly inactive states synchronized in the cerebellum (different levels of sensitivity). This could be applied to more complex motor control problems (than the traditionally used eyelid experiments).

Finally, this review highlights the importance of the multidisciplinary studies where both biology and engineering are involved. While biologists have studied and proposed very detailed models where their experiments are fully reflected and layer property characterization is specifically addressed, yet systems are not thought to carry out specific tasks at a system level. On the other hand, engineers have proposed machine-like systems which try to solve the problems in the same way as engineered electronic systems, but whose systems miss the accuracy and the performance of biological systems. Thus, both, neurophysiologists and system engineers need to work in close collaboration with each other towards understanding how specific problems are solved using neurophysiologically plausible computational principles.

4.3 Materials and methods

For extensive spiking network simulations, we have further developed and used an advanced Event-Driven simulator based on LookUp Tables [4, 227, 228] (for more information about development related with these experiments, we recommend the chapter

2 in this document). EDLUT is an open-source tool which allows the user precompiling the response of a predefined cell model (whose dynamics are driven by a set of differential equations) into lookup tables. Then, complex network simulations can be performed without requiring an intense numerical analysis. In this research, as a first approximation, neurons were evolved versions of leaky integrate-and-fire neuron models with the synapses represented as input-driven conductances.

For the experimental work, we have used a biomorphic robot plant, a simulated LWR (Light-Weight-Robot). This robot has been developed at DLR [34]. The LWR's arms are of specific interest for machine-human interactions in unstructured environments. In these scenarios, the use of low-power actuators prevents potential damage on humans in case of malfunctioning. Although a real impact on robotic applications is beyond the scope of this work, the target application scenario of this robotic platform based on non-stiff low power actuators shares certain characteristics with the daily manipulation tasks performed by humans. Therefore, we considered this robotic platform an appropriate tool for validating the cerebellar-based model inference engine under study.

For the sake of simplicity, in our simulations, we use a simulator of this robot in which we have fixed some joints to reduce the number of actual joints to three, limiting the number of degrees of freedom to three.

4.3.1 Training trajectories

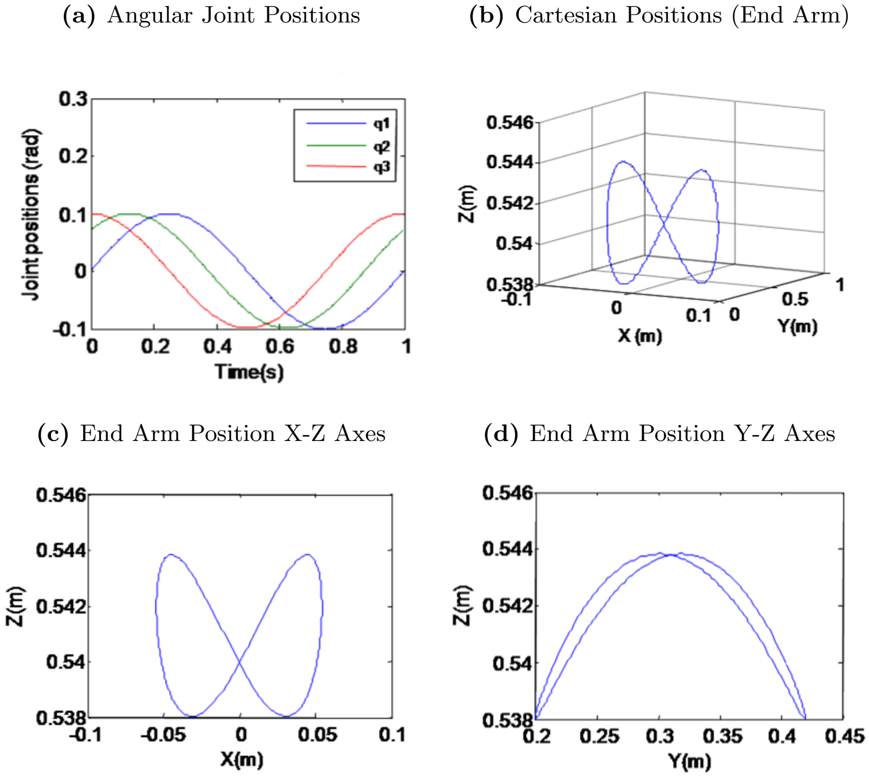
The described cerebellar model has been tested in a smooth pursuit task [24, 112, 159]. A target (desired target movement) moves along a repeated trajectory, which is composed of vertical and horizontal sinusoidal components. The target movement describes the *8-shape* trajectories illustrated in Fig. 4.3, whose equations, in angular coordinates, are given by the following expressions 4.1. We have evaluated the learning capability performing a goal movement along this target trajectory. Each joint movement in our task is defined by $q1$, $q2$, and $q3$, respectively.

$$q_1 = A_1 \cdot sin(\pi \cdot t) + C_1 \tag{4.1a}$$

$$q_2 = A_2 \cdot sin(\pi \cdot t + \theta) + C_2 \tag{4.1b}$$

$$q_3 = A_3 \cdot sin(\pi \cdot t + 2 \cdot \theta) + C_3 \tag{4.1c}$$

This trajectory with the three joints which are moving following sine shapes is shown in Fig. 4.3. We chose fast movements (one second for the whole target trajectory) to study how inertial components (when manipulating objects) are inferred at the cerebellar structure. Slow movements would hide changes in the dynamics of the arm+object model, since they would not have significant impact when performing very slow movements.


(a) Angular Joint Positions

(b) Cartesian Positions (End Arm)

(c) End Arm Position X-Z Axes

(d) End Arm Position Y-Z Axes

Figure 4.3: Three joint periodic trajectories describing 8-shape movements in joint coordinates - (a) Cartesian coordinates of the 8-like trajectory. (b) 3D view of the 8-like trajectory. (c) X and Z-axe representation of this target trajectory. (d) Y and Z-axe representation of the 8-like target trajectory.

Though for the sake of simplicity, we have used a single 8-like trajectory in each trial, consecutive eight-like trajectories have also been tested leading to similar results (provided that the corrective torque values do not get saturated along the global trajectory).

4.3.2 Control loop. Interfacing the cerebellar model and the robot simulator

Some studies indicate that the brain may plan and learn to plan the optimal trajectory in intrinsic coordinates [132, 133, 206, 264]. The CNS (central nervous system) is able to execute three major tasks: the desired trajectory computation in visual coordinates, the task-space coordinates translation into body coordinates, and finally, the motor command generation. In order to deal with variations of the dynamics of the operator-arm, we have adopted a FEL scheme (Feedback-Error Learning) [156] in conjunction with a crude inverse dynamic model. In this scheme, the association cortex provides the motor cortex with the desired trajectory in body coordinates, where the motor command is calculated using an inverse dynamic arm model. On one hand, the spinocerebellum - magnocellular red nucleus system provides an internal neural accurate model of the dynamics of the musculoskeletal system which is learned with practice by sensing the result of the movement. On the other hand, the cerebrocerebellum - parvocellular red nucleus system provides a crude internal neural model of the inverse-dynamics of the musculoskeletal system which is acquired while monitoring the desired trajectory [156].

The crude inverse dynamic model works together with the dynamical model by updating the motor command by predicting a possible error in the movement. As it is illustrated in Fig. 4.4, the cerebellar pathways follow a feedforward architecture, in which only information about sensory consequences of incorrect commands can be obtained (i.e., the difference between actual and desired joint positions of the arm). The natural error signal for learning motor commands is the difference between actual and correct commands, this implies, for example, that if M muscles control N sensor dimensions involved in a task, then N-sensory errors must be converted into M-motor errors (MxN complexity). How to use this sensor information to drive motor learning is the so called distal error problem [112, 219]. In order to overcome this motor error problem, (the cerebellum in our scheme provides torque corrections) the implemented spiking cerebellum used an adaptation mechanism described in Section 4.3.3 which can correlate the actual and desired states towards the generation of accurate corrective motor commands.

In our model, the cerebellum receives well structured inputs encoding the planned trajectory. We assume that the errors occurred during the movement are encoded at the IO and transferred (at low firing rates) to the cerebellum through the climbing fibers. We have built a module to translate a small set of signals (encoding the arm's desired state) into a sparse cell-based spike-timing representation (spatio-temporal population coding). This module has been implemented using a set of input fibers with specific receptive fields covering the working range of the different desired state variables (position and velocity of the different joints). In this way, the robot (analog domain consisting of trajectory planer, trajectory generator, crude inverse dynamic arm model, and arm plant (Fig. 4.4)) has been interfaced with the spiking cerebellar model (spiking domain).

In our control loop, the desired states (positions and velocities) that follow a certain trajectory are obtained from an inverse kinematic model computed by other brain areas [206] and then, they are translated into joint coordinates. These desired arm states are used at each time step by a crude inverse arm dynamics model to compute crude torque commands which are added to the cerebellum corrective torques. This control loop is illustrated in Fig. 4.4.

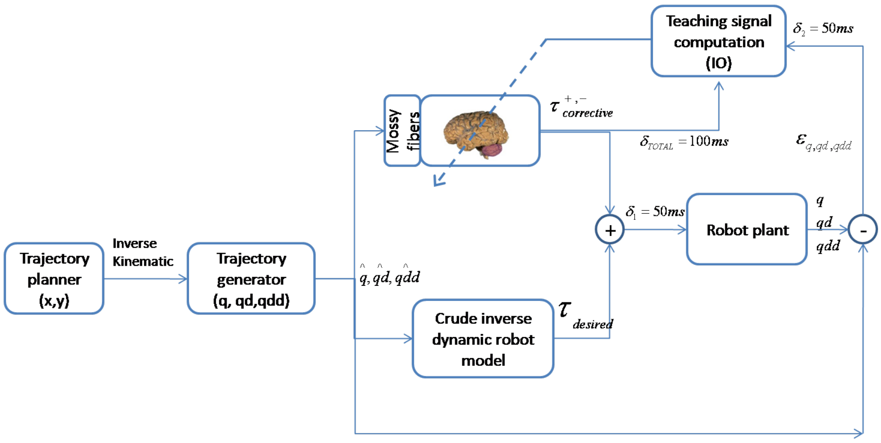


Figure 4.4: System control loop - The adaptive module (cerebellar-like structure) contributes to the actual torques being received by the crude inverse dynamics robot model to enhance the accuracy of the movement.

Fig. 4.4 illustrates how the trajectory planner module delivers desired positions and velocities for a target trajectory. The kinematics module translates the trajectory Cartesian coordinates into joint coordinates. The *crude inverse dynamics arm model* calculates the target torque in each joint which is necessary to roughly follow the target trajectory. But this crude arm model does not take into account modifications in the dynamics model due to object manipulation. Thus, if only these torque values are considered, the actual trajectory may significantly differ from the desired one. The adaptive cerebellar component aims at building corrective models to compensate these deviations, for instance, when manipulating objects.

In Fig. 4.4, the adaptive cerebellar-like structure delivers corrective actions that are added to compensate deviations in the base dynamics plant model when manipulating objects. In this forward control loop, the cerebellum receives a teaching error-dependent signal and the desired arm state so as to produce effective corrective commands. Total torque is delayed (on account of the biological motor pathways) and supplied to the robot plant δ_{total}. The difference between the actual robot trajectory and the desired one is also delayed $\delta_{1,2}$ and used by the teaching-signal computation module to calculate the IO activity that is supplied to the cerebellum as a teaching input signal (for the computation of the cerebellar synaptic weighs). Using this control loop architecture, an accurate explicit model of the musculoskeletal arm inverse dynamics is not necessary. The cerebellum can infer corrective models tuned to different tools which may affect the dynamics of the plant (arm+object).

4.3.3 Spike-based cerebellar model

In order to test the working hypotheses, we have implemented two different oversimplified cerebellar models by using the EDLUT simulation environment (see chapter 2):

- In a first approach, we have tested the parameters which influence the learning at parallel fibers. The cerebellum has been modeled as a system including only Purkinje cell (PC) and Deep Cerebellar Nuclei (DCN) spiking cell models. The granular and molecular layers have been simplified by using the Yamazaki's hypothesis [288]. Following this hypothesis, the granular layer provides random and non-recurrent spiking states representing the passage-of-time. Since our input activity represents different input states (desired positions) by activating different mossy fibers each time, this activity could properly emulate the working model of the granular layer (even though several properties such as the sparse coding or other temporal properties are neglected). Therefore, this abstract cerebellar model where we fully control the input activity reaching the parallel fibers allow us the study of the learning characteristics.

- In a further approach, and as a second stage in the process of designing a more realistic functional cerebellar model, we have built a more complex model including a simplified version of the granular layer. This model includes only the granule cells (in addition to the previous model counting with Purkinje and DCN cells) which detect the coincidence of input spikes in more than one input synapses (mossy fibers) simultaneously. Thus, this abstract granular layer allows testing the influence of different input sources reaching coincident granule cells in the model inference process.

4.3.3.1 First Approach: Studying adaptation at parallel fibers

The proposed cerebellar-like architecture, organized in cerebellar microzones [79] (somatotopic arrangement), tries to capture some cerebellum's functional and topological features [10, 149].This cerebellum model consists of the following layers (Fig. 4.5)

- Input layer (120 cells). This layer represents a simplification of the mossy and granular layers of the cerebellum and drives PCs and cells of the deep cerebellar nuclei (DCN). The goal of this simplification is to facilitate the study of how the sensorimotor corrective models are stored in adapted weights at the PF-PC connections. This input layer has been divided into 6 groups of twenty-grouped cells which carry the desired joint velocity and position information (these desired position and velocity coordinates can be thought as efferent copies of the motor commands or *motor intention*); for the propioceptive encoding, 3 groups of cells encode the desired joint positions (1 group per joint) and the other three encode the desired joint velocities. The analog position and velocity transformation into the fiber spike activity is carried out by using overlapping radial basis functions (RBF) (Fig. 4.6) [17] as receptive fields of the input-variable space, see Expressions 4.2 (joint-specific angular position).

$$I_{mossy_i} = e_i^{\frac{(InputSignal-\mu_i)^2}{2\sigma^2}} \tag{4.2a}$$

$$\tau_{m_i}\frac{dv_i}{dt} = -v_i(t) + R_i I_{mossy_i} \tag{4.2b}$$

where $0 < i < n$, n represents the size of each mossy group, μ_i is the mean and σ the standard deviation of the *ith* RBF. Related to the cell dynamics, τ_{m_i} is the resting time constant, v_i the membrane potential, I_{mossy_i} the input current, and R_i is related to the resting conductance of the membrane. For the sake of simplicity, in our model, we have not included a more detailed cellular structure (Golgi cells, interneurons, mossy fibers, etc). We have adopted well structured, noise free patterns to encode sensorimotor signals to partially embed potential roles typically performed in the granular layer [66, 67] (such as noise reduction, pattern separation, etc). Parallel fibers are the output of this layer.

- Inferior olive cells (IO) (48 cells). This layer consists of 6 groups of 8 cells. It translates the error signals into teaching spikes to the Purkinje cells. The IO-output carries the teaching signal used for supervised learning (see STDP section).

- Purkinje cells (PC) (48 cells). They are divided into 6 groups of 8 cells. Each input cell sends spikes through excitatory connections to PCs, which receive teaching signals from the IO. The PF-PC synaptic conductances are set to an initial average value (15nS) at the beginning of the simulation, and are modified by the learning mechanism during the training process.

- Cells of the Deep Cerebellar Nuclei (DCN) (24 cells). The cerebellum model output is generated by 6 groups of these cells (2 groups of 4 cells per joint) whose activity provides corrective torques to the specified arm commands. The corrective torque of each joint is encoded by a couple of these antagonist groups, being one group dedicated to compensate positive errors and the other one, to compensate negative errors. Each neuron group in the DCN receives excitation from every input layer cell and inhibition from the two corresponding PCs. In this way, the PC-DCN-IO sub circuit is organized in six microstructures (Fig. 4.5), three for positive joint corrections (one per joint) and three for negative joint corrections (one per joint).

We have used leaky integrate-and-fire (I&F) neurons with synapses modeled as variable conductances to simulate Purkinje cells and DCN cells. These models are a modified version of the Spike-Response Model (SRM) [100]. These synaptic conductance responses were modeled as decaying exponential functions triggered by input spikes as stated by Expressions 4.3. Thus, these neuron models account for synaptic

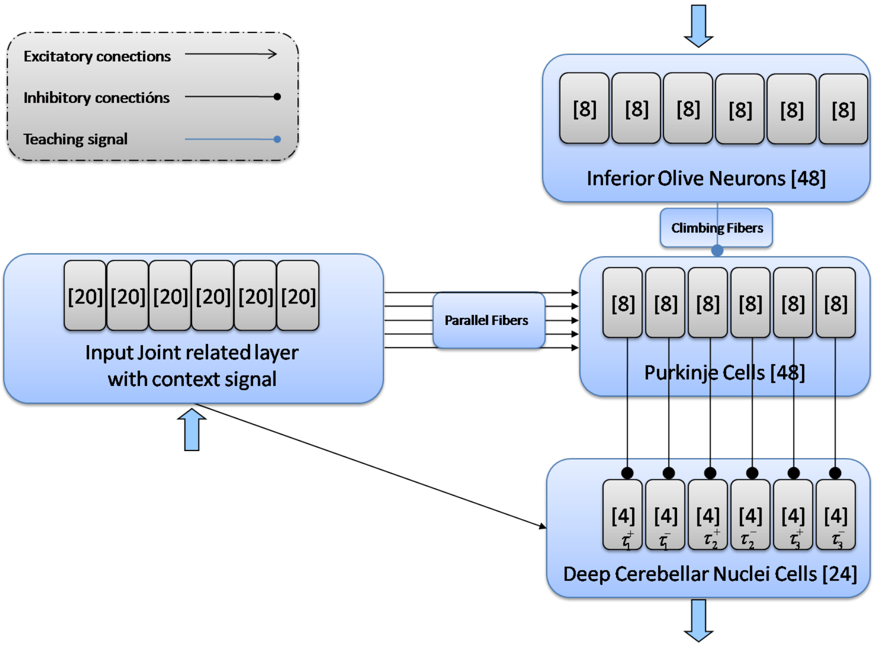


Figure 4.5: First-approach cerebellum model diagram - Inputs encoding the desired position and velocity (arm state) are sent (upward arrow) through the input layer which represents a simplification of the mossy fibers and granular layer. Inputs encoding the error are sent (upper downward arrow) through the inferior olive (IO). Outputs are provided by the deep cerebellar nuclei (DCN) (lower downward arrow). The DCN collects activity from the input layer (excitatory inputs which provide DCN with a basal activity when an input stimulus is presented) and the Purkinje cells (inhibitory inputs). The DCN activity represents the corrective torque generated by the cerebellum. This output activity is transformed into a proper analog torque signal by means of a buffer in which the DCN activity is accumulated. This activity buffer is used to compute an analog average value that acts as a corrective torque.

conductance changes (driven by pre-synaptic activity) rather than simply for current flows, providing an improved description over more basic I&F models. Table 4.1 contains the neuron model parameters of the Purkinje cells and DCN cells.

$$g_{exc}(t) = \begin{cases} 0 & \text{when } t < t_0 \\ g_{exc}(t_0) \cdot e^{-\frac{t-t_0}{\tau_{exc}}} & \text{when } t \geq t_0 \end{cases} \tag{4.3a}$$

$$g_{inh}(t) = \begin{cases} 0 & \text{when } t < t_0 \\ g_{inh}(t_0) \cdot e^{-\frac{t-t_0}{\tau_{inh}}} & \text{when } t \geq t_0 \end{cases} \tag{4.3b}$$

$$C_m \frac{dV_m}{dt} = g_{exc}(t)(E_{exc} - V_m) + g_{inh}(t)(E_{inh} - V_m) + G_{rest}(E_{rest} - V_m) \tag{4.3c}$$

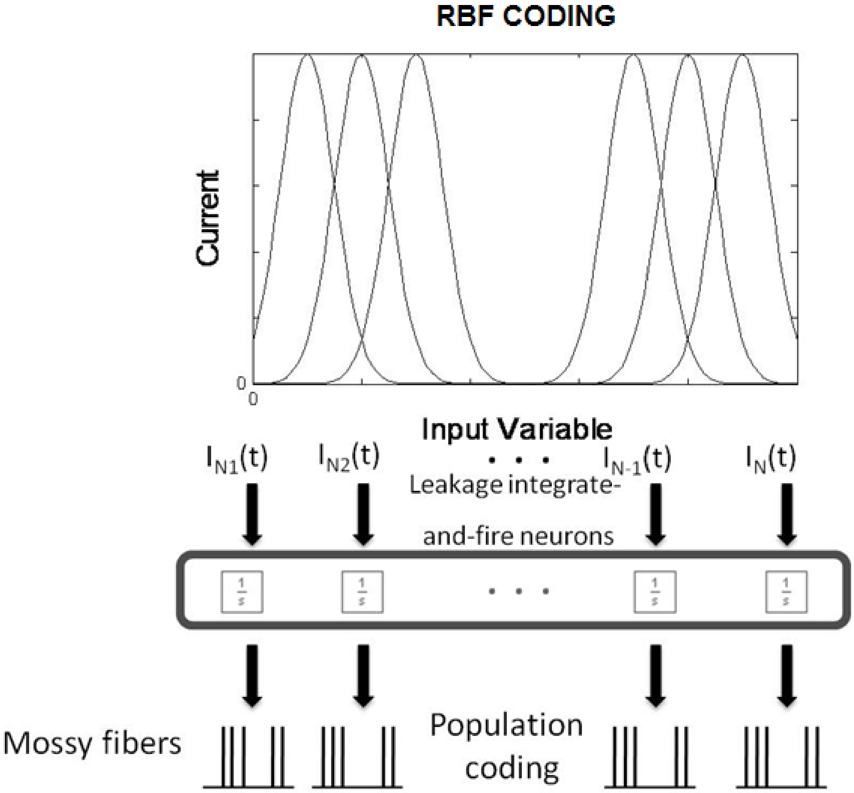


Figure 4.6: RBF bank encoding cerebellar input signals - Translation from joint-related analog variables (angular positions and velocities) into spike trains is carried out using overlapping radial basis functions (RBFs) as receptive fields in the analog domain. One-dimensional values are transformed into multidimensional current vectors (one for each RBF). Each current value is integrated using an integrate-and-fire neuron model and determines the output activity of an input cell of the cerebellum model.

Where g_{exc} and g_{inh} represent the excitatory and inhibitory synaptic conductance (time constant) of the neuron. τ_{exc} and τ_{inh} represent the time constants of the excitatory and inhibitory synapses respectively. Synaptic inputs through several synapses of the same type can simply be recursively summed when updating the total conductance if they have the same time constants, as indicated in Expression 4.4. Membrane potential (V_m) is defined through Expression 4.3c depending on the different reverse potentials and synaptic conductances.

$$g_{exc(post-spike)}(t) = G_{exc,j} + g_{exc(pre-spike)}(t) \tag{4.4}$$

	Purkinje cells	**DCN cells**
Refractory period	2ms	1ms
Membrane capacitance	500pF	2pF
Total excitatory peak conductance	1.3nS·175000syn·10%	1nS·7syn
Total inhibitory peak conductance	3nS·150syn	30nS·1syn
Firing threshold	-52mV	40mV
Resting potential	-70mV	-70mV
Resting conductance	16nS	0.2nS
Mem. pot. time constant (τ_m)	20ms to 30ms	10ms
Exc. syn. time constant (τ_{exc})	1.2ms	0.5ms
Inh. syn. time constant (τ_{inh})	9.3ms	10ms

Table 4.1: Neuron model parameters for the simulations - These parameters have been retrieved from [120, 143, 144, 232, 248]. In the table, nS stands for nanosiemens and syn stands for synapses.

where $G_{exc,j}$ is the weight of synapse j and a similar relation holds for inhibitory synapses.

4.3.3.2 Second Approach: Studying information convergence at the granular layer

In this second approach we have tried to build a more reallistic cerebellum model by means of including a simplified granular layer. Thus, the proposed cerebellar architecture (Fig. 4.7) consists of the following layers:

- Mossy fibers (MF): Mossy fibers carry both contextual information and sensory joint information. A mossy fiber is modeled by a leaky I&F neuron, whose input current is calculated using overlapping radial basis functions (RBF) as receptive fields in the value space of the input signals.

- Granular layer (1500 cells): This layer represents a simplified cerebellar granular layer. The information given by mossy fibers is transformed into a sparse representation in the granule layer [65]. Each granular cell (GR) has four excitatory input connections; three of them from randomly chosen joint-related mossy fiber groups and another one from a context-related mossy fiber. Parallel fibers (PF) are the output of this layer.

- Climbing fibers (CF) (48 climbing fibers): This layer is composed of 6 groups of 8 climbing fibers each. It carries the IO output which encodes teaching spike trains (related to the error) for the supervised learning in the PF-PC connections.
- Purkinje cells (PC) (48 cells): They are divided into 6 groups of 8 cells. Each GR is connected to 80 percent of the PCs. Each PC receives a single teaching signal from a CF. PF-PC synaptic conductances are modified by the learning mechanism during the training process.
- Deep cerebellar nuclei cells (DCN) (24 cells): The cerebellum model output is generated by 6 groups of these cells. The corrective torque value of each joint is encoded by a couple of these groups, one group is dedicated to compensate positive errors (agonist) and the other one is dedicated to compensate negative errors (antagonist). Each neuron group in the DCN receives excitation from every MF cell and inhibition from the two corresponding PCs. In this way, the subcircuit PC-DCN-IO is organized in six microzones; three of them for joint positive corrections (one per joint) and the other three of them for joint negative corrections (one per joint). The DCN outputs are added as corrective activity in the control loop.

Using this cerebellar model, the mossy fibers encode input representation in a rather specific way and the granular cells integrate information from different mossy fibers. These characteristics partially embed functional roles of the inhibitory loop driven by the Golgi cells. Therefore, although Golgi cells have not been explicitly included, part of their functional roles has been integrated into the system.

Mossy layer configuration in the cerebellar model Different mossy layer configuration models have been proposed in order to improve the cerebellum storage capability. All of them consist of 120 joint-related fibers;the mossy fiber layer has been divided into 6 groups of twenty fibers; 3 groups of fibers encoding joint positions (1 group per joint) and the other three, encoding joint velocities.

