UNIVERSITY OF CALIFORNIA,
IRVINE

Cortical Neural Network Models of Visual Motion Perception for Decision-Making and Reactive Navigation

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Michael Beyeler

Dissertation Committee:
Professor Jeffrey L. Krichmar, Co-Chair
Professor Nikil D. Dutt, Co-Chair
Professor Charless C. Fowlkes

2016

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF LISTINGS

ACKNOWLEDGMENTS

First and foremost I would like to thank my PhD advisors, Profs. Jeff Krichmar and Nik Dutt. It was Jeff who gave me the chance to conduct my Master's Thesis in his lab back when I first dared to cross the pond and move to Irvine, and it was Nik who subsequently stepped in and helped us revive the "Computational Neuroscience" track in the Computer Science department at UCI. Both of them were excellent mentors demonstrating perpetual energy, expertise, and vision in research. I am thankful for everything they have taught me, for their patience and kindness, and for the freedom they gave me to explore new ideas. I also thank Prof. Charless Fowlkes for his efforts in pre-reviewing my thesis work and challenging me to approach my ideas from different angles and explain my work in different ways.

A big thanks must go to my labmates and co-authors of the original publications. Nicolas Oros: thank you for teaching me how to break big projects into feasible steps, and for developing the Android Based Robotics (ABR) platform. Ting-Shuo Chou: thank you for your *joie de vivre*, your endless programming knowledge, and for teaching me how to say "piece of cake" in Mandarin. Kristofor Carlson: thank you for being there and saying things no one expects, for the 2AM writing sessions, for Science Fridays, and of course your (un)disputable soccer expertise. Micah Richert, whose legacy most of my work is based on: thank you for your efforts developing an early version of CARLsim, for steering me towards studying visual motion perception, and for linking STDP to NMF. Your work was a great inspiration to me.

I must also thank my funding sources, without whom this research would not have been possible: the National Science Foundation (NSF), the Defense Advanced Research Projects Agency (DARPA), Qualcomm Technologies Inc., Intel Corporation, Northrop Grumman Aerospace Systems, and the Swiss-American Society (SFUSA).

Finally, I would like to thank my fiancée, family, and friends for all their love and support, and for keeping me sane throughout these years.

MICHAEL BEYELER

mbeyeler@uci.edu ⋄ sites.uci/mbeyeler

EDUCATION

University of California, Irvine Sep 2012 – May 2016
Ph.D. in Computer Science *GPA: 3.96 / 4.0*

Specialization in Computational Neuroscience

Dissertation: "Cortical neural network models of visual motion perception for decision-making and reactive navigation." Dissertation defense: May 17, 2016. Advisors: Jeff Krichmar and Nikil Dutt.

ETH Zurich, Switzerland Sep 2009 – Nov 2011
M.Sc. in Biomedical Engineering *GPA: 5.36 / 6.0*

Focus on Bioelectronics

ETH Zurich, Switzerland Oct 2005 – Feb 2009
B.Sc. in Electrical Engineering *GPA: 4.51 / 6.0*

Major in Micro- and Optoelectronics

PUBLICATIONS

Books

· **Beyeler, M.** (2015). OpenCV with Python Blueprints: Design and develop advanced computer vision projects using OpenCV with Python. *Packt Publishing Ltd.*, London, England, 230 pages, ISBN 978-178528269-0.

Peer-Reviewed Journal Publications

· **Beyeler, M.**, Oros, N., Dutt, N., and Krichmar, J. L. (2015). A GPU-accelerated cortical neural network model for visually guided robot navigation. *Neural Networks* 72: 75–87.

· **Beyeler, M.**, Richert, M., Dutt, N. D., and Krichmar, J. L. (2014). Efficient spiking neural network model of pattern motion selectivity in visual cortex. *Neuroinformatics*, 1–20.

· **Beyeler, M.**, Dutt, N. D., and Krichmar, J. L. (2013). Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule. *Neural Networks* 48C: 109–124.

Peer-Reviewed Conference Publications

· **Beyeler*, M.**, Carlson*, K. D. , Chou*, T-S., Dutt, N., Krichmar, J. L. (2015). CARLsim 3: A user-friendly and highly optimized library for the creation of neurobiologically detailed spiking neural networks. Proceedings of *IEEE International Joint Conference on Neural Networks (IJCNN)*, Killarney, Ireland. (*co-first authors)

· **Beyeler, M.**, Mirus, F., and Verl, A. (2014). Vision-based robust road lane detection in urban environments. Proceedings of *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China.

· **Beyeler*, M.**, Stefanini*, F., Proske, H., Galizia, C. G., and Chicca, E. (2010). Exploring olfactory sensory networks: simulations and hardware emulation. Proceedings of *Biomedical Circuits and Systems Conference (BioCAS)*, Paphos, Cyprus. (*co-first authors)

Invited Papers

· Carlson, K. D., **Beyeler, M.**, Dutt, N., and Krichmar, J. L. (2014). GPGPU accelerated simulation and parameter tuning for neuromorphic applications. *19th Asia and South Pacific Design Automation Conference (ASP-DAC)*.

Selected Abstracts

· **Beyeler, M.**, Richert, M., Oros, N., Dutt, N., and Krichmar, J. L. (2016). A cortical neural network model of visual motion perception for decision-making and navigation. Poster presented at *Computational and Systems Neuroscience (Cosyne)*, February 25–28, Salt Lake City, UT.

· **Beyeler, M.**, Carlson, K. D., Chou, T.-S., Dutt, N., and Krichmar, J. L. (2015). An optimized library for the design, simulation, and parameter tuning of biologically detailed spiking neural networks. Poster presented at the *45th Annual Meeting of the Society for Neuroscience (SfN)*, October 17–21, 2015, Chicago, IL.

· **Beyeler, M.**, Dutt, N. D., and Krichmar, J. L. (2013). Spiking neural network model of visual pattern recognition and decision-making using a stochastic STDP learning rule. Poster presented at the *20th Joint Symposium on Neural Computation (JSNC)*, June 1, 2013, California Institute of Technology, Pasadena, CA.

INVITED PRESENTATIONS

· A brain-inspired robot for obstacle avoidance based on visual motion perception. Guest Lecture for PSYCH-268R Cognitive Robotics (Prof. Jeff Krichmar), *University of California, Irvine*, Irvine, CA, April 2016.

· A cortical neural network model for perceptual decision-making and visually guided robot navigation. *UW Institute for Neuroengineering*, University of Washington, Seattle, WA, December 2015.

· CARLsim: Concepts, tools, and applications. Guest Lecture for PSYCH-268A Computational Neuroscience (Prof. Jeff Krichmar), *University of California, Irvine*, Irvine, CA, November 2015.

· TrueNorth implementation of long short-term memory (LSTM). Invited Talk, *IBM Boot Camp*, San Jose, CA, August 2015.

· A short introduction to Computational Neuroscience. Guest Lecture for CS-171 Introduction to Artificial Intelligence (Prof. Richard Lathrop), *University of California, Irvine*, Irvine, CA, March 2015.

· A cortical spiking neural network model for visually guided robot navigation. Invited Talk, *Qualcomm Technologies Incorporated*, San Diego, CA, November 2014.

· A cortical spiking neural network model for visually guided robot navigation. Student Talk, Neurobiologically Inspired Robotics workshop, *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, June 2014. Recognized with workshop award for "Best Student Talk".

· Vision-based robust road lane detection in urban environments. Technical session, *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, June 2014.

· Large-scale spiking neural network model of visual motion processing. Spotlight talk, *Dynamics of Multifunction Brain Networks MURI Winter School*, San Diego, CA, January 2014.

FELLOWSHIPS AND SUPPORT

Travel award for IEEE ICRA 2014 2014
Fraunhofer Institute for Manufacturing Engineering and Automation (IPA)

Stipend 2013
Swiss-American Society (SFUSA)

Chairs Fellowship for outstanding Ph.D. applicants 2012 – present
Donald Bren School of Information and Computer Sciences (ICS), University of California, Irvine

RESEARCH EXPERIENCE

Cognitive Anteaters Robotics Lab (CARL), UC Irvine Sep 2012 – present
Graduate Student Researcher *Irvine, CA*

· Developing large-scale spiking networks of visual and motion perception, navigation, learning, memory, and cognition. Integrating these competences into autonomous brain-inspired robots (Android Based Robotics, eDVS neuromorphic camera) towards solving important real-world problems.
· Co-developing and maintaining CARLsim, an open source software platform for the efficient simulation of large-scale spiking neural networks on x86 and CUDA architectures.
· Performing technical evaluation of Qualcomm's Zeroth Brain-Inspired Computing platform.

Brain-Inspired Computing Group, IBM Research - Almaden Jun 2015 – Aug 2015
Research Intern *San Jose, CA*

· Ported recurrent neural network models to IBM's TrueNorth Neurosynaptic chip (C/C++, MATLAB, Caffe, Corelet Programming).

Fraunhofer IPA Jun 2013 – Sep 2013
Research Assistant *Stuttgart, Germany*

· Developed a fast and robust computer vision algorithm for the automatic detection of ground plane, traffic lane, and traffic lane markings to be used in an autonomous car capable of navigating urban scenes (C/C++, ROS, OpenCV).

Cognitive Anteaters Robotics Lab (CARL), UC Irvine Nov 2011 – Jul 2012
Junior Specialist *Irvine, CA*

· Developed a spiking neural network for classification of handwritten digits and decision-making using a STDP-like learning rule.
· Interacted with Hughes Research Laboratories (HRL), as part of the DARPA SyNAPSE program, to design controllers of autonomous mobile robots based on very-large scale networks of adaptive spiking neurons that are amenable for incorporation into the new generation of low-power high-density VLSI neuromorphic chips (C/C++, Matlab, CUDA).

Lab for Biosensors and Bioelectronics, ETH Zurich Oct 2010 – Dec 2010
Research Assistant *Zurich, Switzerland*

· Produced prototypes and aided in the design of a flexible multi-electrode array for in vivo rats spinal cord stimulation.

TEACHING EXPERIENCE

Department of Computer Science, UC Irvine Jan 2015 – Jun 2015
Teaching Assistant *Irvine, CA*

- CS-143A: Principles of Operating Systems, Prof. Nalini Venkatasubramanian
- CS-171: Introduction to Artificial Intelligence, Prof. Richard Lathrop
- Conducted discussion, laboratory, and quiz sections that supplement faculty lectures; occasionally presented lectures; graded assignments and examinations.

Department of Computer Science, UC Irvine Sep 2013 – Mar 2014
Reader *Irvine, CA*

- CS-143A: Principles of Operating Systems, Prof. Nalini Venkatasubramanian
- CS-151: Digital Logic Design, Prof. Ian Harris
- Designed and graded homeworks, assignments and examinations; related administrative tasks.

Department of Electrical Engineering, ETH Zurich Sep 2009 – Dec 2010
Tutorial Assistant *Zurich, Switzerland*

- Networks & Circuits I & II, Prof. Johann W. Kolar
- Assisted in lab and discussion sessions; graded weekly homework, papers, and lab reports; held office hours to respond to students questions about such assignments; related administrative tasks.

SKILLS AND QUALIFICATIONS

Programming Languages	C/C++, MATLAB, CUDA, Python, Java, Android, PHP
Programming Environments	MATLAB/Simulink, OpenCV, ROS, LabVIEW
Markup Languages	HTML/CSS, LaTeX
Databases	MySQL
Operating Systems	Unix, Windows
Language Skills	German (native language), English (fluent), French (fair), Spanish (beginner)

NOTABLE OPEN-SOURCE SOFTWARE

OpenCV with Python Blueprints `github.com/mbeyeler/opencv-python-blueprints`
All code for the book "OpenCV with Python Blueprints: Design and develop advanced computer vision projects using OpenCV with Python" (Packt Publishing Ltd., 2015).

CARLsim 3 `socsci.uci.edu/~jkrichma/CARLsim/index.html`
Efficient, easy-to-use, GPU-accelerated software framework for simulating large-scale SNN models with a high degree of biological detail.

Motion Energy `github.com/UCI-CARL/MotionEnergy`
Efficient CUDA implementation of the Motion Energy model (Simoncelli & Heeger, 1998).

Visual Stimulus Toolbox `github.com/UCI-CARL/VisualStimulusToolbox`
MATLAB toolbox for generating, storing, and plotting 2D visual stimuli related to neuroscience such as sinusoidal gratings, plaids, random dot fields, and noise.

TECHNICAL REVIEWER

Books

- F. Dornaika (2015), Advances in Face Image Analysis: Theory and Applications, Bertham Science
- J. Howse (2014), OpenCV for Secret Agents, Packt Publishing Ltd.

Journals

Journal of Neuroscience · PLoS Computational Biology · Neural Networks · IEEE Transactions on Neural Networks and Learning Systems (TNNLS) · ACM Journal on Emerging Technologies in Computing Systems (JETC) · Neurocomputing · PLoS One

Conferences

IEEE International Conference on Intelligent Robots and Systems (IROS) · IEEE International Symposium on Circuits and Systems (ISCAS) · Design, Automation and Test in Europe (DATE)

SERVICE CONTRIBUTIONS

Webmaster, `www.socsci.uci.edu/~jkrichma/CARL`	2013 – 2016
Student volunteer for IEEE Robotics and Automation Society	2014 – 2015
Lab tours for Mathobotix summer camp	2014

PROFESSIONAL ORGANIZATIONS

Society for Neuroscience, student member	2013 – present
IEEE Robotics and Automation Society (RAS), student member	2014 – present

REFERENCES

References available upon request.

ABSTRACT OF THE DISSERTATION

Cortical Neural Network Models of Visual Motion Perception for Decision-Making and Reactive Navigation

By

Michael Beyeler

Doctor of Philosophy in Computer Science

University of California, Irvine, 2016

Professor Jeffrey L. Krichmar, Co-Chair
Professor Nikil D. Dutt, Co-Chair

Animals use vision to traverse novel cluttered environments with apparent ease. Evidence suggests that the mammalian brain integrates visual motion cues across a number of remote but interconnected brain regions that make up a visual motion pathway. Although much is known about the neural circuitry that is concerned with motion perception in the Primary Visual Cortex (V1) and the Middle Temporal area (MT), little is known about how relevant perceptual variables might be represented in higher-order areas of the motion pathway, and how neural activity in these areas might relate to the behavioral dynamics of locomotion. The main goal of this dissertation is to investigate the computational principles that the mammalian brain might be using to organize low-level motion signals into distributed representations of perceptual variables, and how neural activity in the motion pathway might mediate behavior in reactive navigation tasks. I first investigated how the aperture problem, a fundamental conceptual challenge encountered by all low-level motion systems, can be solved in a spiking neural network model of V1 and MT (consisting of $153,216$ neurons and 40 million synapses), relying solely on dynamics and properties gleaned from known electrophysiological and neuroanatomical evidence, and how this neural activity might influence perceptual decision-making. Second, when used with a physical robot performing a reactive

navigation task in the real world, I found that the model produced behavioral trajectories that closely matched human psychophysics data. Essential to the success of these studies were software implementations that could execute in real time, which are freely and openly available to the community. Third, using ideas from the efficient-coding and free-energy principles, I demonstrated that a variety of response properties of neurons in the dorsal subregion of the Medial Superior Temporal area (MSTd) area could be derived from MT-like input features. This finding suggests that response properties such as 3D translation and rotation selectivity, complex motion perception, and heading selectivity might simply be a by-product of MSTd neurons performing dimensionality reduction on their inputs. The hope is that these studies will not only further our understanding of how the brain works, but also lead to novel algorithms and brain-inspired robots capable of outperforming current artificial systems.

Chapter 1

Introduction

1.1 Motivation

Animals are capable of traversing novel cluttered environments with apparent ease. In particular, terrestrial mammals use vision to routinely scan the environment, avoid obstacles, and approach goals. Such sensorimotor tasks require the simultaneous integration of perceptual variables within the visual scene in order to rapidly execute a series of finely tuned motor commands. Evidence suggests that these functions might be mediated by a visual motion pathway in the mammalian brain (Britten, 2008; Orban, 2008); that is, a densely interconnected network of dedicated brain regions, where visual information flows from the visual cortex (including the Primary Visual Cortex (V1), the Middle Temporal area (MT), and the Medial Superior Temporal area (MST); which will be the focus of this thesis) along the dorsal stream onto the Posterior Parietal Cortex (PPC) on to the motor cortex. Understanding how humans and other animals integrate visual motion information to adeptly move around in the environment would require simultaneously recording from all of these areas at a fine temporal and spatial scale—all while the animal is performing a behavioral

task.

Whereas it is possible today to record from either individual cells (e.g., using extracellular single-unit recordings) or from entire brain areas on the order of 10^7 neurons (e.g., using functional Magnetic Resonance Imaging (fMRI) or Electroencephalography (EEG)), it is still difficult to record at the microcircuit ($10^2 - 10^4$ neurons) and mesocircuit ($10^4 - 10^7$ neurons) scales, which are believed to be the scales at which phenomena such as perception, emotion, and understanding begin to take shape (Alivisatos et al., 2012). As a result of these limitations, empirical findings typically provide only limited insight into how activity in microcircuits and mesocircuits might relate to visual motion perception and active behavior under natural conditions. On one hand, although there have been studies on the behavioral dynamics of steering in humans (Fajen and Warren, 2003; Wilkie and Wann, 2003), only recently did researchers begin to ask whether individual behavioral performance is reflected in specific cortical regions (Billington et al., 2013; Field et al., 2007). On the other hand, despite a wealth of data regarding the neural microcircuitry concerned with the perception of self-motion variables such as the current direction of travel ("heading") (Britten and van Wezel, 1998; Duffy and Wurtz, 1997; Gu et al., 2006) or the perceived position and speed of objects (Eifuku and Wurtz, 1998; Tanaka et al., 1993), little research has been devoted to investigating how this circuitry may relate to active steering control.

Meso-scale neural network simulations, on the other hand, have the potential to overcome these limitations. Thanks to recent developments in high-performance computing, it is now possible to simulate Spiking Neural Networks (SNNs) on the order of $10^5 - 10^6$ neurons on a single off-the-shelf Graphics Processing Unit (GPU) in close to real time. By taking into account computational and organizational principles of cortical circuits, gleaned from electrophysiological and neuroanatomical evidence, we can gain insight into the guiding computational principles that make these networks so powerful. In contrast to empirical experiments, computer simulations allow to monitor and manipulate any variable in any

neuron or synapse. This is a powerful approach that could allow neuroscientists to eventually build a mechanistic understanding of meso-scale brain architectures and their functional importance, and computer scientists could be able to extract the underlying computational principles and condense them into abstract algorithms. Second, by integrating these models into autonomous brain-inspired robots, we can study (in real time) the link between neural circuitry and a system's behavior in specific tasks, offering new theories of brain function and stimulating future experimental research.

1.2 Contribution

The main goal of this thesis is to investigate the computational principles that the mammalian brain might be using to accurately and efficiently integrate low-level motion signals into coherent percepts, how inferred perceptual variables might be represented at the population level, and how neural activity in these populations might relate to behavior in visually guided navigation tasks.

This research encompasses the design of computational models that reason about visual motion information in a way that is comparable to the mammalian brain, the development of open-source software libraries that provide the means to study these models, and the conception of robotic systems that can evaluate both theories and simulations in a real-world environment. As a result of these efforts, the contributions of this thesis span multiple disciplines including computer science, computational neuroscience, and robotics.

In specific, the contributions of this thesis include:

- A large-scale SNN model of cortical areas V1 and MT, whose simulated neural activity replicates data recorded *in vivo*, and whose behavioral response emulates human choice accuracy in a motion discrimination task. The model demonstrates how the

aperture problem, a fundamental conceptual challenge encountered by all low-level motion systems, can be solved in a biologically detailed SNN, relying solely on dynamics and properties gleaned from known electrophysiological and neuroanatomical evidence, and how this neural activity might influence perceptual decision-making.

- The deployment of said SNN model on a robotics platform, used to steer the robot around obstacles while approaching a visually salient target, making steering decisions based solely on simulated neural activity generated from visual input. The study demonstrates how a model of MT might build a cortical representation of optic flow in the spiking domain, and shows—for the first time—that these simulated motion signals are sufficient to steer a physical robot on human-like smooth paths around obstacles in the real world.

- A computational model of optic flow processing in the dorsal subregion of the Medial Superior Temporal area (MSTd), demonstrating that a wide range of neuronal response properties of MSTd neurons can be derived from MT-like input features, based on a dimensionality reduction technique known as Nonnegative Matrix Factorization (NMF). This finding suggests that response properties such as 3D translation and rotation selectivity, complex motion perception, and heading selectivity might simply be a by-product of MSTd neurons performing dimensionality reduction on their inputs.

- CARLsim 3, an efficient open-source software environment developed as a group effort in the Cognitive Anteater Robotics Laboratory (CARL) at the University of California, Irvine. This C/C++ software is designed to aid in the conceptualization, utilization, and research of brain circuit models of up to 10^6 neurons and 10^8 synapses by leveraging off-the-shelf NVIDIA GPUs.

- An efficient and scalable open-source software implementation of the Motion Energy (Simoncelli and Heeger, 1998), the *de facto* standard cortical model for visual motion integration. This software outperforms Simoncelli and Heeger's own implementation

by orders of magnitude in terms of computational speed and memory usage, which enabled the practical feasibility of the empirical studies mentioned above.

The hope is that these studies will not only further our understanding of how the brain works, but also lead to novel algorithms and computing systems capable of outperforming current artificial systems.

1.3 Organization

The remainder of this thesis is organized as follows.

Chapter 2 provides some theoretical background on computational vision as well as the visual motion pathway in the mammalian brain. First, general conceptual challenges in visual motion processing are briefly reviewed—such as the correspondence problem and aperture problem—and possible mechanistic solutions for these problems are discussed. Second, a brief overview is given of cortical areas that are involved in visual motion processing in the mammalian brain. Third, different approaches to modeling the brain are discussed.

Chapter 3 introduces CARLsim, the open-source software framework used in Chapters 5 and 6 to study the cortical mechanisms of visual motion integration at the neural-circuit level. The chapter gives an overview of CARLsim's user interface, highlights architectural and algorithmic challenges overcome during development, summarizes its computational performance, and discusses related work.

Chapter 4 introduces an efficient and scalable software implementation of the Motion Energy model by Simoncelli and Heeger (1998). The chapter gives both a conceptual explanation of the model as well as an overview of the software's user interface. The software's computational performance is also analyzed. The development of this software, in combination with

CARLsim, enabled the practical feasibility of the studies described in Chapters 5 and 6.

Chapter 5 describes results from an SNN model of visual motion integration, modeled after cortical areas V1 and MT. The purpose of this model was to understand how the aperture problem could be solved in a biologically detailed SNN, relying solely on dynamics and properties gleaned from known electrophysiological and neuroanatomical evidence. The chapter summarizes results from both simulated neural activity as well as behavioral performance in a motion discrimination task.

Chapter 6 describes the deployment of the SNN model outlined in Chapter 5 on a robotics platform, and summarizes behavioral results of the robot performing a visually guided navigation task. The purpose of this model was to investigate how a simulated cortical representation of optic flow in MT might relate to perceptual decision-making as well as active steering control. The chapter summarizes results from both simulated neural activity as well as behavioral performance in a reactive navigation task.

Chapter 7 introduces a novel computational model of optic flow processing in MSTd, based on principles of the Efficient Coding Hypothesis (ECH). The purpose of this model was to demonstrate that a wide range of neuronal response properties of MSTd neurons could be derived from MT-like input features, suggesting that these empirically observed properties might emerge from a statistically efficient representation of optic flow. The chapter compares simulated neural activity to a variety of empirical findings, and aims to provide some structure to the multiple and often conflicting results from MSTd electrophysiology and modeling studies.

Chapter 8 presents the main conclusions defended in this thesis, discusses limitations of the work, and outlines possible avenues for future studies.

Chapter 2

Background

2.1 Visual Motion Perception

2.1.1 Introduction

Visual motion captures a fundamental aspect of our interactions with the world around us. The first steps in seeing begin in the retina, where visual motion appears as the change of structured patterns of light that can occur for a number of reasons—either because something in the environment moves, or because the observer moves within the environment. Because retinal image motion has multiple possible causes, it is both computationally challenging and richly informative, a phenomenon that was first described and conceptualized by von Helmholtz (1925) and Gibson (1950). Gibson (1950) realized that the apparent motion in our retinas that is caused by the relative movement between an observer and a visual scene (termed "optic flow") could be used to detect one's own movement in the scene, to enable perception of the shape, distance, and movement of objects in the world, and to aid the control of locomotion.

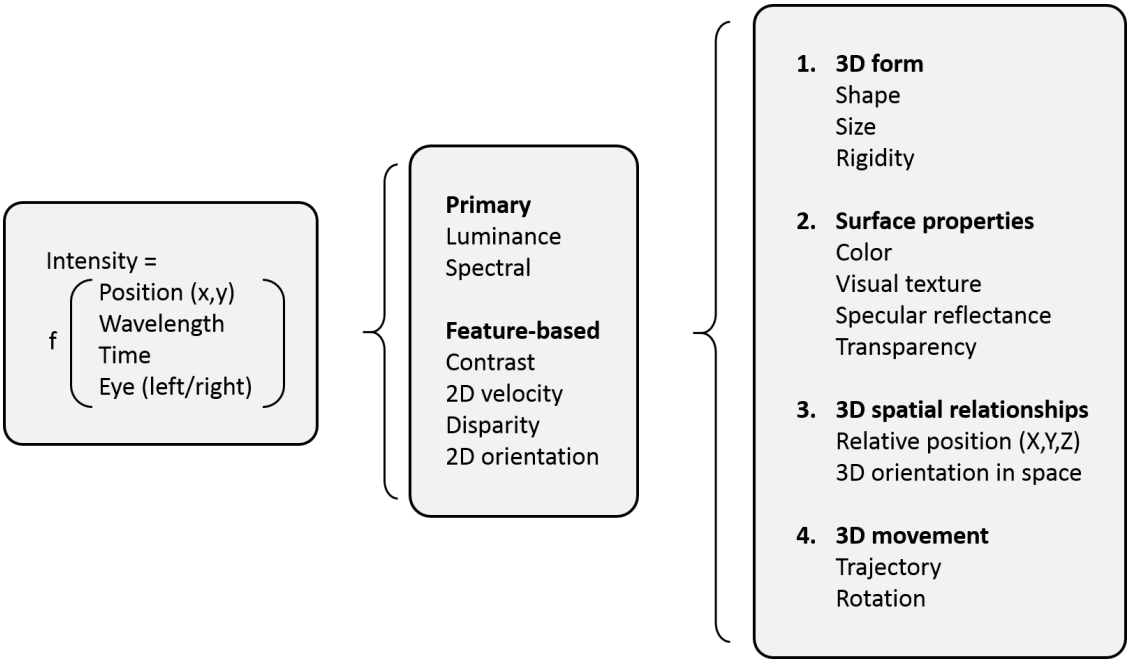

Figure 2.1: From retinal input to cortical processing and perception (adapted from Nassi and Callaway (2009)). Visual input is initially encoded in the retina as a two-dimensional (2D) distribution of light intensity, expressed as a function of position, wavelength, and time in each of the two eyes. This retinal image is transferred to the visual cortex, where sensory cues and (later) inferred attributes are eventually computed.

A general principle in the processing of visual motion by vertebrate brains appears to be the evolution of specialized biological structures capable of efficiently extracting different types of commonly used visual information. For example, when the vertebrate ("camera-like") eye evolved more than 540 million years ago, it already contained photoreceptors, capable of converting incoming patterns of light into an electrochemical signal (Lamb et al., 2007). In the visual system of today's primates, different photoreceptors are specialized for different attributes of light (e.g., color, luminance) and operate as part of a population of cells, where each cell is concerned with only a small part of the visual field ("retinal tiling") while being largely oblivious to light changes that occur outside that small region (Nassi and Callaway, 2009). As a result of photoreceptor arrangement and properties, retinal images are initially expressed as a function of position, wavelength, and time (see Fig. 2.1). Yet much information is lost from the outset, such as the exact spectral composition of the images (Nassi and Callaway, 2009). However, because movement is a feature that distinguishes many items of

great behavioral relevance, biological vision systems must carry out intricate computations that allow an animal to infer more abstract attributes about its surroundings, such as the 3D form, position, and trajectory of potential predators, prey, and mates. In doing so, biological visual systems face fundamental problems associated with low-level motion processing, such as the correspondence problem and the aperture problem, which must be solved in order to provide accurate representations of the visual world around them.

2.1.2 The Correspondence Problem

The correspondence problem (Ullman, 1979) refers to the problem of determining which image elements belong to the same features for two images that are separated in space and/or time (Pack and Born, 2008).

In the brain, this problem might arise in binocular vision, where the images that are projected onto the retinas of the left and right eyes are slightly displaced relative to each other ("horizontal disparity"), and the brain has to match the features in both left and right eyes (i.e., images are separated in space). Alternatively, the problem might arise in visual motion perception, where the images that are projected onto the retina in one eye differ slightly over time (i.e., images are separated in time). The correspondence problem is thus a central issue that the visual system must solve in order to derive 3D information.

How exactly this is achieved in the brain is still an active field of research. For example regarding binocular disparity, evidence suggests that neurons in early visual areas (such as V1 (Cumming and Parker, 1997) or MT (Krug et al., 1999)) cannot discard false binocular matches. But, neurons in the highest-order visual area 7a can (Janssen et al., 2003), indicating that the correspondence problem must have been solved somewhere along the way. A similar picture has emerged for the correspondence problem in motion perception.

The correspondence problem may also result in false matches, where the luminance of a pixel in the second frame happened to correspond to the luminance of a nearby pixel in the first frame (Pack and Born, 2008). Such false matches are inevitable when dealing with inputs that are corrupted by noise, jitter, or local image correlations.

The visual system possesses at least two mechanisms for reducing such noise in natural images: directional opponency and local pooling. Directional opponency is manifested as a suppression of the response to nearby stimuli moving in opposite directions, a mechanism that is very common in early visual areas (Qian and Andersen, 1994; Snowden et al., 1991). Subtracting opposite directions of motion is an efficient way to reduce their contribution (Pack and Born, 2008), since many sources of motion noise have equal motion energy in all directions (see Section 2.1.3.1 and Chapter 4). Another way to reduce motion noise is to pool local measurements over some region of the visual field to produce something like an average of the different local motion vectors (Britten and Heuer, 1999; Lisberger and Ferrera, 1997; Recanzone et al., 1997).

2.1.3 The Aperture Problem

A special case of the correspondence problem is the aperture problem (Marr, 1982), which occurs when a moving pattern is observed through a small window or "aperture" (see Fig. 2.2). Due to the limited field of view, a contour will always appear to be moving perpendicular to the edge's orientation (labeled "visible" in Fig. 2.2), because motion components that are parallel to the edge's orientation (labeled "invisible" in Fig. 2.2) do not contain time-varying information. As a result, incorrect measurements will occur whenever an edge that is more than one pixel in length moves in a direction that is not perpendicular to the orientation of the edge (Pack and Born, 2008).

As it turns out, the aperture problem is prevalent especially in early stages of the primate

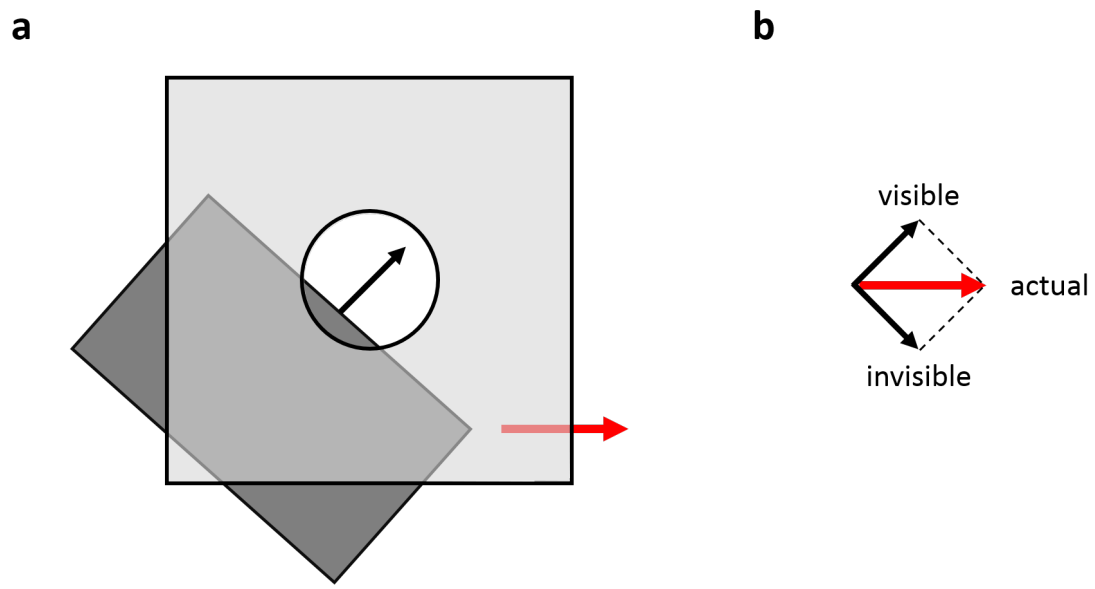

Figure 2.2: The aperture problem (adapted from Bradley and Goyal (2008)). **a**, Even though the rectangle is moving directly to the right (red arrow), it seems to be moving up and to the right when sampled through an aperture (black arrow). **b**, This is because object velocity can be decomposed into two orthogonal vectors, one perpendicular to the visible edge of the edge and one parallel to it (black vectors). The parallel vector is invisible because one cannot detect movement of an edge along its own length; thus, all we detect is the perpendicular vector.

visual cortex, because neurons in these areas have extremely limited fields of view. Whereas most neurons in V1 make systematic mistakes when signaling the direction of motion of a contour, some neurons in MT are able to solve the aperture problem (Movshon et al., 1985; Pack et al., 2001; Rust et al., 2006). How neurons in MT might perform such computations using known biophysical mechanisms will be the topic of Chapter 5.

At a conceptual level, possible approaches to solving the aperture problem can be broadly divided into two categories: integrationist models and selectionist models.

2.1.3.1 Integrationist Models

In the integrationist category of models, which includes what has become the standard model (Heeger et al., 1996; Simoncelli and Heeger, 1998), the aperture problem is solved in two

stages: In a first, relatively unintelligent stage, local 1D motion signals (labeled "visible" components in Fig. 2.2) are extracted, which are then combined nonlinearly, at the second stage, to recover 2D velocity (labeled "actual" in Fig. 2.2). In the model proposed by Simoncelli and Heeger (1998), the first stage is assigned to motion processing in V1, whereas the second stage is assigned to MT.

A neuron that suffers from the aperture problem is referred to as a Component-Direction-Selective (CDS) cell, whereas a neuron that is able to signal the true, actual direction of motion is referred to as a Pattern-Direction-Selective (PDS) cell. This terminology was introduced by Movshon et al. (1985), who noticed that when these cells were presented with a plaid stimulus consisting of two superimposed sine gratings, some neurons preferably responded to the motion of each grating component (CDS cells), whereas others responded to the global motion pattern produced by the combination of the two gratings (PDS cells). Response properties of these two classes of cells will be discussed in detail in Chapter 5.

In the model proposed by Simoncelli and Heeger (1998), each V1 CDS unit is tuned to a relatively narrow range of spatial and temporal frequencies (small Gabor patches in Fig. 2.3a). The aperture problem is solved at the second stage, where MT PDS units collect inputs from many V1 CDS cells of each whose orthogonal, 1D speed measurement is consistent with a given 2D velocity (Fig. 2.3a, c). Conceptually, one can think of each 1D measurement as being consistent with a number of possible 2D velocities, all of which must fall on a line in velocity space (dashed lines in Fig. 2.3b). The intersection of any two of these constraint lines—provided the 1D measurements have different orientations and belong to the same object—provides the solution of the aperture problem. Therefore, this type of model is often referred to as an Intersection-Of-Constraints (IOC) model.

While the velocity-space construction is conceptually useful, in practice the model is implemented in the spatiotemporal-frequency domain (Fig. 2.3c). Here, the Fourier-transformed visual stimulus is multiplied by the frequency-space representations of oriented Gabor fil-

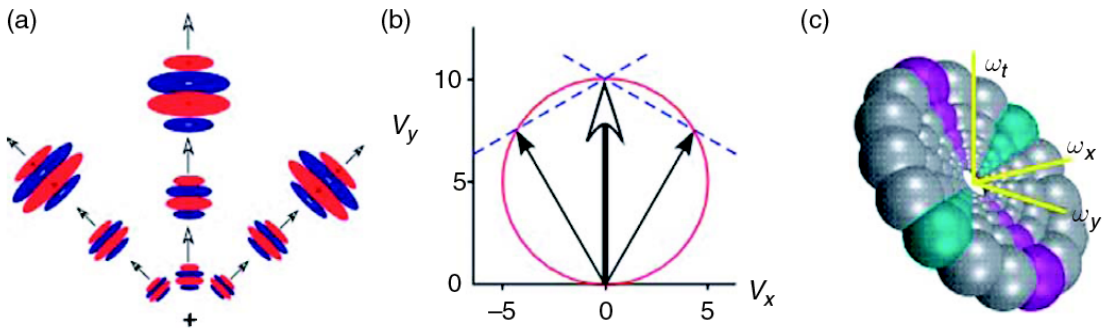

Figure 2.3: Integrationist model of pattern direction selectivity in MT proposed by Simoncelli and Heeger (1998) (reprinted from Pack and Born (2008)). **a**, Partial collection of Primary Visual Cortex (V1) inputs to a pattern cell tuned to a velocity of $10\,^{\circ}\,\mathrm{s}^{-1}$ upwards. Each subunit corresponds to the Receptive Field (RF) of a V1 complex cell. **b**, Velocity-space representation of the Intersection-Of-Constraints (IOC) calculation. Any 1D velocity measurement (solid arrows) is consistent with a range of 2D velocities falling along a constraint line (dashed lines) perpendicular to its motion vector. For two such measurements made at different orientations, the 2D velocity is given by the point of intersection of the two constraint lines. Conversely, all 1D velocities consistent with a given 2D velocity (hollow arrow) fall on a circle in velocity space (Pack and Born, 2008). **c**, The frequency-space representation of the model depicted in **a**.

ters whose output is half-squared (i.e., the negative values are clipped off and the result is squared) and normalized to produce a measurement of motion energy (Adelson and Bergen, 1985) orthogonal to the orientation of the Gabor. When plotted in the 3D space comprised of two dimensions of spatial frequency (ω_x and ω_y) and one of temporal frequency (ω_t), the selectivity of a given model V1 neuron appears as a pair of localized blobs positioned symmetrically about the origin, and the different spatial scales of the filters fall along a line passing through the origin (Adelson and Bergen, 1985; Heeger et al., 1996; Simoncelli and Heeger, 1998). In this frequency space the locus of points corresponding to a unique 2D velocity describes a plane, so a given MT cell sums up all of the spatiotemporal blobs within the plane consistent with its particular preferred direction and speed (see Fig. 2.3c).

This model has been highly successful in explaining not only the original data that motivated it (Movshon et al., 1985; Rodman and Albright, 1987, 1989), but a number of other known MT response properties, such as responses to random dots embedded in noise (Britten et al.,

1992; Newsome et al., 1989), making it one of the most supported working models of cortical motion processing. The development of an efficient, scalable implementation of this model will be the topic of Chapter 4.

2.1.3.2 Selectionist Models

In contrast, a second category of models places the emphasis on carefully selecting local motion features that do not (or only weakly) suffer from the aperture problem, and are thus most indicative of the actual, global direction of motion. Simply speaking, these features are local regions of the image where multiple orientations are present (Pack and Born, 2008). Once the brain has identified the most reliable local motion signals, all it has to do is to track them over time in order to compute the true 2D velocity.

Here, the comparison between orientations happens in a first stage, by way of known mechanisms of the visual cortex such as end-stopping that suppress 1D motion signals in favor of motion signals emanating from 2D features. The second stage then simply averages the outputs from the first stage to compute the final 2D velocity.

2.2 Visual Motion Processing in the Mammalian Brain

There are at least 32 separate neocortical areas in the mammalian brain that are implicated in visual processing (Felleman and van Essen, 1991). However, not all of these areas are exclusively visual in function. Nonvisual contributions include inputs from other sensory modalities (such as auditory and somatosensory), visuomotor activity (i.e., related to eye movements), and attentional influences (Felleman and van Essen, 1991).

It has long been hypothesized that there are at least two distinct major pathways (or

"streams") in the primate brain that process vision (Goodale and Milner, 1992; Ungerleider and Mishkin, 1982): the ventral stream (also known as the "what" pathway) is involved with object recognition, whereas the dorsal stream (also known as the "where" pathway) is involved with processing the object's spatial location relative to the viewer. The visual motion pathway is thought to be part of the dorsal stream (Britten, 2008; Orban, 2008), characterized by a preponderance of neurons selective for the speed and direction of motion (Fig. 2.4).

Each neuron in the early stages of the primate visual system responds to stimulation over a very small part of the visual field. In visual neurophysiology, this limited field of view is called the Receptive Field (RF), and most RFs of the primary visual cortex are smaller than the width of one's thumbnail held at arm's length. It is useful to think of RFs as pinholes (or apertures) through which neurons view the outside world. Their job is to measure certain aspects of the visual world that occur within the pinhole, and they are largely oblivious to events that occur outside this small region. Within any part of the visual cortex, neurons are found whose RFs collectively tile the whole of visual space. In addition to having a spatially delimited RF, each neuron has a limited range of visual stimuli to which it will respond. Some neurons are tuned to the shape of the stimulus, some to the color, and others are tuned to the motion that occurs within their RF. However, even in the early stages of visual processing, there is much lateral and top-down influences on these RFs.

Early and middle areas in the visual motion pathway, such as V1 and MT (yellow and orange tones in Fig. 2.4) have small- to medium-sized RFs and relatively simple, linear directional preferences. Upper visual areas, most notably Medial Superior Temporal area (MST) and Ventral Intraparietal area (VIP), respond to complex motion in relatively large RFs, which makes them well-suited to process visual self-motion. In addition, some of these areas also respond to vestibular, somatosensory, and auditory stimuli. Area 7a is thought to be the highest-order parietal area, most important for spatial perception, which projects to

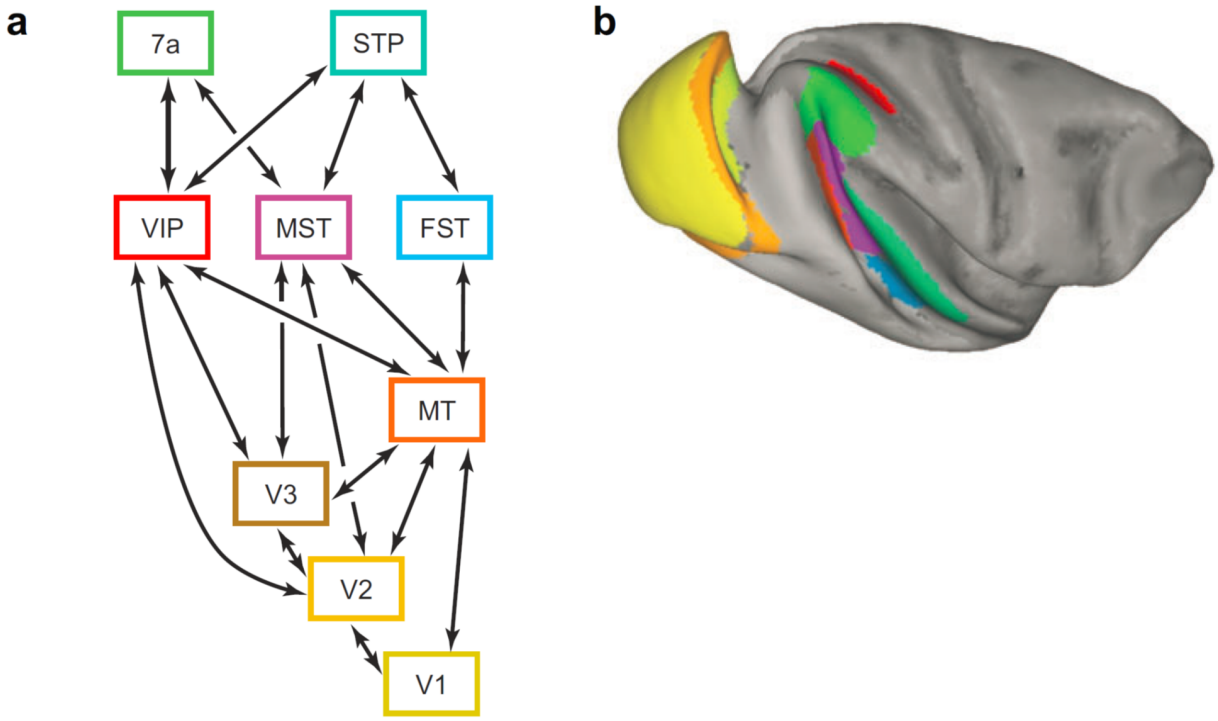

Figure 2.4: Anatomy of the visual motion pathway in the macaque brain (reprinted from (Britten, 2008)). **a**, Simplified schematic of the connections between areas known to play a role in motion analysis. **b**, Anatomical locations of the areas on a slightly "inflated" monkey brain to allow visualization of areas within sulci. The viewpoint is dorsal and lateral. Nomenclature and area boundaries after Felleman and van Essen (1991); image generated with public domain software CARET (`http://brainmap.wustl.edu/caret`; van Essen et al. (2001)).

hippocampus, entorhinal cortex, and various subcortical structures.

In this work, we focus on the visual response properties of motion-selective neurons in areas V1, MT, and MST.

2.2.1 Retina

The first steps in seeing begin in the retina, where a dense array of photoreceptors convert the incoming pattern of light into an electrochemical signal (Nassi and Callaway, 2009). As mentioned above, the photoreceptor mosaic encodes the intensity of light as a function

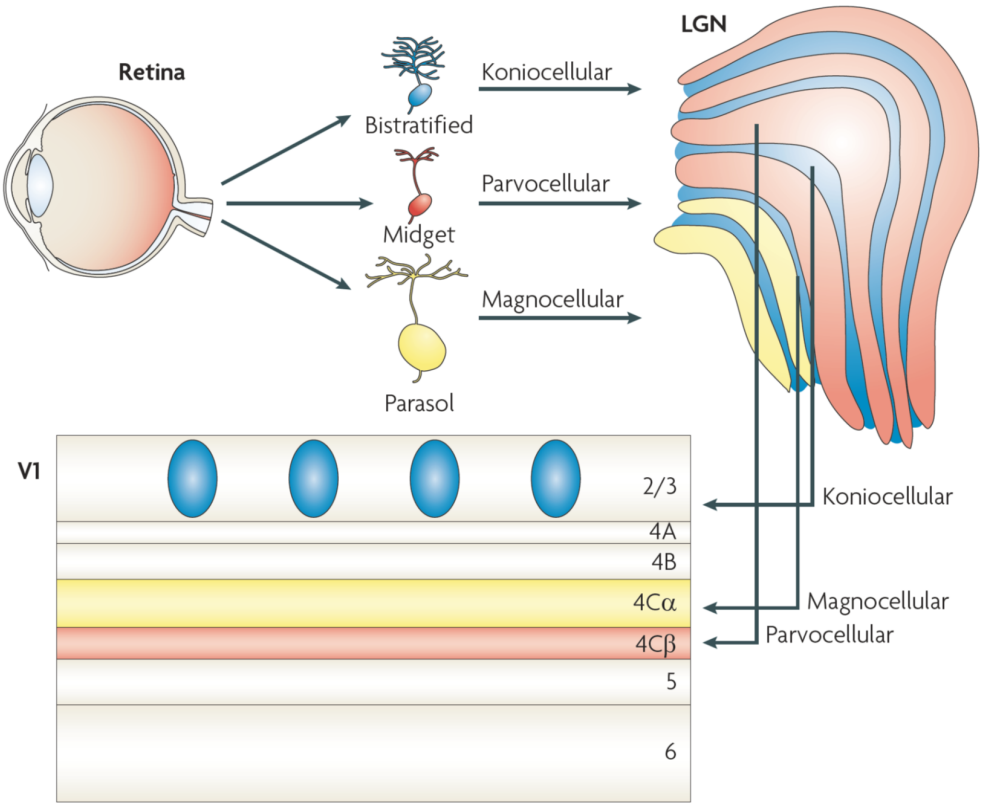

Figure 2.5: Parallel pathways from the retina to the cortex (reprinted from Nassi and Callaway (2009)). Midget, parasol and bistratified ganglion cells are well characterized and have been linked to parallel pathways that remain anatomically separate through the Lateral Geniculate Nucleus (LGN) and the Primary Visual Cortex (V1).

of retinal position (two dimensions), wavelength, and time (see Fig. 2.1). Owing to the anatomical bottleneck of the optic nerve, retinal output must be efficiently condensed. The strategy of the mammalian visual system is to reduce the representation of the visual scene to a limited number of specialized, parallel output channels. Rather than sending visual signals from the eye to the brain along a homogeneous population of ganglion cells, in the primate at least 17 distinct ganglion cell types exist in the retina, and at least 13 of these project in parallel to the LGN (Dacey, 2004). Each ganglion cell is thought to tile the retina, providing a complete representation across the entire visual field (Field and Chichilnisky, 2007).

Midget, parasol, and bistratified ganglion cells (see Fig. 2.5) lie at the origin of three distinct,

functionally complementary pathways that process red-green color opponency (parvocellular pathway), motion (magnocellular pathway), and blue-yellow color opponency (koniocellular pathway), respectively (Nassi and Callaway, 2009). Together they constitute approximately 90 % of all ganglion cells found in the primate retina (Dacey, 2000, 2004).

The most important type of retinal cells for motion perception are parasol ganglion cells, which convey a broadband, achromatic signal to the magnocellular layers of the LGN. Cells in this pathway typically have large RFs, high contrast sensitivity, fast axonal conduction velocities, and sensitivity to high temporal and low spatial frequencies (Dacey, 2000; Nassi and Callaway, 2009).

Midget ganglion cells, on the other hand, convey a red-green color-opponent signal to the parvocellular layers of the LGN. Cells in this pathway typically have small RFs, low contrast sensitivity, slow axonal conduction velocities, and sensitivity to high spatial and low temporal frequencies (Nassi and Callaway, 2009).

2.2.2 Lateral Geniculate Nucleus (LGN)

The Lateral Geniculate Nucleus (LGN) is a relay center in the thalamus for the visual pathway and the main central connection for the optic nerve to the occipital lobe. In mammals, LGN has six layers of gray matter interleaved with layers of white matter (see Fig. 2.5), each layer of neurons having its own characteristic response properties and projection targets in the cortex.

This means that at the level of LGN, magnocellular, parvocellular, and koniocellular streams remain anatomically segregated. Here, signals about visual motion are relayed from parasol retinal ganglion cells to layers 4Cα and 6 of V1 (Blasdel and Lund, 1983; Chatterjee and Callaway, 2003), whereas signals about color are relayed from midget retinal ganglion cells

on to layers 4Cβ and 6 of V1 (Blasdel and Lund, 1983; Chatterjee and Callaway, 2003; Dacey, 2000).

2.2.3 Primary visual cortex (V1)

Once the condensed and parallel signals from the retina and LGN arrive in the Primary Visual Cortex (V1), the original components of the visual scene must be extracted, elaborated on and integrated into a unified percept. The visual cortex uses both hierarchical and modular processing to accomplish these goals (Zeki and Shipp, 1988).

Early studies suggested there is a clean anatomical segregation of magnocellular and parvocellular inputs to layers 4B and 2/3 of V1, respectively, as well as a functional segregation of fast, achromatic responses in layer 4B and color-opponent or orientation-selective responses in blobs and interblobs, respectively (Hawken et al., 1988; Livingstone and Hubel, 1984). More recent studies, however, provided evidence for extensive mixing and convergence of magnocellular, parvocellular and koniocellular pathway inputs, suggesting that V1 outputs bear little or no systematic relationship to its parallel inputs (Sincich and Horton, 2004).

Specialized and distinct populations of cells project from layer 4B of V1 to MT or V2. MT receives input from a population of cells with large cell bodies and dense dendritic trees in layer 4B of V1, which contain a quick, magnocellular-dominated signal (Movshon and Newsome, 1996). These cells are highly functionally specialized and distinct from the general population in V1 (an example cell is shown in Fig. 2.6). Most of these cells (80 %) are stellates, but a smaller number of pyramids (20 % of the cells) also project to MT and are positioned such that their apical dendrites can receive magnocellular inputs from layer 4Cα (Movshon and Newsome, 1996; Nassi and Callaway, 2007).

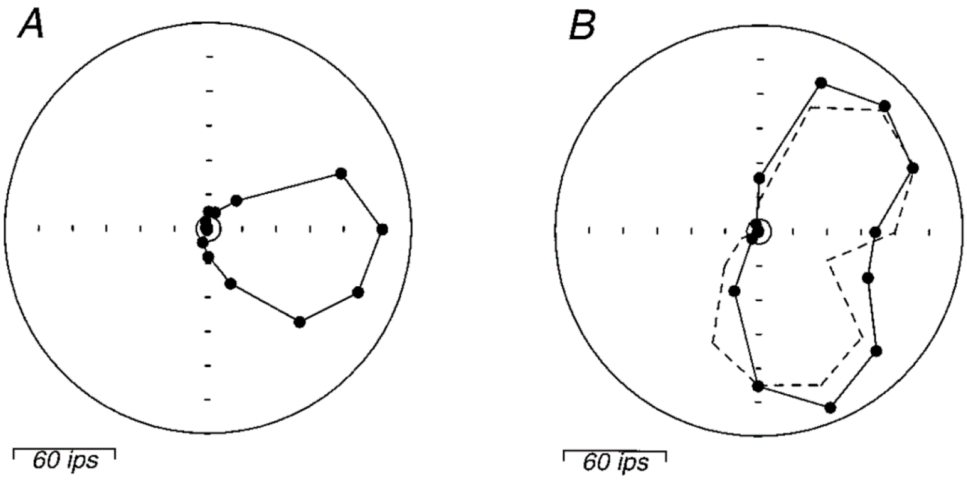

Figure 2.6: Example of a directionally selective neuron in V1 that is projecting to MT (reprinted from Movshon and Newsome (1996)). Polar plots show the responses of the neuron to a drifting sine grating (**A**) and drifting plaids (**B**). Stimuli drifted in one of 16 directions of motion separated by equal intervals of 22.5°. The plaid stimuli were created by superimposing two sine wave gratings of equal contrast and spatial and temporal frequency, whose orientations differed by 135°. The direction of plaid motion is the direction of coherent motion perceived by human observers. The solid lines and data illustrate the actual responses of the neuron; the dashed lines depict the predicted tuning curve if the neuron responded only to the motions of the two component gratings. The small circles at the center of each plot show the spontaneous firing rates.

