# A Neurocomputational Model of Smooth Pursuit Control to Interact with the Real World

by

Seyed Omid Sadat Rezai

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2014

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

## Abstract

Whether we want to drive a car, play a ball game, or even enjoy watching a flying bird, we need to track moving objects. This is possible via smooth pursuit eye movements (SPEMs), which maintain the image of the moving object on the fovea (i.e., a very small portion of the retina with high visual resolution).

At first glance, performing an accurate SPEM by the brain may seem trivial. However, imperfect visual coding, processing and transmission delays, wide variety of object sizes, and background textures make the task challenging. Furthermore, the existence of distractors in the environment makes it even more complicated and it is no wonder why understanding SPEM has been a classic question of human motor control.

To understand physiological systems of which SPEM is an example, creation of models has played an influential role. Models make quantitative predictions that can be tested in experiments. Therefore, modelling SPEM is not only valuable to learn neurobiological mechanisms of smooth pursuit or more generally gaze control but also beneficial to give insight into other sensory-motor functions.

In this thesis, I present a neurocomputational SPEM model based on Neural Engineering Framework (NEF) to drive an eye-like robot. The model interacts with the real world in real time. It uses naturalistic images as input and by the use of spiking model neurons controls the robot. This work can be the first step towards more thorough validation of abstract SPEM control models. Besides, it is a small step toward neural models that drive robots to accomplish more intricate sensory-motor tasks such as reaching and grasping.

**Acknowledgements**

First and foremost, I would like to thank my supervisor Bryan Tripp for his support throughout my Master's study and this thesis. His perceptive comments helped me to establish the overall direction of the research and to move forward with investigation in depth.

I would also like to thank my mother, brother, and sister for all their love and support. I would not have come this far without them.

Lastly, I would like to thank my friends scattered over the world who proved that true friendship knows no distance.

## Dedication

To the loving memory of my father who taught me determination and courage.

# Table of Contents

# List of Acronyms

**DLPN** Dorsolateral Pontine Nucleus

**FEF** Frontal Eye Fields

**LIF** Leaky Integrate-and-Fire

**MST** Medial Superior Temporal Area

**MT** Middle Temporal Area of the Visual Cortex

**NEF** Neural Engineering Framework

**PYRLK** Pyramidal Lucas and Kanade

**SP** Smooth Pursuit

**SPEM** Smooth Pursuit Eye Movement

**V1** Primary Visual Cortex

# List of Tables

# List of Figures

# Chapter 1

# Introduction

In this chapter, smooth pursuit eye movement and some of its general characteristics are explained first. Afterwards, the motivation behind the work presented in this thesis is described. Finally, an overview of the thesis is provided.

## 1.1 Smooth Pursuit Eye Movement

The smooth pursuit system is highly developed in primates in response to a limitation imposed by the evolution. In primates, only a small portion of the retina, the fovea, is designated for detailed vision. The density of light receptors is much higher in the fovea than peripheral parts of the retina. Also, a relatively large part of the visual cortex processes information from the fovea. The brain would have to be many times larger in order to process information from the whole visual field in the same detail. Smooth pursuit eye movements (SPEMs) stabilize an object of interest on the fovea to maintain detailed visual perception of that object even as it moves across a stationary background.

SPEMs consist of two stages, the first stage is target selection and pursuit initiation, and the second stage is maintenance of ongoing pursuit movements [55].

SPEM is not completely under voluntary control since it is not possible to initiate SPEM across a stationary environment, or completely suppress it in an environment consisting of only moving objects [41].

SPEM has been investigated in two species of primates, humans and macaque monkeys. The performance of the two species is qualitatively similar, although monkeys generally respond more strongly to a given visual stimulus [46]. Humans can execute accurate SPEMs for targets that move up to 30 $^{deg}/_{s}$ [46].

## 1.2 Motivation

The smooth pursuit system is a well-studied subject. The brain mechanisms and circuitry behind it is fairly known and it is simple to study compared to other sensory-motor functions.

Many models have been proposed to explain how smooth pursuit is done in neurological systems. However, these models reflect only calculations that are necessary in the system and do not make any statements whether these calculations are biologically plausible or whether they are carried out in specific brain regions [44]. A few neural models have been proposed to show how neurons can compute required functionalities attributed to SPEMs.

This thesis fills this gap by building a neural model based on one of the best known control models of smooth pursuit. Besides, by creating complementary components, it sets the stage to validate smooth pursuit models in interaction with the real world.

The work which has been presented here can also constitute a basis to build neural

models which are able to drive robots to perform more complex sensory-motor tasks such as reaching and grasping. Additionally, it can give insight into neurological mechanisms of object tracking and provide ideas for creation of more robust object tracking algorithms.

## 1.3 Thesis overview

Chapter 2, Neural Circuitry of Smooth Pursuit Eye Movement, discusses the brain areas that are involved in smooth pursuit eye movement (SPEM) and their functions as related to SPEM.

Chapter 3, Review of Past Smooth Pursuit Models, reviews three major control theory models of SPEM and explains their structure. The control model that was the base for the novel neural smooth pursuit controller is also described.

Chapter 4, Optical Flow: A Key to Understanding Motion Perception, describes the notion of optical flow and how it can be used to provide the input for the smooth pursuit controller. Besides, two different algorithms for optical flow estimations are explained.

Chapter 5, Large-Scale Neural Modelling with the NEF, describes the Neural Engineering Framework (NEF) which is a mathematical framework that allows information to be represented and transformed in populations of spiking neurons.

Chapter 6, A New Model of SPEM to Interact with the Real World, presents the novel work of this thesis. All parts of the new model are explained, including the smooth pursuit neural controller which was built based on the NEF, and integration of the model with a servo-mounted camera. Simulations and experimental results of the model are also provided.

Chapter 7, Discussion and Future Work, summarizes the main contributions and findings of this thesis along with the directions for future work.

# Chapter 2

# Neural Circuitry of Smooth Pursuit Eye Movement

This chapter discusses the brain areas which are commonly accepted to be involved in smooth pursuit eye movement (SPEM). It includes a brief overview of their structures, connectivities, and functions to the extent related to SPEM.

Figure 2.1 depicts the brain areas that constitute the neural circuit of the SPEM.

## 2.1 Neurons

The principal cellular units of the brain that underlie its functionality are called neurons. There are many different kinds of neurons, which differ in terms of the shapes of their dendritic trees, and the chemicals they release. For example, there are at least nine major types of inhibitory neurons in the cerebral cortex [49]. Despite the anatomical variation,

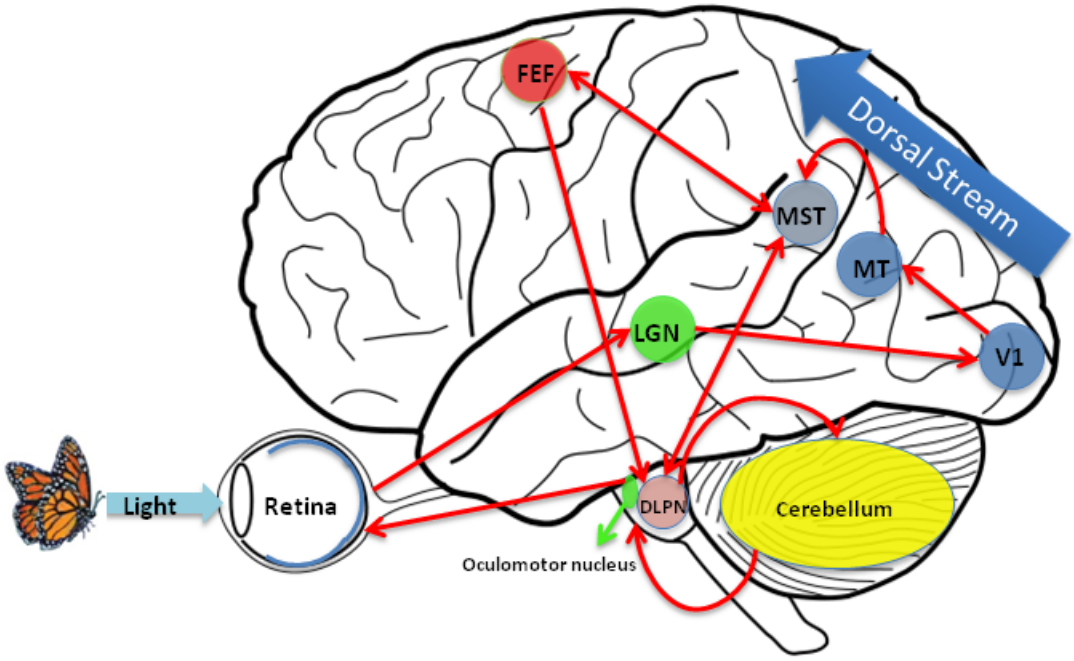



Figure 2.1: Illustration of neural circuitry of smooth pursuit eye movement. LGN = lateral geniculate nucleus; V1 = primary visual cortex; MT = middle temporal area of the visual cortex; MST = medial superior temporal area; FEF = frontal eye fields; DLPN = dorsolateral pontine nucleus.

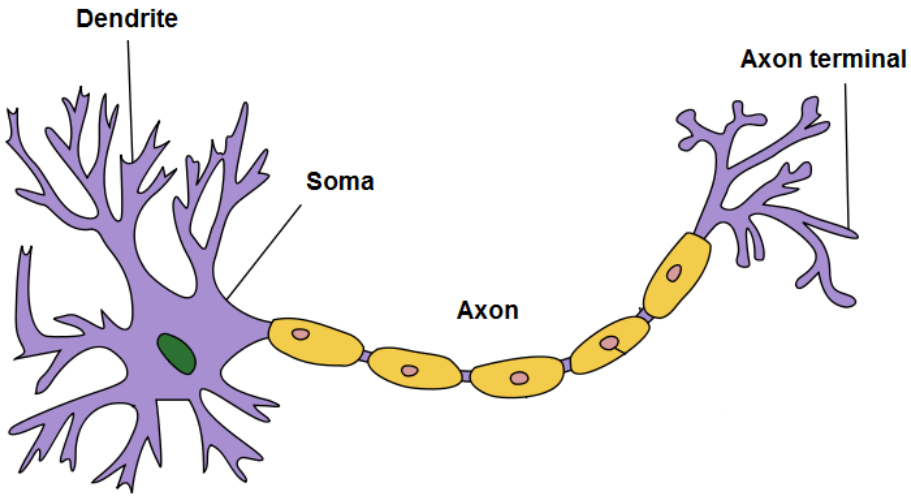



Figure 2.2: Structure of a typical neuron, from [36].

neurons are common in general morphology and being responsiveness to electrochemical signals.

Figure 2.2 depicts the standard neuron morphology. A typical neuron consists of dendrites, a soma, and an axon.

Dendrites are tree-like branching structures that form the input pole of a neuron. Dendrites gather information (in form of electrical spikes) and relay it to the neuron's soma.

The soma or the cell body of a neuron serves to maintain the cell and keep it functional. Production of neurotransmitters (i.e., chemicals that transmit information between neurons through the connections called synapses) happens in the soma.

The axon is a long projection of a neuron that carries information (i.e., electrical spikes) away from its soma to the other neurons, with which it makes synapses. Some axons also directly stimulate muscle or gland cells.

## 2.2 Retina and Fovea

The retina is a thin layer of neural tissue that constitutes the back of the eye. It is considered a part of the brain and consists of photoreceptors and four main classes of neurons [38].

Two basic classes of photoreceptor cells exist in the retina, rods and cones. Rods are more light-sensitive than cones, but they are not sensitive to color. Cones on the other hand, function in bright light and are responsible for color vision. They are much more concentrated in a small central region (i.e., on the visual axis of the eye) of the retina called fovea.

The fovea is a .3mm diameter (i.e., one percent of the retinal surface) rod-free area specialized for high-acuity vision (figure 2.3). Approximately, half of the nerve fibers in the optic nerve carry information from the fovea, while the other half carry information from the rest of the retina [10].

Voluntary eye movements (i.e., saccades and smooth pursuit eye movements) place the image of objects of interest on the fovea to provide clear visual perception [79].

## 2.3 Primary Visual Cortex (V1)

Visual information generated in the retina passes though the lateral geniculate nucleus (LGN) via optic nerve and then comes directly to the primary visual cortex (V1). V1 is one of the brain areas in the visual cortex. The visual cortex itself is a part of the cerebral cortex (outermost layered structure of the brain) responsible for processing visual information. Area V1 has retinotopic organization, meaning that it contains a complete

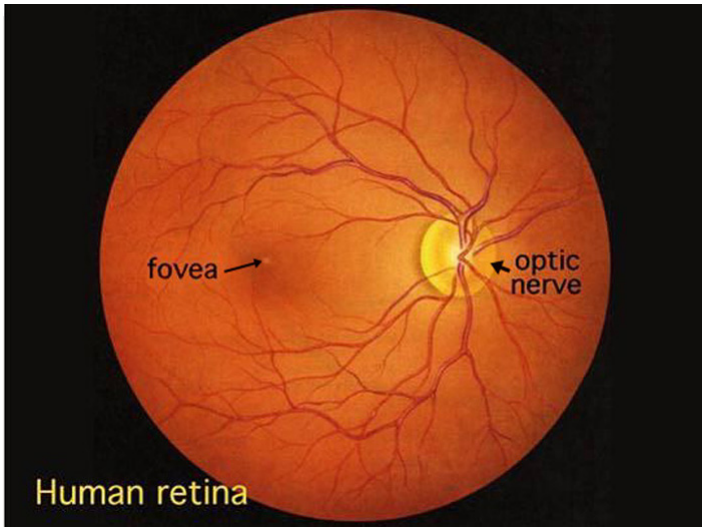



Figure 2.3: Human retina, from [40]. The fovea is a small depression in the retina with high visual acuity.

map of the visual field covered by the two eyes. About fifty percent of the area of human V1 is devoted to the central two percent of the visual field [83]. V1 neurons are classically divided into two categories based on the structure of their receptive fields: simple and complex (Hubel and Wiesel [34, 35]).