- *EC model.* Explicit Context encoding approach (16 context-related fibers plus 120 joint-related fibers); this mossy layer configuration uses the base desired-proprioceptive configuration adding 16 context-related fibers.The contextual information is coded by two groups of eight fibers. An external signal (related to

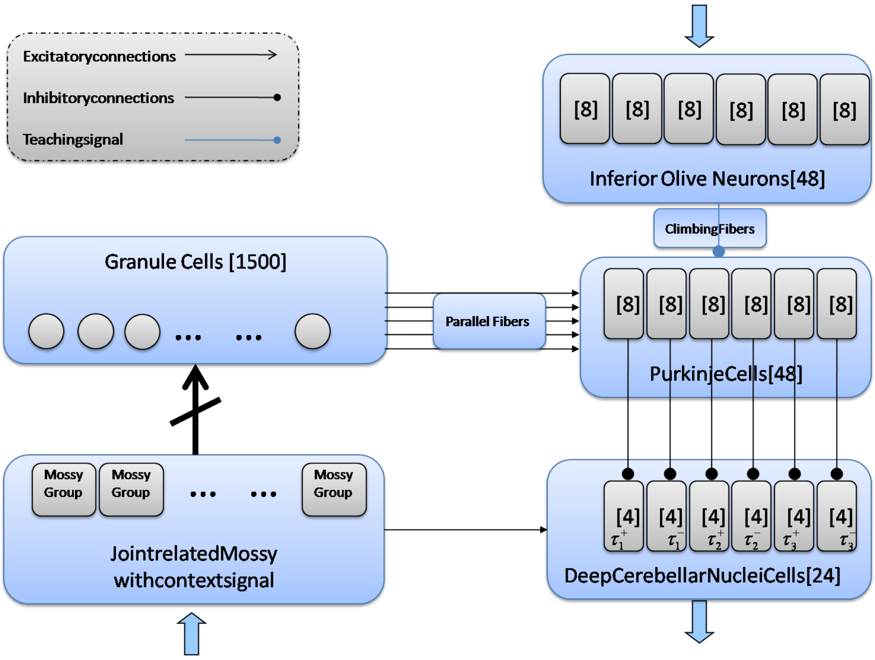


Figure 4.7: Second-approach cerebellum model diagram - Cerebellum configuration inputs encoding the movement (desired arm states, actual sensorimotor signals and context-related signals) are sent (upward arrow) through the parallel fibers (PFs). Error-related inputs are sent (upper downward arrow) through the climbing fibers (CFs). Outputs are provided by the deep cerebellar nuclei (DCN) cells (lower downward arrow).

the *label or any captured property* of the object, for instance assuming information captured through visual sensory system) feeds these dedicated-eight-grouped fibers.

- *IC model.* Implicit Context encoding approach (240 joint-related fibers); the mossy fiber layer consists of 12 groups of twenty fibers and delivers the actual and desired joint velocity and position information. It uses the base desired proprioceptive configuration and adds three groups of fibers encoding actual joint positions and other three groups encoding actual joint velocities. The implicit contextual information is conveyed using these six groups of fibers. The actual position and velocity *helps* the cerebellum to recognize where and how far from the ideal (desired) situation it is. These deviations implicitly encode a *context-like* representation based on sensorimotor complexes.

- *EC and IC*. Explicit and Implicit Context encoding approach (16 context-related fibers plus 240 joint-related fibers). It uses the base desired-proprioceptive and incorporates also IC and EC architectural specifications. Thus, this mossy fiber layer is a combination of the EC and IC models described above.

The main aim of searching a proper mossy layer configuration is to exploit the capability of the granule layer for generating a sequence of active neuron populations without recurrence. This sequence is able to efficiently represent the passage of time (representation of different time passages are related with different input signals). Our system takes advantage of this spatiotemporal discrimination of input signals for learning different contexts.

As indicated in section 4.3.3.2, afferent mossy fibers are randomly connected to granule cells, on average, four mossy fibers [204] per granule cell. When an input signal pattern arrives at the mossy fibers, a spatiotemporal activity pattern is generated and the population of active neurons in the granule layer changes in time according to this received input. In order to evaluate the non-recurrence in this activation train, the following correlation function (Eq. 4.5) is used [288]:

$$C(t_1, t_2) = \frac{\sum\limits_i f_i(t_1) f_i(t_2)}{\sqrt{\sum\limits_i f_i^2(t_1)} \sqrt{\sum\limits_i f_i^2(t_2)}} \tag{4.5}$$

Where f_i corresponds to the instantaneous frequency of the $i-th$ neuron (frequency measured within a 20ms time window). The numerator calculates the inner product of the population vector of active neurons at times t_1 and t_2, and the denominator normalizes the vector length. $C(t1, t2)$ takes values from 0 to 1; 0 if two vectors are complementary, 1 if two vectors are identical. To facilitate the production of accurate corrective terms, different input signals shall generate different spatio-temporal activity patterns. The following correlation function is used to evaluate this point as indicated in Equation 4.6.

$$C(t_1, t_2) = \frac{\sum\limits_i f_i^{(1)}(t_1) f_i^{(2)}(t_2)}{\sqrt{\sum\limits_i f_i^{(1)2}(t_1)} \sqrt{\sum\limits_i f_i^{(2)2}(t_2)}} \tag{4.6}$$

Where $f_i^{(1)}$ and $f_i^{(2)}$ denote the activities of the $i-th$ neuron at time t under different input signals (1 and 2 respectively).

The left panels in Figs. 4.8a, 4.8b, and 4.8c show the similarity index using a t_1xt_2 matrix within the active granular population at t_1 and t_2. A wide white band, surrounding the main diagonal, points out that the index decreases monotonically as the distance $[t_1 - t_2]$ increases. That means a one-to-one correspondence between the active neuron population and time. This implies that a dynamically active neuron activity changing can represent the passage of time.

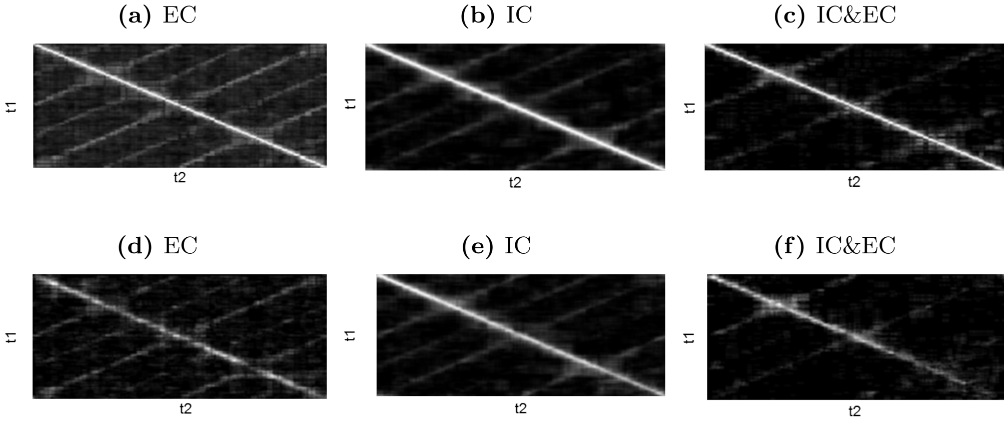


Figure 4.8: Similarity indices for a spatiotemporal activity between two activity patterns using EC, IC, and IC&EC configurations - The values of indices are represented in gray scale; black 0, white 1. Left side panels show a white diagonal band indicating a proper generation of a time-varying granular activity population. EC presents a mean gray value of 0.18, IC leads to a mean gray level of 0.101, and IC&EC leads to a mean gray value of 0.074. Right side panels show similarity indices for two contexts. The darker the matrix is, the better uncorrelated activity patterns are. EC presents a mean gray value of 0.024, IC leads to a mean gray value of 0.096, and IC&EC achieves 0.044.

The right panels in Figs. 4.8d, 4.8e, and 4.8f show how different input signals can be discriminated by different activity patterns. The values of the similarity index are small suggesting that the two represented activity patterns are independent of each one. Actual and desired entries of the IC configuration vary during time leading to a richer codification within a single context, while EC only uses desired entries varying along the trajectory execution. On the other hand, EC gives a better granular activity codification between contexts by using its specific contextual signals. IC has no specific entries helping to distinguish activity patterns when using two different contexts. IC&EC takes advantages from both configurations; it uses the context and the position/velocity entries to produce a better time-varying granular activity population.

4.3.3.3 Spike-Time dependant plasticity

The studied cerebellar model only includes synaptic plasticity at the PF-PC connections. The changes of the synaptic efficacy for each connection are driven by presynaptic activity (spike-timing-dependent plasticity) and are instantaneous. In our model, since there are delays in the transmission of joint torque values and joint position measurements, the trajectory error measurements (which are used to calculate the teaching signal) reach the cerebellum with a 100ms delay. This means that the learning mechanism must adapt to provide corrective torque predictions.

$$k(t) = e^{-(t-t_{postsynapticspike})} sin(t - t_{postsynapticspike})^{20} \tag{4.7}$$

This plasticity has been implemented including LTD and LTP mechanisms in the following way:

- LTD produces a synaptic efficacy decrease when a spike from the IO reaches a PC as indicated in Eq. 4.8a. The amount of efficacy which decreases depends on the previous activity arrived through the PF (input of the cerebellar model). This previous activity is convolved with an integral kernel as defined by Eq. 4.7. This mainly takes into account those PF spikes which arrived 100ms before the IO spike (see Fig. 6). This correction is facilitated by a time-logged *eligibility trace* [20, 24, 159, 258], which takes into account the past activity of the afferent PF. This trace aims to calculate the correspondence in time between spikes from IO (error-related activity) and the previous activity of the PF which is supposed to have provoked this error signal. The eligibility trace idea stems from experimental evidence showing that a spike in the climbing fiber afferent to a Purkinje cell is more likely to depress a PF-PC synapse if the corresponding PF has been firing between 50 and 150 ms before the IO spike (through CF) arrives at the PC [24, 159].

- LTP produces a fixed increase in synaptic efficacy each time a spike arrives through a PF to the corresponding PC as defined by Eq. 4.8b. With this mechanism, we capture how an LTD process, according to neurophysiologists studies [171], can be inverted when the PF stimulation is followed by spikes from the IO or by a strong depression of the Purkinje cell membrane potential.

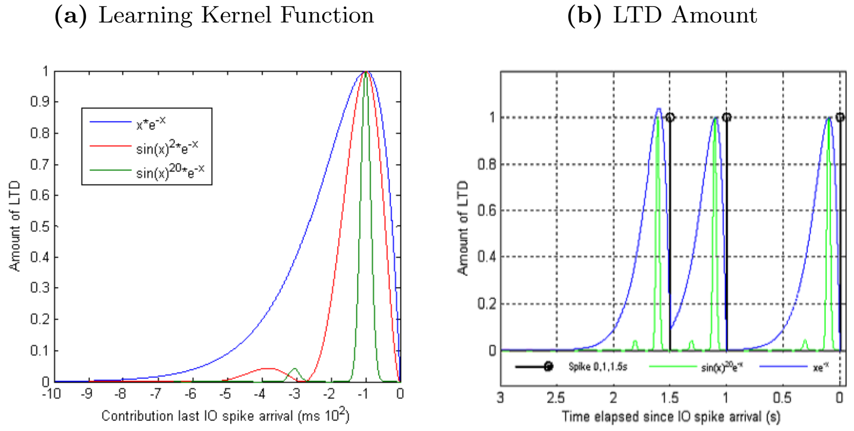
$$LTD.\forall i, \Delta w_i = -\int_{-\inf}^{IOSpike} k(t - t_{IOspike})\delta_{GRspike-i}(t)\,dt \tag{4.8a}$$

$$LTP.\Delta w_i = \alpha \tag{4.8b}$$

The strength of these two mechanisms needs to be tuned to complement and compensate each other. These biological LTP-LTD properties at PF-PC synapses have been tried to be emulated in different fields, i.e. in the adaptive filter [94] theory by using the heterosynaptic covariance learning rule of Sejnowski [239] or in the adaptive control theory by using the Least Mean Square learning rule [279]. Different alternative temporal kernels are shown in Fig. 6. The sharper the integral kernel peak is, the more precise the learning becomes. On the other hand, this leads us to a slower synaptic weight adaptation. However, LTP can lead the weight recruitment to be compensated by future IO activity. This situation drives us to faster synaptic weight saturation where LTP can hardly carry out the weight recruitment for future IO activity. After the main peak in the correlation kernel, a second marginal bump can be seen, as a consequence of the mathematical model used for modeling the correlation engines. The chosen mathematical models of the kernel allow accumulative computation in an event-driven engine, avoiding the necessity of integrating the whole correlation kernel each time a new spike arrives. Therefore, these correlation models are computationally efficient in the framework of an event-driven simulation scheme, such as EDLUT [227], but they suffer from this second marginal peak that can be considered noise in the weight integration engine. This is indicated in the equations 4.8.

4.3.3.4 The Teaching Signal of the Inferior Olive

The crude inverse dynamics controller generates motor torque values for a rough control, but the long delays in the control loop prevent the online correction of the trajectory in a fast reaching task using a classical controller with a continuous feedback. In the



(a) Learning Kernel Function

(b) LTD Amount

Figure 4.9: LTD integral kernel - (a) Representation of a basic integral kernel (xe^{-x}) which has a rather wide peak that makes PC synaptic weights to decrease more prominently and a more complex integral kernel ($sin(x)^{20}e^{-x}$) which has a sharper peak. (b) This plot shows the amount of LTD at a particular synapse depending on the IO spike arrival time elapsed since the PF spikes for different integral kernels. The figure includes a comparison between the basic integral kernel (xe^{-x}) and a more complex integral kernel ($sin(x)^{20}e^{-x}$) which has a peak 100ms after the input spike. The PC receives three spikes through a particular CF at times 0.0, 1, and 1.5s.

studied control model, the trajectory error is used to calculate the teaching signal. This teaching signal follows Expression 4.9.

$$\epsilon_{delayed_i} = K_{p_i} \cdot \epsilon_{position_i} + K_{v_i} \cdot \epsilon_{velocity_i} \tag{4.9a}$$

$$\epsilon_{position_i} = (q_{desired_i} - q_{real_i}) \cdot [(t + t_{pred})_i - t_i] \tag{4.9b}$$

$$\epsilon_{velocity_i} = (\dot{q}_{desired_i} - \dot{q}_{real_i}) \cdot [(t + t_{pred})_i - t_i] \tag{4.9c}$$

Where $i = 1, 2, 3...joints$, $K_{p_i} \cdot \epsilon_{position_i}$ represents the product of a constant value (gain) at each joint K_{p_i} and the position error in this joint (difference between desired joint position and actual joint position $(q_{desired_i} - q_{real_i})$). $K_{v_i} \cdot \epsilon_{velocity_i}$ represents the product between a constant value (gain) at each K_{v_i} joint and the velocity error in this joint (difference between desired joint velocity and actual joint velocity $(\dot{q}_{desired_i} - \dot{q}_{real_i})$).

The IO neurons synapse onto the PCs and contribute to drive the plasticity of PF-PC synapses. These neurons, however, fire at very low rates (less than 10 Hz), which appears problematic to capture the high-frequency information of the error signal of the task being learned. This apparent difficulty may be solved by their irregular or chaotic firing [153, 157, 168]. This is a very important property, which has the beneficial consequence of statistically sampling the entire range of the error signal over multiple trials (see below). Here, we implemented this irregular firing using a Poisson model [173] for spike generation. The weight adaptation was driven by the activity generated by the IO, which encoded the teaching signal into a low frequency probabilistic spike train (from 0 to 10 Hz, average 1 Hz) [168, 238].

We modeled the IO cell responses with probabilistic Poisson processes. Given the normalized error signal $\epsilon(t)$ and a random number $\eta(t)$ between 0 and 1, the cell fired a spike if $\epsilon(t) > \eta(t)$; otherwise, it remained silent [24]. In this way, on one hand, a single spike reported accurately timed information regarding the instantaneous error; and on the other hand, the probabilistic spike sampling of the error ensured that the whole error region was accurately represented over trials with the cell firing almost 10 spikes per second. Hence, the error evolution is accurately sampled even at a low frequency [40]. This firing behavior is similar to the ones obtained in physiological recordings [168].

LTD and LTP play complementary roles in the model inference process. The long-term potentiation (LTP) implemented at the PF-PC synapses was a non-associative weight increase triggered by each input cell spike [171]. The long-term depression (LTD) was an associative weight decrease triggered by spikes from the inferior olive [136, 138]. This model of LTD uses a temporal kernel, shown in Fig. 4.9a, which correlates each spike from the IO with the past activity of the parallel fiber [159, 224, 253]. Correlation-based LTD allows the adjustment of specific PF-PC connections to reduce the error according to the IO activity. When IO spikes are received, the synaptic weights of the PF-PC connections are reduced according to the temporal-correlation kernel and to the activity received through the PF. In this way, we reduce the probability of production of simple spikes by PC due to the activity coming from the PFs through these specific connections. Therefore, the IO effectively modulates the spatio-temporal corrective spike patterns. In this model, a learning state in the cerebellum (PF-PC weights) can be seen as a bi-dimensional function which relates each PF and PC combination with their corresponding synaptic weight (Fig. 4.10c). Physiologically, the time-matching of the desired and actual joint states can be understood by the fact that the trajectory error would be detected at the level of the spinal cord, through a direct drive from the gamma motorneurons to the spinal cord [51].

4.3.4 Quantitative performance evaluation

We have carried out several experiments to evaluate the capability of cerebellar architecture to select and abstract models using different cerebellar topologies and learning parameters. In these experiments, objects which significantly affect the dynamics and kinematics of the base plant model have been manipulated to evaluate the performance of different cerebellar configurations.

By means of the first cerebellar approach (see paragraph 4.3.3.1) we have studied the following issues:

1. How LTD and LTP need to be balanced to optimize the adaptation performance.

2. How the temporal-correlation kernel (integral kernel) works even in the presence of sensorimotor delays.

3. How the same learning mechanism can adapt the system to compensate different deviations in the basic model dynamics (due to manipulating objects of different weights).

Finally, using the second cerebellar approach (see paragraph 4.3.3.2) we have also studied how interpolation/generalization can be naturally done for different plant+object models which have not been used during the training process. We divided the experiments into the following groups:

1. Cerebellar input configuration including only context-related signals (and desired arm states) (EC).

2. Cerebellar input configuration including only sensorimotor representation (IC) (i.e. desired and actual arm states).

3. Cerebellar input configuration including conjointly sensorimotor and context-related signals. (IC&EC).

For this purpose, we have used a set of benchmark trajectories that we repeat in each iteration and evaluate how learning adapts the GR-PC weights to tune accurate corrective actions in the control loop.

The learning performance is characterized by using four estimates calculated from the Mean Absolute Error (MAE) curve [280]. For the calculation of the MAE of a trajectory execution, we have considered the addition of the error in radians produced by each joint independently.

1. Accuracy Gain (estimates the error reduction rate comparing the accuracy before and after learning). This estimate helps to interpret the adaptation capability of the cerebellum when manipulating different objects, since the initial MAE for each of these manipulated objects may be different (see Eq. 4.10).

   $$\text{Accuracy Gain} = MAE_{initial} - \left(\frac{1}{n}\sum_{i=0}^{n} MAE_{final,i}\right) \quad n = 30 \tag{4.10}$$

   Where $MAE_{initial}$ represents the MAE before starting the learning process and $MAE_{final,i}$ represents the MAE after i learning trials.

2. Final Error (average error over the last 30 trials) (see Eq. 4.11).

   $$\text{Final Error} = \frac{1}{n}\sum_{i=0}^{n} MAE_{final-i} \quad n = 30 \tag{4.11}$$

3. Final Error Statibility (average of standard deviation over the last 30 movement trials) (see Eq. 4.12).

   $$\text{Final Error Stability} = \frac{1}{n}\sum_{i=0}^{n} \sigma(MAE_{final-i}) \quad n = 30 \tag{4.12}$$

4. Error Convergence Speed (number of samples to reach the final error average) (see Eq. 4.13).

   $$\text{Error Convergence Speed} = j; \quad \text{where} \quad MAE_j \leq \frac{1}{n}\sum_{i=0}^{n} MAE_{final-i} \quad 0 < j \leq final \tag{4.13}$$

4.4 Results

The experiments that we have carried out show that this system, composed by a control loop and a cerebellum-like network topology, is able to inference the presented dynamic and kinematic models, in order to reproduce accurately the target trajectories. In addition to this, several parameters have significantly modified the learning capabilities and the performance (accuracy and speed) of this learning. As expected, a balanced rate between LTD and LTP has been needed in order to avoid the weight saturation (high rate of LTP) or the activity absence at parallel fibers (predominant LTD). These results suggest that a learning rule kernel as the one used in this work can correlate the activity at parallel fibers and the IO activity (representing the error in the trajectories). On the other hand, the experiments using the granular layer model show how the presence of contextual information such as the system actual state or a signal indicating the presented model (but remaining partially unknown its physical characteristics) could enhance the performance when carrying out the movements. In the following subsections, we further present all these results.

4.4.1 Influence of the learning mechanisms at the parallel fibbers

4.4.1.1 LTD vs LTP trade-off

At the beginning of the learning process (before the connection weights are adjusted), the spikes received from the input fibers excite the DCN cells producing a *bias correction* term on the motor commands. The role of the cerebellar PF-PC-DCN loop is to specifically inhibit this bias term according to a spatio-temporal pattern that is inferred during movement executions and to further compensate other deviations generated by the manipulation of different objects or other elements affecting the dynamics of the initial *arm plant* (without object under manipulation). The PF-PC-DCN loop transmits an activity pattern which is adapted taking into account the teaching signals provided by the IO (described in the previous section).

In the first simulations, the arm is manipulating a 1kg mass object. This mass significantly affects the dynamics of the arm+object. Therefore the actual trajectory (without corrective support) deviates significantly from the target trajectory. We have studied how the cerebellar module compensates this deviation building a corrective model.

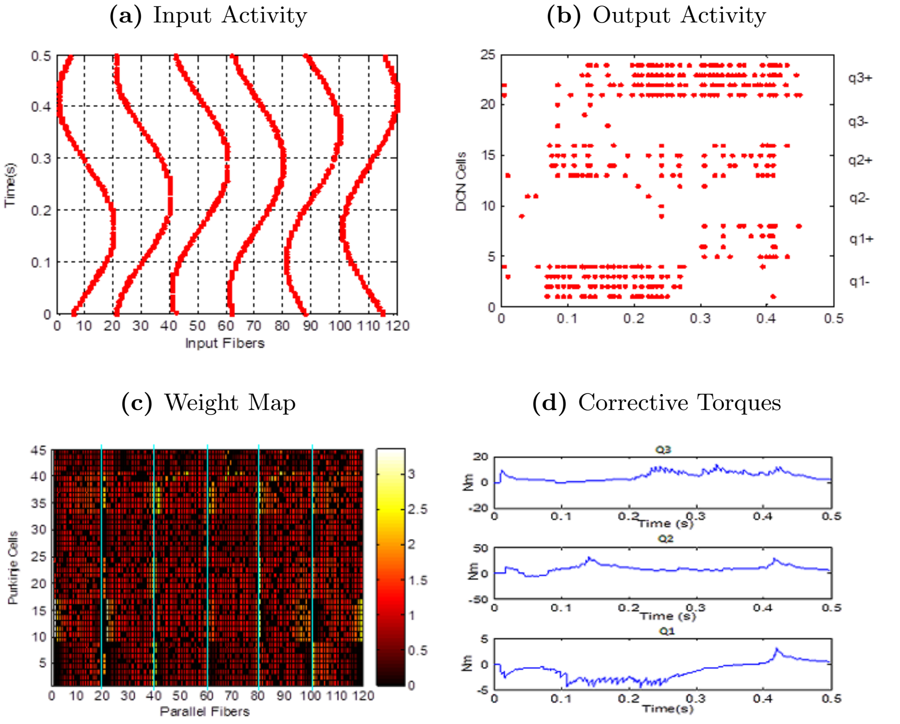


Figure 4.10: Cerebellum state after 300 trajectory-learning iterations - (a) Input activity at the cerebellum. The input layer produces a set of spikes after the transformation from analog domain into spike domain These spikes are transmitted directly by PF. This activity (desired positions and velocities) keeps on constant during all iterations. (b) DCN output activity generated by those synaptic weights. Error corrections are accomplished by changes in the activity of PCs that in turn influence the activity of the DCN [221], which afterwards is translated into analog torque correction signals. Each group of 4 DCN cells encodes the positive or negative part of a joint corrective torque. The more activity the positive/negative group has, the higher/lower corresponding corrective torque is generated. (c) PC-PF synaptic weight representation. In x-axis, we can see the source cells (PFs). In y-axis, target cells (PCs) are shown. Dark colors represent lower synaptic weights, thus, the corresponding DCN cells are more active. We can see 6 well-defined rows, each row represents weights related with the positive and negative torque output of the three joints (q3, q2, and q1), and 6 well-defined columns (that has been remarked with blue lines and are related with the input activity of the PF corresponding to the desired position and velocity for the three joints). (d) Output torque after analog transformation from the DCN output spikes. These corrective torque curves have a profile strongly related with the number of DCN cells assigned per joint; thus, increasing the quantity of DCN per joint will generate a smoother corrective profile.

Fig. 4.10c illustrates how the corrective model is acquired through learning and structured in distributed synaptic weight patterns. When the arm moves along a target trajectory, different input cell populations are activated. They produce a temporal signature of the desired movement. Meanwhile, the IO continuously transfers trajectory-error estimates (teaching signals) which are correlated with the input signature. In Fig. 4.10c, the system adaptation capability is monitored. This helps to interpret how the corrective model is continuously structured. Similar monitoring experiments in much simpler scenarios and smaller scale cell areas are being conducted in neurophysiologic studies [67] to characterize the adaptation capability of neurophysiologic systems at different neural sites.