2.2.4 Middle Temporal (MT) area

The Middle Temporal area (MT) is a region of the extrastriate visual cortex that is famous for its role in visual motion processing. MT is retinotopically organized, each hemisphere containing a more-or-less complete map of the contralateral visual hemi-field, with a marked emphasis on the fovea (the central 15° of the visual field occupies over half of MT's surface area (van Essen et al., 1981)) and a bias toward the lower quadrant of the visual field (Maunsell and van Essen, 1987).

There are multiple input streams from the LGN to MT (see Fig. 2.7). The major ascending input to MT passes through magnocellular layers of the LGN (yellow) and through layers 4Cα and 4B of V1. V2 and V3 provide indirect inputs from layers 4Cα and 4B of V1, with V2 probably providing inputs from parvocellular layers of the LGN (red) and layer 4Cβ

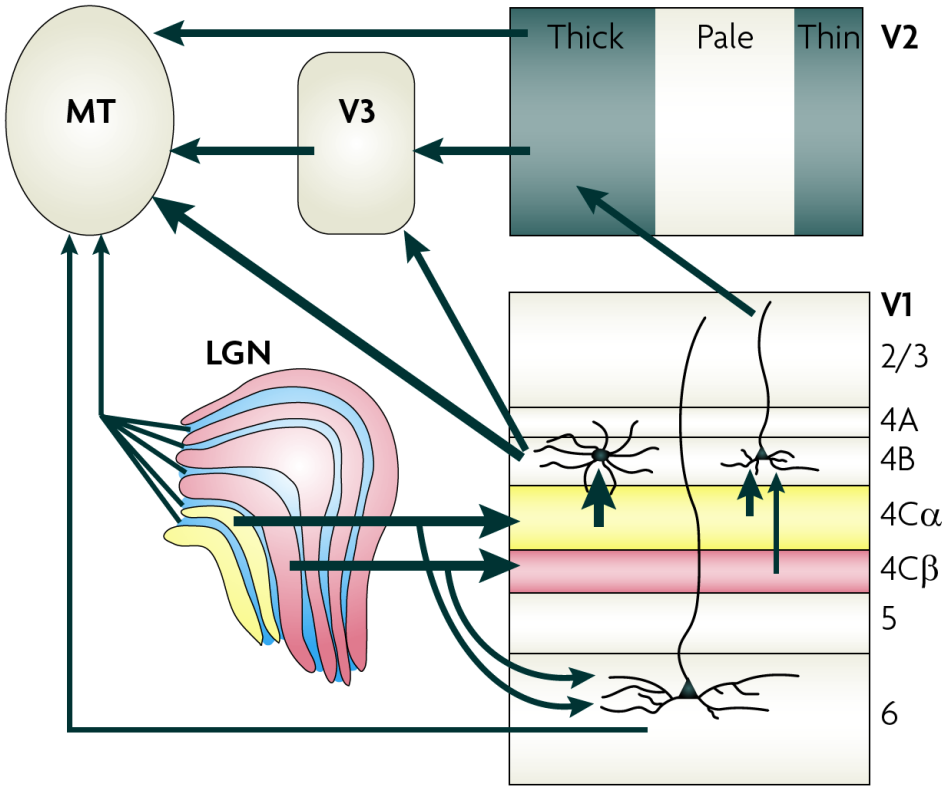

Figure 2.7: Multiple input streams to MT (reprinted from Nassi and Callaway (2009)). The major ascending input to MT passes through magnocellular layers of the LGN (yellow) and through layers 4Cα and 4B of V1, which is the focus of this dissertation. Pathways ascending through V2 and V3 likely provide visual cues about binocular disparity. The thickness of the each arrow represents the approximate strength of the connection.

after a small number of additional synapses (Nassi and Callaway, 2009). Bypassing layer 4C altogether, a sparse monosynaptic projection from koniocellular layers of the LGN (blue) to MT and a disynaptic projection from magnocellular and parvocellular layers of the LGN through layer 6 Meynert cells in V1 to MT have both been identified (Nassi and Callaway, 2009). MT is likely to use similar strategies to those found in V1 to process these parallel inputs and transform their signals into multiple output streams. The thickness of each arrow in Fig. 2.7 represents the approximate strength of the connection. In this work, we focus mainly on the pathway that travels from LGN through V1 to MT.

The visual responses of MT neurons are determined principally by five properties of the stim-

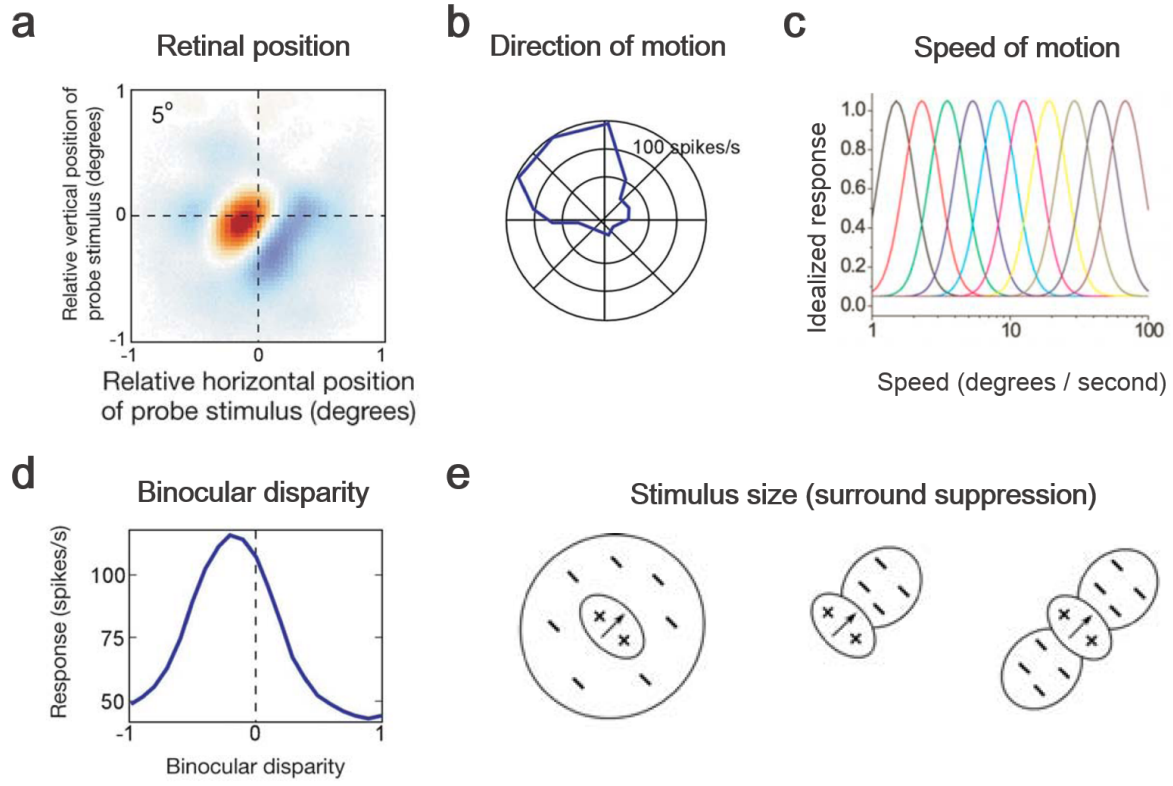

Figure 2.8: MT response properties. **a**, Retinal positon. **b**, Direction of motion. **c**, Speed of motion. **d**, Binocular disparity. **e**, Stimulus size (due to surround suppression). Panel **c** reprinted from Nover et al. (2005), all others reprinted from Born and Bradley (2005).

ulus (see Fig. 2.8): retinal positon, direction of motion, speed of motion, binocular disparity, and stimulus size (due to surround suppression) (Born and Bradley, 2005). Speed tuning might be approximately log-Gaussian and scale-invariant (Nover et al., 2005), whereas direction tuning is broader (Duijnhouwer et al., 2013). Organization for direction and disparity tuning is columnar, but organization for speed tuning is not (only weakly clustered).

MT RFs are up to ten times larger than RFs in V1 (Raiguel et al., 1995). In some neurons, stimulus selectivity varies drastically in different subfields of the RF (Richert et al., 2013). About half of the neurons in MT have receptive field with antagonistic surrounds (Allman et al., 1985; Raiguel et al., 1995), and these surrounds can come in one of several shapes (see Fig. 2.8e). In general, the surround effects are such that maximal suppresion occurs

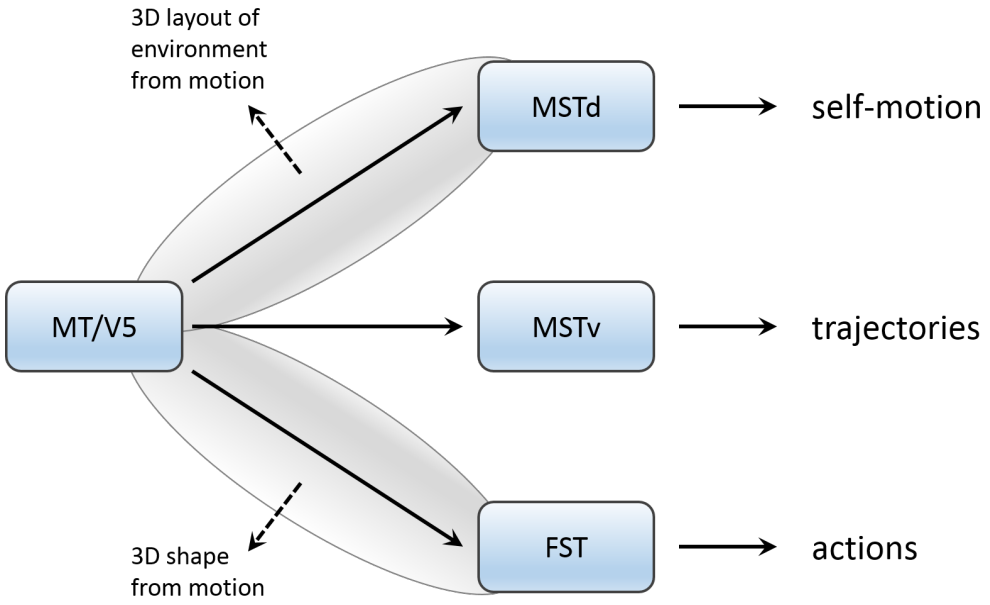

Figure 2.9: Different types of motion signals processed by MT and its satellites (adapted from Orban (2008)).

when the surround stimulus moves in the same direction and at the same disparity as that in the center (Allman et al., 1985). As such, the center-surround apparatus can act as a differentiator over direction and depth, allowing some MT neurons to sense motion gradients and act as a motion salience detector (Born and Bradley, 2005).

The signals from MT are sent in parallel to a number of neighboring regions (see Fig. 2.9, including the dorsal subregion of the Medial Superior Temporal area (MSTd) for processing of self-motion, the ventral subregion of the Medial Superior Temporal area (MSTv) for processing of trajectories of moving objects as well as generating smooth pursuit eye movements, and Floor of the Superior Temporal visual area (FST) for processing of actions and motions of animate entities (Orban, 2008). Interestingly, MT neurons with antagonistic surrounds mostly project to the ventral subregion of the Medial Superior Temporal area (MSTv) and Floor of the Superior Temporal visual area (FST), but not to MSTd (Berezovskii and Born, 2000).

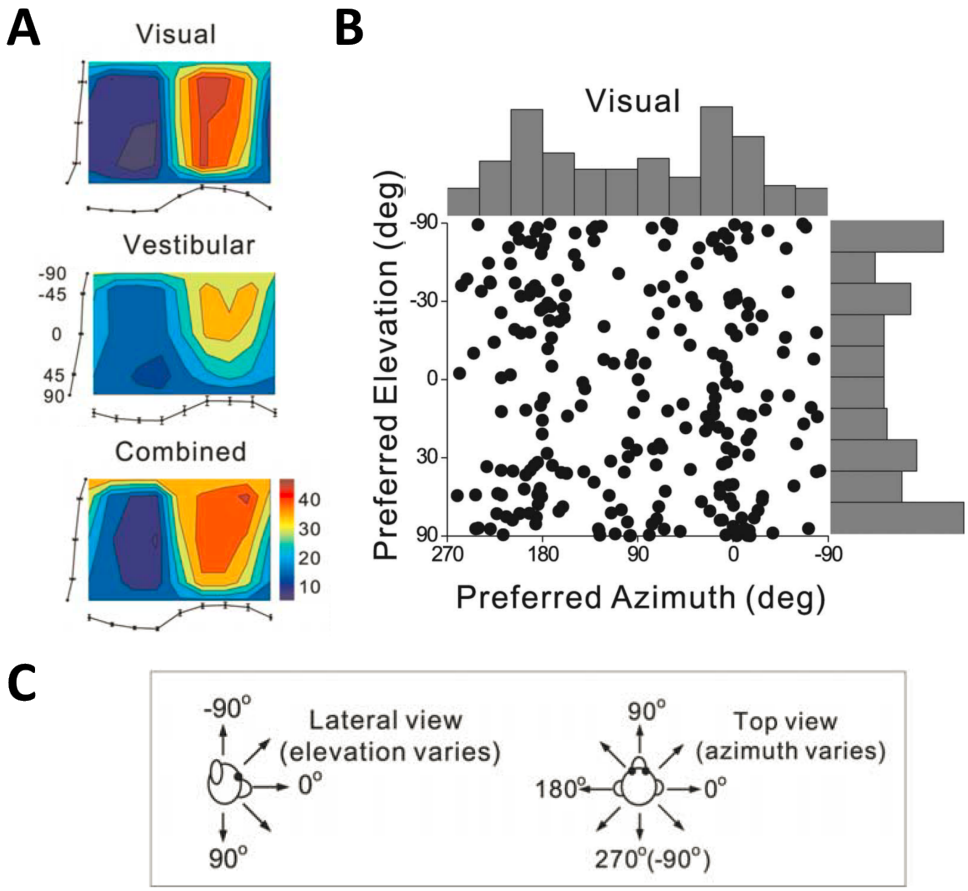

Figure 2.10: 3D heading tuning in MSTd (reprinted from Gu et al. (2006)). **A**, Data from a neuron with congruent tuning for heading defined by visual and vestibular cues. Color contour maps show the mean firing rate as a function of azimuth and elevation angles. Each contour map shows the Lambert cylindrical equal-area projection of the original spherical data (Snyder, 1987). In this projection, the ordinate is a sinusoidally transformed version of elevation angle. Tuning curves along the margins of each color map illustrate mean SEM firing rates plotted as a function of either elevation or azimuth (averaged across azimuth or elevation, respectively). **B**, Distributions of 3D heading preferences of MSTd neurons. Each data point in the scatter plot corresponds to the preferred azimuth (abscissa) and elevation (ordinate) of a single neuron with significant heading tuning. **C**, Definitions of azimuth and elevation angles used to define heading stimuli in 3D.

2.2.5 Middle Superior Temporal (MST) area

The MST receives its main input from neurons in MT (Ungerleider and Desimone, 1986). Whereas neurons in MSTv are involved in the analysis of visual trajectories of relatively small objects and the generation of smooth pursuit eye movements, neurons in MSTd respond to relatively large and complex patterns of retinal flow, implying a role in the analysis of self-motion (Duffy and Wurtz, 1991a,b; Gu et al., 2010, 2012; Saito et al., 1986; Tanaka and Saito, 1989).

The visual processing of MSTd neurons has been summarized as a template matching with radial motion, rotation, and translation, or their combinations (Orban, 2008). Because expansion/contraction and rotation are basically spatial configurations of local translations of the observer, MSTd has been implicated with the analysis of optic flow during self-movement; that is, with the analysis of apparent motion in a visual scene caused by the relative movement of the observer. Indeed, recording from MSTd in a macaque while the animal is physically moving along trajectories in 3D space revealed that nearly all neurons in MSTd were selective to at least one of these directions of travel (or "headings") (Gu et al., 2006; Takahashi et al., 2007). In contrast to earlier studies (Duffy and Wurtz, 1991a,b; Graziano et al., 1994), all directions of heading in space seem to be (roughly) equally represented in MSTd (see Fig. 2.10): there is no preference for straight-ahead trajectories or trajectories along the ground plane.

However, neurons in MSTd are not just selective for heading, but often prefer a mixture of translational, rotational, and to a lesser degree deformational flow components (Duffy and Wurtz, 1991a,b; Graziano et al., 1994; Mineault et al., 2012), with receptive fields that can encompass up to $40^\circ \times 40^\circ$ of the visual field (Raiguel et al., 1997).

Besides pure retinal signals, neurons in MSTd are also driven by vestibular (Gu et al., 2006, 2010; Page and Duffy, 2003; Takahashi et al., 2007) and eye movement-related signals

(Komatsu and Wurtz, 1988; Newsome et al., 1988). Hence, MSTd could contain a number of visual motion-related variables (Hamed et al., 2003), ranging from pure retinal to head-centered stimulus velocity coding (Brostek et al., 2014; Chukoskie and Movshon, 2009; Yu et al., 2010) that include intermediate reference frames (Fetsch et al., 2007). Finally, some neurons are selective not just for heading, but also for path and place (Froehler and Duffy, 2002; Page et al., 2015).

2.2.6 Visual Motion Processing Beyond MST

The dorsal pathway consists of a large number of interconnected extrastriate cortical areas in the parietal cortex downstream of MT, including MST, FST, the Superior Temporal Polysensory (STP) area, Ventral Intraparietal area (VIP), Lateral Intraparietal area (LIP), and visual area 7a, to name just a few (Andersen et al., 1990; Boussaoud et al., 1990; Lewis and van Essen, 2000; Maunsell and van Essen, 1983). Motion processing gets increasingly complex in these areas, involving not just visual but also vestibular, somatosensory, and auditory signals. Collectively, these higher-order parietal areas might be involved in all things related to spatial planning, including the coordination and planning of locomotion, eye movements (i.e., saccades and smooth pursuit), head movements, and arm movements (i.e., reaching). Evidence indicates that at this stage, the dorsal stream might actually branch out into two relatively segregated circuits (Rizzolatti and Matelli, 2003), but motion processing in these areas remains an active field of research to this day.

2.3 Approaches to Modeling the Brain

Because of the brain's sheer complexity, it is important to choose an appropriate abstraction level when designing a model of brain function (see Fig. 2.11). Each level of abstraction

serves a different purpose, from theoretical models designed to understand the statistics of sensory data and expressing cortical function as high-level mathematical principles ("systems neuroscience") to cellular models that aim to include as much biological detail as possible, ranging from molecular, electrical, to morphological properties of neurons and synapses. Of course, although the latter type of models are biologically accurate, they incur tremendous computational costs for simulation, limiting the size and complexity of the network that can be simulated.

Neural circuit models constitute a compromise between accuracy and simulation complexity by abstracting away many molecular and cellular details of a neuron. This type of models considers the brain as a massive circuit composed of four basic components: neurons for computation, synapses for learning and memory storage, axons for communication, and neuromodulatory systems for modeling state (Abbott and Nelson, 2000; Brette et al., 2007; Furber and Temple, 2007). Models of this type are well-suited to study neural circuits at the micro scale ($10^2 - 10^4$ neurons) and meso scale ($10^4 - 10^7$ neurons), as they are able to exhibit a variety of empirically observed network dynamics such as random asynchronous firing, oscillatory (synchronous) firing, sustained responses to transient stimuli, and chaotic network activity. Such networks are complex dynamical systems involving the numerical integration of many thousands of coupled differential equations (Vogels and Abbott, 2005).

In order to make the simulation of such networks feasible, a common compromise is to use a relatively simple neuron model to model each node in the network, using either firing-rate or spiking neurons. In biological neurons, brief electrical pulses (called action potentials or spikes) travel along a wire (axon) to the connection site (synapse) of another neuron, where they cause a small change in the electric membrane potential of the receiving neuron (Kandel et al., 2000). Neurons integrate these small changes until their voltage reaches a threshold, at which point the neuron is said to fire a spike. Firing-rate neurons describe the activity of each neuron as the sum of spikes over a given time interval (i.e., firing rate), whereas spiking

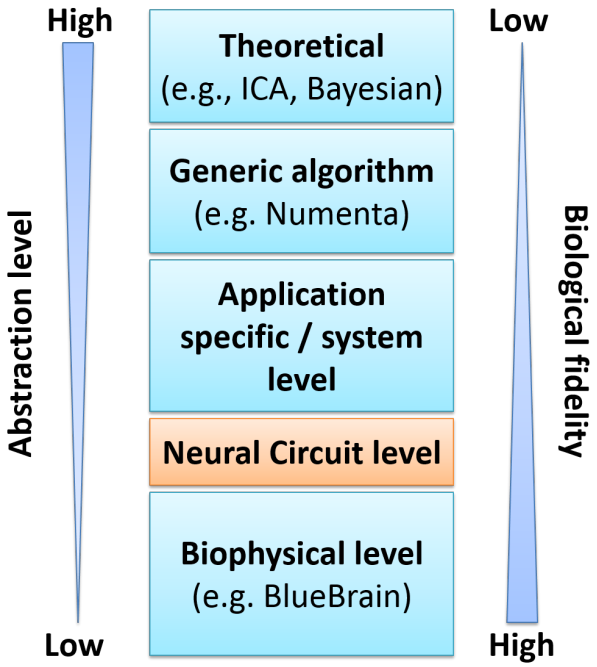

Figure 2.11: Possible levels of modeling abstraction (adapted from Nageswaran et al. (2010)).

neurons model the actual temporal evolution of each neuron's membrane potential.

2.3.1 Firing-Rate Networks

In a firing-rate network, each neuron j is described at time t by a firing rate $r_j(t)$ that relaxes with a time constant τ_r to a steady-state value given by a function F that describes the relationship between firing rate and input current $i(t)$ for the neuron (Vogels and Abbott, 2005):

$$\tau_r \frac{dr_j}{dt} = -r_j(t) + F(i(t)). \tag{2.1}$$

Here, $i(t)$ is the sum of the input from sources outside the network such as sensory input, and the sum of inputs from other neurons within the network. The assumption behind Eq. 2.1 is that, on average, the input from a given presynaptic neuron is proportional to its firing rate, which is only true for some neuron types (refer to Section 2.3.2.1).

2.3.2 Spiking Neural Networks (SNNs)

An alternative is to use a model of spiking neurons to capture (to a varying degree) properties of the neuronal and synaptic state. In contrast to firing-rate models, which describe neuronal activity as the average firing rate of a neuron as a function of its inputs, Spiking Neural Network (SNN) models use an explicit representation of spike timing (Brette et al., 2007; Maass and Bishop, 2001), leading to inherently event-driven communication protocols.

One of the most widely used spiking models in computational neuroscience is the Leaky Integrate-and-Fire (LIF) neuron:

$$\frac{dv}{dt} = I + a - bv, \tag{2.2}$$

$$v(v > v_{\text{thresh}}) = c \tag{2.3}$$

where v is the membrane potential, I is the input current, and a, b, c, and v_{thresh} are the parameters. When the membrane potential v reaches the threshold value v_{thresh}, the neuron is said to fire a spike, and v is reset to c. Due to its simplicity, the LIF is computationally efficient. It can fire tonic spikes with constant frequency, but it cannot exhibit some of the more sophisticated dynamics observed in biological neurons, such as phasic spiking, bursting, rebound responses, threshold variability, bistability of attractors, or autonomous chaotic dynamics (Izhikevich, 2004).

Because other spiking neuron models exist that are more biologically plausible, it has been argued that these models might be better suited than LIF neurons to study how the mammalian neocortex processes spatiotemporal information (Izhikevich, 2004). One of the most important models in computational neuroscience is the Hodgkin-Huxley neuron model (Hodgkin and Huxley, 1954), which consists of four equations and tens of parameters, accurately describing single-cell dynamics with respect to synaptic integration, dendritic cable filtering, and effects of dendritic morphology. However, it is extremely computationally expensive to

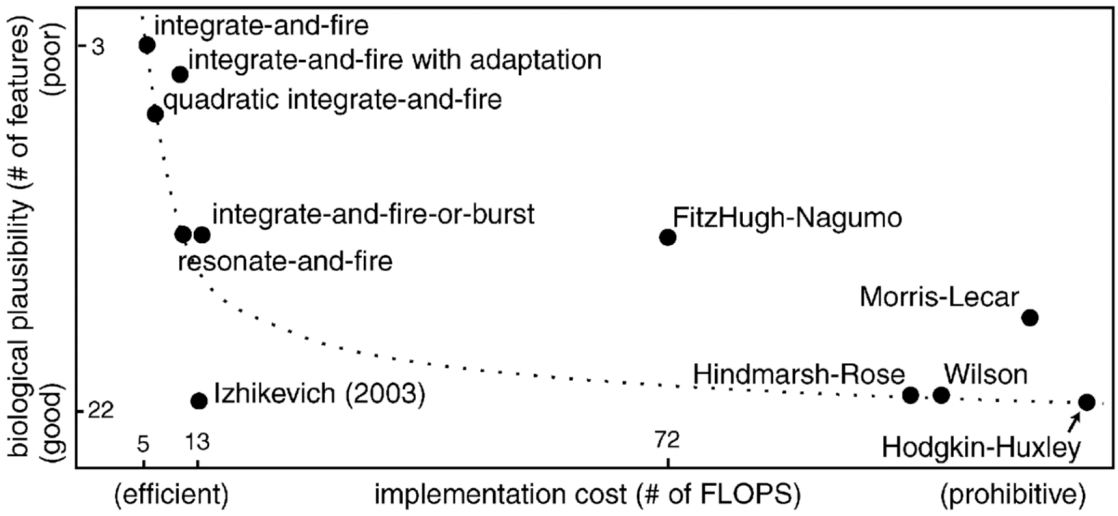

Figure 2.12: Comparison of different neuro-computational properties of spiking and bursting models (Izhikevich, 2004). "# of FLOPS" is an approximate number of floating point operations needed to simulate the model during a 1 ms time span.

implement.

An interesting compromise between the computational efficiency of the LIF model and the biological fidelity of the Hodgkin-Huxley model is the Izhikevich spiking model (Izhikevich, 2003) (see Fig. 2.12). Using only two coupled Ordinary Differential Equations (ODEs), the model is computationally efficient yet exhibits complex neuronal dynamics that closely mimic biological neurons.

2.3.2.1 Izhikevich Spiking Neurons

The Izhikevich model aims to reduce Hodgkin-Huxley-type neuronal models to a two-dimensional system of ODEs:

$$\frac{dv(t)}{dt} = 0.04v^2 + 5v(t) + 140 - u(t) + i_{\text{syn}}(t) \tag{2.4}$$

$$\frac{du(t)}{dt} = a\big(bv(t) - u(t)\big) \tag{2.5}$$

$$v(v > v_{\text{cutoff}}) = c \text{ and } u(v > v_{\text{cutoff}}) = u - d. \tag{2.6}$$

Here, Eq. 2.4 describes the membrane potential v for a given synaptic current i_{syn}, whereas Eq. 2.5 describes a recovery variable u; the parameter a is the rate constant of the recovery variable, and b describes the sensitivity of the recovery variable to the subthreshold fluctuations of the membrane potential. Whenever v reaches peak ($v_{\text{cutoff}} = +30$), the membrane potential is reset to c and the value of the recovery variable is decreased by d (see Eq. 2.6). The inclusion of u in the model allows for the simulation of typical spike patterns observed in biological neurons. The four parameters a, b, c, and d can be set to simulate different types of neurons (see Fig. 2.13). The most common type of excitatory neuron in mammalian neocortex, the Regular Spiking (RS) cell, fires tonic spikes with decreasing frequency, as in Fig. 2.13a (class 1 excitability). Other neurons—such as some local inhibitory interneurons—cannot fire low-frequency spike trains, thus are either quiescent or fire a train at relatively large frequency, as in Fig. 2.13h.

2.3.2.2 Synapses

In this type of neural-circuit models, ionic currents are typically modeled as dynamic synaptic channels with zero rise time and exponential decay:

$$\frac{dg_r(t)}{dt} = -\frac{1}{\tau_r} g_r(t) + \eta_r w \sum_i \delta(t - t_i), \tag{2.7}$$

where g_r is the synaptic conductance, η_r is a receptor-specific efficacy (or "synaptic gain"), w is the weight of the synapse, δ is the Dirac delta, the sum is over all presynaptic spikes arriving at time t_i, and the subscript r denotes the receptor type. The most prominent receptor types in the mammalian brain include AMPA (fast decay, $\tau_{\text{AMPA}} = 5\,\text{ms}$), NMDA (slow decay and voltage-dependent, $\tau_{\text{NMDA}} = 150\,\text{ms}$), GABAa (fast decay, $\tau_{\text{GABAa}} = 6\,\text{ms}$), and GABAb (slow decay, $\tau_{\text{GABAb}} = 150\,\text{ms}$) (Izhikevich et al., 2004). Typically, a spike arriving at a synapse that is post-synaptically connected to an excitatory (inhibitory) neuron increases both g_{AMPA} and g_{NMDA} (g_{GABAa} and g_{GABAb}). Receptor-specific efficacies may vary

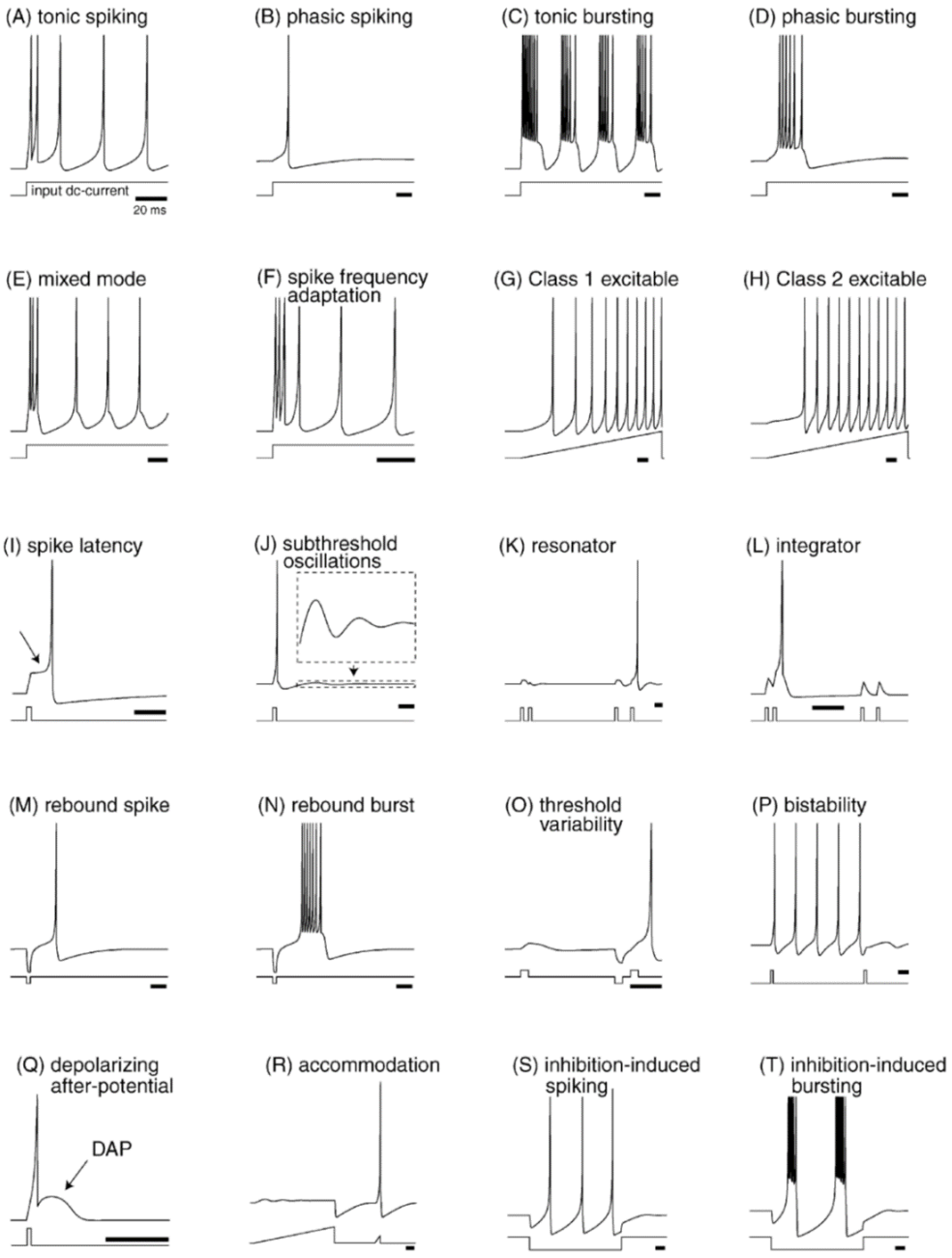

Figure 2.13: Summary of the neuro-computational properties of biological spiking neurons, simulated with the Izhikevich neuron model (Izhikevich, 2004). Each horizontal bar denotes a 20 ms time interval.

depending on which brain area (and cortical layer) is being modeled. However, η_{AMPA} is often greater than η_{NMDA} (Myme et al., 2003).

The total synaptic current i_{syn} in Eq. 2.4 is then given as:

$$\begin{aligned} i_{\text{syn}} = & -g_{\text{AMPA}}(v-0) \\ & - g_{\text{NMDA}} \frac{\left(\frac{v+80}{60}\right)^2}{1+\left(\frac{v+80}{60}\right)^2}(v-0) \\ & - g_{\text{GABAa}}(v+70) \\ & - g_{\text{GABAb}}(v+90). \end{aligned} \tag{2.8}$$

2.3.2.3 Synaptic Plasticity

Synapses in the brain can also undergo plasticity, where they change their efficacy or weight strength according to recent presynaptic and postsynaptic activity.

Synaptic plasticity was first proposed as a mechanism for learning and memory on the basis of theoretical analysis (Hebb, 1949), who postulated that when one neuron drives the activity of another neuron, the connection between these neurons is potentiated. Hebbian learning describes the change in the i-th weight strength, Δw_i, as:

$$\Delta w_i = \eta x_i y, \tag{2.9}$$

where η is the learning rate, x_i is the i-th input to the neuron, and y is the neuron's postsynaptic response. This principle became later known as "Cells that fire together, wire together".

The principle of Hebbian-like synaptic potentiation was later supplemented with the complementary principle of synaptic depression between two neurons that are not sufficiently coactive (Sejnowski, 1977; Stent, 1973). The experimental correlates of these theoretically

proposed forms of synaptic plasticity are called Long-Term Potentiation (LTP) and Long-Term Depression (LTD) (Bi and Poo, 2001; Song et al., 2000), and together form an update rule called Spike-Timing Dependent Plasticity (STDP):

$$\frac{dw_{i,j}}{dt} = \delta + \beta(\mathrm{LTP}_{i,j} + \mathrm{LTD}_{i,j}), \tag{2.10}$$

where $w_{i,j}$ is the synaptic weight from presynaptic neuron i to postsynaptic neuron j, β is the learning rate, and δ is a bias often set to zero or a positive to push the network towards positive weight increases for low synaptic input.

Although prevalent in cortical synapses, the model presented in this thesis does not attempt to model synaptic plasticity. However, Chapter 7 describes a computational principle that might be the result of synaptic plasticity mechanisms in the brain (Carlson et al., 2013). Synaptic plasticity may be included in future iterations of the model presented in this thesis (see Chapter 8).

Chapter 3

CARLsim: A Large-Scale Spiking Neural Network Simulator

3.1 Introduction

Spiking Neural Network (SNN) models play an important role in understanding brain function (Eliasmith et al., 2012; Maass, 1997; Rieke et al., 1999) and in designing neuromorphic devices with the energy efficiency, computational power, and fault-tolerance of the brain (Boahen, 2006; Cassidy et al., 2014; Khan et al., 2008; Minkovich et al., 2014; Schemmel et al., 2010). These models represent a compromise between execution time and biological fidelity by capturing essential aspects of neural communication, such as spike dynamics, synaptic conductance, and plasticity, while foregoing computationally expensive descriptions of spatial voltage propagation found in compartmental models (Izhikevich, 2007; Koch, 2004).

However, developing efficient simulation environments for SNN models is challenging due to the required memory to store the neuronal/synaptic state variables and the time needed to solve the dynamical equations describing these models. A number of simulators are already

available to the public (for an overview see Beyeler et al. (2015a); Brette et al. (2007)), each providing their own unique qualities and trade-offs based on the employed abstraction level and supported computer hardware. Given the considerable potential for parallelization of artificial neural networks (Nordström and Svensson, 1992), it is not surprising that most simulators today offer implementations on parallel architectures, such as computer clusters or GPUs. However, in order to be of practical use to the computational modeling community, an SNN simulator should not only be fast, but also freely available, capable of running on affordable hardware, have a user-friendly interface, include complete documentation, and provide tools to easily design, construct, and analyze SNN models.

To meet these challenges, we have developed CARLsim 3, a user-friendly, GPU-accelerated SNN simulation framework written in C/C++ that is capable of simulating large-scale neural models without sacrificing biological detail. Due to its efficient GPU implementation, CARLsim is useful for designing large-scale SNN models that have real-time constraints (e.g., interacting with neuromorphic sensors or controlling neurorobotics platforms). To promote its use among the computational neuroscience and neuromorphic engineering communities, CARLsim is provided as an open-source package, which can be freely obtained from: `http://www.socsci.uci.edu/~jkrichma/CARLsim`.

3.2 CARLsim

CARLsim is a C/C++ based SNN simulator that allows execution of networks of Izhikevich spiking neurons (Izhikevich, 2003) with realistic synaptic dynamics on both generic x86 CPUs and standard CUDA-enabled GPUs. CARLsim can be understood as a computational framework for the exploration of experimental data, functional applications, and theoretical models of brain function, with applications to theoretical neuroscience, real-time computing, and neuromorphic engineering (Fig. 3.1).

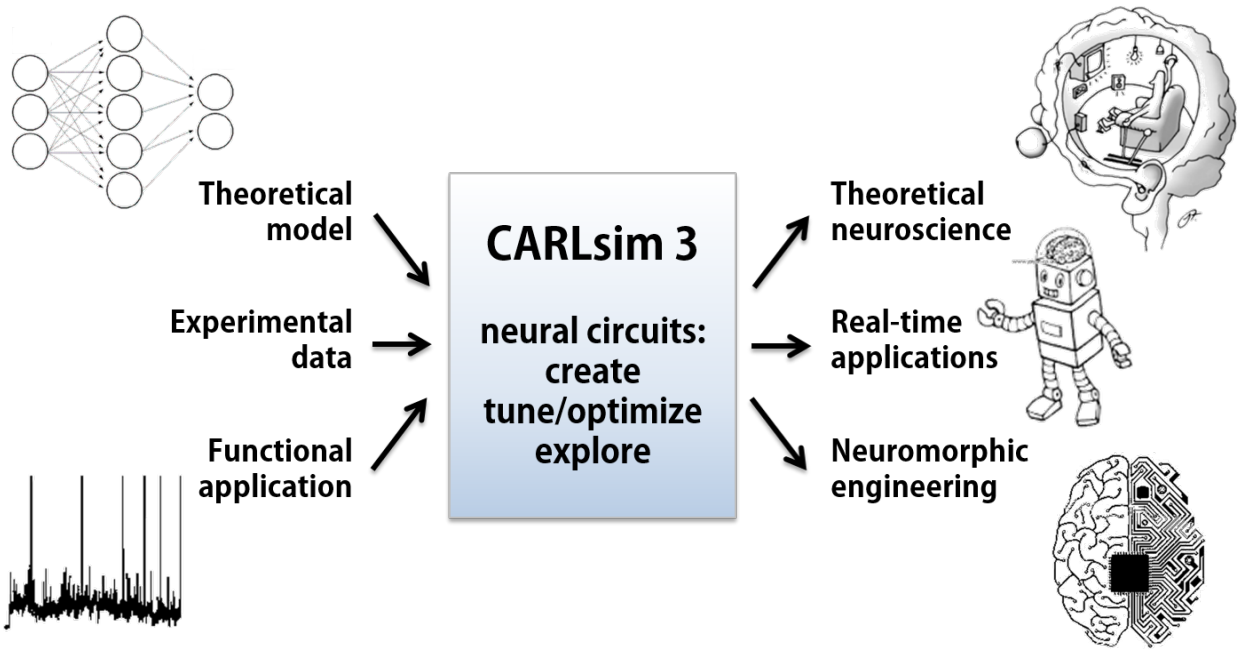

Figure 3.1: CARLsim: A neuromorphic framework for constructing functional SNN models.

The simulator was first introduced in 2009 (now referred to as CARLsim 1), where it demonstrated near real-time performance for 10^5 spiking neurons on a single NVIDIA GTX 280 GPU (Nageswaran et al., 2009). CARLsim 2 added basic support for synaptic conductances, Spike-Timing Dependent Plasticity (STDP), and Short-Term Plasticity (STP) (Richert et al., 2011). The most recent release, CARLsim 3, greatly expands the functionality of the simulator by adding a number of features that enable and simplify the creation, tuning, and simulation of complex networks with spatial structure (Beyeler et al., 2015a). These features include: i) real-time and offline data analysis tools, ii) a more complete STDP implementation that includes dopaminergic neuromodulation, and iii) an automated parameter tuning interface. In addition, several software engineering techniques and more complete documentation were introduced in the latest release to ensure the integrity of the current code base and to lay the groundwork for the success of future releases. The following subsections will explain these achievements in detail.

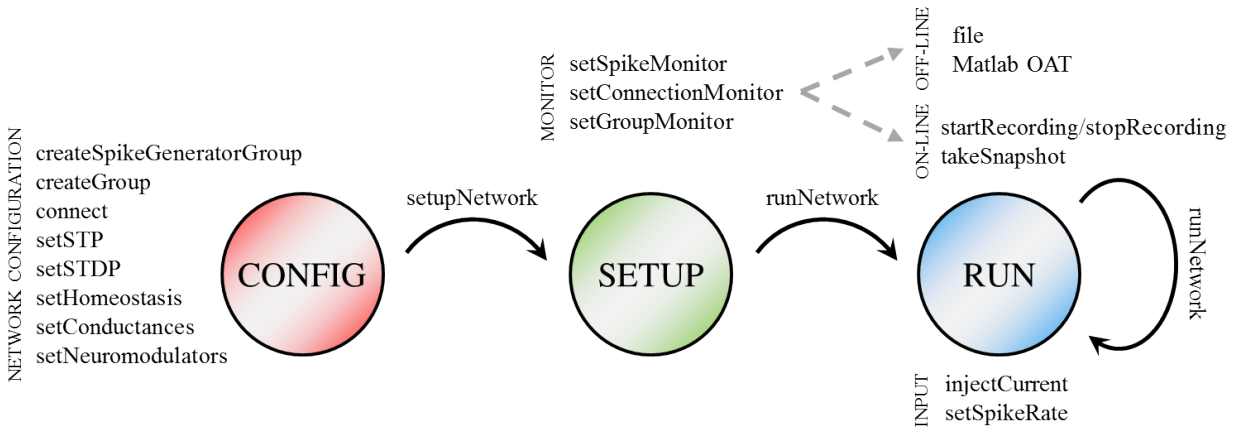

Figure 3.2: CARLsim state diagram. The user specifies neuronal and network details in the `CONFIG` state. Calling `setupNetwork` moves the simulation to the `SETUP` state, where data monitors can be set. Calling `runNetwork` moves the simulation to the `RUN` state, where input stimuli are set and the simulation is executed. Data can be analyzed either on-line (e.g., by accessing collected spike data or by taking a snapshot of the current synaptic weights), or off-line via the Offline Analysis Toolbox (OAT).

3.2.1 CARLsim User Interface: General Workflow

The workflow of a typical CARLsim 3 simulation is organized into three distinct, consecutive states (see Fig. 3.2): the configuration state (`CONFIG`), the set up state (`SETUP`), and the run state (`RUN`). User functions in the C++ Application Programming Interface (API) are grouped according to these stages, which streamline the process and prevent race conditions. State transitions are handled by special user methods such as `CARLsim::setupNetwork` and `CARLsim::runNetwork`. The basic workflow is outlined in Listings 3.1 – 3.3, which are explained in detail below.

3.2.1.1 The Configuration State

The first step in using CARLsim 3 (`libCARLsim`) is to import the library and instantiate the main simulation object, which prepares the simulation for execution in either `CPU_MODE` or `GPU_MODE`. From then on, the simulation is in `CONFIG` state, allowing the properties of the

neural network to be specified.

Similar to PyNN (Davison et al., 2009) and many other simulation environments, CARLsim uses groups of neurons and connections as an abstraction to aid defining synaptic connectivity. Different groups of neurons can be created from a one-dimensional array to a three-dimensional grid via `CARLsim::createSpikeGeneratorGroup` or `CARLsim::createGroup`, and connections can be specified depending on the relative grid placement of neurons via `CARLsim::connect`. This allows for the creation of networks with complex spatial structure.

Listing 3.1: CARLsim CONFIG state

```
1  #include <carlsim.h>
2
3  // ---------------------------------- CONFIG STATE ---------------------------------- //
4  CARLsim sim("basic", CPU_MODE, USER);
5
6  // create groups
7  Grid3D grpSize(10, 10, 5); // 10x10x5 neurons in each group
8  int gIn = sim.createSpikeGeneratorGroup("input", nNeur, EXCITATORY_NEURON);
9  int gOut = sim.createGroup("output", nNeur, EXCITATORY_NEURON);
10 sim.setNeuronParameters(gOut, 0.02f, 0.2f, -65.0f, 8.0f); // RS
11
12 // connect input to output group
13 sim.connect(gIn, gOut, "gaussian", RangeWeight(0.01f), 0.1f, RangeDelay(1, 20),
14     RadiusRF(3, 3, 3), SYN_PLASTIC);
15
16 // enable STDP on all incoming synapses to gExc
17 float alphaPlus = 0.1f, tauPlus = 20.0f, alphaMinus = 0.1f, tauMinus = 20.0f;
18 sim.setESTDP(gOut, true, STANDARD, ExpCurve(alphaPlus, tauPlus, -alphaMinus, tauMinus));
19
20 // disable conductances to run CUBA mode
21 sim.setConductances(false);
```

An example network is being configured in Listing 3.1. Here, the CARLsim object is instantiated (line 4), two neuronal groups are created (lines 8–10), a connection between these two groups is established (lines 13–14) as well as subjected to STDP (lines 17–18), and all synapses are defined to be conductance-based (line 21).

CARLsim currently supports Izhikevich spiking neurons (Izhikevich, 2003) with either current-based or conductance-based synapses, but more neuron types are planned for the future. In the above example, two groups of 500 Izhikevich neurons each are arranged on a $10 \times 10 \times 5$ three-dimensional grid (lines 7–9). The keyword `EXCITATORY_NEURON` denotes that all neurons in these groups have glutamatergic synapses, although it is also possible to create neurons with GABAergic synapses (`INHIBITORY_NEURON`) or dopaminergic synapses (`DOPAMINERGIC_NEURON`). In order to refer to these groups in later method calls, the methods `createSpikeGeneratorGroup` and `createGroup` return a group ID, stored in `gIn` and `gOut`, respectively. Line 10 specifies the Izhikevich parameters of group `gOut`, making these neurons of class 1 excitability (i.e., regular spiking neurons), where 0.02, 0.2, -65.0, and 8.0 correspond respectively to the a, b, c, and d parameters of the Izhikevich neuron.

The two groups, `gIn` and `gOut`, are connected topographically using a Gaussian connection profile (lines 13–14), which connects neurons based on their relative distance in 1D, 2D, or 3D space as defined by the `Grid3D` struct (line 7). The spatial extent of these connections is given by the `RadiusRF` struct (line 14), which allows the receptive field of each neuron to be specified in 1D, 2D, or 3D space. The connection pattern in the above example script has a synaptic weight of value 0.01, 10 % connection probability, a synaptic delay uniformly distributed between 1 ms and 20 ms, and plastic synapses (`SYN_FIXED`) whose weights can range between 0 and 0.01. If one is not satisfied with the built-in connection types (`"one-to-one"`, `"full"`, `"random"`, and `"gaussian"`), a callback mechanism is available for arbitrary, user-specified connectivity.

CARLsim 3 allows users to choose from a number of synaptic plasticity mechanisms. These include standard equations for STP (Senn et al., 2001; Tsodyks et al., 1998), various forms of additive nearest-neighbor STDP (Izhikevich and Desai, 2003), and homeostatic plasticity in the form of synaptic scaling (Carlson et al., 2013). STDP is a paradigm that modulates the weight of synapses according to their degree of causality. In CARLsim, STDP is

specified post-synaptically, and can either apply to all glutamatergic synapses (i.e., where the pre-synaptic group is excitatory; E-STDP) or all GABAergic synapses (i.e., where the pre-synaptic group is inhibitory; I-STDP). In the present example, E-STDP is enabled on group `gOut`, and the shape of the STDP curve is set to the standard exponential curve (see Section 2.3.2.3).

For a selective list of other available function calls in `CONFIG` state, please refer to Fig. 3.2.

3.2.1.2 The Set Up State

Once the network has been specified, `CARLsim::setupNetwork` optimizes the network state for the chosen back-end (CPU or GPU) and moves the simulation into `SETUP` state (see Fig. 3.2). In this state, monitors can be set to record variables of interest (e.g., spikes, weights, state variables).

An example is shown in Listing 3.2, where a `SpikeMonitor` is set to record spikes for the duration of the simulation and dump the data to a `"DEFAULT"` file location (line 3). Analogously, a `ConnectionMonitor` will track synaptic weight changes over the time course of the simulation and dump the data to file (line 4). The default file location is given by the name of the group (as specified on lines 8 and 9 in Listing 3.1), which is a means to easily find relevant files during off-line analysis (see Section 3.2.3).

Listing 3.2: CARLsim SETUP state

```
1 // ---------------------------------- SETUP STATE ------------------------------------ //
2 sim.setupNetwork();
3 SpikeMonitor* spkOut = sim.setSpikeMonitor(gOut, "DEFAULT");
4 sim.setConnectionMonitor(gIn, gOut, "DEFAULT");
```

New in CARLsim 3 is a means to access recorded variables at run-time, without the computational overhead of writing data to disk. This is achieved by means of a `SpikeMonitor`

handle that is returned by `CARLsim::setSpikeMonitor`. This handle then allows read access of collected data during the `RUN` state.

For a selective list of other available function calls in `SETUP` state, please refer to Fig. 3.2.

3.2.1.3 The Run State

The first call to `CARLsim::runNetwork` will take the simulation into `RUN` state. In this state, input stimuli can be specified, and the simulation can be repeatedly run (or "stepped") for an arbitrary number of milliseconds.

Input can be generated via current injection or spike injection (the latter using Poisson spike generators, or a callback mechanism to specify arbitrary spike trains). CARLsim also provides plug-in code to generate Poisson spike trains from animated visual stimuli such as sinusoidal gratings, plaids, and random dot fields created via the `VisualStimulus` toolbox[1].

In Listing 3.3, input is generated via a Poisson spike generator object, `poissonIn` (line 3), which is associated with the group `gIn` (line 4). The `PoissonRate` object will then generate Poisson spike trains from a specified mean firing rate (i.e., given by the λ parameter of the Poisson process), which increases as a function of simulation time (line 9). The network is run for a total of ten trials, each of which has a 1 s duration (line 14).

In the above listing, the `SpikeMonitor` handle from the `SETUP` state is used to selectively record spike events of the group `gOut`. This is achieved by surrounding `CARLsim::runNetwork` with calls to `SpikeMonitor::startRecording` and `SpikeMonitor::stopRecording`. Public members of the `SpikeMonitor` class can then be accessed to either print a summary of the recorded network activity (line 18) or retrieve sophisticated activity metrics, such as the

[1] `VisualStimulus` is a MATLAB toolbox for generating, storing, and plotting 2D visual stimuli related to neuroscience. The toolbox was created during CARLsim development, but is also available as a standalone download from: `https://github.com/UCI-CARL/VisualStimulusToolbox`.

Listing 3.3: CARLsim RUN state

```
1  // ********************************** RUN STATE ********************************** //
2  // associate PoissonRate container with group gIn
3  PoissonRate poissonIn(nNeur.N);
4  sim.setSpikeRate(gIn, &poissonIn);
5
6  // run the network repeatedly for 1s (1*1000 + 0 ms) with different Poisson input
7  for (int i=1; i<=10; ++i) {
8      // update Poisson mean firing rate
9      float inputRateHz = i * 10.0f;
10     poissonIn.setRates(inputRateHz);
11
12     // run for 1 second, record spikes
13     spkOut->startRecording();
14     sim.runNetwork(1, 0);
15     spkOut->stopRecording();
16
17     // print summary of spike information
18     spkOut->print();
19
20     // return the percent of neurons with firing rates from 0 to 5 Hz
21     spikeMon->getPercentNeuronsWithFiringRate(0.0, 5.0);
22 }
```

population's mean firing rate or the percentage of neurons whose mean firing rates range between 0 Hz and 5 Hz (arbitrary range; line 21). CARLsim provides a wide range of these metrics, which become particularly useful when the simulation is used to search large parameter spaces using the built-in parameter tuning interface (Beyeler et al., 2015a; Carlson et al., 2014).

In addition to the real-time monitors, CARLsim 3 provides a versatile OAT written in MATLAB (see Section 3.2.3), which allows users to quickly visualize and easily analyze their network simulations.

For a selective list of other available function calls in `RUN` state, please refer to Fig. 3.2.

3.2.2 CARLsim Kernel: Simulation Engine

An important feature of CARLsim is the ability to run spiking networks not only on CPUs, but also on off-the-shelf NVIDIA GPUs using the CUDA framework.

The basic structure of the CUDA GPU architecture is shown in Fig. 3.3. Each GPU consists of multiple Streaming Multiprocessors (SMs) and a global memory accessible by all SMs. Each SM is built from multiple floating-point scalar processors, a cache/shared memory, and one or more special functions units, which execute transcendental functions such as sine, cosine, and square root operations. CUDA groups parallel threads into "warps", where the number of threads per warp varies depending on the specific CUDA architecture. Each SM also has at least one warp scheduler that is built to maximize the number of threads running concurrently. The warp scheduler monitors threads within a warp and automatically switches to another warp when a thread makes a time-consuming memory access.