The receptive field of a visual neuron comprises a two-dimensional region in the visual space in which the presence of a stimulus alters the activity of that neuron. Receptive field sizes increase at successive processing stages in the visual pathway and, at each processing stage, they increase with the distance from the point of fixation. Visual receptive field sizes can range from a few minutes of arc (a dot in this page at reading distance) to tens of degrees (the entire page) [2].

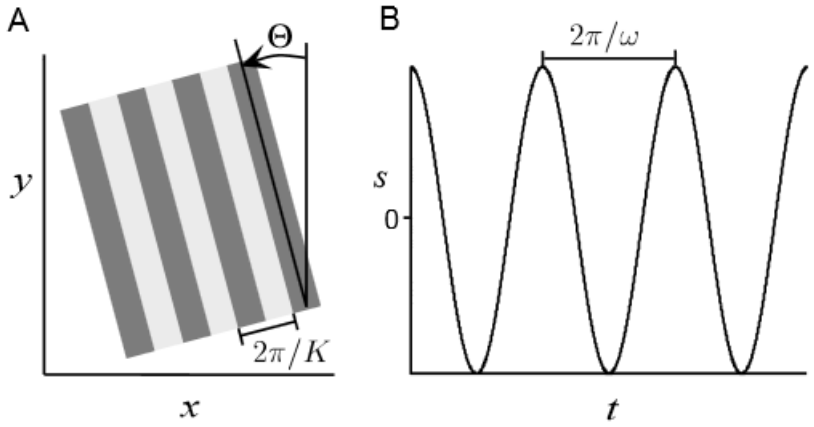



Figure 2.4: A counterphase grating, from [12]. A) A portion of a square-wave grating analogous to the sinusoidal grating of (2.1). The lighter stripes are regions where $s > 0$, and $s < 0$ within the darker stripes. $K$ determines the wavelength of the grating and $\Theta$ its orientation. Changing its spatial phase, $\Phi$, shifts the entire light-dark pattern in the direction perpendicular to the stripes. B) The light-dark intensity at any point of the spatial grating oscillates sinusoidally in time with period $2\pi/\omega$ [12].

### 2.3.1 Simple Cells

One of the commonly used stimuli during recordings from visual neurons, is the counterphase sinusoidal grating that can be mathematically describe as:

$$s(x, y, t) = Acos(Kxcos\Theta + Kysin\Theta - \Phi)cos(\omega t), \tag{2.1}$$

where $K$ and $\omega$ are the spatial and temporal frequencies of the grating, $\Theta$ is its orientation, $\Phi$ is its spatial phase, and $A$ is its contrast amplitude. Figure 2.4 depicts a counterphase grating.

In the early sixties, Hubel and Wiesel [34, 35] used such gratings to examine the re-

sponses of neurons in V1. They discovered that some cells in V1 are selective to the orientation, spatial frequency, and spatial phase of the visual gratings. They called these neurons "simple cells" (in comparison to the complex cells, section 2.3.2).

The spatial receptive filed of simple cells can be mathematically approximated with a Gabor function:

$$\begin{aligned} D_s(x,y) &= \frac{1}{2\pi\sigma_x\sigma_y}\exp(-\frac{x_j^2}{2\sigma_x^2}-\frac{y_j^2}{2\sigma_y^2})cos(kx_j-\phi) \\ x_j &= xcos(\theta)-ysin(\theta) \\ y_j &= xsin(\theta)+ycos(\theta), \end{aligned} \tag{2.2}$$

where $\sigma_x$ and $\sigma_y$ determine the size of the receptive field, $k$ is the preferred spatial frequency, $\phi$ is the preferred spatial phase shift, and $\theta$ is the preferred orientation [12] (see section 2.5).

Simple cells respond strongly to motion of an edge at a certain velocity.

### 2.3.2 Complex Cells

While examining the responses of V1 cells to drifting gratings, Hubel and Wiesel [34, 35] also discovered that some cells in V1 are selective to the spatial frequency and orientation of gratings, but their responses are independent of the spatial phase. They named these neurons "complex cells". Complex cells receive input from many simple cells (see figure 2.6).

Complex cells respond strongly to edge motion at a certain preferred velocity, invariant to position within the receptive field.

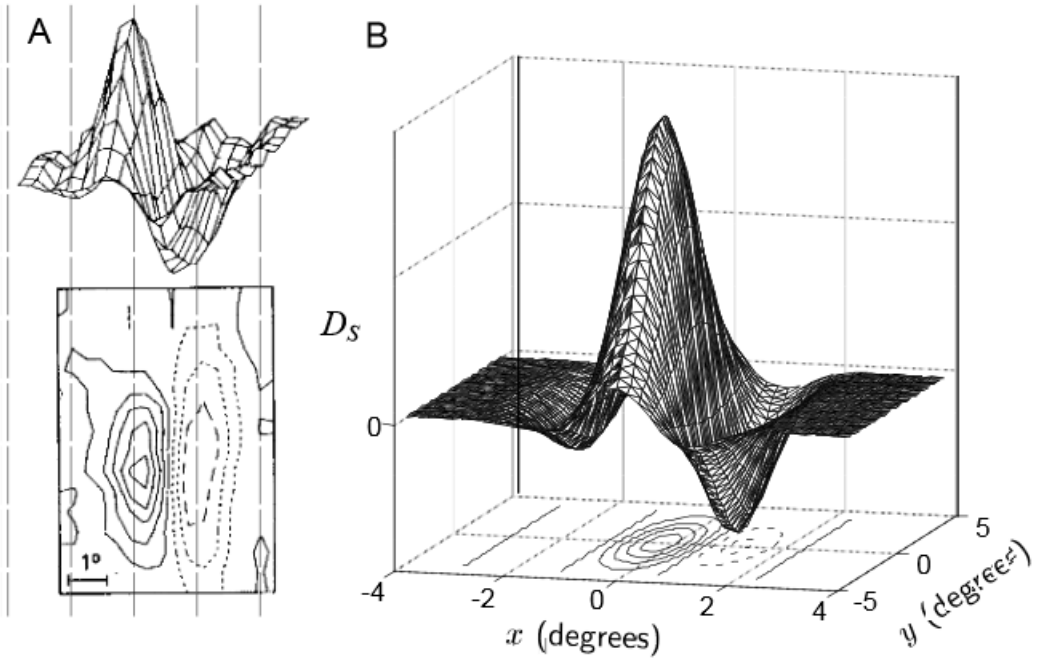



Figure 2.5: Spatial receptive field structure of a simple cell, from [12]. A) Spatial structure of the receptive field of a simple cell in cat primary visual cortex. The upper plot is a three-dimensional representation with the horizontal dimensions acting as the x-y plane and the vertical dimension indicating the magnitude and sign of $D_s(x, y)$. The lower contour plots represent the x-y plane. Regions with solid contour curves show where $D_s(x, y) > 0$ (ON areas), and regions with dashed contours show where $D_s(x, y) < 0$ (OFF areas). B) A Gabor function of the form of equation 2.2 with $\sigma_x = 1°$, $\sigma_y = 2°$, $1/k = .56°$, and $\phi = 1 - \pi/2$ chosen to match A [12].

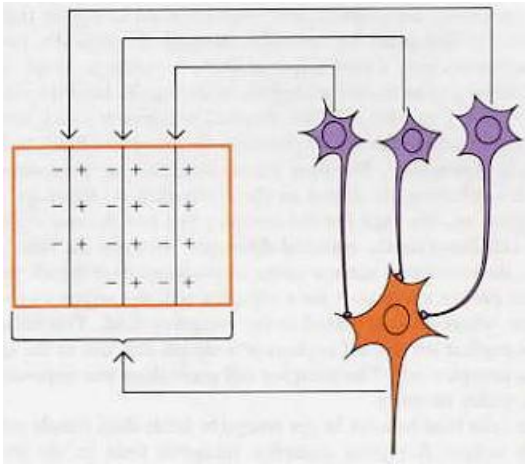


Figure 2.6: Wiring diagram that accounts for the properties of a complex cell, from [33]. A large number of simple cells (only three are shown here) make excitatory synapses with a single complex cell. In this example, each simple cell responds optimally to a vertically oriented edge with light to the right, and the receptive fields are scattered in overlapping fashion throughout the rectangle. An edge falling anywhere within the rectangle evokes a response from a few simple cells, and this in turn evokes a response in the complex cell [33].

### 2.3.3 End-stopped (Hypercomplex) Cells

In 1965, Hubel and Wiesel discovered that in addition to simple and complex cells another type of visual neuron exists in V1 [24]. They called this type of neurons "hypercomplex" or "end-stopped" cells.

An ordinary complex cell usually shows length summation: longer stimulus line evokes larger response in the neuron, until the line is as long as the neuron's receptive field; making the line longer has no effect (as shown in figure 2.7.a). On the other hand, for an end-stopped cell, lengthening the line improves the response up to some limit, but exceeding that limit in one or both directions results in a weaker response (as shown in figure 2.7.b). The total receptive field of an end-stopped cell is made up of an activating region and an inhibitory region, or regions at the ends [33].

It is thought that the end-stopped cells receive excitatory input from cells with small receptive fields and inhibitory input from cells with large receptive fields. The cells supplying inhibition are maximally excited by long slits but poorly excited by short ones (see figure 2.8). End-stopped cells are thought to play a major role in solving the aperture problem in the visual system by their selectivity to the curvature and corners (see section 2.3.4).

### 2.3.4 Aperture Problem

If the aperture (receptive field) of a motion detector (visual neuron) is much smaller than the contour that it observes, the motion detector can only be sensitive to the component of the contour's motion that is perpendicular to the edge of the contour, but it is completely blind to any motion parallel to the contour (apertures 1 and 3 in figure 2.9). This is

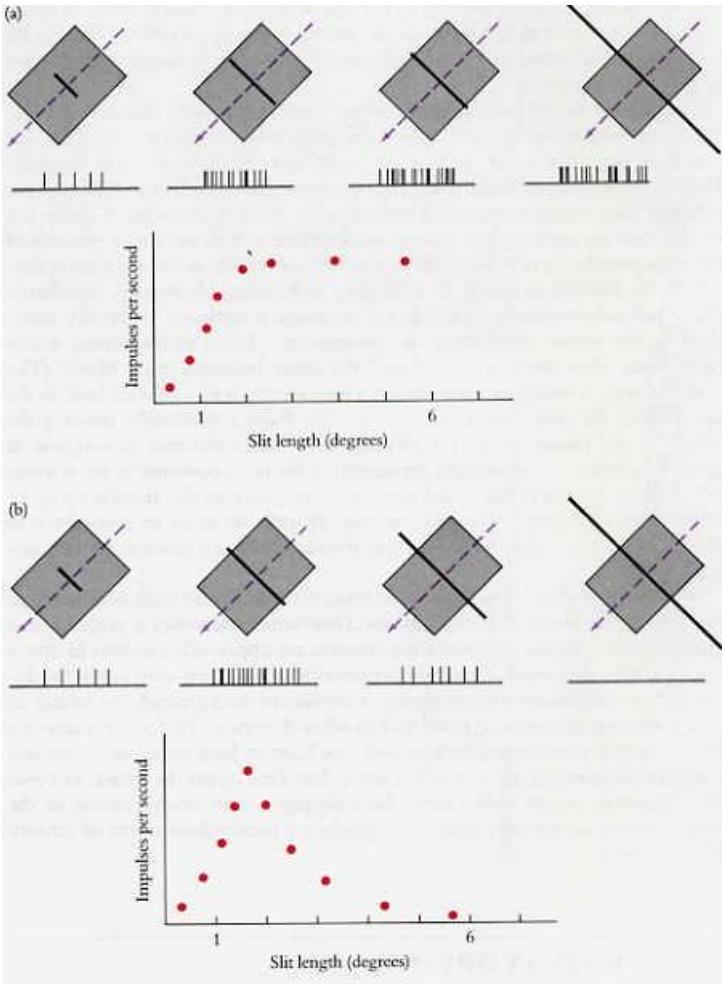



Figure 2.7: Comparison between the responses of a complex cell and an end-stopped cell to various lengths of a slit of light, from [33]. a: An ordinary complex cell response. b: end-stopped cell response.

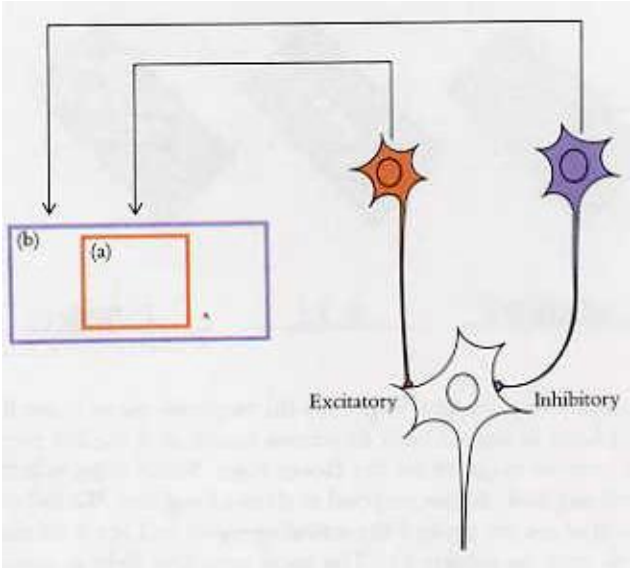



Figure 2.8: Wiring diagram that accounts for the properties of an end-stopped cell, from [33]. See section 2.3.3 for explanation.