When manipulating heavy objects which do not properly fit the basic plant model, the followed trajectory drifts from the desired one before learning. This deviation is more prominent when the desired trajectory changes direction (see Fig. 4.10a) due to the arm's inertia. After learning, the cerebellum output counteracts this inertia generating higher torques during these changes of the desired trajectory direction (see Fig. 4.10d). The weight matrix learned by the cerebellum reflects the moments when higher corrective torque values are supplied. By looking at Fig. 4.10b and Fig. 4.10d, we can see that the higher corrective torque is produced when the desired trajectory joint coordinates change direction. This occurs in the peaks of the sine waves describing the desired trajectory and corresponds to the activation of the higher and lower input fibers of each block (left and right side of the six weight columns -blue lines- of Fig. 4.10c). To generate a high corrective torque, the cerebellum must unbalance the magnitude of the positive and negative parts of the joint corrective output (q+ and q- in Fig. 4.10b) which is calculated from the activity of the DCN cells. These DCN cells are grouped by joints. A higher activity affecting positive corrections in a joint produces higher corrective torque. Since PCs inhibit DCN cells, a low PC activity is required for a high DCN activity and vice versa. To obtain a low PC activity, low PF-PC weights are required, which corresponds to small dark squares in Fig. 4.10c. Small light squares correspond to high values of the weights. Looking at both sides of the six weight columns of Fig. 4.10c, we can observe how the weight values alternate between high and low in adjacent rows which alternately encode the weights corresponding to the positive and negative parts of each joint corrective torque.

(a) MAE Global LTP=0.006 & LTD=0.075

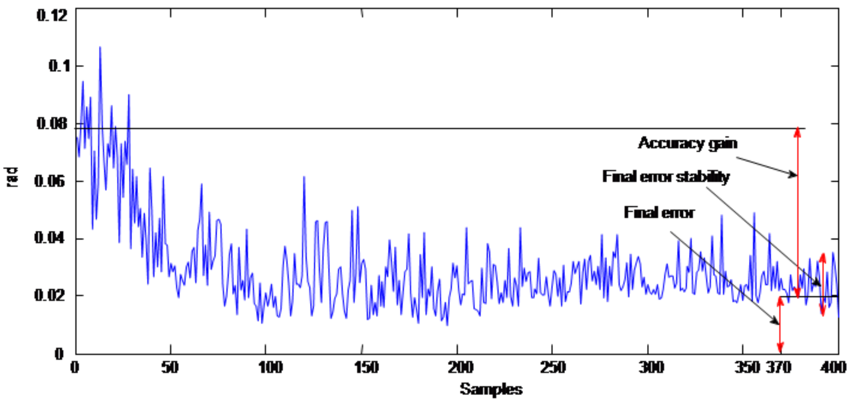
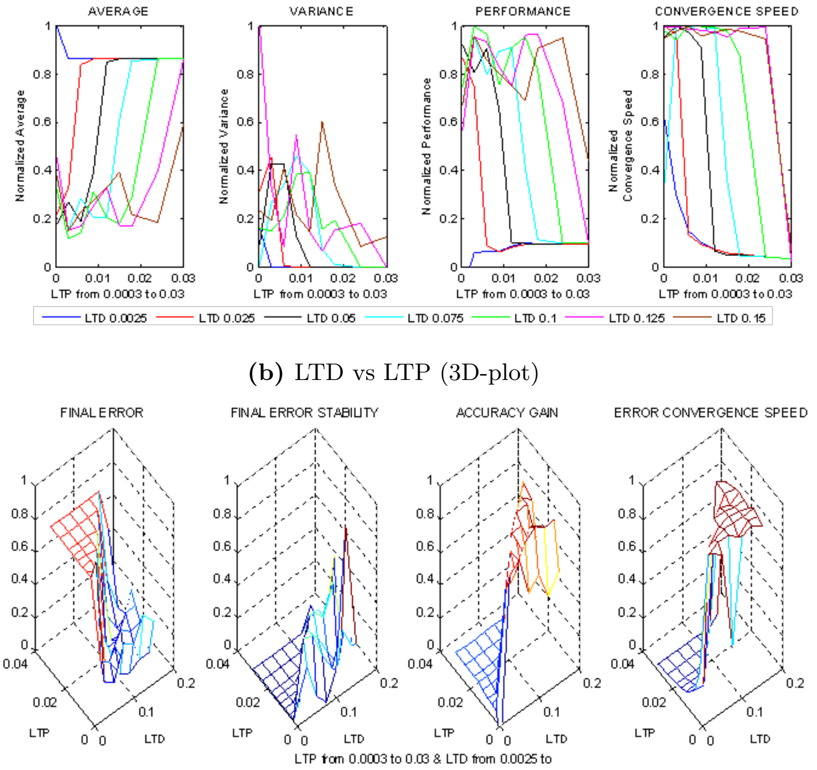


Figure 4.11: Learning estimates - We evaluate the learning performance using four estimates extracted from the Mean Absolute Error (MAE) curve. 1) Accuracy Gain, 2) Final Error, 3) Final Error Stability, and 4) Error Convergence Speed.

During the learning process, the corrective model is captured in the PF-PC connections. In this way, the movements become more accurate, the error decreases and therefore, also the activity of the IO is reduced. This allows the learned models to become stable once the error reaches appropriate values.

We have carried out 70 simulations of a complete training process, where each training process consists of 400 trajectory executions and each trajectory execution is carried out in 1 second simulation time (i.e. the whole system is executed 28000 times). During each of these training processes, the obtained error in each trajectory execution decreases until it reaches a final stable value. The obtained Mean Absolute Error (MAE) of a single complete training process is shown in Fig. 4.11. We have tested this learning process with different LTD and LTP components to evaluate how they affect the adaptation capability of the system. From each of these training processes (with different LTD and LTP values), we obtain the performance estimates defined above (Accuracy Gain, Final Error, Final Error Stability, and Error Convergence Speed). These performance estimates characterize the adaptation mechanism capability.

As it is shown in Figs. 4.12a. and 4.12b, both LTP and LTD must be compensated. Low LTD values combined with high LTP values cause high weight saturation. This can be seen in Fig. 4.12b. in which 3D *Final normalized Error* values of the first figure are represented in a high flat surface corresponding to high errors. We also have



(a) LTD vs LTP

(b) LTD vs LTP (3D-plot)

Figure 4.12: Learning characterization - During the learning process, the movement error decreases along several iterative executions of trials of an 8-like trajectory benchmark. (a) Using these four estimators, we can evaluate how LTP and LTD affect the learning process. We have conducted multiple simulations with different LTD-LTP trade-offs to characterize the learning behavior. The goal of an appropriate learning process is to achieve a high accuracy gain and a low and stable final error. (b) In the same way, LTP and LTP effect in the learning process showing 3-dimensional surfaces.

a flat surface close to zero in Fig 4.12b (3D *Final normalized Error Stability* figure); the cerebellum output is totally saturated. Therefore, when LTP-LTD trade off is unbalanced (LTP dominating LTD) the system adaptation capability is low, leading to high error estimators and useless high stability. On the other hand, when high LTD values are combined with low LTP values, this causes low weight saturation. In Fig. 4.12b, 3D plots, we see a good *Final Average Error* and a good *Accuracy Gain* and *Convergence Speed* but very unstable output. This is also indicated by the error variance figure estimates which are high in this LTD-LTP area. A compensated LTD-LTP setting drives us to a high accuracy gain and also, to a low and stable Final Error with high Convergence Speed. For instance, if our LTD choice is 0.075, our LTP must be lower than 0.015 to achieve a proper stable learning mechanism. In all the following simulations, we have fixed the LTD and LTP parameters to these values. Therefore, we illustrate how different model deviations (by different object manipulations) can be compensated with a fixed and balanced temporal-correlation kernel and how this correction loop works even in the presence of different sensorimotor transmission delays.

4.4.1.2 Corrective learning in presence of signal delays

The cerebellum-like structure previously described works even with sensorimotor delays by means of the temporal-correlation kernel which determines the amount of LTD to be applied. This is summarized in Fig. 4.13. The results have been obtained after performing 4 simulations (each one for different delay setups) of 400 trajectory executions each. On the other hand, this temporal-correlation kernel remains robust not only with different unbalanced delays but also with a non-perfect matching between sensorimotor delays and the temporal correlation kernel peak as it is shown in Fig. 4.14. These results have been obtained after performing 5 simulations (each one for a different time deviation) of 400 trajectory executions each.

This robustness is achieved because the scheme is using desired coordinates (positions and velocities) which remain stable across different trials. Nevertheless, with delays mismatching (between learning kernel inherent time shift and sensorimotor delays) over 70ms, this scheme becomes unstable, while with delay mismatches below 70 ms the obtained performance is farily good as shown in Fig. 4.14b)

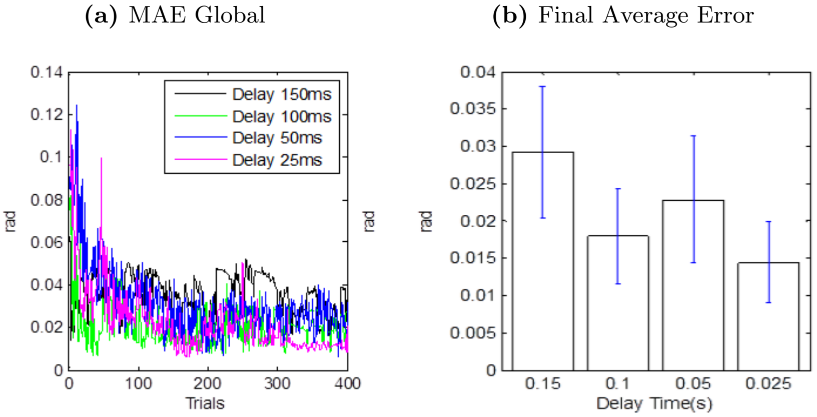


Figure 4.13: Temporal-correlation kernel for different sensorimotor delays (delays from 25ms to 150ms have been tested) - We have adjusted the correlation kernel peak position to match (see Fig. 4.9) the sensorimotor delays of the control loop illustrated in Fig. 4.4. As it is shown, the delay value does not affect to a large extent the obtained performance. The final average error is nearly constant in these different simulations.

4.4.1.3 Adaptation to several dynamical models

The presented cerebellum micro-structure and the long-term plasticity, side by side, facilitate internal model inference. The cerebellum model adapts itself to infer a new model by using error signals which are obtained when manipulating this new object. We study the ability of the cerebellar architecture to infer different corrective models for dynamics changes on a base manipulator model.

	MAE improvement		**Stability improvement**	
	200 trials	400 trials	200 trials	400 trials
0.5kg	40.4%	49%	82%	60.1%
1kg	64.6%	64.5%	49.6%	42.1%
1.5kg	72.5%	74.4%	46.1%	26.4%
2kg	78.6%	79.3%	26.4%	25.1%

Table 4.2: Performance improvement using several dynamical models - We have measured the MAE after 200-trial and 400-trial learning. In the same way, the average standard deviation over 30 trials has been calculated after 200 and 400 trials. Values are expressed as the percent regarding the initial value (in the earliest iterations).

Under normal conditions, without adding any extra mass to the end of the effector

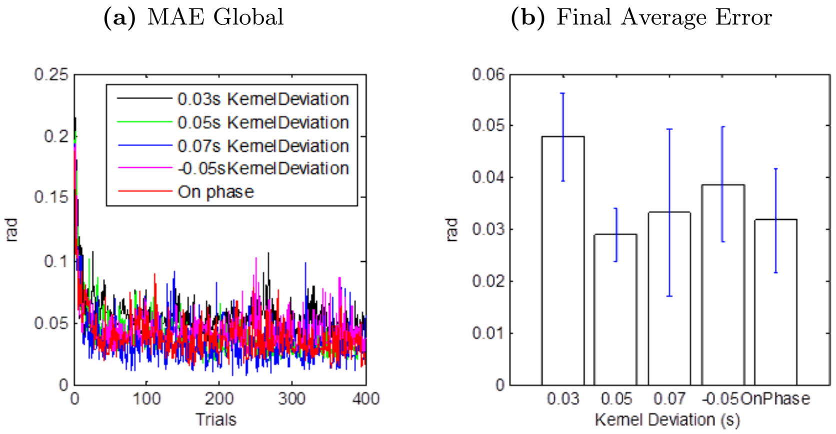


Figure 4.14: Temporal-correlation kernel behavior with different deviations between sensorimotor delays and the kernel peak (deviation from -50ms to 70ms have been tested) - We have evaluated different deviations between the correlation kernel peak position (see Fig. 4.9) and the sensorimotor delays of the control loop illustrated in Fig. 4.4. As it is shown, despite the kernel peak does not exactly match with sensorimotor delays, the cerebellum still works and the final average error keeps on constant. The cerebellum is able to correlate the delayed sinusoidal inputs and the non-in phase peak kernel.

(arm), the crude inverse dynamics model calculates rough motor commands to control the arm-plant. In contrast, under altered dynamics conditions, the motor commands are inaccurate to compensate for the new undergone forces (inertia, etc.), and this leads to distortions in the performed trajectories. During repeated trials, the cerebellum learns to supply the corrective motor commands when the arm-plant model dynamics differs from the initial one. These corrective motor commands are added to the normal-condition motor commands. Then, improved trajectories are obtained as the learning process goes on. The cerebellum gradually builds up internal models by experience and uses them in combination with the crude inverse dynamics controller. This cerebellum adaptation is assumed to involve changes in the synaptic efficacy of neurons constituting the inverse dynamics model [155], as it is shown in our simulation results (Fig. 4.15).

The performance results of the followed trajectory have been evaluated during 400 trajectory executions manipulating different objects attached at the end of the last segment of the arm of 0.5kg, 1kg, 1.5kg, and 2kg. Fig. 4.15 illustrates the performed trajectory for each simulation with an object of a different mass. Fig. 4.16 shows

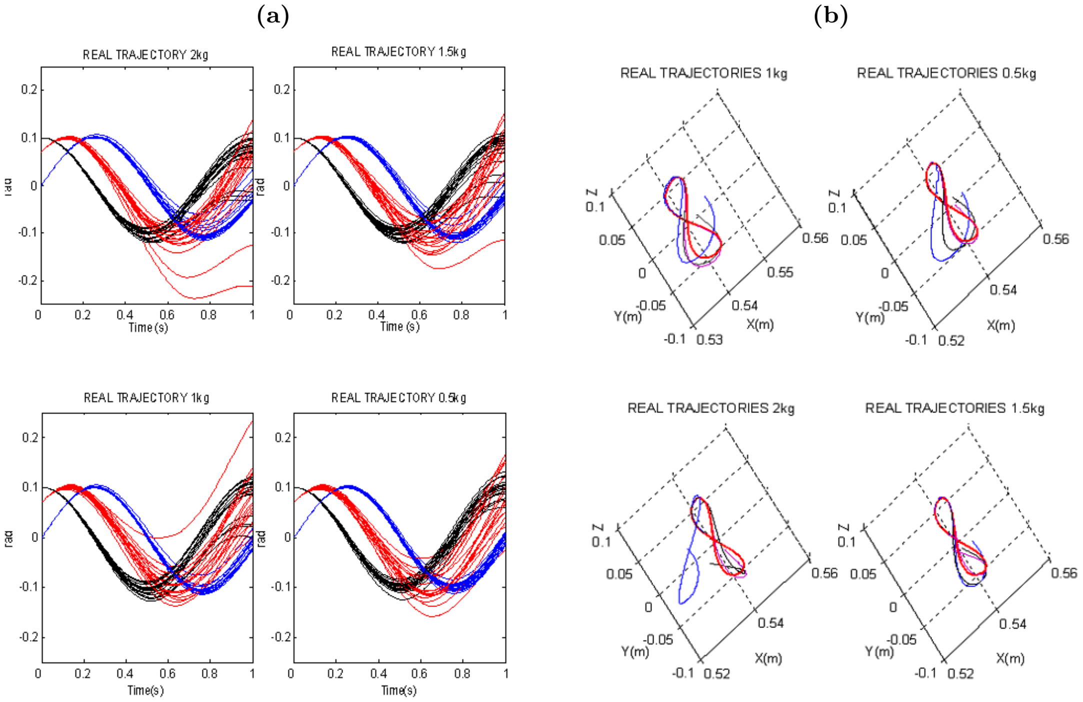
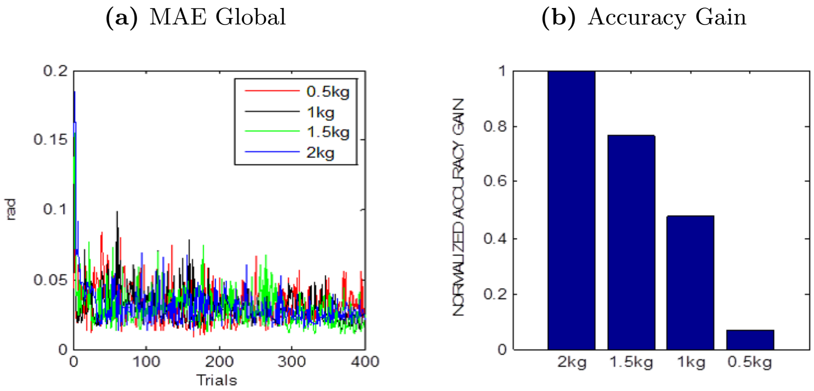


Figure 4.15: Learning the corrective models for the 8-like target trajectory when manipulating objects with different masses (2kg, 1.5kg, 1kg, and 0.5kg) - (a) 3-joint value representation for the performed trajectory. The three joints are shown. The followed trajectory is shown each 25 trials during a 400-trial complete learning process. (b) 2D representation of the performed trajectory (Desired trajectory in red; in blue. initial trial; in black, trial number 200; and in cyan, final trial).

how the cerebellar model is able to learn/infer the corrective dynamics model for the different objects. Further detail about the performance improvement can be found in table 4.2. The error curves of Fig. 4.16a (where each sample represents the error along one 8-like trajectory) show how the control loop with the adaptive cerebellar module is able to significantly reduce the error during the training process. Fig. 4.15 shows that manipulating heavier objects means that the starting error is higher, since the arm dynamics differ from the original one to a larger extent. Therefore, the cerebellum learns to supply higher corrective torques, which makes a bigger difference between the initial and final error. This makes the Accuracy Gain estimate higher than in the other cases. On the other hand, for improving the global Accuracy Gain, higher forces have to be counteracted to follow the desired trajectory.


(a) MAE Global

(b) Accuracy Gain

Figure 4.16: Learning Performance when manipulating different objects (0.5kg, 1kg, 1.5kg, and 2kg) during 400-trial learning processes - (a) MAE evolution. Learning occurs on a continuous basis providing incremental adaptability throughout the simulation time. (b) Accuracy Gain.

4.4.2 Influence of the information convergence at the granular layer

EC cerebellar input As commented in previous sections, the Explicit Context Encoding Approach (EC) uses a set of MFs to explicitly identify the context, assuming that they carry information provided by other areas of the central nervous system (such as vision which helps to identify the correct model to be used) or even cognitive signals. Therefore, a specific group of context-based MFs becomes active when the corresponding context is present. In this way, when a certain context becomes active, a GR population is pre-sensitized due to the specific context-related signals. We have randomly combined the sensor signals (desired position and velocity) of the different joints and the context-related signals (in the MF to GR connections) allowing granule cells to receive inputs from different randomly selected MFs (at the network-topology definition stage). In order to explicitly evaluate the capability of these signals to separate neural populations for different object models, each granule cell has four synaptic input connections: three random MF entries which deliver joint-related information and one MF which delivers context-related signals. In this case, we have evaluated the capability of the cerebellum model to efficiently use these context-related signals to learn to separate models when manipulating objects of different weights or different kinematics (deformation in the robot-plant end-segment) (Fig. 4.17).

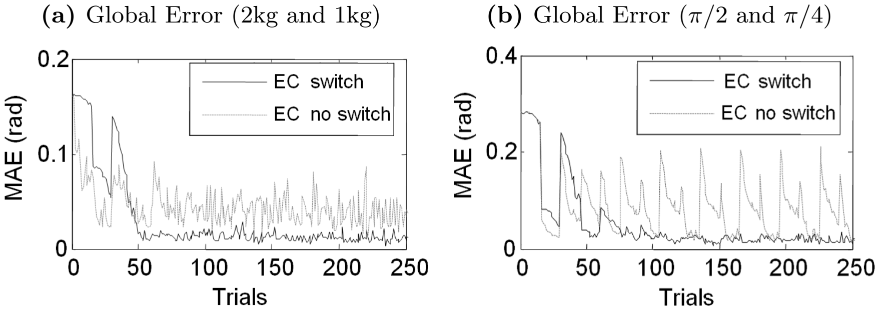


Figure 4.17: Multi-context simulation with changes in dynamics and kinematics using EC cerebellar input - Each sample represents the MAE evolution (sum of error at different joints) for a trajectory execution (trial) during learning with no context-related signals and with explicit context-related signals. (a) Manipulating two different loads with and without context signals. Explicit context signals reduce 68.31% the final average error and 70.73% the final standard deviation. (b) The equivalent end-segment of the arm has been rotated in certain angles $\pi/2$ and $\pi/4$. The corrective torque values should compensate these different deviations in each context (with and without activated context signals). Explicit context switching signals reduce 62.04% the final average error and 26% the final standard deviation.

IC cerebellar input In this section, we define an Implicit Context Encoding Approach (IC), where no context-identifying signals are used. The sensor signals (actual position and velocity of the robot) implicitly encode (through MFs) the context during object manipulation. We have randomly combined the sensor signals (position and velocity) of the different joints (in the MF to GR connections) allowing granule cells to receive four inputs from different randomly selected MFs. The context models are distributed along cell populations. These cell populations are dynamically changing during the learning process (because the actual trajectory changes as corrective torque values are learned and integrated). Each time a new context is activated, the specific neural population is tuned due to the slightly different sensorimotor signals during the trajectory execution. The context switching in IC is done automatically and learning is carried out in a non-destructive manner, learned contexts are not destroyed (Fig. 4.19).The fact that IC transitions do not need explicit contextual information may indicate that this configuration allows interpolation between different learned contexts. This capability is explored by making the cerebellum learn two contexts alternately and then, presenting a new intermediate context (Fig. 4.20).

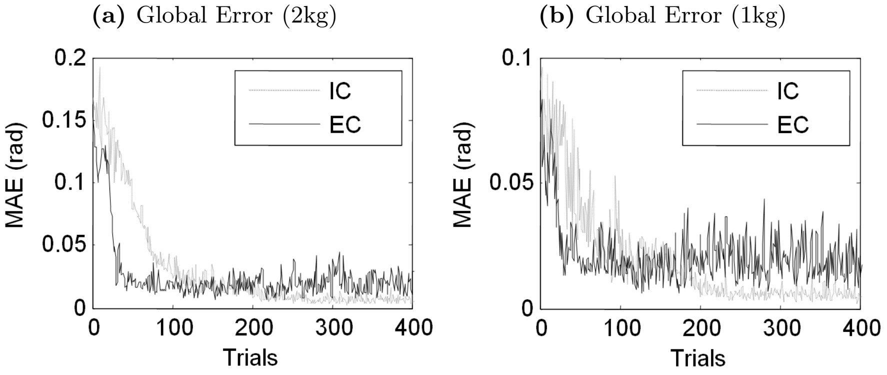

(a) Global Error (2kg) **(b)** Global Error (1kg)

Figure 4.18: Single-context simulation using EC and IC cerebellar input - (a) and (b) Manipulation of objects of different loads (2kg/1kg) without context signals. Each sample represents the MAE for a trajectory execution (trial)

As shown in Fig. 4.18 and 4.19, although EC has a faster convergence speed, IC presents a lower final error (0.007 rad. average final error in IC against 0.018 rad. in EC) and a more stable behavior (0.002 rad. of standard deviation in IC against 0.006 rad. of standard deviation in EC) after the learning process.

Accuracy gain, final error average, and final standard deviation are similar in IC and EC. EC develops a better inter-context transition. Comparing EC with IC in a dynamic context switching experiment, we obtain context switching error discontinuities 47.6% larger and a standard deviation 24.7% higher in the EC explicitly cancelling context switching signals than in the IC configuration (Fig. 4.17a vs. Fig. 4.19a). This highlights the importance of actual sensorimotor signals efficiently used in the IC configuration, compared to EC which only used desired states during manipulation.

Finally, comparing EC with IC in a kinematic context switching experiment, we obtain context switching error discontinuities 32.85% larger and a final standard deviation 16.71% higher in the EC without activating context switching signals than in the IC configuration (Fig. 4.17b vs. Fig. 4.19b explicit context signals are efficiently used in EC configuration).

IC plus EC cerebellar input In this section, we evaluate how the previous EC and IC input representations are complementary. In this case, the cerebellar architecture includes both inputs.The MFs arriving in the cerebellum encode the desired states, the actual states(positions and velocities), and also, context signals which identify the

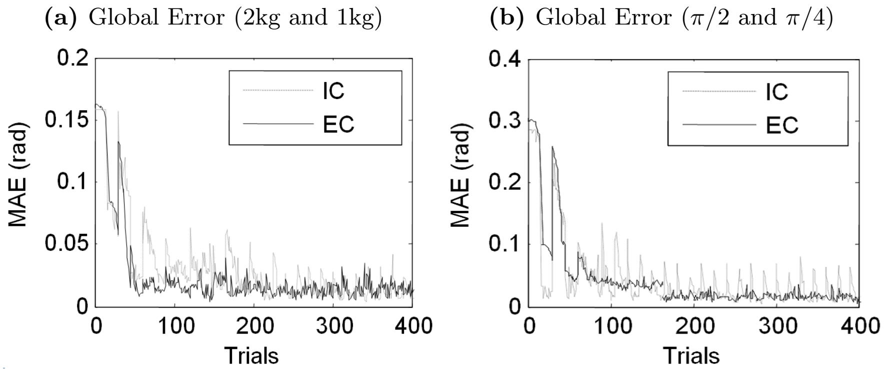
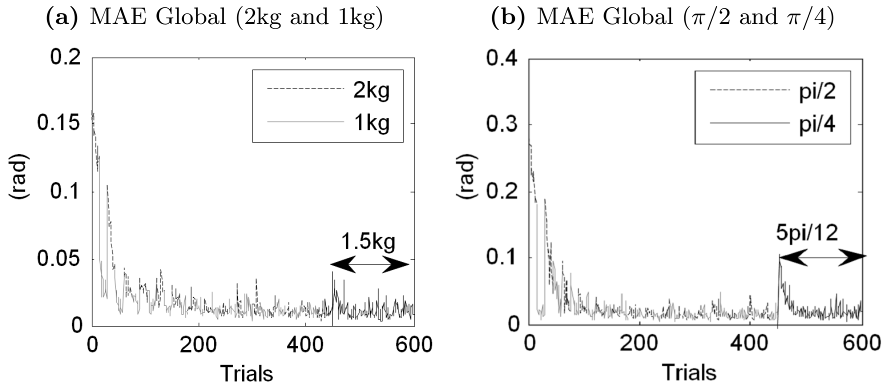


Figure 4.19: Multi-context simulation with changes in kinematics and dynamics using IC and EC cerebellar input configurations - (a) MAE evolution during the learning process in EC and IC with dynamics-changing contexts. Two contexts with different loads are manipulated, switching every 15 trials. (b) MAE evolution for EC and IC configurations and two contexts with different ending deformation (kinematics change), switching every 15 trials.

current contexts. Fig. 4.21 shows the learning curves by using EC and IC compared to use only one of those information sources (only EC or only IC). Both dynamics correction (see Fig. 4.21a) and kinematics correction (see Fig. 4.21b) achieve a faster and accurate adaptation by taking advantage of the correlation between the teaching signals and inputs encoding the context and the actual state (positions and velocities).