The GPU implementation of CARLsim was written to optimize four main performance metrics: parallelism, memory bandwidth, memory usage, and thread divergence (Nageswaran et al., 2009). The simulation is broken into steps that update neuronal state variables in parallel (N-parallelism), and steps that update synaptic state variables in parallel (S-parallelism). Sparse representation techniques for spiking events and neuronal firing decrease both memory and memory bandwidth usage. Thread/warp divergence occurs when a thread executes a different operation than other threads in a warp causing the other threads to wait for its completion. CARLsim prevents thread/warp divergence by buffering data until all threads are ready to execute the same operation. At the same time, this technique enables efficient run-time access of recorded variables (e.g., via SpikeMonitor) that are stored on the GPU (Beyeler et al., 2015a).

The CARLsim 3 kernel introduces a number of software engineering techniques to assure the integrity of the current code base and to enable successful growth of the project in the

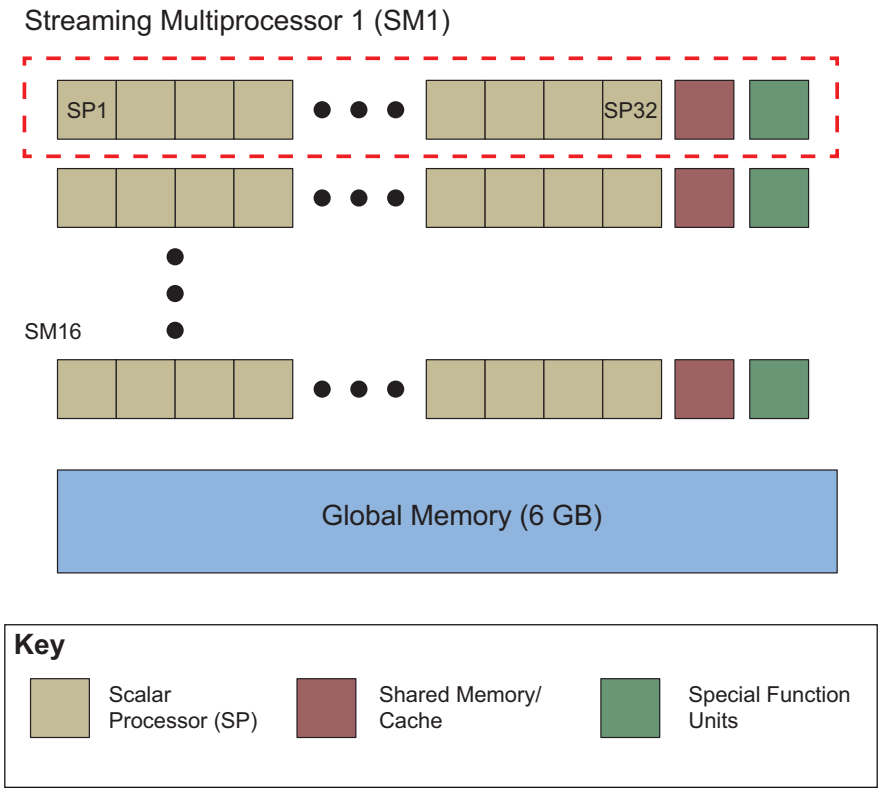

Figure 3.3: Simplified diagram of NVIDIA CUDA GPU architecture.

future (Beyeler et al., 2015a). For example, the pointer to implementation idiom (Eckel, 2000, Chapter 5: Hiding the Implementation) is used to programmatically separate user interface from implementation, which will simplify the addition of different front-ends and back-ends in the future. Regression testing (Kolawa and Huizinga, 2007), via Google Test[2], is used to ensure that the development of new functionality does not compromise the existing code base, and new features are validated using both functional and unit testing. The full test suite is visible to the user for automatic quality assurance after installation.

CARLsim development was mostly based on the M2090 Tesla GPU that utilizes the CUDA Fermi architecture. The Tesla M2090 has 6 GB of global memory, 512 cores (each operating at 1.30 GHz) grouped into 16 SMs, and a single precision compute power of 1331.2 GFLOPS. Each SM is composed of 32 SPs, 16 load/store units, 4 special function units, and two warp schedulers (Corporation, 2009). However, CARLsim is compatible with any NVIDIA GPU

[2]Google's C++ test framework can be obtained from: `https://github.com/google/googletest`.

that supports either the Fermi, Kepler, or Maxwell versions of the CUDA architecture. In addition, the software was tested on a variety of platforms such as Windows 7 and 8, Linux (Ubuntu 12.04, 12.10, 13.04, 13.10, and 14.04; CentOS 6; Arch Linux; OpenSUSE 13.1), and Mac OS X to ensure maximal user support.

3.2.3 MATLAB Offline Analysis Toolbox (OAT)

In addition to real-time monitors, CARLsim provides a versatile Offline Analysis Toolbox (OAT) written in MATLAB for the visualization and analysis of neuronal, synaptic, and network information. The OAT is designed to produce meaningful plots with minimal user input, by operating on binary files that were created during CARLsim simulations.

The OAT is organized into a hierarchy of classes, deriving either from a `Reader` base class to operate on raw binary files or from a `Monitor` base class to visualize data. Reader objects include a `SimulationReader` class for reading simulation configuration files, `SpikeReader` for reading Adress Event Representation (AER) spike files, and `ConnectionReader` for reading synaptic weight files. Monitor objects utilize reader objects to visualize network activity, either on a per-group basis via `GroupMonitor`, or for every group in the network via `NetworkMonitor`. Similarly, synaptic connections can be visualized and analyzed via `ConnectionMonitor`.

Listing 3.4: Visualizing network activity using the OAT.

```
1 initOAT
2 NM = NetworkMonitor('../results/sim_carl_logo.dat');
3 NM.plot;
```

The easiest way to visualize network activity after running a CARLsim simulation is illustrated in Listing 3.4. In three simple lines of code, the OAT environment is initialized by adding relevant directories to the MATLAB search path (line 1), a `NetworkMonitor` object

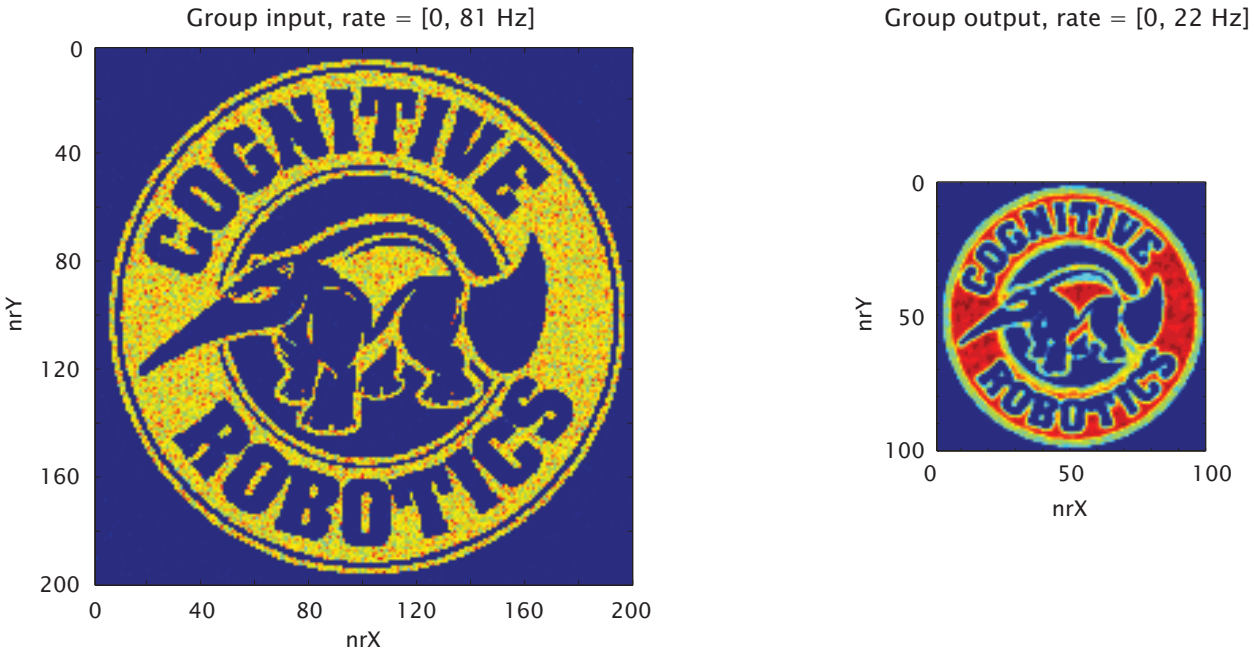

Figure 3.4: Visualizing network activity with the `NetworkMonitor` object of the OAT. A 200×200 input group is connected with a Gaussian connection profile to a 100×100 output group, effectively implementing a blur and subsampling step of the Gaussian image pyramid. Each pixel in the heatmap corresponds to the mean firing rate of a single neuron recorded over a 1 s duration (the hotter the color, the higher the activity). Plots can be retrieved by the user with only three lines of MATLAB code (see Listing 3.4).

is created and pointed to `../results/sim_carl_logo.dat`, a file automatically generated by running a CARLsim network with name "carl_logo" (line 2), and network activity for every group is visualized in a way that best suits both group size and simulation duration (line 3).

An example output is shown in Fig. 3.4 for a network consisting of two groups (`"input"` and `"output"`) connected with a Gaussian connection profile, with neurons arranged on a 200×200 and 100×100 grid, respectively. Because in this example neurons are arranged on a 2D grid, the `NetworkMonitor` object decided to visualize neuronal activity as 2D heatmap, where every pixel corresponds to the mean firing rate of a particular neuron (the hotter the color, the higher the activity).

Plots can be easily customized by passing optional arguments to the `plot` method of the

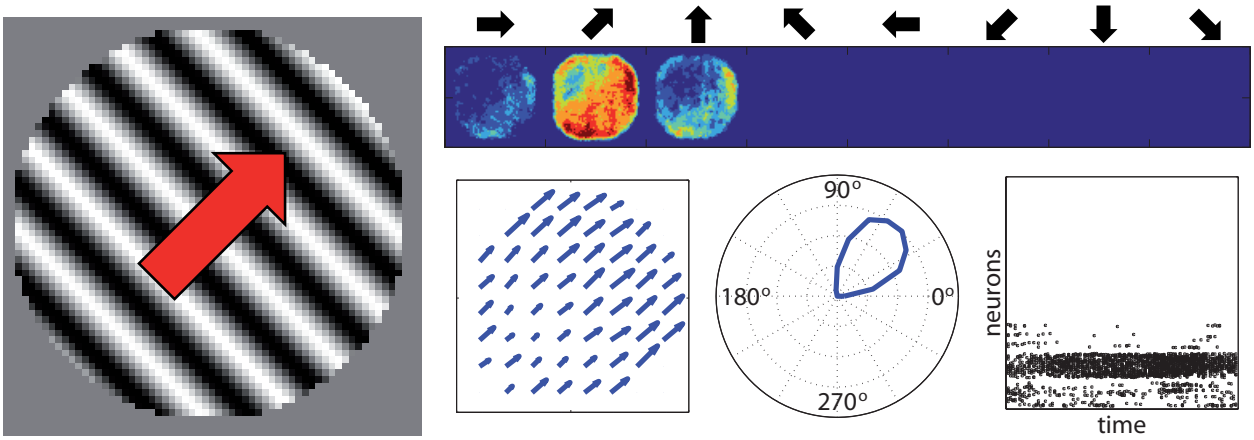

Figure 3.5: Supported OAT plot types include heat maps, flow fields, polar plots, and raster plots. Shown is the activity of a network of directionally selective neurons in response to a drifting sinusoidal grating (left).

`NetworkMonitor` class, such as by specifying the plot type or the binning window. A single-line command is available to store the frame-by-frame animation of network activity to a movie file. Similar functionality is provided on a per-group basis via the `GroupMonitor` object.

The OAT provides a number of different ways to visualize data, as illustrated in Fig. 3.5. Here, a grayscale movie of a sinusoidal grating drifting in an up-right direction was generated using the `VisualStimulus` toolbox (left), and served as input to a network of directionally selective neurons, akin to neurons in the primary visual cortex (see Chapter 4). The resulting network activity can be visualized either as a heat map (top), a flow field, a polar plot, or a raster plot (bottom row), making it easy for the user to visually inspect and verify arbitrarily complex networks created with CARLsim.

3.3 CARLsim Performance

In order to demonstrate the efficiency and scalability of CARLsim 3, we ran simulations consisting of various sized SNNs and measured their execution time (Beyeler et al., 2015a).

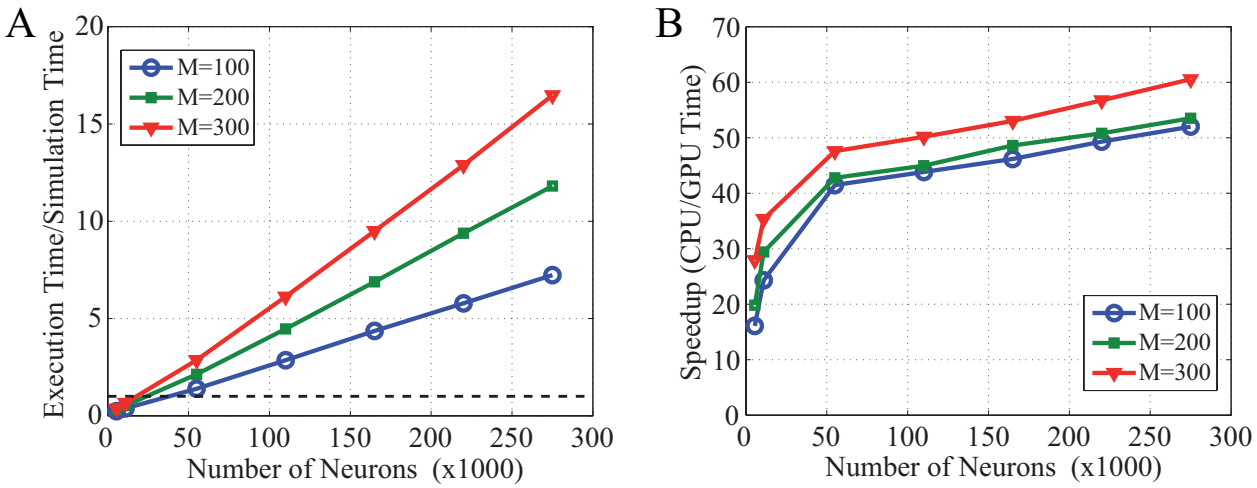

Figure 3.6: **A**, Ratio of execution time to simulation time versus number of neurons for simulations with 100, 200, and 300 synapses per neuron. The dotted line represents simulations running in real time. **B**, Simulation speedup versus number of neurons. Speedup increases with neuron number. Models with 10^4 neurons or more have the most impressive speedup.

GPU simulations were run on a single NVIDIA GTX 780 (3 GB of memory) using CUDA, and CPU simulations were run on an Intel Core i7 CPU 920 operating at 2.67 GHz.

The results of these benchmarks are summarized in Fig. 3.6. Networks consisted of 80 % excitatory and 20 % inhibitory neurons, with an additional 10 % of neurons being Poisson spike generators that drove network activity. The two neuronal populations were randomly connected with a fixed connection probability, and E-STDP was used to generate sustained irregular activity as described by Vogels and Abbott (2005). The number of synapses per neuron ranged from 100 to 300 connections, and the overall network activity was ∼10 Hz.

GPU simulation speed, given as the ratio of GPU execution time over real-time, is plotted in Fig. 3.6A. Execution time scaled linearly with workload, both in terms of number of neurons as well as number of synapses. In Richert et al. (2011) we reported that networks with 110,000 neurons and 100 synaptic connections per neuron took roughly two seconds of clock time for every second of simulation time (CARLsim 2). The current release (CARLsim 3) was slightly slower in this scenario, taking roughly 2.5 s of clock time for every second of simulation time, which is not surprising given the large number of additional features in the

present release. On the other hand, GPU mode still significantly outperformed CPU mode (see Fig. 3.6B). GPU execution time was up to 60 times faster than the CPU. This result is due to a combination of newer hardware (GTX 780) as well as code-level optimization that allowed the effective use of S-parallelism (Nageswaran et al., 2009) for new features such as synaptic receptor ratios and different shaped STDP curves, which are not possible in a single-threaded CPU simulation. In summary, despite the additional features and new programming interface, CARLsim 3 is similar in performance to our previous releases and is highly optimized for SNNs on the order of 105 neurons and 107 synapses.

3.4 CARLsim Applications

CARLsim has been used to construct large-scale simulations of cognitive processes on the order of $10^4 - 10^5$ neurons and millions of synapses, with examples that include models of visual processing (Beyeler et al., 2013, 2014, 2015b; Richert et al., 2011), neuromodulation (Avery and Krichmar, 2015; Avery et al., 2012), neural plasticity (Carlson et al., 2013), and somatosensory processing (Chou et al., 2015). Although their efficient implementation is challenging due to the associated computational cost, investigating large-scale models of cortical networks in more biological detail is widely regarded as crucial in order to understand brain function (Honey et al., 2007).

One of our recent works (Beyeler et al., 2013) concerned the ability of a large-scale spiking neural network model to rapidly categorize highly correlated patterns of neural activity such as handwritten digits from the MNIST database (LeCun et al., 1998). Although many studies have focused on the design and optimization of neural networks to solve visual recognition tasks, most of them either lack neurobiologically plausible learning rules or decision-making processes. In contrast, our model demonstrated how a low-level memory encoding mechanism based on synaptic plasticity could be integrated with a higher-level decision-making paradigm

to perform the visual classification task in real-time.

The model consisted of Izhikevich neurons and conductance-based synapses for realistic approximation of neuronal dynamics, an STDP-like synaptic learning rule for memory encoding (previously described by Brader et al. (2007)), and an accumulator model for memory retrieval and categorization (Smith and Ratcliff, 2004). Grayscale input images were fed through a feed-forward network consisting of visual cortical areas V1 and V2 (selective to one of four spatial orientations, in 45° increments), which then projected to a layer of downstream classifier neurons through plastic synapses that implement the STDP-like learning rule mentioned above. Decision neurons were equally divided into ten pools, each of which would develop selectivity to one class of input stimuli from the MNIST dataset (i.e., one of the ten digits) as a result of training. Population responses of these classifier neurons were then integrated over time to make a perceptual decision about the presented stimulus.

The model constitutes an important proof of concept; that is, i) to show how considerably hard problems such as visual pattern recognition and perceptual decision-making can be solved by general-purpose neurobiologically inspired cortical models solely relying on local learning rules that operate on the abstraction level of a synapse, and ii) to do it in real-time. The network achieved 92% correct classifications on MNIST in 100 rounds of random sub-sampling, which provides a conservative performance metric, yet is comparable to other SNN approaches (Brader et al., 2007; Querlioz et al., 2011). Additionally, the model correctly predicted both qualitative and quantitative properties of reaction time distributions reported in psychophysical experiments. The full network, which comprised of $71,026$ neurons and approximately 133 million synapses, ran in real-time on a single NVIDIA Tesla M2090, which demonstrates the efficacy of the CARLsim implementation. Moreover, because of the scalability of the approach and its neurobiological fidelity, the model can be extended to an efficient neuromorphic implementation that supports more generalized object recognition and decision-making architectures found in the brain.

3.5 Related Work

Today, a wide variety of simulation environments are available to the computational neuroscience community that allow for the simulation of SNNs. Although many simulators share similar approaches and features, most have unique qualities that distinguish them from the others. In an effort to identify and highlight these qualities, we compared a list of features provided by a number of other SNN simulators with CARLsim 3 (Beyeler et al., 2015a). We limited this comparison to large-scale SNN simulators that are most common to CARLsim in that they: i) are open source, ii) support the clock-driven and parallelized simulation of point neurons, such as LIF, Izhikevich or aIF neurons, iii) have conductance-based synapses, and iv) provide some form of synaptic plasticity. Specifically, we chose (in alphabetical order) Brian 2 (Goodman and Brette, 2008), GeNN 2 (Nowotny, 2011), NCS 6 (Hoang et al., 2013), NeMo 0.7 (Fidjeland et al., 2009), Nengo 2 (Bekolay et al., 2014), NEST 2.6 (Gewaltig and Diesmann, 2007) and PCSIM 0.5 (Pecevski et al., 2009).

Table 3.1 compares different simulation environments with respect to features of both biological realism and technical capability. A specific feature is indicated as either fully implemented (‘ X ‘) or absent (blank, ‘ ‘). A ‘ / ‘ denotes a feature that is only partially implemented (e.g., neuromodulation in NEST), requires substantial user efforts to implement (e.g., DA-STDP in Brian), or is reportedly untested (e.g., Windows support for PCSIM). Although we tried to be as objective as possible in assigning labels, categorization remains subjective in nature. In addition, some of these features were not well-documented; but we were still able to verify their existence by reading through the actual source code and reaching out to the authors for clarification. Overall, we believe that the table accurately and fairly reflects the development status for each of the listed SNN simulators at the time of this publication.

All of the simulators listed in Table 3.1 offer many features that allow the simulation of complex neural networks on several back-ends. In addition, if users are interested in modeling

Table 3.1: Feature comparison for some common open-source SNN simulators (non-exhaustive list)

	Neuron model			Synapse model			Synaptic plasticity			Tools			Integration methods			Front-ends			Back-ends			Platforms		
	Leaky Integrate-and-Fire (LIF)	Izhikevich 4-param	Hodgkin-Huxley	Current-Based (CUBA)	Conductance-Based (COBA)	Neuromodulation	Short-Term Plasticity (STP)	STDP	Synaptic scaling / homeostasis	Parameter tuning	Analysis and visualization	Regression suite	Forward / exponential Euler	Exact integration*	Runge-Kutta	Python / PyNN	C / C++	Java	Single/Multi-threaded	distributed	GPU	Linux	Mac OS X	Windows
CARLsim 3		X		X	X	/	X	X	X	X	X	X	X				X		/		X	X	X	X
CARLsim 2		X		X	X		X	X			/		X				X		/		X	X	X	X
Brian 2	X	X	X	X	X	/	X	X	X	/	/	X	X	X	X	X	X		X	/		X	X	X
GeNN 2		X	X	X	X	/	/	X	/				X				X		/		X	X	X	X
NCS 6	X	X	X	X	/		X	X			/		X			X	X		X	X	X	X		
NeMo 0.7		X			/	/	X	X			X	X	X			X	X		X		X	X	X	X
Nengo 2	X	X	X	X	X	X	X	/	X	X	X	X			X	X		X	X		X	X	X	X
NEST 2.6	X	X	X	X	X	/	X	X	X			X	X	X	X	X			X	X		X	X	
PCSIM 0.5	X	X	X	X	X	/	X	X	X		/	X	X			X	/	X	X	X		X	X	/

*as defined in Rotter and Diesmann (1999)

certain details of the biological model that are not natively supported, all of the simulators offer ways to extend the code base, either by ways of plug-in code, implementation inheritance, or dynamic code generation. For example, both Brian and GeNN offer ways for the user to formulate any neuronal, synaptic, or learning model they please. This fact makes the simulators invaluable for power-users, but may be difficult for less-experienced programmers and may prohibit code-level optimizations for certain user-defined functionality. Nengo provides flexibility through its scripting interface, and also provides a graphical user interface to construct SNNs at different levels of abstraction. In Nengo, large-scale functional networks can be achieved using the Neural Engineering Framework (NEF), which is a theoretical framework that can use anatomical constraints, functional objectives, and control theory to find the set of weights that approximate some desired functionality (Eliasmith and Anderson, 2002).

Although all of the simulation environments listed in Table 3.1 have their own pros and cons, we believe that CARLsim 3 has advantages when it comes to efficiently simulating large-scale SNNs without having to sacrifice biological realism. In particular, we have made serious efforts to improve the usability of our platform by means of platform compatibility (Linux, Mac OS X, and Windows), rigorous code documentation (including an extensive user guide and tutorials), a regression suite for functional code verification, and a MATLAB toolbox for the visualization and analysis of neuronal, synaptic, and network information. CARLsim 3 provides native support for a range of spike-based synaptic plasticity mechanisms and topographic synaptic projections, as well as being among the first to provide support for a network-level parameter tuning interface (Carlson et al., 2014). Additionally, the PyNN-like interface, flexible visualization tools, and much improved documentation make CARLsim 3 easy to use.

3.6 Conclusion

CARLsim is an open-source, C/C++ based SNN simulator that allows the execution of networks of Izhikevich spiking neurons with realistic synaptic dynamics on both generic x86 CPUs and standard off-the-shelf CUDA-enabled GPUs. The simulation library has minimal external dependencies and provides users with a PyNN-like programming interface. Additionally, CARLsim provides online and offline data analysis tools as well as support for an automated parameter tuning framework. The library, documentation, tutorials and examples can be obtained from: `http://www.socsci.uci.edu/~jkrichma/CARLsim`.

Chapter 4

CUDA Implementation of the Motion Energy Model

4.1 Introduction

Several software packages are available to the public that deal with the neurobiologically plausible modeling of motion perception in the mammalian brain, such as spatiotemporal-energy models like the motion energy model of Simoncelli and Heeger (1998) (see Section 2.1.3.1), or gradient-based models like ViSTARS (Browning et al., 2009a,b). However, in order for these frameworks to become practical in, for example, neuromorphic or robotics applications, they must be capable of running large-scale networks in real time. Yet real-world integration is often prohibited due to engineering requirements, programming intricacies, and the sheer computational cost that come with large-scale biological models. Instead, such models often find application only in constrained or virtual environments, which may severely limit their explanatory power when it comes to generalizing findings to real-world conditions.

In this chapter I describe an efficient software implementation of the motion energy model (Si-

moncelli and Heeger, 1998), one of the most supported cortical models for visual motion processing. After a rigorous mathematical treatment of the motion energy model (henceforth referred to as the S&H model), I will discuss details of the new implementation, and compare its computational performance to Simoncelli and Heeger's own C/MATLAB based code. By leveraging the parallel processing capabilities of GPUs, I aim to develop a CUDA implementation of the S&H model that can handle constant video streams ($\sim 360 \times 480$ pixels) of up to 30 frames per second in (near) real time, which will play a key role in enabling the practical feasibility of the empirical studies described in the following chapters.

4.2 Spatiotemporal-Energy Model of V1

As described in Section 2.1.3.1, the S&H model belongs to the "integrationist" class of models, which solve the aperture problem in two stages: In a first stage (assigned to motion processing in V1), local 1D motion signals are extracted. In the second stage (assigned to motion processing in MT), these signals are combined nonlinearly to recover 2D velocity.

In the following I will concentrate on the V1 stage, which transforms a series of image frames into simulated neural activity that is modulated by the direction and speed of local motion. At the end of the V1 stage, the model is conceptually equivalent to an elaborated Reichardt detector (van Santen and Sperling, 1985). How signals from V1 can be combined in MT to recover 2D velocity, by relying solely on dynamics and properties gleaned from known electrophysiological and neuroanatomical evidence, will be the focus of Chapter 5.

4.2.1 Input Stimuli

Input to the S&H model is a visual stimulus that is represented as a light intensity distribution $I(x, y, t)$, which is a function of two spatial dimensions (x, y) and time t.

Typically, a visual stimulus is processed at different spatiotemporal resolutions (or scales), r, in order to extract motion cues that are invariant to feature size. In the following we will consider three different scales. The first scale, $r = 0$, was equivalent to processing at the original image (and time) resolution. The other two scales ($r = 1$ and $r = 2$) were achieved by successively blurring the image with a Gaussian kernel. The three stimuli $I_r(x, y, t)$ can thus be expressed as:

$$I_0(x, y, t) = I(x, y, t) \tag{4.1}$$

$$I_1(x, y, t) = \exp\left(\frac{-(x^2 + y^2 + t^2)}{2}\right) * I_0(x, y, t) \tag{4.2}$$

$$I_2(x, y, t) = \exp\left(\frac{-(x^2 + y^2 + t^2)}{2}\right) * I_1(x, y, t), \tag{4.3}$$

where $*$ denotes convolution. In order to circumvent the noncausality of these convolutions (i.e., where the response depends both on past and future stimulus intensities), a time delay of four frames was introduced (see Simoncelli and Heeger (1998)).

4.2.2 V1 Simple Cells

A large body of research has found that neurons located in V1 that project to MT are directionally selective and may be regarded as local motion energy filters (Adelson and Bergen, 1985; DeAngelis et al., 1993; Movshon and Newsome, 1996). In our network, V1 simple cells are modeled as linear space-time oriented filters whose receptive fields are third derivatives of a Gaussian (Simoncelli and Heeger, 1998). These filters are very similar to a Gabor filter, but more computationally convenient as they allow for separable convolution computations.

An example of a spatiotemporal receptive field is illustrated in Fig. 4.1, where the colored ovals correspond to the orientation of the positive (green) and negative (red) lobes of the spatiotemporal filter. If a drifting dot traces out a path (dashed line) in space (x, for now

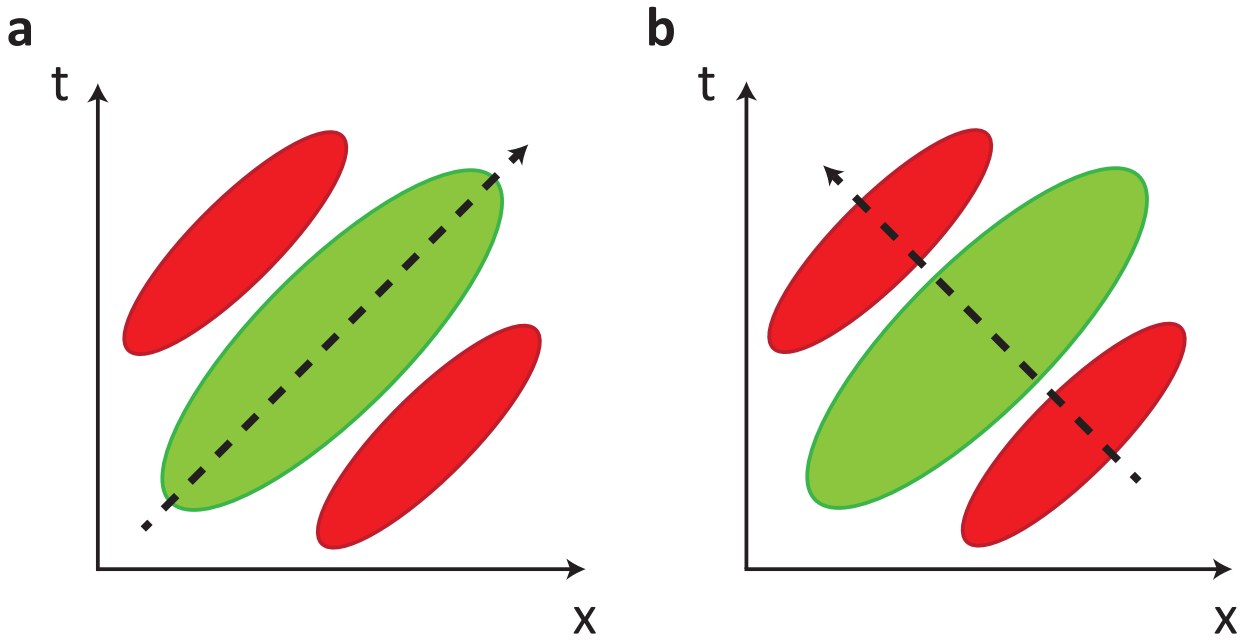

Figure 4.1: A drifting dot traces out a path (dashed line) in space (x, ignoring y) and time (t). The colored ovals correspond to the orientation of the positive (green) and negative (red) lobes of a spatiotemporal filter. **A**, If the filter is oriented in the same way as the dot's space-time path, it could be activated by this motion. **B**, A dot moving in the opposite direction would always contact both positive and negative lobes of the filter and therefore could never produce a strong filter response. Adapted from Bradley and Goyal (2008).

ignoring y) and time (t) that is oriented in the same way as the lobes, then the filter could be activated by this motion (Fig. 4.1A). A dot moving in the orthogonal direction would not elicit a filter response because its path intersects both positive and negative lobes of the filter (as depicted in Fig. 4.1B).

The full set of V1 linear receptive fields consisted of 28 space-time orientations that are evenly distributed on the surface of a sphere in the spatiotemporal frequency domain (Fig. 2.3). The k-th space-time oriented filter in the V1 population can be described by a unit vector $\vec{u}_k = (u_{k,x}, u_{k,y}, u_{k,t})'$ that is parallel to the filter orientation, where $k = 1, 2, \ldots, 28$, and $'$ denotes vector transposition. For more information please refer to Simoncelli and Heeger (1998).

First, input images were filtered with a 3D Gaussian corresponding to the receptive field size

of a V1 simple cell:

$$f_r(x,y,t) = \exp\left(\frac{-(x^2+y^2+t^2)}{2\sigma^2_{\text{v1simple}}}\right) * I_r(x,y,t), \tag{4.4}$$

where $*$ is the convolution operator, r denotes the scale, and $\sigma_{\text{v1simple}} = 1.25$ pixels.

Then the underlying linear response of a simple cell at spatial location (x, y) and scale r with space-time orientation k is equivalent to the third-order derivative in the direction of $\vec{u}_k$; that is,

$$L_{k,r}(x,y,t) = \alpha_{\text{v1lin}} \sum_{T=0}^{3} \left[\sum_{Y=0}^{3-T} \left[\frac{3!}{X!\,Y!\,T!} \left(u_{k,x}\right)^X \left(u_{k,y}\right)^Y \left(u_{k,t}\right)^T \frac{\partial^3 f_r(x,y,t)}{\partial x^X \partial y^Y \partial t^T} \right] \right], \tag{4.5}$$

where ! denotes the factorial, $X = 3 - Y - T$, and $\alpha_{\text{v1lin}} = 6.6084$ is a scaling factor. Note that the two sums combined yield exactly 28 summands. This operation is equivalent to Eq. 2 in the original paper, and can also be expressed using vector notation:

$$\vec{L}_r = \alpha_{\text{v1lin}}\, \mathbf{M}\, \vec{b}_r, \tag{4.6}$$

where $\vec{L}_r$ is the set of all k V1 responses at scale r, each element of $\vec{b}_r$ is one of the separable derivatives in Eq. 4.5 at scale r, and each element of the 28×28 matrix $\mathbf{M}$ is a number $3!/(X!\,Y!\,T!)\left(u_{k,x}\right)^X \left(u_{k,y}\right)^Y \left(u_{k,t}\right)^T$. Each row of $\mathbf{M}$ has a different value for k, and each column of $\mathbf{M}$ has different values for X, Y, and T. I will make use of this notation in Section 4.2.3, where I will explain the construction of synaptic projections from V1 simple cells to V1 complex cells.

At this stage of the model it is possible that filter responses $\mathbf{L}_{k,r}$ at positions (x, y) close to the image border have become unreasonably large. These edge effects can be suppressed by applying a scaling factor to $\mathbf{L}_{k,r}$ whenever (x, y) is near an image border.

Simple cell responses can then be constructed by half-squaring and normalizing the linear responses $\mathbf{L}_{k,r}$ from Eq. 4.5 within a large Gaussian envelope (Fig. 4.2). Critically, the normalization process implements a division operation, which allows for a close approximation

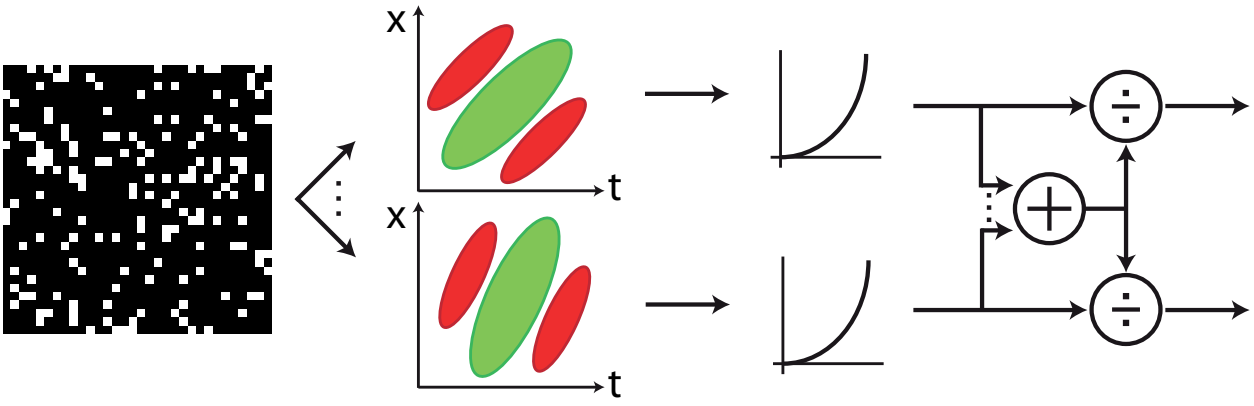

Figure 4.2: V1 simple cell responses are generated from a visual stimulus (such as a field of randomly moving dots) by applying a bank of linear spatiotemporal filters, a half-square operation, and divisive normalization.

of maximum likelihood estimation (Denève et al., 1999), and is a ubiquitous operation in the brain (Carandini and Heeger, 2012).

A simple cell response was then described as follows:

$$S_{k,r}(x,y,t) = \frac{\alpha_{\text{filt}\to\text{rate},r}\,\alpha_{\text{v1rect}}\,L^2_{k,r}(x,y,t)}{\alpha_{\text{v1norm}}\exp\left(\frac{-(x^2+y^2)}{2\sigma^2_{\text{v1norm}}}\right) * \left(\frac{1}{28}\sum_{k=1}^{28} L^2_{k,r}(x,y,t)\right) + \alpha^2_{\text{v1semi}}} \tag{4.7}$$

where $*$ is the convolution operator. The scaling factors $\alpha_{\text{v1rect}} = 1.9263$ and $\alpha_{\text{v1semi}} = 0.1$ (the semi-saturation constant) had the same values as in the original S&H model. Instead of having a single global normalization, our normalization occurs within a large spatial neighborhood (Gaussian half-width $\sigma_{\text{v1norm}} = 3.35$ pixels), which is thought to be more biologically realistic. Therefore the scaling factor $\alpha_{\text{v1norm}} = 1.0$ had to be adjusted to compensate for the implementation difference. This was done simultaneously by setting $\alpha_{\text{filt}\to\text{rate},r} = 15\,\text{Hz}$, a scaling factor to map the unit-less filter responses at each scale r onto more meaningful mean firing rates, as will be explained below. In brief, we opted to reproduce the contrast sensitivity function reported for V1 cells projecting to MT (Movshon and Newsome, 1996). Other than that, the computation in Eq. 4.7 is conceptually equivalent to Eqs. 3–4 in Simoncelli and Heeger (1998).

4.2.3 V1 Complex Cells

V1 complex cell responses were computed as local weighted averages of simple cell responses,

$$C_{k,r}(x,y,t) = \alpha_{\text{v1comp}} \exp\left(\frac{-(x^2+y^2)}{2\sigma^2_{\text{v1comp}}}\right) w_{k,r,\vec{\psi}}\, S_{k,r}(x,y,t), \tag{4.8}$$

where the half-width of the Gaussian was $\sigma_{\text{v1comp}} = 1.6$, $\alpha_{\text{v1comp}} = 0.1$ was a scaling factor, and $w_{k,r,\vec{\psi}}$ was a weight value as follows.

In order to arrive at V1 complex cells that are tuned for a particular direction and speed of motion, filter responses needed to be "steered" into a specific direction. Freeman and Adelson (1991) showed that the response of an arbitrarily oriented filter could be synthesized from a fixed bank of basis filters (such as the 28 V1-like filters described above). Thus the projection weights from V1 simple cells to V1 complex cells, $w_{k,r,\vec{\psi}}$, could be interpolated as follows. Let $\vec{\psi} = (\psi_x, \psi_y, \psi_t)'$ be the unit vector parallel to an arbitrary space-time orientation (i.e., direction and speed of motion), akin to the unit vectors $\vec{u}_k$ in Eq. 4.5. Then we can write the third directional derivative in direction of $\vec{\psi}$ analogously to Eq. 4.5 as:

$$\frac{\partial^3 f_r}{\partial \vec{\psi}^3} = \left[\alpha_{\text{collapse}}\, \mathbf{M}^{-1}\right] \vec{b}_r \tag{4.9}$$

$$= \vec{w}_{\vec{\psi}}\, \vec{b}_r, \tag{4.10}$$

where the matrix $\mathbf{M}$ and the vector $\vec{b}_r$ were the same as in Eq. 4.6, each element of the vector $\vec{w}_{\vec{\psi}}$ was a number $6!/(X!\,Y!\,T!)\left(\psi_x\right)^X\left(\psi_y\right)^Y\left(\psi_t\right)^T$, and $'$ denoted vector transposition. The product $\left[\alpha_{\text{collapse}}\, \mathbf{M}^{-1}\right]$ thus was a set $\vec{w}_{\vec{\psi}} = (w_{\vec{\psi},1}, \ldots, w_{\vec{\psi},28})$ of interpolated weights, where the k-th element determined the strength of the projection from the k-th V1 simple cell onto a V1 complex cell. The two cells were connected only if they were in a local spatial neighborhood (Eq. 4.8). A V1 complex cell received projections from V1 simple cells at all three spatiotemporal resolutions, r.

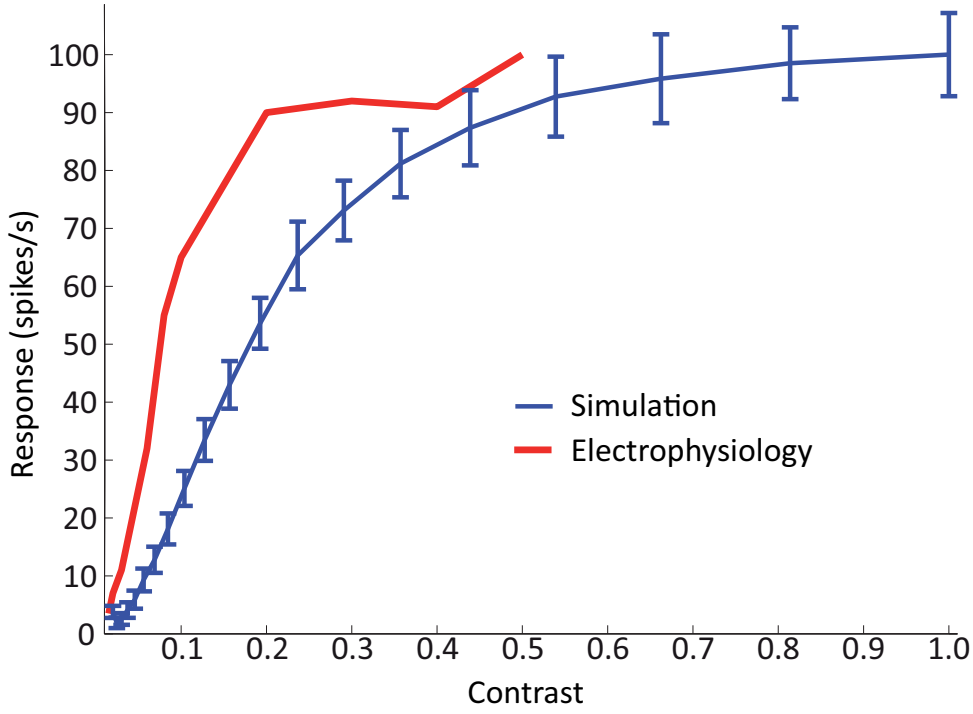

Figure 4.3: The contrast sensitivity function of model V1 simple cells (blue) is plotted against electrophysiological data adapted from Fig. 7 of Movshon and Newsome (1996). Each data point is a V1 mean response to a drifting grating, averaged over both 1 s of stimulus presentation and all neurons in the subpopulation. Vertical bars are the standard deviation on the population average.

4.2.4 Converting Filter Responses to Firing Rates

In order to find a meaningful mapping from unit-less filter responses to neuronal mean firing rates, we opted to reproduce the contrast sensitivity function reported for V1 cells projecting to MT (Movshon and Newsome, 1996), which is shown in Fig. 4.3. The red line is the electrophysiological data adapted from Fig. 7 of Movshon and Newsome (1996), whereas the blue line is our simulated data.

In order to arrive at this plot, we presented a drifting sinusoidal grating of varying contrast to V1 simple cells coding for scale $r = 0$, and computed their mean response S_{k0} from Eq. 4.7 over a stimulation period of 1 s. The drifting grating had a spatial frequency of $\omega_{\text{spat}} = 0.1205$ cycles per pixel and a temporal frequency of $\omega_{\text{temp}} = 0.1808$ cycles per frame.

Because the grating was drifting to the right, we only looked at the subpopulation of V1 simple cells that responded maximally to this stimulus (which was true for $k = 24$). The mean firing rate of neurons in this subpopulation, $S_{24,0}$, was then averaged over all cells in the subpopulation and plotted in Fig. 4.3 (blue curve) for $\alpha_{\text{v1norm}} = 1.0$ and $\alpha_{\text{filt}\to\text{rate},0} = 15\,\text{Hz}$. Vertical bars are the standard deviation on the population average. The scaling factor α_{v1norm} was gradually changed until the curvature of the blue graph approximated the curvature of the electrophysiological data. The scaling factor $\alpha_{\text{filt}\to\text{rate},0}$ was then adjusted such that the simulated responses saturated at approximately 100 Hz. In order to tune V1 simple cells at the other two scales, that is, $S_{k,1}$ and $S_{k,2}$ from Eq. 4.7, we used a RDK stimulus, which is depicted as the sample input in Fig. 4.2. We chose scaling factors that would give equal response magnitudes at all three scales in response to the RDK stimulus, which resulted in $\alpha_{\text{filt}\to\text{rate},1} = 17\,\text{Hz}$ and $\alpha_{\text{filt}\to\text{rect},2} = 11\,\text{Hz}$.

4.3 The Software

The software for this motion energy model is freely and publicly available under the MIT license, and can be obtained from `https://github.com/UCI-CARL/MotionEnergy`. In order to run the code, users need to have the NVIDIA CUDA Toolkit installed. The code also easily integrates with other software development efforts from the Cognitive Anteater Robotics Laboratory (CARL), such as `VisualStimulus`[1] and CARLsim (see Chapter 3).

[1] A MATLAB toolbox for generating, storing, and plotting 2D visual stimuli related to neuroscience such as sinusoidal gratings, plaids, random dot fields, and noise: `https://github.com/UCI-CARL/VisualStimulusToolbox`.

Listing 4.1: Motion Energy Python Interface

```
import numpy as np
import matplotlib.pyplot as plt
from motionenergy import MotionEnergy

# grayscale stimulus showing sine grating and plaid
file_str = 'inp_grating_plaid_gray_32x32x2400.dat'
num_stims = 2  # number of stimuli (grating, plaid)
num_frames_per_trial = 50  # number of trials per stimulus

file_stream, dim = read_input_header(file_str)
width, height, length = dim

ME = MotionEnergy(width, height, 1)
speed = 1.5  # pixels per frame

for stim in xrange(num_stims):
        for trial in xrange(length/num_frames_per_trial/num_stims):
                for _ in xrange(num_frames_per_trial):
                        # read new frame from file
                        frame = read_input_frame(file_stream, dim)

                        # calculate V1 complex cell responses
                        out = ME.calcV1complex(frame, speed)
                        out = out.astype(np.float32)

                        # plot, ...

file_stream.close()
```

4.3.1 User Interface

For ease of use, both a C/C++ and Python user interface are provided. Both interfaces are based on functionality from the same underlying CUDA code. Similar to the original C/MATLAB code by Simoncelli and Heeger (1998), a number of API calls are provided that streamline the work flow. Using methods such as `MotionEnergy.calcV1linear` and `MotionEnergy.calcV1complex`, it is possible to easily retrieve simulated neural activity at various stages of the model.

An example is shown in Listing 4.1, where an input file is parsed that contains 2400 frames of 32×32 pixel images depicting a sinusoidal gratings and plaids drifting in different directions. For convenience, two functions are provided that either read the header section of the input file (`read_input_header`, line 10), or retrieve the next frame in the file (`read_input_frame`, line 20) (source code not shown). An input frame is then converted to a V1 complex cell response with a single API call (line 23).

The resulting network activity is summarized as a tuning curve in Fig. 4.4. Here, polar plots of V1 direction tuning were calculated in response to a sinusoidal grating (Fig. 4.4A, B) and a plaid stimulus made of two superimposed sinusoidal gratings of equal contrast (Fig. 4.4C, D), where the angle denotes motion direction and the radius indicates neuronal activity (arbitrary units). The direction tuning is unimodal for gratings, but bimodal for plaids. This is due to V1 complex cells picking up the motion direction of the individual grating components (black arrows in Fig. 4.4C) instead of the overall direction of plaid motion (red arrow in Fig. 4.4C), which can only be inferred if the cell is able to solve the aperture problem.

4.3.2 Kernel

In order for our implementation to be useful to researchers already working with the S&H model, we tried to stay as close to the S&H C/MATLAB implementation as possible. However, there are a few minor differences worth mentioning. First, as explained in Section 4.2.2, we normalize V1 simple cell responses in a large Gaussian neighborhood rather than across the whole population. Second, whereas the S&H model deals with edge effects by temporarily "padding" the input image with an invisible border, we opted for the computationally more economical alternative to simply decrease the responses of V1 simple cells located close to image borders. Third, in the S&H C/MATLAB implementation there are two additional

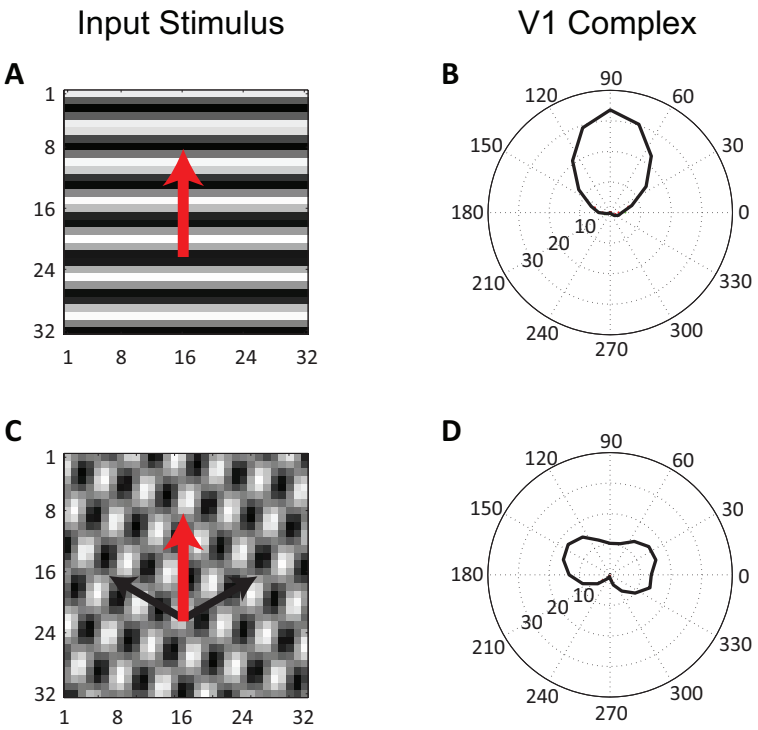

Figure 4.4: Polar plots of direction tuning for V1 complex cells in response to a sinusoidal grating (**A**, **B**) and a plaid stimulus (**C**, **D**) drifting upwards, where the angle denotes motion direction and the radius is the simulated neural activity (arbitrary units). The plaid stimulus was constructed from two superimposed sine wave gratings of equal contrast, drifting into two different directions (black arrows) 120° apart. The red arrow indicates the perceived direction of motion.

scaling factors (called `v1Blur` and `v1Complex`, with values 0.99 and 1.02, respectively) that we do not apply in order to save execution time. Fourth, our model processes input images at three different scales as described in Eqs. 4.1–4.3, which is a feature that is not implemented in the original S&H model. The most crucial mathematical operation in the V1 stage of the model is the convolution. Because the filter kernels used in our implementation are relatively small, employing the Fast Fourier Transform (FFT) would actually hurt performance. Instead we perform all convolution operations in the space-time domain using a custom function, which makes use of the fact that the Gaussian filter and its derivative are dimensionally separable. Future work could be directed towards further optimizing the convolution operation in CUDA.

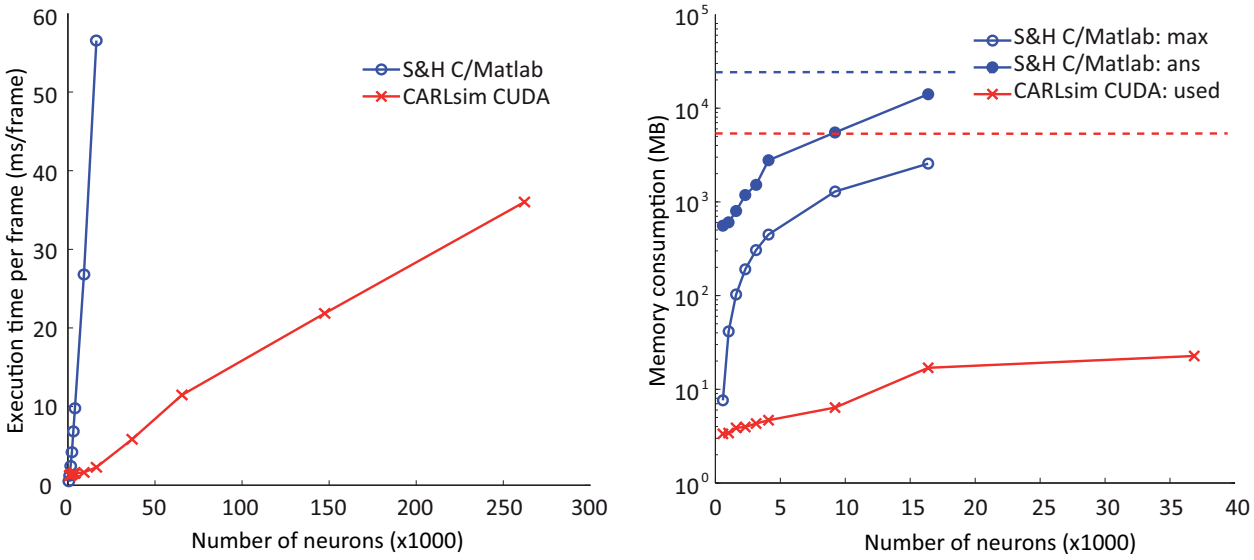

Figure 4.5: **A**, Execution time of a MATLAB implementation (blue) of V1 complex cells versus a CUDA implementation (red). **B**, Observed memory usage for the MATLAB implementation (blue) and CUDA implementation (red).

4.3.3 Computational Performance

In order to compare our CUDA implementation of V1 to the original, unmodified S&H implementation (which features code in both C and MATLAB) we computed V1 complex cell responses (see Section 4.2.3) at a single spatiotemporal scale to a drifting sinusoidal grating and recorded the model's execution time. The S&H C/MATLAB code was executed as `shModel(stim, pars, 'v1Complex')`, where `stim` was the input stimulus, and `pars` were the default parameters (`shPars`).