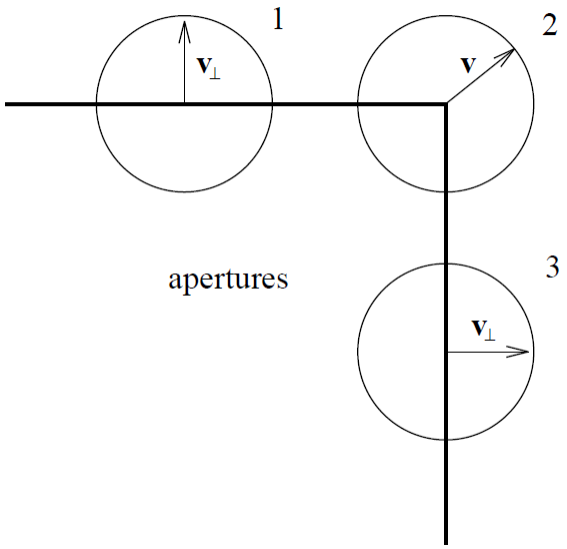



Figure 2.9: Illustration of aperture problem, from [3]. A square is moving up and right. Through apertures 1 and 3 only normal (i.e perpendicular) motions of the edges forming the square can be estimated due to a lack of local structure. From aperture 2, which resides at the corner point, the motion can be fully measured since there is sufficient local structure (i.e., both normal motions are visible) [3].

because movement in this direction will not change the appearance of anything within the aperture. As a result, the motion detector is unable to detect the true movement of the contour [31]. Figure 2.9 illustrates the aperture problem.

## 2.4 Dorsal Visual Stream

The two-streams hypothesis (which was proposed by Ungerleider and Mishkin [82] and has become popular since Goodale and Milner's paper in 1992 [28]) is a widely accepted

influential model of the neural processing of vision [23]. It argues that as visual information exits the occipital lobe (i.e., the visual processing center of the brain), it follows two main pathways or streams: the ventral stream (also known as the "what pathway") and the dorsal stream (also known as the "how pathway" or "where pathway").

The ventral stream is involved with object identification and recognition, while the dorsal stream is involved in finding where objects are in space and providing information needed to interact with them.

The focus of this research, like the rest of SPEM literature, has been on the dorsal stream since tracking is possible by finding "where in space". However, the ventral stream is also involved.

### 2.4.1 Middle Temporal Visual Area (MT)

The middle temporal visual area (MT) is an important motion-sensitive area in the dorsal stream.

Recordings from area MT have shown that a large portion of its cells are tuned to the speed and direction of moving visual stimuli [18, 51]. Neurons in MT have larger receptive fields compared to V1 neurons, and aperture problem is solved in MT [56].

MT has a topographic organization so that different parts of the visual field are represented in an orderly way across the full extent of MT [51].

Lesion studies in monkeys have confirmed the role of MT in motion perception and eye movements [45]. These studies have shown that if a target moves across the part of the visual field related to a lesion in MT, impaired monkeys are not able to initiate SPEM.

### 2.4.2 Medial Superior Temporal (MST) Cortex

MT neurons supply afferents to a brain area called medial superior temporal (MST) cortex [81]. MST neurons have large receptive fields compared to MT neurons and respond selectively to optical flow components (see chapter 4), such as expansion, contraction, and clockwise or counterclockwise rotation [71, 70, 69, 19, 54, 43].

Microstimulation within MST influences the velocity of SPEMs [55]. Moreover, Lesions of MST create a directional deficit, impairing the animal's ability to execute SPEM when the target moves toward the lesioned hemisphere, irrespective of position in the visual field [55].

MST consists of dorsal and lateral parts, MSTd and MSTv. Mustari et al. [52] have shown that MSTd neurons are sensitive to eye velocity and retinal slip velocity during SPEM (see figure 2.10).

## 2.5 Frontal Eye Fields (FEF)

The frontal eye fields (FEF) is a region located in frontal cortex. The FEF appears to play a significant role in planning and execution of saccades (i.e., fast eye movements) [73]. It also participates in the control of visual selective attention [72].

Neurons in this area have connections with different brain areas in the visual cortex including MST [25].

There is a part in the FEF where neurons respond selectively only during SPEM but not other kinds of eye movements. This part is called smooth pursuit frontal eye fields ($\text{FEF}_{\text{SPEM}}$) and may have a role in controlling the gain of the SPEM [77, 25]. Recordings

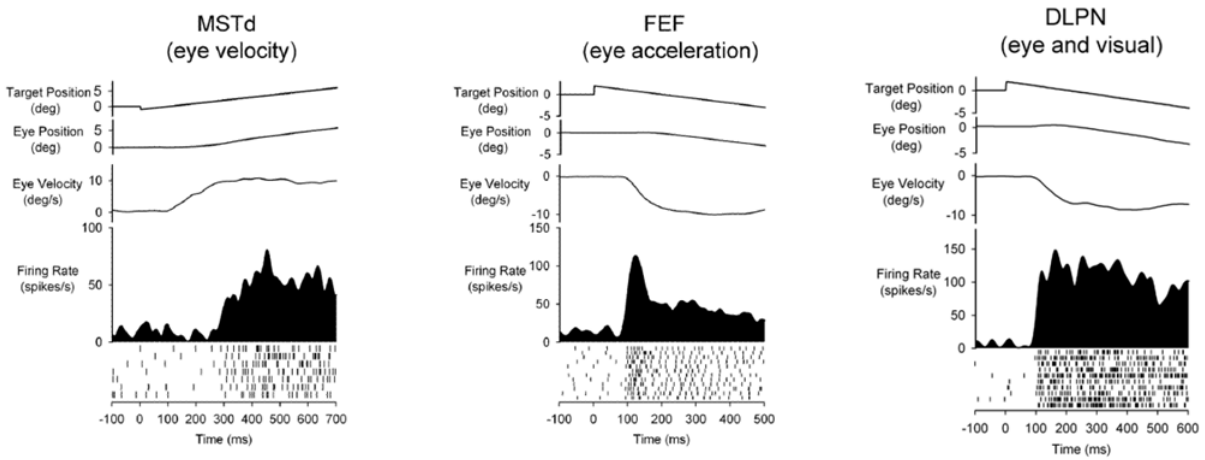



Figure 2.10: Examples of SPEM related neurons in MSTd, FEF, and DLPN, from [52]. MSTd = dorsal part of medial superior temporal area; FEF = frontal eye fields; DLPN = dorsolateral pontine nucleus. MSTd and DLPN neurons are sensitive to eye velocity, while FEF neurons are sensitive to eye acceleration.

from neurons in $\text{FEF}_{\text{SPEM}}$ have shown that they are best responsive to the eye acceleration during SPEM [52] (see figure 2.10).

## 2.6 Brianstem

Several regions of the brainstem seem to be involved in SPEM. This section explains the areas with major roles in SPEM.

### 2.6.1 Nuclei Prepositus Hypoglossi (NPH), Rostral medial Vestibular Nucleus (VN), and Interstitial Nucleus of Cajal (ICC)

The nuclei prepositus hypoglossi (NPH) and rostral medial vestibular nucleus (VN) participate in gaze control by integrating a velocity input to determine a desired eye displacement. More specifically, neurons in these areas are responsible for representing and controlling horizontal eye position [22, 67]. Moreover, interstitial nucleus of Cajal (ICC) controls the vertical eye position [9].

### 2.6.2 Dorsolateral Pontine Nucleus (DLPN) and Nucleus Reticularis Tegmenti Pontis (NRTP)

Neurons in dorsolateral pontine nucleus (DLPN) and nucleus reticularis tegmenti pontis (NRTP) receive their SPEM related inputs mostly from MST and $\text{FEF}_{\text{SPEM}}$ [52]. DLPN neurons are mostly responsive to eye velocity during SPEM [52] (see figure 2.10).

## 2.7 Cerebellum and Eye Motoneurones

Neurons in the cerebellum receive motor commands for SPEM, and send them to the oculomotor nuclei where eye motoneurons generate the appropriate levels of muscular force in extraocular (i.e., eye) muscles so that eyes can track the object of interest.

While SPEMs are initiated by visual stimuli, the ability to maintain eye velocity in the brief absence of image motion supports the existence of a non-visual (i.e., extra-retinal) input to the SPEM system. This extra-retinal input is an efference copy of eye velocity.

Lisberger has suggested that the cerebellum plays the primary role in providing this input for the SPEM system [45].

# Chapter 3

# Review of Past Smooth Pursuit Models

Studying smooth pursuit eye movements (SPEMs) gives us insight into sensation, sensory-motor transformations, and motor control [45]. To understand smooth pursuit system, creation of models has been an important step. A large number of SPEM models, based on control theory, have been built to make quantitative predictions that can be tested in experiments. The main focus of this chapter is to discuss major control theory models of SPEM.

## 3.1 Smooth Pursuit as a Control System

A challenge in studying human motor control is to understand how biological systems cope with the delay between the sensory detection of a signal and the execution of a motor action as a response [63]. This delay occurs due to the information transmission time between

different brain areas involved in the motor task and the processing time to plan appropriate motor action. Long delays may render a control system unstable and cause oscillations [4]. Therefore, the brain should have mechanisms to compensate for the effect of delay to successfully accomplish motor control tasks such as tracking objects and grasping.

Because eye movements are simpler than other movements in many ways, the smooth pursuit system provides an ideal opportunity for investigation of the brain mechanisms of control [46]. The mechanics of the eyeball are straightforward [65], and the eyeball is not subject to sudden changes in load [46]. Moreover, the study of the eye movements is not impeded by the problems of complex kinematics and dynamics that complicate the study of limb motion [46].

The smooth pursuit can be considered a control system whose input is target velocity, output is eye velocity, and goal is to make the output follow the input as closely and fast as possible [60, 66, 46]. Figure 3.1 shows how one can think of smooth pursuit as a control system. The only input to the smooth pursuit (SP) controller in this schematic diagram is retinal slip velocity of the target. However, due to the transmission and processing delays, this system is unstable, suggesting that there should be other inputs to the SP controller.

Different models have been proposed for the SP controller, each adopting different strategies to overcome instability due to delays. Three key SP models [44, 8] are: the target velocity model of Robinson et al. [64], the tachometer model of Ringach [63], and the image motion model of Krauzlis and Lisberger [42].

In the following sections, these three models are explained in addition to a more recent work in which modified versions of the target velocity model and image motion model have been implemented and compared to experimental data.

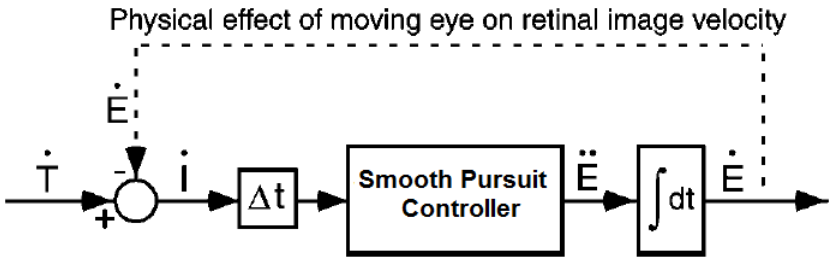



Figure 3.1: Smooth pursuit as a control system, from [8]. Target velocity, $\dot{T}$, and eye velocity, $\dot{E}$, are compared at the retina to yield retinal slip velocity of the target, $\dot{I}$. Retinal slip velocity is delayed, $\Delta t$, and processed by the smooth pursuit controller to produce a command for eye acceleration, $\ddot{E}$. The eye acceleration command is integrated to produce eye velocity, $\dot{E}$, which is treated as the output of the pursuit system. Solid arrows show the flow of neural signals, while the dashed arrow shows physical negative feedback due to the fact that the retina is attached to the moving eyeball [8]. Due to existence of delay this retinal slip velocity alone is not enough to render the system stable.

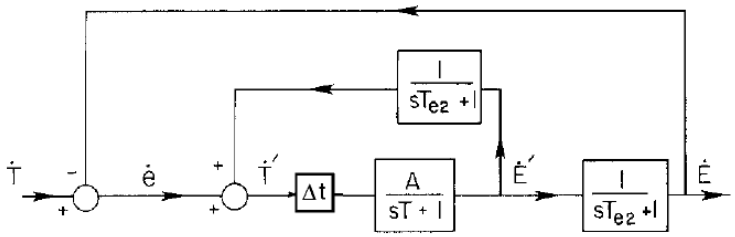



Figure 3.2: Model of smooth pursuit controller by Robinson, from [64]. In this model, a positive feedback from eye velocity command adds to retinal slip velocity (see section 3.2).

## 3.2 Target Velocity Model of Robinson et al.

The solution Robinson et al. suggested to cope with delay in SPEM models was to add an efference copy of the velocity command, which goes to the eye as input, to the smooth pursuit controller [64]. Based on their model, the visual pathways driving pursuit are sensitive only to retinal slip velocity.

Figure 3.2 shows the block diagram of the target velocity model. Note that by adding the efference copy of eye velocity, which ideally reconstructs target velocity, a gain close to unity is required [64].

## 3.3 Image Motion Model of Krauzlis and Lisberger

The basic idea of the image motion model, suggested by Krauzlis and Lisberger [42], is to utilize delayed retinal slip acceleration to predict the present value of retinal slip velocity by linear extrapolation:

$$\begin{aligned} \dot{I}(t) &\approx \dot{I}(t-\Delta t) + \frac{d\dot{I}(t-\Delta t)}{dt}\Delta t \\ &\approx \dot{I}(t-\Delta t) + \ddot{I}(t-\Delta t)\Delta t \end{aligned} \tag{3.1}$$

where $\dot{I}(t)$ is current retinal slip velocity, $\dot{I}(t-\Delta t)$ is delayed retinal slip velocity, and $\ddot{I}(t-\Delta t)$ is delayed retinal slip acceleration.

Figure 3.3 shows the schematic of the image motion model.