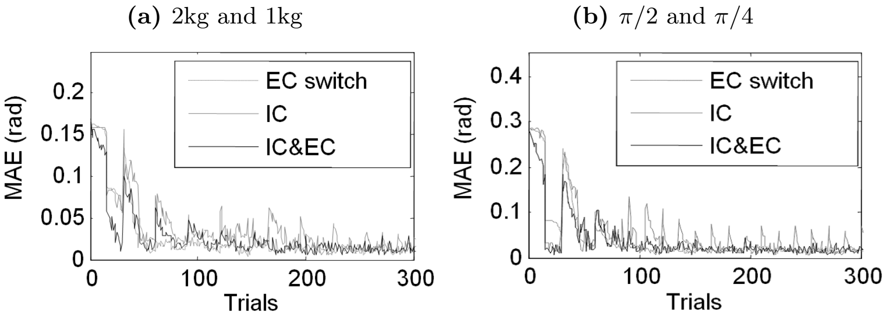
In addition to this, in Fig. 4.22a IC&EC uses the pre-learned synaptic weights obtained in previous contexts to deal with a new payload. Nevertheless, sensorimotor state signals feeding MFs drive fast to a new contextual adaptation. The kinematics interpolation is not efficient (Fig. 4.22b); interpolation across kinematics changes is not an easy task (not linear).

IC&EC configuration also becomes robust against incongruent external context-related signals(for instance,extracted from vision). As we show in Fig. 4.22c, during each epoch,the external context signal changes do not match the actual object switching (i.e. the external context signal does not remain constant while manipulating a 2kg object and it does not do it either when using a 1kg object). Thus, *context*1 value in the first 2kg- 15-trial-context equals A and *context*2 value in the first 1kg-15-trial-contexts equals B. In the following 15-trial-context switching trials, values A and B are interchanged. The incoming external contextual information is not congruent but,


(a) MAE Global (2kg and 1kg)

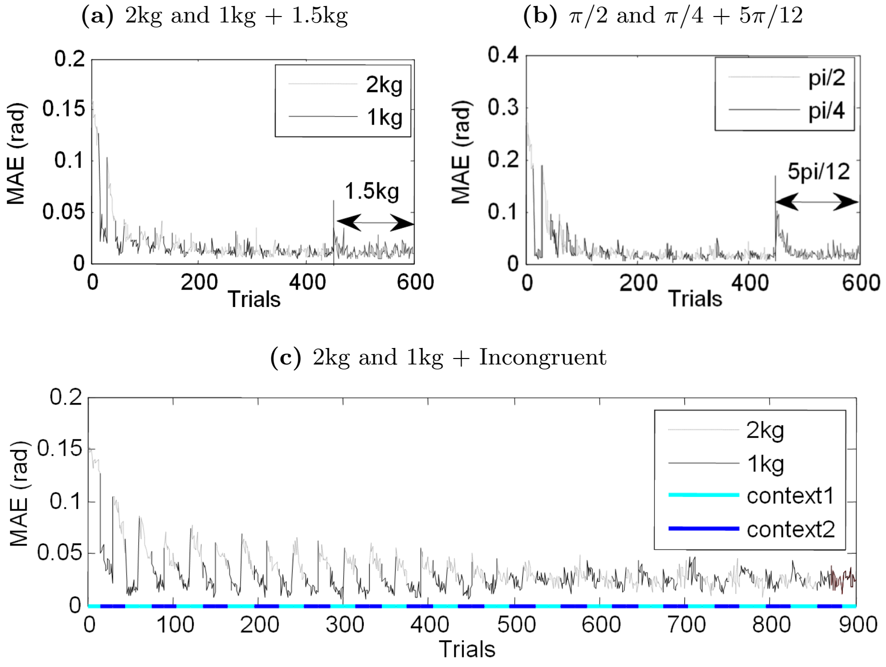
(b) MAE Global ($\pi/2$ and $\pi/4$)

Figure 4.20: Multi-context simulation with changes in kinematics and dynamics using IC cerebellar input. Interpolation capability - (a) After 450 trials of 15 iterations per context (2kg/1kg added alternatively to the robot arm), a new 1.5kg context is presented to the cerebellum. (b) After 450 trials of 15 iterations per context (the end-segment of the robot arm includes different rotations: $\pi/2$ and $\pi/4$ angles alternatively), a new $5\pi/12$ context is presented to the cerebellum.


(a) 2kg and 1kg

(b) $\pi/2$ and $\pi/4$

Figure 4.21: Multi-context simulation with changes in kinematics and dynamics using EC, IC, and IC&EC cerebellar input - (a) Dynamics correction task with different loads in the robot arm. (b) Kinematics correction task with different deviations in the end-segment of the robot arm.


(a) 2kg and 1kg + 1.5kg

(b) $\pi/2$ and $\pi/4$ + $5\pi/12$

(c) 2kg and 1kg + Incongruent

Figure 4.22: Multi-context simulation by interpolating in kinematics and dynamics using EC, IC, and IC&EC cerebellar input - (a) A 1.5kg load is fixed to the end-segment of the robot. (b) A 5pi/12 rotation in the end-segment of the robot is presented. (c) IC&EC configuration is able to avoid non-congruent contextual signals. Context-related input signals are indicated with highlighted colors in the x axis of the plot.

thanks to sensorimotor state signals (actual position and velocity of IC configuration); the cerebellum is able to deal with these *misleading* external signals.

4.5 Discussion

The study described in this chapter has focused on how a cerebellar-like adaptive module operating together with a crude inverse dynamics model can effectively provide corrective torque to compensate deviations in the dynamics of a base plant model (due to object manipulation). This is relevant to understand how the cerebellar structure embedded in a biologically plausible control loop can infer internal corrective models when manipulating objects which affect the base dynamics model of the arm. The spiking neural cerebellum connected to a biomorphic robot plant represents a tool to study how the cerebellar structure and learning kernels (including time-shifts for compensating sensorimotor delays) provide adaptation mechanisms to infer dynamics correction models towards accurate object manipulation. Concretely, we have evaluated how a temporal-correlation kernel driving an error-related LTD and a compensatory LTP component (complementing each other) can achieve effective adaptation of the corrective cerebellar output. We have shown how the temporal-correlation kernel can work even in the presence of sensorimotor delays. However, considering the results obtained for several sensorimotor delays, we can state that the desired trajectory must be coded using a univocal population coding in each time step, that is, the codification of the desired position/velocity during the trajectory must be different for each point of the trajectory. And thus, as our cerebellar structure can adaptively generate any suitable output for each trajectory-point codification, the delay of the sensorimotor pathways is not remarkably relevant, even if this delay does not match the intrinsic compensatory delay of the learning integration kernel.

In this simple cerebellar-like structure, we have shown how the representation of the cerebellar weight matrix corresponding to the PF-PC connections can be interpreted in terms of the generated corrective torque (which in turn is a direct consequence of this representation). This allows us to study the performance of this corrective model storage and how the changes of the arm dynamics (manipulating different object) are inferred on different synaptic weight patterns.

We have also shown how LTD and LTP need to be balanced with each other to achieve high performance adaptation capabilities. We have studied the behavior of these two complementary adaptation mechanisms. We have evaluated how the learning behaves when they are balanced and also when they are in value ranges in which one of them dominates saturating the adaptation capability of the learning rule. We have evaluated how well balanced LTD and LTP components lead to an effective reduction of error in manipulation tasks with objects which significantly affect the dynamics of the base arm plant.

We have used a simplified version of the cerebellum to focus on the way that the cerebellar corrective models are stored and structured in neural population weights. This is of interest to inform neurophysiologic research teams to drive attention to potential footprints of inferred models within the PF-PC connections.

We have proposed a new simple biologically plausible cerebellar module which can abstract models of manipulated objects that significantly affect the initial dynamics and also kinematics of the plant (arm+object), providing corrective torque values towards more accurate movements.The results are obtained from object manipulation experiments.This new cerebellar approach, with two representations, receiving context-related inputs (EC) and actual sensory robot signals (IC) encoding the context during the experiments, has been studied.The IC&EC cerebellar configuration takes advantage of both configurations which complement each other. Smoother inter-context transitions are achieved at a fast convergence speed. It allows the interpolation of new contexts (different loads under manipulation) based on previously acquired models. Moreover, a good learning curve profile in long-term epochs can be achieved and finally, the capability of *overcoming* misleading external contextual information, making this cerebellar configuration robust against incongruent representations (Fig. 4.22c), is remarkable.

Furthermore, the results obtained with this kind of cerebellar architecture are coherent with the experiments [5, 113].Therefore, when both representations congruently encode the context, they shall complement each other, while when they are incongruent, they interfere with each other. This is so because in the implemented cerebellar architecture, context classification and model abstraction tasks are carried out in a distributed manner. No pre-classification process is executed to disambiguate incongruent context identification. In our approach, we have also evaluated how sensorimotor representation can overcome incongruent incidental context-related signals (i.e. sensorimotor representation dominating a context-related incongruent signal).

In a classical machine learning approach, disambiguation is usually explicitly done through a classification module (decision making) that can be tuned to adopt a winner-takes-all strategy and leads to a single context model to be recalled even in this incongruent context representation. In biological systems, this kind of pre-classification (disambiguation) mechanisms may be processed in other nervous centers, although it may reduce the interpolation and generalization capabilities of the cerebellar model presented.

As future work, we will study how to dynamically optimize the LTD-LTP integration kernel, instead of a single, stable, and balanced LTD-LTP kernel, we will evaluate the capability of improving the adaptation mechanism, shifting this balance to acquire the corrective models faster and then, decrease the plasticity once an acceptable performance is reached. This approach can optimize the learning capability of the system. We will also develop further real-time interfaces between analog signals and spiking neurons (between the robot and the EDLUT simulator) to perform simulations with real robots and new cerebellar architectures working in a manipulation task scenario in which granular layer, Golgi cells and stellate cells will be included. This will be addressed in a starting EU project (REALNET). The neuron models, cerebellar models, and adaptation mechanisms will be available at the EDLUT simulator site to facilitate the reproduction of the presented work.

5

Discussion

5.1 Future works

In future work, the time-driven implementation by using a variable and independent time step for each neuron will be addressed, speeding up and enhancing the precision of this method. This evolved simulation strategy has previously been tested in other simulation tools [124] with impressive results. In addition to this, the spread of the high-performance clusters of computers provide a promising chance to highly increase the performance of the EDLUT simulation environment. However, the difficulty of leading with event queues distributed over different nodes make this implementation a challenge. In the same line of upgrading the EDLUT simulator, the integration with other generic tools to describe the model in different abstraction levels (cell, neural network, system,...) such as PyNN or NeuroML could make easier the learning curve to potential EDLUT users and would allow the simulation models being largely reusable from well-known tools such as NEURON or GENESIS.

The natural evolution of this work leads us to take advantage of temporal filtering capabilities at the granular layer within a whole cerebellar model in the framework of a control task. Although these capabilities are clear after this work, how these properties could improve a cerebellar model requires finding an explanation on the signal codification at both mossy and parallel fibers. Recent simulation and experimental studies [27] might help to clarify this debate (at least partially).

On the other hand, the simulation of the biologically detailed cerebellar models still represent a open issue. The inclusion of synaptic plasticity at most of the cerebellar synapses (where experimental studies have shown the existence of these mechanisms) means a powerful tool capable of automatically setting the whole system to the actual environment in order to get the best performance. This mechanisms could be specially relevant in the granular layer, where plasticity might lead the synaptic weights to those values which enhance the sensitivity rebound at the frequency range of the incoming inputs.

A similar approach could be followed in other parts of the cerebellum, such as the connections $MF \rightarrow DCN$ or $PC \rightarrow DCN$. This plasticity, and the plasticity between $GrC \rightarrow SC \rightarrow PC$ has been proven recently to influence the consolidation of the learning. These future works will be addressed in a recently started EU project (REALNET), and all these neuron models, cerebellar models, and adaptation mechanisms will be available at the EDLUT simulator site to facilitate the reproduction of this and other future studies.

5.2 Publication of results

Our research work has been evaluated in the framework of international conferences and scientific journals (with impact factor on the JCR). In short, the scientific production can be summarized as: 2 journal paper published (Q1 -Top 3- and Q2 in their respective JCR categories), 2 accepted papers (Q1 journals in their JCR categories, in fact one of them is also Top 3 at its respective categories), 3 other papers under review (Q1 and Q2 in their respective categories in JCR). 5 International conferences (COGSYS 2008, ICJNN 2010, COGSYS 2010, and IWANN 2011).

5.2.1 International Peer-review Journals

1. D'Angelo, E.; Koekkoek, S.K.E.; Lombardo, P.; Solinas, S.; Ros, E.; *Garrido, J.*; Schonewille, M.; De Zeeuw, C.I.: "Timing in the Cerebelllum: Oscillations and Resonance in the Granular Layer". Neuroscience, 162, 805–815, 2009. doi:10.1.1.154.4873.

2. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Coenen, O. J. M. D.; Ros, E: "Cerebellar input configuration towards object model abstraction in manipulation tasks". IEEE transaction on neural networks, 22(8), 1321–1328, 2011.

doi:10.1109/TNN.2011.2156809. (Top 3 Journal on Computer Sciences, Hardware & Architecture, 1st quartile on Computer Sciences, Artificial Intelligence, Theory & Methods, and Engineering, Electric & Electronic).

3. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Tolu, S.; Ros, E.: "Adaptive cerebellar spiking model in a bio-inspired robot-control loop". International journal on neural systems, 21(5), 385–401, 2011. doi:10.1142/S0129065711002900. (1st quartile on Computer Sciences, Artificial Intelligence).

4. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Coenen, O. J. M. D.; Ros, E: "Cerebellar-like corrective-model abstraction engine for robot movement control". IEEE Transaction on system, man, and cybernetics - Part B, 41(5), 1299–1312, 2011. doi:10.1109/TSMCB.2011.2138693. (Top 3 Journal on Automation & Control Systems, and Computer Sciences, Cybernetics. 1st quartile on Computer Sciences, Artificial Intelligence).

5. *Garrido, J. A.*; Carrillo, R. R.; Ros, E.; D'Angelo, E.: "The impact of synaptic weights on spike timing in the cerebellar input layer: a simulation study". Under Review on Neural Computation.

6. Luque, N. R.; *Garrido, J. A.*; Ralli, J.; Laredo, J. J.; Ros, E.: "From Sensors to Spikes: Evolving Receptive Fields to Enhance Sensory Motor Information in a Robot-Arm Scenario". Under Review on International journal on neural systems.

7. Tolu, S.; Vanegas, M.; Luque, N. R.; *Garrido, J. A.*; Ros, E.: "Bio-inspired Adaptive Feedback Error Learning Architecture For Motor Control". Under Review on Biological Cybernetics.

5.2.2 International Conferences

1. Carrillo, R. R.; *Garrido, J.*; Ros, E.; Tolu, S.; Boucheny, C.; Coenen, O. J. M. D. (2008): "A real-time spiking cerebellum model for learning robot control". International Conference On Cognitive Systems (COGSYS 2008).

2. D'Angelo, E.; Van der Smagt, P.; Solinas, S.; Jorntell, H.; Ros, E.; *Garrido, J.* (2010): "Realistic circuit modelling: large-scale simulations of the cerebel-

lar granular layer". International Conference On Cognitive Systems (COGSYS 2010).

3. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Ros, E. (2010): "Cerebellar spiking engine: Towards object model abstraction in manipulation". International Joint Conference on Neural Networks (ICJNN 2010). doi:10.1109/IJCNN.2010.5596531.

4. *Garrido, J. A.*; Carrillo, R. R.; Luque, N. R.; Ros, E.: "Event and time driven hybrid simulation of spiking neural networks". International Work-Conference on Artificial Neural Networks (IWANN 2011). Advances in Computational Intelligence. Lecture Notes in Computer Science, 6691, pp. 554–561. Springer, Heidelberg (2011). doi:10.1007/978–3–642–21501–8_69.

5. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Ros, E.: "Context Separability Mediated by the Granular Layer in a Spiking Cerebellum Model for Robot Control". International Work-Conference on Artificial Neural Networks (IWANN 2011). Advances in Computational Intelligence. Lecture Notes in Computer Science, 6691, pp. 537–546. Springer, Heidelberg (2011). doi:10.1007/978–3–642–21501–8_67.

5.2.3 Other dissemination sources

The dissemination of the knowledge represents one of the key points in the development of an European project. Since this work has been carried out during the evolution of the SENSOPAC and REALNET projects (see section 5.3), an remarkable effort has been done in order to release the advances emerging from these projects. Thus, the EDLUT simulation environment was released as an open-source application under GPLv3 license. This allows the usage of this software for any research or industrial project with almost no restriction. Therefore, after this required exhaustive documentation process of the source code, it was released on the website [98] in April, 2009. Since them, several groups have shown their interest in using EDLUT on their own projects and we have supported all their questions and suggestions. The following statistics quantitatively show the awoken interest in the scientist community:

- The website has received more than 11000 visits since its release in 2009.

- The presentation of the EDLUT software has been downloaded about 2000 times, while the source code of the last released version has been downloaded more than 800 times.
- Several groups have used EDLUT for their developments such as University of Pavia (Egidio D'Angelo and Sergio Solinas), University of Pierre and Marie Curie at Paris (Angelo Arleo, Luca Leonardo Bologna and Jean Baptiste Passot), University of Erasmus (Chris de Zeeuw and Jornt de Gruijl), University of Lund (Henrik Jörntell and Carl Fredrik Ekerot), SICS (Martin Nilson) and other researchers such as Boris Barbour (CNRS), Olivier Coenen and Mike Arnold.

The release of the source code of EDLUT was favorably received in the media, and specially, in the open-source community, as we show in the following headlines published in the media:

1. "Move to create less clumsy robots"at BBC News.
2. "EDLUT human nervous system researches disease and tests medicine" at Compute Scotland.
3. "Artificial simulator of the nervous system created for research into diseases" at Science daily.
4. "Development of an artificial simulator of the nervous system to do research into diseases" at Eureka Alert.
5. "Liberado simulador del sistema nervioso: EDLUT" at Oficina de Software Libre UGR.
6. "EDLUT: un simulador neuronal" at Microsiervos.
7. "Desarrollan un simulador del sistema nervioso para investigación biomédica" at Universidades.
8. "Simulador artificial del sistema nervioso humano" at `http://www.solociencia.com/medicina/09052105.htm`.
9. "Desarrollan un simulador del sistema nervioso" at Orion.

10. "La UGR participa en la construcción de un cerebelo artificial" at Universia.

11. "La Universidad de Granada desarrolla un simulador del sistema nervioso humano" at Ideal.

5.3 General Scientific Framework

This scientific work has been done and partially funded by the European Projects SENSOPAC: SENSOrimotor structuring of Perception and Action for emergent Cognition (IST-028056), and REALNET: Realistic Real-time Networks: computation dynamics in the cerebellum (IST-270434). This has represented an excellent collaborative framework with diverse research groups at other European Universities and research institutions. The presented work represents the major contribution of the University of Granada in this SENSOPAC consortium. Therefore, a high responsibility in obtaining timely the planned results was necessary during the whole investigation process. Besides the required technical reports, presentations for EU scientific reviews, a final demo (proof of concept) was required and implemented jointly with the University Pierre and Marie Curie in Paris and the DLR (Munich). The effort invested in this and another demo is significant but allows easy evaluating of the system capabilities and also facilitates the dissemination of results beyond a pure scientific scenario, towards industrial future collaborations and also impact in the media (newspapers, TV, etc.).

Since these kind of projects require a multidisciplinary approach, this work mainly presents the results from the biological systems point of view. This is specially relevant in the last part of this work (see chapter 4), where we use a cerebellum-like simulated architecture in order to manipulate tools with a robotic arm. In addition to this, this work implies dealing with robotic system (developing a robotic arm simulator, studying the biologically plausible control loops, or the conversion from/to spiking signals). All these other tasks have been mainly accomplished by Niceto Luque at the University of Granada. In addition to this, a lot of efforts were accomplished in designing and implementing an easy-to-use and intuitive simulator (taking into account the difficulties of understanding this kind of systems per se) capable of running in different environments and with a wide variety of controlled devices. This has required high collaborative and coordinated work with the working team.

5.4 Main contributions

We now summarize the main contributions of the presented work:

- An event-driven simulator based on look-up tables has been widely upgraded to accomplish the main requirements of a real-time neural network simulator capable of interfacing with a huge quantity of external systems. In addition to this, a Java-based graphical user interface has been developed as well, which facilitates the usage of this simulation tools for beginner researchers.

- The event-driven simulation environment has been widely improved to make it capable of natively simulating time-driven methods (alone or in conjunction with event-driven methods).

- Event-driven model (or more generally hybrid networks) showed good performance and accuracy when working with low rates of activity. In the opposite, time-driven model would be preferred when the characteristics of the experiment produce high neural activity. However, a hybrid simulation system can be convenient in the simulation of biological systems which present different levels of activity.

- A computational model of the granular layer has been developed and analyzed showing that coincidence detection at this layer was critically regulated by the feed-forward inhibitory loop ($MF \rightarrow GoC \rightarrow GrC$).

- In addition to this, the granular layer computational model has shown sensitivity rebound capabilities (recently predicted on the basis of various experimental data). The feed-forward dis-hinibitory loop ($MF \rightarrow GrC \rightarrow SC \rightarrow GoC$) proved to be the most important one by affecting both the sensitivity rebound amplitude and the inter-stimulus interval which maximizes network activity.

- This granular layer model also showed oscillations of electrical activity in the β/γ band as a response to random input activity which had been observed in the granular layer of the cerebellum in vivo and in vitro.

- A cerebellar model embedded in a control loop with a crude inverse dynamics model is presented. This model can effectively provide corrective torque to compensate deviations in the dynamics of a base plant model (due to object manipulation).

- This adaptive model has shown how LTD and LTP at parallel fibers need to be balanced with each other to achieve accurate adaptation capabilities. We have evaluated how well balanced LTD and LTP components lead to an effective reduction of error in manipulation tasks with objects which significantly affect the dynamics of the base arm plant.
- An evaluation of the influence of the input sensory signals to the cerebellum is presented: with two representations, receiving context-related inputs (EC) and actual sensory robot signals (IC) encoding the context during the experiments. The IC&EC cerebellar configuration takes advantage of both configurations which complement each other. Smoother inter-context transitions are achieved at a fast convergence speed.
- This work shows how this model with IC&EC cerebellar input signals provides the capability of *overcoming* misleading external contextual information, making this cerebellar configuration robust against incongruent representations.

Capítulo 6

Introducción en castellano

En la tercera edición del libro *Principles of Neural Science* (que podría considerarse como un clásico de la neurociencia moderna) se afirma que "Tal vez la última frontera de la ciencia —su último desafío— es entender las bases biológicas de la conciencia y los procesos mentales por los que percibimos, actuamos, aprendendemos y recordamos". Pero este reto continua vigente, y podría considerarse uno de los principales desafíos para el siglo XXI. El conocimiento profundo de las bases biológicas de los sistemas neuronales puede aportarnos numerosos beneficios. En primer lugar, una mayor comprensión del cerebro podría hacer posible el desarrollo de tratamientos innovadores para las patologías relacionadas con el Sistema Nervioso Central. En segundo lugar, el cerebro humano puede ser considerado el mejor sistema de procesamiento de la información debido a su fiabilidad, precisión, rendimiento y capacidad de almacenamiento. Ser capaz de *emular* la forma de procesar la información de los sistemas biológicos podría permitir el diseño de una nueva generación de arquitecturas de procesamiento capaz de sobrepasar los límites de los sistemas actuales.

6.1 La neurociencia computacional

Desde esa perspectiva, y en buena medida debido a las crecientes capacidades de los recursos de cómputo, la neurociencia computacional se ha posicionado en los últimos años como una prometedora rama de la neurociencia. Neurociencia computacional se refiere al uso de modelos matemáticos y computacionales para el estudio de los sistemas neuronales. El diseño de modelos matemáticos y cuantitativos ha sido un componente clave de la investigación en neurociencia desde hace muchas décadas. De hecho, uno de los logros más celebrados en el campo — el *modelo de Hodgkin-Huxley* para la generación de potenciales de acción [126] en 1952 — supuso un hito del enfoque cuantitativo y significó un importante impulso para el desarrollo de nuevas herramientas. Además, gran parte de lo que se entiende acerca de la funcionalidad de los sistemas biológicos visuales, auditivos y olfativos, así como las bases neuronales del aprendizaje y la memoria, han sido obtenidos en buena medida gracias a los modelos matemáticos y computacionales desarrollados. Sin embargo, es justo decir que, hasta hace poco, el modelado computacional representaba sólo una pequeña parte del esfuerzo investigador en neurociencia, que tradicionalmente ha estado dominada por los estudios experimentales (de hecho, la mayor parte de lo investigadores en neurociencia solían tener una fuerte formación en sistemas biológicos, dejando de lado los sistemas de cómputo). El reciente movimiento hacia el modelado computacional ha abierto nuevas líneas de investigación, y permitió la investigación de sistemas biológicos más allá de lo accesible mediante el estudio experimental. Incluso podríamos decir más, la aparición de la neurociencia computacional ha traído e integrado nuevas ideas de campos tan dispares como la física, la estadística, la teoría de la información, la teoría de sistemas no lineales y la ingeniería, lo cual proporciona un marco conceptual suficientemente rico como para responder gran parte de las preguntas fundamentales existentes.