Fig. 4.5A shows the execution time per video frame for both models. Our GPU implementation (red) was not only faster (except for relatively small networks) than the S&H C/MATLAB implementation (blue), but it also scaled better with network size. Note that the C/MATLAB implementation was a single-threaded computation. The largest speedup, a factor of 12, was observed for a network consisting of $96 \times 96 = 9,216$ neurons. It is likely that even greater speedups could have been achieved on larger networks, but these networks could not run with the S&H C/MATLAB implementation because they ran out of

memory. Timing was performed using standard commands `tic` and `toc` in MATLAB, and the `<ctime>` function `time` in C++/CUDA. For the S&H C/MATLAB implementation, the time it took to create the stimulus was not included in the time measurement. On the other hand, in the CUDA implementation the stimulus had to be read from file frame-by-frame and copied to the GPU card. However, we did not include the time it takes to transfer the response back from the device to the host.

Additionally, the S&H C/MATLAB implementation is memory-intensive (see Fig. 4.5B), and execution times for networks above size $128 \times 128 = 16,384$ could not be computed because the CPU ran out of memory, even though we had a relatively large amount of Random-Access Memory (RAM) (24 GB) available. Measuring memory usage in MATLAB is not straight-forward. In order to demonstrate the excessive memory consumption of the S&H C/MATLAB implementation (see Fig. 4.5B) we opted to measure two metrics: the size of the output argument ans to function call `shModel` (blue, filled circle in Fig. 4.5B) and the maximum memory usage of the MATLAB process at any point in time (blue, open circle). The first was measured with native MATLAB command `whos`, and the latter was measured by running a bash script in the background that reported the memory usage of the process every second (using linux command `ps`). The blue dashed line is the 24 GB limit of the system's RAM. Note the log scale on the ordinate. Less memory was required to run the process than to store the output argument, which consisted of a matrix whose size was proportional to the product of the stimulus dimensions and the number of frames. A straightforward way of making the S&H C/MATLAB implementation capable of handling large inputs would thus be to break up the output argument into smaller chunks of data. On the other hand, the memory usage of the GPU implementation was significantly lower (red line in Fig. 4.5B) and scaled better with network size. We used the CUDA command `cuMemGetInfo` to identify the amount of allocated memory on the GPU. The red dashed line is the upper limit of GPU memory available to the user (roughly 5.2 GB on our card).

Chapter 5

Efficient Spiking Neural Network Model of Pattern Motion Selectivity

5.1 Introduction

In the last chapter I described an efficient implementation of the motion energy model (Simoncelli and Heeger, 1998), which is based on the observation that directionally selective neurons in V1 act as linear spatiotemporal filters. The resulting V1 complex cells were tuned to a specific direction and speed of motion, but when presented with a more complicated pattern such as a plaid stimulus, they were unable to resolve local motion ambiguities (see Section 2.1.3) and signal the overall direction of pattern motion.

In this chapter I turn to the question of how V1-like motion signals can be integrated in order to solve the aperture problem. Conceptually speaking, this computation can be performed using the Intersection-Of-Constraints (IOC) principle (see Section 2.1.3.1). Rust et al. (2006) revealed a procedure that could implement an IOC solution to the aperture problem (see Section 2.1.3.1) in three consecutive steps: 1) spatial pooling over V1 or MT

CDS cells with a wide range of preferred directions, 2) strong motion opponent suppression, and 3) a tuned normalization that may reflect center-surround interactions in MT. Whereas the implementation by Rust et al. (2006) was restricted to simple firing-rate neurons (see Section 2.3.1) and inputs that are mixtures of sinusoidal gratings of a fixed spatial and temporal frequency, it remains to be demonstrated how these computations can be performed in a network of spiking neurons (Izhikevich, 2003) and conductance-based synapses that respect the detailed temporal dynamics of neuronal and synaptic integration of biological neurons.

5.2 Methods

In the following section I describe a two-stage SNN model of MT that can not only solve the aperture problem, but also be used to make perceptual decisions about ambiguous visual motion stimuli.

5.2.1 Two-Stage Spiking Model of MT

The two-stage model of MT is based on the idea that Component-Direction-Selective (CDS) cells represent an earlier stage of motion processing than Pattern-Direction-Selective (PDS) cells (Movshon et al., 1985; Smith et al., 2005) (see Section 2.1.3.1), making MT CDS cells similar in terms of direction and speed tuning to the model V1 complex cells used by Simoncelli and Heeger (1998). In fact, it has been shown that some MT cells exhibit speed tuning characteristics similar to V1 complex cells (Priebe et al., 2006), which has led to the suggestion that speed tuning in MT might be inherited from V1. Livingstone and Conway (2007) have shown that even some V1 simple cells are speed-tuned in macaque. Whereas CDS cells give responses whose selectivity is stable and consistent from the time they are

first activated, PDS cells often respond with different and broader selectivity when first activated, sometimes even resembling CDS cells, and only over a time-course on the order of 100 ms do they establish pattern selectivity (Smith et al., 2005). At least in anesthetized monkeys, MT is believed to consist of roughly 40 % CDS cells, 25 % PDS cells, and 35 % unclassified cells (Movshon et al., 1985). However, in awake animals the situation might be more complicated (Pack et al., 2001). All cells in MT were Izhikevich spiking neurons, whose membrane potential was thus described by a pair of coupled differential equations (see Section 2.3.2.1).

5.2.1.1 Component-Direction-Selective (CDS) Cells

Component-Direction-Selective (CDS) cells are selective to a particular direction and speed of motion (an orientation in space-time). The name is an indication that these cells, when presented with a plaid stimulus consisting of two superimposed sinusoidal gratings, preferably respond to the motion of each grating (component) rather than the global motion pattern produced by the combination of the two gratings (Movshon et al., 1985).

MT CDS cells had the same direction and speed preferences as V1 cells; that is, they preferentially responded to motion in one of eight different directions (in 45° increments) and three different speeds (1.5 pixels per frame, 0.125 pixels per frame, and 9 pixels per frame) at any pixel location. The only difference was that MT cells had a larger receptive field, achieved by Gaussian spatial pooling.

In order to model response normalization equivalent to the one in Eq. 6 of Simoncelli and Heeger (1998), we introduced a pool of inhibitory interneurons, which integrated the activity of all MT CDS cells within a large Gaussian neighborhood (across direction and speed), and projected back to all three pools of MT CDS cells with one-to-one connections. This response normalization is important to qualitatively reproduce the speed tuning curves (see

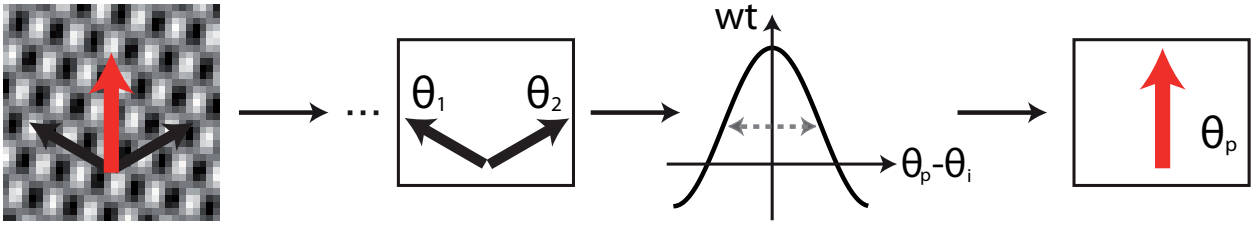

Figure 5.1: PDS cells can be constructed in three steps: 1) spatial pooling over MT CDS cells with a wide range of preferred directions, 2) strong motion opponent suppression, and 3) a tuned normalization that may reflect center-surround interactions in MT (Rust et al., 2006).

Section 5.3.2).

5.2.1.2 Pattern-Direction-Selective (PDS) Cells

Pattern-Direction-Selective (PDS) cells differ from CDS cells in that they, when presented with a plaid stimulus consisting of two superimposed sine gratings, preferentially respond to the overall motion direction, not the individual components (Movshon et al., 1985). Because visual stimuli typically contain many oriented components, local motion measurements must be appropriately combined in order to sense the true global motion of the stimulus (aperture problem). Thus it has been suggested that PDS neurons reflect a higher-order computation that acts on V1 or MT CDS afferents (Movshon et al., 1985). MT PDS cells in our model received direct input from CDS cells, and thus conserved their speed and direction preferences.

Pooling over MT CDS cells and opponent suppression were implemented by pooling CDS responses across spatial position and across direction preference (see Fig. 5.1), such that the strength of a projection from a CDS cell selective to motion direction θ_{CDS} at location $(x_{\text{CDS}}, y_{\text{CDS}})$ to a PDS cell selective to motion direction θ_{PDS} at location $(x_{\text{PDS}}, y_{\text{PDS}})$ can be

expressed as:

$$w_{\text{CDS}\to\text{PDS}} = \alpha_{\text{CDS}\to\text{PDS}} \cos(\Delta\theta) \exp\left(\frac{-\Big((\Delta x)^2 + (\Delta y)^2\Big)}{2\sigma^2_{\text{PDS,pool}}}\right), \tag{5.1}$$

where $\Delta\theta = \theta_{\text{PDS}} - \theta_{\text{CDS}}$, $\Delta x = x_{\text{PDS}} - x_{\text{CDS}}$, $\Delta y = y_{\text{PDS}} - y_{\text{CDS}}$, the half-width of the Gaussian neighborhood $\sigma_{\text{PDS,pool}} = 3$ pixels, and $\alpha_{\text{CDS}\to\text{PDS}}$ is a scaling factor. If the resulting weight was negative, due to $|\Delta\theta| > \pi/2$, the projection was relayed to a population of inhibitory interneurons. Following the reasoning of Rust et al. (2006), the pattern index of a MT cell can be reduced simply by sharpening the cosine tuning component in Fig. 5.1 (see third column of Fig. 6 in Rust et al. (2006)).

Tuned normalization was implemented by an inhibitory self-connection with a narrowly tuned Gaussian across direction (see second column of Fig. 6 in Rust et al. (2006)). Analogous to previous projections, this was implemented by relaying the inhibitory projection to a pool of inhibitory interneurons:

$$w_{\text{PDS}\to\text{PDS,inh}} = \exp\left(\frac{-(\Delta\theta)^2}{2\sigma^2_{\text{PDS,tuned,dir}}}\right) \exp\left(\frac{-\Big((\Delta x)^2 + (\Delta y)^2\Big)}{2\sigma^2_{\text{PDS,tuned,loc}}}\right), \tag{5.2}$$

where $\sigma_{\text{PDS,tuned,dir}} < 45^\circ$ (such that only one of the eight subpopulations was activated), $\sigma_{\text{PDS,tuned,loc}} = 2$ pixels, and the inhibitory population sent one-to-one connections back to the pool of MT PDS cells.

5.2.2 Spiking Layer of LIP Decision Neurons

A layer of decision neurons was responsible for integrating over time the direction-specific sensory information that is encoded by the responses of MT PDS cells. This information was then used to make a perceptual decision about the presented visual stimulus, such as determining the global drift direction of a field of random moving dots in a motion discrimination task (presented in Section 5.3.3). A good candidate for such an integrator area

in macaques might be Lateral Intraparietal area (LIP), where neurons have been found whose firing rate are predictive of the behavioral RT in a motion discrimination task (Roitman and Shadlen, 2002; Shadlen and Newsome, 2001). Spiking neurons in a simulated LIP area were grouped into eight pools of 50 neurons, each pool receiving projections from exactly one of the eight pools of MT PDS cells with 10 % connection probability. As a result of this connectivity profile, each pool of decision neurons accumulated sensory evidence for a particular direction of motion, based on the response of MT PDS cells. Additionally, each decision pool received inhibitory projections from other decision pools if the two preferred directions of motion were close to opposite. More precisely, a decision neuron in pool i (thus selective to direction θ_i) received an inhibitory projection from neurons in pool j (selective to direction θ_j) with strength:

$$w_{\text{dec,inh}\to\text{dec}} = \cos(\theta_i - \theta_j + \pi), \tag{5.3}$$

and 10 % connection probability.

LIP did not employ any internal noise.

5.3 Results

We conducted a number of experiments to ensure the accuracy and efficiency of our implementation. Here we demonstrate that the network is able to exhibit direction and speed tuning for drifting bar and plaid stimuli that are in agreement with neurophysiological recordings, and that the network qualitatively reproduces both the psychometric and chronometric function in a Two-Alternative Forced Choice (2AFC) motion discrimination task. Additionally, we measured both the computational performance and memory consumption of our model and compared it to the S&H C/MATLAB implementation.

GPU simulations were run on a NVIDIA Tesla M2090 (6 GB of memory) using CUDA, and

CPU simulations (including Matlab) were run on an Intel Xeon X5675 at 3.07 GHz (24 GB of RAM). The same exact network running on a single GPU produced all results; the only difference per experiment was the presented input stimulus. The full network consisted of $153,216$ neurons and approximately 33 million synapses, which corresponds to a 32×32 pixels input resolution.

5.3.1 Direction Tuning

We tested the ability of our model MT cells to signal the direction of motion for drifting grating and plaid stimuli. Responses were simulated for CDS cells and PDS cells in MT. The first stimulus was a drifting sinusoidal grating consisting of spatial and temporal frequency components that were preferred by MT neurons selective to a speed of 1.5 pixels per frame (that is, $\omega_{\text{spat}} = 0.1205$ cycles per pixel, $\omega_{\text{temp}} = 0.1808$ cycles per frame). The second stimulus was a pair of superimposed gratings drifting in a direction orthogonal to their orientation, which together formed a coherently drifting plaid pattern. The two gratings both had the same spatial frequency ω_{spat}, but their orientation and drift direction differed by 120°. The direction of these particular patterns lay equidistant between the directions of motion of the two component gratings. The stimulus contrast for both grating and plaid was 30 %.

Our model was able to reproduce direction tuning curves that are in agreement with single-cell electrophysiological data (Movshon and Newsome, 1996; Movshon et al., 1985; Rodman and Albright, 1989) for V1 cells, MT CDS cells, and MT PDS cells. Fig. 5.2 shows polar plots of direction tuning for V1 neurons (Panels B and F), MT CDS cells (Panels C and G), and MT PDS cells (Panels d and h), where the angle denotes motion direction and the radius is the firing rate in spikes per second (compare also Fig. 9 in Simoncelli and Heeger (1998) and Fig. 1 in Rust et al. (2006)). Tuning curves were obtained by calculating the

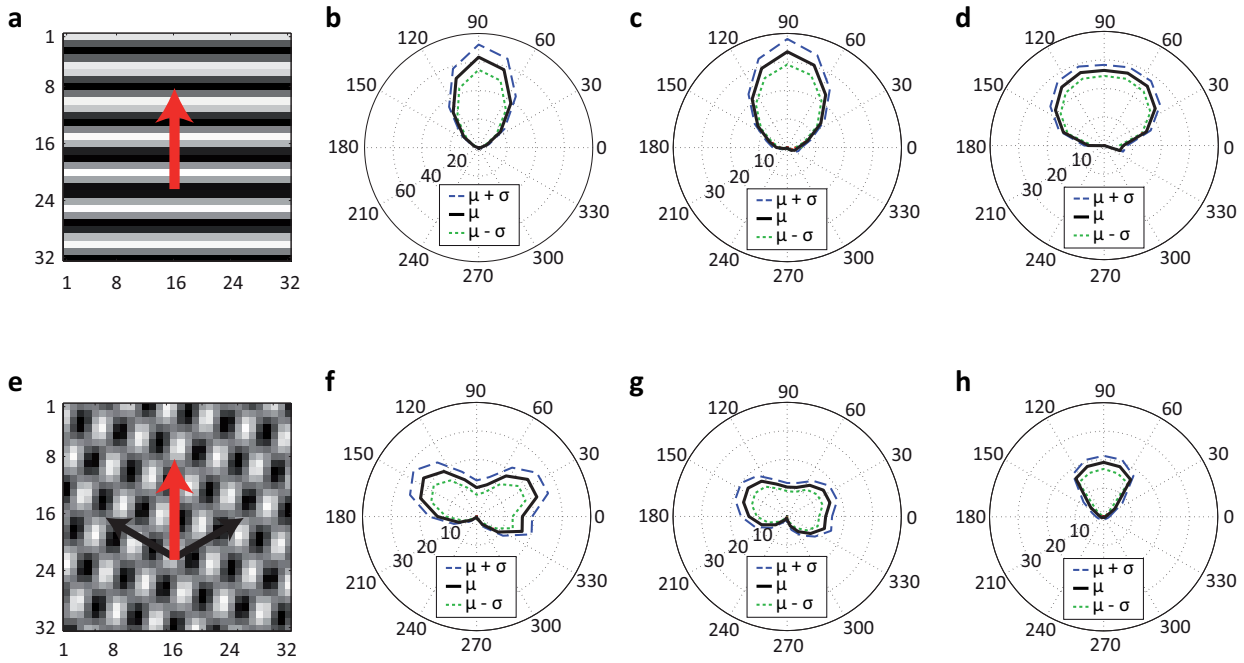

Figure 5.2: Polar plots of direction tuning for a sinusoidal grating **a**–**d** and a plaid stimulus **e**–**h** drifting upwards, where the angle denotes motion direction and the radius is the firing rate in spikes per second. Tuning curves were obtained by taking the mean firing rate of a neuron to a drifting grating during 2 s of stimulus presentation, averaged over all neurons in the population selective to the same stimulus direction (black: mean neuronal response, blue: mean plus standard deviation on the population average, green: mean minus standard deviation). Shown are mean responses for V1 complex cells (**b** and **f**), MT CDS cells (**c** and **g**), and MT PDS cells (**d** and **h**). Only MT PDS cells h responded to the motion of the entire plaid pattern rather than to the motions of the individual component gratings.

mean firing rate of a neuron's response to a drifting grating during two seconds of stimulus presentation. These responses were averaged over all neurons in the population selective to the same direction of motion (black: mean neuronal response, blue: mean plus standard deviation on the population average, green: mean minus standard deviation). As a result of suppressing edge effects, neurons that coded for locations closer than five pixels from the image border were only weakly activated, and were thus excluded from the plot. The tuning curves in the top row were generated in response to the sinusoidal grating drifting upwards, which is illustrated in Panel A. Analogously, the tuning curves in the bottom row were generated in response to the plaid stimulus drifting upwards, which is illustrated in Panel E (red arrow: pattern motion direction, black arrows: motion direction of the grating

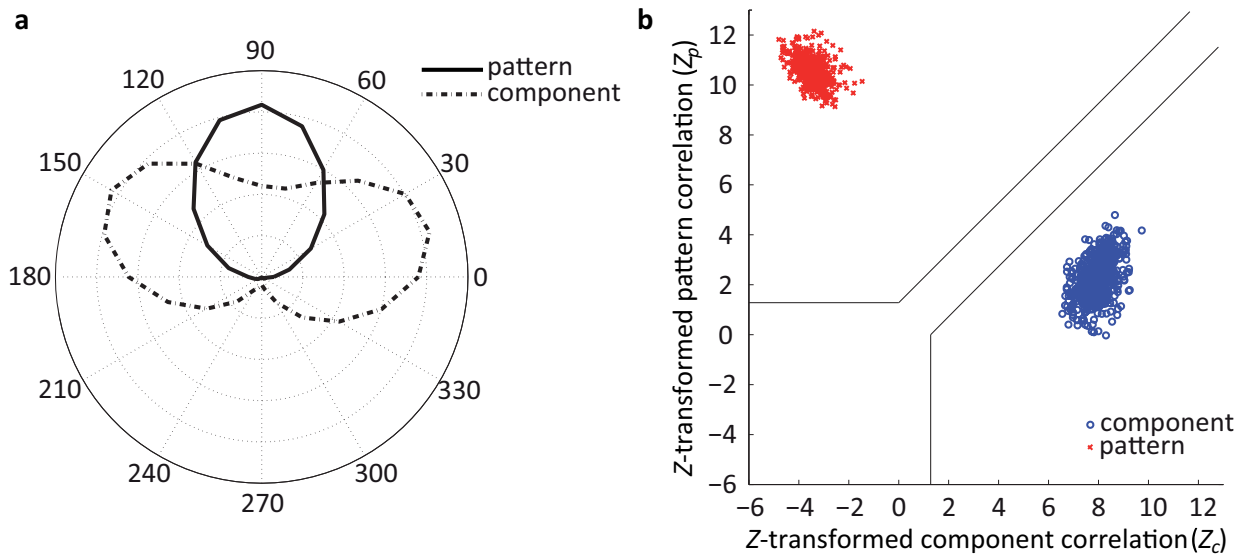

Figure 5.3: The pattern index is computed for all MT CDS cells (blue) and all MT PDS cells (red), and plotted as a Fisher Z-score. The black solid lines are the classification region boundaries, indicating that all MT CDS cells have indeed been classified as component-selective, and all MT PDS cells have been classified as pattern-selective.

components). The direction tuning curve for gratings is unimodal for all three neuron classes, but the direction tuning curve for plaids shows two distinct lobes for V1 complex cells (Panel F) and MT CDS cells (Panel G). Each lobe corresponds to one of the component gratings of the plaid. Only MT PDS cells (Panel H) responded to the motion of the entire plaid pattern rather than to the motions of the individual component gratings.

In order to quantify the pattern selectivity of our model PDS cells, we computed the pattern index for each CDS and PDS cell (see Fig. 5.3) using the standard technique (Movshon and Newsome, 1996; Movshon et al., 1985; Smith et al., 2005). Based on the tuning curve for the drifting grating described above, we generated two predictions for each cell's tuning curve to drifting plaids (Fig. 5.3A); either the cell would respond to the plaid in the same way as it responded to the grating ("pattern" prediction, black solid line), or it would respond independently to the two grating components ("component" prediction, black dashed line). We then computed the correlation (r_c, r_p) between the cell's actual response to a plaid stimulus and the component and pattern predictions. To remove the influence of correlations

between the predictions themselves, we calculated partial correlations R_c and R_p for the component and pattern predictions, respectively, using the standard formulas:

$$R_c = \frac{r_c - r_p r_{pc}}{\sqrt{(1 - r_p^2)(1 - r_{pc}^2)}}, \tag{5.4}$$

$$R_p = \frac{r_p - r_c r_{pc}}{\sqrt{(1 - r_c^2)(1 - r_{pc}^2)}}, \tag{5.5}$$

where r_c and r_p are the simple correlations between the data and the component and pattern predictions, respectively, and r_{pc} is the simple correlation between the predictions (Movshon and Newsome, 1996). Because the sampling distribution of Pearson's r is not normal, we converted the correlation measures R_c and R_p to a Fisher Z-score,

$$Z_c = \frac{0.5 \ln\left(\frac{1+R_c}{1-R_c}\right)}{\sqrt{\frac{1}{df}}} = \frac{\operatorname{arctanh}(R_c)}{\sqrt{\frac{1}{df}}}, \tag{5.6}$$

$$Z_p = \frac{\operatorname{arctanh}(R_p)}{\sqrt{\frac{1}{df}}}, \tag{5.7}$$

where the numerator is the Fisher r-to-Z transformation and df is the degrees of freedom, equal to the number of values in the tuning curve (in our case 24) minus three (Smith et al., 2005). The Z-scores of all CDS and PDS cells (excluding neurons coding for locations closer than five pixels from the image border) in the network are plotted in Fig. 5.3B. Each value of Z_c and Z_p was tested for significance using a criterion of 1.28, which is equivalent to $p = 0.90$ (Smith et al., 2005). For a PDS cell (red) to be judged as pattern-selective, the value of Z_p had to exceed the value of Z_c by a minimum of 1.28 (black solid lines). All PDS cells in Fig. 5.3B met this criterion and, therefore, were indeed pattern-selective. Analogously, all CDS cells (blue) could be judged as component-selective.

5.3.2 Speed Tuning

We next considered the ability of our implementation to reproduce MT speed tuning curves as demonstrated in Simoncelli and Heeger (1998). MT neurons have been divided into three distinct classes based on their speed tuning properties (Rodman and Albright, 1987). The first class of neurons is relatively sharply tuned for a particular speed and direction of motion ("speed-tuned" or "band-pass"). This class of neurons is also strongly suppressed by motion in the anti-preferred (opposite) direction; the suppression is strongest when the stimulus moves in the opposite direction at roughly the preferred speed. The second class of neurons prefers low speeds in both the preferred and anti-preferred direction ("low-pass"). The third class responds to high speed stimuli in both directions ("high-pass").

As shown in Fig. 5.4, our implementation faithfully reproduces the speed tuning characteristics of these three distinct classes (compare also Fig. 10 in Simoncelli and Heeger (1998)). The stimulus consisted of a single bar drifting over the entire visual field either to the right (preferred direction) or to the left (anti-preferred direction) at different speeds. Each data point is the mean firing rate of a particular MT CDS neuron located near the center of the visual field, averaged over the time course of a specific speed and direction configuration. The relatively low mean firing rates can be explained by the fact that the stimulus resides outside the neuron's receptive field for most of the time. The first neuron class (Panel a, "band-pass") preferentially responded to a bar moving at 1.5 pixels per frame to the right, and was strongly suppressed when the bar moved at the same speed to the left. The second neuron class (Panel b, "low-pass") exhibited a preference for low speeds (0.125 pixels per frame) in both directions. With increasing speed the response of the neuron to dots moving in the anti-preferred direction weakened. This behavior can be explained by the fact that the Fourier planes corresponding to low speed motions in opposite directions are both close to the $\omega_t = 0$ plane, and thus close to each other (Simoncelli and Heeger, 1998). Also, this class of neurons was suppressed by fast stimuli moving in either direction. Similarly, the

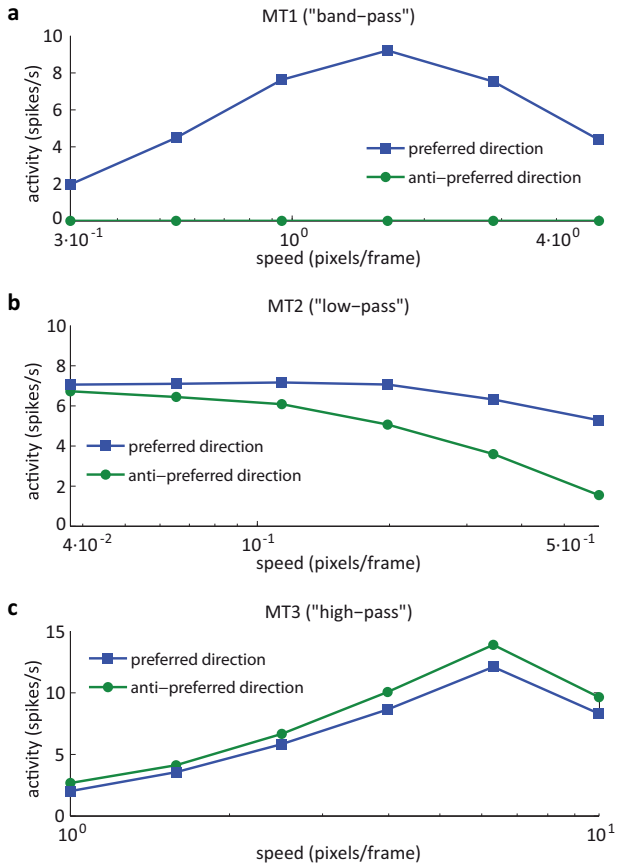

Figure 5.4: Speed tuning curves for three different classes of MT neurons. The stimulus consisted of a single bar drifting over the entire visual field either to the right (preferred direction) or to the left (anti-preferred direction) at different speeds. **A**, Response of a "speed-tuned" neuron (selective to motion at 1.5 pixels per frame). **B**, Response of a "low-pass" neuron (selective to motion at 0.125 pixels per frame). **C**, Response of a "high-pass" neuron (selective to motion at 9 pixels per frame).

third neuron class (Panel c, "high-pass"), which had a high preferred speed (9 pixels per frame) in one direction, was excited by fast stimuli moving in the opposite direction, but was suppressed by slow stimuli moving in either direction.

5.3.3 Random Dot Kinematogram

In order to compare the performance of the model with behavioral data from 2AFC motion discrimination tasks, we developed a paradigm equivalent to the RDK experiments

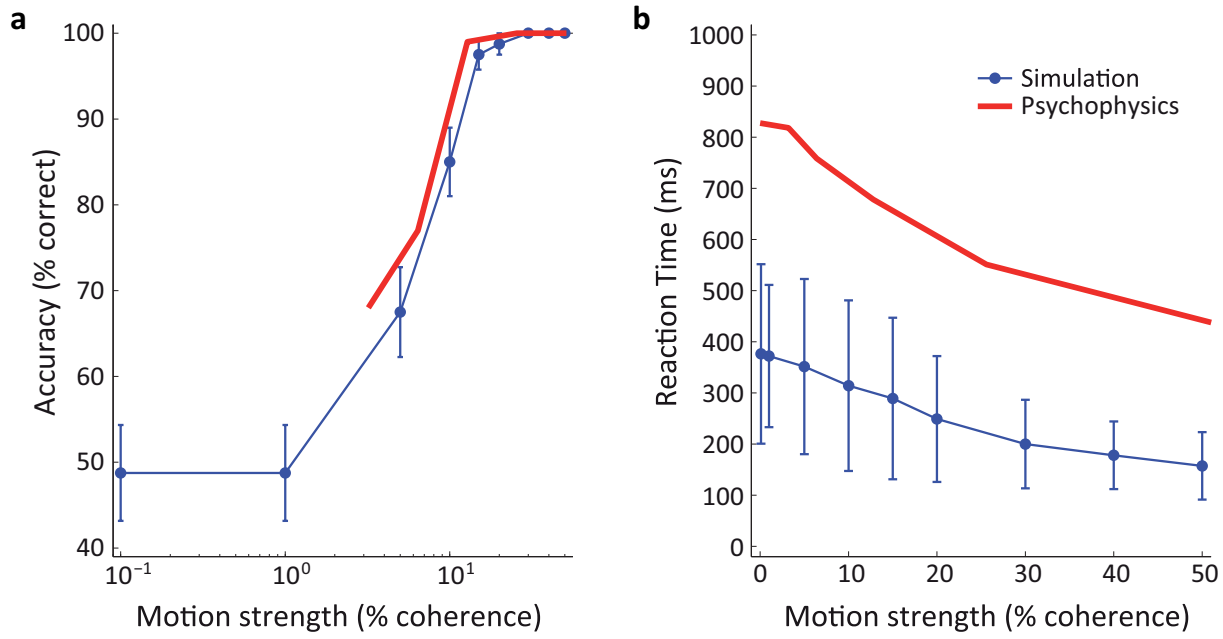

Figure 5.5: Random Dot Kinematogram (RDK). The RDK stimulus was constructed of approximately 150 dots (15 % dot density, maximum stimulus contrast) on a 32×32 input movie. **a**, Psychometric function. The network's accuracy increased with increasing motion strength (coherence level). **b**, Chronometric function. The network's RT decreased with increasing motion strength.

performed with monkeys and humans (Resulaj et al., 2009; Roitman and Shadlen, 2002). We constructed a simple decision criterion based on the race model (Shadlen and Newsome, 2001; Smith and Ratcliff, 2004), in which eight pools of decision neurons (one for each of the directions of motion, 50 neurons per pool) sum the responses of MT PDS cells selective to a particular direction and speed of motion. The first decision pool to emit 500 spikes (on average ten spikes per neuron) "won the race" and thus signaled a choice for that direction. A correct decision was the event in which the winning decision pool was selective to the actual motion direction of the stimulus. The time it took the network to reach the decision threshold was termed the RT.

The RDK stimulus was constructed of approximately 150 dots (15 % dot density, maximum stimulus contrast) on a 32×32 input movie. Each stimulus frame was presented to the network for 50 ms. A trial consisted of 20 stimulus frames of a particular motion direction

and coherence level. Motion coherence in the stimulus was varied between 0 and 50 %. Coherently moving dots drifted in one of eight possible directions, in 45° increments, at a speed of 1.5 pixels per frame. Note that, therefore, only MT PDS cells that were selective to this particular stimulus speed were connected to the decision layer.

Choice accuracy and RT as a function of task difficulty (coherence of dot motion) are shown in Fig. 5.5 (Panel a and b, respectively), where the thick red lines are human behavioral data extracted from a RT experiment (see Fig. 3 and Table 2 in Roitman and Shadlen (2002)) and simulated data is shown in blue. Each data point (blue) is the mean outcome of 80 trials (fixed coherence level, ten repetitions per motion direction), and the vertical bars are the standard error and standard deviation for accuracy (Panel a) and RT (Panel b), respectively. As in Fig. 3 in Roitman and Shadlen (2002), we did not show RTs on error trials. Our network performance is comparable to human accuracy, and it qualitatively emulates the effect of motion strength on RT. Decreasing RT for a relatively easy task (e.g., high motion coherence) is a direct consequence of the race model. Conversely, when the difficulty of a decision is high (e.g., low coherence level), information favoring a particular response grows more slowly (Smith and Ratcliff, 2004), and the probability of making an error is higher (Shadlen and Newsome, 2001). The quantitative difference between behavioral and simulated RT in Fig. 5.5 could be eradicated by fine-tuning the excitatory weights from MT cells to the decision layer. However, such an exercise would be meaningless, because our model does not take into consideration neural areas involved in characteristics of the decision-making process that influence the length of RT, such as the time-course of LIP neuronal dynamics or the gating of saccadic eye movements (Shadlen and Newsome, 2001), which have been successfully modeled in detail by others (Grossberg and Pilly, 2008).

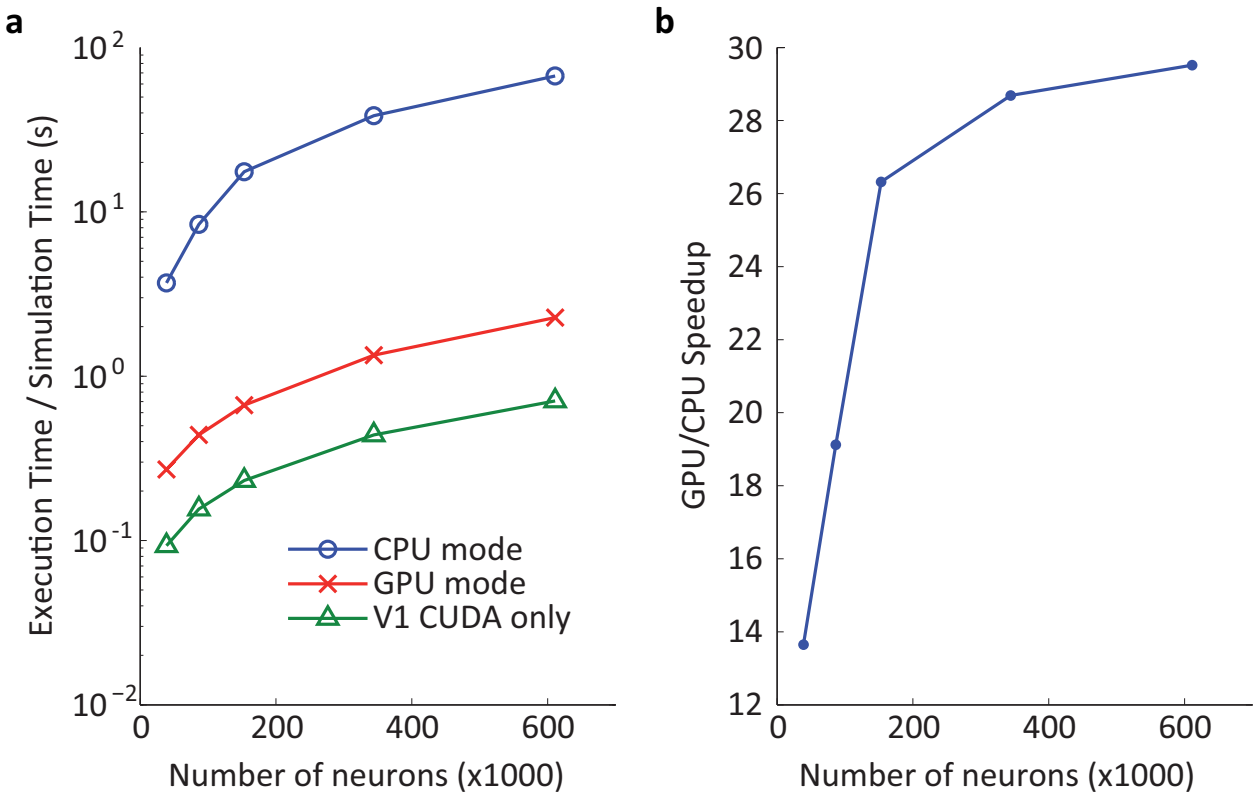

Figure 5.6: **a**, Simulation speed is given as the ratio of execution time over the simulation time for networks run in CPU mode (blue) and GPU mode (red). In both cases, the V1 CUDA implementation was executed (green), which is part of the total simulation time (in blue and red). Note the log scale on the ordinate. The GPU simulations did not only run faster, but simulation speed scaled better with network size. **b**, Speedup is given as the ratio of CPU execution time over GPU execution time.

5.3.4 Computational Performance

Comparing the performance between GPU simulation mode and CPU simulation mode with the full network on the specific processor remains to be demonstrated. In GPU mode all data structures are allocated on the GPU, whereas in CPU mode the network would be allocated on the CPU's memory, and only the generation of motion energy responses (written in CUDA) would be delegated to the GPU. Hence we evaluated the computational performance by running the full network in both CPU and GPU mode with input images from 16×16 pixels ($38,784$ neurons) to 64×64 pixels ($610,944$ neurons).

The result is shown in Fig. 5.6. The simulation speed is given as the ratio of execution time over the simulation time (see Fig. 5.6a) for networks run in CPU mode (blue) and GPU mode (red). Note that in both modes, the V1 CUDA implementation was executed (green), whose run-time is part of the total simulation time (in blue and red). The GPU simulations not only ran faster, but also simulation speed scaled better with network size. Note that the CPU simulation was a single-threaded computation. The full network at 40×40 input resolution ($239,040$ neurons) ran in real-time on the GPU. At 32×32 input resolution ($153,216$ neurons) the simulation was 1.5 times faster than real time. This result compares favorably with previous releases of our simulator (Nageswaran et al., 2009; Richert et al., 2011), which is partly due to code-level optimizations, but mostly due to differences in GPU hardware and the V1 stage of the network being spatiotemporal filters instead of spiking neurons. Speedup was computed as the ratio of CPU to GPU execution time. As the network size increased, the GPU simulations showed a significant speedup over the CPU (see Fig. 5.6b). The largest network we could fit on a single GPU roughly corresponded to 64×64 input resolution ($610,944$ neurons), which ran approximately 30 times faster than on the CPU. Larger networks currently do not fit on a single GPU and as such must be run on the CPU, which would be more than 70 times slower than real time judging from Fig. 5.6a.

5.4 Discussion

We presented a large-scale spiking model of visual area MT that i) is capable of exhibiting both component and pattern motion selectivity, ii) generates speed tuning curves that are in agreement with electrophysiological data, iii) reproduces behavioral responses from a 2AFC task, iv) outperforms a previous rate-based implementation of the motion energy model (Simoncelli and Heeger, 1998) in terms of computational speed and memory usage, v) is implemented on a publicly available SNN simulator that allows for real-time execution

on off-the-shelf GPUs, and vi) is comprised of a neuron model, synapse model, and AER, which is compatible with recent neuromorphic hardware, such as HRL (Srinivasa and Cruz-Albrecht, 2012), IBM (Cassidy et al., 2014), NeuroGrid (Boahen, 2006), SpiNNaker (Khan et al., 2008), and BrainScaleS (Schemmel et al., 2010).

The model is based on two previous models of motion processing in MT (Rust et al., 2006; Simoncelli and Heeger, 1998), but differs from these models in several ways. First, our model contains the tuned normalization in the MT stage that was not present in Simoncelli and Heeger (1998) but introduced by Rust et al. (2006). Second, the implementation by Rust et al. (2006) was restricted to inputs that are mixtures of 12 sinusoidal gratings of a fixed spatial and temporal frequency, whereas our model can operate on any spatiotemporal image intensity. Third, MT PDS cells in our model sum over inputs from MT CDS cells as opposed to inputs from V1 cells, although the two approaches are conceptually equivalent. Fourth, instead of using linear summation and a static nonlinear transformation, all neuronal and synaptic dynamics in our model MT were achieved using Izhikevich spiking neurons and conductance-based synapses.

One could argue that the inclusion of Izhikevich spiking neurons and conductance-based synapses is unnecessary, since previous incarnations of the motion energy model did not feature these mechanisms yet were perfectly capable of reproducing speed tuning and motion selectivity. However, our approach is to be understood as a first step towards modeling large-scale networks of visual motion processing in more biological detail, with the ultimate goal of understanding how the brain solves the aperture problem, among other open issues in motion perception. Integrating the functionality demonstrated in previous models with more neurobiologically plausible neuronal and synaptic dynamics is a necessary first step for analyzing the temporal dynamics of model neurons in MT, which may i) help to explain how MT PDS cell establish their pattern selectivity not instantly but over a time-course on the order of 100 ms (Smith et al., 2005), and ii) enable the addition of spike-based learning rules

such as STDP; both of which might be harder to achieve with previous model incarnations. Additionally, the introduction of the present neuron model, synapse model, and AER did not affect performance, yet enabled the integration of the S&H model with recent neuromorphic hardware (Srinivasa and Cruz-Albrecht, 2012) (see also Section 5.4.3).

On the other hand, it is possible (if not likely) that some response dynamics produced by the neural circuitry in the retina, the LGN, and V1 may account for certain response properties of neurons in MT that the current model is not yet able to capture. Thus future work could be directed towards implementing the entire early visual system in the spiking domain. However, for the purpose of this study we deem a rate-based preprocessor to be an adequate abstraction, as the core functionality of directionally selective cells in V1 seem to be well-characterized by local motion energy filters (Adelson and Bergen, 1985; DeAngelis et al., 1993; Movshon and Newsome, 1996).

5.4.1 Neurophysiological Evidence and Model Alternatives

There is evidence that MT firing rates represent the velocity of moving objects using the IOC principle. A psychophysical study showed that the perception of moving plaids depends on conditions that specifically affect the detection of individual grating velocities (Adelson and Movshon, 1982). This is consistent with a two-stage model in which component velocities are first detected and then pooled to compute pattern velocity. Subsequent physiological studies broadly support such a cascade model (Perrone and Thiele, 2001; Rust et al., 2006; Smith et al., 2005).

However, other psychophysical results exist where the perceived direction of plaid motion deviates significantly from the IOC direction (Burke and Wenderoth, 1993; Ferrera and Wilson, 1990). Alternatives to the IOC principle are, for example, Vector Average (VA) or feature tracking. VA predicts that the perceived pattern motion is the vector average of the

component velocity vectors. Blob or feature tracking is the process of locating something (a "feature") that does not suffer from the aperture problem, such as a bright spot or a T-junction, and tracking it over time (Wilson et al., 1992). Ultimately, one needs to consider the interactions of the motion pathway with form mechanisms (Majaj et al., 2007), and model the processing of more complex stimuli (e.g., motion transparency, additional self-motion, multiple moving objects) (Layton et al., 2012; Raudies et al., 2011). Clarifying by which rule (or combination of rules) the brain integrates motion signals is still a field of ongoing research. For recent reviews on the topic see Bradley and Goyal (2008); Nishida (2011).

Although clear evidence for spatiotemporal frequency inseparability in MT neurons has been found (Perrone and Thiele, 2001), which supports the idea of a motion energy model, later studies reported it to be a weak effect (Priebe et al., 2003, 2006). The actual proportion of neurons in the primate visual system that are tuned to spatiotemporal frequency is currently not known.

5.4.2 Model Limitations

Although our model is able to capture many attributes of motion selectivity (e.g., direction selectivity, speed tuning, component and pattern motion), it is not yet complete for the following reasons. First, it does not explicitly specify the exact pattern velocity, but instead reports an activity distribution over the population of MT neurons, whose firing rates are indicative of the observed pattern motion. In order to estimate the speed of a target stimulus, it has been proposed to use a suitable population decoding mechanism that operates on MT responses (Hohl et al., 2013; Perrone, 2012). Second, our model does not attempt to predict the temporal dynamics of MT PDS cells, which often respond with broad selectivity when first activated, sometimes even resembling CDS cells, and only over a time-course on the

order of 100 ms establish their pattern motion selectivity (Smith et al., 2005). A possible explanation for these temporal dynamics is given in Chey et al. (1997). Third, it does not consider the visual form pathway and abstracts early visual details that may be critical for operation in natural settings. Fourth, the extent to which each stage in the motion energy model can be mapped onto specific neuronal populations is rather limited. Tiling the spatiotemporal frequency space according to the motion energy model is biologically implausible, and the temporal extent of the filters is unrealistically long (especially the low speed filters). However, a way to combine spatiotemporal filters based on V1 neuron properties into a pattern motion detector has been proposed in Perrone and Thiele (2002).

Another more fundamental limitation is that the S&H model (or for that matter, any spatiotemporal-energy based model including the elaborated Reichardt detector) can only sense so-called first-order motion, which is defined as spatiotemporal variations in image intensity (first-order image statistics) that give rise to a Fourier spectrum. Second-order stimuli, such as the motion of a contrast modulation over a texture, are non-Fourier and thus invisible to the model, yet can be readily perceived by humans (Chubb and Sperling, 1988). In addition, the existence of a third motion channel has been suggested, which is supposed to operate through selective attention and saliency maps (Lu and Sperling, 1995). Also, MT has been shown to be involved in color-based motion perception (Thiele et al., 2001).

There is also a plainly technical limitation to our model, which is manifested in the amount of available GPU memory. Due to their size, large-scale spiking networks have demanding memory requirements. The largest network that could fit on a single NVIDIA Tesla M2090 (with 6 GB of memory) was comprised of $610,944$ neurons and approximately 137 million synapses, which corresponds to processing a 64×64 input video. In order to run larger networks on current-generation GPU cards, a change in model (or software and hardware) architecture is required. One should note that this is only a temporary limitation and could

be overcome by the next generation of GPU cards. Another possible solution would be to employ multi-GPU systems; however, more work is required to efficiently integrate our SNN simulator with such a system.

5.4.3 Practical Implications

The present network might be of interest to the neuroscientist and computer vision research communities for the following reasons.

First, our implementation outperforms the S&H C/MATLAB implementation by orders of magnitude in terms of computational speed and memory usage. Thus our CUDA implementation can be used to save computation time, as well as be applied to input resolutions that the C/MATLAB implementation cannot handle due to memory constraints. Additionally, the CUDA implementation can act as a stand-alone module that could potentially be used in computer vision as an alternative to computationally expensive operations such as Gabor filtering for edge detection or dense optic flow computations.

Second, we have demonstrated that our approach is fast, efficient, and scalable; although current GPU cards limit the size of the simulations due to memory constraints. Nevertheless, our model processes a 40×40 input video at 20 frames per second in real-time,which corresponds to a total of $239,040$ neurons in the simulated V1, MT, and LIP areas, at 20 frames per second using a single GPU, which enables the potential use of our software in real-time applications ranging from robot vision to autonomous driving.

Third, our implementation might be of particular interest to the neuromorphic modeling community, as the present neuron model, synapse model, and AER are compatible with recent neuromorphic hardware (Srinivasa and Cruz-Albrecht, 2012). Thus our algorithm could be used as a neural controller in neuromorphic and neurorobotics applications. Future

work could be directed toward creating an interface by which networks can be automatically exported onto neuromorphic hardware.

Fourth, because of the modular code structure, our implementation can be readily extended to include, for example, higher-order visual areas or biologically plausible synaptic learning rules such as STDP. Thus our implementation may facilitate the testing of hypotheses and the study of the temporal dynamics that govern visual motion processes in area MT, which might prove harder to study using previous (rate-based) model incarnations. Lastly, the network was constructed using CARLsim (see Chapter 3), with all source code for the simulator, the spiking network, and analysis scripts publicly available. As such it is the next step towards our goal of making efficient simulations of large-scale spiking networks available to a wide range of researchers, without the need of a cluster or supercomputer.

Chapter 6

GPU-Accelerated Cortical Neural Network Model for Visually Guided Robot Navigation

6.1 Introduction

Despite a wealth of data regarding the neural circuitry concerned with the perception of self-motion variables such as the current direction of travel ("heading") (Britten and Newsome, 1998; Duffy and Wurtz, 1997; Gu et al., 2006) or the perceived position and speed of objects (Eifuku and Wurtz, 1998; Tanaka et al., 1993), little research has been devoted to investigating how this neural circuitry may relate to the behavioral dynamics of steering around obstacles and towards goals (Fajen and Warren, 2003; Wilkie and Wann, 2003).

In the last two chapters, I focused on an efficient implementation of a large-scale SNN model that could build a cortical representation of visual motion in the spiking domain. In this chapter, I aim to investigate whether the model can be used to generate appropriate steering

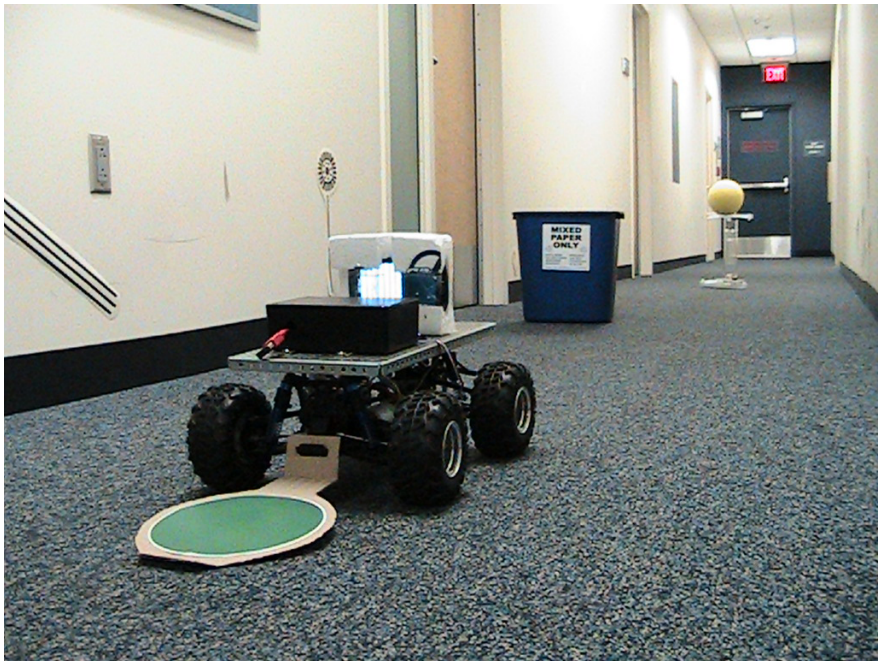

Figure 6.1: "Le Carl" Android based robot, which was constructed from the chassis of an R/C car. The task of the robot was to navigate to a visually salient target (bright yellow foam ball) while avoiding an obstacle (blue recycle bin) along the way. The robot's position throughout the task was monitored by an overhead camera that tracked the position of the green marker.

commands in a visually guided navigation task when embodied on a physical robot exploring a real-world environment (see Fig. 6.1).

Visually guided navigation has traditionally attracted much attention from the domains of both vision and control, producing countless computational models with excellent navigation and localization performance (for a recent survey see Bonin-Font et al. (2008)). However, the majority of these models have taken a computer science or engineering approach, without regard to biological or psychophysical fidelity. To date, only a handful of neural network models have been able to explain the trajectories taken by humans to steer around stationary objects toward a goal (Browning et al., 2009a; Elder et al., 2009), and none of them have been tested in real-world environments. The real world is the ultimate test bench for a model that is trying to link perception to action, because even carefully devised simulated experiments typically fail to transfer to real-world settings. Real environments are rich,

multimodal, and noisy; an artificial design of such an environment would be computationally intensive and difficult to simulate (Krichmar and Edelman, 2006). Yet real-world integration is often prohibited due to engineering requirements, programming intricacies, and the sheer computational cost that come with large-scale biological models. Instead, such models often find application only in constrained or virtual environments, which may severely limit their explanatory power when it comes to generalizing findings to real-world conditions.

In contrast, developing a robotic platform whose behavior is guided by a biologically detailed network of spiking neurons might allow us to evaluate both theories and simulations in real-world environments. Since SNN models are also compatible with recent neuromorphic architectures (Boahen, 2006; Cassidy et al., 2014; Khan et al., 2008; Schemmel et al., 2010; Srinivasa and Cruz-Albrecht, 2012) and neuromorphic sensors (Lichtsteiner et al., 2008; Liu et al., 2010; Wen and Boahen, 2009), developing neurorobotic agents that display cognitive functions or learn behavioral abilities through autonomous interaction may also represent an important step toward realizing functional SNN networks on neuromorphic hardware.

6.2 Methods

6.2.1 The Robotic Platform

In order to engineer a system capable of real-time execution and real-world integration of large-scale biological models, we needed to address several technical challenges as outlined below.

First, the Android Based Robotics (ABR) framework (Oros and Krichmar, 2012, 2013a,b) was used as a flexible and inexpensive open-source robotics platform. In specific, we made use of the "Le Carl" robot, an ABR platform constructed from the chassis of an R/C car

(see Fig. 6.2). The main controller of the platform was an Android mobile phone (Samsung Galaxy S3), which mediated communication via Bluetooth to an IOIO electronic board (SparkFun Electronics[1]), which in turn sent Pulse-Width Modulation (PWM) commands to the actuators and speed controllers of the R/C car. Since today's smartphones are equipped with a number of sensors and processors, it is possible to execute computationally demanding software based on real-time sensory data directly on the phone. The first neurorobotics study to utilize the ABR platform implemented a neural network based on neuromodulated attentional pathways to perform a reversal learning task by fusing a number of sensory inputs such as GPS location, a compass reading, and the values of on-board IR sensors (Oros and Krichmar, 2012). However, the complexity and sheer size of the present model did not allow for on-board processing, and instead required hosting the computationally expensive components on a remote machine.