## 3.4 Tachometer Model of Ringach

In 1992, Goldreich et al. designed an experiment in which they artificially changed the visual feedback delay during trials of recording SPEMs in monkeys [27]. They compared the result of that experiment with simulation results of both the target velocity model and the image motion model. The comparison showed that the target velocity model cannot account for the changes in spontaneous oscillation frequency produced by artificially altering visual feedback delay [27]. Three years later, Ringach suggested a model that used an efference copy of eye velocity (like the target velocity model) and was also compatible with experimental data.

In Ringach's tachometer model, a delayed eye acceleration signal is used to predict the current value of retinal slip velocity with the assumption that target velocity is constant or varying slowly in time compared with the changes in eye velocity:

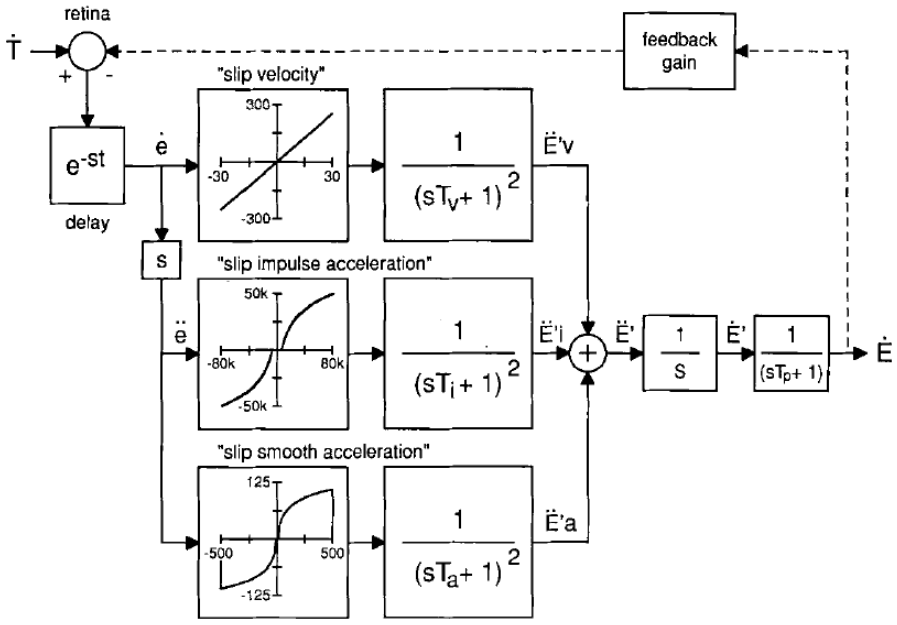



Figure 3.3: Model of smooth pursuit controller by Krauzlis and Lisberger, from [42]. Krauzlis and Lisberger designed three pathways in this model. The first pathway is sensitive to slip velocity and its gain is linear. The second and third pathways are both sensitive to slip acceleration but in different ways. The impulse acceleration pathway is sensitive to the large accelerations that accompany step changes in target velocity, but has a dead zone in the gain element that makes it insensitive to smaller accelerations. The smooth acceleration pathway is sensitive to gradual changes in image velocity. The outputs of the gain elements in each pathway are low-pass filtered to produce three signals with different dynamics ($\ddot{E}'_v$, $\ddot{E}'_i$, and $\ddot{E}'_a$) that are then summed and integrated to give a command for eye velocity ($\dot{E}'$). Eye velocity ($\dot{E}$) is obtained by passing the eye velocity command through a low-pass filter that represented the eye muscles and orbital tissues. The non-linearities the in second and third pathways derived from behavioral experiments [42].

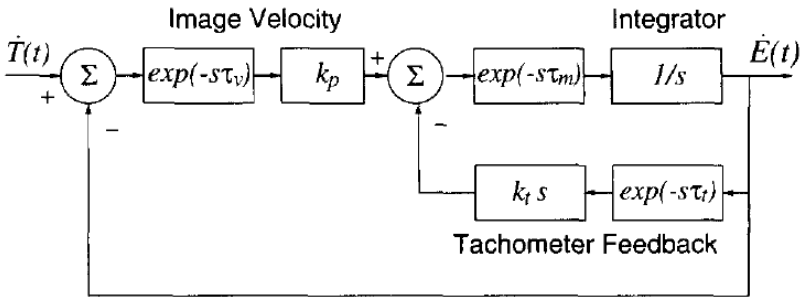



Figure 3.4: Model of smooth pursuit controller by Ringach, from [63]. In this model, a negative "tachometer" feedback from eye velocity command adds to retinal slip velocity.

$$\begin{aligned}\dot{I}(t) &\approx \dot{I}(t-\Delta t) + \frac{d\dot{I}(t-\Delta t)}{dt}\Delta t \\ &\approx \dot{I}(t-\Delta t) + \frac{d(\dot{T}(t-\Delta t) - \dot{E}(t-\Delta t))}{dt}\Delta t \\ &\approx \dot{I}(t-\Delta t) - \ddot{E}(t-\Delta t)\Delta t,\end{aligned} \tag{3.2}$$

where $\dot{I}(t)$ is current retinal slip velocity, $\dot{I}(t-\Delta t)$ is delayed retinal slip velocity, $\dot{T}(t-\Delta t)$ is delayed target velocity, and $\ddot{E}(t-\Delta t)$ is delayed eye acceleration.

Figure 3.4 shows the block diagram of the tachometer model.

## 3.5 Churchland and Lisberger's Models

Churchland and Lisberger implemented modified versions of the image motion model and the tachometer model [8]. They tested their modified models against results of behavioural experiment with Rhesus monkeys. They concluded that for both step and sine wave target

velocities, the modified image motion model could more accurately fit the closed-loop responses of monkeys.

Figure 3.5 depicts the block diagrams of Churchland and Lisberger's models. Note that although their models consist of three pathways, they did not use the motion onset pathway to reproduce responses of all monkeys. In other words, image velocity and image acceleration pathways were enough to successfully fit the data for some monkeys [8].

## 3.6 Neural Models

The models we have seen so far only reflect calculations that are necessary in a smooth pursuit system, and do not make any statements whether these calculations are biologically plausible or if they are carried out in specific brain regions [44].

Only a few neural models have been proposed to show how different brain areas, based on their receptive fields and connectivity to each other, can perform the required functions in SPEMs. For example, the best-known neural model of SPEM, which has been created by Pack, Grossberg, and Mingolla, shows how to build receptive fields for neurons in area MT and then appropriately connect them to four neurons in area MST so they can encode the velocity of the target with their activities [55].

To build a complete functional smooth pursuit model in neurons, I first tried to use Pack and colleagues' model. However, after some simulations, it became clear that the model, which is described by a set of non-linear equations, becomes unstable for some values of target velocity. Furthermore, finding the region of stability of the model is not feasible due to highly non-linear nature of the model. Additionally, this model has not been validated against any experimental data.

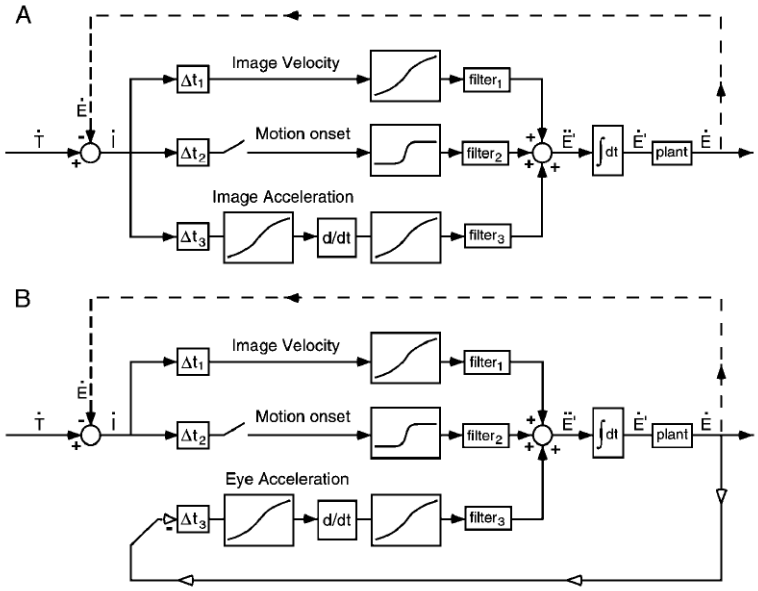



Figure 3.5: Models of smooth pursuit controller by Churchland and Lisberger, from [8]. A) Modified image motion model. B) Modified tachometer model. Inputs arise on the left and outputs are shown on the right. Solid arrows show the flow of signals within the pursuit system. The dashed arrow reflects negative feedback that results because the retina is attached to the moving eye. Circular nodes operate as summing junctions. The input to the both models is retinal image velocity $\dot{I}$, obtained by taking the difference between target velocity, $\dot{T}$ , and eye velocity, $\dot{E}$. Each model contains 3 pathways whose outputs are summed to generate an eye acceleration command. The top 2 pathways are the same for the 2 models and receive an image velocity input. For the image motion model, the bottom pathway also receives an image velocity input. For the tachometer feedback model, the bottom pathway receives an eye velocity input. Each pathway contains a delay, $\Delta t$, a first-order low-pass filter, and one or more non-linear gain elements. The middle pathway also contains a switch so that it is active only for the 1st 30 ms after the target begins to move. The bottom-most pathway adds a second non-linear gain, situated prior to differentiation of the input signal. The outputs of the three pathways are summed to create the net eye acceleration command, $\ddot{E}'$, which is integrated to produce an eye velocity command, $\dot{E}'$. This command then passes through a final low-pass filter, labelled "plant", to yield actual eye velocity, $\dot{E}$ [8].

Due to the limitations of Pack and colleagues' model, I chose the modified image motion model suggested by Churchland and Lisberger (which seems to be the most accurate control model to fit experimental data) as a starting point for a neural model (see chapter 6 for description of the neural model).

# Chapter 4

# Optical Flow: A Key to Understanding Motion Perception

Almost all past models of smooth pursuit eye movement (SPEM) have used an scalar retinal velocity as the input to their models (see chapter 3). However, computing the retinal velocity associated with the moving object is a sophisticated task. A complex process takes place in the brain from the retina to primary visual cortex and later in the dorsal visual stream (see chapter 2) to compute the retinal velocity and perceive motion [38]. This retinal velocity is then fed into the neural circuitry designated to control SPEM in the brain [41].

The purpose of this chapter is to discuss two motion estimation algorithms which were implemented to provide the retinal velocity for the SPEM controller. Towards this goal, first, the notion of optical flow will be introduced. Then, the motion energy model that resembles the motion estimation computations happening in human visual system will be described. Afterwards, the results related to my implementation of this method will be

explained. Finally, a more abstract, yet more accurate mathematical motion estimation method proposed by researchers in the field of computer vision along with its implementation on graphical processing unit (GPU) will be presented.

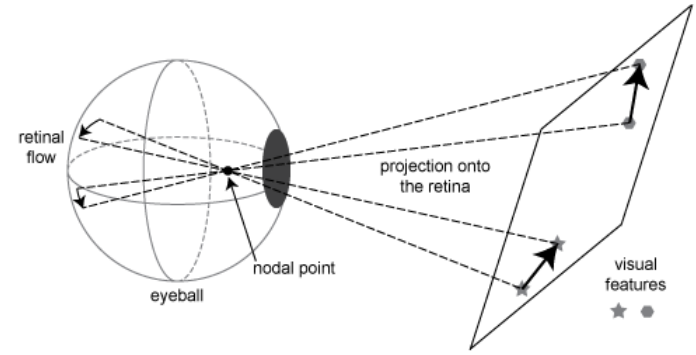
## 4.1 Optical Flow

In the biological context, optical flow (sometimes called retinal velocity) is the change of structured patterns of light on the retina that leads to an impression of movement of the scene projected onto the retina [61].

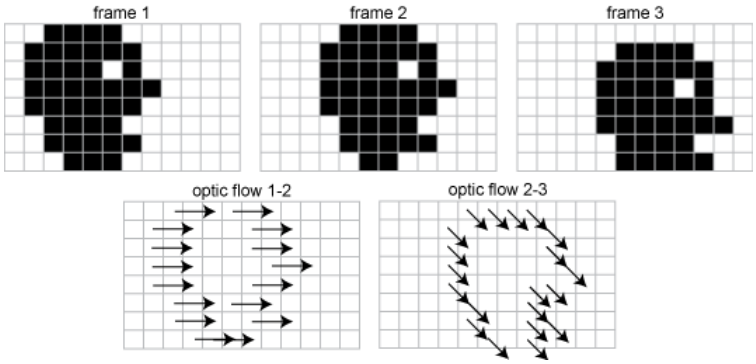
In computer vision, the camera becomes the surrogate eye and changes in the environment are represented by a series of image frames. These frames are obtained by a spatiotemporal sampling of the incoming light that hits the camera's sensor. In this context, optical flow is defined as the vector field that captures the movement of the pixels in the acquired frames over time [3].

Figure 4.1 illustrates the optical flow in both the retina and the image frames. Figure 4.1a shows the movements of two visual features (i.e., star and hexagon) in the environment and their respective optical flow generated on the retina. Figure 4.1b demonstrates three frames illustrating the movement of a head silhouette. The resultant optical flow is depicted as the correspondence between contour pixels in frame 1 and 2, as well as frame 2 and 3.

The first algorithms for the optical flow estimation were proposed in the early eighties ([32] and [48]). Since then, optical flow has found a variety of applications. Object segmentation and tracking [13], video stabilization [57], video compression [84], and depth estimation [74] are some examples. Due to this wide range of applications, researchers are




(a)




(b)

Figure 4.1: Illustration of optical flow in both the retina and the image frames, from [61]. See text for explanation.

still working on new methods for close to real-time, pixel-wise (i.e., dense) estimation of optical flow. SimpleFlow [78], Correlation Flow [17], and Ramp [75] are just some examples of recently proposed optical flow algorithms. These methods are accurate but slow to be used in the smooth pursuit system. There is an online database for evaluation of different optical flow methods (http://vision.middlebury.edu/flow/eval/).