La principal motivación para el uso de modelos computacionales es, por supuesto, para entender el comportamiento del sistema objeto de estudio mediante el análisis matemático y la simulación por ordenador. Este es el caso de la neurociencia. Los modelos computacionales se han utilizado tradicionalmente para muchos otros sistemas físicos (tales como los sistemas astronómicos, los flujos de fluidos, dispositivos mecánicos, estructuras, etc.) Sin embargo, y a diferencia de este tipo de sistemas, la aplicación de modelos computacionales en sistemas biológicos — y especialmente en el caso del sistema nervioso — es especialmente adecuado, ya que los sistemas biológicos se pueden ver de forma explícita como procesadores de información. Es por ello que los modelos computacionales en estos sistemas no son meras herramientas para el cálculo o la predicción, sino que a menudo ayudan a aclarar la funcionalidad. En el caso de las neurociencias, esto puede ser visto en términos de dos funciones relacionadas y explicadas por los modelos computacionales: 1) determinar QUÉ hacen las distintas partes del sistema nervioso, y 2) determinar CÓMO lo hacen.

Los estudios experimentales sobre el sistema nervioso en todos los niveles (subcelular, celular y de sistema) son fundamentales para la comprensión de las estructuras anatómicas y los procesos fisiológicos del sistema, pero estas observaciones deben estar organizadas dentro de un modelo coherente de la funcionalidad del sistema. Esto sólo es posible si los elementos conceptuales apropiados para esa descripción funcional están disponibles. Los psicólogos y neurólogos han utilizado tradicionalmente el rendimiento (o su déficit en presencia de patologías), como base para la asignación de funciones a los componentes del sistema nervioso, que ha producido modelos cualitativos y fenomenológicos de gran utilidad. Este es el caso de los experimentos del condicionamiento del parpadeo o del reflejo vestíbulo-ocular, que tradicionalmente han estado estrechamente ligados al estudio del cerebelo. Aunque estos son a menudo suficientes para fines clínicos, sin embargo proporcionan una comprensión muy limitada del sistema completo. Un enfoque alternativo (o complementario) puede ser obtenido mediante la visión del sistema nervioso como un sistema que adquiere, transforma, almacena y usa la información con el objetivo de controlar un sistema extremadamente complejo — el cuerpo — dentro de un entorno complejo y dinámico. Desde este punto de vista, la funcionalidad del sistema emerge desde los fenómenos de bajo nivel de abstracción, tales como la dinámica del potencial de membrana, el flujo de corrientes en las dendritas, la plasticidad sináptica, etc., tanto como la funcionalidad de un ordenador emerge de las corrientes y voltajes en sus componentes.

Al igual que con el ordenador, la funcionalidad emergente del sistema nervioso depende de los fenómenos subyacentes, pero no se puede describir por completo en esos términos. Para entender de verdad esta funcionalidad, es necesario relacionar los fenómenos concretos medidos por medio de experimentos con las abstracciones de procesamiento de la información — y por último a los fenómenos de la cognición y el comportamiento —. Con este fin, el modelado computacional proporciona un formalismo bien desarrollado sobre las señales y la información. A través de este tipo de modelos, se pueden aplicar directamente modelos matemáticos y computacionales al sistema nervioso, dando lugar a una descripción cuantitativa, verificable, funcional y coherente del cerebro en lugar de un modelo cualitativo o un compendio de observaciones. Esta conclusión ya fue predicha por Andrew Huxley (uno de los autores del model del potencial de acción de Hodgkin-Huxley) cuando comentó en su discurso de los premios Nóbel en 1963: "Lo que aprendí de estos trabajos de computación fue la completa inadecuación de nuestra intuición en intentar tratar con un sistema de este grado de complejidad". Esta frase recoge uno de los principios del modelado. Cualquier sistema aspira a recoger absolutamente todos los detalles del sistema real análogo, pero la primera decisión en el desarrollo de un modelo será el análisis de los factores que podrían influir en el comportamiento final, y elegir el nivel de detalle con el que nuestro modelo funcionará.

La misma filosofía debe aplicarse con el fin de modelar el sistema nervioso central. Ésta procesa la información en muchos niveles, desde el molecular hasta las grandes redes con millones de neuronas. Para que un modelo basado en información de la funcionalidad del sistema nervioso funcione es esencial explicar cómo los fenómenos en cada nivel surgen de los niveles inferiores. Por ejemplo, cómo el reconocimiento de objetos en el campo visual se infiere de las señales generadas por las neuronas del sistema visual, o cómo la actividad de las neuronas motoras individuales produce trayectorias extremadamente suaves y precisas. Desafortunadamente, los métodos experimentales a menudo no ofrecen los datos necesarios para ello. En particular, es muy difícil de obtener los datos necesarios para entender cómo las redes de neuronas procesan información colectivamente. La tecnología actual permite el acceso restringido al sistema nervioso vivo, principalmente en los regiones más externas: mediante la alta resolución de los datos intracelulares y extracelulares obtenidas con grabaciones de un único electrodo, y la baja resolución de los datos de la actividad regional a través de resonancias magnéticas funcionales por imágenes (fMRI) y la magnetoencefalografía (MEG). Aunque las matrices de electrodos son ya muy frecuentes, únicamente proporcionan el acceso extracelular a unos cientos de neuronas en el mejor de los casos. Sin embargo, la mayoría de las redes funcionales en ámbitos tales como el hipocampo y el cerebelo (dos de las regiones mejor estudiadas) comprenden desde unos pocos cientos de miles a varios millones de células. El procesamiento de la información en estas redes se produce a través de patrones dinámicos auto-organizados de la actividad que abarca gran parte del sistema [12, 99, 107, 158, 174]. Estos patrones emergentes difícilmente pueden ser entendidos mirando la actividad de células individuales (o incluso de unos cientos de células), de la misma forma que parece complicado entender un libro leyendo letras individuales. De la misma manera, tampoco los datos de gran escala provenientes de estudios fMRI proporcionan la resolución necesaria para observar estos patrones y relacionarlos con las interacciones celulares.

Los modelos computacionales ofrecen una solución a este dilema, permitiendo el estudio de los modelos de red — tan grande como se desee — construido usando modelos neuronales que en sí mismos han sido diseñados sobre la base de datos de células medidos experimentalmente [69, 100, 172, 226, 240]. Estos modelos de red se pueden simular computacionalmente en variedad de situaciones para dar una idea de cómo las redes correspondientes en el cerebro podrían funcionar. Temas específicos tales como el efecto de la plasticidad sináptica, la modulación de las señales externas, o la importancia de los patrones de conectividad pueden ser estudiados, y las hipótesis (que no se pueden probar directamente) puede ser provisionalmente validadas o rechazadas en la simulación. En muchos casos, los modelos se están convirtiendo en una herramienta indispensable en el ciclo de la hipótesis y la experimentación de la neurociencia. Los modelos computacionales permiten a los investigadores probar sus "qué pasa-si" en la simulación, lo que lleva a mejores hipótesis y mejores experimentos diseñados con una mayor probabilidad de éxito. Por supuesto, la calidad de los resultados depende de la calidad de los modelos, pero los modelos han ido progresando con los avances en técnicas numéricas, la potencia de cálculo y los métodos experimentales [25, 37].

A medida que el foco de interés en neurociencia se desplaza desde los fenómenos a la funcionalidad, el modelado computacional también se está utilizando para hacer frente a problemas anteriormente inaccesibles, tales como las bases neurales de la cognición e incluso la conciencia [93, 139, 163]. Cuestiones relacionadas con la representación, la intención y el control ejecutivo están siendo exploradas en el límite entre la neurociencia y la inteligencia artificial, y la comprensión del cerebro como un sistema de procesamiento de información extremadamente sofisticado continúa avanzando de una manera altamente multidisciplinar.

6.2 La importancia de estudiar el cerebelo

El cerebelo (cuyo nombre en latín significa pequeño cerebro) es una región del cerebro que juega un papel fundamental en el control motor. Sin embargo, también está involucrado en algunas funciones cognitivas como la atención y el lenguaje, y probablemente en algunas de las funciones emocionales como el miedo y el placer que regulan las respuestas [282]. Aunque el cerebelo no planifica los movimientos, sí contribuye a la coordinación, la precisión y a la sincronización exacta de estos. El cerebelo recibe información de los sistemas somatosensorial y de otras partes del cerebro y la médula espinal, e integra estas entradas para ajustar de forma precisa la actividad motora. Debido a esta función de ajuste, las enfermedades en el cerebelo no producen parálisis, sino que producen trastornos en los movimientos que requieren gran precisión, el equilibrio, la postura y el aprendizaje motor.

El número estimado de células en el cerebelo también demuestra la importancia de esta parte del cerebro. Las células granulares son, con mucho, las neuronas más numerosas en el cerebro: en los seres humanos, las estimaciones de su número total lo cifran en torno a 50 billones, lo que significa que aproximadamente tres cuartas partes del número total de neuronas en el cerebro son las células granulares del cerebelo. El cerebelo tiene la apariencia de una estructura independiente unido a la parte inferior del cerebro, escondido justo debajo de los hemisferios cerebrales. La superficie del cerebelo está cubierto con finas ranuras paralelas espaciadas, en contraste con las circunvoluciones irregulares amplias de la corteza cerebral. Estos surcos paralelos ocultan el hecho de que el cerebelo es en realidad una capa continua delgada de tejido neuronal (la corteza cerebelosa), bien doblado al estilo de un acordeón, como se muestra en la figura 1.1. Dentro de esta capa delgada hay varios tipos de neuronas con una disposición muy regular, siendo las más importantes las células de Purkinje y las células granulares. Esta red neuronal compleja da lugar a una enorme capacidad de procesamiento de señales, pero casi todas sus salidas se conectan con un conjunto de pequeños núcleos profundos del cerebelo situados en el interior del cerebelo.

Además de su función principal de control motor, el cerebelo es también necesario para varios tipos de aprendizaje motor, el más notable es aprender a adaptarse a los cambios en las primitivas sensoriales. Varios modelos teóricos se han desarrollado para explicar la calibración sensoriomotora en términos de plasticidad sináptica en el cerebelo, la mayoría de ellos derivados de los primeros modelos formulados por David Marr [192] y James Albus [9], que fueron motivados por la observación de que cada célula de Purkinje del cerebelo recibe dos tipos radicalmente diferentes de entradas. Por un lado, miles de entradas provenientes de las fibras paralelas, cada una individualmente muy débil por sí sola, y por otro lado, la entrada de una sola fibra trepadora que, sin embargo, es tan fuerte que un solo potencial de acción de una de estas fibras es suficiente para que una célula de Purkinje de destino dispare en respuesta un potencial de acción complejo [11]. El concepto básico de la teoría de Marr-Albus es que las fibras trepadoras actúan como una “señal de aprendizaje”, lo cual induce a cambios de larga duración en la fuerza de las entradas simultáneamente activas de las fibras paralelas. Observaciones de la depresión a largo plazo en las entradas por las fibras paralelas han apoyado este grupo de teorías, pero su validez es todavía controvertida (véase la sección 4.2.1 para una extensa revisión sobre los modelos funcionales propuestos en la literatura).

6.3 Motivación general del trabajo

El objetivo principal de este trabajo es el estudio y la implementación de un modelo cerebelar. Es bien sabido que el cerebelo tiene un papel especialmente importante en el control motor, pero también participa en otras funciones cognitivas como la atención y el lenguaje. El cerebelo no inicia el movimiento, sino que contribuye a la coordinación, la precisión y la sincronización exacta. Los aportes de los sistemas sensoriales y de otras partes del cerebro y la médula espinal, se reciben y se integran con órdenes motoras para producir la actividad motora correctamente ajustada. Sin embargo, el diseño de un modelo del cerebelo de mediana escala requiere de varias etapas antes de alcanzar la totalidad del sistema.

En primer lugar, es necesario un simulador ultra-rápido de redes neuronales biológicamente realistas. De hecho, si el objetivo final es el control de un sistema robótico real (en contraposición a un simulador), los requisitos de rendimiento nos llevan a necesitar un simulador en tiempo real. EDLUT [98] puede ser considerado el único simulador de redes neuronales biológicas capaz de cumplir con estos requisitos, tal y como se muestra en trabajos previos de la literatura [40, 227]. Es por ello que el uso de esta arquitectura biológica para el control de sistemas robóticos significa un reto y una novedad en sí mismo. Sin embargo, la estrategia de simulación eficiente basada en tablas de búsqueda (que implementa EDLUT) necesita una cuidada validación para evitar posibles errores derivados de la compilación previa de la conducta neuronal en tablas de consulta. Por lo tanto, en una primera etapa, en el capítulo 2 continuamos el desarrollo del simulador EDLUT para cumplir con todos los requisitos que los modelos posteriores pueden necesitar.

En segundo lugar, podemos aprovechar este software de simulación para llevar a cabo un estudio exhaustivo de la influencia de los parámetros del modelo (y su interpretación en los sistemas biológicos) de la capa granular. La capa granular se puede considerar la primera capa de procesamiento en el cerebelo, y conduce las señales de entrada hasta las fibras paralelas. La importancia de la capa granular está fuera de toda duda, ya que las células granulares (el principal componente de la capa granular) son las neuronas más numerosas en el cerebro (en los seres humanos, las estimaciones del promedio de su número total está alrededor de los 50 billones, lo que significa que alrededor de tres cuartas partes de las neuronas del cerebro son las células de la capa granular del cerebelo). Pero además de este enfoque cuantitativo, los datos experimentales han demostrado la existencia de un procesamiento complejo de señales en esta capa [58, 59, 64]. Por lo tanto, en el capítulo 3 estudiamos las capacidades de filtrado temporal en la capa granular.

Por último, la simulación de un modelo completo de cerebelo dentro de un ciclo de control con el objeto de controlar de forma precisa un brazo robótico supone un exigente banco de prueba para nuestro sistema biológico [40]. Desde el punto de vista de ingeniería, el mejor sistema es aquel que consigue el mejor rendimiento (en términos de precisión, velocidad, etc.). Este punto de vista también podría encajar en la neurociencia (o la neurofisiología), donde el sistema biológicamente inspirado con el mejor rendimiento podría ser un indicio para lograr un conocimiento más profundo del sistema biológico real. Siguiendo este enfoque, en el capítulo 4 se implementa un modelo cerebelar capaz de controlar con precisión un brazo robótico mediante un sistema adaptativo con aprendizaje.

6.4 Nuestra contribución

En este trabajo se desarrolló y evolucionó un entorno de simulación de redes neuronales de spikes capaz de simular redes con un alto nivel de detalle en entornos de tiempo real: EDLUT [98]. Esta herramienta está especialmente indicada para interactuar con aplicaciones externas, robóticas o sistemas de procesamiento visual basado en redes neuronales realistas. Un trabajo previo en la literatura [227] sentó los fundamentos del algoritmo de simulación eficiente conducido por eventos y basado en tablas de consulta. Durante este trabajo, este algoritmo fue re-implementado como un entorno de simulación completo, incluyendo una interfaz gráfica de usuario (GUI) y sistemas de comunicación variados (por medio de archivos de registro de actividad, sockets TCP/IP, la interfaz de programación de aplicaciones —API— o la ejecución como un módulo de Simulink®). También se implementaron en EDLUT las simulaciones de sistemas híbridos incluyendo modelos celulares dirigidos por tiempo y por eventos, y se estudiaron las ventajas que esta estrategia de simulación puede proporcionar en redes realistas tales como la capa granular del cerebelo. Por último, se implementaron para este simulador modelos celulares realistas de gran parte de las células existentes en el cerebelo (como granular, Golgi, estrelladas y las células de Purkinje) y los modelos de spike-respuesta (SRM) del núcleo cuneiforme.

Además de esto, se hizo un uso extensivo de estas herramientas implementadas con el fin de estudiar el filtrado temporal que se realiza en la capa granular del cerebelo. Estudios anteriores demostraron la predisposición de esta capa a producir oscilaciones en varias bandas de frecuencia (sobre todo β y γ entre 12 y 80 Hz) y transmitir estos ritmos a otras regiones del cerebro [58]. En este estudio hemos demostrado la influencia de cada conexión sináptica para mejorar esas capacidades de filtrado.

Por último, se desarrolló un modelo biológicamente plausible del cerebelo (a pequeña escala) que fue utilizado con el fin de controlar un brazo robótico con articulaciones no rígidas realizando tareas de manipulación de objetos. Usando ese sistema, se muestra cómo este lazo de control incluyendo un módulo inspirado en el cerebelo logró una inferencia precisa del modelo dinámico modificado al manipular herramientas pesadas que afectan sustancialmente al modelo base. Además, se estudió la influencia de los parámetros de aprendizaje (LTD y LTP) y la forma en que la información que retroalimenta el módulo pseudocerebelar, tal como el estado actual, puede mejorar de manera notable la capacidad de aprendizaje (en términos de precisión y rendimiento).

6.5 Objetivos

El objetivo principal de este trabajo es contribuir a la comprensión del sistema nervioso central (y específicamente del cerebelo) desde el punto de vista que los estudios de simulación pueden proporcionar. Por lo tanto, un modelo de red neuronal biológicamente plausible es estudiado funcionalmente y utilizado posteriormente con el fin de resolver problemas prácticos (por ejemplo, tareas de control robótico). Para lograr este objetivo, esta tesis aborda los siguientes objetivos intermedios:

- Desarrollo y evolución de un sistema de simulación eficiente para sistemas neuronales de impulsos (spikes) de tamaño medio, haciendo hincapié en la usabilidad de las herramientas resultantes con el fin de ser usada en grupos de investigación relacionados con la neurofisiología.
- Implementación de modelos realistas de los principales tipos de neuronas del cerebelo en el software de simulación propuesto y análisis de las ventajas de la utilización de cada una de las diferentes alternativas estudiadas en función de las propiedades del modelo de red objeto de estudio.
- Estudio de las capacidades de procesamiento temporal de la capa granular (detección de coincidencias y segmentación temporal), el comportamiento oscilatorio y el papel funcional de estas propiedades emergentes en el marco de los modelos del cerebelo completo.
- Estudio de cómo las primitivas sensoriomotoras pueden ser eficientemente inferidas mediante la adaptación de los pesos (LTD y LTP) en las fibras paralelas.
- Evaluación de las capacidades de inferencia de los modelos cinemáticos y dinámicos en el cerebelo mediante la adaptación a largo plazo en la conexión entre las fibras paralelas y las células de Purkinje.
- Evaluación de cómo las señales contextuales pueden ser utilizadas eficientemente para el cambio de contexto en el marco de una tarea de control preciso.

6.6 Marco del proyecto

El trabajo descrito en este documento se ha desarrollado en el marco dos proyectos europeos “SENSOrimotor structuring of Perception and Action for emergent Cognition” (SENSOPAC) [2] y “Realistic Real-time Networks: computation dynamics in the cerebellum” (REALNET).

El proyecto SENSOPAC (financiado por el 6º Programa Marco de la UE en su iniciativa de sistemas cognitivos) se extendió desde enero de 2006 a julio de 2010, en colaboración con 12 instituciones de 9 países diferentes. Dicho proyecto combinaba conceptos de inteligencia artificial y técnicas de modelado de sistemas biológicos para desarrollar un sistema capaz de abstraer nociones cognitivas de las relaciones sensorimotoras durante las interacciones con su entorno, y generalizar este conocimiento a situaciones nuevas. A través de sensores activos y acciones de exploración, el sistema descubre las relaciones sensorimotoras y por consiguiente, aprende la estructura intrínseca de sus interacciones con el mundo y desentraña las relaciones causales y predictivas. El proyecto ha demostrado que un sistema robótico puede arrancar su desarrollo mediante procesos de generalización y el descubrimiento de abstracciones basadas en sensores, mediante los resultados obtenidos de estudios de neurociencia sobre la representación y el procesamiento de datos sensoriales táctiles. El grupo de investigación de la Universidad de Granada ha participado activamente en el desarrollo del entorno de computación de neuronas de spikes (EDLUT) (véase el capítulo 2 en este documento) y la aplicación de los resultados neurológicos a fin de diseñar sistemas de control biológicamente inspirados capaces de llevar a cabo tareas de manipulación. La figura 1.2 muestra la organización en módulos del proyecto SENSOPAC. La Universidad de Granada (y este trabajo como una parte) se centró en el cuarto módulo, a medio camino entre la neurofisiología (módulo 5) y los sistemas más abstractos y la robótica (módulos 1, 2, y 3).

Como continuación a este proyecto, el proyecto REALNET (financiado dentro del 7^{o} Programa Marco de la UE sobre Tecnologías de la Información y la Comunicación) se inició en febrero de 2011 y se extenderá hasta febrero del año 2014. Este proyecto tiene como objetivo comprender el procesamiento de las redes de neuronas mediante el uso de un enfoque diferente: la elaboración de redes realistas de spikes y su utilizacion, junto con registros experimentales de la actividad de la red, para investigar las bases teóricas de la computación del sistema nervioso central. Como punto de referencia de este proyecto se utiliza el circuito cerebelar. Basándose en datos experimentales, en este proyecto se desarrollará el primer modelo de cerebelo realista en tiempo real y será conectado con sistemas robóticos para evaluar el funcionamiento del circuito en condiciones ciclo cerrado percepción-acción. Los datos extraidos de los registros de actividad, las simulaciones a gran escala y los robots se utilizan para explicar el funcionamiento del circuito mediante la teoría del filtro adaptativo [94]. Dentro de este proyecto, el grupo de investigación de la Universidad de Granada se centrará en la evolución del entorno de simulación EDLUT con el fin de simular estructuras biológicas más realistas y aumentar el rendimiento de las simulaciones (véase el capítulo 2).

Dado que este tipo de proyectos requieren un enfoque altamente multidisciplinar, en este trabajo se presentan los resultados desde el punto de vista de los sistemas biológicos. Esto es especialmente relevante en la última parte de este trabajo (véase el capítulo 4), donde se utiliza una arquitectura que emula el funcionamiento del cerebelo con el fin de manipular herramientas con un brazo robótico. Además de esto, este trabajo implica tratar con un sistema robótico (el desarrollo de un simulador de brazo robótico, el estudio de los circuitos de control biológicamente plausible, o la conversión de/en señales de spikes). Todas estas y otras tareas han sido principalmente realizadas por Niceto Luque, perteneciente al mismo departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada.

6.7 Organización de capítulos

Este documento ha sido organizado en tres capítulos principales de acuerdo a las tres cuestiones principales que se han abordado en este trabajo. En primer lugar, en el capítulo 2, desarrollamos y evolucionados un simulador de redes neuronales de spikes extremadamente eficiente (EDLUT). Hemos incluido un apartado a modo de estado del arte, donde las herramientas de simulación más conocidas se presentan y se comparan (ver sección 2.2.2). El desarrollo de EDLUT se ha centrado en dos cuestiones principales: facilitar el uso de EDLUT para los investigadores expertos en neurociencia (mediante el desarrollo de una interfaz gráfica de usuario, nuevos métodos de comunicación o la liberación como una herramienta de código abierto), y la implementación de estructuras biológicamente detalladas y eficientes (como modelos de neurona dirigidos por tiempo, modelos estocásticos de células basadas en tablas de búsqueda o modelos red híbridos). De esta forma, el capítulo 2 representa la herramienta base que se utilizará en los siguientes capítulos.

En el capítulo 3, llevamos a cabo un estudio de simulación de las propiedades de filtrado temporal que tienen lugar en la capa granular y la influencia que la fuerza sináptica de las conexiones puede tener en estas capacidades. En este estudio, hacemos uso de la simulación eficiente mediante el uso de EDLUT para simular la respuesta de la capa granular de forma iterativa utilizando la misma estimulación, pero con parámetros de red diferentes. El procesamiento temporal en la capa granular se piensa que puede tener una gran influencia no sólo en la actividad del cerebelo, sino también en los ritmos de todo el cerebro también (ver el estado del arte en la sección 3.2.1).

Finalmente, en el capítulo 4, se estudia la influencia de la plasticidad sináptica en las fibras paralelas del cerebelo, así como la convergencia de información en la capa granular en el marco de un problema *real* (como es una tarea de manipulación). Para lograr este sistema de aprendizaje, que hace uso de uno de los principales modelos funcionales del cerebelo (ver sección 4.2.1 donde se discuten las principales teorías funcionales), se implementa este modelo por medio de un modelo de red realista funcionando en EDLUT y se utiliza para corregir el control de un brazo robótico con articulaciones no rígidas realizando trayectorias rápidas con precisión.

En resumen, este trabajo representa una evolución en el desarrollo de la herramienta de simulación, para la implementación de un sistema robótico preciso. Cada capítulo trata de los diferentes niveles de abstracción (modelos celulares, modelos de red, y los modelos del sistema, respectivamente), e incluye un estado del arte del problema concreto.

Capítulo 7

Discusión en castellano

7.1 Trabajo futuro

En trabajos futuros se abordará la implementación dirigida por tiempo usando un paso de tiempo variable e independiente para cada neurona, lo que nos llevaría a la aceleración y la mejora de la precisión de las simulaciones. Esta avanzada estrategia de simulación ha sido previamente probada en otras herramientas de simulación [124] con impresionantes resultados. Además de esto, la difusión de los clusters de ordenadores de alto rendimiento ofrecen una oportunidad prometedora para aumentar considerablemente el rendimiento del entorno de simulación EDLUT. Sin embargo, la dificultad de procesar las colas de eventos de manera distribuida a lo largo de los diferentes nodos suponen un importante desafío para esta implementación. En la misma línea de evolucionar el simulador EDLUT, la integración con otras herramientas genéricas de descripción de modelos de neurociencia en diferentes niveles de abstracción (celulares, redes neuronales, sistema,...) como PyNN o NeuroML podría hacer más rápida y sencilla la curva de aprendizaje para potenciales usuarios de EDLUT y permitiría crear modelos que fueran en gran medida reutilizables en herramientas más conocidas como *Neuron* o *Genesis*.

La evolución natural de este trabajo nos lleva a hacer uso de las capacidades de filtrado temporal en la capa granular del cerebelo dentro de un modelo completo de cerebelo aplicado a tareas de control robótico. Aunque el filtrado temporal ha sido puesto de manifiesto en este trabajo, el conocimiento de cómo estas propiedades podrían mejorar un modelo de cerebelo requiere la búsqueda de una teoría completa sobre la codificación de las señales en las fibras musgosas y paralelas. Simulaciones y estudios experimentales recientes [27] pueden ayudar a clarificar este debate (al menos parcialmente).