Second, an Android app (labeled "ABR client" in Fig. 6.2) was developed to allow the collection of sensory data and communication with a remote machine (labeled "ABR server" in Fig. 6.2) via WiFi and 3G[2]. The only sensor used in the present study was the Android phone's camera, which collected 320×240 pixel images while the robot was behaving. These images were then sent via User Datagram Protocol (UDP) to the ABR server, at a rate of roughly 20 frames per second. The neural network processed the stream of incoming images in real-time and generated motor commands, which were received by the ABR client app running on the phone via Transmission Control Protocol (TCP). The ABR client communicated motor commands directly to the corresponding actuators and speed controllers of the R/C car via the IOIO board. Third, we developed a Qt software interface that allowed integration of the ABR server with different C/C++ libraries such as CUDA, OpenCV, and the SNN simulator CARLsim (Beyeler et al., 2015a; Nageswaran et al., 2009; Richert et al., 2011). In order to adhere to the real-time constraints of the system, all computationally

[1]The IOIO board can be obtained from `http://www.sparkfun.com`.
[2]All ABR source code is available from `https://www.github.com/UCI-ABR`.

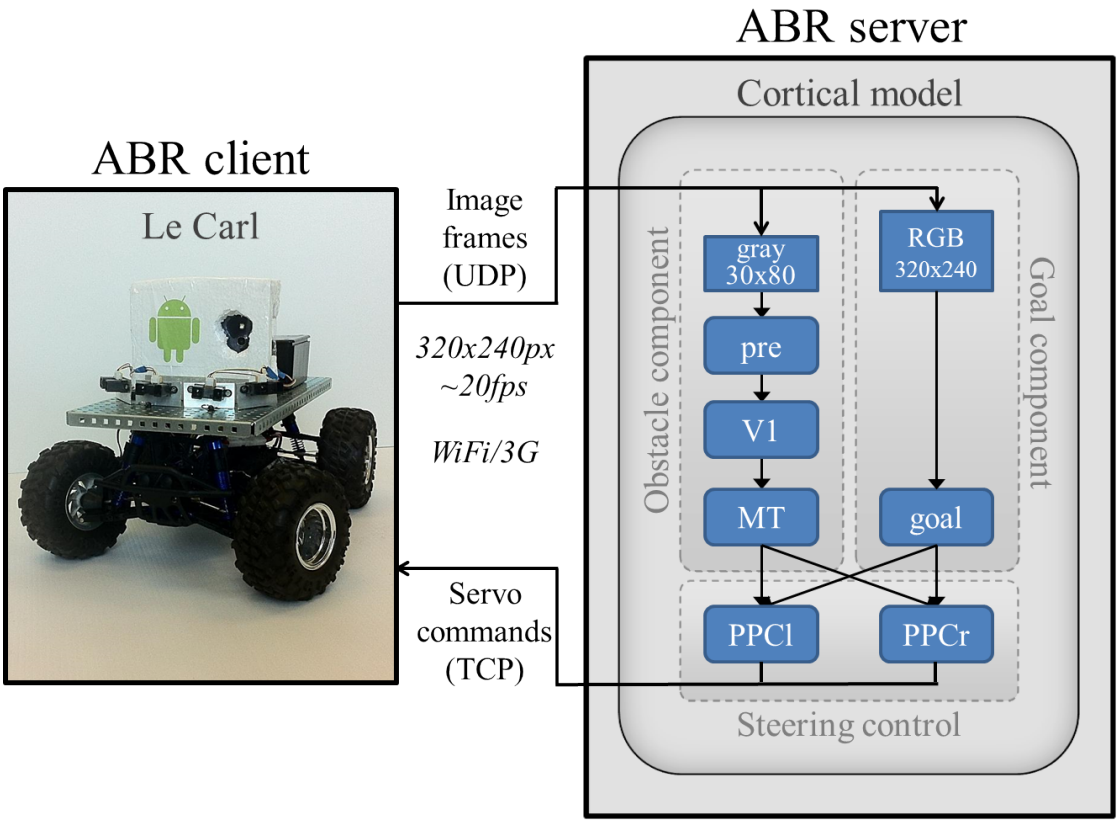

Figure 6.2: Technical setup. An Android app (ABR client) was used to record 320 × 240 images at 20 fps and send them to a remote machine (ABR server) hosting a cortical model made of two processing streams: an obstacle component responsible for inferring the relative position and size of nearby obstacles by means of motion discontinuities, and a goal component responsible for inferring the relative position and size of a goal object by means of color blob detection. These streams were then fused in a model of the Posterior Parietal Cortex (PPC) to generate steering commands that were sent back to the ABR platform.

intensive parts of the model were accelerated on a GPU (i.e., a single NVIDIA GTX 780 with 3 GB of memory) using the CUDA programming framework.

6.2.2 The Cortical Neural Network Model

The high-level architecture of the cortical neural network model is shown in Fig. 6.2. The model was based on the cortical model of visual motion processing presented in Chapters 4 and 5. The model used an efficient GPU implementation of the motion energy model (Si-

moncelli and Heeger, 1998) to generate cortical representations of motion in a model of the Primary Visual Cortex (V1). Spiking neurons in a model of the Middle Temporal area (MT) then located nearby obstacles by means of motion discontinuities (labeled "Obstacle component" in Fig. 6.2). The MT motion signals projected to a simulated Posterior Parietal Cortex (PPC), where they interacted with the representation of a goal location (labeled "Goal component" in Fig. 6.2) to produce motor commands to steer the robot around obstacles toward a goal. The following subsections will explain the model in detail.

6.2.2.1 Visual Input

Inputs to the model were provided by the built-in camera of an Android phone (Samsung Galaxy S3) mounted on the ABR platform. The phone took 320×240 pixel snapshots of the environment at a rate of roughly 20 frames per second. These frames entered the model in two ways: First, mimicking properties of visual processing in the magnocellular pathway, a grayscale, down-scaled version of the lower half of the frame (80×30 pixels) was sent to the network processing visual motion (obstacle component). This pathway consisted of a preprocessing stage as well as a model of V1, MT, and the PPC. Second, mimicking properties of visual processing in the parvocellular pathway, the originally collected RGB frame (320×240 pixels) was sent to a model component concerned with locating a visually salient target in the scene (goal component). Because a neural implementation of this pathway was considered out of scope for the present study, the goal object was located using OpenCV-based segmentation in the Hue, Saturation, Value (HSV) color space. The location of the goal in the visual field was then communicated to the PPC.

6.2.2.2 Preprocessing

The first stage of the obstacle component pathway (labeled "pre" in Fig. 6.2) enhanced contrast and normalized the input image using OpenCV's standard implementation of Contrast-Limited Adaptive Histogram Equalization (CLAHE). The processed frame was then sent to the V1 stage.

6.2.2.3 Primary Visual Cortex (V1)

The second stage of the obstacle component pathway (labeled "V1" in Fig. 6.2) used the motion energy model to implement model neurons that responded to a preferred direction and speed of motion (Simoncelli and Heeger, 1998).

This stage of the model is described in detail in Chapter 4. In short, the motion energy model used a bank of linear space-time oriented filters to model the receptive field of directionally selective simple cells in V1. The filter responses were half-rectified, squared, and normalized within a large spatial Gaussian envelope. The output of this stage was equivalent to rate-based activity of V1 complex cells, whose responses were computed as local weighted averages of simple cell responses in a Gaussian neighborhood of $\sigma_{\mathrm{V1c}} = 1.6$ pixels (see Fig. 6.2). The resulting size of the receptive fields was roughly one degree of the visual field (considering that the horizontal field of view of a Samsung Galaxy S3 is roughly 60°), which is in agreement with electrophysiological evidence from recordings in macaque V1 (Freeman and Simoncelli, 2011). V1 complex cells responded to eight different directions of motion (in 45° increments) at a speed of 1.5 pixels per frame. We interpreted these activity values as neuronal mean firing rates, which were scaled to match the contrast sensitivity function of V1 complex cells (see Section 4.2.4). Based on these mean firing rates we then generated Poisson spike trains (of 50 ms duration), which served as the spiking input to Izhikevich neurons representing cells in area MT.

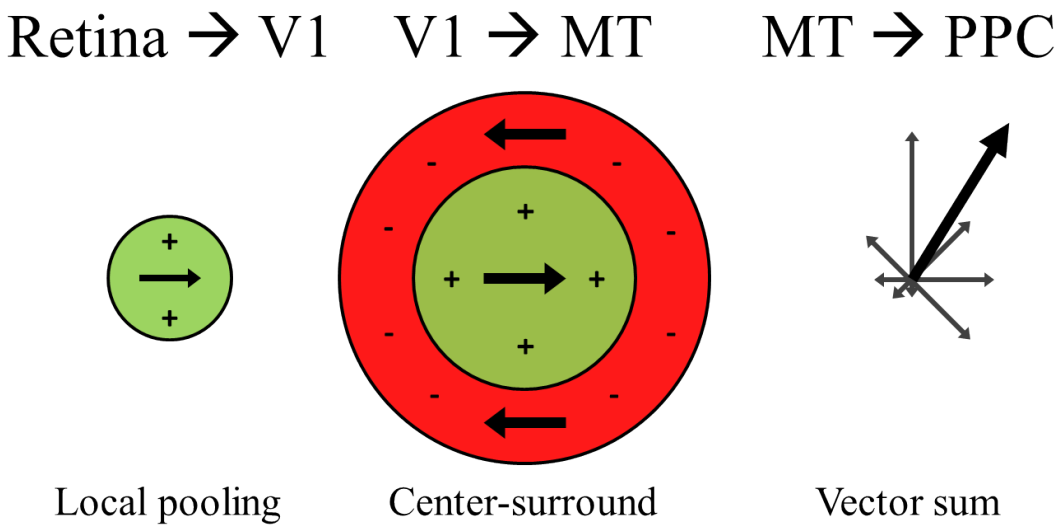

Figure 6.3: Schematic of the spatial receptive fields in the network. V1 neurons pooled retinal afferents and computed directional responses according to the motion energy model (Simoncelli and Heeger, 1998). The receptive fields of MT neurons had a circular center preferring motion in a particular direction, surrounded by a region preferring motion in the anti-preferred direction, implemented as a difference of Gaussians. PPC neurons computed a net vector response weighted by the firing rates of MT neurons.

6.2.2.4 Middle temporal (MT) area

The simulated area MT (labeled "MT" in Fig. 6.2) consisted of $40,000$ Izhikevich spiking neurons and roughly $1,700,000$ conductance-based synapses, which aimed to extract the position and perceived size of any nearby obstacles by means of detecting motion discontinuities.

This stage of the model is based on the SNN model of MT described in detail in Chapter 5. Neurons in model MT received optic flow-like input from V1 cells, thus inheriting their speed and direction preferences (Born and Bradley, 2005). Their spatial receptive fields had a circular center preferring motion in a particular direction, surrounded by a region preferring motion in the anti-preferred direction (Allman et al., 1985; Born, 2000). These receptive fields were implemented as a difference of Gaussians: Excitatory neurons in MT received input from Poisson spike generators in V1 in a narrow spatial neighborhood ($\sigma_{\mathrm{MTe}} = 6.0$ pixels) and from

inhibitory neurons in MT in a significantly larger spatial neighborhood ($\sigma_{\mathrm{MTi}} = 12.0$ pixels; see Fig. 6.3), which are comparable in size to receptive fields of neurons in layers 4 and 6 of macaque MT (Raiguel et al., 1995). The weights and connection probabilities scaled with distance according to the Gaussian distribution. The maximum excitatory weight (at the center of a Gaussian kernel) was 0.01 and the maximum inhibitory weight was 0.0015. As a result of their intricate spatial receptive fields, excitatory neurons in MT were maximally activated by motion discontinuities in the optic flow field, which is thought to be of use for detecting object motion (Allman et al., 1985; Bradley and Andersen, 1998). Note that cortical cells with such a receptive field organization usually exhibit different binocular disparity preferences in their center and surround regions (Bradley and Andersen, 1998). However, because the model had access to only a single camera, we were unable to exploit disparity information (for more information please refer to Section 6.4.2).

All neurons in MT were modeled as Izhikevich spiking neurons (Izhikevich, 2003) (Eqs. 2.4–2.6; see Section 2.3.2.1). All excitatory neurons were modeled as Regular Spiking (RS) neurons (class 1 excitable, $a = 0.02$, $b = 0.2$, $c = 65$, $d = 8$), and all inhibitory neurons were modeled as Fast Spiking (FS) neurons (class 2 excitable, $a = 0.1$, $b = 0.2$, $c = 65$, $d = 2$) (Izhikevich, 2003).

Ionic currents were modeled as dynamic synaptic channels with zero rise time and exponential decay (Eqs. 2.7, 2.8; see Section 2.3.2.3). Synaptic weights were set to 0.01 at the center of an excitatory Gaussian kernel, and 0.0015 at the center of an inhibitory Gaussian kernel. Time constants were set to standard values for the different channels; that is, AMPA (fast decay, $\tau_{\mathrm{AMPA}} = 5\,\mathrm{ms}$), NMDA (slow decay and voltage-dependent, $\tau_{\mathrm{NMDA}} = 150\,\mathrm{ms}$), GABAa (fast decay, $\tau_{\mathrm{GABAa}} = 6\,\mathrm{ms}$), and GABAb (slow decay, $\tau_{\mathrm{GABAb}} = 150\,\mathrm{ms}$). A spike arriving at a synapse that was postsynaptically connected to an excitatory (inhibitory) neuron increased both g_{AMPA} and g_{NMDA} (g_{GABAa} and g_{GABAb}) with receptor-specific efficacy $\eta_{\mathrm{AMPA}} = 1.5$ and $\eta_{\mathrm{NMDA}} = 0.5$ ($\eta_{\mathrm{GABAa}} = 1.0$ and $\eta_{\mathrm{GABAb}} = 1.0$). Having $\eta_{\mathrm{AMPA}} > \eta_{\mathrm{NMDA}}$

agrees with experimental findings (Myme et al., 2003) and allowed the network to quickly react to changing sensory input.

For more information on the exact implementation of the Izhikevich model, please refer to Chapter 2 and to the CARLsim 2.0 release paper (Richert et al., 2011).

6.2.2.5 Posterior Parietal Cortex (PPC)

The simulated PPC (labeled "PPCl" and "PPCr" in Fig. 6.2) combined visual representations of goal and obstacle information to produce a steering signal. The resulting dynamics of the steering signal resembled the Balance Strategy, which is a simple control law that aims to steer away from large sources of optic flow in the visual scene. For example, honeybees use this control law to steer collision-free paths through even narrow gaps by balancing the apparent speeds of motion of the images in their eyes (Srinivasan and Zhang, 1997). Interestingly, there is also some evidence for the Balance Strategy in humans (Kountouriotis et al., 2013). However, the present model differs in an important way from the traditional Balance Strategy, in that it tries to balance the flow generated from motion discontinuities in the visual field, which are thought to correspond to obstacles in the scene, instead of balancing a conventional optic flow field. For more information see Section 6.4.1.

The steering signal took the form of a turning rate, $\dot{\theta}$:

$$\dot{\theta} = \hat{\theta} - \theta, \tag{6.1}$$

which was derived from the difference between the robot's current angular orientation, θ, and an optimal angular orientation estimated by the cortical network, $\hat{\theta}$. The variable θ was based on a copy of the current PWM signal sent to the servos of the robot (efference

copy), whereas the variable $\hat{\theta}$ was derived from the neural activation in PPC, as described in Eq. 6.2. The resulting turning rate, $\dot{\theta}$, was then directly mapped into a PWM signal sent to the servos that controlled the steering of the robot. In order not to damage the servos, we made sure that the computed instantaneous change in turning rate, $\ddot{\theta}$, never exceeded a threshold (set at roughly 10 % of the full range of PWM values). Steering with the second derivative also contributed to the paths of the robot being smoother.

The cortical estimate of angular orientation was realized by separately summing optic flow information from the left (F_{L}) and right halves (F_{R}) of the visual scene according to the Balance Strategy, and combining these flow magnitudes with information about the target location (T_{L} and T_{R}) weighed with a scaling factor α (set at 0.6):

$$\hat{\theta} \propto \frac{F_{\mathrm{L}} - F_{\mathrm{R}} + \alpha(T_{\mathrm{R}} - T_{\mathrm{L}})}{F_{\mathrm{L}} + F_{\mathrm{R}} + \alpha(T_{\mathrm{R}} + T_{\mathrm{L}})}. \tag{6.2}$$

Here, the total optic flow in the left (F_{L}) and right (F_{R}) halves was computed as follows:

$$F_{\mathrm{L}} = \sum_{\theta} \sum_{y=0}^{R-1} \sum_{x=0}^{C/2-1} ||r_{\theta}(x,y)e_{\theta}||, \tag{6.3}$$

$$F_{\mathrm{R}} = \sum_{\theta} \sum_{y=0}^{R-1} \sum_{x=C/2}^{C-1} ||r_{\theta}(x,y)e_{\theta}||, \tag{6.4}$$

where R is the number of rows in the image, C is the number of columns in the image, and $||.||$ denotes the Euclidean 2-norm. Depending on the experiment, $r_{\theta}(x,y)$ is the firing rate of either an MT or a V1 neuron that is selective to direction of motion θ at spatial location (x,y), and e_{θ} is a unit vector pointing in the direction of θ.

The target location was represented as a two-dimensional blob of activity centered on the target's center of mass $(x_{\mathrm{G}}, y_{\mathrm{G}})$. If the target was located in the left half of the image, it

contributed to a term T_{L}:

$$T_{\mathrm{L}} = \alpha A_{\mathrm{G}} \sum_{y=0}^{R-1} \sum_{x=0}^{C/2-1} \exp\left(- \frac{(x - x_{\mathrm{G}})^2 + (y - y_{\mathrm{G}})^2}{2\sigma_{\mathrm{G}}^2} \right), \tag{6.5}$$

and if it was located in the right half of the image, it contributed to a term T_{R}:

$$T_{\mathrm{R}} = \alpha A_{\mathrm{G}} \sum_{y=0}^{R-1} \sum_{x=C/2}^{C-1} \exp\left(- \frac{(x - x_{\mathrm{G}})^2 + (y - y_{\mathrm{G}})^2}{2\sigma_{\mathrm{G}}^2} \right), \tag{6.6}$$

where $\alpha = 0.6$, $\sigma_{\mathrm{G}} = 0.2C$, and A_{G} was the perceived area of the target, which was determined with OpenCV using color blob detection. This allowed the target contribution to increase with target size, which we assumed to scale inversely with target distance (also compare Section 6.4.2). Note that if the target was perfectly centered in the visual field, it contributed equally to T_{L} and T_{R}. Also, note that the contribution of the target component from the right half of the image, T_{R}, to θ was positive, whereas the contribution of the obstacle component F_{R} was negative. This led to the target component exhibiting an attractor-like quality in the steering dynamics, whereas the goal component exhibited a repeller-like quality.

6.3 Results

6.3.1 Experimental Setup

In order to investigate whether the motion estimates in the simulated MT were sufficient for human-like steering performance, the model was tested on a reactive navigation task in

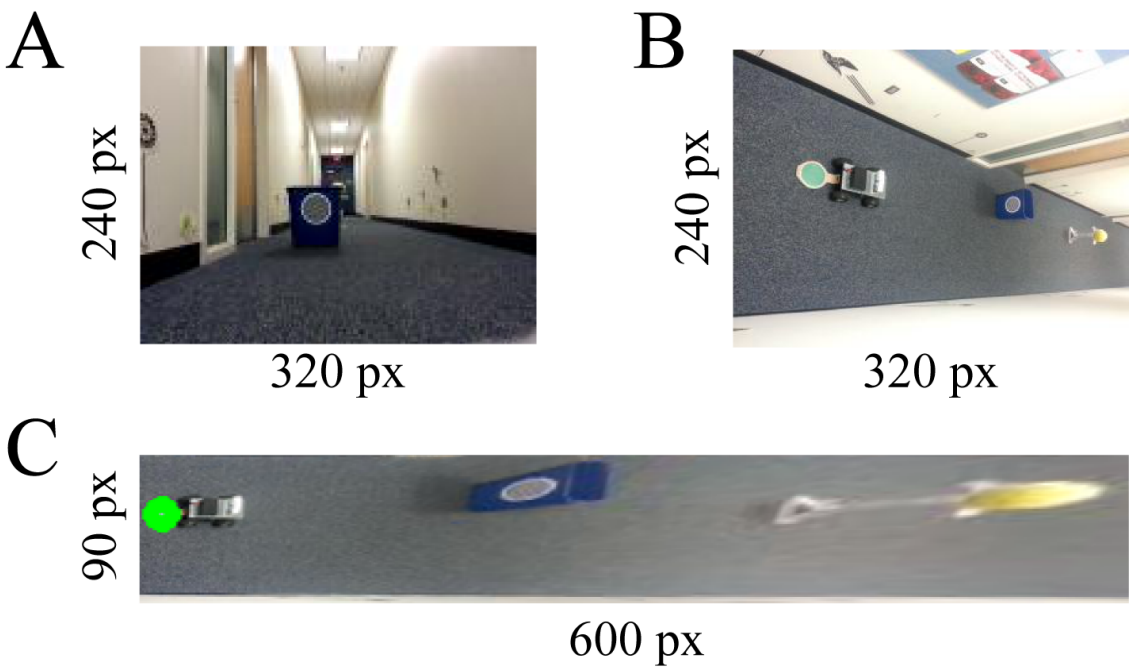

Figure 6.4: Camera setup. **A**, The robot's initial view of the scene from the onboard Android phone. **B**, View of an overhead camera that tracked the green marker attached to the robot. **C**, Birds-eye view image of the scene, obtained via perspective transformation from the image shown in (B). The robot's location in each frame was inferred from the location of the marker, which in turn was determined using color blob detection.

the hallway of the University of California, Irvine Social and Behavioral Sciences Gateway, where our laboratory is located (see Fig. 6.1). This setup provided a relatively narrow yet highly structured environment with multiple sources of artificial lighting, thus challenging the cortical model to deal with real-world obstacles as well as to quickly react to changing illumination conditions.

Analogous to the behavioral paradigm of Fajen and Warren (2003), the robot's goal was to steer around an obstacle placed in its way (i.e., a recycle bin) in order to reach a distant goal (i.e., a yellow foam ball). The obstacle was placed between the location of the target and the robot's initial position, at three different angles (off-set by one, four, and eight degrees to the left of the robot's initial view) and different distances (2.5, 3, and 3.5 m from the robot's initial position). For each of these configurations we ran a total of five trials, during which we collected both behavioral and simulated neural data. An example setup is

shown in Fig. 6.4A. At the beginning of each trial, the robot was placed in the same initial position, directly facing the target. The robot would then quickly accelerate to a maximum speed of roughly $1\,\mathrm{m\,s^{-1}}$. Speed was then held constant until the robot had navigated to the vicinity of the goal (roughly 0.5 m apart), at which point the trial was ended through manual intervention.

An over-head camera was used to monitor the robot's path throughout each trial (see Fig. 6.4B). The camera was mounted on the wall such that it overlooked the hallway, and was used to track the position of a large green marker attached to the rear of the robot (best seen in Fig. 6.1). Using a perspective transformation, a 320×240 pixel image (taken every 100 ms) was converted into a birds-eye view of the scene (see Fig. 6.4C). We used the standard OpenCV implementation of this transformation and fine-tuned the parameters for the specific internal parameters of the camera. Although three-dimensional objects will appear distorted (as is evident in Fig. 6.4C), flat objects lying on the ground plane (i.e., the hallway floor) will be drawn to scale. This allowed us to detect and track the green marker on the robot directly in the transformed image (Fig. 6.4C), in which the size of the marker does not depend on viewing distance (as opposed to Fig. 6.4B). Color blob detection was applied to find the marker in each frame, and the marker's location was used to infer the robot's location. In order to aid tracking of the marker, we added temporal constraints between pairs of subsequent frames for outlier rejection. This procedure not only allowed us to automatically record the robot's location at any given time, but also allowed direct comparison of the robot's behavioral results with human psychophysics data.

6.3.2 Behavioral Results

Fajen and Warren (2003) studied how humans walk around obstacles toward a stationary goal, and developed a model to explain the steering trajectories of the study's participants.

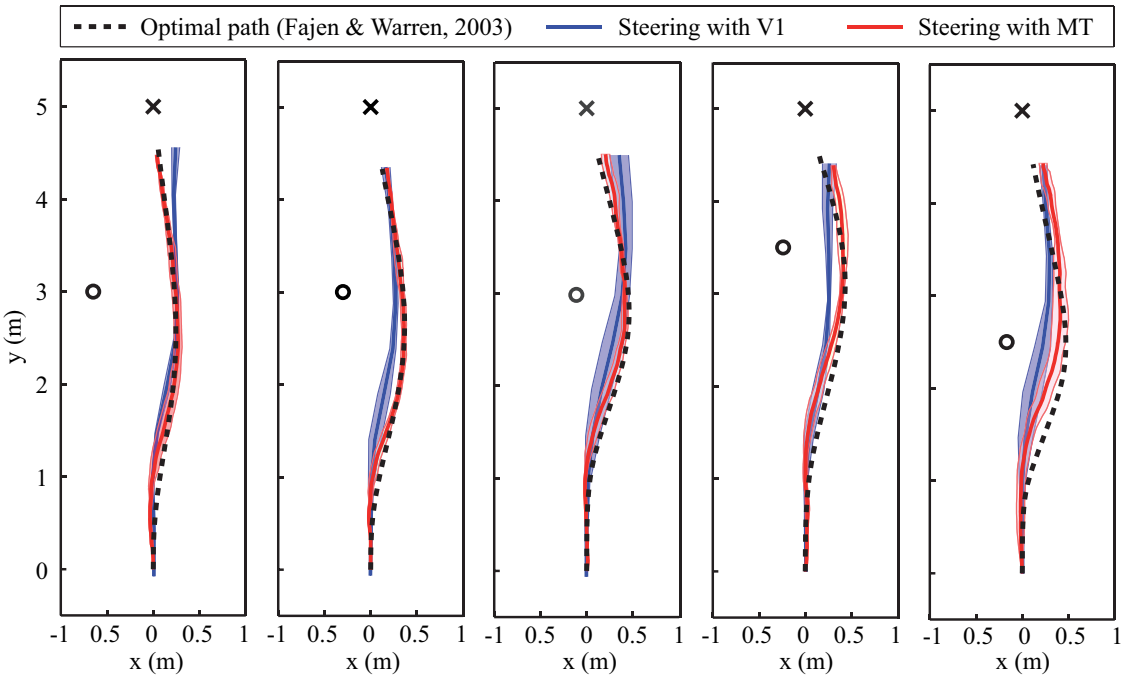

Figure 6.5: Behavior paths of the robot (colored solid lines) around a single obstacle (recycle bin, 'O') toward a visually salient goal (yellow foam ball, 'X') for five different scene geometries, compared to "ground truth" (black dashed lines) obtained from the behavioral model by Fajen and Warren (2003). Results are shown for steering with V1 (blue) as well as for steering with MT (red). Solid lines are the robot's mean path averaged over five trials, and the shaded regions correspond to the standard deviation.

Their work revealed that accurate steering trajectories can be obtained directly from the scene geometry, namely the distance and angles to the goal and obstacles. Using their behavioral model, we calculated "ground truth" paths for the scene geometry of our experimental setup, and compared them to steering trajectories generated by the robot's cortical neural network model. In order to assess the contribution of motion processing in MT we conducted two sets of experiments: one where steering commands were based solely on V1 activity, and one where steering commands were based on MT activity, by adjusting $r_\theta(x, y)$ in Eq. 6.3 and Eq. 6.4.

Fig. 6.5 illustrates the robot's path around a single obstacle (recycle bin, 'O') toward a visually salient goal (yellow foam ball, 'X') for five different scene geometries, analogous to Fig. 10 in Fajen and Warren (2003). Results are shown for steering with V1 (blue) as well

as for steering with MT (red). In the first three setups, the obstacle was placed 3 m away from the robot's initial location, off-set by eight (red), four (green), and one (blue) degrees to the left of the robot's initial view. Two other setups were tested, in which the obstacle distance was 3.5 m (yellow) and 2.5 m (magenta) at an angle of 4°. The robot's mean path from five experimental runs is shown as a thick solid line in each panel, and the shaded region corresponds to the standard deviation. The black dashed lines correspond to the "ground truth" paths calculated from the behavioral model by Fajen and Warren (2003) using the corresponding scene geometry and standard parameter values. Each of these five obstacle courses was run five times for a total of 25 successful trials.

Steering paths were generally more accurate and more robust when steering with MT as opposed to steering with V1. Paths generated from neural activity in MT not only closely matched human behavioral data, but they also were surprisingly robust. That is, the robot successfully completed all 25 trials without hitting an obstacle, the walls of the hallway, or missing the goal. In fact, the only setup that proved difficult for the robot had the obstacle placed only 2.5 m away from the robot (rightmost panel in Fig. 6.5). In this scenario, "ground truth" data indicates that humans start veering to the right within the first second of a trial, which seemed not enough time for the robot. In contrast, steering from neural activity in V1 led to a total of 7 crashes, which involved the robot driving either into the obstacle or the wall, upon which the trial was aborted and all collected data discarded. We repeated the experiment until the robot successfully completed five trials per condition. When steering from V1 neural activity, the robot tended to react more strongly to behaviorally irrelevant stimuli, leading to more variability in the robot's trajectories.

For all five tested conditions we calculated the path area error (using the trapezoidal rule for approximating the region under the graph) as well as the maximum distance the robot's path deviated from the ground truth at any given time. Mean and standard deviation of these data (averaged over five trials each) are summarized in Table 6.1. All path errors were on the

Table 6.1: Summary of mean path errors when steering with signals in V1 vs. MT (5 trials each).

Obstacle distance (m)	Obstacle angle (deg)	Area error (m)		Maximum deviation (m)	
		V1	MT	V1	MT
3	-1°	0.472 ± 0.174	0.193 ± 0.080	0.237 ± 0.077	0.112 ± 0.045
3	-4°	0.407 ± 0.097	0.140 ± 0.024	0.193 ± 0.052	0.091 ± 0.013
3	-8°	0.354 ± 0.050	0.170 ± 0.055	0.165 ± 0.020	0.102 ± 0.026
3.5	-4°	0.332 ± 0.110	0.264 ± 0.100	0.169 ± 0.039	0.143 ± 0.029
2.5	-4°	0.575 ± 0.176	0.319 ± 0.093	0.288 ± 0.099	0.193 ± 0.056

order of 0.1 m, with errors generated from steering with MT consistently smaller than errors generated from steering with V1. When the robot was steering from signals in MT, trial-by-trial variations were on the order of 0.01 m, suggesting that under these circumstances the robot reliably exhibited human-like steering behavior in all tested conditions. The maximum distance that the robot's path deviated from the human-like path was on the order of 10 cm, which is much smaller than the obstacle width (37 cm). In contrast, steering with V1 often led to much larger deviations in some cases. The best single-trial result was 0.084 m area error and 5 cm maximum deviation on the -1° condition. These results suggest that in most trials the task could still be performed even when steering with V1, but that motion processing in MT was critical for reliability and accuracy, both of which are distinct qualities of human-like steering.

Furthermore, these results are comparable to simulated data obtained with the STARS model, where the authors reported path area errors and maximum deviations on the same order of magnitude (see Table 2 in Elder et al. (2009)). Their best result was 0.097 m area error and 2.52 cm maximum deviation on a single trial of the -2° condition. However, the STARS model did not directly process visual input, nor was it tested in a real-world environment. The follow-up study to the STARS model (called ViSTARS) did not provide quantitative data on path errors in the same manner that would allow comparison here. Thus similar performance to the STARS model is assumed.

A movie showcasing the behavioral trajectories of the robot is available on our website: `www.socsci.uci.edu/jkrichma/ABR/lecarl_obstacle_avoidance.mp4`.

6.3.3 Neural Activity

Fig. 6.6 shows the activity of neurons in V1 (Panel A) and MT (Panel B) recorded in a single trial of the third setup in Fig. 6.5 (blue). For the sake of clarity, only every eighth

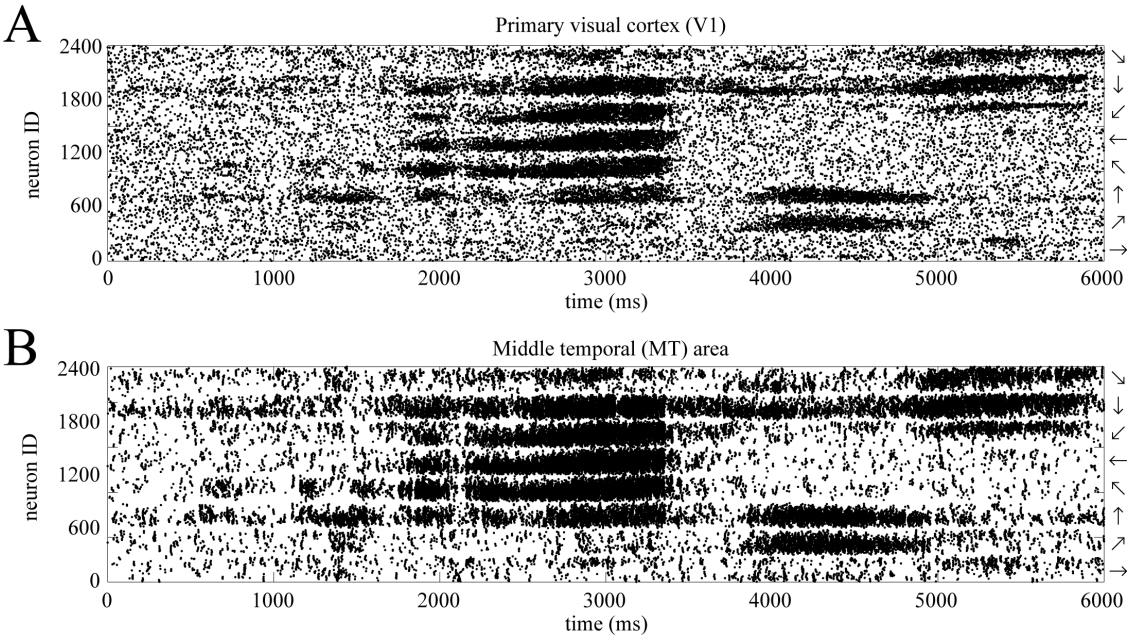

Figure 6.6: Raster plot of neuronal activity for V1 and MT recorded in a single trial where the obstacle was 3 m away and offset by 8° (red colored line in Fig. 6.5). A dot represents an action potential from a simulated neuron. For the sake of visualization, only every eighth neuron in the population is shown over the time course of six seconds that included both the obstacle avoidance and the goal approach. Neurons are organized according to their direction selectivity, where neurons with ID $0-299$ were most selective to rightward motion (0°), IDs $300-599$ mapped to upward-rightward motion at 45°, IDs $600-899$ mapped to upward motion at 90°, IDs $900-1199$ mapped to 135°, IDs $1200-1499$ mapped to 180°, IDs $1500-1799$ mapped to 225°, IDs $1800-2099$ mapped to 270°, and IDs $2100-2299$ mapped to 315°. **A**, V1 spike trains generated from a Poisson distribution with mean firing rate equal to the linear filter response. Average activity in the population was 18.6 ± 15.44 Hz. **B**, Spike trains of Izhikevich neurons in MT. Average activity in the population was 5.74 ± 8.98 Hz.

neuron in the population is shown over a time course of six seconds, which included both the obstacle avoidance and the goal approach. Neurons are organized according to their direction selectivity, where neurons with ID $0-299$ were most selective to rightward motion (0°), IDs $300-599$ mapped to upward-rightward motion at 45°, IDs $600-899$ mapped to upward motion at 90°, IDs $900-1199$ mapped to 135°, IDs $1200-1499$ mapped to 180°, IDs $1500-1799$ mapped to 225°, IDs $1800-2099$ mapped to 270°, and IDs $2100-2299$ mapped to 315°. Panel A shows the spike trains generated from the rate-based activity of V1 neurons using a Poisson spike generator. The firing of V1 neurons was broad and imprecise in response to stimuli. In contrast, spiking responses of neurons in MT (Panel B) were more

selective, and sharper than V1. As the robot approached the obstacle roughly from 1000 to 2500 ms of the trial shown in Fig. 6.6, neural activity in both V1 and MT steadily increased in both strength and spatial extent. As activity in MT increased a repeller-like behavioral response was triggered, which in turn introduced even more apparent motion on the retina (roughly from 2200 to 3200 ms). As soon as the obstacle moved out of the robot's view due to successful avoidance (around 3200 ms), activity in MT rapidly dropped off, causing the robot's turning rate to momentarily decrease. With the goal still in view and growing in size, turning rate was now increased with opposite sign (roughly from 3500 to 4500 ms), leading to an attractor-like behavioral response that led the robot directly to the goal. As the robot navigated to the goal (within a 0.5 m distance), the trial was ended manually. Note that near the end of the trial (roughly from 4000 to 5000 ms), neurons in MT rightly perceived the goal object also as an obstacle, which is something the Fajen and Warren model does not take into account. But, because the goal component of the steering command grew more rapidly with size than the obstacle component, the robot continued to approach the goal, rather than starting to avoid it.

Overall network activity and corresponding behavioral output is illustrated in Figs. 6.7 and 6.8. Each Panel summarizes the processing of a single visual input frame, which corresponded to a time interval of 50 ms. The indicated time intervals are aligned with the neuronal traces presented in Fig. 6.6. A frame received at time $t - 50$ ms led to a motor response at time t. This sensorimotor delay was due to the fact that the SNN component of the model needed to be executed for 50 ms. Neural activity during these 50 ms is illustrated as an optic flow field, overlaid on visual input, for both Poisson spike generators in V1 and Izhikevich spiking neurons in MT. Here, every arrow represents the population vector of the activity of eight neurons (selective to the eight directions of motion) coding for the same spatial location. Note that, because these neurons were maximally selective to a speed of 1.5 pixels per frame, vector length does not indicate velocity as is the case in a conventional optic flow field, but rather confidence in the direction estimate. For the sake of clarity, only every

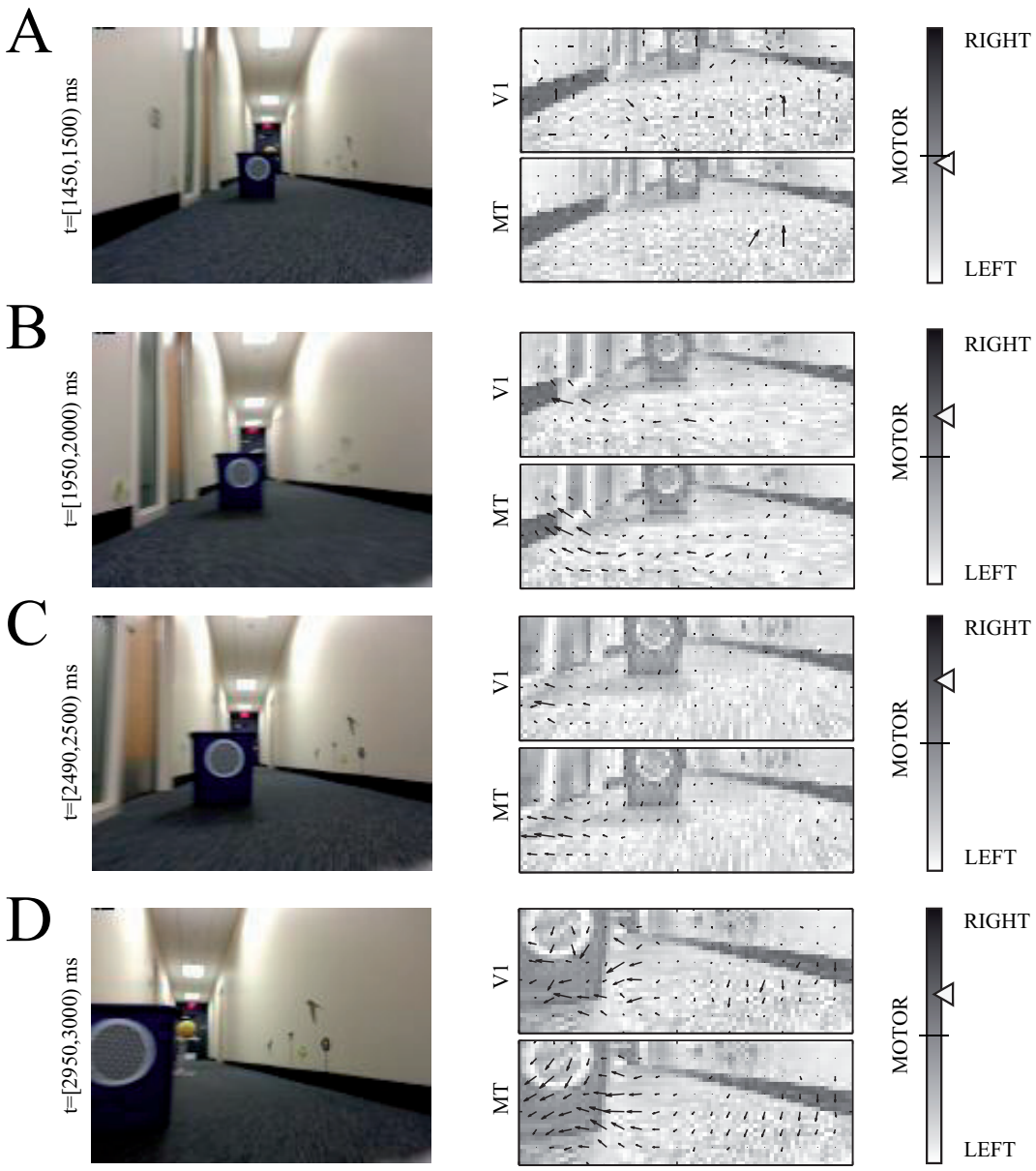

Figure 6.7: Overall network activity and corresponding behavioral output during obstacle avoidance in a single trial. Each Panel summarizes the processing of a single visual input frame, which corresponded to a time interval of 50 ms. The indicated time intervals are aligned with the neuronal traces presented in Fig. 6.6. A frame received at time $t - 50$ ms led to a motor response at time t. Neural activity during these 50 ms is illustrated as an optic flow field, overlaid on visual input, for both Poisson spike generators in V1 and Izhikevich spiking neurons in MT (population vector). Vector length indicates "confidence" of the direction estimate. For the sake of clarity, only every fourth pixel location is visualized. The resulting turning rate, $\hat{\theta}$, which was mapped to a PWM signal for the robot's servos, is illustrated using a sliding bar, where the position of the triangle indicates the sign and magnitude of the turning rate. The small horizontal line indicates $\hat{\theta} = 0$.

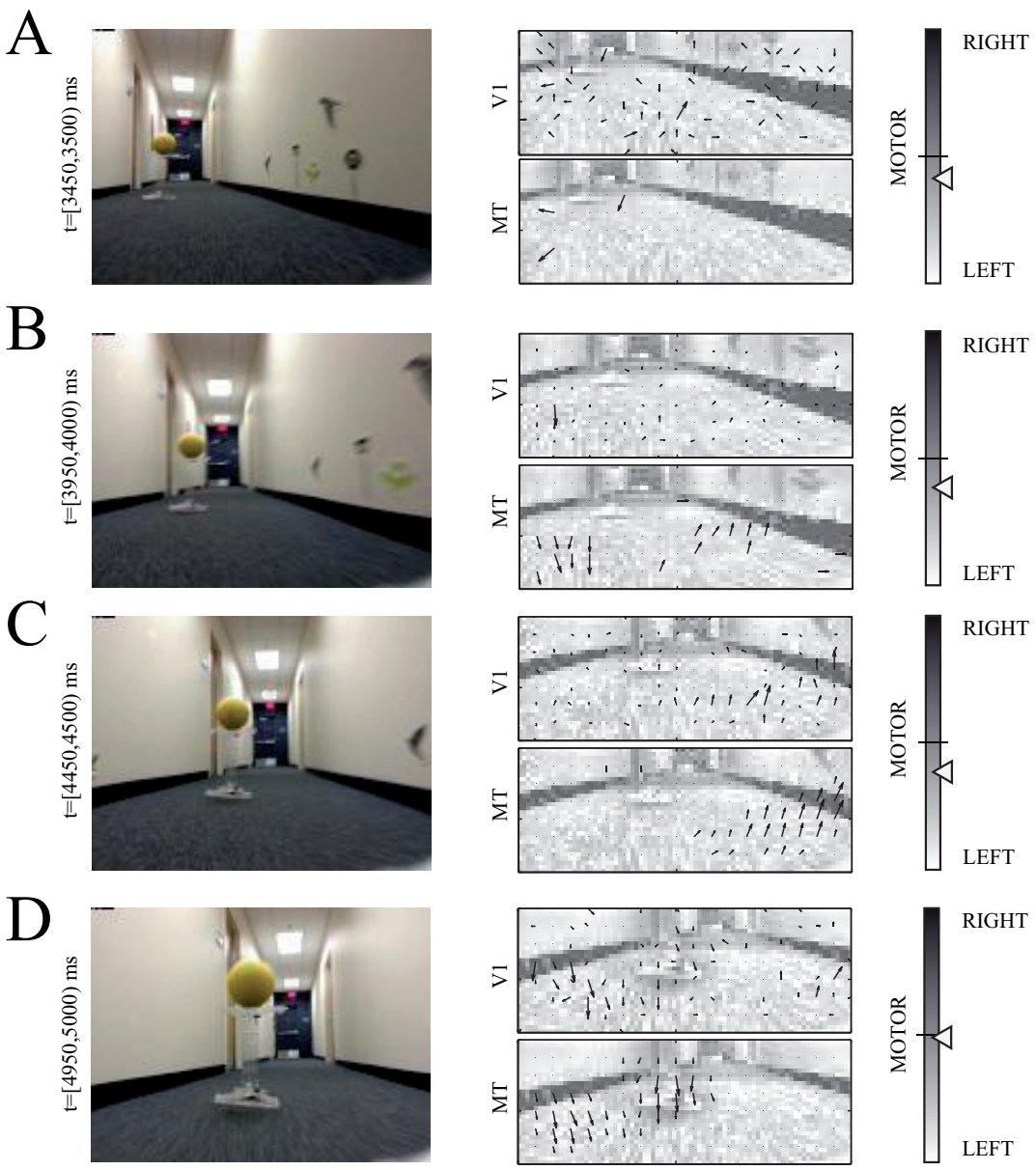

Figure 6.8: Overall network activity and corresponding behavioral output during obstacle avoidance in a single trial. Each Panel summarizes the processing of a single visual input frame, which corresponded to a time interval of 50 ms. The indicated time intervals are aligned with the neuronal traces presented in Fig. 6.6. A frame received at time $t - 50$ ms led to a motor response at time t. Neural activity during these 50 ms is illustrated as an optic flow field, overlaid on visual input, for both Poisson spike generators in V1 and Izhikevich spiking neurons in MT (population vector). Vector length indicates "confidence" of the direction estimate. For the sake of clarity, only every fourth pixel location is visualized. The resulting turning rate, $\hat{\theta}$, which was mapped to a PWM signal for the robot's servos, is illustrated using a sliding bar, where the position of the triangle indicates the sign and magnitude of the turning rate. The small horizontal line indicates $\hat{\theta} = 0$.

fourth pixel location is visualized. The resulting turning rate, $\hat{\theta}$, which was mapped to a PWM signal for the robot's servos, is illustrated using a sliding bar, where the position of the triangle indicates the sign and magnitude of the turning rate. The small horizontal line indicates $\hat{\theta} = 0$. Note that the turning rate depended both on an obstacle term (from optic flow) and a goal term (from color blob detection) (see Section 6.2.2.5).

Obstacle avoidance is illustrated in Fig. 6.7. At the beginning of the trial (Panel A), the image of the obstacle on the retina is too small to produce significant optic flow. Instead, V1 responds to arbitrary features of high contrast in the visual scene. Because these responses were relatively low in magnitude and roughly uniformly distributed across the scene, MT neurons successfully suppressed them, effectively treating them as noise. As a result, the turning rate was near zero, informing the robot to steer straight ahead. In the following second (Panels B–D) the obstacle as well as patterned regions in its vicinity generated an increasingly uniform pattern of motion, leading to strong responses in MT and a strongly positive turning rate. In turn, the initiated robot motion introduced even more apparent motion on the retina, leading to a repeller-like behavioral response and successful obstacle avoidance. Note that it is sometimes possible for the motion signals to slightly extend from the object to neighboring contrast-rich regions, which is due to the strong motion pooling in MT.

Goal-directed steering is illustrated in Fig. 6.8. As soon as the obstacle moved out of the robot's view due to successful avoidance (Panel A), activity in MT rapidly dropped off, causing the robot's turning rate to momentarily decrease. With the goal still in view and growing in size (Panels B–D), turning rate was now increased with opposite sign, eventually overpowering the flow signals that would have instructed the robot to turn away from the goal (Panel D), and instead leading to an attractor-like behavioral response that drew the robot directly to the goal. As the robot navigated to the goal (within a 0.5 m distance; following Panel D), the trial was ended manually.

Overall these results demonstrate how a spiking model of MT can generate sufficient information for the detection of motion boundaries, which can be used to steer a robot around obstacles in a real-world environment.

6.3.4 Computational Performance

The complete cortical model ran in real-time on a single GPU (NVIDIA GTX 780 with 2304 CUDA cores and 3 GB of GDDR5 memory). In fact, during an average trial, most of the visual frames were processed faster than real-time. In order for the network to satisfy the real-time constraint, it must be able to process up to 20 frames per second, which was the upper bound of the ABR client's frame rate. In practice, the effective frame rate might be reduced due to UDP packet loss and network congestion (depending on the quality and workload of the WiFi connection). In other words, the cortical model must process a single frame in no more than 50 ms.

The majority of the computation time was spent on neural processing in V1 and MT. Every 50 ms, model V1 computed a total of $201,600$ filter responses (80×30 pixel image, 28 spatiotemporal filters at three different scales), taking up roughly 15 ms of execution time. Model MT then calculated the temporal dynamics of $40,000$ Izhikevich spiking neurons and roughly $1,700,000$ conductance-based synapses, which took roughly 25 ms of execution time. During this time, a different thread performed color blob detection using OpenCV, which allowed the total execution time to stay under 50 ms. Compared to these calculations, the time it took to perform CLAHE on the GPU was negligible.

6.4 Discussion

I presented a neural network modeled after visual motion perception areas in the mammalian neocortex that is controlling a physical robot performing a visually-guided navigation task in the real world. The system described here builds upon previous work on visual motion perception in large-scale spiking neural networks (Beyeler et al., 2014). The prior system was tested only on synthetic visual stimuli to demonstrate neuronal tuning curves, whereas the present work demonstrates how perception may relate to action. As this is a first step toward an embodied neuromorphic system, we wanted to ensure that the model could handle real sensory input from the real world. Constructing an embodied model also ensured that the algorithm could handle noisy sensors, imprecise motor responses, as well as sensory and motor delays. The model generated a cortical representation of dense optic flow using thousands of interconnected spiking neurons, determined the position of objects based on motion discontinuities, and combined these signals with the representation of a goal location in order to calculate motor commands that successfully steered the robot around obstacles toward to the goal. Therefore, the present study demonstrates how cortical motion signals in a model of MT might relate to active steering control, and suggests that these signals might be sufficient to generate human-like trajectories through a cluttered hallway. This emergent behavior might not only be difficult to achieve in simulation, but also strengthens the claim that MT contributes to these smooth trajectories in natural settings. This finding exemplifies the importance of embodiment, as behavior is deeply coupled not only with the underlying model of brain function, but also with the anatomical constraints of the physical body.

The behavior of the robot was contingent on the quality of a cortical representation of motion generated in a model of visual area MT. While it is generally assumed that vector-based representations of retinal flow in both V1 and MT are highly accurate, modeling work has suggested that the generation of highly accurate flow representations in complex

environments is challenging due to the aperture problem (Baloch and Grossberg, 1997; Bayerl and Neumann, 2004; Chey et al., 1997; Simoncelli and Heeger, 1998). As a result, there is a degree of uncertainty in all retinal flow estimations. Nevertheless, the present model is able to generate optic flow fields of sufficient quality to steer a physical robot on human-like trajectories through a cluttered hallway. Paramount to this competence are the receptive fields of neurons in model MT, which are able to vastly reduce ambiguities in the V1 motion estimates (see Figs. 6.7 and 6.8) and enhance motion discontinuities through spatial pooling and directional opponent inhibition. As a result, the robot is able to correctly infer the relative angle and position of nearby obstacles, implicitly encoded by the spatial arrangement and overall magnitude of optic flow, and produce steering commands that lead to successful avoidance.

It is interesting to consider whether there is a benefit in using spiking neurons instead of rate based neurons. Under the present experimental conditions, the steering network may have worked just as well with a model based on mean-firing rate neurons. However, there is evidence that humans adjust their steering in response not only to spatial but also to temporal asymmetries in the optic flow field (Duchon and Warren, 2002; Kountouriotis et al., 2013). Therefore, having at our disposal a tested, embodied model of brain function that respects the detailed temporal dynamics of neuronal and synaptic integration is likely to benefit follow-up studies that aim to quantitatively investigate how such spatiotemporal behavior can arise from neural circuitry. In addition, because the main functional components of this system were event-driven spiking neurons, the model has the potential to run on neuromorphic hardware, such as HRL (Srinivasa and Cruz-Albrecht, 2012), IBM (Cassidy et al., 2014), NeuroGrid (Boahen, 2006), SpiNNaker (Khan et al., 2008), and BrainScaleS (Schemmel et al., 2010). Some of these platforms are now capable of emulating neural circuits that contain more than a million neurons, at rates that are significantly faster than real time. In addition, because spiking neurons communicate via the AER protocol, the model could also be interfaced with an event-based, neuromorphic vision sensor as a sensory front-

end (Lichtsteiner et al., 2008). Thus, developing complex spiking networks that display cognitive functions or learn behavioral abilities through autonomous interaction may also represent an important initial step into realizing functional large-scale networks on neuromorphic hardware.

Additionally, we have developed software to further extend the functionality of the ABR platform (Oros and Krichmar, 2013b). A set of tools and interfaces exists that provides a standard interface for non-experts to bring their models to the platform, ready for exploration of networked computation principles and applications. The fact that our platform is open source, extensible, and affordable makes ABR highly attractive for embodiment of brain-inspired models. For more information and software downloads, see our website: `www.socsci.uci.edu/~jkrichma/ABR`.

6.4.1 Neurophysiological Evidence and Model Alternatives

Human behavioral data suggests that visually-guided navigation might be based on several possible perceptual variables, which can be flexibly selected and weighted depending on the environmental constraints (Kountouriotis et al., 2013; Morice et al., 2010). In the case of steering to a stationary goal, evidence suggests that human subjects rely on both optic flow, such that one aligns the heading specified by optic flow with the visual target (Gibson, 1958; Warren et al., 2001), and egocentric direction, such that one aligns the locomotor axis with the egocentric direction of the goal (Rushton et al., 1998). Optic flow tends to dominate when there is sufficient visual surface structure in the scene (Li and Warren, 2000; Warren et al., 2001; Wilkie and Wann, 2003), whereas egocentric direction dominates when visual structure is reduced (Rushton et al., 1998, 2002). Interestingly, optic flow asymmetries are able to systematically bias the steering of human subjects, even in the presence of explicit path information (Kountouriotis et al., 2013). This finding may hint at the possibility of a

cortical analog to the Balance Strategy (Duchon and Warren, 2002; Srinivasan and Zhang, 1997), which suggests that humans adjust their steering in response to both spatial and temporal asymmetries in the optic flow field. However, the present model differs in an important way from the traditional Balance Strategy, in that it does not try to balance a conventional optic flow field. Instead, the model considers only signals generated from motion discontinuities (due to motion processing in MT) in the balance equation, which seems to have similar functional implications as the steering potential function from (Huang et al., 2006), generating a repeller signal that gets stronger the closer the robot gets to the obstacle.