## 4.2 Motion Energy Model for Optical Flow Estimation

Researchers' interest to understand the mechanism of motion perception in the brain has led to bio-inspired models for optical flow (e.g., [62, 1, 30]). The motion energy model, introduced by Adelson and Bergen in 1984, is the best-known bio-inspired model.

### 4.2.1 Extracting Spatiotemporal Energy

Adelson and Bergen showed that motion can be thought of as orientation in space-time. They used spatiotemporal filters to detect this orientation (see figure 4.2). These filters were Gabor functions [26] in the spatial domain followed by causal temporal filters [1]. The idea of using Gabor filters came from biology where the receptive fields of simple cells in primary visual cortex had been approximated with Gabor functions [11] (see section 2.3.1). The mathematical description of a Gabor function is:

$$\begin{aligned} G(x,y) &= \tfrac{1}{2\pi\sigma_x\sigma_y}\exp(-\tfrac{x_j^2}{2\sigma_x^2}-\tfrac{y_j^2}{2\sigma_y^2})cos(2\pi f x_j-\phi) \\ x_j &= xcos(\Theta)-ysin(\Theta) \\ y_j &= xsin(\Theta)+ycos(\Theta), \end{aligned} \tag{4.1}$$

where spatial scales, $\sigma_x$ and $\sigma_y$, spatial frequency, $f$, phase shift, $\phi$, and orientation, $\Theta$, are the Gabor filter parameters [12].

Two Gabor filters with the same spatial scales, spatial frequency, and orientation which are 90 degrees out of phase form a quadrature pair. If the responses of a quadrature pair of motion-selective spatiotemporal filters are squared and summed, the resultant signal gives

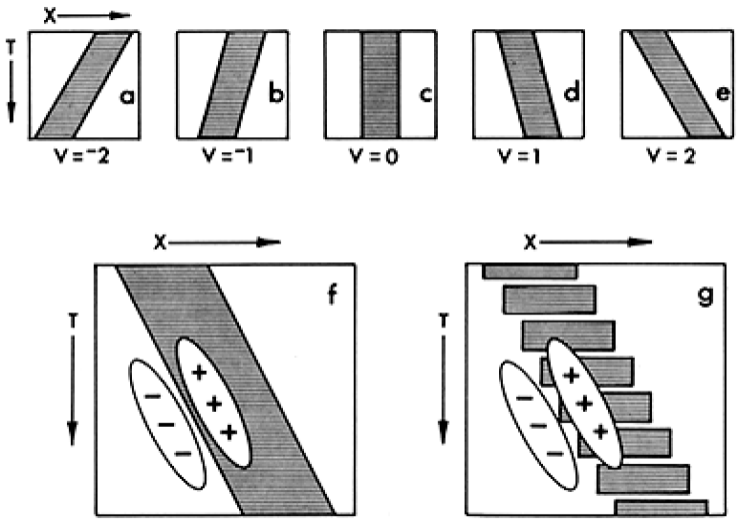



Figure 4.2: Motion as orientation in space-time, from [1]. Top: A moving bar along x-axis with different speeds creates different orientations in space-time. Bottom: A spatiotemporally oriented filter can be used to detect the speed both in continuous (left) and sampled (right) motion.

a phase-independent measure of local motion energy within a given spatial-frequency band (like complex cells [12]).

### 4.2.2 Exploration of the Motion Energy Model as a means of Extracting the Retinal Velocity

As mentioned in the beginning of this chapter, in order to have a SPEM model to work with naturalistic images instead of an abstract velocity, an estimate of optical flow is needed.

Among the different methods of optical flow estimation, the motion energy model seemed to be an obvious candidate since it is a biologically plausible method. However, there were two complications with using this method. First, it was computationally intensive. Second, since each spatiotemporal filter is only sensitive to a small part of the visual scene, it does not solve the aperture problem on its own.

An implementation of the motion energy model on GPU addressed the first issue (see section 4.2.2 for details). However, despite reviewing researchers' proposed ideas to solve the aperture problem (see [30] and [15] for example), it was not clear how the aperture problem should be solved in a fast and biologically plausible manner. Due to time constraints, I focused on searching for another method which could solve the aperture problem and run with close to real-time speed.

#### Details of Motion Energy GPU Implementation

A bank of sptaiotemporal filters was created. The filters were Gabor spatial filters with 2nd-order low-pass temporal dynamics and frame-wise shifts at different velocities. Spatial

frequencies, $f$, and scales of the filters, $\sigma_x$ and $\sigma_y$, were chosen based on [7] to reasonably cover the spatial frequency spectrum.

Several CUDA (http://www.nvidia.ca/object/cuda_home_new.html) kernels were developed to run parallelizable parts of the code on GPU to accelerate run-time.

For a filter bank that consisted of 56 filters (4 different frequencies, 7 different velocities, and 2 different orientations ($\Theta = 0°$ and $\Theta = 90°$)) and the $640 \times 480$ frame resolution, the speed of 20 frames/s was achieved on an *NVIDIA GeForce GTX 560* GPU. Therefore, it is possible to run the motion energy model at practical framerates. However, this approach was not pursued further because of the remaining complexity of solving the aperture problem.

## 4.3 Pyramidal Lucas and Kanade for Optical Flow Estimation

Since both time-efficiency and accuracy were important for the SPEM model, a fast method that could solve the aperture problem was needed. One relatively simple and fast method that solves the aperture problem is the pyramidal method of Lucas and Kanade. A recent GPU implementation of this method was adopted that runs in real time [50].

This section explains both classic and pyramidal Lucas and Kanade algorithms. Moreover, the details of the adopted GPU implementation are mentioned.

### 4.3.1 Classic Lucas and Kanade algorithm

To explain the pyramidal Lucas and Kanade method, the classic Lucas and Kanade algorithm should be discussed first. For this purpose, consider if the sampling time between the image frames is small enough (i.e., high frame rate is provided), it is reasonable to assume that intensity of a visual feature remains approximately constant as it moves from one frame to the consequent one:

$$I(x, y, t) = I(x + dx, y + dy, t + dt). \tag{4.2}$$

By using first order Taylor approximation (4.2) becomes:

$$I(x, y, t) = I(x, y, t) + \frac{\delta I}{\delta x}dx + \frac{\delta I}{\delta y}dy + \frac{\delta I}{\delta t}dt. \tag{4.3}$$

The first term on the right hand side of (4.3) cancels the term on the left hand side:

$$I_x dx + I_y dy + I_t dt = 0, \tag{4.4}$$

where $I_x = \frac{\delta I}{\delta x}$, $I_y = \frac{\delta I}{\delta y}$, and $I_t = \frac{\delta I}{\delta t}$.

Dividing both sides of (4.4) by $dt$ and assuming $u = \frac{dx}{dt}$ and $v = \frac{dy}{dt}$, gives us the optical flow constraint equation:

$$I_x u + I_y v + I_t = 0, \tag{4.5}$$

where $u$, and $v$, are horizontal and vertical components of the optical flow vector associated with the considered pixel. Note that this equation has two unknowns and cannot

be solved uniquely. This ambiguity in fact is a demonstration of the aperture problem discussed in section 2.3.4.

To solve this ambiguity, Lucas and Kanade [48] assumed that the flow is essentially constant in a patch of n pixels centred on the pixel under consideration:

$$\begin{aligned} I_{x_1}u + I_{y_1}v &= -I_{t_1} \\ I_{x_2}u + I_{y_2}v &= -I_{t_2} \\ &\vdots \\ I_{x_n}u + I_{y_n}v &= -I_{t_n}. \end{aligned} \tag{4.6}$$

Writing (4.6) in the matrix form gives:

$$A \begin{bmatrix} u \\ v \end{bmatrix} = b, \tag{4.7}$$

where:

$$A = \begin{bmatrix} I_{x_1} & I_{y_1} \\ I_{x_2} & I_{y_2} \\ \vdots & \vdots \\ I_{x_n} & I_{y_n} \end{bmatrix}, b = \begin{bmatrix} -I_{t_1} \\ -I_{t_2} \\ \vdots \\ -I_{t_n}. \end{bmatrix} \tag{4.8}$$

To solve (4.7) for $u$, and $v$, Lucas and Kanade used least squares method that leads to:

$$\begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b. \tag{4.9}$$

Since $A^T A$ can be a singular matrix, using a regularized least-square method will improve the robustness of the solution [50]. This yields:

$$\begin{bmatrix} u \\ v \end{bmatrix} = (A^T A + \alpha I)^{-1} A^T b, \tag{4.10}$$

where $0 < \alpha < 10^{-3}$ and $I$ is the identity matrix [50].

The computations related to the classic Lucas and Kanade method stop at this point. However, there is a problem: the classic Lucas and Kanade fails to capture large displacements. To tackle this issue, the notion of pyramids can be applied to build coarser versions of the image frames, so large displacements will be detected in coarser copies.

### 4.3.2 Pyramidal Lucas and Kanade Algorithm

Now that the classic Lucas and Kanade method in addition to benefits of applying pyramids are explained, the complete pyramidal Lucas and Kanade algorithm [50] to estimate optical flow between two frames can be described as follows (see figure 4.3 for illustration):

Building a Gaussian pyramid with $n$ levels:

1. The level 0 of the pyramid is filled with the original image.
2. For levels $i = 1$ to $n - 1$:

   The level $i$ is built with the image of level $i - 1$ sub-sampled by a factor of two.

Optical flow calculations:

1. The optical flow is computed at level $n - 1$ (i.e., the lowest resolution) based on (4.10).
2. For levels $i = n - 2$ to 0:

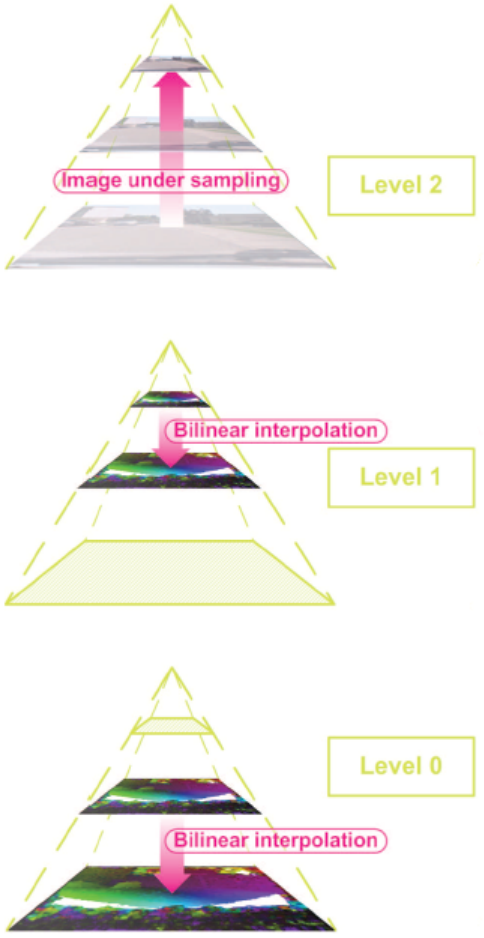



Figure 4.3: Illustration of the pyramidal Lucas and Kanade optical flow method with 3 levels, from [50]. See the text for explanation.

(a) The initial value is equal to twice the over-sampled optical flow computed at level $i-1$ using bilinear interpolation.

(b) The initial value calculated in 2.$a$ is used to warp the second frame.

(c) Optical flow between the first frame and the warped version of the second frame is computed based on equation (4.10).

(d) Final optical flow value is the summation of the initial value from 2.$a$ and the value calculated in 2.$c$.

Note that the number of levels in the pyramid should be chosen with respect to the original frame resolution (e.g., 3 or 4 levels for $640 \times 480$). Furthermore, since the the pixel computations in each level are highly parallelizable, a parallel implementation on the GPU results in a faster run-time (compared to the CPU implementation) [50].

Also note that both classic and pyramidal Lucas and Kanade methods can be implemented in an iterative manner to improve the accuracy of the optical flow estimation. However, my experiments indicated that adding iterations does not lead to a noticeable improvement in accuracy while it made the code to run much slower. Therefore, the algorithm which was used (explained above) did not include iterations.

### 4.3.3 Implementation Details

To conclude this chapter, it should be mentioned that for optical flow calculations used in the SPEM model, a CUDA kernel (based on [50]) has been written to be called via MATLAB. For the $640 \times 480$ resolution with 4 levels, the speed of 25 frames/s has been achieved on an *NVIDIA GeForce GTX 560* GPU.

# Chapter 5

# Large-Scale Neural Modelling with the NEF

The Neural Engineering Framework (NEF) developed by Eliasmith and Anderson [22] employs common tools in engineering such as signal processing, information theory, and control theory to quantitatively characterise and model the functionality of neural systems. It provides a framework to encode information and then implement fairly complex functions or algorithms using populations of spiking neurons.

The NEF consists of three principles: representation, transformation, and dynamics. This chapter describes these three principles after giving a general description of the neuron models followed by a rather detailed explanation of the leaky integrate-fire neuron model. The material of this chapter is adapted from [22].

## 5.1 Single Neuron Model

Neuron models are sets of mathematical equations describing how the neuron's input (i.e., a change in the current over time on the dendrites of a neuron) produces its output (i.e., a voltage spike train) at the soma. This output spike train then travels down the axon to cause further current changes in the dendrites of its receiving (i.e., post-synaptic) neurons.

There is a broad spectrum of these neuron models. At one end of this spectrum are sophisticated models like Hodgkin-Huxley model or compartmental models which are computationally expensive to simulate (especially if one wants to simulate millions of neurons in a network). At the other end, we have models such as artificial neuron model which depart much farther from real biological neurons.