Por otro lado, la simulación de modelos biológicos detallados del cerebelo todavía representan una cuestión sin resolver. La inclusión de plasticidad sináptica en la mayoría de las conexiones del cerebelo (estudios experimentales han demostrado la existencia de estos mecanismos) es una potente herramienta capaz de configurar automáticamente todo el sistema, adaptándolo al entorno existente, con el fin de obtener el mejor rendimiento. Estos mecanismos podrían ser especialmente relevantes en la capa granular, donde la plasticidad podría llevar los pesos de las conexiones a aquellos valores que optimizan el rebote de sensibilidad en el mismo rango de frecuencias que presenta la actividad de entrada.

Un enfoque similar podría llevarse a cabo en otras partes del cerebelo, como por ejemplo las conexiones $MF \rightarrow DCN$ o $PC \rightarrow DCN$. Esta plasticidad, así como aquella entre $GrC \rightarrow SC \rightarrow PC$, se ha demostrado recientemente que influyen en la consolidación del aprendizaje. Estos trabajos futuros se abordarán en el marco de un proyecto europeo iniciado recientemente (REALNET), y conforme se vayan publicando resultados, todos estos modelos neuronales y de cerebelo y sus mecanismos de adaptación estarán disponibles en el sitio web del simulador EDLUT para facilitar la reproducción de este y otros resultados futuros.

7.2 Publicación de resultados

Nuestro trabajo de investigación ha sido evaluado en el marco de conferencias internacionales y publicaciones científicas (con índice de impacto en el JCR). Haciendo un rápido resumen, la producción científica se puede concretar como: 2 artículos publicados en revista (Q1 —Top 3— y Q2 en sus respectivas categorías JCR), 2 artículos aceptados (revistas Q1 en sus respectivas categorías JCR, y de hecho, uno de ellos es Top 3), 3 otros artículos que se encuentran actualmente bajo revisión (Q1 y Q2 en sus respectivas categorías JCR). 5 Conferencias internacionales (COGSYS 2008, ICJNN 2010, COGSYS 2010, y IWANN 2011).

7.2.1 Revistas internacionales

1. D'Angelo, E.; Koekkoek, S.K.E.; Lombardo, P.; Solinas, S.; Ros, E.; *Garrido, J.*; Schonewille, M.; De Zeeuw, C.I.: "Timing in the Cerebelllum: Oscillations and Resonance in the Granular Layer". Neuroscience, 162, 805–815, 2009. doi:10.1.1.154.4873.

2. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Coenen, O. J. M. D.; Ros, E: "Cerebellar input configuration towards object model abstraction in manipulation tasks". IEEE transaction on neural networks, 22(8), 1321–1328, 2011. doi:10.1109/TNN.2011.2156809. (Revista Top 3 en Ciencias de la Computación, Hardware & Arquitectura, 1er cuartil en Ciencias de la Computación, Inteligencia Artificial, Teoría & Métodos, y en Ingeniería, Eléctrica & Electrónica).

3. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Tolu, S.; Ros, E.: "Adaptive cerebellar spiking model in a bio-inspired robot-control loop". International journal on neural systems, 21(5), 385–401, 2011. doi:10.1142/S0129065711002900. (1er cuartil en Ciencias de la Computación, Inteligencia Artificial).

4. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Coenen, O. J. M. D.; Ros, E: "Cerebellar-like corrective-model abstraction engine for robot movement control". IEEE Transaction on system, man, and cybernetics - Part B, 41(5), 1299–1312, 2011. doi:10.1109/TSMCB.2011.2138693. (Revista Top 3 en Automatización & Sistemas de Control, y en Ciencias de la Computación, Cibernética. 1er cuartil en Ciencias de la Computación, Inteligencia Artificial).

5. *Garrido, J. A.*; Carrillo, R. R.; Ros, E.; D'Angelo, E.: "The impact of synaptic weights on spike timing in the cerebellar input layer: a simulation study". Bajo revisión en Neural Computation.

6. Luque, N. R.; *Garrido, J. A.*; Ralli, J.; Laredo, J. J.; Ros, E.: "From Sensors to Spikes: Evolving Receptive Fields to Enhance Sensory Motor Information in a Robot-Arm Scenario". Bajo revisión en International journal on neural systems.

7. Tolu, S.; Vanegas, M.; Luque, N. R.; *Garrido, J. A.*; Ros, E.: "Bio-inspired Adaptive Feedback Error Learning Architecture For Motor Control". Bajo revisión en Biological Cybernetics.

7.2.2 Conferencias Internacionales

1. Carrillo, R. R.; *Garrido, J.*; Ros, E.; Tolu, S.; Boucheny, C.; Coenen, O. J. M. D. (2008): "A real-time spiking cerebellum model for learning robot control". International Conference On Cognitive Systems (COGSYS 2008).

2. D'Angelo, E.; Van der Smagt, P.; Solinas, S.; Jorntell, H.; Ros, E.; *Garrido, J.* (2010): "Realistic circuit modelling: large-scale simulations of the cerebellar granular layer". International Conference On Cognitive Systems (COGSYS 2010).

3. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Ros, E. (2010): "Cerebellar spiking engine: Towards object model abstraction in manipulation". International Joint Conference on Neural Networks (ICJNN 2010). doi:10.1109/IJCNN.2010.5596531.

4. *Garrido, J. A.*; Carrillo, R. R.; Luque, N. R.; Ros, E.: "Event and time driven hybrid simulation of spiking neural networks". International Work-Conference on Artificial Neural Networks (IWANN 2011). Advances in Computational Intelligence. Lecture Notes in Computer Science, 6691, pp. 554–561. Springer, Heidelberg (2011). doi:10.1007/978–3–642–21501–8_69.

5. Luque, N. R.; *Garrido, J. A.*; Carrillo, R. R.; Ros, E.: "Context Separability Mediated by the Granular Layer in a Spiking Cerebellum Model for Robot Control". International Work-Conference on Artificial Neural Networks (IWANN 2011). Advances in Computational Intelligence. Lecture Notes in Computer Science, 6691, pp. 537–546. Springer, Heidelberg (2011). doi:10.1007/978–3–642–21501–8_67.

7.2.3 Otros medios de difusión

La difusión del conocimiento representa uno de los puntos clave en el desarrollo de un proyecto europeo. Dado que este trabajo se ha realizado a la par que la evolución de los proyectos SENSOPAC y REALNET (ver sección 7.3), se ha hecho un esfuerzo notable en liberar los avances que surgen de estos trabajos. Es por ello que el entorno de simulación EDLUT fue liberado como una aplicación de código abierto bajo licencia GPLv3. Esto permite el uso de este software para cualquier proyecto de investigación o industrial con casi ninguna restricción. Por lo tanto, después de un proceso de documentación exhaustivo del código fuente, EDLUT fue lanzado en el sitio web [98] en abril de 2009. Desde entonces, varios grupos han mostrado su interés en el uso de EDLUT en sus propios proyectos y hemos apoyado todas sus preguntas y sugerencias. Las siguientes estadísticas muestran cuantitativamente el interés despertado por este proyecto:

- El sitio web ha recibido más de 11.000 visitas desde su lanzamiento en 2009.
- La presentación general del software EDLUT ha sido descargado unas 2.000 veces, mientras que el código fuente de la última versión liberada ha sido descargado más de 800 veces.
- Varios grupos han utilizado EDLUT para sus desarrollos tales como la Universidad de Pavía (Egidio D'Angelo y Sergio Solinas) de la Universidad Pierre y Marie Curie en París (Angelo Arleo, Luca Leonardo Bolonia y Jean Baptiste Passot), la Universidad Erasmus de Rotterdam (Chris de Zeeuw y Jornt de Gruijl), la Universidad de Lund (Henrik Jörntell y Carl Fredrik Ekerot), SICS (Martin Nilson) y otros investigadores, como Boris Barbour (CNRS), Olivier Coenen o Mike Arnold.

La liberación del código fuente de EDLUT tuvo una calurosa acogida en los medios de comunicación, y especialmente en la comunidad de software libre, tal y como demuestran los siguientes titulares publicados en diversos medios de comunicación digitales:

1. "Move to create less clumsy robots" en BBC News.
2. "EDLUT human nervous system researches disease and tests medicine" en Compute Scotland.
3. "Artificial simulator of the nervous system created for research into diseases" en Science daily.
4. "Development of an artificial simulator of the nervous system to do research into diseases" en Eureka Alert.
5. "Liberado simulador del sistema nervioso: EDLUT" en Oficina de Software Libre UGR.

6. “EDLUT: un simulador neuronal” at Microsiervos.
7. “Desarrollan un simulador del sistema nervioso para investigación biomédica” en Universidades.
8. “Simulador artificial del sistema nervioso humano” en `http://www.solociencia.com/medicina/09052105.htm`.
9. “Desarrollan un simulador del sistema nervioso” en Orion.
10. “La UGR participa en la construcción de un cerebelo artificial” en Universia.
11. “La Universidad de Granada desarrolla un simulador del sistema nervioso humano” en Ideal.

7.3 Marco científico general

Este trabajo científico se ha realizado y financiado parcialmente dentro de los proyectos europeos SENSOPAC: SENSOrimotor structuring of Perception and Action for emergent Cognition (IST-028056), y REALNET: Realistic Real-time Networks: computation dynamics in the cerebellum (IST-270.434). Esto ha representado un excelente marco de colaboración con diversos grupos de investigación en otras universidades y centros de investigación europeos. El trabajo presentado representa la principal contribución de la Universidad de Granada dentro del consorcio SENSOPAC. Esto acarreó una gran responsabilidad por la necesidad de obtener los resultados en las fechas previstas durante el proceso de investigación. Además de los informes técnicos requeridos y presentaciones para las revisiones científicas de la UE, una demostración final (prueba de concepto) fue requerida e implementada conjuntamente con la Universidad Pierre y Marie Curie de París y el DLR (Munich). El esfuerzo invertido en esta demostración y otras es significativo, pero permite una fácil y vistosa evaluación de las capacidades del sistema y facilita la difusión de los resultados más allá de un escenario puramente científico, con el objeto de iniciar futuras colaboraciones con la industria y llamar la atención en los medios de comunicación (periódicos, TV, etc.).

Dado que este tipo de proyectos requieren un enfoque multidisciplinar, en este trabajo se presentan los resultados principalmente desde el punto de vista de los sistemas biológicos. Esto es especialmente relevante en la última parte de este trabajo (véase el capítulo 4), donde se utiliza una arquitectura emulando el cerebelo con el fin de manipular herramientas con un brazo robótico. Además de esto, este trabajo implica tratar con sistemas robóticos (el desarrollo de un simulador de brazo robótico, el estudio de los ciclos de control biológicamente plausibles, o la conversión de valores analógicos a señales de impulsos). Todas estas tareas han sido principalmente realizadas por Niceto Luque, de la Universidad de Granada. Además de esto, un esfuerzo considerable se realizó en el diseño e implementación de una herramienta fácil de usar, intuitiva y un simulador (teniendo en cuenta las dificultades de comprensión de este tipo de sistemas en sí mismos) capaz de funcionar en diferentes entornos y con una amplia variedad de dispositivos controlados. Esto ha requerido un trabajo con alto nivel de colaboración y coordinación con el resto del equipo.

7.4 Principales contribuciones

A continuación se resumen las principales aportaciones del trabajo presentado:

- Un simulador dirigido por eventos basado en tablas de consulta ha sido ampliamente evolucionado para cumplir con los requisitos principales de un simulador de redes neuronales en tiempo real, capaz de interactuar con gran cantidad de sistemas externos. Además de esto, se ha desarrollado una interfaz gráfica de usuario basada en Java, lo que facilita el uso de estas herramientas de simulación para investigadores que no están familiarizados con este simulador.

- El esquema de simulación dirigido por eventos ha sido reimplementado para ser capaz de simular de forma nativa modelos de neurona dirigidos por tiempo (solos o en combinación con otros dirigidos por eventos).

- Los modelos dirigidos por eventos (o más genéricamente, las redes híbridas incluyendo ambos) mostraron un buen rendimiento y precisión cuando se trabaja con bajos niveles de actividad. Por el contrario, los modelos dirigidos por tiempo son preferibles en aquellos casos donde las características del experimento producen una actividad neuronal alta. Finalmente, un sistema de simulación híbrido puede ser conveniente en la simulación de sistemas biológicos que presentan niveles de actividad diversos.

- Un modelo computacional de la capa granular ha sido desarrollado y analizado, mostrando que la detección de coincidencias en esta capa está regulada de manera crítica por el ciclo inhibitorio hacia adelante ($MF \rightarrow GoC \rightarrow GRC$).

- Además de esto, este modelo computacional de la capa granular ha demostrado su capacidad para producir un rebote de sensibilidad (recientemente predicho sobre la base de diversos datos experimentales). El ciclo de realimentación hacia adelante deshinibitorio ($MF \rightarrow GRC \rightarrow SC \rightarrow GoC$) resultó ser el más influyente ya que afecta tanto a la amplitud del rebote de sensibilidad como al intervalo entre estímulos que maximiza la respuesta de la red.

- Este modelo de la capa granular también mostró oscilaciones de la actividad eléctrica en la banda de frecuencias β/γ como respuesta a una actividad de entrada aleatoria tal y como se ha podido observar en la capa granular del cerebelo en experimentos *in vivo* e *in vitro*.

- Se ha presentado un modelo del cerebelo integrado en un circuito con un modelo dinámico inverso crudo. Este modelo cerebelar es capaz de proporcionar con eficacia pares correctivos para compensar las desviaciones en la dinámica de un modelo de la planta original (debido a la manipulación de objetos).

- Este modelo de adaptación ha mostrado cómo los valores de LTD y LTP en las fibras paralelas tienen que estar equilibrados para lograr las capacidades de adaptación precisas. Hemos evaluado hasta qué punto unos valores correctamente ajustados de las componentes LTD y LTP conducen a una reducción efectiva del error en la realización de tareas de manipulación de objetos que puedan afectar considerablemente la dinámica de la planta base del brazo.

- Se ha presentado una evaluación de la influencia de las señales sensoriales de entrada en el cerebelo: con dos representaciones distintas, recibiendo entradas relacionadas con el contexto (EC) y recibiendo sólo señales sensoriales del estado actual del robot (IC) codificando el contexto durante los experimentos. Una configuración IC&EC del cerebelo es capaz de aprovecharse de las dos configuraciones de forma complementaria. Con el modelo propuesto se consiguen transiciones entre contextos más suaves con una velocidad de convergencia mayor.

- Este trabajo muestra cómo este modelo con señales de entrada cerebelares IC&EC ofrece la posibilidad de *sobreponerse* a la aparición de información contextual externa errónea (incoherente), haciendo que esta configuración del cerebelo sea robusta contra representaciones incongruentes.

References

[1] Facets official webpage, 2005. URL `http://facets.kip.uni-heidelberg.de/index.html`. 25

[2] Sensorimotor structuring of perception and action for emergent cognition (sensopac) website, 2006. URL `http://www.sensopac.org`. 10, 182

[3] E.D. Adrian and B.H.C. Matthews. The interpretation of potential waves in the cortex. *The Journal of Physiology*, 81(4):440, 1934. 69

[4] R. Agis, E. Ros, J. Diaz, R. Carrillo, and E.M. Ortigosa. Hardware event-driven simulation engine for spiking neural networks. *International journal of electronics*, 94(5): 469–480, 2007. 126

[5] A.A. Ahmed, D.M. Wolpert, and J.R. Flanagan. Flexible representations of dynamics are used in object manipulation. *Current Biology*, 18(10):763–768, 2008. 163

[6] C.D. Aizenman and D.J. Linden. Rapid, synaptically driven increases in the intrinsic excitability of cerebellar deep nuclear neurons. *nature neuroscience*, 3:109–112, 2000. 72

[7] C.D. Aizenman, P.B. Manis, and D.J. Linden. Polarity of long-term synaptic gain change is related to postsynaptic spike firing at a cerebellar inhibitory synapse. *Neuron*, 21(4): 827–835, 1998. 72

[8] C. Akazawa, R. Shigemoto, Y. Bessho, S. Nakanishi, and N. Mizuno. Differential expression of five n-methyl-d-aspartate receptor subunit mrnas in the cerebellum of developing and adult rats. *The Journal of comparative neurology*, 347(1):150–160, 1994. 81

[9] J.S. Albus. A theory of cerebellar function. *Mathematical Biosciences*, 10(1-2):25–61, 1971. 6, 111, 112, 116, 178

[10] B.B. Andersen, L. Korbo, and B. Pakkenberg. A quantitative study of the human cerebellum with unbiased stereological techniques. *The Journal of Comparative Neurology*, 326(4):549–560, 1992. 132

[11] G. Andersson and D.M. Armstrong. Complex spikes in purkinje cells in the lateral vermis (b zone) of the cat cerebellum during locomotion. *The Journal of Physiology*, 385(1):107, 1987. 6, 179

[12] M.A. Arbib, P. Érdi, and J. Szentágothai. *Neural organization: Structure, function, and dynamics*. The MIT Press, 1998. 4, 64, 176

[13] A. Arenz, R.A. Silver, A.T. Schaefer, and T.W. Margrie. The contribution of single synapses to sensory representation in vivo. *Science*, 321(5891):977, 2008. 66

[14] S. Armano, P. Rossi, V. Taglietti, and E. D'angelo. Long-term potentiation of intrinsic excitability at the mossy fiber-granule cell synapse of rat cerebellum. *The Journal of Neuroscience*, 20(14):5208, 2000. 68, 98, 104

[15] D.M. Armstrong. Functional significance of connections of the inferior olive. *Physiological Reviews*, 54(2):358, 1974. ISSN 0031-9333. 124

[16] D.M. Armstrong and J.A. Rawson. Activity patterns of cerebellar cortical neurones and climbing fibre afferents in the awake cat. *The Journal of Physiology*, 289(1):425, 1979. ISSN 0022-3751. 124

[17] C. Assad, S. Dastoor, S. Trujillo, and L. Xu. Cerebellar dynamic state estimation for a biomorphic robot arm. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 1, pages 877–882. IEEE, 2005. 132

[18] M.W. Bagnall, L.E. McElvain, M. Faulstich, and S. du Lac. Frequency-independent synaptic transmission supports a linear vestibular behavior. *Neuron*, 60(2):343–352, 2008. 66

[19] N.H. Barmack and V. Yakhnitsa. Functions of interneurons in mouse cerebellum. *The Journal of Neuroscience*, 28(5):1140, 2008. 64, 69, 71, 72, 85, 92, 120

[20] A.G. Barto, R.S. Sutton, and C.W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, & Cybernetics*, 1983. 141

[21] E. Bauswein, F.P. Kolb, B. Leimbeck, and F.J. Rubia. Simple and complex spike activity of cerebellar purkinje cells during active and passive movements in the awake monkey. *The Journal of Physiology*, 339(1):379, 1983. 112

[22] J.R. Bloedel and V. Bracha. Current concepts of climbing fiber function. *The Anatomical Record*, 253(4):118–126, 1998. ISSN 0003-276X. 118

[23] A. Borst and F.E. Theunissen. Information theory and neural coding. *Nature neuroscience*, 2:947–958, 1999. 106

[24] C. Boucheny, R. Carrillo, E. Ros, and O.J.M.D. Coenen. Real-time spiking neural network: An adaptive cerebellar model. *Computational Intelligence and Bioinspired Systems*, pages 136–144, 2005. 118, 127, 141, 144

[25] J.M. Bower, D. Beeman, and A.M. Wylde. *The book of GENESIS: exploring realistic neural models with the GEneral NEural SImulation System*. Telos New York, NY, 1995. 4, 15, 23, 177

[26] R. Brasselet, R.S. Johansson, and A. Arleo. Optimal context separation of spiking haptic signals by second-order somatosensory neurons. *Y. Bengio, D. Schuurmans, J. Lafferty, CKI Williams, and A. Culotta, editors, Advances in Neural Information Processing Systems*, 22:180–188, 2009. 44, 51, 119

[27] R. Brasselet, R.S. Johansson, and A. Arleo. Quantifying neurotransmission reliability through metrics-based information analysis. *Neural Comput*, 23(4):852–881, Apr 2011. doi: 10.1162/NECO_a_00099. 119, 122, 165, 188

[28] R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J.M. Bower, M. Diesmann, A. Morrison, P.H. Goodman, F.C. Harris, et al. Simulation of networks of spiking neurons: A review of tools and strategies. *Journal of computational neuroscience*, 23(3):349–398, 2007. 19, 20

[29] S.G. Brickley, C. Misra, M.H. Mok, M. Mishina, and S.G. Cull-Candy. Nr2b and nr2d subunits coassemble in cerebellar golgi cells to form a distinct nmda receptor subtype restricted to extrasynaptic sites. *The Journal of neuroscience*, 23(12):4958, 2003. 82

[30] I.E. Brown and J.M. Bower. Congruence of mossy fiber and climbing fiber tactile projections in the lateral hemispheres of the rat cerebellum. *The Journal of comparative neurology*, 429(1):59–70, 2001. 65

[31] N. Brunel, V. Hakim, P. Isope, J.P. Nadal, and B. Barbour. Optimal information storage and the distribution of synaptic weights:: Perceptron versus purkinje cell. *Neuron*, 43(5): 745–757, 2004. 70, 73

[32] C. Buisseret-Delmas and P. Angaut. Anatomical mapping of the cerebellar nucleocortical projections in the rat: a retrograde labeling study. *The Journal of Comparative Neurology*, 288(2):297–310, 1989. 71

[33] D. Bullock, J.C. Fiala, and S. Grossberg. A neural model of timed response learning in the cerebellum. *Neural Networks*, 7(6-7):1101–1114, 1994. ISSN 0893-6080. 119

[34] J. Butterfass, M. Grebenstein, H. Liu, and G. Hirzinger. Dlr-hand ii: Next generation of a dextrous robot hand. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 1, pages 109–114. IEEE, 2001. 127

[35] G. Buzsáki. *Rhythms of the Brain*. Oxford University Press, USA, 2006. 73, 79

[36] N.T. Carnevale and M.L. Hines. Efficient discrete event simulation of spiking neurons in neuron. In *Poster presented at the 2002 SFN Meeting, Yale University, New Haven. SfN abstract*, volume 312. Citeseer, 2002. 20, 21

[37] N.T. Carnevale and M.L. Hines. *The NEURON book*. Cambridge Univ Pr, 2006. 4, 16, 21, 177

[38] R. Carrillo. *Efficient implementation of neural structures based in the nervious system*. PhD thesis, University of Granada, 2009. 14, 30

[39] R.R. Carrillo, E. Ros, B. Barbour, C. Boucheny, and O. Coenen. Event-driven simulation of neural population synchronization facilitated by electrical coupling. *Biosystems*, 87 (2-3):275–280, 2007. 78

[40] R.R. Carrillo, E. Ros, C. Boucheny, and O.J. Coenen. A real-time spiking cerebellum model for learning robot control. *BioSystems*, 94(1-2):18–27, 2008. 7, 8, 75, 106, 111, 124, 144, 179, 180

[41] R.R. Carrillo, E. Ros, S. Tolu, T. Nieus, and E. D'Angelo. Event-driven simulation of cerebellar granule cells. *BioSystems*, 94(1-2):10–17, 2008. 40, 112, 124

[42] A.G. Carter and W.G. Regehr. Quantal events shape cerebellar interneuron firing. *Nature neuroscience*, 5(12):1309–1318, 2002. 77

[43] M. Casado, S. Dieudonne, and P. Ascher. Presynaptic n-methyl-d-aspartate receptors at the parallel fiber–purkinje cell synapse. *Proceedings of the National Academy of Sciences*, 97(21):11593, 2000. 74

[44] M. Casado, P. Isope, and P. Ascher. Involvement of presynaptic n-methyl-d-aspartate receptors in cerebellar long-term depression. *Neuron*, 33(1):123–130, 2002. 74

[45] L. Cathala, N.B. Holderith, Z. Nusser, D.A. DiGregorio, and S.G. Cull-Candy. Changes in synaptic structure underlie the developmental speeding of ampa receptor-mediated epscs. *Nature neuroscience*, 8(10):1310–1318, 2005. 68

[46] P. Chadderton, T.W. Margrie, and M. Häusser. Integration of quanta in cerebellar granule cells during sensory processing. *Nature*, 428(6985):856–860, 2004. 66, 106

[47] J. Chavas and A. Marty. Coexistence of excitatory and inhibitory gaba synapses in the cerebellar interneuron network. *The Journal of neuroscience*, 23(6):2019, 2003. 77

[48] E. Chorev, Y. Yarom, and I. Lampl. Rhythmic episodes of subthreshold membrane potential oscillations in the rat inferior olive nuclei in vivo. *The Journal of neuroscience*, 27(19):5043, 2007. 69

[49] O.J.-M.D. Coenen, M.P. Arnold, T.J. Sejnowski, and M.A. Jabri. Parallel fiber coding in the cerebellum for life-long learning. *Autonomous Robots*, 11:291–297, 2001. ISSN 0929-5593. URL `http://dx.doi.org/10.1023/A:1012403510221`. 10.1023/A:1012403510221. 15, 18, 58

[50] M. Coesmans, J.T. Weber, C.I. De Zeeuw, and C. Hansel. Bidirectional parallel fiber plasticity in the cerebellum under climbing fiber control. *Neuron*, 44(4):691–700, 2004. 69, 70

[51] J.L. Contreras-Vidal, S. Grossberg, and D. Bullock. A neural model of cerebellar learning for arm movement control: cortico-spino-cerebellar dynamics. *Learning & memory*, 3(6): 475, 1997. 145

[52] R. Courtemanche and Y. Lamarre. Local field potential oscillations in primate cerebellar cortex: synchronization with cerebral cortex during active and passive expectancy. *Journal of neurophysiology*, 93(4):2039, 2005. 69

[53] R. Courtemanche, J.P. Pellerin, and Y. Lamarre. Local field potential oscillations in primate cerebellar cortex: modulation during active and passive expectancy. *Journal of neurophysiology*, 88(2):771, 2002. 69

[54] W.E. Crill. Unitary multiple-spiked responses in cat inferior olive nucleus. *Journal of neurophysiology*, 33(2):199, 1970. ISSN 0022-3077. 124