Visual self-motion is processed in a number of brain areas located in the Intraparietal Sulcus (IPS) and cingulate sulci, including MST, the VIP region, and the Cingulate Sulcus Visual Region (CSv) (Wall and Smith, 2008). Lesions to the IPS lead to navigational impairments when retracting a journey shown from an egocentric viewpoint (Seubert et al., 2008). There is evidence that object position and velocity might be encoded by a pathway that includes center-surround cells in MT and cells in MSTv (Berezovskii and Born, 2000; Duffy and Wurtz, 1991b; Tanaka et al., 1993). This object information might then be combined with information about self-motion gathered in MST and VIP from cues such as optic flow, eye rotations, and head movements (Andersen et al., 1985; Bradley et al., 1996; Colby et al., 1993; Duffy and Wurtz, 1991a). How neural processing in these regions could relate to steering control has been modeled in detail elsewhere (Browning et al., 2009a; Elder et al., 2009).

The present study demonstrates that it might not be necessary to explicitly model these areas in order to explain human psychophysics data about visually-guided steering and obstacle avoidance. In the case of a robot with a single stationary camera, there is no need to calculate eye or head rotations. As a result of this, there is no need for coordinate transformation, because retinotopic coordinates are equivalent to craniotopic coordinates.

These simplified anatomical constraints allow for gross simplification of the underlying neural network model without restricting behavioral performance, at least under the present experimental conditions.

6.4.2 Model Limitations

Although the present model is able to generate human-like steering paths for a variety of experimental setups, it does not attempt to explain how the direction of travel is extracted from optic flow, how these signals are made independent of eye rotations and head movements, and how these signals can be converted into a craniotopic reference frame. However, these processes would need to be modeled if one were to consider a robot that could freely move around its head (or its eye). A suitable future study could investigate how these computations and transformations could be achieved for a robot that could freely move around its head (or its eye), for example via a pan/tilt unit. The resulting cortical model might not only reproduce the present behavioral results, but also lead to robust behavior in more complicated experimental setups.

As mentioned above, the behavioral model by Fajen and Warren (2003) computes a steering trajectory using third-person information obtained directly from the scene geometry, namely the distance and angles to the goal and obstacle. In contrast, our model steers from a first-person view using active local sensing, which cannot directly compute the distance and angle to the goal and obstacle. Instead, distance is implicitly represented by neural activity, that is, the overall magnitude of the goal and activity related to obstacles. This is based on the assumption that perceived object size scales inversely with distance. While this assumption is generally true, it might not necessarily enable the model to generalize to arbitrarily complex settings. For example, the model might try to avoid a large but far away obstacle with the same dynamics as it would be a small but close-by one. In this case, an accurate estimate of

object distance would be imperative. Also, in the current setup it is sometimes possible that motion signals caused by the obstacle can extend into contrast-rich neighboring regions, due to strong motion pooling in MT. A future study could alleviate this issue by giving the robot a means to establish clearer boundaries between objects and their surroundings, perhaps by modeling motion-form interactions (Baloch and Grossberg, 1997).

Binocular vision is an important source of depth perception that influences object segmentation in depth (Xiao et al., 1997) as well as the pre-planning and on-line control of movement (Patla and Vickers, 1997). Therefore it is possible that binocular vision might significantly benefit the task at hand (Pauwels et al., 2010). A suitable follow-up study would thus be to quantify the contribution of depth perception to the quality of stimulus responses in areas MT as well as to the behavioral performance of the robotic agent. It is interesting to note that most MT cells in the macaque receive balanced input from both eyes (Maunsell and van Essen, 1983), but it turns out that neurons in (at least) macaque MT are not strongly tuned to motion in depth; i.e., no units are truly activated for stimuli changing disparity, which would simulate trajectories with components of motion toward or away from the animal (Maunsell and van Essen, 1983). Therefore, extending the single camera version of the model by adding areas important for segmenting a scene and recognizing objects (Baloch and Grossberg, 1997), would also improve planning trajectories through space. Because vergence is not a factor for distances on the order of meters, scene contributions from brain areas such as V4, MST, and parietal cortex are probably important for long-range trajectory planning. Moreover, these areas strongly interact with area MT.

Also, evidence suggests that humans use a different behavioral strategy when it comes to intercepting moving targets (Fajen and Warren, 2003). Having an agent steer toward a moving target according to the present behavioral dynamics is likely to result in pursuit behavior (as opposed to interception behavior), in which the agent always lags behind the target. Thus we do not expect the model to generalize to moving targets. However, it might

be possible to extend the model to account for these scenarios. The Superior Parietal Lobe (SPL) might be involved in encoding future path information, such as the location of targets and obstacles, which are indicative of impending changes in heading and using this for the purpose of accurately timing motor responses (Billington et al., 2013; Field et al., 2007). Computing and updating of object locations in space, which might be relevant especially for moving targets, might be done by a network involving the precuneus and dorsal precentral gyrus (Leichnetz, 2001; Parvizi et al., 2006). However, not much is known about the neuronal representation of these signals.

6.4.3 Practical Implications

The present work might be of interest to the neuroscientist, neurorobotics, and neuromorphic engineering communities for the following reasons.

First, we have shown that the present approach is practical for studying the link between neural circuitry and behavioral performance. The real-world consequences of neural activity can be observed in real-time through visualization software and analyzed off-line. The behavioral responses were comparable to psychophysical data and the neuronal responses were comparable to neurophysiological data. We believe this to be a powerful approach to studying models of neural circuitry in real-world environments.

Second, we have shown that the present system can handle neural network models of non-trivial size. By making use of the CUDA programming framework, we were able to accelerate computationally intensive parts of the model on a GPU. Every 50 ms, model V1 computed a total of $201,600$ filter responses (80×30 pixel image, 28 spatiotemporal filters at three different scales), and model MT calculated the temporal dynamics of $40,000$ Izhikevich spiking neurons and roughly $1,700,000$ conductance-based synapses. If necessary, the execution of the model could be sped up further, for example by parallelizing processing in the goal and

obstacle components of the model.

Third, implementing the entire model using spiking neurons would make the model amenable to emulation on neuromorphic hardware (Cassidy et al., 2014; Schemmel et al., 2010; Srinivasa and Cruz-Albrecht, 2012). This could enable the development of a self-contained, fully autonomous neurorobotic platform that combines the algorithmic advantages of the present model with the speed, efficiency, and scalability of neuromorphic hardware. In addition, since spiking neural networks communicate via the AER protocol, the model could be interfaced with an event-based, neuromorphic camera as a sensory front-end (Lichtsteiner et al., 2008). This would make it possible for the cortical network to react more quickly (at least on the millisecond scale, but theoretically even on the microsecond scale) to temporal changes in optic flow. It would also open the door for a complete neuromorphic vision system that operates with low power and rapid responses.

Fourth, the system presented here is open source, extensible, and affordable. The complete ABR source code is hosted on GitHub (`www.github.com/UCI-ABR`), and CARLsim can be freely obtained from our website (`www.socsci.uci.edu/jkrichma/CARLsim`). The ABR framework can be combined with a variety of R/C based vehicles, sensors, actuators, and C/C++ based software components. For example, it is straightforward to plug additional software components (such as image transformation with OpenCV) into the client-server loop, or mount IR sensors on the robot and read out the sensory values directly with the ABR client software. We also provide online instructions and video tutorials to assemble the ABR platform, which has an estimated cost of $200 (excluding the phone). Because of this, we believe ABR to be an attractive platform for students and researchers alike that will simplify both the development of neurorobotics platforms as well as the study of how neural machinery can be used to realize cognitive function.

Chapter 7

3D Visual Response Properties of MSTd Emerge from an Efficient, Sparse Population Code

7.1 Introduction

Following information processing in MT, visual signals are sent in parallel to a number of neighboring regions, including the dorsal subregion of the Medial Superior Temporal area (MSTd) for self-motion processing, the ventral subregion of the Medial Superior Temporal area (MSTv) for processing of trajectories of moving objects, and the Floor of the Superior Temporal visual area (FST) for processing of actions and motions of animate entities (Orban, 2008) (see Section 2.2.4). In this chapter I focus on the processing of visual self-motion cues in area MSTd.

Neurons in the dorsal subregion of the Medial Superior Temporal area (MSTd) of monkey extrastriate cortex respond to relatively large and complex patterns of retinal flow, often

preferring a mixture of translational, rotational, and to a lesser degree deformational flow components (Duffy and Wurtz, 1991a; Lagae et al., 1994; Mineault et al., 2012; Orban et al., 1992; Saito et al., 1986; Tanaka and Saito, 1989). This has led researchers to suggest that MSTd might play a key role in visual motion processing for self-movement perception. In fact, one of the most commonly documented response properties of MSTd neurons is that of heading selectivity (Britten and Wezel, 2002; Duffy and Wurtz, 1995; Gu et al., 2006, 2012; Lappe et al., 1996; Logan and Duffy, 2006; Page and Duffy, 2003; Tanaka and Saito, 1989), and computational models can account for this finding (Beintema and van den Berg, 1998, 2000; Lappe et al., 1996; Perrone, 1992; Perrone and Stone, 1994, 1998; Zemel and Sejnowski, 1998; Zhang et al., 1993). However, a number of recent studies have found that nearly all visually responsive neurons in macaque MSTd signal the three-dimensional (3D) direction of self-translation (i.e., "heading") (Gu et al., 2006, 2012; Logan and Duffy, 2006) and self-rotation (Takahashi et al., 2007) in response to both simulated and actual motion. In contrast to earlier studies that concentrated on a small number of MSTd neurons preferring fore-aft (i.e., forward and backward) motion directions (e.g., Duffy and Wurtz (1995)), these more recent studies demonstrated that heading preferences in MSTd were distributed throughout the spherical stimulus space, and that there was a significant predominance of cells preferring lateral as opposed to fore-aft headings. However, little is known about the underlying computational principles by which these response properties are derived.

In this chapter I describe a computational model of MSTd based on the hypothesis that neurons in MSTd efficiently encode the continuum of large-field retinal flow patterns encountered during self-movement on the basis of inputs received from neurons in MT. Nonnegative Matrix Factorization (NMF) (Lee and Seung, 1999, 2001; Paatero and Tapper, 1994), a linear dimensionality reduction technique, is used to find a set of nonnegative basis vectors, which are interpreted as MSTd-like receptive fields. NMF is similar to Principal Component Analysis (PCA) and Independent Component Analysis (ICA), but unique among these dimensionality reduction techniques in that it can recover representations that are often sparse

and parts-based, much like the intuitive notion of combining parts to form a whole (Lee and Seung, 1999). I aim to show that this computational principle can account for a wide range of MSTd visual response properties, ranging from single-unit selectivity to population statistics (such as 3D translation and rotation selectivity, spiral tuning, and heading selectivity), and that seemingly contradicting findings in the literature can be resolved by eliminating differences in neuronal sampling procedures.

7.2 Methods

The overall architecture of the model is depicted in Fig. 7.1. Visual input to the system encompassed a range of idealized two-dimensional (2D) flow fields caused by observer translations and rotations in a three-dimensional (3D) world. We sampled flow fields that mimic natural viewing conditions during locomotion over ground planes and towards back planes located at various depths, with various linear and angular observer velocities, to yield a total of S flow fields, comprising the input stimuli. Each flow field was processed by an array of F MT-like motion sensors (MT-like model units), each tuned to a specific direction and speed of motion. The activity values of the MT-like model units were then arranged into the columns of an $F \times S$ matrix, $\mathbf{V}$, which served as input for Nonnegative Matrix Factorization (NMF) (Lee and Seung, 1999, 2001; Paatero and Tapper, 1994). NMF belongs to a class of dimensionality reduction methods that can be used to linearly decompose a multivariate data matrix, $\mathbf{V}$, into an inner product of two reduced-rank matrices, $\mathbf{W}$ and $\mathbf{H}$, such that $V \approx WH$. The first of these reduced-rank matrices, $\mathbf{W}$, contains as its columns a total of B nonnegative basis vectors of the decomposition. The second matrix, $\mathbf{H}$, contains as its rows the contribution of each basis vector in the input vectors (the hidden coefficients). These two matrices are found by iteratively reducing the residual between $\mathbf{V}$ and $\mathbf{WH}$ using an alternating nonnegative least-squares method. In our experiments, the only open parameter

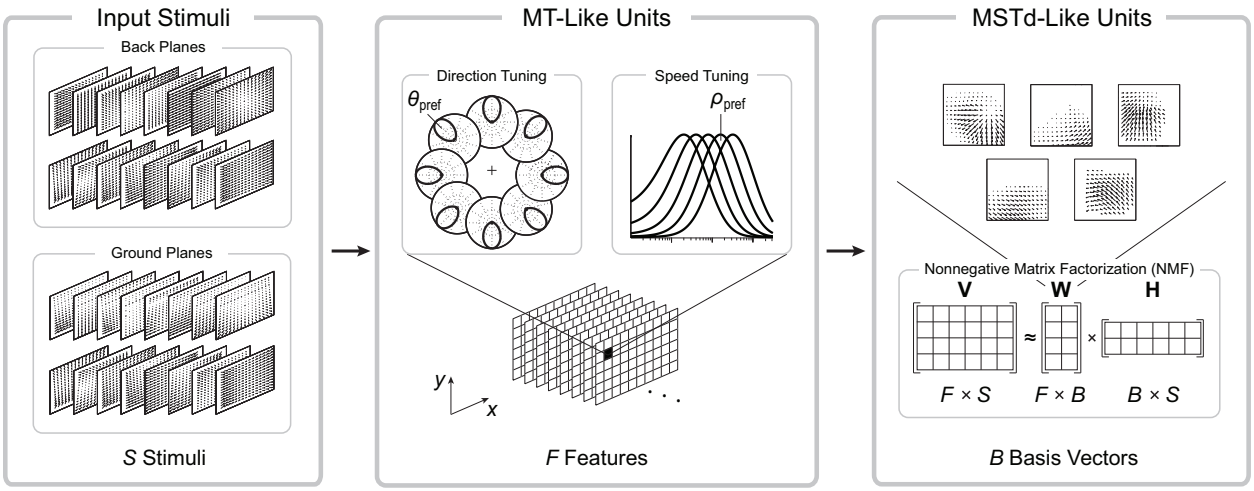

Figure 7.1: Overall model architecture. A number S of 2D flow fields depicting observer translations and rotations in a 3D world were processed by an array of F MT-like motion sensors, each tuned to a specific direction and speed of motion. MT-like activity values were then arranged into the columns of a data matrix, $\mathbf{V}$, which served as input for Nonnegative Matrix Factorization (NMF). The output of NMF were two reduced-rank matrices, $\mathbf{W}$ (containing B nonnegative basis vectors) and $\mathbf{H}$ (containing hidden coefficients). Columns of $\mathbf{W}$ (basis vectors) were then interpreted as weight vectors of MSTd-like model units.

of the NMF algorithm was the number of basis vectors, B. We interpreted the columns of $\mathbf{W}$ as the weight vectors of a total of B MSTd-like model units. Each weight vector had F elements representing the synaptic weights from the array of MT-like model units onto a particular MSTd-like model unit. The response of an MSTd-like model unit was given as the dot product of the F MT-like unit responses to a particular input stimulus and the corresponding nonnegative synaptic weight vector, W. Crucially, once the weight matrix W was found all parameter values remained fixed across all experiments. The following subsections explain the model in detail.

7.2.1 Optic Flow Stimuli

Input to the model was a computer-generated 15×15 pixel array of simulated optic flow. We simulated the apparent motion on the retina that would be caused by an observer undergoing translations and rotations in a three-dimensional (3D) world using the motion field model

(Longuet-Higgins and Prazdny, 1980), where a pinhole camera with focal length, f, is used to project 3D real-world points, $\vec{P} = [X, Y, Z]^t$, onto a 2D image plane, $\vec{p} = [x, y]^t = f/Z[X, Y]^t$ (i.e., the retina). Local motion at a particular position $\vec{p}$ on the image plane was specified by a vector, $\dot{\vec{p}} = [\dot{x}, \dot{y}]^t$, with local direction and speed of motion given as $\tan^{-1}(\dot{y}/\dot{x})$ and $||\dot{\vec{x}}||$, respectively. The vector $\dot{\vec{p}}$ was expressed by the sum of a translational flow component, $\dot{\vec{x_{\mathrm{T}}}} = [\dot{x_{\mathrm{T}}}, \dot{y_{\mathrm{T}}}]^t$, and a rotational flow component, $\dot{\vec{x_{\mathrm{R}}}} = [\dot{x_{\mathrm{R}}}, \dot{y_{\mathrm{R}}}]^t$:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \dot{x_{\mathrm{T}}} \\ \dot{y_{\mathrm{T}}} \end{bmatrix} + \begin{bmatrix} \dot{x_{\mathrm{R}}} \\ \dot{y_{\mathrm{R}}} \end{bmatrix} \tag{7.1}$$

where the translational component depended on the camera's linear velocity, $\vec{v} = [v_x, v_y, v_z]^t$, and the rotational component depended on the camera's angular velocity, $\vec{\omega} = [\omega_x, \omega_y, \omega_z]^t$:

$$\begin{bmatrix} \dot{x_{\mathrm{T}}} \\ \dot{y_{\mathrm{T}}} \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{7.2}$$

$$\begin{bmatrix} \dot{x_{\mathrm{R}}} \\ \dot{y_{\mathrm{R}}} \end{bmatrix} = \frac{1}{f} \begin{bmatrix} xy & -(f^2 + x^2) & fy \\ f^2 + y^2 & -xy & -fx \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{7.3}$$

In our simulations, we set $f = 0.01\,\mathrm{m}$ and $x, y \in [-0.01\,\mathrm{m}, 0.01\,\mathrm{m}]$ (Raudies and Neumann, 2012). The 15×15 optic flow array thus subtended $90^\circ \times 90^\circ$ of visual angle.

We sampled flow fields that mimic natural viewing conditions during locomotion over a ground plane (tilted $\alpha = -30^\circ$ down from the horizontal) and toward a back plane (Fig. 7.2). Linear velocities corresponded to comfortable walking speeds $||\vec{v}|| = \{0.5, 1, 1.5\}$ meters per second, and angular velocities corresponded to common eye rotation velocities for gaze stabilization $||\vec{\omega}|| = 0, \pm 5, \pm 10$ degrees per second (Perrone and Stone, 1994). Movement directions were uniformly sampled from all possible 3D directions (i.e., including backward translations). Back and ground planes were located at distances $d = 2, 4, 8, 16, 32$ meters

from the observer. This interval of depths was exponentially sampled due to the reciprocal dependency between depth and the length of vectors of the translational visual motion field (Eq. 7.2) (Perrone and Stone, 1994). Note that $\dot{\vec{x}}_{\mathrm{T}}$ depends on the distance to the point of interest (Z) (Eq. 7.2), but $\dot{\vec{x}}_{\mathrm{R}}$ does not (Eq. 7.3). The point at which $\dot{\vec{x}}_{\mathrm{T}} = 0$ is referred to as the epipole or Center of Motion (COM). If the optic flow stimulus is radially expanding, as is the case for translational forward motion, the COM is called the Focus of Expansion (FOE). In the absence of rotational flow, the FOE coincides with the direction of travel, or "heading" (Fig. 7.2A). However, in the presence of rotational flow, the FOE appears shifted with respect to the true direction of travel (Fig. 7.2B).

7.2.2 MT Stage

Each virtual flow field was processed by an array of idealized MT-like model units, each selective to a particular direction of motion, θ_{pref}, and a particular speed of motion, ρ_{pref}, at a particular spatial location, (x, y). The activity of each unit, r_{MT}, was given as:

$$r_{\mathrm{MT}}(x, y; \theta_{\mathrm{pref}}, \rho_{\mathrm{pref}}) = d_{\mathrm{MT}}(x, y; \theta_{\mathrm{pref}}) s_{\mathrm{MT}}(x, y; \rho_{\mathrm{pref}}), \tag{7.4}$$

where d_{MT} was the unit's direction response and s_{MT} was the unit's speed response.

A unit's direction tuning was given as a von Mises function based on the difference between the local direction of motion at a particular spatial location, $\theta(x, y)$, and the unit's preferred direction of motion, θ_{pref} (Jazayeri and Movshon, 2006; Mardia, 1972; Swindale, 1998):

$$d_{\mathrm{MT}}(x, y; \theta_{\mathrm{pref}}) = \exp\left(\sigma_\theta\Big(\cos\big(\theta(x, y) - \theta_{\mathrm{pref}}\big) - 1\Big)\right) \tag{7.5}$$

where the bandwidth parameter was $\sigma_\theta = 3$, so that the resulting tuning width (full width at half-maximum) was roughly 90°, consistent with reported values in the literature (Britten and Newsome, 1998).

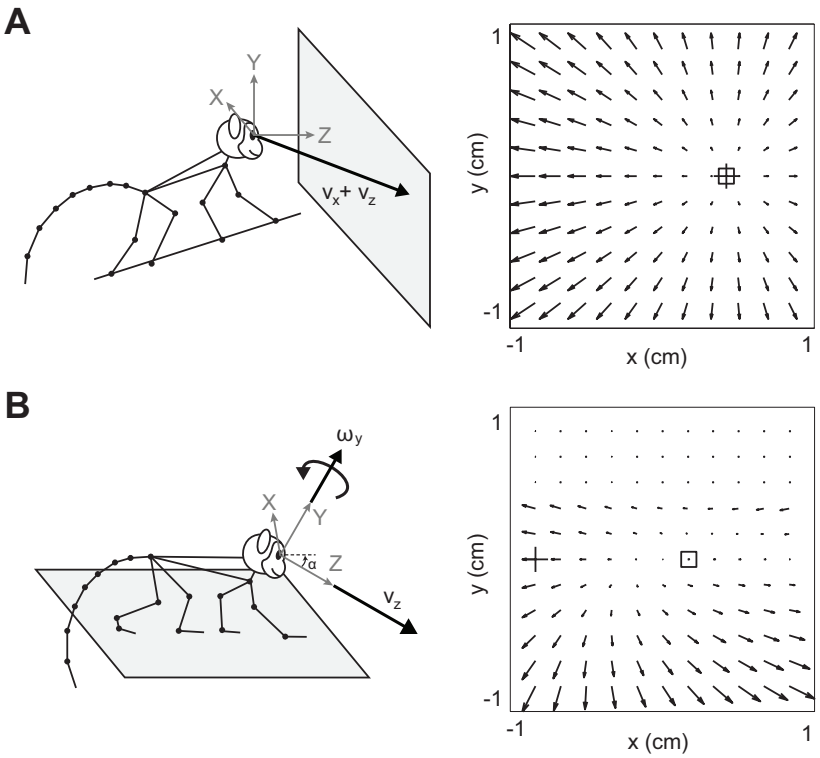

Figure 7.2: Example flow fields generated with the motion field model (Longuet-Higgins and Prazdny, 1980) (modified from Raudies (2013)). We sampled flow fields that mimic natural viewing conditions during upright locomotion toward a back plane (**A**) and over a ground plane (**B**). Gray arrows indicate the axes of the 3D coordinate system, and bold black arrows indicate self-movement (translation, straight arrows; rotation, curved arrows). Crosses indicate the direction of self-movement (i.e., heading), and squares indicate the Center of Motion (COM). In the absence of rotation, the COM indicates heading (**B**). **A**, Example of forward/sideward translation ($v_x = 0.45\,\mathrm{m\,s^{-1}}, v_z = 0.89\,\mathrm{m\,s^{-1}}$) toward a back plane located at a distance $Z(x, y) = 10\,\mathrm{m}$. **B**, Example of curvilinear motion ($v_x = v_z = 0.71\,\mathrm{m\,s^{-1}}$, and yaw rotation $\omega_y = 3\,^\circ\mathrm{s^{-1}}$) over a ground plane located at distance $Z(y) = df/(y\cos(\alpha) + f\sin(\alpha))$, where $d = -10\,\mathrm{m}$ and $\alpha = -30^\circ$.

A unit's speed tuning was given as a log-Gaussian function of the local speed of motion, $\rho(x, y)$, relative to the unit's preferred speed of motion, ρ_{pref} (Nover et al., 2005):

$$s_{\mathrm{MT}}(x, y; \rho_{\mathrm{pref}}) = \exp\left(- \frac{\log^2\left(\frac{\rho(x,y)+s_0}{\rho_{\mathrm{pref}}+s_0}\right)}{2\sigma_\rho^2} \right), \tag{7.6}$$

where the bandwidth parameter was $\sigma_\rho = 1.16$ and the speed offset parameter was $s_0 = 0.33$, both of which corresponded to the medians of physiological recordings (Nover et al., 2005). Note that the offset parameter, s_0, was necessary to keep the logarithm from becoming

undefined as stimulus speed approached zero.

As a result, the population prediction of speed discrimination thresholds obeyed Weber's law for speeds larger than $\sim 5\,^{\circ}\,\mathrm{s}^{-1}$ (Nover et al., 2005). We chose 5 octave-spaced bins and a uniform distribution between $0.5\,^{\circ}\,\mathrm{s}^{-1}$ and $32\,^{\circ}\,\mathrm{s}^{-1}$ (Nover et al., 2005); that is, $\rho_{\mathrm{pref}} = \{2, 4, 8, 16, 32\}$ degrees per second.

At each spatial location there were a total of 40 MT-like model units (selective for eight directions times five speeds of motion), yielding a total of $F = 15 \times 15 \times 8 \times 5 = 9000$ units. The activity pattern of these 40 units per pixel thus acted as a population code for the local direction and speed of motion. We assumed that the receptive fields of all MT-like model units had a single pixel radius (subtending $\sim 3^{\circ}$ of visual angle), which is comparable to receptive field sizes near the fovea as found in macaque MT (Raiguel et al., 1995).

7.2.3 Nonnegative Matrix Factorization (NMF)

We hypothesized that appropriate synaptic weights of the feed-forward projections from MT to MSTd could be learned with NMF (Lee and Seung, 1999, 2001; Paatero and Tapper, 1994). NMF belongs to a class of methods that can be used to decompose multivariate data into an inner product of two reduced-rank matrices, where one matrix contains nonnegative basis vectors and the other contains nonnegative activation vectors (hidden coefficients). The nonnegativity constraints of NMF enforce the combination of different basis vectors to be additive, leading to representations that are often parts-based and sparse (Lee and Seung, 1999). When applied to neural networks, these nonnegativity constraints correspond to the notion that neuronal firing rates are never negative and that synaptic weights are either excitatory or inhibitory, but they do not change sign (Lee and Seung, 1999).

Assume that we observe data in the form of a large number of i.i.d. random vectors, $\vec{v}^{(i)}$,

such as the neuronal activity of a population of MT neurons in response to a visual stimulus, where i is the sample index. When these vectors are arranged into the columns of a matrix $\mathbf{V}$, linear decompositions describe these data as $\mathbf{V} \approx \mathbf{WH}$, where $\mathbf{W}$ is a matrix that contains as its columns the basis vectors of the decomposition. The rows of $\mathbf{H}$ contain the corresponding hidden coefficients that give the contribution of each basis vector in the input vectors. Like Principal Component Analysis (PCA), the goal of NMF is then to find a linear decomposition of the data matrix $\mathbf{V}$, with the additional constraint that all elements of the matrices $\mathbf{W}$ and $\mathbf{H}$ be nonnegative. In contrast to Independent Component Analysis (ICA), NMF does not make any assumptions about the statistical dependencies of $\mathbf{W}$ and $\mathbf{H}$. The resulting decomposition is not exact; $\mathbf{WH}$ is a lower-rank approximation to $\mathbf{V}$, and the difference between $\mathbf{WH}$ and $\mathbf{V}$ is termed the reconstruction error. Perfect accuracy is only possible when the number of basis vectors approaches infinity, but good approximations can usually be obtained with a reasonably small number of basis vectors (Pouget and Sejnowski, 1997).

We used the standard `nnmf` function provided by MATLAB R2014a (MathWorks, Inc.), which using an alternating least-squares algorithm, aims to iteratively minimize the root-mean-squared residual $\mathbf{D}$ between $\mathbf{V}$ and $\mathbf{WH}$:

$$\mathbf{D} = \frac{||\mathbf{V} - \mathbf{WH}||}{\sqrt{FS}} \tag{7.7}$$

where F is the number of rows in $\mathbf{W}$ and S is the number of columns in $\mathbf{H}$ (see Fig. 7.1). $\mathbf{W}$ and $\mathbf{H}$ were normalized so that the rows of $\mathbf{H}$ had unit length. The output of NMF is not unique because of random initial conditions (i.e., random weight initialization). The only open parameter was the number of basis vectors, B, whose value had to be determined empirically. In our simulations we examined a range of values ($B = 2^i$, where $i = \{4, 5, 6, 7, 8\}$, see Section 7.3.2), but found that $B = 64$ led to basis vectors that most resembled receptive fields found in macaque MSTd. In order to facilitate statistical comparisons between model responses and biological data, NMF with $B = 64$ was repeated 14 times with different

random initial conditions to generate a total of $N = 896$ MSTd-like model units.

7.2.4 MSTd Stage

We interpreted the resulting columns of $\mathbf{W}$ as the weight vectors from the population of F MT-like model units onto a population of B MSTd-like model units. The activity of the b-th MSTd-like unit, r^b_{MSTd}, was given as the dot product of all F MT-like responses to a particular input stimulus, i, and the unit's corresponding nonnegative weight vector:

$$r^b_{\text{MSTd}}(i) = \vec{v}^{(i)}\vec{w}^{(b)}, \tag{7.8}$$

where $\vec{v}^{(i)}$ was the i-th column of $\mathbf{V}$ and $\vec{w}^{(b)}$ was the b-th column of $\mathbf{W}$. This is in agreement with the finding that MSTd responses are approximately linear in terms of their feed-forward input from area MT (Duffy and Wurtz, 1991b; Tanaka et al., 1989), which provides one of the strongest projections to MST in the macaque (Boussaoud et al., 1990). In contrast to other models (Grossberg et al., 1999; Lappe et al., 1996; Mineault et al., 2012; Zemel and Sejnowski, 1998), no additional nonlinearities were required to fit the data presented in this work.

7.2.5 3D Translation/Rotation Protocol

In order to determine 3D translation and rotation tuning profiles, MSTd-like model units were probed with two sets of virtual optic flow stimuli as described by Takahashi et al. (2007).

The "translation protocol" consisted of straight translational movements along 26 directions in 3D space, corresponding to all combinations of azimuth and elevation angles in increments of 45° (Fig. 7.3A). This included all combinations of movement vectors having eight different

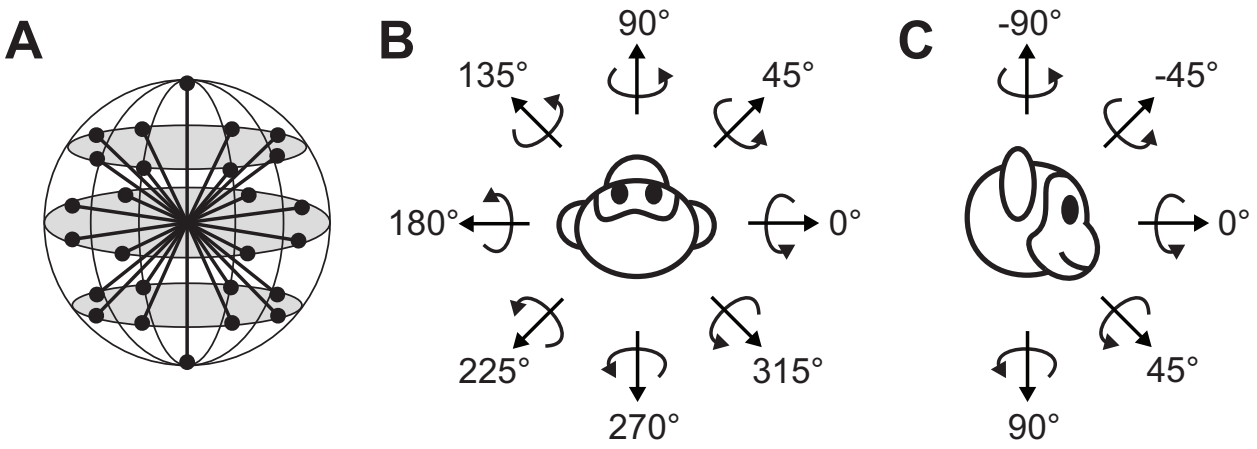

Figure 7.3: Schematic of the 26 translational and rotational directions used to test MSTd-like model units (modified from Takahashi et al. (2007)). **A**, Illustration of the 26 movement vectors, spaced 45° apart on the sphere, in both azimuth and elevation. **B**, Top view: Definition of azimuth angles. **C**, Side view: Definition of elevation angles. Straight arrows illustrate the direction of movement in the translation protocol, whereas curved arrows indicate the direction of rotation (according to the right-hand rule) about each of the movement vectors.

azimuth angles (0, 45°, 90°, 135°, 180°, 225°, 270°, and 315°; Fig. 7.3B), each of which was presented at three different elevation angles (Fig. 7.3): 0 (the horizontal plane) and ±45° (for a sum of $8 \times 3 = 24$ directions). Two additional directions were specified by elevation angles of −90° and 90°, which corresponded to upward and downward movement directions, respectively.

The "rotation protocol" consisted of rotations about the same 26 directions, which now represented the corresponding axes of rotation according to the right-hand rule (Fig. 7.3B, C). For example, azimuths of 0 and 180° (elevation, 0) corresponded to pitch-up and pitch-down rotations, respectively. Azimuths of 90° and 270° (elevation, 0) corresponded to roll rotations (right-ear down and left-ear down, respectively). Finally, elevation angles of −90° and 90° corresponded to leftward or rightward yaw rotation, respectively.

7.2.6 Heading Tuning Index

To quantify the strength of heading tuning of a model unit, we followed a procedure described by Gu et al. (2006) to compute a Heading Tuning Index (HTI). In each trial, the activity of a model unit was considered to represent the magnitude of a 3D vector whose direction was defined by the azimuth and elevation angles of the respective movement trajectory (Gu et al., 2006). A Heading Tuning Index (HTI) was then computed for each model unit using the following equation:

$$\mathrm{HTI}^b = \frac{|\sum_{i=1}^{n} r^b_{\mathrm{MSTd}}(i)\vec{e_i}|}{\sum_{i=1}^{n} |r^b_{\mathrm{MSTd}}(i)\vec{e_i}|} \tag{7.9}$$

where $r^b_{\mathrm{MSTd}}(i)$ was the activity of the b-th model unit for the i-th stimulus, $\vec{e_i}$ was a unit vector indicating the 3D Cartesian heading direction of the i-th stimulus, $|\cdot|$ denoted vector magnitude, and $n = 26$ corresponded to the number of different heading directions tested (see Section 7.2.5). The HTI ranged from 0 to 1, where 0 indicated weak and 1 indicated strong direction tuning. The preferred heading direction for each stimulus was then computed from the azimuth and elevation of the vector sum of the individual responses (numerator in Eq. 7.9).

7.2.7 Uniformity Test

To determine whether a measured distribution was significantly different from uniform, we performed a resampling analysis as described by Takahashi et al. (2007). First, we calculated the sum-squared error (across bins) between the measured distribution and an ideal uniform distribution containing the same number of observations. This calculation was repeated for 1000 different random distribution generated using the `unifrnd` function provided by MATLAB R2014a (MathWorks, Inc.) in order to produce a distribution of sum-squared error values that represent random deviations from an ideal uniform distribution. If the

sum-squared error for the experimentally measured distribution lay outside the 95 % confidence interval of values from the randomized distributions, the measured distribution was considered to be significantly different from uniform ($p < 0.05$) (Takahashi et al., 2007).

7.2.8 Decoding Population Activity

We tested whether perceptual variables such as heading or angular velocity could be decoded from the population activity of MSTd-like model units using simple linear regression. We assembled a data set consisting of 10^4 flow fields with randomly selected headings, which depicted linear observer movement (velocities sampled uniformly between $0.5\,\mathrm{m\,s^{-1}}$ and $2\,\mathrm{m\,s^{-1}}$; no eye rotations) towards a back plane located at various distances $d = \{2, 4, 8, 16, 32\}$ meters away. As part of a ten-fold cross-validation procedure, stimuli were split repeatedly into a training set containing 9000 stimuli and a test set containing 1000 stimuli. Using linear regression, we then obtained a set of $N \times 2$ linear weights used to decode MSTd population activity in response to samples from the training set, and monitored performance on the test set (Picard and Cook, 1984).

For forward headings and in the absence of eye rotations, heading coincides with the location of the FOE on the retina (see Section 7.2.1). Conceptually, the 2D Cartesian coordinates of the FOE can be found by setting $\dot{\vec{x}}_{\mathrm{T}} = 0$ in Eq. 7.2, and solving for (x, y). However, because these coordinates could approach infinity for lateral headings, we only considered cases of headings (in spherical coordinates) with azimuth angles between 45° and 135°, as well as elevation angles between −45° and 45°.

Similarly, we obtained a set of $N \times 2$ linear weights to predict the 2D Cartesian components of angular velocity, $\dot{\vec{x}}_{\mathrm{R}}$. In this case, the data set contained stimuli consisting only of rotational flow components (i.e., $\vec{x_{\mathrm{T}}} = 0$ in Eq. 7.1). In order to compare model performance to neurophysiological studies (Hamed et al., 2003), we randomly selected $N = 144$ model units

from the population and limited rotations to the X–Z plane (i.e., $\omega_y = 0$ in Eq. 7.3).

7.2.9 Sparseness

We computed a sparseness metric for the modeled MSTd population activity according to the definition of sparseness by Vinje and Gallant (2000):

$$s = \left(1 - \frac{1}{N}\frac{(\sum_i r_i)^2}{\sum_i r_i^2}\right) \Big/ \left(1 - \frac{1}{N}\right) \tag{7.10}$$

Here, $s \in [0, 1]$ is a measure of sparseness for a signal r with N sample points, where $s = 1$ denotes maximum sparseness and is indicative of a local code, and $s = 0$ is indicative of a dense code. In order to measure how many MSTd-like model units were activated by any given stimulus (population sparseness), r_i was the response of the i-th neuron to a particular stimulus and N was the number of model units. In order to determine how many stimuli any given model unit responded to (lifetime sparseness), r_i was the response of a unit to the i-th stimulus and N was the number of stimuli. Population sparseness was averaged across stimuli and lifetime sparseness was averaged across units.

7.2.10 Fisher Information Analysis

To investigate whether simulated MSTd population activity could account for the dependence of psychophysical thresholds on reference heading, we followed a procedure described by Gu et al. (2010) to compute Fisher information, I_{F}, which provides an upper limit on the precision with which an unbiased estimator can discriminate small variations in a variable (x) around a reference value (x_{ref}):

$$I_{\mathrm{F}} = \sum_{i=1}^{N} \frac{R_i'(x_{\mathrm{ref}}^2)}{\sigma_i(x_{\mathrm{ref}}^2)}, \tag{7.11}$$

where N denotes the number of neurons in the population, $R_i'(x_{\text{ref}})$ denotes the derivative of the tuning curve for the i-th neuron at x_{ref}, and $\sigma_i(x_{\text{ref}})$ is the variance of the response of the i-th neuron at x_{ref}. Neurons with steeply sloped tuning curves and small variance will contribute most to Fisher information. In contrast, neurons having the peak of their tuning curve at x_{ref} will contribute little.

In order to calculate tuning curve slope, $R_i'(x_{\text{ref}})$, we used a spline function (0.01° resolution) to interpolate among coarsely sampled data points (30° spacing). In order to calculate variance, $\sigma_i(x_{\text{ref}})$, Gu et al. (2010) assumed that neurons have independent noise and Poisson spiking statistics. Because the model units described in this paper are deterministic, we instead calculated signal variance directly from the response variability of the network when presented with random dot clouds generated from a particular 3D heading (150 repetitions).

7.3 Results

7.3.1 3D Translation and Rotation Selectivity

We tested whether individual units in our model of MSTd could signal the 3D direction of self-translation and self-rotation when presented with large-field optic flow stimuli, and compared our results to neurophysiological data from Gu et al. (2006) and Takahashi et al. (2007). Visual stimuli depicted translations and rotations of the observer through a 3D cloud of dots that occupied a space 40 cm deep and subtended $90° \times 90°$ of visual angle (see Section 7.2.5). In the translation protocol, movement trajectories were along 26 directions or "headings" (Fig. 7.3A), corresponding to all combinations of azimuth and elevation angles in increments of 45° (Fig. 7.3B, C; straight arrows). In the rotation protocol, these 26 directions served as rotation axes instead (Fig. 7.3B, C; curved arrows). Linear velocity was $1\,\text{m}\,\text{s}^{-1}$, and angular velocity was $20\,^\circ\,\text{s}^{-1}$ (Takahashi et al., 2007).

Fig. 7.4 shows the 3D translation and rotation tuning of a particular neuron in macaque MSTd (Fig. 7.4A, C) (Takahashi et al., 2007) and of an MSTd-like model unit with similar tuning (Fig. 7.4B, D). The 3D directional tuning profile is shown as a contour map in which activity (mean firing rate for neuron in macaque MSTd; unit activation for MSTd-like model unit, see Eq. 7.1), here represented by color, is plotted as a function of azimuth (abscissa) and elevation (ordinate). Each contour map shows the Lambert cylindrical equal-area projection of the original spherical data, where the abscissa corresponds to the azimuth angle and the ordinate is a sinusoidally transformed version of elevation angle (Snyder, 1987). The peak response of this particular MSTd neuron occurred at 291° azimuth and −18° elevation in the case of rotation (corresponding approximately to a left ear-down roll rotation, with small components of pitch and yaw rotation), and at 190° azimuth and −50° elevation in the case of translation (corresponding to backwards motion) (Takahashi et al., 2007). The peak response of the MSTd-like model unit occurred at 268° azimuth, −21° elevation and 176° azimuth, −41° elevation for rotation and translation, respectively. Typical for MSTd neurons whose visual translation and rotation preferences are linked by the two-dimensional (2D) visual motion selectivity of the cell, peak responses occur at stimulus directions roughly 90° apart on the sphere (Takahashi et al., 2007). Model MSTd appropriately captured this effect.

Fig. 7.5 shows the distribution of direction preferences of MSTd cells (Fig. 7.5A, C) (Gu et al., 2006; Takahashi et al., 2007) and MSTd-like model units (Fig. 7.5B, D) for visual translation and rotation. Each data point in these scatter plots specifies the preferred 3D direction of a single neuron or model unit. Histograms along the boundaries show the marginal distributions of azimuth and elevation preferences. The direction preference (for rotation and translation, separately) was defined by the azimuth and elevation of the vector average of the neural responses (see Section 7.2.5). The distributions of azimuth and elevation preference were significantly nonuniform for both rotation and translation conditions ($p < 0.05$, see Section 7.2.7), tightly clustered around 0 and 180° azimuth as well as −90° and 90° elevation.

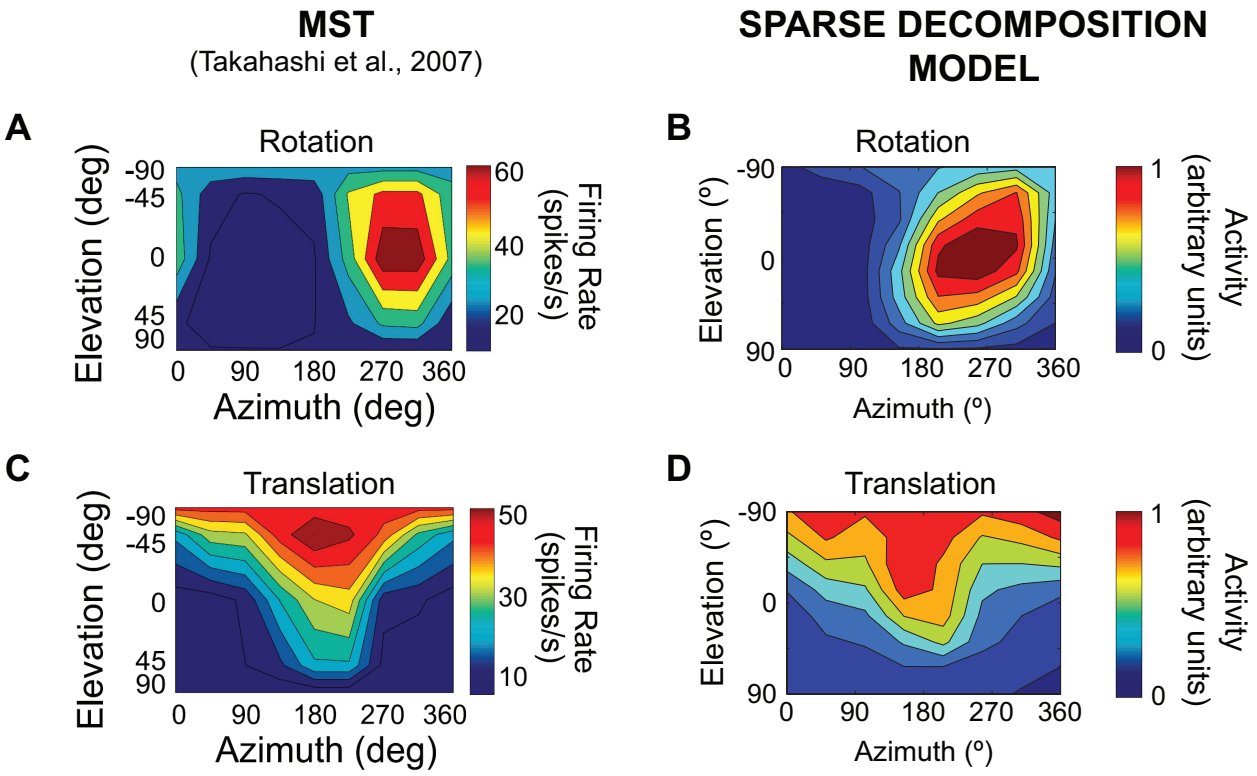

Figure 7.4: Example of 3D direction tuning for an MSTd neuron (rotation, **A**; translation, **C**), reprinted from Takahashi et al. (2007) (their Fig. 4), and a similarly tuned MSTd-like model unit (rotation, **B**; translation, **D**). Color contour maps show the mean firing rate or model activation as a function of azimuth and elevation angles. Each contour map shows the Lambert cylindrical equal-area projection of the original data, where the abscissa corresponds to the azimuth angle and the ordinate is a sinusoidally transformed version of elevation angle (Snyder, 1987).

In agreement with macaque MSTd, the majority of all 896 MSTd-like model units had rotation preferences within 30° of the yaw or pitch axes, and only few model unit had their rotation preference within 30° of the roll axis (Table 7.1). For translation, the majority of direction preferences were within 30° of the lateral or vertical axes, with only a handful of fore-aft direction preferences (Table 7.2).

Table 7.1: Percentage of neurons and model units with preferred rotation directions within ±30° of each of the cardinal axes.

Visual rotation	Yaw	Pitch	Roll
Takahashi et al. (2007)	36/127 (28 %)	27/127 (21 %)	1/127 (1 %)
This work	216/896 (24 %)	330/896 (37 %)	4/896 (1 %)

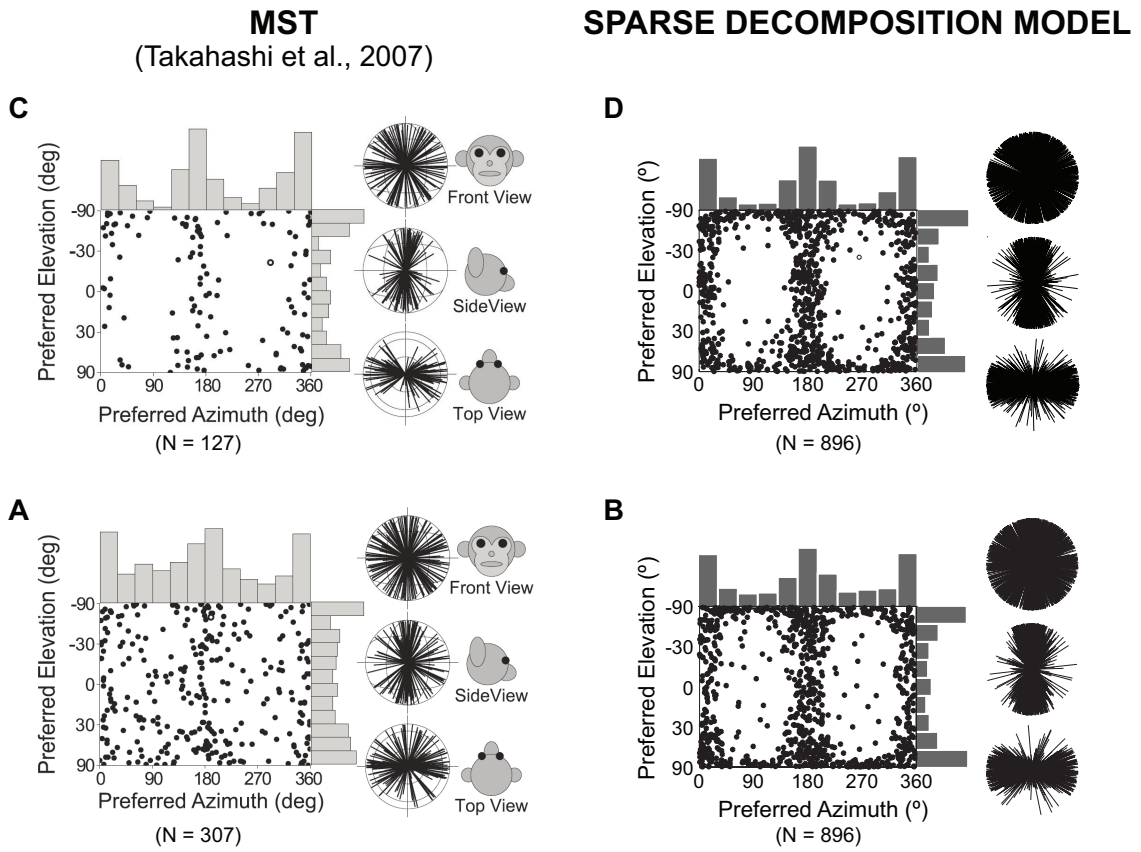

Figure 7.5: Distribution of 3D direction preferences of MSTd neurons (rotation, **A**; translation, **C**), reprinted from Takahashi et al. (2007) (their Fig. 6), and the population of MSTd-like model units (rotation, **B**; translation, **D**). Each data point in the scatter plot corresponds to the preferred azimuth (abscissa) and elevation (ordinate) of a single neuron or model unit. Histograms along the top and right sides of each scatter plot show the marginal distributions. Also shown are 2D projections (front view, side view, and top view) of unit-length 3D preferred direction vectors (each radial line represents one neuron or model unit). The neuron in Fig. 7.4 is represented as open circles in each panel.

We also quantified the strength of heading tuning in model MSTd using a Heading Tuning Index (HTI) (Gu et al., 2006), which ranged from 0 to 1 indicating poor and strong tuning, respectively (see Section 7.2.6). For reference, a model unit with idealized cosine tuning would have an HTI value of 0.31, whereas an HTI value of 1 would be reached when firing rate is zero for all but a single stimulus direction. In the translation condition, Gu et al. (2006) reported HTI values averaging 0.48 ± 0.16 SD for their sample of 251 MSTd cells. Model MSTd achieved very similar HTI values averaging 0.43 ± 0.11 SD and 0.47 ± 0.11 SD in the translation and rotation conditions, respectively.

Table 7.2: Percentage of neurons and model units with preferred translation directions within ±30° of each of the cardinal axes.

Visual rotation	Lateral	Fore-aft	Vertical
Takahashi et al. (2007)	57/307 (19 %)	20/307 (7 %)	76/307 (25 %)
This work	245/896 (27 %)	5/896 (1 %)	192/896 (21 %)

In addition, Takahashi et al. (2007) tested a subset of MSTd cells with both the rotation and translation protocols in order to directly compare the difference in 3D direction preference (|Δ preferred direction|) between rotation and translation (Fig. 7.6A). The distribution was strongly nonuniform with a mode near 90°. The simulated MSTd units appropriately captured this distribution (Fig. 7.6B). However, because |Δ preferred direction| is computed as the smallest angle between a pair of preferred direction vectors in three dimensions, it is only defined between 0 and 180°. Therefore, the observed peak near 90° in Fig. 7.6A, B could be derived from a single mode at 90° or from two modes at ±90°. Only the former but not the latter would indicate that the visual preferences for translation and rotation are linked via the 2D visual motion selectivity of the cell or model unit (Takahashi et al., 2007). Therefore, we also illustrate the differences in 2D direction preferences by projecting each 3D preference vector onto the three cardinal planes: X–Y (front view), Y–Z (side view), and X–Z (top view) for both MSTd (Fig. 7.6C) and model MSTd (Fig. 7.6D). Since some planar projections might be small in amplitude, we also calculated the ratio of the lengths of each difference vector in 2D and 3D and plotted them against the 2D preferred direction difference (Fig. 7.6E, F). Consistent with macaque MSTd, the projections of the visual difference vector onto the X–Z and Y–Z planes were relatively small (Fig. 7.6E, F; green and blue data points). In contrast, projections onto the X–Y plane were notably larger (Fig. 7.6E, F; red data points) and tightly centered at −90° (Fig. 7.6C, D), with no cells or model units having direction differences of 90° between visual translation and rotation. Thus, these simulated data are consistent with the idea that preferred directions for translation and rotation are related through the 2D visual motion selectivity of neurons in MSTd.