A model called Leaky Integrate-and-Fire (LIF) stands in the middle of this spectrum capturing both simplicity and high fidelity [37]. For this reason, in the NEF simulations we mostly use LIF neurons.

### 5.1.1 Leaky Integrate-fire Neuron

As mentioned, LIF is a simple model that approximates the behavior exhibited by different types of real neurons under broad range of conditions.

In the LIF neuron, the membrane voltage, $V(t)$, is changing by the sum of currents from its dendrites, $J_M(t)$. If this voltage reaches a threshold, $V_{th}$, a spike is generated in the soma and the membrane voltage resets to its resting value (normally zero). It remains at this resting value for a certain period of time, $\tau^{ref}$. On the other hand, if the total current in the dendrites does not cause the membrane voltage to reach to its threshold, over time this voltage will 'leak' back to its resting value without producing any spikes.

This behavior can be well explained using a passive $RC$ circuit in parallel with a branch containing a spike generator serried by a switch (figure 5.1). Equation (5.1) shows the mathematical expression of the LIF model:

$$\frac{dV}{dt} = -\frac{1}{RC}(V(t) - J_M(t)R). \tag{5.1}$$

The $RC$ value, $V_{th}$ and $\tau^{ref}$ are parameters of the model that are set and remain constant for every simulation.

## 5.2 Network Model

When one desires to model the functions of a neural system, understanding the math behind the model of a single neuron is essential but not sufficient. This stems from the fact that all sophisticated computations related to a neural system happens when individual neurons connect to one another creating a network.

The first two principles of the neural engineering framework (i.e., representation and transformation) demonstrate how connections between neurons can lead to the complex functionalities accomplished by the neural mechanisms.

## 5.3 Representation

Based on the NEF, we can create a population of neurons of which activities can represent a scalar, a 2-dimensional vector, or an n-dimensional vector. This scalar or vector can be an external stimulus, or an internal state variable.

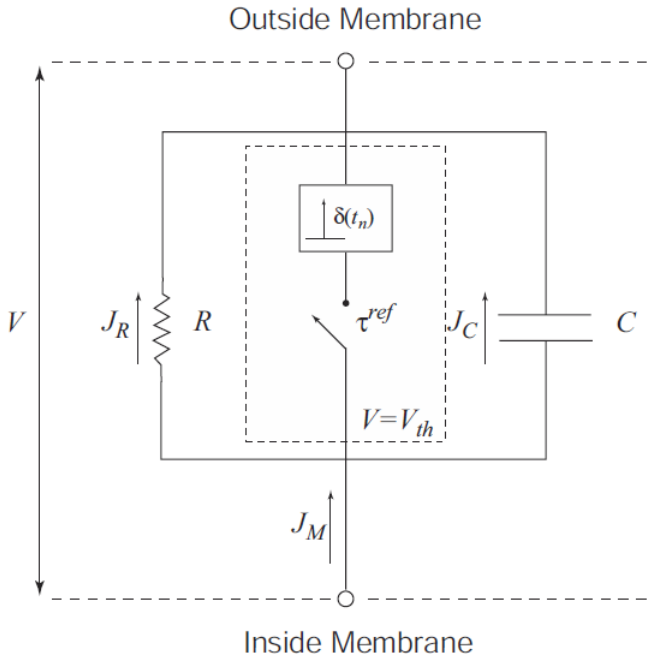



Figure 5.1: An RC circuit that shows the behavior of the LIF neuron, from [22]. At the time, $t_n$, when the membrane potential is equal to the threshold voltage, $V_{th}$, the switch is closed creating a spike, $\delta(t_n)$, and making the membrane potential go to zero. The switch remains closed for refactory period, $t^{ref}$, leading the membrane voltage to stay at its zero resting value for this period of time.

To demonstrate this with an example, consider a real life situation. In the smooth pursuit eye movement, as discussed in the previous chapter, it is essential to have a neural representation of the retinal velocity associated with a moving target. This retinal velocity has two components (i.e., it is a 2-dimensional vector) and it can be written as:

$$\boldsymbol{V}_{ret} = \begin{bmatrix} speed\ in\ x\ coordinate \\ speed\ in\ y\ coordinate \end{bmatrix}. \tag{5.2}$$

Now, for the sake of simplicity, assume that there are only two neurons in a population each having one preferred direction vector orthogonal to the other one's. To make the explanation even more clear, also assume that the preferred direction vector of the first one is $\hat{i} = [1, 0]$, while the preferred direction vector of the other one is $\hat{j} = [0, 1]$ (i.e., unit vectors in the Cartesian coordinate system). Under these assumptions, the spike-rates (the frequency at which spikes are generated along the axon) of the first and second neurons represent the values of the retinal velocity along the $x$ and the $y$ axes, respectively.

If there were no error due to noise and each neuron in the pair spanned the whole $x$ and $y$ axes respectively (see [22] for details), these two neurons would be enough to represent the retinal velocity. However, in reality where noise exists, there are thousands of neurons responsive to any given stimulus in the nervous system of the primates. Each of these neurons has its own preferred direction and represents only a portion of 2-dimensional space [59].

The above mentioned example helps us qualitatively understand how it is possible for a population of neurons to represent the retinal velocity. However, the discussion remains incomplete if we do not explain this process with the aid of mathematical equations.

As mentioned in section 5.1.1, the input to a neuron is the sum of the currents received

to its soma from all dendrites. We can think of this current, $J_M$, as summation of two currents $J_{bias}$ and $J_d$:

$$J_M = J_d(\boldsymbol{V}_{ret}) + J_{bias}, \tag{5.3}$$

where $J_{bias}$ accounts for the current that the neuron receives regardless of any stimulus and causes its background firing rate. $J_d$, on the other hand, is the current that changes with respect to the neuron's stimulus (i.e., retinal velocity in our example) and can be written as:

$$J_d(\boldsymbol{V}_{ret}) = \alpha\, \boldsymbol{e}.\boldsymbol{V}_{ret}, \tag{5.4}$$

where $\boldsymbol{e}$ is the neuron's preferred direction vector and $\alpha$ is a gain.

$J_M$, is the input to a non-linear function, $G\langle\rangle$, that resembles the behavior of the LIF neuron explained in section 5.1.1. Therefore, for the activity of a neuron we can write:

$$a(J_M) = G\langle J_M \rangle. \tag{5.5}$$

From (5.4) we can see that $J_d$ equals to the dot product between the neuron's preferred direction vector and the retinal velocity (i.e., stimulus vector) multiplied by a gain, $\alpha$. This means that preferred direction vector, $\boldsymbol{e}$, determines how a neuron would respond to its stimulus. Figure 5.2 shows how the two neurons in our example would respond to the changes of the retinal velocity. Plots like figure 5.2 that depicts how the activity of a neuron (i.e., number of spikes produced per second in the soma) changes based on its stimulus are called tuning curves.

Looking at figure 5.2 and also knowing that firing rates of neurons are polluted with noise, one can realize a single neuron cannot represent a 1-dimensional space without

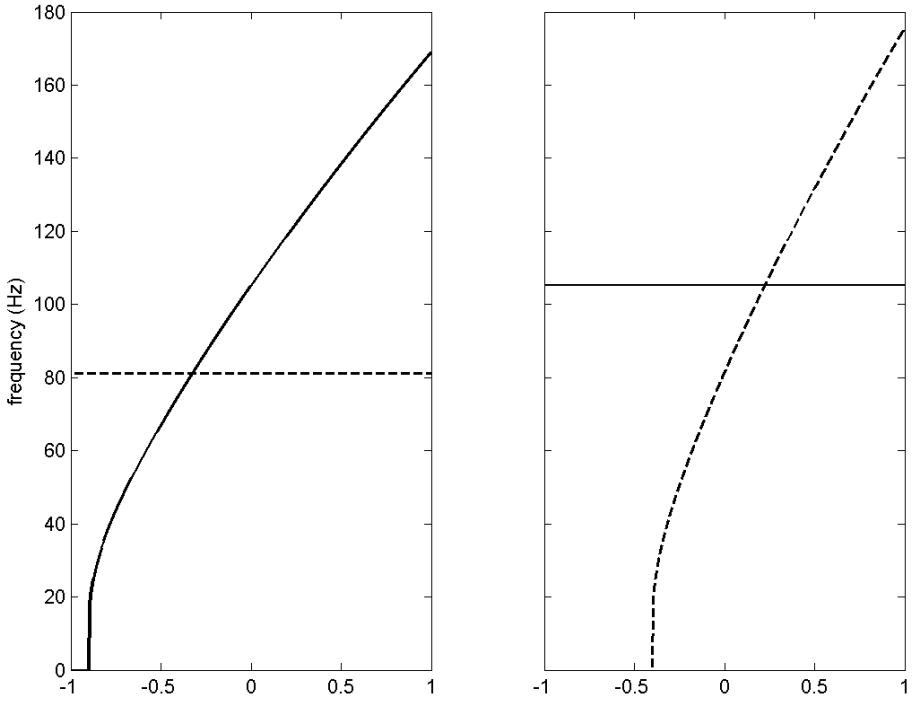



Figure 5.2: Tuning curves for two neurons in the mentioned example. Left: Retinal speed changing in x coordinate while its projection onto y coordinate is zero . Right: Retinal speed changing in y coordinate while its projection onto x coordinate is zero. Solid lines show the activity of the neuron with $\boldsymbol{e} = \hat{i}$ and dashed lines show the activity of neuron with $\boldsymbol{e} = \hat{j}$. $J_{bias}$ and $\alpha$ for each neuron were chosen randomly in the suitable range.

error. Elaismith and Anderson have shown that by increasing the number of neurons in a population, $N$, the mean square error of the representation by that population decreases at $1/N$ (see [22] for details). So it can be shown that the more neurons exist in a population the better representation it can achieve.

What was discussed so far, only shows us how a stimulus can change the neurons' activities in a population (i.e., how encoding happens). However, for a complete representation we should also talk about how it is possible to reconstruct the stimulus from neurons' activities (i.e., decoding process).

Assume that there exists a population of $N$ neurons that respond to an n-dimensional vector stimulus, $X$. If $a_i(t)$ shows the activity of the neuron $i$ related to a specific point in the n-dimensional space, by linear decoding we can represent this n-dimensional point ($\hat{X}$ is the estimate of $X$ from the activity of the population) (see equation (5.6)). For linear decoding, we need to find the right decoders associated with each neuron, $d_i$. This can be done using least square method (see [22] for details).

$$\hat{X} = \sum_{i=1}^{n} d_i a_i(t) \tag{5.6}$$

As discussed earlier, the activity of a neuron is a voltage spike-train:

$$a_i(t) = \sum_{s_i=1}^{nspikes} \delta(t - t_{s_i}) \tag{5.7}$$

Substituting (5.7) in (5.6) leads to:

$$\hat{X} = \sum_{i=1}^{n} d_i \sum_{s_i=1}^{nspikes} \delta(t - t_{s_i}) \tag{5.8}$$

However, (5.8) is not completely accurate. Spike trains are filtered in the brain by postsynaptic current (PSC) dynamics. PSC dynamics are approximately: $h_{PSC}(t) = \frac{1}{\tau_{syn}} e^{-t/\tau_{syn}}$, where $\tau_{syn}$ is the synaptic time constant. So it is more appropriate to rewrite (5.8) as:

$$\begin{aligned} \hat{X} &= \sum_{i=1}^{n} d_i \Big( \sum_{s_i=1}^{nspikes} \delta(t - t_{s_i}) \Big) * h_{PSC}(t) \\ &= \sum_{i=1}^{n} d_i \sum_{s_i=1}^{nspikes} h_{PSC}(t - t_{s_i}) \end{aligned} \tag{5.9}$$

By explaining how encoding and decoding happen in a population of LIF neurons we have completed our discussion on representation. In the next chapter, the second principle of the NEF (i.e., transformation) is described. This helps us understand how to compute a function between populations of neurons by setting appropriate synaptic weights.

## 5.4 Transformation

In previous section, we explained how a population of LIF neurons can represent their stimuli. However, when one thinks of the complex tasks accomplished by the neural system of animals, it becomes clear to him that it could not be all. To survive not only should a neural system be able to represent external stimuli but also should it be able to combine different stimuli and compute linear and/or non-linear functions from them. For example in the smooth pursuit system (see chapter 3) a nonlinear function from the retinal velocity needed to be computed. Also some information should be added together. In this section by describing two examples, it will be shown how population of LIF neurons can do such tasks.

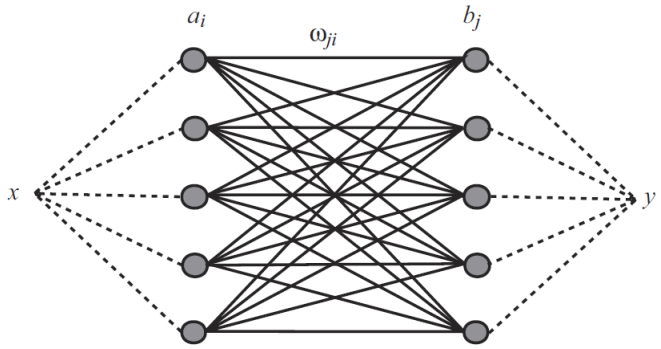



Figure 5.3: A communication channel in a neural network, from [22]. Solid lines indicate possible connections. Dotted lines indicate an encoding/decoding relation.

The first example is a communication channel. A communication channel simply sends information from one location (here one population) to another. For this purpose, let us consider two populations (i.e., population $a$ and population $b$). Neurons in these two populations are connected in the way depicted in figure 5.3.