[55] S. Crook, D. Beeman, P. Gleeson, and F. Howell. Xml for model specification in neuroscience. *Brains, Minds and Media*, 1(2), 2005. 24

[56] S. Cull-Candy, S. Brickley, and M. Farrant. Nmda receptor subunits: diversity, development and disease. *Current opinion in neurobiology*, 11(3):327–335, 2001. 82

[57] S.S. Dalal, A.G. Guggisberg, E. Edwards, K. Sekihara, A.M. Findlay, R.T. Canolty, M.S. Berger, R.T. Knight, N.M. Barbaro, H.E. Kirsch, et al. Five-dimensional neuroimaging: localization of the time-frequency dynamics of cortical activity. *Neuroimage*, 40(4):1686–1700, 2008. 103, 106

[58] E. D'Angelo. The critical role of golgi cells in regulating spatio-temporal integration and plasticity at the cerebellum input stage. *Frontiers in Neuroscience*, 2(1):35, 2008. 8, 9, 64, 68, 85, 180, 181

[59] E. D'Angelo and C.I. De Zeeuw. Timing and plasticity in the cerebellum: focus on the granular layer. *Trends in neurosciences*, 32(1):30–40, 2009. 8, 64, 103, 106, 180

[60] E. D'Angelo, P. Rossi, and V. Taglietti. Different proportions of n-methyl-d-aspartate and non-n-methyl-d-aspartate receptor currents at the mossy fibre-granule cell synapse of developing rat cerebellum. *Neuroscience*, 53(1):121–130, 1993. 77

[61] E. D'Angelo, G. De Filippi, P. Rossi, and V. Taglietti. Synaptic excitation of individual rat cerebellar granule cells in situ: evidence for the role of nmda receptors. *The Journal of physiology*, 484(Pt 2):397, 1995. 64, 68, 77, 83, 102

[62] E. D'Angelo, G.D. Filippi, P. Rossi, and V. Taglietti. Ionic mechanism of electroresponsiveness in cerebellar granule cells implicates the action of a persistent sodium current. *Journal of neurophysiology*, 80(2):493, 1998. 77

[63] E. D'Angelo, P. Rossi, S. Armano, and V. Taglietti. Evidence for nmda and mglu receptor-dependent long-term potentiation of mossy fiber-granule cell transmission in rat cerebellum. *Journal of neurophysiology*, 81(1):277, 1999. 68, 104

[64] E. D'Angelo, T. Nieus, A. Maffei, S. Armano, P. Rossi, V. Taglietti, A. Fontana, and G. Naldi. Theta-frequency bursting and resonance in cerebellar granule cells: experimental evidence and modeling of a slow k+-dependent mechanism. *The Journal of Neuroscience*, 21(3):759, 2001. 8, 64, 67, 68, 69, 77, 103, 105, 180

[65] E. D'Angelo, T. Nieus, M. Bezzi, A. Arleo, and O.J.M.D. Coenen. Modeling synaptic transmission and quantifying information transfer in the granular layer of the cerebellum. *Computational Intelligence and Bioinspired Systems*, pages 107–114, 2005. 112, 136

[66] E. D'Angelo, P. Rossi, D. Gall, F. Prestori, T. Nieus, A. Maffei, and E. Sola. Long-term potentiation of synaptic transmission at the mossy fiber-granule cell relay of cerebellum. *Progress in Brain Research*, 148:69–80, 2005. 133

[67] E. D'Angelo, S.K.E. Koekkoek, P. Lombardo, S. Solinas, E. Ros, J. Garrido, M. Schonewille, and C.I. De Zeeuw. Timing in the cerebellum: oscillations and resonance in the granular layer. *Neuroscience*, 162(3):805–815, 2009. 6, 68, 88, 105, 110, 112, 121, 133, 149

[68] A.P. Davison, D. Brüderle, J. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger. Pynn: a common interface for neuronal network simulators. *Frontiers in neuroinformatics*, 2, 2008. 27

[69] P. Dayan and L.F. Abbott. *Theoretical neuroscience: Computational and mathematical modeling of neural systems.* MIT Press, Cambridge, MA, 2001. 4, 177

[70] E. De Schutter and J.M. Bower. An active membrane model of the cerebellar purkinje cell. i. simulation of current clamps in slice. *Journal of Neurophysiology*, 71(1):375–400, 1994. 23

[71] C. de Solages, G. Szapiro, N. Brunel, V. Hakim, P. Isope, P. Buisseret, C. Rousseau, B. Barbour, and C. Lena. High-frequency organization and synchrony of activity in the purkinje cell layer of the cerebellum. *Neuron*, 58(5):775–788, 2008. 69

[72] C.I. De Zeeuw, J.C. Holstege, T.J.H. Ruigrok, and J. Voogd. Ultrastructural study of the gabaergic, cerebellar, and mesodiencephalic innervation of the cat medial accessory olive: anterograde tracing combined with immunocytochemistry. *The Journal of Comparative Neurology*, 284(1):12–35, 1989. 70

[73] C.I. De Zeeuw, D.R. Wylie, P.L. DiGiorgi, and J.I. Simpson. Projections of individual purkinje cells of identified zones in the flocculus to the vestibular and cerebellar nuclei in the rabbit. *The Journal of Comparative Neurology*, 349(3):428–447, 1994. 65

[74] C.I. De Zeeuw, C.C. Hoogenraad, S.K.E. Koekkoek, T.J.H. Ruigrok, N. Galjart, and J.I. Simpson. Microcircuitry and function of the inferior olive. *Trends in neurosciences*, 21 (9):391–400, 1998. 65, 69, 112, 118

[75] C.I. De Zeeuw, F.E. Hoebeek, and M. Schonewille. Causes and consequences of oscillations in the cerebellar cortex. *Neuron*, 58(5):655–658, 2008. 72, 105

[76] I. Dean, S.J. Robertson, and F.A. Edwards. Serotonin drives a novel gabaergic synaptic current recorded in rat cerebellar purkinje cells: a lugaro cell to purkinje cell synapse. *The Journal of neuroscience*, 23(11):4457, 2003. 65

[77] P. Dean and J. Porrill. Adaptive-filter models of the cerebellum: computational analysis. *The Cerebellum*, 7(4):567–571, 2008. 111, 116, 118, 119, 125

[78] P. Dean and J. Porrill. The cerebellum as an adaptive filter: a general model? *Functional neurology*, 25(3):173, 2010. ISSN 0393-5264. 118, 119, 125

[79] P. Dean, J. Porrill, C.F. Ekerot, and H. Jörntell. The cerebellar microcircuit as an adaptive filter: experimental and computational evidence. *Nature Reviews Neuroscience*, 11(1):30–43, 2009. 64, 65, 106, 108, 115, 116, 118, 119, 125, 132

[80] A. Delorme, J. Gautrais, R. Van Rullen, and S.J. Thorpe. Spikenet: A simulator for modeling large networks of integrate and fire neurons. *Neurocomputing*, 26-27:989–996, 1999. 15

[81] M. D'Haene, B. Schrauwen, J. Van Campenhout, and D. Stroobandt. Accelerating event-driven simulation of spiking neurons with multiple synaptic time constants. *Neural Computation*, 21:1068–1099, 2009. doi: 10.1162/neco.2008.02-08-707. 15

[82] M. Diesmann and M.O. Gewaltig. Nest: An environment for neural systems simulations. *Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen*, page 43, 2002. 24

[83] S. Dieudonné. Submillisecond kinetics and low efficacy of parallel fibre-golgi cell synaptic currents in the rat cerebellum. *The Journal of Physiology*, 510(3):845–866, 1998. 64

[84] S. Dieudonné and A. Dumoulin. Serotonin-driven long-range inhibitory connections in the cerebellar cortex. *The Journal of Neuroscience*, 20(5):1837, 2000. 65

[85] M. Djurfeldt, C. Johansson, Ö. Ekeberg, M. Rehn, M. Lundqvist, and A. Lansner. Massively parallel simulation of brain-scale neuronal network models. *Computational biology and neurocomputing, School of Computer Science and Communication. Stockholm: Royal Institute of Technology TRITA-NA- P*, 2005. 26

[86] G.P. Dugué, N. Brunel, V. Hakim, E. Schwartz, M. Chat, M. Lévesque, R. Courtemanche, C. Léna, and S. Dieudonné. Electrical coupling mediates tunable low-frequency oscillations and resonance in the cerebellar golgi cell network. *Neuron*, 61(1):126–139, 2009. 96, 102, 103

[87] A. Dumoulin, A. Triller, and S. Dieudonné. Ipsc kinetics at identified gabaergic and mixed gabaergic and glycinergic synapses onto cerebellar golgi cells. *The Journal of Neuroscience*, 21(16):6045, 2001. 64, 72

[88] J.C. Eccles et al. *The cerebellum as a neuronal machine*. Springer-Verlag, 1967. 71, 79

[89] B. Ermentrout. *Simulating, analyzing, and animating dynamical systems: a guide to XPPAUT for researchers and students*. Society for Industrial Mathematics, 2002. 26

[90] T. Flash and T.J. Sejnowski. Computational approaches to motor control. *Current Opinion in Neurobiology*, 11(6):655–662, 2001. ISSN 0959-4388. 119

[91] P. Flourens. *De la vie et de l'intelligence*. Garnier, Paris, 1858. 123

[92] L. Forti, E. Cesana, J. Mapelli, and E. D'Angelo. Ionic mechanisms of autorhythmic firing in rat cerebellar golgi cells. *The Journal of physiology*, 574(3):711, 2006. 64, 67, 77

[93] W.J. Freeman. *How brains make up their minds*. Columbia Univ Pr, 2000. 5, 177

[94] M. Fujita. Adaptive filter model of the cerebellum. *Biological Cybernetics*, 45(3):195–206, 1982. 11, 115, 116, 119, 125, 142, 183

[95] F. Gabbiani, J. Midtgaard, and T. Knopfel. Synaptic integration in a model of cerebellar granule cells. *Journal of neurophysiology*, 72(2):999, 1994. 77, 78, 81, 82

[96] D. Gall, F. Prestori, E. Sola, A. D'Errico, C. Roussel, L. Forti, P. Rossi, and E. D'Angelo. Intracellular calcium regulation by burst discharge determines bidirectional long-term synaptic plasticity at the cerebellum input stage. *The Journal of neuroscience*, 25(19): 4813, 2005. 68, 98, 104

[97] J.L. Gardner, S.N. Tokiyama, and S.G. Lisberger. A population decoding framework for motion aftereffects on smooth pursuit eye movements. *The Journal of neuroscience*, 24 (41):9035, 2004. 112

[98] J.A Garrido. Edlut official website, 2009. URL `http://edlut.googlecode.com`. 7, 8, 14, 168, 179, 180, 190

[99] A.P. Georgopoulos, A.B. Schwartz, and R.E. Kettner. Neuronal population coding of movement direction. *Science*, 233(4771):1416, 1986. 4, 176

[100] W. Gerstner and W.M. Kistler. *Spiking neuron models: Single neurons, populations, plasticity.* Cambridge Univ Pr, 2002. 4, 50, 51, 74, 75, 76, 133, 177

[101] F.J. Geurts, E. De Schutter, and S. Dieudonné. Unraveling the cerebellar cortex: cytology and cellular physiology of large-sized interneurons in the granular layer. *The Cerebellum*, 2(4):290–299, 2003. 64

[102] A.R. Gibson, K.M. Horn, and M. Pong. Activation of climbing fibers. *The Cerebellum*, 3 (4):212–221, 2004. ISSN 1473-4222. 118

[103] S. Gilman. The cerebellum and neural control. *Archives of Neurology*, 42(4):310, 1985. 112

[104] P. Gleeson, V. Steuber, and R.A. Silver. neuroconstruct: a tool for modeling networks of neurons in 3d space. *Neuron*, 54(2):219–235, 2007. 29

[105] P. Gleeson, S. Crook, R.C. Cannon, M.L. Hines, G.O. Billings, M. Farinella, T.M. Morse, A.P. Davison, S. Ray, U.S. Bhalla, et al. Neuroml: a language for describing data driven models of neurons and networks with a high degree of biological detail. *PLoS computational biology*, 6(6):e1000815, 2010. 28

[106] N.H. Goddard, M. Hucka, F. Howell, H. Cornelis, K. Shankar, and D. Beeman. Towards neuroml: model description methods for collaborative modelling in neuroscience. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 356 (1412):1209, 2001. 24

[107] P.S. Goldman-Rakic. Cellular basis of working memory. *Neuron*, 14(3):477, 1995. 4, 176

[108] D. Goodman and R. Brette. Brian: a simulator for spiking neural networks in python. *Frontiers in neuroinformatics*, 2, 2008. 26

[109] B. Gottwald, Z. Mihajlovic, B. Wilde, and H.M. Mehdorn. Does the cerebellum contribute to specific aspects of attention? *Neuropsychologia*, 41(11):1452 – 1460, 2003. ISSN 0028-3932. doi: DOI:10.1016/S0028-3932(03)00090-3. 113

[110] J. Gross, L. Timmermann, J. Kujala, M. Dirks, F. Schmitz, R. Salmelin, and A. Schnitzler. The neural basis of intermittent motor control in humans. *Proceedings of the National Academy of Sciences*, 99(4):2299, 2002. 106

[111] A. Gruart, B.G. Schreurs, E.D. del Toro, and J.M. Delgado-García. Kinetic and frequency-domain properties of reflex and conditioned eyelid responses in the rabbit. *Journal of neurophysiology*, 83(2):836, 2000. 70, 73

[112] A. Haith and S. Vijayakumar. Robustness of vor and okr adaptation under kinematics and dynamics transformations. In *Development and Learning, 2007. ICDL 2007. IEEE 6th International Conference on*, pages 37–42. IEEE, 2007. 127, 129

[113] A.F. de C. Hamilton, D.W. Joyce, J.R. Flanagan, C.D. Frith, and D.M. Wolpert. Kinematic cues in perceptual weight judgement and their origins in box lifting. *Psychological Research*, 71(1):13–21, 2007. 163

[114] C. Hansel, D.J. Linden, and E. D Angelo. Beyond parallel fiber ltd: the diversity of synaptic and non-synaptic plasticity in the cerebellum. *nature neuroscience*, 4:467–476, 2001. 64, 70, 104, 112

[115] M.J. Hartmann and J.M. Bower. Oscillatory activity in the cerebellar hemispheres of unrestrained rats. *Journal of neurophysiology*, 80(3):1598, 1998. 69

[116] M.J. Hartmann and J.M. Bower. Tactile responses in the granule cell layer of cerebellar folium crus iia of freely behaving rats. *The Journal of Neuroscience*, 21(10):3549, 2001. 104

[117] R.J. Harvey and R.M.A Napper. Quantitative study of granule and purkinje cells in the cerebellar cortex of the rat. *The Journal of Comparative Neurology*, 274(2):151–157, 1988. 79

[118] R.J. Harvey and R.M.A. Napper. Quantitative studies on the mammalian cerebellum. *Progress in neurobiology*, 36(6):437–463, 1991. 79

[119] E.I. Hassid. A case of language dysfunction associated with cerebellar infarction. *Neurorehabilitation and Neural Repair*, 9(3):157–160, 1995. doi: 10.1177/154596839500900304. 113

[120] M. Häusser and B.A. Clark. Tonic synaptic inhibition modulates neuronal output pattern and spatiotemporal synaptic integration. *Neuron*, 19(3):665–678, 1997. 77, 136

[121] S.A. Heine, S.M. Highstein, and P.M. Blazquez. Golgi cells operate as state-specific temporal filters at the input stage of the cerebellar cortex. *The Journal of Neuroscience*, 30(50):17004, 2010. ISSN 0270-6474. 118

[122] M. Hines. Efficient computation of branched nerve equations. *International journal of bio-medical computing*, 15(1):69–76, 1984. 21

[123] M. Hines. A program for simulation of nerve equations with branching geometries. *International journal of bio-medical computing*, 24(1):55–68, 1989. 21

[124] M.L. Hines and N.T. Carnevale. The neuron simulation environment. *Neural computation*, 9(6):1179–1209, 1997. 15, 21, 22, 165, 187

[125] M.L. Hines and N.T. Carnevale. Neuron: a tool for neuroscientists. *The Neuroscientist*, 7(2):123, 2001. 16

[126] A.L. Hodgkin and A.F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952. 2, 174

[127] F.E. Hoebeek, J.S. Stahl, A.M. Van Alphen, M. Schonewille, C. Luo, M. Rutteman, A. Van den Maagdenberg, P.C. Molenaar, H. Goossens, M.A. Frens, et al. Increased noise level of purkinje cell activities minimizes impact of their modulation during sensorimotor control. *Neuron*, 45(6):953–965, 2005. 70, 73

[128] G. Holmes. The symptoms of acute cerebellar injuries due to gunshot injuries. *Brain*, 40 (4):461–535, 1917. doi: 10.1093/brain/40.4.461. 123

[129] G. Holmes. On the clinical symptoms of cerebellar disease and their interpretation. *Lancet 1*, pages 1177–1182, 1922. 123

[130] T. Holtzman, N.L. Cerminara, S.A. Edgley, and R. Apps. Characterization in vivo of bilaterally branching pontocerebellar mossy fibre to golgi cell inputs in the rat cerebellum. *European Journal of Neuroscience*, 29(2):328–339, 2009. 103

[131] T. Honda, T. Yamazaki, S. Tanaka, and T. Nishino. A possible mechanism for controlling timing representation in the cerebellar cortex. In Liqing Zhang, Bao-Liang Lu, and James Kwok, editors, *Advances in Neural Networks - ISNN 2010*, volume 6063 of *Lecture Notes in Computer Science*, pages 67–76. Springer Berlin / Heidelberg, 2010. 123

[132] J.C. Houk, J.T. Buckingham, and A.G. Barto. Models of the cerebellum and motor learning. *Behavioral and Brain Sciences*, 19(3):368–383, 1996. 111, 129

[133] E.J. Hwang and R. Shadmehr. Internal models of limb dynamics and the encoding of limb state. *Journal of neural engineering*, 2:S266, 2005. 129

[134] P. Isope, S. Dieudonné, and B. Barbour. Temporal organization of activity in the cerebellar cortex: a manifesto for synchrony. *Ann. NY Acad. Sci*, 978:164–174, 2002. 74

[135] M. Ito. *The cerebellum and neural control*. New York, Raven press, 1984. 66, 123

[136] M. Ito. Cerebellar long-term depression: characterization, signal transduction, and functional roles. *Physiological reviews*, 81(3):1143, 2001. 112, 144

[137] M. Ito. Cerebellar circuitry as a neuronal machine. *Progress in neurobiology*, 78(3-5): 272–303, 2006. 72, 111, 112

[138] M. Ito and M. Kano. Long-lasting depression of parallel fiber-purkinje cell transmission induced by conjunctive stimulation of parallel fibers and climbing fibers in the cerebellar cortex. *Neuroscience letters*, 33(3):253–258, 1982. 69, 112, 144

[139] M. Ito, Y. Miyashita, and E.T. Rolls. *Cognition, computation, and consciousness.* Oxford University Press, USA, 1997. 5, 177

[140] E.M. Izhikevich. Simple model of spiking neurons. *Neural Networks, IEEE Transactions on*, 14(6):1569–1572, 2003. 23

[141] G.A. Jacobson, D. Rokni, and Y. Yarom. A model of the olivo-cerebellar system as a temporal pattern generator. *Trends in neurosciences*, 31(12):617–625, 2008. 70, 112

[142] G.A. Jacobson, I. Lev, Y. Yarom, and D. Cohen. Invariant phase structure of olivo-cerebellar oscillations and its putative role in temporal pattern generation. *Proceedings of the National Academy of Sciences*, 106(9):3579, 2009. 112

[143] D. Jaeger. No parallel fiber volleys in the cerebellar cortex: evidence from cross-correlation analysis between purkinje cells in a computer model and in recordings from anesthetized rats. *Journal of Computational Neuroscience*, 14(3):311–327, 2003. 136

[144] D. Jaeger, E. De Schutter, and J.M. Bower. The role of synaptic and voltage-gated currents in the control of purkinje cell spiking: a modeling study. *The Journal of neuroscience*, 17(1):91, 1997. 136

[145] C.E. Jahr and C.F. Stevens. Voltage dependence of nmda-activated macroscopic conductances predicted by single-channel kinetics. *The Journal of Neuroscience*, 10(9):3178, 1990. 78

[146] R.L. Jakab and J. Hamori. Quantitative morphology and synaptology of cerebellar glomeruli in the rat. *Anatomy and embryology*, 179(1):81–88, 1988. ISSN 0340-2061. 120

[147] R.S. Johansson and I. Birznieks. First spikes in ensembles of human tactile afferents code complex spatial fingertip events. *Nature Neuroscience*, 7(2):170–177, 2004. 73

[148] H. Jörntell and C.F. Ekerot. Properties of somatosensory synaptic integration in cerebellar granule cells in vivo. *The Journal of neuroscience*, 26(45):11786, 2006. 64, 83, 102, 106

[149] E.R. Kandel, J. Schwartz, and T. Jessell, editors. *Principles of Neuroscience.* Elsevier, 1981. 14, 15, 67, 109, 132

[150] R.T. Kanichay and R.A. Silver. Synaptic and cellular properties of the feedforward inhibitory circuit within the input layer of the cerebellar cortex. *The Journal of Neuroscience*, 28(36):8955, 2008. 64, 85, 103, 105

[151] M. Kano. Plasticity of inhibitory synapses in the brain: a possible memory mechanism that has been overlooked. *Neuroscience research*, 21(3):177–182, 1995. 70

[152] M. Kase, D.C. Miller, and H. Noda. Discharges of purkinje cells and mossy fibres in the cerebellar vermis of the monkey during saccadic eye movements and fixation. *The Journal of physiology*, 300(1):539, 1980. 66

[153] M. Kawato. Brain controlled robots. *HFSP journal*, 2(3):136–142, 2008. 111, 144

[154] M. Kawato and H. Gomi. A computational model of four regions of the cerebellum based on feedback-error learning. *Biological cybernetics*, 68(2):95–103, 1992. 112

[155] M. Kawato and D. Wolpert. Internal models for motor control. In *Novartis Foundation Symposium 218-Sensory Guidance of Movement*, pages 291–307. Wiley Online Library, 1998. 154

[156] M. Kawato, K. Furukawa, and R. Suzuki. A hierarchical neural-network model for control and learning of voluntary movement. *Biological cybernetics*, 57(3):169–185, 1987. 129

[157] J.G. Keating and W.T. Thach. Nonclock behavior of inferior olive neurons: interspike interval of purkinje cell complex spike discharge in the awake behaving monkey is random. *Journal of neurophysiology*, 73(4):1329, 1995. 144

[158] J.A.S. Kelso. *Dynamic patterns: The self-organization of brain and behavior*. The MIT Press, 1995. 4, 176

[159] R.E. Kettner, S. Mahamud, H.C. Leung, N. Sitkoff, J.C. Houk, B.W. Peterson, and A.G. Barto. Prediction of complex two-dimensional trajectories by a cerebellar model of smooth pursuit eye movement. *Journal of Neurophysiology*, 77(4):2115, 1997. 127, 141, 144

[160] S. Khemaissia and A. Morris. Use of an artificial neuroadaptive robot model to describe adaptive and learning motor mechanisms in the central nervous system. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 28(3):404–416, 1998. 111

[161] S. Khosrovani, R.S. Van Der Giessen, C.I. De Zeeuw, and M.T.G. De Jeu. In vivo mouse inferior olive neurons exhibit heterogeneous subthreshold oscillations and spiking patterns. *Proceedings of the National Academy of Sciences*, 104(40):15911, 2007. 69

[162] W.M. Kistler and C.I. De Zeeuw. Time windows and reverberating loops: a reverse-engineering approach to cerebellar function. *The Cerebellum*, 2(1):44–54, 2003. 65

[163] C. Koch. *The quest for consciousness: A neurobiological approach*. Roberts & Company Publishers, 2004. 5, 177

[164] S.K.E. Koekkoek, W.L. Den Ouden, G. Perry, S.M. Highstein, and C.I. De Zeeuw. Monitoring kinetic and frequency-domain properties of eyelid responses in mice with magnetic distance measurement technique. *Journal of neurophysiology*, 88(4):2124, 2002. 73

[165] L. Korbo, B.B. Andersen, O. Ladefoged, and A. Moller. Total numbers of various cell types in rat cerebellar cortex estimated using an unbiased stereological method. *Brain research*, 609(1-2):262–268, 1993. 79

[166] M.A. Kramer, A.K. Roopun, L.M. Carracedo, R.D. Traub, M.A. Whittington, and N.J. Kopell. Rhythm generation through period concatenation in rat somatosensory cortex. *PLoS Computational Biology*, 4(9):e1000169, 2008. 104

[167] A.C. Kreitzer, A.G. Carter, and W.G. Regehr. Inhibition of interneuron firing extends the spread of endocannabinoid signaling in the cerebellum. *Neuron*, 34(5):787–796, 2002. 77

[168] S. Kuroda, K. Yamamoto, H. Miyamoto, K. Doya, and M. Kawato. Statistical characteristics of climbing fiber spikes necessary for efficient cerebellar learning. *Biological cybernetics*, 84(3):183–192, 2001. 112, 115, 124, 144

[169] J. Lainé and H. Axelrad. Lugaro cells target basket and stellate cells in the cerebellar cortex. *Neuroreport*, 9(10):2399, 1998. 65

[170] H.C. Leiner, A.L. Leiner, and R.S. Dow. Cognitive and language functions of the human cerebellum. *Trends in Neurosciences*, 16(11):444–447, 1993. 108

[171] V. Lev-Ram, S.B. Mehta, D. Kleinfeld, and R.Y. Tsien. Reversing cerebellar long-term depression. *Proceedings of the National Academy of Sciences of the United States of America*, 100(26):15989, 2003. 142, 144

[172] L. Lin, R. Osan, and J.Z. Tsien. Organizing principles of real-time memory encoding: neural clique assemblies and universal neural codes. *TRENDS in Neurosciences*, 29(1): 48–57, 2006. 4, 177

[173] A. Linares-Barranco, M. Oster, D. Cascado, G. Jiménez, A. Civit, and B. Linares-Barranco. Inter-spike-intervals analysis of aer poisson-like generator hardware. *Neurocomputing*, 70(16-18):2692–2700, 2007. 144