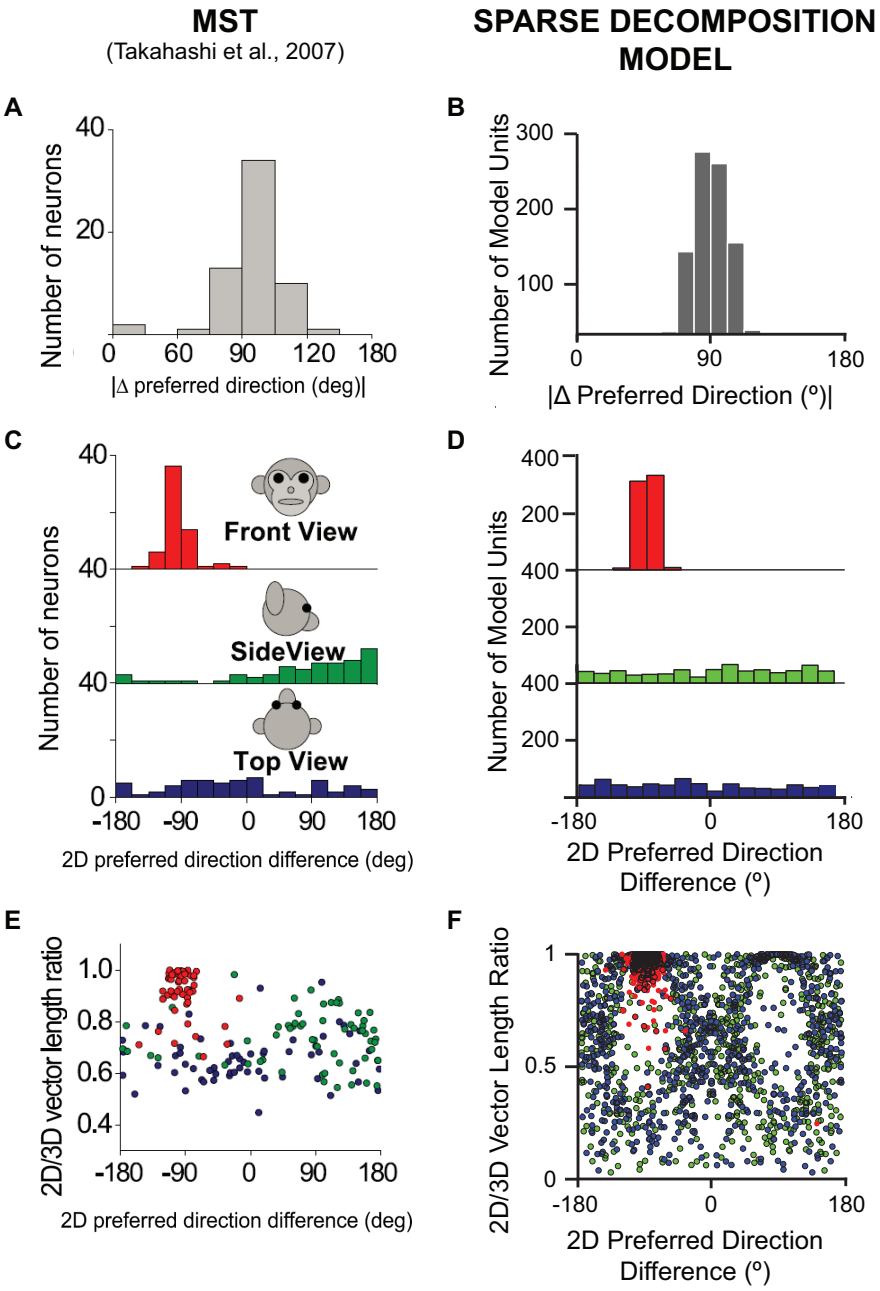

Figure 7.6: Direction differences between rotation and translation, for MSTd neurons (**A**, **C**, **E**), reprinted from Takahashi et al. (2007) (their Fig. 6), and the population of MSTd-like model units (**B**, **D**, **F**). **A**, **B**, Histograms of the absolute differences in 3D preferred direction (|Δ preferred direction|) between rotation and translation. **C**, **D**, Distributions of preferred direction differences as projected onto each of the three cardinal planes, corresponding to front view, side view, and top view. **E**, **F**, The ratio of the lengths of the 2D and 3D preferred direction vectors is plotted as a function of the corresponding 2D projection of the difference in preferred direction (red, green, and blue circles for each of the front view, side view, and top view data, respectively).

7.3.2 Efficient Encoding of Multiple Perceptual Variables

MSTd activity plays a causal role in the perception of heading from optic flow (Gu et al., 2010, 2012). During forward movement, retinal flow radiates out symmetrically from a single point, the FOE, from which heading can be inferred (Gibson, 1950). Page and Duffy (1999) found that the location of the FOE could be decoded to a very high degree of precision (i.e., within $\pm 10^\circ$) from the trial-averaged population response of neurons in MSTd. Hamed et al. (2003) then went on to show that the FOE in both eye-centered and head-centered coordinates could be decoded from MSTd population activity with an optimal linear estimator even on a single-trial basis, with an error of $0.5 - 1.5^\circ$ and SD of $2.4 - 3^\circ$. Interestingly, they found that other perceptual variables such as eye position and the direction of ocular pursuit could be decoded with similar error rates, and that most MSTd neurons were involved in the simultaneous encoding of more than one of these variables (Hamed et al., 2003).

We therefore wondered whether the 2D coordinates $(x_{\text{FOE}}, y_{\text{FOE}})$ of the FOE of arbitrary expansive flow fields could be decoded from a population of N MSTd-like model units with similar precision. To answer this question, we assembled a data set of 10^4 flow fields with randomly selected headings (azimuth between 45° and 135°, elevation between -45° and 45°), depicting linear observer movements towards a back plane located at various distances, $d = \{1, 2, 4, 8\}$, in front of the observer. Using simple linear regression in a cross-validation procedure (see Section 7.2.8), we obtained a set of $N \times 2$ linear weights used to decode MSTd population activity in response to samples from the training set. We then tried to predict heading in flow samples from the test set (i.e., headings the network had not seen before) using the learned weights, and assessed prediction error rates. The same procedure was repeated for a set of rotational flow fields designed to mimic visual consequences of slow, pursuit-like eye movements (i.e., restricted to the frontoparallel plane, $\omega_y = 0$). Here, the goal of the network was to predict the 2D Cartesian components of angular velocity (ω_x, ω_z).

Table 7.3: Sparse population code for perceptual variables ($N = 144$).

	FOE (x, y)	Eye velocity (ω_x, ω_y)
Hamed et al. (2003)	$(3.62^\circ \pm 6.78^\circ, 3.87^\circ \pm 4.96^\circ)$	$(1.39^\circ \pm 1.38^\circ \pm 3.02^\circ)$
This work	$(5.75^\circ \pm 5.62^\circ, 6.02^\circ \pm 5.51^\circ)$	$(0.82^\circ \pm 0.89^\circ, 0.92^\circ \pm 0.99^\circ)$

The results are summarized in Table 7.3. Both heading and eye velocity could be inferred from the population activity of model MSTd ($N = 144$) with error rates on the order of $0.1^\circ - 1^\circ$, consistent with neurophysiological data (Hamed et al., 2003; Page and Duffy, 2003). Note that these numbers were achieved on the test set; that is, on flow fields the model had never seen before. Heading prediction error on the test set thus corresponded to the model's ability to generalize; that is, to deduce the appropriate response to a novel stimulus using what it had learned from other previously encountered stimuli (Hastie et al., 2009; Spanne and Jorntell, 2015).

What might be the nature of the population code in MSTd that underlies the encoding of heading and eye velocity? One possibility would be that MSTd contains a set of distinct subpopulations, each specialized to encode a particular perceptual variable. Instead, Hamed et al. (2003) found that neurons in MSTd acted more like basis functions, where a majority of cells was involved in the simultaneous encoding of multiple perceptual variables. A similar picture emerged when we investigated the involvement of MSTd-like model units in the encoding of both heading and simulated eye rotation velocity (Fig. 7.7A). A model unit was said to contribute to the encoding of a perceptual variable if both of its linear decoding weights exceeded a certain threshold, set at 1 % of the maximum weight magnitude across all $N \times 2$ weight values. If this was true for both heading (i.e., FOE) and eye velocity (i.e, pursuit, P), the unit was said to code for both (labeled "FOE&P" in Fig. 7.7A), which was the case for 57 % of all units. Analogously, if the weights exceeded the threshold for only variable or the other, the unit was labeled either "FOE" or "P". Only a few units did not contribute at all (labeled "none".

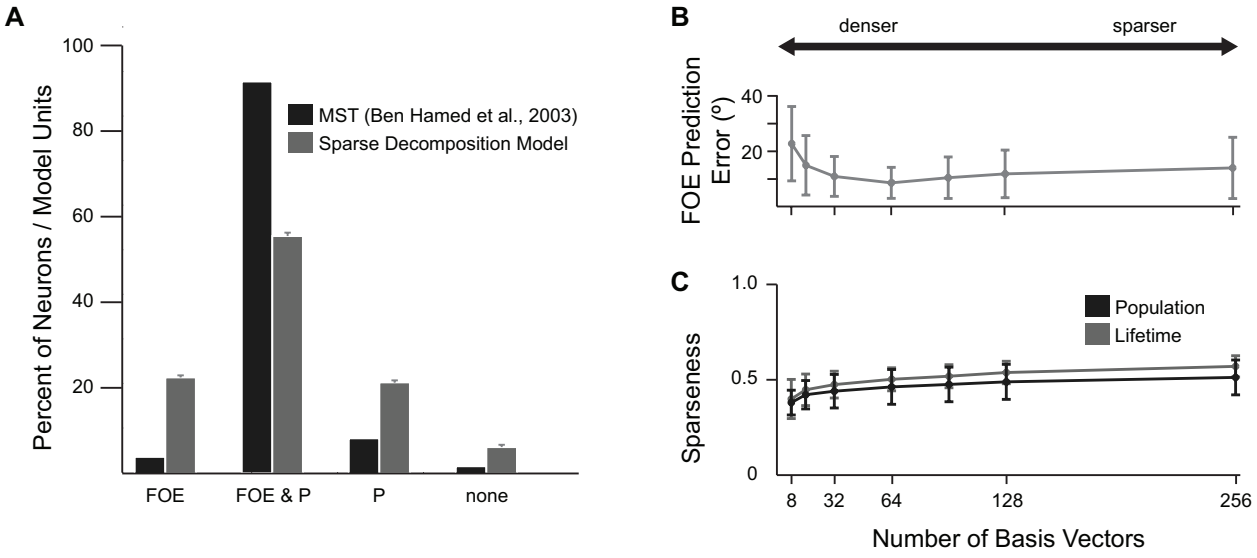

Figure 7.7: Population code underlying the encoding of perceptual variables such as heading (focus of expansion, FOE) and eye velocity (pursuit, P). **A**, Distribution of FOE and pursuit selectivities in MSTd (dark gray), adapted from Hamed et al. (2003) (their Fig. 3), and in the population of MSTd-like model units (light gray). Neurons or model units were involved in encoding either heading ("FOE"), eye velocity ("P"), both ("EYE & P"), or neither ("none"). **B**, Heading prediction (generalization) error as a function of the number of basis vectors using ten-fold cross-validation. Vertical bars are the SD. **C**, Population and lifetime sparseness as a function of the number of basis vectors. Operating the sparse decomposition model with $B = 64$ basis vectors co-optimizes for both accuracy and efficiency of the encoding, and leads to basis vectors that resemble MSTd receptive fields.

Finally, we asked how the accuracy and efficiency of the encoding would change with the number of basis vectors (i.e., the only open parameter of NMF, see Section 7.2.3). In order to determine accuracy, we repeated the heading decoding experiment for a number of basis vectors $B = 2^i$, where $i \in \{4, 5, \ldots, 8\}$, an d measured the Euclidean distance between the predicted and actual 2D FOE coordinates. The result is shown in Fig. 7.7B. Each data point shows the model's heading prediction error on the test set for a particular number of basis vectors, averaged over ten trials (i.e., the ten folds of the cross-validation procedure). The vertical bars are the SD. The lowest prediction error was achieved with $B = 64$ (Kolmogorov-Smirnov test, $p \approx 10^{-12}$). In order to determine the sparseness of the population code, we investigated how many MSTd-like model units were activated by any given stimulus in the data set (population sparseness) as well as how many stimuli any given model unit

responded to (lifetime sparseness) (Fig. 7.7C). Sparseness metrics were computed according to the definition by Vinje and Gallant (2000), which can be understood as a measure of both the nonuniformity ("peakiness") and strength of the population response (see Section 7.2.9). A sparseness value of zero would be indicative of a dense code (where every stimulus would activate every neuron), whereas a sparseness value of one would be indicative of a local code (where every stimulus would activate only one neuron) (Spanne and Jorntell, 2015). As expected, the analysis revealed that both population and lifetime sparseness monotonously increased with an increasing number of basis vectors. Sparseness values ranging from roughly 0.41 for $B = 16$ to 0.65 for $B = 256$ suggested that the population code was sparse in all cases, as it did not get close to the extreme case of either a local or a dense code.

Overall these results suggest that our model is compatible with a series of findings demonstrating that macaque MSTd might encode a number of self-motion related variables using a distributed, versatile population code that is not restricted to a specific subpopulation of neurons within MSTd (Bremmer et al., 1998; Gu et al., 2010; Hamed et al., 2003; Xu et al., 2014). Interestingly, we found that the sparseness regime in which model MSTd achieved the lowest heading prediction error (Fig. 7.7C) was also the regime in which MSTd-like model units reproduced a variety of known MSTd visual response properties (Figs. 7.4–7.6, 7.9–7.12).

7.3.3 Heading Perception During Eye Movements

We also studied how the accuracy of the population code for heading in model MSTd is affected by eye movements. Eye movements distort the pattern of full-field motion on the retina, causing the FOE of an expanding flow field to no longer indicate heading (see Section 7.2.1). Under these circumstances, the brain must find a way to discount the motion components caused by eye rotations. The classical viewpoint is that this is achieved with

the help of extraretinal signals (Wallach, 1987) (but see Kim et al. (2015)). Indeed, macaque MSTd might be well-equipped to account for eye movement-related signals (Bradley et al., 1996; Komatsu and Wurtz, 1988; Morris et al., 2012; Newsome et al., 1988; Page and Duffy, 1999).

In order to investigate the effect of eye movements on heading perception, we asked if model MSTd could predict heading for a number of simulated scene layouts using the linear decoding weights described in the previous section, and compared the results to the psychophysical study of Royden et al. (1994). In these experiments, observers viewed displays of randomly-placed dots whose motions simulated translation in the absence and presence of rotations toward a back plane (Fig. 7.8A), a ground plane (Fig. 7.8D), and a 3D dot cloud (Fig. 7.8G). Observers were instructed to fixate a cross that either remained stationary in the center of the display (simulated eye movement condition) or moved horizontally across the display on the horizontal midline (real eye movement condition) with speeds of 0, $\pm 1\,^{\circ}\,\mathrm{s}^{-1}$, $\pm 2.5\,^{\circ}\,\mathrm{s}^{-1}$, and $\pm 5\,^{\circ}\,\mathrm{s}^{-1}$. At the end of the motion sequence, seven equally-spaced lines appeared 4° apart, and the observers indicated the line closest to their perceived headings (seven-alternative, forced-choice paradigm (7AFC)). Behavioral results are shown in Fig. 7.8B, E, and H for the simulated eye movement condition (closed symbols) and real eye movement condition (open symbols), averaged over 20 trials.

We applied the same experimental protocol to model MSTd. Visual stimuli were generated according to the motion parameters provided by Royden et al. (1994), and subtended 90° × 90° of visual angle. Heading predictions were generated by applying the linear decoding weights described in the previous section to MSTd population activity. These predictions were then rounded to the nearest available heading in the 7AFC task (4° apart, therefore spanning ±12° around straight ahead). Fig. 7.8C, F, and I summarize heading predictions of model MSTd averaged over 20 trials for reference headings of −4° (blue circles), 0 (green squares), and 4° (red triangles) relative to straight ahead. Consistent with the simulated eye

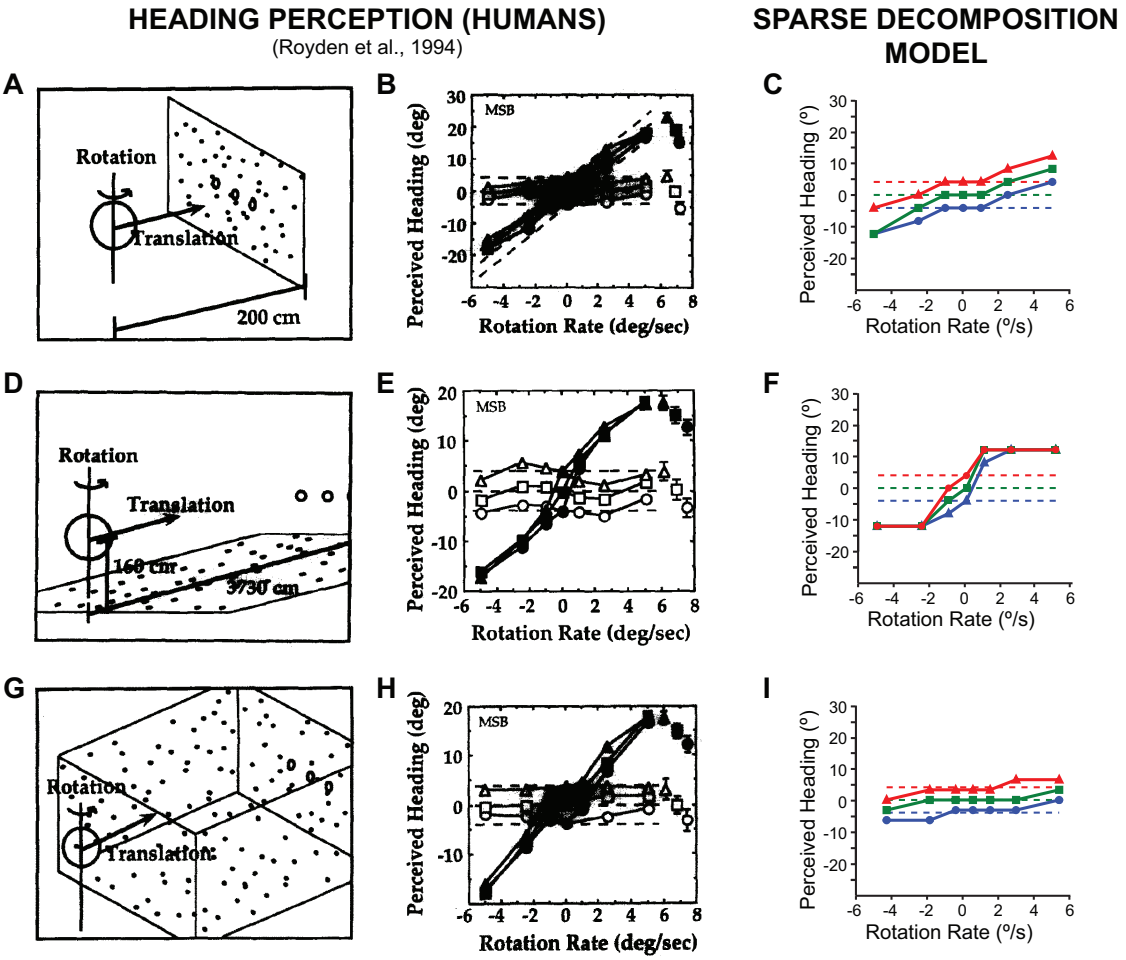

Figure 7.8: Heading perception during observer translation in the presence of eye movements. Shown are three different scene geometries (back plane, **A**–**C**; ground plane, **D**–**F**; dot cloud, **G**–**I**), reprinted from Royden et al. (1994) (their Figs. 6, 8-9, 12-13). Observer translation was parallel to the ground and was in one of three directions (open circles), coinciding with the fixation point. Real and simulated eye movements were presented with rates of 0, $\pm 1\,^{\circ}\,s^{-1}$, $\pm 2.5\,^{\circ}\,s^{-1}$, or $\pm 5\,^{\circ}\,s^{-1}$. **B**, **E**, **H**, Perceived heading reported by human subjects for real and simulated eye movements (open and closed symbols, respectively). **C**, **F**, **I**, Behavioral performance of model MSTd for simulated eye movements. Horizontal dotted lines indicate the actual headings of -4° (blue triangles), 0 (green squares), and 4° (red circles) relative to straight ahead.

movement condition, model MSTd made systematic prediction errors, which were largest for observer movement over a ground plane (Fig. 7.8F).

7.3.4 Population Code Underlying Heading Discrimination

Another interesting behavioral effect that might be due to MSTd population coding is the fact that heading discrimination is most precise when subjects have to discriminate small variations around straight-ahead headings (Crowell and Banks, 1993; Gu et al., 2010). It was long believed that this behavioral effect could be explained by an abundance of MSTd neurons preferring straight-ahead headings with sharp tuning curves (e.g., Duffy and Wurtz (1995)). However, Gu et al. (2010) were able to demonstrate that this behavior might instead be due to an abundance of MSTd neurons preferring lateral headings with broad, cosine-like tuning curves (Gu et al., 2006; Lappe et al., 1996), causing their peak discriminability (steepest tuning-curve slopes) to lie near straight-ahead headings.

We tested whether simulated population activity in model MSTd could account for these behavioral effects by computing peak discriminability and Fisher information from heading tuning curves, following the experimental protocol of Gu et al. (2010). Peak discriminability occurs for motion directions where the slope of a neuronal tuning curve is steepest (Jazayeri and Movshon, 2006; Purushothaman and Bradley, 2005; Seung and Sompolinsky, 1993), and Fisher information puts an upper bound on the precision with which population activity can be decoded with an unbiased estimator (Pouget et al., 1998; Seung and Sompolinsky, 1993).

The heading tuning curve of every model unit was measured by presenting 24 directions of self-translation in the horizontal plane (0, $\pm 15^\circ$,$\pm 30^\circ$, ..., $\pm 165^\circ$ and 180° relative to straight ahead, while elevation was fixed at 0) through a 3D cloud of dots that occupied a space that was 0.75 m deep, subtended $90^\circ \times 90^\circ$ of visual angle, and drifted at $0.3\,\mathrm{m\,s^{-1}}$ (see Section 7.2.5). Here we adapted the coordinate system to coincide with the one used by Gu

et al. (2010), so that azimuth angles were expressed relative to straight ahead. The slope of the tuning curve was computed by interpolating the coarsely sampled data using a spline function (resolution: 0.01°) and then taking the spatial derivative of the fitted curve at each possible reference heading. Peak discriminability was achieved where the slope of the tuning curve was steepest.

Fig. 7.9 shows the distribution of reference headings at which neurons in MSTd exhibit their minimum neuronal threshold (Fig. 7.9A) (Gu et al., 2010), compared to peak discriminability computed from simulated activity of MSTd-like model units (Fig. 7.9B). Both neurophysiologically measured and simulated distributions had clear peaks around forward (0) and backward (180°) headings. To further illustrate the relationship between peak discriminability and peak firing rate, the tuning width at half maximum is plotted versus heading preference (location of peak firing rate) for neurons in macaque MSTd (Fig. 7.9C) and MSTd-like model units (Fig. 7.9D). Consistent with neurons in macaque MSTd, the distribution of preferred headings in model MSTd was significantly nonuniform ($p < 0.05$, see Section 7.2.7) with peaks at ±90° azimuth relative to straight ahead (i.e., lateral headings), and most model units had broad tuning widths ranging between 90° and 180°. Surprisingly, our model was also able to recover what appears to be a distinct subpopulation of narrowly-tuned units that prefer forward headings (open circles in Fig. 7.9C, D).

Fig. 7.10 shows population Fisher information derived from interpolated tuning curves for 882 neurons in macaque MSTd (Fig. 7.10A, red) and 896 MSTd-like model units (Fig. 7.10B). According to Eq. 7.1, the contribution of each neuron (model unit) to Fisher information is the square of its tuning curve slope at a particular reference heading divided by the corresponding mean firing rate (unit activation). Note that even though MSTd-like model units are deterministic, they exhibit response variability when repeatedly presented with 3D clouds made from dots with random depth (150 repetitions, see Section 7.2.10). Consistent with data from macaque MSTd, there is a clear dependence of Fisher information on reference

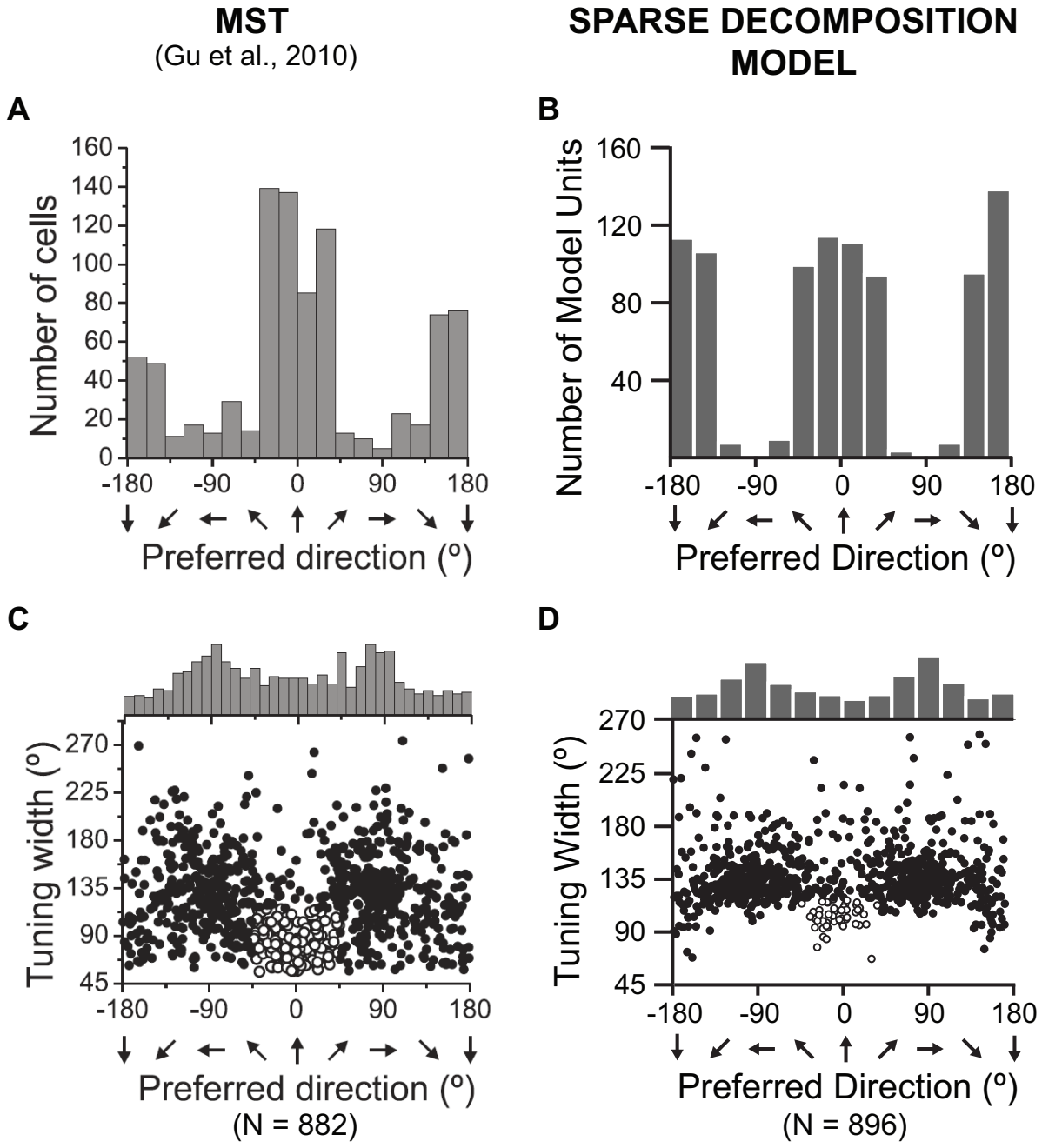

Figure 7.9: Heading discriminability based on population activity in macaque MSTd (**A**, **C**), reprinted from Gu et al. (2010) (their Fig. 4), and in model MSTd (**B**, **D**). **A**, **B**, Distribution of the direction of maximal discriminability. **C**, **D**, Scatter plot of each neuron's or model unit's tuning width at half maximum versus preferred direction. The top histogram shows the marginal distribution of heading preferences. Also highlighted is a subpopulation of neurons or model units with direction preferences within 45° of straight ahead and tuning width $< 115°$ (open symbols).

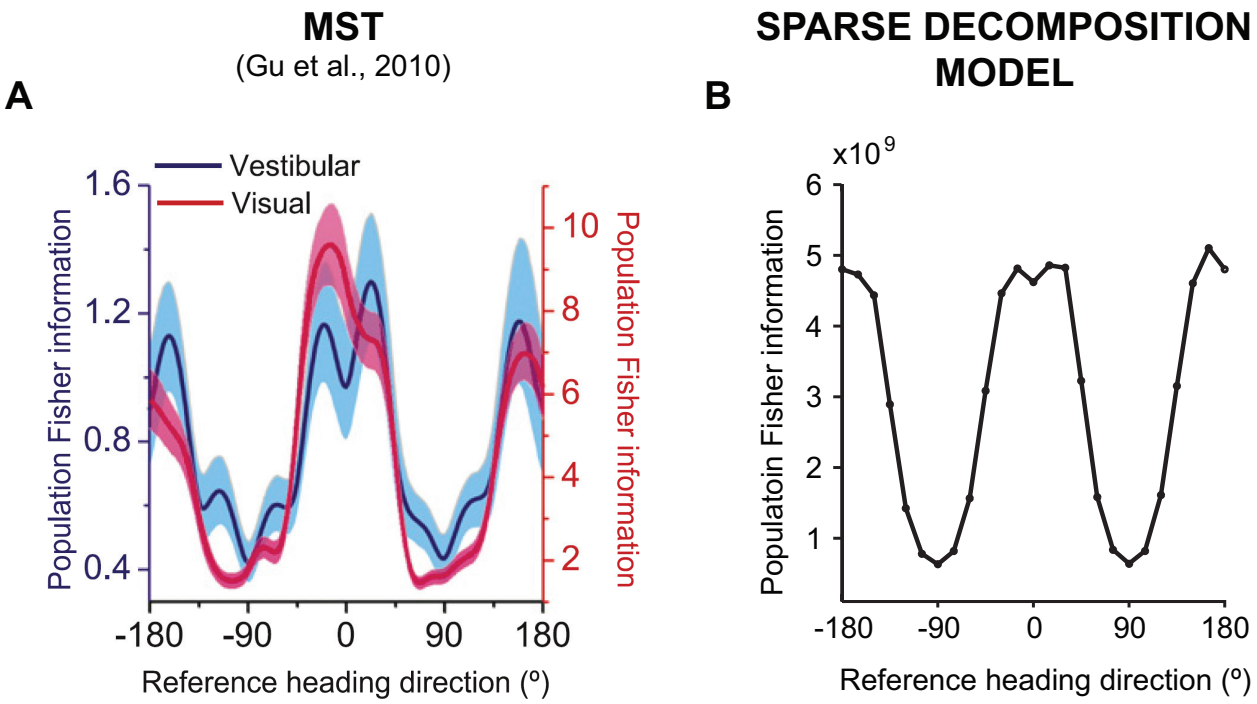

Figure 7.10: Population Fisher information of macaque MSTd (**A**), reprinted from Gu et al. (2010) Fig. 5), and of model MSTd (**B**). Error bands in **A** represent 95 % confidence intervals derived from a bootstrap procedure.

heading for our population of MSTd-like model units, with a maximum occurring for headings near 0 (i.e., straight ahead) and a minimum occurring for headings near $\pm 90^\circ$ (i.e., lateral headings).

Overall these results are in good agreement with population statistics reported by Gu et al. (2010), and provide further computational support that behavioral dependence on reference heading can be largely explained by the precision of an MSTd-like population code.

7.3.5 Gaussian Tuning in Spiral Space

The finding that lateral headings are overrepresented in MSTd (Gu et al., 2006, 2010; Takahashi et al., 2007) then raise the question as to how these recent data might be reconciled with earlier studies reporting an abundance of MSTd cells preferring forward motions (i.e.,

expanding flow stimuli) (e.g., Duffy and Wurtz (1995)). A possible explanation was offered by (Gu et al., 2010)1, who hypothesized that observed differences in population statistics might be (at least partially) due to a selection bias: In order to locate visually responsive cells in MSTd, the authors of earlier studies tended to screen for cells that discharge in response to expansion stimuli (Duffy and Wurtz, 1991a, 1995; Graziano et al., 1994). In contrast, later studies recorded from any MSTd neuron that was spontaneously active or responded to a large-field flickering random-dot stimulus (Gu et al., 2006, 2010; Takahashi et al., 2007).

To test their hypothesis, we applied the same selection bias to our sample of MSTd-like model units, and restricted further analysis to the selected subpopulation. The selection process was mimicked simply by making it more likely for a model unit to be selected the stronger it responded to an expansion stimulus. Out of a total of 896 MSTd-like model units, 188 were selected for further processing.

We then proceeded with the experimental protocol described by Graziano et al. (1994) establish a visual tuning profile for the remaining 188 model units. Model units were presented with eight optic flow stimuli that spanned what they termed "spiral space": expansion, contraction, clockwise rotation (CW), counterclockwise rotation (CCW), and four intermediate spiral patterns. These stimuli differed from previously described optic flow stimuli in that they did not accurately depict observer motion in 3D. Instead, stimulus diameter was limited to 30°, and angular velocity simply increased with distance to the center of the stimulus, with a maximum speed of $17.2\,^{\circ}\,\mathrm{s}^{-1}$. As in the study of Graziano et al. (1994), we fit the resulting tuning curves with a Gaussian function to find the peak (the mean of the Gaussian) that corresponded to the preferred direction in spiral space, and to provide a measure of bandwidth (σ, the SD of the Gaussian) and goodness-of-fit (r, the correlation coefficient).

The results are shown in Fig. 7.11. When looking at the preferred motion direction of MSTd cells (Fig. 7.11A), Graziano et al. (1994) observed Gaussian tuning across the full

range of rotational flow fields. Here, 0 corresponded to clockwise rotation, 90° to expansion, 180° to counterclockwise rotation, 270° to contraction, and the oblique directions (45°, 135°, 225°, and 315°) corresponded to four intermediate spiral stimuli. Each arrow indicated the preferred direction or peak response (the mean of the Gaussian) of each neuron ($N = 57$) in spiral space. Consistent with these data from macaque MSTd, the majority of preferred spiral directions in our subpopulation of MSTd-like model units ($N = 188$) clustered near expansion, with only few units preferring contraction, and roughly 51 % of units were spiral-tuned (vs. 35 % in their sample) (Fig. 7.11B, E). Similar to 86 % of isolated MSTd cells having smooth tuning curves and good Gaussian fits (i.e., a correlation coefficient of $r \geq 0.9$, with an average r of 0.97), 112 of 188 MSTd-like model units (60 %) had $r \geq 0.9$, with an average r of 0.924. Compared to real MSTd neurons, the model units had comparable, although slightly broader Gaussian widths (an average σ of 111° and SE of 24° in our case versus an average σ of 61° and SE of 5.9° in their case).

If the predominance of expansion cells in Fig. 7.11D is truly due to a pre-selection procedure, then any bias in the data should be removed when the above experimental procedure is applied to all cells in MSTd. Indeed, extending the analysis to the entire population of MSTd-like model units revealed a more uniform sampling of direction selectivity across spiral space (Fig. 7.11C), which flattened the distribution of preferred motion directions (Fig. 7.11F). 677 of 896 MSTd-like model units (76 %) had $r \geq 0.9$, with an average r of 0.929, an average σ of 108°, and SE of 9.1°.

7.3.6 Continuum of 2D Response Selectivity

Interestingly, the same narrative can be applied to other early studies of MSTd in which cells were supposedly selected depending on how well they responded to radial motion. For example, using a "canonical" set of twelve flow stimuli (eight directions of planar motion; ex-

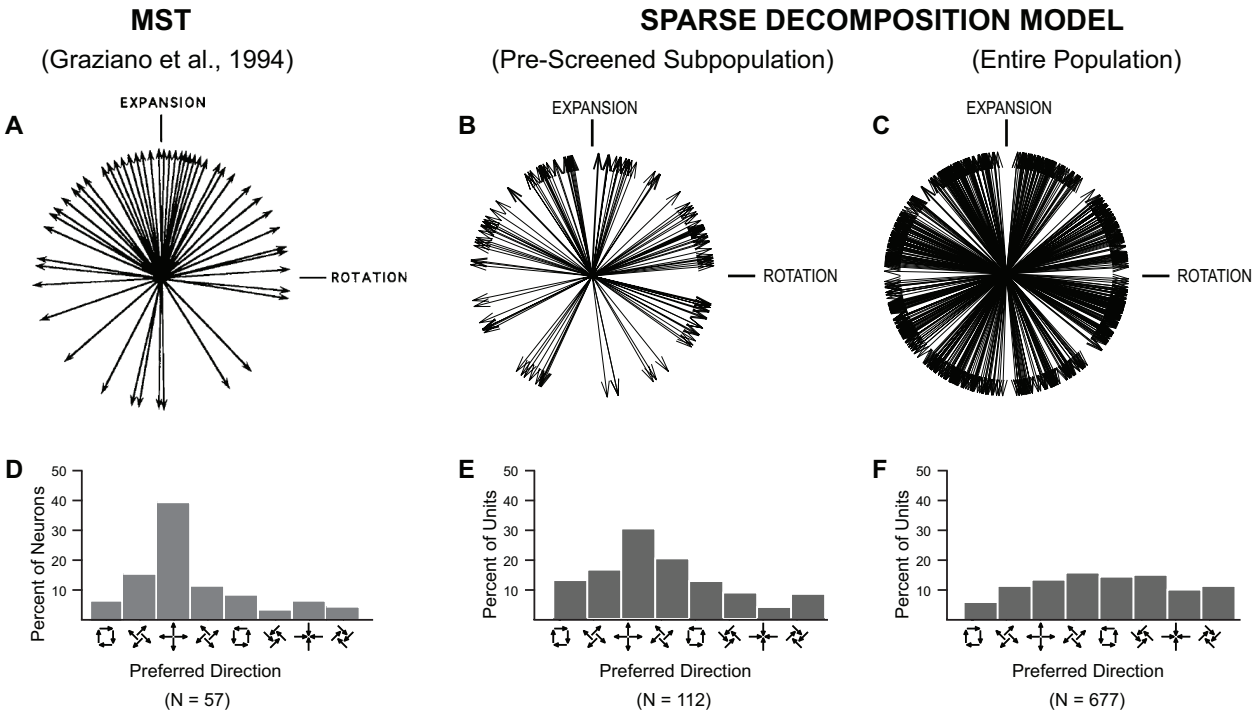

Figure 7.11: Gaussian tuning in spiral space. **A**, Gaussian tuning of a sample of 57 neurons across the full range of rotational flow fields, reprinted from Graziano et al. (1994) (their Fig. 9). Each arrow indicates the peak response (the mean of the Gaussian fit) of each neuron in spiral space. **B**, The distribution of preferred spiral directions of a sample of 112 MSTd-like model units whose tuning curves were well-fit with a Gaussian. Model units were more likely to be included in the sample the stronger they responded to an expansion stimulus. **C**, The distribution of preferred spiral directions applied to the entire population of 896 MSTd-like model units, of which 677 had smooth Gaussian fits. **D**, Bar plot of the data in **A**, for better comparison, adapted from Clifford et al. (1999). **E**, Bar plot of the data in **B**. **F**, Bar plot of the data in **C**.

pansion and contraction; clockwise and counterclockwise rotation), Duffy and Wurtz (1995) showed that most neurons in MSTd were sensitive to multiple flow components, with only few neurons responding exclusively to either planar, circular, or radial motion. In a sample of 268 MSTd cells, Duffy and Wurtz (1995) found that 18 % of cells primarily responded to one component of motion (planar, circular, or radial), that 29 % responded to two components (planocircular or planoradial, but rarely circuloradial), and that 39 % responded to all three components (Fig. 7.12A).

We simulated their experiments by presenting the same twelve stimuli to the subpopulation

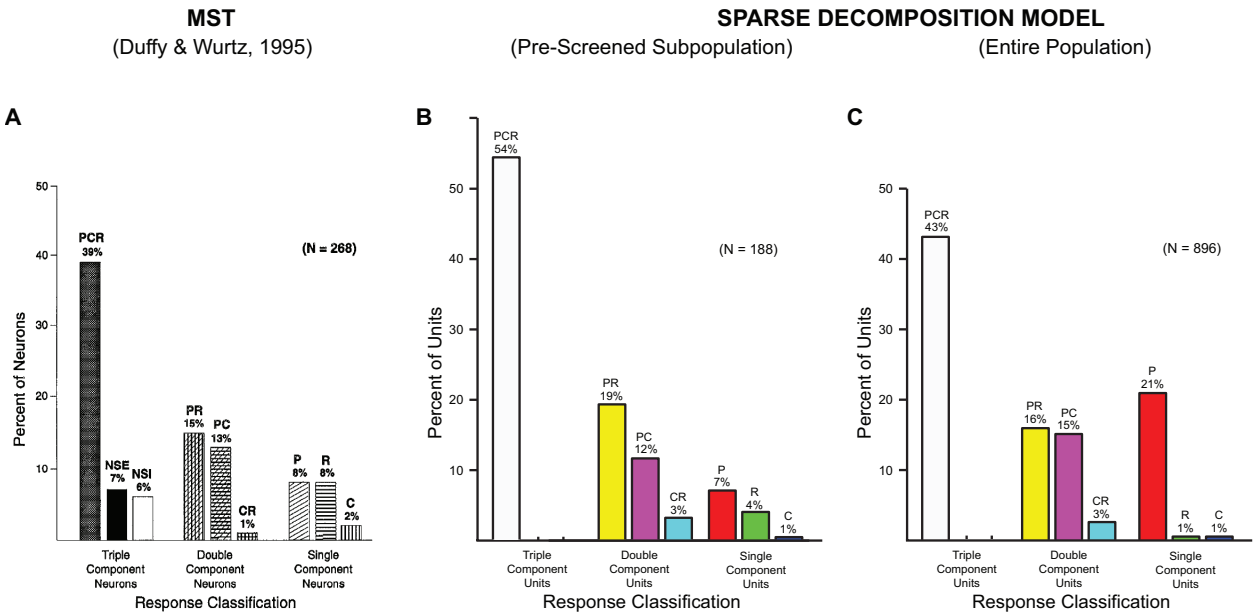

Figure 7.12: Continuum of response selectivity. **A**, Response classification of a sample of 268 MSTd neurons, reprinted from Duffy and Wurtz (1995) (their Fig. 3). **B**, Response classification of a sample of 188 MSTd-like model units. Model units were more likely to be included in the sample the stronger they responded to an expansion stimulus. **C**, Response classification of the entire population of 896 MSTd-like model units. Triple-component cells were: planocirculoradial (PCR), nonselective excitatory (NSE), and nonselective inhibitory (NSI). Double-component cells were: planoradial (PR), planocircular (PC), and circuloradial (CR). Single-component cells were: planar (P), radial (R), and circular (C). 81 % of neurons in **A**, 88 % of model units in **B**, and 77 % of model units in **C** responded to more than one type of motion.

of 188 MSTd-like model units described above, and classified their responses. Duffy and Wurtz (1995) classified neurons according to the statistical significance of their responses, which was difficult to simulate since MSTd-like model units did not have a defined noise level or baseline output. Instead we mimicked their selection criterion by following a procedure from Perrone and Stone (1998), where the response of a model unit was deemed "significant" if it exceeded 12 % of the largest response that was observed for any model unit in response to any of the tested stimuli.

Without further adjustments, our model recovered a distribution of response selectivities very similar to those reported by Duffy and Wurtz (1995) (Fig. 7.12B): The largest group was formed by triple-component or planocirculoradial (PCR) MSTd-like model units with

selective responses to planar, circular, and radial stimuli (white). Double-component units were mainly planoradial (PR) with responses to a planar or radial stimulus (magenta), with few planocircular (PC) units (yellow) and only a handful of circuloradial (CR) units (cyan). Single-component cells were mainly planar (P) units (red), with only few radial (R) units and circular (C) units (blue). We did not find any nonselective excitatory (NSE) or nonselective inhibitory (NSI) units (because our model did not include inhibitory units).

Only 1 % of the cells observed by Duffy and Wurtz (1995) were circuloradial (CR) cells, which they suggested were equivalent to the spiral-selective neurons reported by Graziano et al. (1994). Thus they concluded that spiral tuning was rare in MSTd. Interestingly, our model recovers distributions that are comparable to both empirical studies; that is, an abundance of spiral-tuned cells in Fig. 7.11, and an abundance of PCR cells in Fig. 7.12. Our results thus offer an alternative explanation to these seemingly contradictive findings, by considering that most spiral-tuned cells, as identified by Graziano et al. (1994), might significantly respond to planar stimuli when analyzed under the experimental protocol of Duffy and Wurtz (1995), effectively making most spiral-tuned cells part of the PCR class of cells, as opposed to the CR class of cells.

What might these data look like in the absence of any pre-selection procedure? To answer this question, we extended the response classification to the entire population of 896 MSTd-like model units. As is evident in Fig. 7.12C, this analysis revealed a large fraction of P units that had not previously been included. In addition, the relative frequency of units responding to radial motion (i.e., R, PR, and PCR units) decreased noticeably.

Overall, these results suggest that our model is in agreement with a wide variety of MSTd data collected under different experimental protocols, and offers an explanation of how these data might be brought into agreement when differences in the sampling procedure are accounted for.

7.4 Discussion

We found that applying NMF (Lee and Seung, 1999, 2001; Paatero and Tapper, 1994) to MT-like patterns of activity can account for several essential response properties of MSTd neurons, such as 3D translation and rotation selectivity (Gu et al., 2006; Takahashi et al., 2007), tuning to radial, circular, and spiral motion patterns (Duffy and Wurtz, 1995; Graziano et al., 1994; Lagae et al., 1994), as well as heading selectivity (Hamed et al., 2003; Page and Duffy, 1999, 2003). This finding suggests that these properties might emerge from MSTd neurons performing a biological equivalent of dimensionality reduction on their inputs. Furthermore, the model accurately captures prevalent statistical properties of visual responses in macaque MSTd, such as an overrepresentation of lateral headings (Gu et al., 2006, 2010; Lappe et al., 1996; Takahashi et al., 2007) that can predict behavioral thresholds of heading discrimination (Gu et al., 2010) and heading perception during eye movements (Royden et al., 1994). At the population level, model MSTd efficiently and accurately predicts a number of perceptual variables (such as heading and eye rotation velocity) using a sparse distributed code (Hamed et al., 2003; Olshausen and Field, 1996, 1997), consistent with ideas from the efficient-coding and free-energy principles (Attneave, 1954; Barlow, 1961; Friston, 2010; Friston et al., 2006; Linsker, 1990; Simoncelli and Olshausen, 2001).

7.4.1 Sparse Decomposition Model of MSTd

It is well-known that MSTd neurons do not decompose optic flow into its first-order differential invariants (i.e., into components of divergence, curl, and deformation (Koenderink and van Doorn, 1975)). Instead, neurons in MSTd act as "templates" (Perrone, 1992) that perform a dot product-type operation to signal how well the apparent motion on the retina matches their preferred flow component or mixture of components (Orban et al., 1992; Saito et al., 1986; Tanaka et al., 1989). This "template matching" was later demonstrated to be

a powerful approach to self-motion analysis possibly at work at least in mammals (Lappe and Rauschecker, 1994; Perrone and Stone, 1994, 1998) and insects (Krapp, 2000; Krapp and Hengstenberg, 1996; Srinivasan et al., 1996); all without the need to perform a mathematical decomposition of optic flow.

Alternatively, we provide computational evidence that MSTd neurons might decompose optic flow in the sense of matrix factorization, in order to reduce the number of "templates" used to represent the spectrum of retinal flow fields encountered during self-movement in an efficient and parsimonious fashion. Such a representation would be in agreement with the efficient-coding or infomax principle (Attneave, 1954; Barlow, 1961; Linsker, 1990; Simoncelli and Olshausen, 2001), which posits that sensory systems should employ knowledge about statistical regularities of their input in order to maximize information transfer. If information maximization is interpreted as optimizing for both accuracy (prediction error) and efficiency (model complexity), then the efficient-coding principle can be understood as a special case of the free-energy principle (Friston, 2010; Friston et al., 2006).

A sparse, parts-based decomposition model of MSTd implements these principles in that it can co-optimize accuracy and efficiency by representing high-dimensional data with a relatively small set of highly informative variables. As such the model is intimately related to the framework of nonnegative sparse coding (Eggert and Korner, 2004; Hoyer, 2002). Efficiency is achieved through sparseness, which is a direct result of NMF's nonnegativity constraints (Lee and Seung, 1999). Accuracy trades off with efficiency, and can be achieved by both minimizing reconstruction error and controlling for model complexity (i.e., by tuning the number of basis vectors) (see Fig. 7.9). Statistical knowledge enters the model via relative frequencies of observations in the input data, which are controlled by both natural viewing conditions and natural scene statistics. NMF explicitly discourages statistically inefficient representations, because strongly accounting for a rare observation at the expense of ignoring a more common stimulus component would result in an increased reconstruction error.

Interestingly, we found that the sparseness regime in which model MSTd achieved the lowest heading prediction error and thus showed the greatest potential for generalization (Fig. 7.7B, C) was also the regime in which MSTd-like model units reproduced a variety of known MSTd visual response properties (Figs. 7.4–7.6, 7.9–7.12). In contrast to findings about early sensory areas (Olshausen and Field, 1996, 1997; Vinje and Gallant, 2000), this regime does not utilize an overcomplete dictionary, yet can still be considered a sparse coding regime (Spanne and Jorntell, 2015). Sparse codes are a trade-off between dense codes (where every neuron is involved in every context, leading to great memory capacity but suffering from cross-talk among neurons) and local codes (where there is no interference but also no capacity for generalization) (Spanne and Jorntell, 2015). We speculate that sparse codes with a relatively small dictionary akin to the one described in this paper might be better suited (as opposed to overcomplete basis sets) for areas such as MSTd, because the increased memory capacity of such a code might lead to compact and multi-faceted encodings of various perceptual variables (Bremmer et al., 1998; Brostek et al., 2014; Hamed et al., 2003).

7.4.2 Model Limitations

Despite its simplicity the present model is able to explain a variety of MSTd visual response properties. However, a number of issues remain to be addressed in the future, such as the fact that neurons in MSTd are also driven by vestibular (Gu et al., 2006; Page and Duffy, 2003; Takahashi et al., 2007) and eye movement-related signals (Bradley et al., 1996; Komatsu and Wurtz, 1988; Morris et al., 2012; Newsome et al., 1988; Page and Duffy, 1999). In addition, some neurons are selective not just for heading, but also for path and place (Froehler and Duffy, 2002; Page et al., 2015). Also, we have not yet attempted to model the complex, nonlinear interactions found among different subregions of the receptive field (Duffy and Wurtz, 1991b; Lagae et al., 1994; Mineault et al., 2012), but speculate that they might be similar to the ones reported in MT (Majaj et al., 2007; Richert et al., 2013).

At a population level, MSTd exhibits a range of tuning behaviors from pure retinal to head-centered stimulus velocity coding (Brostek et al., 2014; Hamed et al., 2003; Yu et al., 2010) that include intermediate reference frames (Fetsch et al., 2007). From a theoretical standpoint, the sparse decomposition model seems a good candidate to find an efficient, reference frame-agnostic representation of various perceptual variables (Hamed et al., 2003; Pouget and Sejnowski, 1997; Pouget and Snyder, 2000), but future iterations of the model will have to address these issues step-by-step.

7.4.3 Model Alternatives

Several computational models have tried to account for heading-selective cells in area MSTd. In the heading-template model (Perrone and Stone, 1994, 1998), MSTd forms a "heading map", with each MSTd-like unit receiving input from a mosaic of MT-like motion sensors that correspond to the flow that would arise from a particular heading. Heading is then inferred by the preferred FOE of the most active heading template. A complication of this type of model is that it requires an extremely large number of templates to cover the combinatorial explosion of heading parameters, eye rotations, and scene layouts (Perrone and Stone, 1994), even when the input stimulus space is restricted to gaze-stabilized flow fields (Perrone and Stone, 1998). The velocity gain field model (Beintema and van den Berg, 1998, 2000) tries to reduce the number of combinations by using templates that uniquely pick up the rotational component of flow, but recent evidence argues against this type of model (Berezovskii and Born, 2000; Duijnhouwer et al., 2013). In a related approach to the one presented in this paper, Zemel and Sejnowski (1998) proposed that neurons in MST encode hidden causes of optic flow, by using a sparse distributed representation that facilitates image segmentation and object- or self-motion estimation by downstream read-out neurons. Unfortunately, they did not quantitatively assess many of the response properties described in this paper. In addition, their model showed an overrepresentation of expanding flow stimuli, which is hard

to reconcile with recent findings (Gu et al., 2006).

In contrast, our model offers a biologically plausible account of a wide range of visual response properties in MSTd, ranging from single-unit activity to population statistics. Adoption of the sparse decomposition model supports the view that single-unit preferences emerge from the pressure to find an efficient representation of large-field motion stimuli (as opposed to encoding a single variable such as heading), and that these units act in concert to represent perceptual variables both accurately (Gu et al., 2010) and efficiently (Hamed et al., 2003).

Chapter 8

Summary and Conclusion

Simulating large-scale models of biological motion perception is challenging, due to the required memory to store the network structure and the computational power needed to quickly solve the neuronal dynamics. In Chapters 3 and 4 I introduced software that enabled the practical feasibility of most of the research presented in this thesis. CARLsim allowed us to simulate meso-scale neural networks (on the order of 10^5 neurons and 10^7 synapses) that require a high degree of biological detail without sacrificing performance. Analogously, development of an efficient and scalable implementation of the Motion Energy model (Simoncelli and Heeger, 1998) was necessary in order to handle video streams ($\sim 360 \times 480$ pixels) of up to 30 frames per second in (quasi) real time. These software efforts allowed us to embody the computational models presented in this thesis on a neurorobotics platform so that we could investigate the link between simulated neural circuitry and real-world robot behavior. All software was released under open-source licenses, making it freely and openly available for other researchers to use the software, adapt it for their purposes, and expand for future studies.

In Chapter 5, I presented a large-scale spiking model of visual area MT that 1) was capable

of exhibiting both component and pattern motion selectivity, 2) generated speed tuning curves that were in agreement with electrophysiological data, and 3) emulated human choice accuracy and the effect of motion strength on reaction time in a motion discrimination task. Although the model was able to demonstrate how the aperture problem could be solved with solely biologically plausible mechanisms and network dynamics, future studies could investigate how the model might generalize to more perceptually challenging stimuli that include transparent motion, second-order motion, or binocular cues.