Based on what was discussed about decoding in section 5.3, we can write:

$$\hat{x} = \sum_i a_i(x) d_i^x, \tag{5.10}$$

where $a_i(x)$ is the activity of the neuron $i$ caused by the stimulus, $x$:

$$\begin{aligned} a_i(x) &= G_i\Big[J_i(x)\Big] \\ &= G_i\Big[\alpha_i \boldsymbol{e}_i x + J_i^{bias}\Big] \end{aligned} \tag{5.11}$$

Same equations are true for population $b$:

$$\hat{y} = \sum_j b_j(y) d_y^y, \tag{5.12}$$

$$\begin{aligned} b_j(y) &= G_j[J_j(y)] \\ &= G_j\Big[\alpha_j e_j y + J_j^{bias}\Big] \end{aligned} \tag{5.13}$$

Notice that for a communication channel y = x, and we should substitute $y$ with $\hat{x}$ in (5.13):

$$b_j(x) = G_j\Big[\alpha_j e_j \hat{x} + J_j^{bias}\Big] \tag{5.14}$$

Since $\hat{x} = \sum_i a_i(x) d_i^x$ it becomes:

$$\begin{aligned} b_j(x) &= G_j\Big[\alpha_j e_j \sum_i a_i(x) d_i^x + J_j^{bias}\Big] \\ &= G_j\Big[\sum_i \omega_{ji} a_i(x) + J_j^{bias}\Big], \end{aligned} \tag{5.15}$$

where $\omega_{ji} = \alpha_i e_j d_i^x$ are synaptic weights between neurons of population $a$ and population $b$. Therefore, we can conclude that by choosing synaptic weights in this way population $a$ and population $b$ work as a communication channel.

Note that in this example, if decoders of population $b$, $d_j^x$, were optimized to decode a function of $x$ instead of simply decoding $x$ (i.e., $d_j^{f(x)}$ instead of $d_j^x$), this population would represent $f(x)$ (i.e., a function of $x$) instead of $x$ itself:

$$\hat{f}(x) = \sum_j b_j(x) d_j^{f(x)} \tag{5.16}$$

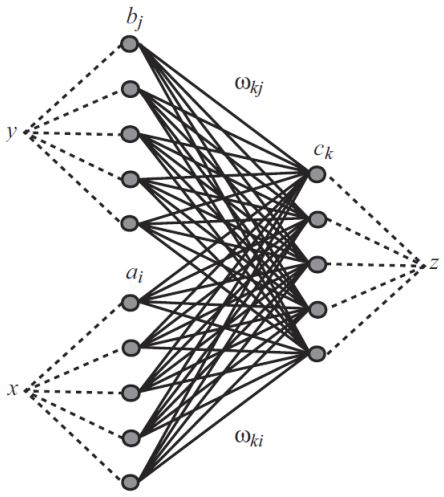



Figure 5.4: A network to sum two variables, from [22]. Solid lines indicate possible connections. Dotted lines indicate an encoding/decoding relation.

The second example in this section depicts how one population can represent the addition of two variables, each represented by a population.

Figure 5.4 shows connections between neurons in the populations $a$, $b$, and $c$. This time we want to see what synaptic weight should be chosen so: $z = x + y$

Let us start with the activities of the neurons in population $c$:

$$c_k(x+y) = G_k\Big[\alpha_k e_k(\hat{x}+\hat{y}) + J_k^{bias}\Big] \tag{5.17}$$

Now, by substituting $\hat{x}$ and $\hat{y}$ based on the linear decoding we discussed, (5.17) becomes:

$$\begin{aligned} c_k(x+y) &= G_j\Big[\alpha_k e_k\Big(\sum_i a_i(x)d_i^x + \sum_j a_j(x)d_j^y\Big) + J_k^{bias}\Big] \\ &= G_j\Big[\sum_i \omega_{ki}a_i(x) + \sum_j \omega_{kj}b_j(y) + J_k^{bias}\Big], \end{aligned} \tag{5.18}$$

where $\omega_{ki} = \alpha_k e_k d_i^x$ and $\omega_{kj} = \alpha_k e_k d_j^y$. Therefore, we found the synaptic weights between neurons of these three populations so that $z$ $(= x + y)$ can be decoded from population $c$:

$$\hat{z} = \sum_k c_k(z) d_k^k, \tag{5.19}$$

## 5.5 Dynamics

To this point, the first two principles of the NEF have been described. With the use of these two principles, one can explain both linear and non-linear transformations happening in the neurobiological systems. However, since these systems are dynamic, it is impossible to ignore time. Namely, we need to know that how long it takes for these systems to get from one state to another. The third principle of the NEF uses linear control theory to study the dynamics of the neural systems. Moreover, it shows how to find a neural equivalent for an arbitrary standard linear system.

In control theory, there are two main equations that summarize how internal states of a linear system, $x(t)$, are related to its input and output (i.e., $u(t)$ and $y(t)$, respectively):

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5.20}$$

$$y(t) = Cx(t) + Du(t), \tag{5.21}$$

where $A$, $B$, $C$, and $D$ are the dynamic, input, output, and feedthrough matrices, respectively (see figure 5.5).

Equations (5.20) and (5.21) can be written in Laplace domain as:

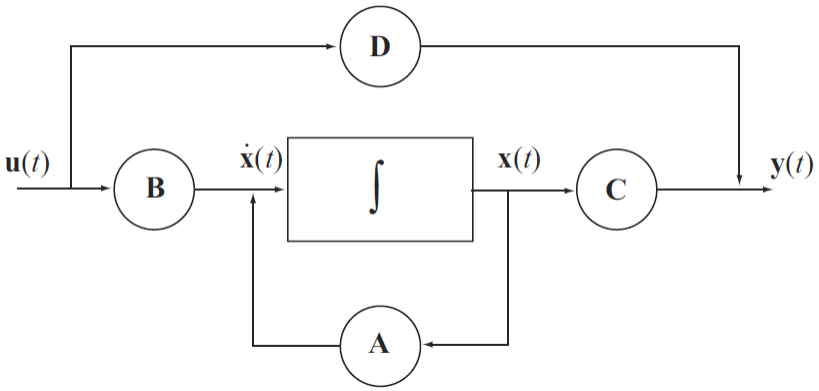



Figure 5.5: Block diagram for a time-invariant linear system showing the elements of equations (5.20) and (5.21), from [22]. For a standard linear system the transfer function which relates $\dot{x}(t)$ to $x(t)$ is integration.

$$s\boldsymbol{x}(s) = A\boldsymbol{x}(s) + B\boldsymbol{u}(s) \tag{5.22}$$

$$\boldsymbol{y}(s) = C\boldsymbol{x}(s) + D\boldsymbol{u}(s) \tag{5.23}$$

To find a neural equivalent for the standard linear system, we should find out what can replace the integrator (i.e., the transfer function in figure 5.5 that relates $\dot{x}(t)$ to $x(t)$). Based on what has been described for the LIF neurons, there are two possibilities for the transfer function of a neural population: synaptic filter, $h_{PSC}(t)$, and the spiking dynamics. In [22] it is shown that synaptic filter dominates the spiking dynamics. Therefore, the transfer function for a population of LIF neurons can be written as: $h(t) = \frac{1}{\tau}e^{-t/\tau}$, where $\tau$ is the synaptic time constant (see figure 5.6).

An equation analogous to (5.20) for the neural system can be written as:

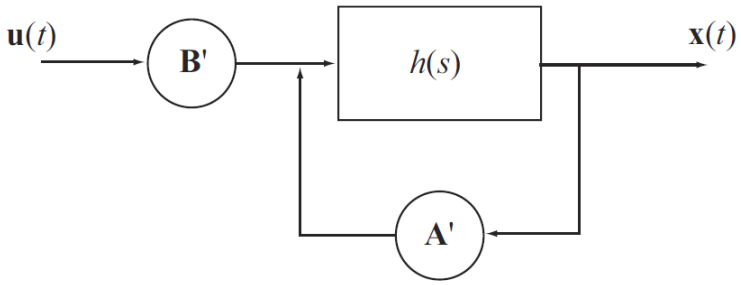



Figure 5.6: Block diagram for a population of LIF neurons as a linear system, from [22]. This diagram shows the elements of equation (5.24). Notice that for this system feedthrough matrix, $D(t)$, is zero and the output matrix, $C(t)$, is identity. Furthermore, the transfer function which relates $\dot{x}(t)$ to $x(t)$ is the synaptic filter. (See the text for more details.)

$$x(t) = h(t) * [Ax(t) + Bu(t)] \tag{5.24}$$

Knowing that the synaptic filter in Laplace domain is: $\boldsymbol{h}(s) = \frac{1}{1+s\tau}$, one can write (5.24) in Laplace domain as:

$$\boldsymbol{x}(s) = \frac{1}{1+s\tau}[A'\boldsymbol{x}(s) + B'\boldsymbol{u}(s)] \tag{5.25}$$

Rearranging (5.25) results in:

$$s\boldsymbol{x}(s) = \tau^{-1}[A' - I]\boldsymbol{x}(s) + \tau^{-1}B'\boldsymbol{u}(s) \tag{5.26}$$

Equating the right hand sides of (5.22) and (5.26) leads to the relationship between the dynamics and input matrices for the standard linear system and the neural system:

$$\boldsymbol{A}' = \tau \boldsymbol{A} + \boldsymbol{I} \tag{5.27}$$

$$\boldsymbol{B}' = \tau \boldsymbol{B} \tag{5.28}$$

By the use of (5.27) and (5.28) one can easily find the equivalent neural system of a desired linear system. For example, $\boldsymbol{A} = 0$ and $\boldsymbol{B} = 1$ describe an integrator in standard control. Therefore, $\boldsymbol{A}' = I$ and $\boldsymbol{B}' = \tau$, leads to a population of neurons that works as an integrator.

# Chapter 6

# A New Model of SPEM to Interact with the Real World

## 6.1 Novelty of the Model

Object tracking is a well-studied topic in both computer vision and machine vision. It has a variety of uses in robotics [53], human-computer interaction [6], and other applications [16]. However, despite its extensive applications, robust and efficient tracking is still considered an open research area [76]. For example, very recently, Doyle et al. published a paper closely related to this thesis, in which they explain the use of an optical flow estimation method for real-time general purpose tracking with a pan-tilt camera [16].

Since in general animals operate much more effectively in natural environments than current robots, it makes sense to examine neurobiological mechanisms of control (e.g., gaze control during object tracking). Moreover, the brain network that controls eye movements in humans and other primates is relatively simple and well-studied. Therefore, learning the

brain mechanisms of gaze control is useful not only to develop more robust tracking algorithms, but also to pave the way to understand other more complicated control strategies employed in the brain [45]. The current model attempts to implement such an idea.

To my knowledge, this model is the first neurocomputational SPEM model that interacts with the real world in real time. It uses naturalistic images as input and spiking model neurons to control servomotors to drive a pan-tilt camera.

## 6.2 Model Description

In this section, we see how the model relates anatomically and functionally to different parts of the human body (the brain in particular). Then, the implementation of smooth pursuit controller in spiking neurons will be explained.

### 6.2.1 Hardware Analogy between the Model and the Human Body

Figure 6.1 shows the pan-tilt camera which operates as the eye and eye muscles. It consists of a GoPro Hero3 camera (http://gopro.com/cameras/) and two Hitec HSR-5990TG servomotors (http://hitecrcd.com/). The camera is connected to the computer via an HDMI cable for frame acquisition and the servomotors are conencted to an Arduino board (i.e., an open-source microcontroller, http://arduino.cc/). The Arduino board receives position commands from the computer via a serial port and sends the appropriate pulse width modulation (PWM) signals to place the servomotors at the right position.

Figure 6.2 shows how one can compare different parts of the human body to the hardware of the model. The camera works as a surrogate eye to capture scenes, the computer

Figure 6.1: Illustration of the pan-tilt camera.

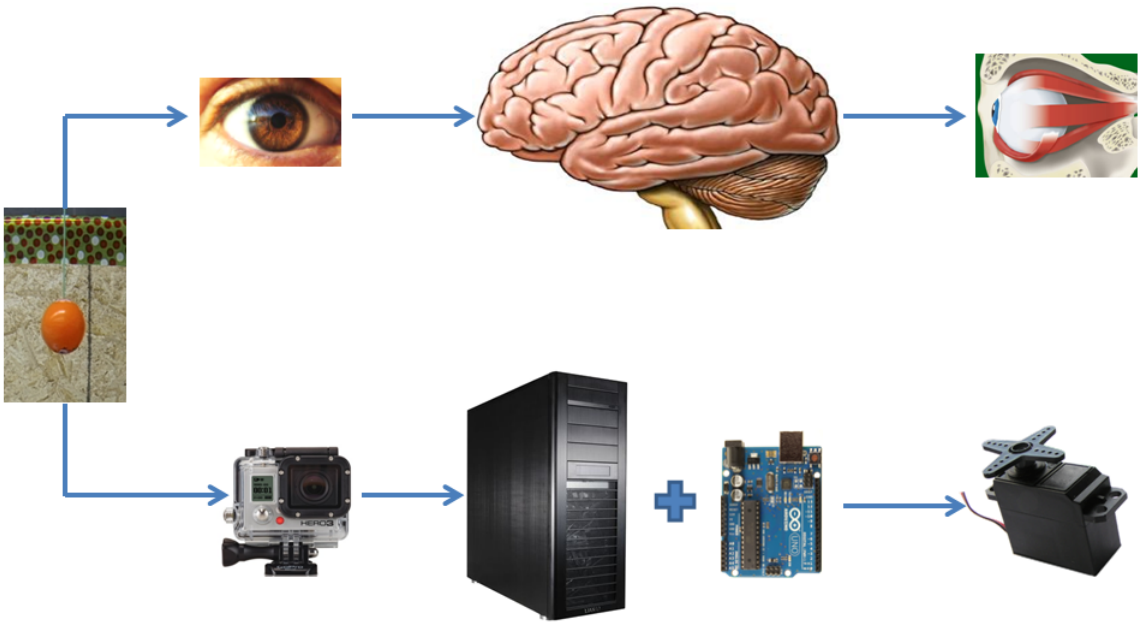


Figure 6.2: The hardware analogy between the model and the human body. See section 6.2.1 for details.

and Arduino board operate as a brain to produce and send necessary commands to the servomotors. The servomotors act as extraocular muscles (i.e., eye muscles) to move the camera (i.e., surrogate eye) in the right direction for object tracking.