[174] R. Linsker. From basic network principles to neural architecture (series). *Proc. Natl. Acad. Sci. USA*, 83(75082):7512, 1986. 4, 176

[175] S.G. Lisberger and T.J. Sejnowski. Motor learning in a recurrent network model based on the vestibulo–ocular reflex. *Nature*, 360(6400):159–161, 1992. 72

[176] R. Llinas and Y. Yarom. Properties and distribution of ionic conductances generating electroresponsiveness of mammalian inferior olivary neurones in vitro. *The Journal of Physiology*, 315(1):569, 1981. 69

[177] R. Llinás, E.J. Lang, and J.P. Welsh. The cerebellum, ltd, and memory: alternative views. *Learning & Memory*, 3(6):445, 1997. 70

[178] R.R. Llinas. The intrinsic electrophysiological properties of mammalian neurons: insights into central nervous system function. *Science*, 242(4886):1654, 1988. 73

[179] Y. Loewenstein, S. Mahon, P. Chadderton, K. Kitamura, H. Sompolinsky, Y. Yarom, and M. Hausser. Bistability of cerebellar purkinje cells modulated by sensory stimulation. *Nature neuroscience*, 8(2):202–211, 2005. 70

[180] L. Luciani. *Il cerveletto: Nuovi studi di fisiologia normale e patologica.* R. Istituto di studi superiori pratici e di perfezionamento, 1891. 123

[181] N.R. Luque, J.A. Garrido, R.R. Carrillo, and E. Ros. Cerebellar spiking engine: Towards objet model abstraction in manipulation. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–8. IEEE, 2010. 118

[182] N.R. Luque, J.A. Garrido, R.R. Carrillo, O. Coenen, and E. Ros. Cerebellar-like corrective-model inference engine for manipulation tasks. *Accepted in IEEE T. on System, Man and Cybernetics B*, 2011. 124

[183] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002. doi: 10.1162/089976602760407955. 115, 122

[184] W. Maass, P. Joshi, and E.D. Sontag. Computational aspects of feedback in neural circuits. *PLOS Computational Biology*, 3(1):e165, 2007. 74

[185] R. Maex and E.D. Schutter. Synchronization of golgi and granule cell firing in a detailed network model of the cerebellar granule cell layer. *Journal of neurophysiology*, 80(5):2521, 1998. 81, 82, 96, 98, 100, 102, 103, 105, 115, 119, 121, 124, 125

[186] A. Maffei, F. Prestori, P. Rossi, V. Taglietti, and E. D'Angelo. Presynaptic current changes at the mossy fiber–granule cell synapse of cerebellum during ltp. *Journal of neurophysiology*, 88(2):627, 2002. 68

[187] J. Magistretti, L. Castelli, L. Forti, and E. D'Angelo. Kinetic and functional analysis of transient, persistent and resurgent sodium currents in rat cerebellar granule cells in situ: an electrophysiological and modelling study. *The Journal of Physiology*, 573(1):83, 2006. 105

[188] T. Makino. A discrete-event neural network simulator for general neural models. *Neural Computing and Applications*, 11:210–223, 2003. 15

[189] P. Mann-Metzer and Y. Yarom. Electrotonic coupling interacts with intrinsic properties to generate synchronized activity in cerebellar networks of inhibitory interneurons. *The Journal of neuroscience*, 19(9):3298, 1999. 96, 103

[190] J. Mapelli and E. D'Angelo. The spatial organization of long-term synaptic plasticity at the input stage of cerebellum. *The Journal of neuroscience*, 27(6):1285, 2007. 64, 68, 79, 85, 103, 105

[191] J. Mapelli, D. Gandolfi, and E. D'Angelo. Combinatorial responses controlled by synaptic inhibition in the cerebellum granular layer. *Journal of neurophysiology*, 103(1):250, 2010. 64, 106

[192] D. Marr. A theory of cerebellar cortex. *The journal of physiology*, 202(2):437, 1969. 6, 64, 83, 102, 112, 116, 178

[193] S.G. Massaquoi and J.J.E. Slotine. The intermediate cerebellum may function as a wave-variable processor. *Neuroscience Letters*, 215(1):60–64, 1996. 109

[194] N. Masuda and B. Doiron. Gamma oscillations of spiking neural populations enhance signal discrimination. *PLoS computational biology*, 3(11):e236, 2007. 103

[195] A. Mathy, S.S.N. Ho, J.T. Davie, I.C. Duguid, B.A. Clark, and M. Häusser. Encoding of oscillations by axonal bursts in inferior olive neurons. *Neuron*, 62(3):388–399, 2009. 112

[196] J.F. Medina and M.D. Mauk. Computer simulation of cerebellar information processing. *Nature Neuroscience*, 3:1205–1211, 2000. 105, 122

[197] R.C. Miall. The cerebellum, predictive control and motor coordination. In *Novartis Foundation Symposium 218-Sensory Guidance of Movement*, pages 272–290. Wiley Online Library, 1998. 111

[198] R.C. Miall, J.G. Keating, M. Malkmus, and W.T. Thach. Simple spike activity predicts occurrence of complex spikes in cerebellar purkinje cells. *nature neuroscience*, 1(1):13–15, 1998. 69

[199] S.J. Middleton, C. Racca, M.O. Cunningham, R.D. Traub, H. Monyer, T. Knopfel, I.S. Schofield, A. Jenkins, and M.A. Whittington. High-frequency network oscillations in cerebellar cortex. *Neuron*, 58(5):763–774, 2008. 69, 103, 106

[200] M. Möck, S. Butovas, and C. Schwarz. Functional unity of the ponto-cerebellum: evidence that intrapontine communication is mediated by a reciprocal loop with the cerebellar nuclei. *Journal of neurophysiology*, 95(6):3414, 2006. 67

[201] J.W. Moore, J.E. Desmond, and N.E. Berthier. Adaptively timed conditioned responses and the cerebellum: a neural network approach. *Biological Cybernetics*, 62(1):17–28, 1989. ISSN 0340-1200. 119

[202] A. Morrison, C. Mehring, T. Geisel, A.D. Aertsen, and M. Diesmann. Advancing the boundaries of high-connectivity network simulation with distributed computing. *Neural Computation*, 17(8):1776–1801, 2005. 24, 25

[203] A. Morrison, S. Straube, H.E. Plesser, and M. Diesmann. Exact subthreshold integration with continuous spike times in discrete-time neural network simulations. *Neural Computation*, 19(1):47–79, 2007. 24

[204] E. Mugnaini, R.L. Atluri, and J.C. Houk. Fine structure of granular layer in turtle cerebellum with emphasis on large glomeruli. *Journal of Neurophysiology*, 37(1):1, 1974. 139

[205] Z. Nadasdy, G. Buzsaki, and L. Zaborszky. Functional connectivity of the brain: Reconstruction from static and dynamic data. *Neuroanatomical Tract-Tracing 3*, pages 631–681, 2006. 64

[206] E. Nakano, H. Imamizu, R. Osu, Y. Uno, H. Gomi, T. Yoshioka, and M. Kawato. Quantitative examinations of internal representations for arm trajectory planning: minimum commanded torque change model. *Journal of Neurophysiology*, 81(5):2140, 1999. 129, 130

[207] Z. Nenadic, B.K. Ghosh, and P. Ulinski. Propagating waves in visual cortex: A large-scale model of turtle visual cortex. *Journal of computational neuroscience*, 14(2):161–184, 2003. 23

[208] T. Nieus, E. Sola, J. Mapelli, E. Saftenku, P. Rossi, and E. D'angelo. Ltp regulates burst initiation and frequency at mossy fiber–granule cell synapses of rat cerebellum: experimental observations and theoretical predictions. *Journal of neurophysiology*, 95(2): 686, 2006. 64, 68, 77, 106

[209] Z. Nusser, S. Cull-Candy, and M. Farrant. Differences in synaptic gabaa receptor number underlie variation in gaba mini amplitude. *Neuron*, 19(3):697–709, 1997. 77

[210] S.M. O'Connor, R.W. Berg, and D. Kleinfeld. Coherent electrical activity between vibrissa sensory areas of cerebellum and neocortex is enhanced during free whisking. *Journal of neurophysiology*, 87(4):2137, 2002. 73

[211] R.C. O'Reilly and Y. Munakata. *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain.* MIT Press, Cambridge, 2000. 15

[212] O.P. Ottersen and J. Storm-Mathisen. Glutamate. *Handbook of chemical neuroanatomy*, 18, 2000. 82

[213] J.B. Passot, N. Luque, and A. Arleo. Internal models in the cerebellum: a coupling scheme for online and offline learning in procedural tasks. *From Animals to Animats 11*, pages 435–446, 2010. 118

[214] M. G. Paulin. The role of the cerebellum in motor control and perception. *Brain Behav Evol*, 41(1):39–50, 1993. 113

[215] J.P. Pellerin and Y. Lamarre. Local field potential oscillations in primate cerebellar cortex during voluntary movement. *Journal of neurophysiology*, 78(6):3502, 1997. 69, 104

[216] D. Philipona and O.J. Coenen. Model of granular layer encoding in the cerebellum. *Neurocomputing*, 58:575–580, 2004. 109

[217] A. Pijpers, R. Apps, J. Pardoe, J. Voogd, and T.J.H. Ruigrok. Precise spatial relationships between mossy fibers and climbing fibers in rat cerebellar cortical zones. *The Journal of neuroscience*, 26(46):12067, 2006. 65

[218] J. Porrill and P. Dean. Cerebellar motor learning: when is cortical plasticity not enough? *PLoS computational biology*, 3(10):e197, 2007. 64, 74

[219] J. Porrill, P. Dean, and J.V. Stone. Recurrent cerebellar architecture solves the motor-error problem. *Proceedings of the Royal Society of London-B*, 271(1541):789–796, 2004. 129

[220] J.R. Pugh and I.M. Raman. Mechanisms of potentiation of mossy fiber epscs in the cerebellar nuclei by coincident synaptic excitation and inhibition. *The Journal of Neuroscience*, 28(42):10549, 2008. 72

[221] D. Purves, G.J. Augustine, D. Fitzpatrick, L.C. Katz, A.S. LaMantia, J.O. McNamara, and S.M. Williams. Neuroscience. sunderland, ma. *Sinauer Associates, Inc*, 235:487, 1997. 148

[222] I.M. Raman and B.P. Bean. Ionic currents underlying spontaneous action potentials in isolated cerebellar purkinje neurons. *The Journal of neuroscience*, 19(5):1663, 1999. 112

[223] E.A. Rancz, T. Ishikawa, I. Duguid, P. Chadderton, S. Mahon, and M. Hausser. High-fidelity transmission of sensory information by single cerebellar mossy fibre boutons. *Nature*, 450(7173):1245–1248, 2007. 66, 106

[224] J.L. Raymond and S.G. Lisberger. Neural learning rules for the vestibulo-ocular reflex. *The Journal of neuroscience*, 18(21):9112, 1998. 111, 144

[225] Q. Robberechts, M. Wijnants, M. Giugliano, and E. De Schutter. Long-term depression at parallel fiber to golgi cell synapses. *Journal of neurophysiology*, 104(6):3413, 2010. 105

[226] E.T. Rolls, A. Treves, and E.T. Rolls. *Neural networks and brain function.* Oxford University Press Oxford, 1998. 4, 177

[227] E. Ros, R. Carrillo, E.M. Ortigosa, B. Barbour, and R. Agís. Event-driven simulation scheme for spiking neural networks using lookup tables to characterize neuronal dynamics. *Neural computation*, 18(12):2959–2993, 2006. 7, 8, 14, 15, 18, 30, 32, 46, 47, 49, 65, 75, 77, 106, 110, 126, 142, 179, 180

[228] E. Ros, E.M. Ortigosa, R. Agís, R. Carrillo, and M. Arnold. Real-time computing platform for spiking neurons (rt-spike). *Neural Networks, IEEE Transactions on*, 17(4):1050–1063, 2006. 126

[229] D.J. Rossi and M. Hamann. Spillover-mediated transmission at inhibitory synapses promoted by high affinity [alpha] 6 subunit gabaa receptors and glomerular geometry. *Neuron*, 20(4):783–795, 1998. 77

[230] P. Rossi, E. Sola, V. Taglietti, T. Borchardt, F. Steigerwald, K. Utvik, OP Ottersen, G. Kohr, and E. D?Angelo. Cerebellar synaptic excitation and plasticity require proper nmda receptor positioning and density in granule cells. *J Neurosci*, 22:9687–9697, 2002. 68

[231] P. Rossi, L. Mapelli, L. Roggeri, D. Gall, A. De Kerchove, S.N. Schiffmann, V. Taglietti, and E. D'Angelo. Inhibition of constitutive inward rectifier currents in cerebellar granule cells by pharmacological and synaptic activation of gabab receptors. *European Journal of Neuroscience*, 24(2):419–432, 2006. 104

[232] A. Roth and M. Häusser. Compartmental models of rat cerebellar purkinje cells based on simultaneous somatic and dendritic patch-clamp recordings. *The Journal of physiology*, 535(2):445, 2001. 136

[233] M. Schonewille, S. Khosrovani, B.H.J. Winkelman, F.E. Hoebeek, M.T.G. De Jeu, I.M. Larsen, J. Van der Burg, M.T. Schmolesky, M.A. Frens, and C.I. De Zeeuw. Purkinje cells in awake behaving animals operate at the upstate membrane potential. *Nature Neuroscience*, 9(4):459–461, 2006. 70

[234] E.D. Schutter, B. Vos, and R. Maex. The function of cerebellar golgi cells revisited. *Progress in Brain Research*, 124:81–93, 2000. 104

[235] C. Schwarz and P. Thier. Binding of signals relevant for action: towards a hypothesis of the functional role of the pontine nuclei. *Pan*, 124:4759–4767, 1999. 67

[236] N. Schweighofer, J. Spoelstra, M.A. Arbib, and M. Kawato. Role of the cerebellum in reaching movements in humans. ii. a neural model of the intermediate cerebellum. *European Journal of Neuroscience*, 10(1):95–105, 1998. 112, 118

[237] N. Schweighofer, K. Doya, and F. Lay. Unsupervised learning of granule cell sparse codes enhances cerebellar adaptive control. *Neuroscience*, 103(1):35–50, 2001. 65, 104, 118

[238] N. Schweighofer, K. Doya, H. Fukai, J.V. Chiron, T. Furukawa, and M. Kawato. Chaos may enhance information transmission in the inferior olive. *Proceedings of the National Academy of Sciences of the United States of America*, 101(13):4655, 2004. 109, 112, 115, 118, 124, 144

[239] T.J. Sejnowski. Storing covariance with nonlinearly interacting neurons. *Journal of mathematical biology*, 4(4):303–321, 1977. 116, 142

[240] T.J. Sejnowski. *Neural codes and distributed representations: foundations of neural computation*. The MIT Press, 1999. 4, 177

[241] S.L. Shin, F.E. Hoebeek, M. Schonewille, C.I. De Zeeuw, A. Aertsen, and E. De Schutter. Regular patterns in cerebellar purkinje cell simple spike trains. *PLoS One*, 2(5):e485, 2007. 70, 73

[242] R.A. Silver, S.F. Traynelis, and S.G. Cull-Candy. Rapid-time-course miniature and evoked excitatory currents at cerebellar synapses in situ. *Nature*, 335:163–166, 1992. 68

[243] R.A. Silver, D. Colquhoun, S.G. Cull-Candy, and B. Edmonds. Deactivation and desensitization of non-nmda receptors in patches and the time course of epscs in rat cerebellar granule cells. *The Journal of Physiology*, 493(Pt 1):167, 1996. 77

[244] J.I. Simpson, D.R. Wylie, and C.I. De Zeeuw. On climbing fiber signals and their consequence (s). *Behavioral and Brain Sciences*, 19(3):384–398, 1996. 112

[245] R.E. Sims and N.A. Hartell. Differential susceptibility to synaptic plasticity reveals a functional specialization of ascending axon and parallel fiber synapses to cerebellar purkinje cells. *The Journal of neuroscience*, 26(19):5153, 2006. 74

[246] R. Soetedjo and A.F. Fuchs. Complex spike activity of purkinje cells in the oculomotor vermis during behavioral adaptation of monkey saccades. *The Journal of neuroscience*, 26(29):7741, 2006. 112

[247] E. Sola, F. Prestori, P. Rossi, V. Taglietti, and E. D'Angelo. Increased neurotransmitter release during long-term potentiation at mossy fibre-granule cell synapses in rat cerebellum. *The Journal of physiology*, 557(3):843, 2004. 68, 98

[248] S. Solinas, R. Maex, and E. De Schutter. Synchronization of purkinje cell pairs along the parallel fiber axis: a model. *Neurocomputing*, 52:97–102, 2003. 136

[249] S. Solinas, L. Forti, E. Cesana, J. Mapelli, E. De Schutter, and E. D'Angelo. Computational reconstruction of pacemaking and intrinsic electroresponsiveness in cerebellar golgi cells. *Frontiers in Cellular Neuroscience*, 1, 2007. 64, 68, 69, 77, 92, 103, 105

[250] S. Solinas, L. Forti, E. Cesana, J. Mapelli, E. De Schutter, and E. D'Angelo. Fast-reset of pacemaking and theta-frequency resonance patterns in cerebellar golgi cells: simulations of their impact in vivo. *Frontiers in Cellular Neuroscience*, 1, 2007. 64, 67, 68, 69, 77, 92, 103, 105, 106

[251] S. Solinas, T. Nieus, and E. D'Angelo. A realistic large-scale model of the cerebellum granular layer predicts circuit spatio-temporal filtering properties. *Frontiers in Cellular Neuroscience*, 4, 2010. 22, 64, 65, 103, 104, 105, 106, 115, 120, 121, 124, 125

[252] J. Spoelstra, M.A. Arbib, and N. Schweighofer. Cerebellar adaptive control of a biomimetic manipulator. *Neurocomputing*, 26:881–889, 1999. 112

[253] J. Spoelstra, N. Schweighofer, and M.A. Arbib. Cerebellar learning of accurate predictive control for fast-reaching movements. *Biological cybernetics*, 82(4):321–333, 2000. 144

[254] V. Steuber, D. Willshaw, and A. Van Ooyen. Generation of time delays: Simplified models of intracellular signalling in cerebellar purkinje cells. *Network: Computation in Neural Systems*, 17(2):173–191, 2006. 109

[255] V. Steuber, W. Mittmann, F.E. Hoebeek, R.A. Silver, C.I. De Zeeuw, M. Hausser, and E. De Schutter. Cerebellar ltd and pattern recognition by purkinje cells. *Neuron*, 54(1): 121–136, 2007. 70

[256] B. Stricanne and J.M. Bower. A network model of the somatosensory system cerebellum, exploring recovery from peripheral lesions at various developmental stages in rats. In *Society of Neuroscience Abstracts*, volume 24, page 669, 1998. 23

[257] K.J. Suter and D. Jaeger. Reliable control of spike rate and spike timing by rapid input transients in cerebellar stellate cells. *Neuroscience*, 124(2):305–317, 2004. 77

[258] R.S. Sutton and A.G. Barto. Toward a modern theory of adaptive networks: expectation and prediction. *Psychological review*, 88(2):135, 1981. 141

[259] G. Szapiro and B. Barbour. Multiple climbing fibers signal to molecular layer interneurons exclusively via glutamate spillover. *Nature neuroscience*, 10(6):735–742, 2007. 72

[260] C. Takayama. Formation of gabaergic synapses in the cerebellum. *The Cerebellum*, 4(3): 171–177, 2005. 64

[261] P. Telgkamp and I.M. Raman. Depression of inhibitory synaptic transmission between purkinje cells and neurons of the cerebellar nuclei. *The Journal of neuroscience*, 22(19): 8447, 2002. 72

[262] S. Tia, J.F. Wang, N. Kotchabhakdi, and S. Vicini. Developmental changes of inhibitory synaptic currents in cerebellar granule neurons: role of gabaa receptor $\alpha 6$ subunit. *The Journal of neuroscience*, 16(11):3630, 1996. 77

[263] D.B. Timmermann, R.E. Westenbroek, A. Schousboe, and W.A. Catterall. Distribution of high-voltage-activated calcium channels in cultured γ-aminobutyric acidergic neurons from mouse cerebral cortex. *Journal of neuroscience research*, 67(1):48–61, 2002. 103

[264] E. Todorov. Optimality principles in sensorimotor control (review). *Nature neuroscience*, 7(9):907, 2004. 129

[265] J.R. Trott, R. Apps, and D.M. Armstrong. Zonal organization of cortico-nuclear and nucleo-cortical projections of the paramedian lobule of the cat cerebellum. 2. the c2 zone. *Experimental brain research*, 118(3):316–330, 1998. 71, 72

[266] M. Uusisaari and T. Knöpfel. Gabaergic synaptic communication in the gabaergic and non-gabaergic cells in the deep cerebellar nuclei. *Neuroscience*, 156(3):537–549, 2008. 72

[267] M. Uusisaari, K. Obata, and T. Knöpfel. Morphological and electrophysiological properties of gabaergic and non-gabaergic cells in the deep cerebellar nuclei. *Journal of neurophysiology*, 97(1):901, 2007. 72

[268] R.S. Van Der Giessen, S.K. Koekkoek, S. van Dorp, J.R. De Gruijl, A. Cupido, S. Khosrovani, B. Dortland, K. Wellershaus, J. Degen, J. Deuchars, et al. Role of olivary electrical coupling in cerebellar motor learning. *Neuron*, 58(4):599–612, 2008. 69

[269] P.L. Van Kan, J.C. Houk, and A.R. Gibson. Output organization of intermediate cerebellum of the monkey. *Journal of neurophysiology*, 69(1):57, 1993. 70

[270] P.L. Van Kan, K.M. Horn, and A.R. Gibson. The importance of hand use to discharge of interpositus neurones of the monkey. *The Journal of physiology*, 480(Pt 1):171, 1994. 66

[271] H. Voicu and M.D. Mauk. Parametric analysis of cerebellar ltd in eyelid conditioning. *Neurocomputing*, 69(10-12):1187–1190, 2006. 111

[272] J. Voogd, J. Pardoe, T.J.H. Ruigrok, and R. Apps. The distribution of climbing and mossy fiber collateral branches from the copula pyramidis and the paramedian lobule: congruence of climbing fiber cortical zones and the pattern of zebrin banding within the rat cerebellum. *The Journal of neuroscience*, 23(11):4645, 2003. 65

[273] B.P. Vos, R. Maex, and E. De Schutter. Correlation of firing between rat cerebellar golgi cells. In *27th Annual Meeting of the Society for Neuroscience, Louisiana, USA*, October 1997. 103

[274] B.P. Vos, R. Maex, A. Volny-Luraghi, and E. De Schutter. Parallel fibers synchronize spontaneous activity in cerebellar golgi cells. *Journal of Neuroscience*, 19, 1999. 69, 103

[275] N. Wada, Y. Kishimoto, D. Watanabe, M. Kano, T. Hirano, K. Funabiki, and S. Nakanishi. Conditioned eyeblink learning is formed and stored without cerebellar granule cell transmission. *Proceedings of the National Academy of Sciences*, 104(42):16690, 2007. 72

[276] M.J. Wall and M.M. Usowicz. Development of Action Potential-dependent and Independent Spontaneous GABAA Receptor-mediated Currents in Granule Cells of Postnatal Rat Cerebellum. *European Journal of Neuroscience*, 9(3):533–548, 1997. ISSN 1460-9568. 120

[277] D. Watanabe, H. Inokawa, K. Hashimoto, N. Suzuki, M. Kano, R. Shigemoto, T. Hirano, K. Toyama, S. Kaneko, M. Yokoi, et al. Ablation of cerebellar golgi cells disrupts synaptic

integration involving gaba inhibition and nmda receptor activation in motor coordination. *Cell*, 95(1):17–27, 1998. 81

[278] L. Watts. Event-driven simulation of networks of spiking neurons. *Advances in neural information processing systems*, 6:927–934, 1994. 15

[279] B. Widrow and S.D. Stearns. Adaptive signal processing. *Englewood Cliffs, NJ, Prentice-Hall, Inc., 1985, 491 p.*, 1, 1985. 142

[280] C.J. Willmott and K. Matsuura. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate Research*, 30(1):79, 2005. 146

[281] B. Winkelman and M. Frens. Motor coding in floccular climbing fibers. *Journal of neurophysiology*, 95(4):2342, 2006. ISSN 0022-3077. 118

[282] U. Wolf, M.J. Rapoport, and T.A. Schweizer. Evaluating the affective component of the cerebellar cognitive affective syndrome. *J. Neuropsychiatry Clin. Neurosci.*, 21 (3): 245–253, 2009. 5, 113, 177

[283] D.M. Wolpert and Z. Ghahramani. Computational principles of movement neuroscience. *nature neuroscience*, 3:1212–1217, 2000. 112

[284] D.M. Wolpert, R.C. Miall, and M. Kawato. Internal models in the cerebellum. *Trends in Cognitive Sciences*, 2(9):338–347, 1998. 111

[285] S. Wu, S. Amari, and H. Nakahara. Population coding and decoding in a neural field: a computational study. *Neural Computation*, 14(5):999–1026, 2002. ISSN 0899-7667. 119

[286] W. Xu and S.A. Edgley. Climbing fibre-dependent changes in golgi cell responses to peripheral stimulation. *The Journal of Physiology*, 586(20):4951, 2008. 69, 72

[287] T. Yamazaki and S. Tanaka. Neural modeling of an internal clock. *Neural Comput*, 17 (5):1032–1058, May 2005. doi: 10.1162/0899766053491850. 122, 125

[288] T. Yamazaki and S. Tanaka. The cerebellum as a liquid state machine. *Neural Networks*, 20(3):290–297, 2007. 105, 115, 120, 122, 124, 125, 131, 139

[289] T. Yamazaki and S. Tanaka. A spiking network model for passage-of-time representation in the cerebellum. *Eur J Neurosci*, 26(8):2279–2292, Oct 2007. doi: 10.1111/j.1460-9568.2007.05837.x. 123, 124

[290] T. Yamazaki and S. Tanaka. Computational models of timing mechanisms in the cerebellar granular layer. *Cerebellum*, 8(4):423–432, Dec 2009. doi: 10.1007/s12311-009-0115-7. 122, 125