In Chapter 6, I described the deployment of the model outlined in Chapter 5 on a robotics platform, used to steer the robot around obstacles toward a visually salient target. The study demonstrated how a model of MT might build a cortical representation of optic flow in the spiking domain, and showed that these simulated motion signals were sufficient to steer the robot on human-like smooth paths around obstacles in the real world. In the future, it would be interesting to see the model being extended to cortical areas beyond MT, in order to investigate the functional real-world implications of visual motion processing in higher-order areas of the motion pathway.

Finally, Chapter 7 introduced a novel computational model of optic flow processing in MSTd based on ideas from the efficient-coding and free-energy principles. I demonstrated that a variety of visual response properties of MSTd neurons could be derived from MT-like input features using NMF, suggesting that response properties such as 3D translation and rotation selectivity, complex motion perception, and heading selectivity might simply be a by-product of MSTd performing dimensionality reduction on their inputs. Despite its simplicity, the model was able to explain a variety of MSTd visual response properties, ranging from single-unit activity to population statistics. These findings provide a further step towards a scientific understanding of the often nonintuitive response properties of MSTd neurons. In the future, this work might lead to new intellectual property (patent pending) and find embodiment in brain-inspired robots and drones. The model itself might be extended to

account for nonvisual response properties of MSTd neurons, which include the involvement of vestibular and eye movement-related signals.

Overall, the contributions of this thesis span multiple disciplines including computational neuroscience, computer science/engineering, and robotics. The present work might be of interest to the neuroscience community, as it 1) provides a mechanistic description of visual motion processing in the brain and 2) makes specific predictions that can be empirically verified. The present work might also be of interest to the computer science and engineering communities, as it 1) sheds light on the computational principles that might be at play in the visual brain, 2) makes significant contributions to the open-source movement, and 3) describes algorithms that are compatible with recent neuromorphic hardware. Finally, the present work might be of interest to the robotics community, as it 1) investigates the functional link between models of brain function and an agent's real-world behavior, and 2) constitutes a first step into building fully functional and autonomous neurorobotics platforms.

It is my hope that these studies will not only further our understanding of how the brain works, but also lead to novel algorithms and brain-inspired robots capable of outperforming current artificial systems.

Bibliography

L. F. Abbott and S. B. Nelson. Synaptic plasticity: taming the beast. *Nature Neuroscience*, 3:1178–1183, 2000.

E. H. Adelson and J. R. Bergen. Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am. A*, 2(2):284–299, Feb 1985. doi: 10.1364/JOSAA.2.000284.

E. H. Adelson and J. A. Movshon. Phenomenal coherence of moving visual patterns. *Nature*, 300(5892):523–525, 1982.

A. P. Alivisatos, M. Chun, G. M. Church, R. J. Greenspan, M. L. Roukes, and R. Yuste. The brain activity map project and the challenge of functional connectomes. *Neuron*, 74: 970–974, 2012.

J. Allman, F. Miezin, and E. McGuinness. Direction- and velocity-specific responses from beyond the classical receptive field in the middle temporal visual area (mt). *Perception*, 14(2):105–126, 1985. doi: 10.1068/p140105.

R. A. Andersen, C. Asanuma, G. Essick, and R. M. Siegel. Corticocortical connections of anatomically and physiologically defined subdivisions within the inferior parietal lobule. *Journal of Computational Neurology*, 296:65–113, 1990.

RA Andersen, GK Essick, and RM Siegel. Encoding of spatial location by posterior parietal neurons. *Science*, 230(4724):456–458, 1985. ISSN 0036-8075. doi: 10.1126/science.4048942.

F. Attneave. Some informational aspects of visual perception. *Psychology Review*, 61:183–193, 1954.

M. Avery and J. L. Krichmar. Improper activation of d1 and d2 receptors leads to excess noise in prefrontal cortex. *Frontiers in Computational Neuroscience*, 9(31), 2015. ISSN 1662-5188. doi: 10.3389/fncom.2015.00031.

M. C. Avery, D. A. Nitz, A. A. Chiba, and J. L. Krichmar. Simulation of cholinergic and noradrenergic modulation of behavior in uncertain environments. *Frontiers in Computational Neuroscience*, 6(5), 2012. ISSN 1662-5188. doi: 10.3389/fncom.2012.00005.

A. A. Baloch and S. Grossberg. A neural model of high-level motion processing: Line motion and formotion dynamics. *Vision Research*, 37(21):3037 – 3059, 1997. ISSN 0042-6989. doi: 10.1016/S0042-6989(97)00103-X.

H. B. Barlow. Possible principles underlying the transformation of sensory messages. in: Sensory comunication. In W. A. Rosenblinth, editor, *Sensory communication*. MIT Press, 1961.

P. Bayerl and H. Neumann. Disambiguating visual motion through contextual feedback modulation. *Neural Computation*, 16, 2004. doi: 10.1162/0899766041732404.

J. A. Beintema and A. V. van den Berg. Heading detection using motion templates and eye velocity gain fields. *Vision Research*, 38:2155–2179, 1998.

J. A. Beintema and A. V. van den Berg. Perceived heading during simulated torsional eye movements. *Vision Research*, 40:549–566, 2000.

T. Bekolay, J. Bergstra, E. Hunsberger, T. DeWolf, T. C. Stewart, D. Rasmussen, X. Choo, A. Voelker, and C. Eliasmith. Nengo: A python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics*, 7(48), 2014. ISSN 1662-5196. doi: 10.3389/fninf.2013.00048.

V. K. Berezovskii and R. T. Born. Specificity of projections from wide-field and local motion-processing regions within the middle temporal visual area of the owl monkey. *The Journal of Neuroscience*, 20(3):1157–1169, 2000.

M. Beyeler, N. D. Dutt, and J. L. Krichmar. Categorization and decision-making in a neurobiologically plausible spiking network using a STDP-like learning rule. *Neural Networks*, 48:109 – 124, 2013. doi: 10.1016/j.neunet.2013.07.012.

M. Beyeler, M. Richert, N. D. Dutt, and J. L. Krichmar. Efficient spiking neural network model of pattern motion selectivity in visual cortex. *Neuroinformatics*, 12(3):435–454, 2014. doi: 10.1007/s12021-014-9220-y.

M. Beyeler, K. D. Carlson, T.-S. Chou, N. D. Dutt, and J. L. Krichmar. CARLsim 3: A user-friendly and highly optimized library for the creation of neurobiologically detailed spiking neural networks. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8, July 2015a. doi: 10.1109/IJCNN.2015.7280424.

M. Beyeler, N. Oros, N. D. Dutt, and J. L. Krichmar. A GPU-accelerated cortical neural network model for visually guided robot navigation. *Neural Networks*, 72:75 – 87, 2015b. ISSN 0893-6080. doi: 10.1016/j.neunet.2015.09.005.

G. Bi and M. Poo. Synaptic modification by correlated activity: Hebb's postulate revisitied. *Annual Review of Neuroscience*, 24:1139–1150, 2001.

J. Billington, R. M. Wilkie, and J. P. Wann. Obstacle avoidance and smooth trajectory control: Neural areas highlighted during improved locomotor performance. *Frontiers in Behavioral Neuroscience*, 7(9), 2013. ISSN 1662-5153. doi: 10.3389/fnbeh.2013.00009.

G. G. Blasdel and J. S. Lund. Termination of afferent axons in macaque striate cortex. *Journal of Neuroscience*, 3:1389–1413, 1983.

K. Boahen. Neurogrid: Emulating a million neurons in the cortex. In *Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE*, volume Supplement, pages 6702–6702, Aug 2006. doi: 10.1109/IEMBS.2006.260925.

F. Bonin-Font, A. Ortiz, and G. Oliver. Visual navigation for mobile robots: A survey. *Journal of Intelligent and Robotic Systems*, 53(3):263–296, 2008. ISSN 1573-0409. doi: 10.1007/s10846-008-9235-4.

R. T. Born and D. C. Bradley. Structure and function of visual area MT. *Annual Review of Neuroscience*, 28(1):157–189, 2005. doi: 10.1146/annurev.neuro.26.041002.131052.

Richard T. Born. Center-surround interactions in the middle temporal visual area of the owl monkey. *Journal of Neurophysiology*, 84(5):2658–2669, 2000. ISSN 0022-3077.

D. Boussaoud, L. G. Ungerleider, and R. Desimone. Pathways for motion analysis: cortical connections of the medial superior temporal and fundus of the superior temporal visual area in the macaque. *Journal of Comparative Neurology*, 296:462–495, 1990.

J. M. Brader, W. Senn, and S. Fusi. Learning real-world stimuli in a neural network with spike-driven synaptic dynamics. *Neural Computation*, 19(11):2881–2912, November 2007. ISSN 0899-7667. doi: 10.1162/neco.2007.19.11.2881.

D. C. Bradley, M. Maxwell, R. A. Andersen, M. S. Banks, and K. V. Shenoy. Mechanisms of heading perception in primate visual cortex. *Science*, 273(5281):1544–1547, 1996. ISSN 0036-8075. doi: 10.1126/science.273.5281.1544.

David C. Bradley and Richard A. Andersen. Centersurround antagonism based on disparity in primate area MT. *The Journal of Neuroscience*, 18(18):7552–7565, 1998.

David C. Bradley and Manu S. Goyal. Velocity computation in the primate visual system. *Nature Reviews Neuroscience*, 9(9):686–695, aug 2008. ISSN 1471-003X. doi: 10.1038/nrn2472.

F. Bremmer, A. Pouget, and K. P. Hoffmann. Eye position encoding in the macaque posterior parietal cortex. *European Journal of Neuroscience*, 10:153–160, 1998.

R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J. M. Bower, M. Diesmann, A. Morrison, P. H. Goodman, F. C. Harris, M. Zirpe, T. Natschläger, D. Pecevski, B. Ermentrout, M. Djurfeldt, A. Lansner, O. Rochel, T. Vieville, E. Muller, A. P. Davison, S. El Boustani, and A. Destexhe. Simulation of networks of spiking neurons: A review of tools and strategies. *Journal of Computational Neuroscience*, 23(3):349–398, 2007. ISSN 1573-6873. doi: 10.1007/s10827-007-0038-6.

K. H. Britten. Mechanisms of self-motion perception. *Annu. Rev. Neurosci.*, 31:389–410, 2008.

K. H. Britten and H. W. Heuer. Spatial summation in the receptive fields of MT neurons. *Journal of Neuroscience*, 19:5074–5084, 1999.

K. H. Britten and W. T. Newsome. Tuning bandwidths for near-threshold stimuli in area MT. *Journal of Neurophysiology*, 80:762–770, 1998.

K. H. Britten and R. J. A. van Wezel. Electrical microstimulation of cortical area MST biases heading perception. *Nature Neuroscience*, 1:1383–1398, 1998. doi: 10.1038/259.

K. H. Britten and R. J. Van Wezel. Area MST and heading perception in macaque monkeys. *Cerebral Cortex*, 12:692–701, 2002.

K. H. Britten, M. N. Shadlen, and W. T. Newsome. The analysis of visual motion: a comparison of neuronal and psychophysical performance. *Journal of Neuroscience*, 12: 4745–4765, 1992.

L. Brostek, U. Buttner, M. J. Mustari, and S. Glasauer. Eye velocity gain fields in MSTd during optokinetic stimulation. *Cerebral Cortex*, 2014.

N. A. Browning, S. Grossberg, and E. Mingolla. Cortical dynamics of navigation and steering in natural scenes: Motion-based object segmentation, heading, and obstacle avoidance. *Neural Netw.*, 22(10):1383–1398, December 2009a. ISSN 0893-6080. doi: 10.1016/j.neunet.2009.05.007.

N. A. Browning, S. Grossberg, and E. Mingolla. A neural model of how the brain computes heading from optic flow in realistic scenes. *Cognitive Psychology*, 59(4):320 – 356, 2009b. ISSN 0010-0285. doi: http://dx.doi.org/10.1016/j.cogpsych.2009.07.002.

D. Burke and P. Wenderoth. The effect of interactions between one-dimensional component gratings on 2-dimensional motion perception. *Vision Research*, 33(3):343–350, 1993. doi: 10.1016/0042-6989(93)90090-J.

M. Carandini and D. J. Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13:51–62, 2012.

K. D. Carlson, M. Richert, N. Dutt, and J. L. Krichmar. Biologically plausible models of homeostasis and stdp: Stability and learning in spiking neural networks. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8, Aug 2013. doi: 10.1109/IJCNN.2013.6706961.

K. D. Carlson, J. M. Nageswaran, N. Dutt, and J. L. Krichmar. An efficient automated parameter tuning framework for spiking neural networks. *Frontiers in Neuroscience*, 8 (10), 2014. ISSN 1662-453X. doi: 10.3389/fnins.2014.00010.

A. S. Cassidy, R. Alvarez-Icaza, F. Akopyan, J. Sawada, J. V. Arthur, P. A. Merolla, P. Datta, M. G. Tallada, B. Taba, A. Andreopoulos, A. Amir, S. K. Esser, J. Kusnitz, R. Appuswamy, C. Haymes, B. Brezzo, R. Moussalli, R. Bellofatto, C. Baks, M. Mastro, K. Schleupen, C. E. Cox, K. Inoue, S. Millman, N. Imam, E. McQuinn, Y. Y. Nakamura, I. Vo, C. Guo, D. Nguyen, S. Lekuch, S. Asaad, D. Friedman, B. L. Jackson, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha. Real-time scalable cortical computing at 46 giga-synaptic ops/watt with 100× speedup in time-to-solution and 100,000×

reduction in energy-to-solution. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '14, pages 27–38, Piscataway, NJ, USA, 2014. IEEE Press. ISBN 978-1-4799-5500-8. doi: 10.1109/SC.2014.8.

S. Chatterjee and E. M. Callaway. Parallel colour-opponent pathways to primary visual cortex. *Nature*, 426:668–671, 2003.

J. Chey, S. Grossberg, and E. Mingolla. Neural dynamics of motion grouping: from aperture ambiguity to object speed and direction. *Journal of the Optical Society of America A-Optics Image Science and Vision*, 14:2570–2594, 1997.

T.-S. Chou, L. D. Bucci, and J. L. Krichmar. Learning touch preferences with a tactile robot using dopamine modulated stdp in a model of insular cortex. *Frontiers in Neurorobotics*, 9(6), 2015. ISSN 1662-5218. doi: 10.3389/fnbot.2015.00006.

C. Chubb and G. Sperling. Drift-balanced random stimuli–a general basis for studying non-fourier motion perception. *Journal of the Optical Society of America a–Optics Image Science and Vision*, 5(11):1986–2007, 1988. doi: 10.1364/Josaa.5.001986.

L. Chukoskie and J. A. Movshon. Modulation of visual signals in macaque MT and MST neurons during pursuit eye movement. *Journal of Neurophysiology*, 102(6):3225–3233, 2009.

C. W. Clifford, S. A. Beardsley, and L. M. Vaina. The perception and discrimination of speed in complex motion. *Vision Research*, 39:2213–2227, 1999.

C. L. Colby, J. R. Duhamel, and M. E. Goldberg. Ventral intraparietal area of the macaque: anatomic location and visual response properties. *Journal of Neurophysiology*, 69:902–914, 1993.

NVIDIA Corporation. White paper: NVIDIA's next generation CUDA compute architecture: Fermi'. Technical report, NVIDIA Corporation, 2009. URL `http://www.nvidia.com/content/pdf/fermi_white_papers/nvidia_fermi_compute_architecture_whitepaper.pdf`.

J. A. Crowell and M. S. Banks. Perceiving heading with different retinal regions and types of optic flow. *Perception and Psychophysics*, 53:325–337, 1993.

B. G. Cumming and A. J. Parker. Responses of primary visual cortical neurons to binocular disparity without depth perception. *Neuron*, 389:280–283, 1997.

D. M. Dacey. Parallel pathways for spectral coding in primate retina. *Annu. Rev. Neurosci.*, 23:743–775, 2000.

D. M. Dacey. *The Cognitive Neurosciences*. MIT Press, 2004.

A. P Davison, D. Brüderle, J. M Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger. PyNN: a common interface for neuronal network simulators. *Frontiers in Neuroinformatics*, 2(11), 2009. ISSN 1662-5196. doi: 10.3389/neuro.11.011.2008.

G. C. DeAngelis, I. Ohzawa, and R. D. Freeman. Spatiotemporal organization of simple-cell receptive fields in the cat's striate cortex. i. general characteristics and postnatal development. *Journal of Neurophysiology*, 69(4):1091–1117, 1993. ISSN 0022-3077.

S. Denève, A. Pouget, and P. E. Latham. Divisive normalization, line attractor networks and ideal observers. In *NIPS*, pages 1–7, 1999.

A. P. Duchon and W. H. Jr. Warren. A visual equalization strategy for locomotor control: of honeybees, robots, and humans. *Psychological Science*, 13:272–278, 2002.

C. J. Duffy and R. H. Wurtz. Sensitivity of MST neurons to optic flow stimuli. I. a continuum of response selectivity to large-field stimuli. *Journal of Neurophysiology*, 65:1329–1345, 1991a.

C. J. Duffy and R. H. Wurtz. Sensitivity of MST neurons to optic flow stimuli. II. mechanisms of response selectivity revealed by small-field stimuli. *Journal of Neurophysiology*, 65:1346–1359, 1991b.

C. J. Duffy and R. H. Wurtz. Response of monkey MST neurons to optic flow stimuli with shifted centers of motion. *Journal of Neuroscience*, 15(7 Pt 2):5192–5208, 1995.

C. J. Duffy and R. H. Wurtz. Planar directional contributions to optic flow responses in MST neurons. *Journal of Neurophysiology*, 77:782–796, 1997.

J. Duijnhouwer, A. J. Noest, M. J. Lankheet, A. V. van den Berg, and R. J. van Wezel. Speed and direction response profiles of neurons in macaque MT and MST show modest constraint line tuning. *Frontiers in Behavioral Neuroscience*, 7, 2013.

B. Eckel. *Thinking in C++, Volume 1: Introduction to Standard C++*. Prentice Hall, 2 edition, 2000. ISBN 0-13-979809-9.

J. Eggert and E. Korner. Sparse coding and NMF. *IEEE IJCNN*, pages 2529–2533, 2004.

S. Eifuku and R. H. Wurtz. Response to motion in extrastriate area MSTl: centersurround interactions. *Journal of Neurophysiology*, 80:282–296, 1998.

D. M. Elder, S. Grossberg, and E. Mingolla. A neural model of visually guided steering, obstacle avoidance, and route selection. *Journal of Experimental Psychology: Human Perception and Performance*, 29:343–362, 2009.

C. Eliasmith and C. H. Anderson. *Neural Engineering (Computational Neuroscience Series): Computational, Representation, and Dynamics in Neurobiological Systems*. MIT Press, Cambridge, MA, USA, 2002. ISBN 0262050714.

C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen. A large-scale model of the functioning brain. *Science*, 338(6111):1202–1205, 2012. ISSN 0036-8075. doi: 10.1126/science.1225266.

B. R. Fajen and W. H. Warren. Behavioral dynamics of steering, obstacle avoidance, and route selection. *Journal of Experimental Psychology: Human Perception and Performance*, 29:343–362, 2003.

D. Felleman and D. van Essen. Distributed hierarchical processing in the primate cerebral cortex. *Cereb. Cortex*, 1:1–47, 1991.

V. P. Ferrera and H. R. Wilson. Perceived direction of moving two-dimensional patterns. *Vision Research*, 30(2):273–287, 1990.

C. R. Fetsch, S. Wang, Y. Gu, G. C. DeAngelis, and D. E. Angelaki. Spatial reference frames of visual, vestibular, and multimodal heading signals in the dorsal subdivision of the medial superio temporal area. *Journal of Neuroscience*, 27:700–712, 2007.

A. K. Fidjeland, E. B. Roesch, M. P. Shanahan, and W. Luk. Nemo: A platform for neural modelling of spiking neurons using gpus. In *Application-specific Systems, Architectures and Processors, 2009. ASAP 2009. 20th IEEE International Conference on*, pages 137–144, July 2009. doi: 10.1109/ASAP.2009.24.

D. T. Field, R. M. Wilkie, and J. P. Wann. Neural systems in the visual control of steering. *Journal of Neuroscience*, 27:8002–8010, 2007.

G. D. Field and E. J. Chichilnisky. Information processing in the primate brain: circuitry and coding. *Annu. Rev. Neurosci.*, 30:1–30, 2007.

W. T. Freeman and E. H. Adelson. The design and use of steerable filters. In *IEEE Pattern Analysis and Machine Intelligence*, volume 13, pages 891–906, 1991.

W. T. Freeman and E. Simoncelli. Metamers of the ventral stream. *Nature Neuroscience*, 14:1195–1201, 2011.

K. Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11:127–138, 2010.

K. Friston, J. Kilner, and L. Harrison. A free energy principle for the brain. *Journal of Physiology*, 100:70–87, 2006.

M. T. Froehler and C. J. Duffy. Cortical neurons encoding path and place: where you go is where you are. *Science*, 295:2462–2465, 2002.

S. Furber and S. Temple. Neural systems engineering. *Journal of the Royal Society Interface*, 4:193–206, 2007.

M. Gewaltig and M. Diesmann. NEST (NEural Simulation Tool). *Scholarpedia*, 2(4):1430, 2007. revision #130182.

J. J. Gibson. *The perception of the visual world*. Houghton Mifflin, 1950.

J. J. Gibson. Visually controlled locomotion and visual orientation in animals. *British Journal of Psychology*, 49:182–194, 1958.

M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. *Trends in Neuroscience*, 15(1):20–25, 1992.

D. F. M. Goodman and R. Brette. Brian: a simulator for spiking neural networks in Python. *Frontiers in Neuroinformatics*, 2(5), 2008. ISSN 1662-5196. doi: 10.3389/neuro.11.005.2008.

M. S. Graziano, R. A. Andersen, and R. J. Snowden. Tuning of MST neurons to spiral motions. *Journal of Neuroscience*, 14:54–67, 1994.

S. Grossberg and P. K. Pilly. Temporal dynamics of decision-making during motion perception in the visual cortex. *Vision Research*, 48(12):1345–1373, 2008. doi: 10.1016/j.visres.2008.02.019.

S. Grossberg, E. Mingolla, and C. Pack. A neural model of motion processing and visual navigation by cortical area MST. *Cerebral Cortex*, 9:878–895, 1999.

Y. Gu, P. V. Watkins, D. E. Angelaki, and G. C. DeAngelis. Visual and nonvisual contributions to three-dimensional heading selectivity in the medial superior temporal area. *Journal of Neuroscience*, 26:73–85, 2006.

Y. Gu, C. R. Fetsch, B. Adeyemo, G. C. DeAngelis, and D. E. Angelaki. Decoding of MSTd population activity accounts for variations in the precision of heading perception. *Neuron*, 66:596–609, 2010.

Y. Gu, G. C. DeAngelis, and D. E. Angelaki. Causal links between dorsal medial superior temporal area neurons and multisensory heading perception. *Journal of Neurosciecne*, 32:2299–2313, 2012.

S. Ben Hamed, W. Page, C. Duffy, and A. Pouget. MSTd neuronal basis functions for the population encoding of heading direction. *Journal of Neurophysiology*, 90:549–558, 2003.

T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Verlag, 2nd edition, 2009.

M. J. Hawken, A. J. Parker, and J. S. Lund. Laminar organization and contrast sensitivity of direction-selective cells in the striate cortex of Old World monkey. *Journal of Neuroscience*, pages 3541–3548, 1988.

D. O. Hebb. *The organization of behavior; a neurophysiological theory*. Wiley, 1949.

D. J. Heeger, E. P. Simoncelli, and J. A. Movshon. Computational models of cortical visual processing. *Proc. Natl. Acad. Sci. USA*, 93:3398–3408, 1996.

R. V Hoang, D. Tanna, Laurence C. Jayet B., S. M. Dascalu, and F. C. Harris. A novel cpu/gpu simulation environment for large-scale biologically-realistic neural modeling. *Frontiers in Neuroinformatics*, 7(19), 2013. ISSN 1662-5196. doi: 10.3389/fninf.2013.00019.

A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and application to conduction and excitation in nerve. *Journal of Physiology*, 117:500–544, 1954.

S. S. Hohl, K. S. Chaisanguanthum, and S. G. Lisberger. Sensory population decoding for visually guidedmovements. *Neuron*, 79(1):167–179, 2013. doi: 10.1016/j.neuron.2013.05.026.

C. J. Honey, R. Kötter, M. Breakspear, and O. Sporns. Network structure of cerebral cortex shapes functional connectivity on multiple time scales. *Proceedings of the National Academy of Sciences*, 104(24):10240–10245, 2007. doi: 10.1073/pnas.0701519104.

P. O. Hoyer. Non-negative sparse coding. In *Proceedings in Neural Networks for Signal Processing XII*, pages 557–565, 2002.

W. H. Huang, B. R. Fajen, J. R. Fink, and W. H. Warren. Visual navigation and obstacle avoidance using a steering potential function. *Robotics and Autonomous Systems*, 54: 288–299, 2006.

E. M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, Nov 2003. ISSN 1045-9227. doi: 10.1109/TNN.2003.820440.

E. M. Izhikevich. Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070, 2004. doi: 10.1109/Tnn.2004.832719.

E. M. Izhikevich and N. S. Desai. Relating stdp to bcm. *Neural Computation*, 15(7):1511–1523, 2003.

E. M. Izhikevich, J. A. Gally, and G. M. Edelman. Spike-timing dynamics of neuronal groups. *Cerebral Cortex*, 14(8):933–944, 2004. doi: 10.1093/cercor/bhh053.

Eugene M. Izhikevich. *Dynamical systems in neuroscience : the geometry of excitability and bursting*. Computational neuroscience. MIT Press, Cambridge, Mass., London, 2007. ISBN 0-262-09043-0.

P. Janssen, R. Vogels, Y. Liu, and G. A. Orban. At least at the level of inferior temporal cortex, the stereo correspondence problem is solved. *Neuron*, 37:693–701, 2003.

M. Jazayeri and J. A. Movshon. Optimal representation of sensory information by neural populations. *Nature Neuroscience*, 9:690–696, 2006.

E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. A. Siegelbaum, and A. J. Hudspeth. *Principles of neural science*, volume 4. McGraw-Hill, New York, 2000.

M. M. Khan, D. R. Lester, L. A. Plana, A. Rast, X. Jin, E. Painkras, and S. B. Furber. Spinnaker: Mapping neural networks onto a massively-parallel chip multiprocessor. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 2849–2856, June 2008. doi: 10.1109/IJCNN.2008.4634199.

H. R. Kim, D. E. Angelaki, and G. C. DeAngelis. A novel role for visual perspective cues in the neural computation of depth. *Nature Neuroscience*, 18:129–137, 2015.

C. Koch. *Biophysics of Computation: Information Processing in Single Neurons*. Computational Neuroscience Series. Oxford University Press, USA, 2004. ISBN 9780195181999.

J. J. Koenderink and A. J. van Doorn. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta*, 22:773–791, 1975.

A. Kolawa and D. Huizinga. *Automated Defect Prevention: Best Practices in Software Management*. Wiley-IEEE Computer Society Press, 2007.

H. Komatsu and R. H. Wurtz. Relation of cortical areas MT and MST to pursuit eye movements. III. Interaction with full-field visual stimulation. *Journal of Neurophysiology*, 60(2):621–644, 1988.

G. K. Kountouriotis, K. A. Shire, C. D. Mole, P. H. Gardner, N. Merat, and R. M. Wilkie. Optic flow asymmetries bias high-speed steering along roads. *Journal of Vision*, 13:23, 2013.

H. G. Krapp. Neuronal matched filters for optic flow processing in flying insects. *Int Rev Neurobiol*, 44:93–120, 2000.

H. G. Krapp and R. Hengstenberg. Estimation of self-motion by optic flow processing in single visual interneurons. *Nature*, 384:463–466, 1996.

J. L. Krichmar and G. M. Edelman. Principles underlying the construction of brain-based devices. In T. Kovacs and J. A. R. Marshall, editors, *Adaptation in artificial and biological systems*, volume 6, pages 37–42. Bristol, UK: Society for the Study of Artificial Intelligence and the Simulation of Behaviour, 2006.

K. Krug, B. G. Cumming, and A. J. Parker. Responses of single MT neurons to anti-correlated stereograms in the awake macaque. *Soc. Neurosci. Abstr.*, 25:275, 1999.

L. Lagae, H. Maes, S. Raiguel, D. K. Xiao, and G. A. Orban. Responses of macaque STS neurons to optic flow components: a comparison of areas MT and MST. *Journal of Neurophysiology*, 71:1597–1626, 1994.

T. D. Lamb, S. P. Collin, and E. N. Pugh. Evolution of the vertebrate eye: opsins, photoreceptors, retina and eye cup. *Nature Reviews Neuroscience*, 8:960–976, 2007.

M. Lappe and J. P. Rauschecker. Heading detection from optic flow. *Nature*, 369:712–713, 1994.

M. Lappe, F. Bremmer, M. Pekel, A. Thiele, and K. P. Hoffmann. Optic flow processing in monkey STS: a theoretical and experimental approach. *Journal of Neuroscience*, 16: 6265–6285, 1996.

O. W. Layton, E. Mingolla, and N. A. Browning. A motion pooling model of visually guided navigation explains human behavior in the presence of independently moving objects. *Journal of Vision*, 12(1), 2012. doi: doi:10.1167/12.1.20.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998. ISSN 0018-9219. doi: 10.1109/5.726791.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–91, 1999. doi: 10.1038/44565.

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13*, 13:556–562, 2001. ISSN 1049-5258.

G. R. Leichnetz. Connections of the medial posterior parietal cortex (area 7m) in the monkey. *Anat. Rec.*, 263:215–236, 2001.

J. W. Lewis and D. C. van Essen. Corticocortical connections of visual, sensorimotor, and multimodel processing areas in the parietal lobe of the macaque monkey. *Journal of Computational Neurology*, 428:112–137, 2000.

L. Li and W. .H. Jr. Warren. Perception of heading during rotation: sufficiency of dense motion parallax and reference objects. *Vision Research*, 40:3873–3894, 2000.

P. Lichtsteiner, C. Posch, and T. Delbruck. A 128x128 120 dB 15 us latency asynchronous temporal contrast vision sensor. *Solid-State Circuits, IEEE Journal of*, 43(2):566–576, Feb 2008. ISSN 0018-9200. doi: 10.1109/JSSC.2007.914337.

R. Linsker. Perceptual neural organization: some approaches based on network models and information theory. *Annu Rev Neurosci*, 13:257–281, 1990.

S. G. Lisberger and V. P. Ferrera. Vector averaging for smooth pursuit eye movements initiated by two moving targets in monkeys. *Journal of Neuroscience*, 17:7490–7502, 1997.

S.-C. Liu, A. van Schaik, B. A. Minch, and T. Delbruck. Event-based 64-channel binaural silicon cochlea with q enhancement mechanisms. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 2027–2030, May 2010. doi: 10.1109/ISCAS.2010.5537164.

M. S. Livingstone and B. R. Conway. Contrast affects speed tuning, space-time slant, and receptive-field organization of simple cells in macaque v1. *Journal of Neurophysiology*, 97 (1):849–857, 2007. doi: 10.1152/jn.00762.2006.

M. S. Livingstone and D. H. Hubel. Anatomy and physiology of a color system in the primate visual cortex. *Journal of Neuroscience*, pages 309–356, 1984.

D. J. Logan and C. J. Duffy. Cortical area MSTd combines visual cues to represent 3-d self-movement. *Cerebral Cortex*, 16:1494–1507, 2006.

H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In *Proceedings of the Royal Society of London*, volume B 208, pages 385–397, 1980.

Z. L. Lu and G. Sperling. Attention-generated apparent motion. *Nature*, 377:237–239, 1995. doi: 10.1038/377237a0.

W. Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659 – 1671, 1997. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/S0893-6080(97)00011-7.

W. Maass and C. M. Bishop. *Pulsed neural networks*. MIT Press, 2001.

N. J. Majaj, M. Carandini, and J. A. Movshon. Motion integration by neurons in macaque mt is local, not global. *Journal of Neuroscience*, 27(2):366–370, 2007. doi: 10.1523/JNEUROSCI.3183-06.2007.

K. V. Mardia. *Statistics of directional data*. Academic Press, 1972.

D. Marr. *Vision*. MIT Press, 1982. ISBN 9780262514620.

J. H. Maunsell and D. C. van Essen. Functional properties of neurons in middle temporal visual area of the macaque monkey. II. binocular interactions and sensitivity to binocular disparity. *Journal of Neurophysiology*, 49:1148–1167, 1983.

J. H. Maunsell and D. C. van Essen. Topographic organization of the middle temporal visual area in the macaque monkey: representational biases and the relationship to callosal connections and myeloarchitectonic boundaries. *Journal of Computational Neurology*, 266 (4):535–555, 1987.

P. J. Mineault, F. A. Khawaja, D. A. Butts, and C. C. Pack. Hierarchical processing of complex motion along the primate dorsal visual pathway. *Proc Natl Acad Sci USA*, 109: E972–980, 2012.

K. Minkovich, C.M. Thibeault, M.J. O'Brien, A. Nogin, Youngkwan Cho, and N. Srinivasa. HRLSim: A high performance spiking neural network simulator for GPGPU clusters. *Neural Networks and Learning Systems, IEEE Transactions on*, 25(2):316–331, Feb 2014. doi: 10.1109/TNNLS.2013.2276056.

A. H. Morice, M. Francois, D. M. Jacobs, and G. Montagne. Environmental constraints modify the way an interceptive action is controlled. *Experimental Brain Research*, 202: 397–411, 2010.

A. P. Morris, M. Kubischik, K. P. Hoffmann, B. Krekelberg, and F. Bremmer. Dynamics of eye-position signals in the dorsal visual system. *Current Biology*, 22:173–179, 2012.

J. A. Movshon and W. T. Newsome. Visual response properties of striate cortical neurons projecting to area MT in macaque monkeys. *The Journal of Neuroscience*, 16(23):7733–7741, December 1996. ISSN 0270-6474.

J. A. Movshon, E. H. Adelson, M. S. Gizzi, and W. T. Newsome. *The analysis of moving visual patterns (Pattern recognition mechanisms.* New York: Springer, 1985.

C. I. Myme, K. Sugino, G. G. Turrigiano, and S. B. Nelson. The NMDA-to-AMPA ratio at synapses onto layer 2/3 pyramidal neurons is conserved across prefrontal and visual cortices. *Journal of Neurophysiology*, 90:771–779, 2003.

J. M. Nageswaran, N. D., J. L. Krichmar, A. Nicolau, and A. V. Veidenbaum. A configurable simulation environment for the efficient simulation of large-scale spiking neural networks on graphics processors. *Neural Networks*, 22(56):791 – 800, 2009. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/j.neunet.2009.06.028. Advances in Neural Networks Research: (IJCNN'09) International Joint Conference on Neural Networks.

J. M. Nageswaran, M. Richert, N. Dutt, and J. L. Krichmar. Towards reverse engineering the brain: Modeling abstractions and simulation frameworks. In *IEEE/IFIP VLSI System on Chip Conference (VLSI-SoC)*, pages 1–6, 2010.

J. J. Nassi and E. M. Callaway. Specialized circuits from primary visual cortex to V2 and area MT. *Neuron*, 55:799–808, 2007.

J. J. Nassi and E. M. Callaway. Parallel processing strategies of the primate visual system. *Nature Reviews Neuroscience*, 10:360–372, 2009.

W. T. Newsome, R. H. Wurtz, and H. Komatsu. Relation of cortical areas MT and MST to pursuit eye movements. ii. differentiation of retinal from extraretinal inputs. *Journal of Neurophysiology*, 60(2):604–620, 1988.

W. T. Newsome, K. H. Britten, and J. A. Movshon. Neuronal correlates of a perceptual decision. *Nature*, 341:52–54, 1989.

S. Nishida. Advancement of motion psychophysics: review 20012010. *Journal of Vision*, 11 (5), 2011.

T. Nordström and B. Svensson. Using and designing massively parallel computers for artificial neural networks. *J. Parallel Distrib. Comput.*, 14(3):260–285, March 1992. ISSN 0743-7315. doi: 10.1016/0743-7315(92)90068-X.

H. Nover, C. H. Anderson, and G. C. DeAngelis. A logarithmic, scale-invariant representation of speed in macaque middle temporal area accounts for speed discrimination performance. *Journal of Neuroscience*, 25(43):10049–10060, 2005.

Thomas Nowotny. Flexible neuronal network simulation framework using code generation for NVidia CUDA™. *BMC Neuroscience*, 12(1):1–2, 2011. ISSN 1471-2202. doi: 10.1186/1471-2202-12-S1-P239.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.

G. A. Orban. Higher order visual processing in macaque extrastriate cortex. *Physiological Reviews*, 88:59–89, 2008.

G. A. Orban, L. Lagae, A. Verri, S. Raiguel, D. Xiao, H. Maes, and V. Torre. First-order analysis of optical flow in monkey brain. *Proceedings of the National Academy of Science of the United States of America*, 89:2595–2599, 1992.

N. Oros and J. L. Krichmar. Neuromodulation, attention and localization using a novel ANDROID robotic platform. In *IEEE international conference on development and learning and epigenetic robotics*, 2012.

N. Oros and J. L. Krichmar. Android^{TM} based robotics: Powerful, flexible and inexpensive robots for hobbyists, educators, students and researcher. Technical report, Cognitive Anteater Robotics Laboratory: University of California, Irvine, 2013a.

N. Oros and J. L. Krichmar. Smartphone based robotics: Powerful, flexible and inexpensive robots for hobbyists, educators, students and researchers. Technical report, CECS technical report. Center for Embedded Computer Systems: University of California, Irvine, 2013b.

P. Paatero and U. Tapper. Positive matrix factorization - a nonnegative factor model with optimal utilization for error-estimates of data values. *Environmetrics*, 5:111–126, 1994.

C. C. Pack and R. Born. Cortical mechanisms for the integration of visual motion. In A. I. Basbaum, A. Kaneko, G. M. Shepherd, and G. Westheimer, editors, *The Senses: A comprehensive reference*, volume 2 of *Vision II*, chapter 2, pages 189–218. Academic Press, San Diego, 2008.

C. C. Pack, V. K. Berezovskii, and R. T. Born. Dynamic properties of neurons in cortical area mt in alert and anaesthetized macaque monkeys. *Nature*, 414(6866):905–908, 2001.

W. K. Page and C. J. Duffy. MST neuronal responses to heading direction during pursuit eye movements. *Journal of Neurophysiology*, 81:596–610, 1999.

W. K. Page and C. J. Duffy. Heading representation in MST: sensory interactions and population encoding. *Journal of Neurophysiology*, 89(4):1994–2013, 2003.

W. K. Page, N. Sato, M. T. Froehler, W. Vaughn, and C. J. Duffy. Navigational path integration by cortical neurons: origins in higher-order direction selectivity. *Journal of Neurophysiology*, 113:1896–1906, 2015.

J. Parvizi, G. W. van Hoesen, J. Buckwalter, and A. Damasio. Neural connections of the posteromedial cortex in the macaque. *Proceedings of the National Academy of Sciences U.S.A.*, 103:1563–3665, 2006.

A. E. Patla and J. N. Vickers. Where and when do we look as we approach and step over an obstacle in the travel path? *NeuroReport*, 8:3661–3665, 1997.

K. Pauwels, N. Kruger, M. Lappe, F. Worgotter, and M. M. van Hulle. A cortical architecture on parallel hardware for motion processing in real time. *Journal of Vision*, 10:18, 2010.

D. Pecevski, T. Natschläger, and K. Schuch. PCSIM: a parallel simulation environment for neural circuits fully integrated with python. *Frontiers in Neuroinformatics*, 3(11), 2009. ISSN 1662-5196. doi: 10.3389/neuro.11.011.2009.

J. A. Perrone. Model for the computation of self-motion in biological systems. *Journal of the Optical Society of America A, Optics and image science*, 9:177–194, 1992.

J. A. Perrone. A neural-based code for computing image velocity from small sets of middle temporal (mt/v5) neuron inputs. *Journal of Vision*, 12(8), 2012. doi: 10.1167/12.8.1.

J. A. Perrone and L. S. Stone. A model of self-motion estimation within primate extrastriate visual cortex. *Vision Research*, 34:2917–2938, 1994.

J. A. Perrone and L. S. Stone. Emulating the visual receptive-field properties of MST neurons with a template model of heading estimation. *Journal of Neuroscience*, 18:5958–5975, 1998.

J. A. Perrone and A. Thiele. Speed skills: measuring the visual speed analyzing properties of primate mt neurons. *Nature Neuroscience*, 4(5):526–532, 2001.

J. A. Perrone and A. Thiele. A model of speed tuning in mt neurons. *Vision Research*, 42 (8):1035–1051, 2002.

R. R. Picard and R. D. Cook. Cross-validation of regression-models. *Journal of the American Statistics Association*, 79:575–583, 1984.

A. Pouget and T. J. Sejnowski. Spatial transformations in the parietal cortex using basis functions. *Journal of Cognitive Neuroscience*, 9:222–237, 1997.

A. Pouget and L. H. Snyder. Computational approaches to sensorimotor transformations. *Nature Neuroscience*, 3 Suppl:1192–1198, 2000.

A. Pouget, K. Zhang, S. Deneve, and P. E. Latham. Statistically efficient estimation using population coding. *Neural Computation*, 10:373–401, 1998.

N. J. Priebe, C. R. Cassanello, and S. G. Lisberger. The neural representation of speed in macaque area mt/v5. *Journal of Neuroscience*, 23(13):5650–5661, 2003.

N. J. Priebe, S. G. Lisberger, and J. A. Movshon. Tuning for spatiotemporal frequency and speed in directionally selective neurons of macaque striate cortex. *Journal of Neuroscience*, 26(11):2941–2950, 2006.

G. Purushothaman and D. C. Bradley. Neural population code for fine perceptual decisions in area MT. *Nature Neuroscience*, 8:99–106, 2005.

N. Qian and R. A. Andersen. Transparent motion perception as detection of unbalanced motion signals. *Journal of Neuroscience*, 14:7367–7380, 1994.

D. Querlioz, O. Bichler, and C. Gamrat. Simulation of a memristor-based spiking neural network immune to device variations. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1775–1781, July 2011. doi: 10.1109/IJCNN.2011.6033439.

S. Raiguel, M. M. van Hulle, D. K. Xiao, V. L. Marcar, and G. A. Orban. Shape and spatial distribution of receptive fields and antagonistic motion surrounds in the middle temporal area (V5) of the macaque. *European Journal of Neuroscience*, 7:2064–2082, 1995.

S. Raiguel, M. M. van Hulle, D. K. Xiao, V. L. Marcar, L. Lagae, and G. A. Orban. Size and shape of receptive fields in the medial superior temporal area (mst) of the macaque. *Neuroreport*, 8:2803–2808, 1997.

F. Raudies. Optic flow. *Scholarpedia*, 8(7):30724, 2013. revision #149632.

F. Raudies and H. Neumann. A review and evaluation of methods estimating ego-motion. *Comput Vis Image Und*, 116:606–633, 2012.

F. Raudies, E. Mingolla, and H. Neumann. A model of motion transparency processing with local center-surround interactions and feedback. *Neural Computation*, 23(11):2868–2914, 2011.

G. H. Recanzone, R. H. Wurtz, and U. Schwartz. Responses of MT and MST neurons to one and two moving objects in the receptive field. *Journal of Neurophysiology*, 78:2904–2915, 1997.

A. Resulaj, R. Kiani, D. M. Wolpert, and M. N. Shadlen. Changes of mind in decision-making. *Nature*, 461(7261):263–U141, 2009.

M. Richert, J. M. Nageswaran, N. Dutt, and J. L. Krichmar. An efficient simulation environment for modeling large-scale cortical processing. *Frontiers in Neuroinformatics*, 5 (19), 2011. doi: 10.3389/fninf.2011.00019.

M. Richert, T. D. Albright, and B. Krekelberg. The complex structure of receptive fields in the middle temporal area. *Frontiers in Systems Neuroscience*, 7, 2013.

F. Rieke, D. Warland, R. de Ruyter van Steveninck, and W. Bialek. *Spikes: Exploring the Neural Code*. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-18174-6.

G. Rizzolatti and M. Matelli. Two different streams form the dorsal visual system: anatomy and functions. *Experimental Brain Research*, 153:146–157, 2003.

H. R. Rodman and T. D. Albright. Coding of visual stimulus velocity in area MT of the macaque. *Vision Research*, 27(12):2035–2048, 1987.

H. R. Rodman and T. D. Albright. Single-unit analysis of pattern motion selective properties in the middle temporal visual area (MT). *Experimental Brain Research*, 75(1):53–64, 1989.

J. D. Roitman and M. N. Shadlen. Response of neurons in the lateral intraparietal area during a combined visual discrimination reaction time task. *Journal of Neuroscience*, 22 (21):9475–9489, 2002.

S. Rotter and M. Diesmann. Exact digital simulation of time-invariant linear systems with applications to neuronal modeling. *Biological Cybernetics*, 81(5):381–402, 1999. ISSN 1432-0770. doi: 10.1007/s004220050570.

C. S. Royden, J. A. Crowell, and M. S. Banks. Estimating heading during eye movements. *Vision Research*, 34:3197–3214, 1994.

S. K. Rushton, J. M. Harris, M. R. Lloyd, and J. P. Wann. Guidance of locomotion on foot uses perceived target location rather than optic flow. *Current Biology*, 8:19, 1998. doi: 10.1016/S0960-9822(07)00492-7.

S. K. Rushton, J. Wen, and R. S. Allison. Egocentric direction and the visual guidance of robot locomotion background, theory and implementation. In *Biologically motivated computer vision, proceedings*, volume 2525, pages 576–591, 2002.

N. C. Rust, V. Mante, E. P. Simoncelli, and J. A. Movshon. How MT cells analyze the motion of visual patterns. *Nature Neuroscience*, 9(11):1421–1431, Nov 2006. doi: 10.1038/nn1786.

H. Saito, M. Yukie, K. Tanaka, K. Hikosaka, Y. Fukada, and E. Iwai. Integration of direction signals of image motion in the superior temporal sulcus of the macaque monkey. *Journal of Neuroscience*, 6(1):145–157, 1986.

J. Schemmel, D. Brüderle, A. Grübl, M. Hock, K. Meier, and S. Millner. A wafer-scale neuromorphic hardware system for large-scale neural modeling. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 1947–1950, May 2010. doi: 10.1109/ISCAS.2010.5536970.

T. J. Sejnowski. Statistical constraints on synaptic plasticity. *Journal of Theoretical Biology*, 69:385–389, 1977.

W. Senn, H. Markram, and M. Tsodyks. An algorithm for modifying neurotransmitter release probability based on pre- and post-synaptic spike timing. *Neural Computation*, 13 (1):35–67, 2001.

J. Seubert, G. W. Humphreys, H. J. Muller, and K. Gramann. Straight after the turn: the role of the parietal lobes in egocentric space processing. *Neurocase*, 14:204–219, 2008.

H. S. Seung and H. Sompolinsky. Simple models for reading neuronal population codes. *Proceedings of the National Academy of Sciences of the United States of America*, 90: 10749–10753, 1993.

M. N. Shadlen and W. T. Newsome. Neural basis of a perceptual decision in the parietal cortex (area LIP) of the rhesus monkey. *Journal of Neurophysiology*, 86(4):1916–1936, 2001.

E. P. Simoncelli and D. J. Heeger. A model of neuronal responses in visual area {MT}. *Vision Research*, 38(5):743 – 761, 1998. ISSN 0042-6989. doi: 10.1016/S0042-6989(97)00183-1.

E. P. Simoncelli and B. A. Olshausen. Natural image statistics and neural representation. *Annu Rev Neurosci*, 24:1193–1216, 2001.

L. C. Sincich and J. C. Horton. The circuitry of V1 and V2: integration of color, form, and motion. *Annu. Rev. Neurosci*, 28:303–326, 2004.

M. A. Smith, N. J. Majaj, and J. A. Movshon. Dynamics of motion signaling by neurons in macaque area MT. *Nature Neuroscience*, 8(2):220–228, 2005.

P. L. Smith and R. Ratcliff. Psychology and neurobiology of simple decisions. *Trends in Neurosciences*, 27(3):161 – 168, 2004. ISSN 0166-2236. doi: http://dx.doi.org/10.1016/j.tins.2004.01.006.

R. J. Snowden, S. Treue, R. G. Erickson, and R. A. Andersen. The response of area MT and V1 neurons to transparent motion. *Journal of Neuroscience*, 11:2768–2785, 1991.

J. P. Snyder. *Map projections–a working manual*. Geological Survey (U.S.), 1987.

S. Song, K. D. Miller, and L. F. Abbott. Competitive hebbian learning through spike-timing dependent plasticity. *Nature Neuroscience*, 3:919–926, 2000.

A. Spanne and H. Jorntell. Questioning the role of sparse coding in the brain. *Trends Neurosci*, 38:417–427, 2015.

N. Srinivasa and J. M. Cruz-Albrecht. Neuromorphic adaptive plastic scalable electronics: Analog learning systems. *Pulse, IEEE*, 3(1):51–56, Jan 2012. ISSN 2154-2287. doi: 10.1109/MPUL.2011.2175639.

M. Srinivasan, S. Zhang, M. Lehrer, and T. Collett. Honeybee navigation en route to the goal: visual flight control and odometry. *Journal of Experimental Biology*, 199:237–244, 1996.

M. V. Srinivasan and S. W. Zhang. Visual control of honeybee flight. *EXS*, 84:95–113, 1997.

G. Stent. A physiological mechanism for hebb's postulate of learning. *Proceedings of the National Academy of Sciences, U.S.A*, 70:997, 1973.

N. V. Swindale. Orientation tuning curves: empirical description and estimation of parameters. *Biological Cybernetics*, 78:45–56, 1998.

K. Takahashi, Y. Gu, P. J. May, S. D. Newlands, G. C. DeAngelis, and D. E. Angelaki. Multimodal coding of three-dimensional rotation and translation in area MSTd: comparison of visual and vestibular selectivity. *Journal of Neuroscience*, 27:9742–9756, 2007.

K. Tanaka and H. Saito. Analysis of motion of the visual field by direction, expansion/contraction, and rotation cells clustered in the dorsal part of the medial superior temporal area of the macaque monkey. *Journal of Neurophysiology*, 62(3):626–641, 1989.

K. Tanaka, Y. Fukada, and H. A. Saito. Underlying mechanisms of the response specificty of expansion/contraction and rotation cells in the dorsal part of the medial superior temporal area of the macaque monkey. *Journal of Neurophysiology*, 62:642–656, 1989.

K. Tanaka, Y. Sugita, M. Moriya, and H. Saito. Analysis of object motion in the ventral part of the medial superior temporal area of the macaque visual cortex. *Journal of Neurophysiology*, 69:128–142, 1993.

A. Thiele, K. R. Dobkins, and T. D. Albright. Neural correlates of chromatic motion perception. *Neuron*, 32(2):351–358, 2001.

M. Tsodyks, K. Pawelzik, and H. Markram. Neural networks with dynamic synapses. *Neural Computation*, 10(4):821–835, 1998.

S. Ullman. *The interpretation of visual motion*. Artificial Intelligence. MIT Press, 1979. ISBN 978-026271011-4.

L. G. Ungerleider and R. Desimone. Cortical connections of visual area MT in the macaque. *Journal of Comparative Neurology*, 248(2):190–222, 1986.

L. G. Ungerleider and M. Mishkin. Two cortical visual systems. In *Analysis of visual behavior*, pages 549–586. MIT Press, 1982.

D. C. van Essen, J. H. Maunsell, and J. L. Bixby. The middle temporal visual area in the macaque: myeloarchitecture, connections, functional properties and topographic organization. *Journal of Computational Neurology*, 199(3):293–326, 1981.

D. C. van Essen, H. A. Drury, J. Dickson, J. Harwell, D. Hanlon, and C. H. Anderson. An integrated software suite for surface-based analyses of cerebral cortex. *J. Am. Med. Inform. Assoc.*, 8:443–459, 2001.

Jan P. H. van Santen and George Sperling. Elaborated reichardt detectors. *J. Opt. Soc. Am. A*, 2(2):300–321, Feb 1985. doi: 10.1364/JOSAA.2.000300.

W. E. Vinje and J. L. Gallant. Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*, 287:1273–1276, 2000.

T. P. Vogels and L. F. Abbott. Signal propagation and logic gating in networks of integrate-and-fire neurons. *The Journal of Neuroscience*, 25(46):10786–10795, 2005. doi: 10.1523/JNEUROSCI.3508-05.2005.

H. von Helmholtz. *Treatise on physiological objects*. Dover Publications, 1925.

M. B. Wall and A. T. Smith. The representation of egomotion in the human brain. *Current Biology*, 18:191–194, 2008.

H. Wallach. Perceiving a stable environment when one moves. *Annu Rev Psychol*, 38:1–27, 1987.

W. H. Jr. Warren, B. A. Kay, W. D. Zosh, A. P. Duchon, and S. Sahuc. Optic flow is used to control human walking. *Nature Neuroscience*, 4:213–216, 2001.

B. Wen and K. Boahen. A silicon cochlea with active coupling. *IEEE Transactions on Biomedical Circuits and Systems*, 3:444–455, 2009.

R. Wilkie and J. Wann. Controlling steering and judging heading: retinal flow, visual direction, and extraretinal information. *Journal of Experimental Psychology: Human Perception and Performance*, 29:363–378, 2003.

H. R. Wilson, V. P. Ferrera, and C. Yo. A psychophysically motivated model for 2-dimensional motion perception. *Visual Neuroscience*, 9(1):79–97, 1992.

D. K. Xiao, S. Raiguel, V. Marcar, and G. A. Orban. The spatial distribution of the antagonistic surround of MT/V5 neurons. *Cerebral Cortex*, 7:662–677, 1997.

H. Xu, P. Wallisch, and D. C. Bradley. Spiral motion selective neurons in area MSTd contribute to judgments of heading. *Journal of Neurophysiology*, 111:2332–2342, 2014.

C. P. Yu, W. K. Page, R. Gaborski, and C. J. Duffy. Receptive field dynamics underlying MST neuronal optic flow selectivity. *Journal of Neurophysiology*, 103:531–547, 2010.

S. Zeki and S. Shipp. The functional logic of cortical connections. *Nature*, 335:311–317, 1988.

R. S. Zemel and T. J. Sejnowski. A model for encoding multiple object motions and self-motion in area MST of primate visual cortex. *Journal of Neuroscience*, 18:531–547, 1998.

K. C. Zhang, M. I. Sereno, and M. E. Sereno. Emergence of position-independent detectors of sense of rotation and dilation with Hebbian learning - an analysis. *Neural Computation*, 5:597–612, 1993.