In order to make sure that the dynamics of the model's hardware (i.e., servomotors, the camera, and the HDMI cable connected to the camera) would not affect the dynamics of the SP controller in the frequency range of the experiment, the approximate bode plot of the setting was created.

A simple model of the eye dynamics (i.e., a low-pass filter) was simulated in software.

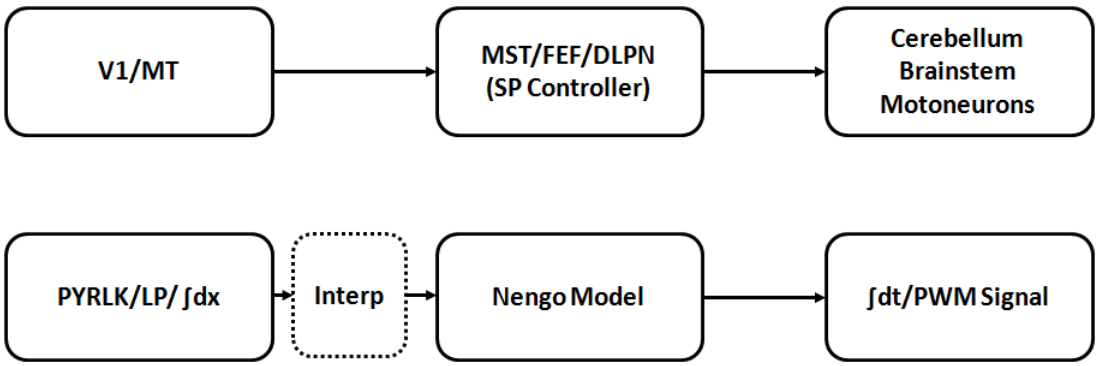



Figure 6.3: Different parts of the brain involved in SPEM along with their counterparts in the model. See section 6.2.2 for details.

### 6.2.2 Functional Analogy between the Model and the Brain

In chapter 2, neural circuitry of SPEM was discussed in detail. We saw different brain areas involved in SPEM and their possible roles. This section describes different modules in the software of the model which are implemented to mimic the functionality of these brain areas (see figure 6.3).

In chapter 2, we saw that the primary visual cortex (V1) along with the middle temporal visual area (MT) are responsible to provide the retinal velocity input for the network in the brain that controls SPEM. To provide the retinal velocity for the model, the pyramidal Lucas and Kanade (PYRLK) method was implemented on GPU to accurately estimate optical flow (see section 4.3.1). Furthermore, because of the low-pass temporal-filtering characteristics of MT [47, 59], a second order low-pass (LP) Butterworth filter was created

to make sure that only low-frequency components of the velocity signal reach the SP controller.

To extract the velocity information that corresponds to the target, a spatial Gaussian average was applied on the central region of the frames. This produced a single velocity vector that could serve as input to the SPEM system.

Frames from the camera were captured at 25Hz (this frame rate was imposed by PYRLK method see section 4.3.3). However, since neural spikes occur with approximately 1ms temporal resolution, the velocity signal was interpolated to obtain a .001s time step for the SP controller.

For the SP controller, a network of LIF neurons was generated in Nengo (http://nengo.ca), which is a Python package based on the NEF (see chapter 5 to learn about the NEF and see section 6.2.4 for more details on this part of the model).

Since Arduino accepts a position (angle) input to create the right PWM signal for the servomotors, the velocity command produced by the Nengo network (i.e SP controller) is integrated over time to create the position command. Time integration resembles the performance of the brainstem nuclei (see section 2.6). Moreover, PWM signals produced by the Arduino board can be thought of as the electro-chemical signals that cerebellum and brainstem produce and then send to motoneurons to appropriately stimulate eye muscles so that the eyes are placed at the right position (i.e., holding the retinal image of the object on the fovea) during object tracking.

As mentioned earlier, the SP neural controller was generated in Python. All other modules including PYRLK, spatial integration, and temporal interpolation and integration were generated in MATLAB.

The communication between the MATLAB modules and the Python module is possible

| Area | Function within Model | | |
|---|---|---|---|
| MT | Retinal velocity representation | | |
| MST | Nonlinear saturations of retinal velocity | | |
| FEF | Differentiation | Nonlinear saturation of retinal acceleration | Low-pass filtering |
| DLPN | Low-pass filtering | Summation | Integration |

Table 6.1: Functional role of each brain area in the smooth pursuit control model suggested by Churchland and Lisberger [8]. MT = middle temporal area of the visual cortex; MST = medial superior temporal area; FEF = frontal eye fields; DLPN = dorsolateral pontine nucleus.

by using User Datagram Protocol (UDP).

### 6.2.3 Mapping the Smooth Pursuit Controller onto Brain Areas

The starting point for the smooth pursuit controller was the model suggested by Churchland and Lisberger, 2001 [8]. As discussed in section 3.5, this model is a well-established high-level control model for controlling smooth pursuit eye movement which has been validated against Rhesus monkeys.

Figure 6.4 shows how different parts of this model can be mapped onto different areas of the brain involved in the SPEM neural circuity based on neural recording of different brain areas (see section 2.5).

Table 6.1 summarizes the functional role of each brain area in the smooth pursuit control model suggested by Churchland and Lisberger.

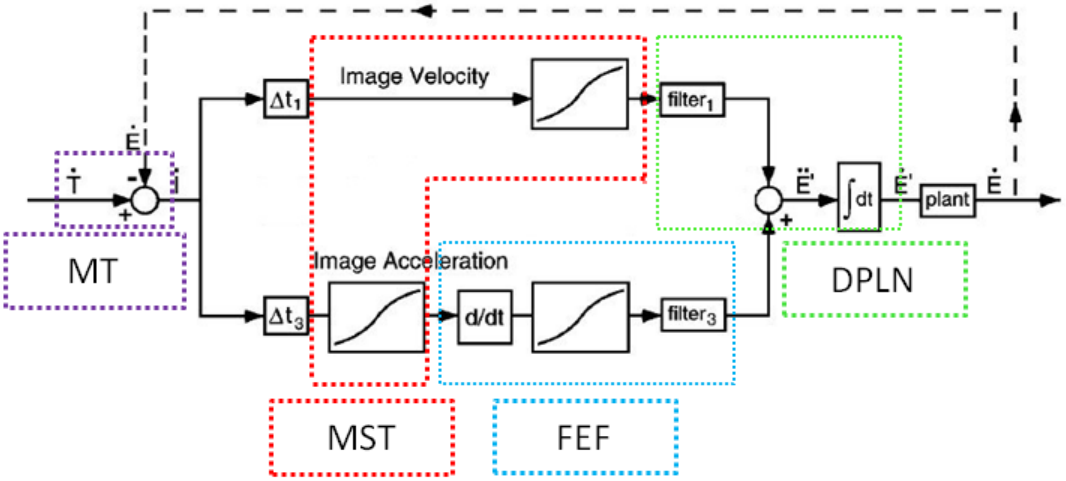



Figure 6.4: Mapping the smooth pursuit control model suggested by Churchland and Lisberger onto different brain areas. MT = middle temporal area of the visual cortex; MST = medial superior temporal area; FEF = frontal eye fields; DLPN = dorsolateral pontine nucleus. (see chapter 2 to learn about different brain areas and their functionality). See table 6.1.

### 6.2.4 The NEF Model of the Smooth Pursuit Controller

Based on table 6.1, a network of LIF neurons was created to have populations for representation, non-linear decoding, differentiation, low-pass filtering, summation and integration.

From what was discussed in sections 5.3 and 5.4, the NEF allows us to create populations of LIF neurons to represent a scaler or vector, and decode linear or non-linear functions from their activities. Moreover, it was explained in section 5.5 how to build a neural integrator. Therefore, if two remaining functions (i.e., low-pass filter and differentiator) can be implemented in LIF neurons, we have all necessary components to build a smooth pursuit neural controller.

However, another look at the functionalities attributed to FEF in table 6.1 reveals that it is possible to combine the differentiator, $\frac{s}{1+\tau_d s}$, with the low-pas filter, $\frac{1}{1+\tau_{lp} s}$, to have one second-order band-pass filter, $\frac{s}{(1+\tau_d s)(1+\tau_{lp} s)}$, instead. Consequently, in place of implementing two low-pass filters and a differentiator, it is possible to implement one low-pass and one band-pass filter.

#### Low-pass Filter Design with the NEF

A low-pass filter occurs naturally in the connection from one population to another (synapses). Biological synapses are low-pass because an incoming spike opens a large number of ions channels in the cell membrane of the receiving neuron, which then close at random with roughly exponential continuum dynamics. This behavior can be approximated with a first-order low-pass filter: $h_{PSC}(s) = \frac{1}{1+\tau_{syn} s}$, where $\tau_{syn}$ is the synaptic time constant. This time constant is most typically in the 5-10ms range [14]. To have first-order low-pass neural filters with any desired time constant, one can add a feedback connection as will be explained in the following paragraphs.

The transfer function of a first-order low-pass filter with a desired time constant, $\tau_{filter}$, can be written as:

$$h(s) = \frac{y(s)}{u(s)} = \frac{1}{1 + \tau_{filter} s}. \tag{6.1}$$

Rearranging (6.1) and using $y(s) = x(s)$ lead to:

$$sx(s) = \frac{-1}{\tau_{filter}} x(s) + \frac{1}{\tau_{filter}} u(s), \tag{6.2}$$

where $\frac{-1}{\tau_{filter}}$ is the dynamics matrix, $A$, and $\frac{1}{\tau_{filter}}$ is the input matrix, $B$.

Based on (6.1) and the relationship between the standard linear system and the neural system which was discussed in section 5.5 (see equations (5.27) and (5.28)), $A' = 1 - \frac{\tau_{syn}}{\tau_{filter}}$ and $B' = \frac{\tau_{syn}}{\tau_{filter}}$ describe a first-order low-pass neural filter with $\tau_{filter}$ as the time constant ($A'$ and $B'$ are dynamics and input matrices of the neural system).

### Second-order Band-pass Filter Design with the NEF

Tripp and Eliasmith [80], have shown how to implement a second-order band-pass Butterworth filter in neurons using a feedback network. The transfer function of this filter is:

$$\frac{y(s)}{u(s)} = \frac{\omega^2 s}{s^2 + 2\omega^{1/2} s + \omega^2}, \tag{6.3}$$

where $\omega$ is the corner frequency, in radians/s.

On the other hand, in the smooth pursuit controller suggested by Churchland and Lisberger, the differentiator and the low-pass filter in the image acceleration pathway (see

figure 6.4) have the same time constant (4ms). Therefore, the resultant bandpass filter coming from the combination of these two can be written as:

$$\frac{y(s)}{u(s)} = \frac{s}{\tau^2 s^2 + 2\tau s + 1}, \tag{6.4}$$

where $\tau$ is the time constant.

Dividing both numerator and denominator of the right hand side of (6.4) by $\tau^2$ and replacing $\frac{1}{\tau}$ by $\omega$ result in:

$$\frac{y(s)}{u(s)} = \frac{\omega^2 s}{s^2 + 2\omega s + \omega^2}, \tag{6.5}$$

where $\omega$ is the corner frequency, in radians/s.

Since (6.3) and (6.5) are similar, same idea from [80] can be applied. First, we can write a realization for (6.5):

$$\begin{aligned} \dot{x}_1(t) &= -\omega x_1(t) + (\tfrac{p_1\omega}{p_2})x_2(t) + p_1\omega^2 u(t), \\ \dot{x}_2(t) &= -\omega x_2(t) - p_2\omega^2 u(t), \\ y(t) &= \tfrac{1}{p_1}x_1(t), \end{aligned} \tag{6.6}$$

where $\omega$ is the corner frequency. The variables $p_1$ and $p_2$ scale the two state variables to the suitable range in which the firing rates of the neurons that encode the state variables, $x_1$ and $x_2$, do not saturate [80].

As mentioned in section 5.5, to find a neural network with a desired dynamics, the desired state space should be transformed into a form that uses post-synaptic current dynamics instead of integration ([22] and [80]). To apply this, we can first write the state space equations (6.6) in the Laplace domain and use vector and matrix notation:

$$s \begin{bmatrix} x_1(s) \\ x_2(s) \end{bmatrix} = \begin{bmatrix} -\omega & \frac{p_1\omega}{p_2} \\ 0 & -\omega \end{bmatrix} \begin{bmatrix} x_1(s) \\ x_2(s) \end{bmatrix} + \begin{bmatrix} p_1\omega^2 \\ -p_2\omega^2 \end{bmatrix} u(s). \tag{6.7}$$

Now, it is possible to rewrite (6.7) in terms of post-synaptic current (PSC) dynamics, which have transfer function $\frac{1}{1+\tau s}$:

$$\begin{bmatrix} x_1(s) \\ x_2(s) \end{bmatrix} = \frac{1}{1+\tau s} \begin{bmatrix} 1-\tau\omega & \frac{p_1\tau\omega}{p_2} \\ 0 & 1-\tau\omega \end{bmatrix} \begin{bmatrix} x_1(s) \\ x_2(s) \end{bmatrix} + \begin{bmatrix} p_1\tau\omega^2 \\ -p_2\tau\omega^2 \end{bmatrix} u(s). \tag{6.8}$$

These equations lead to the structure shown in figure 6.5. This structure shows that two populations of LIF neurons with proper connections can encode two state variables in a second-order band-pass filter.

Now that we see the implementation of both low-pass and band-pass neural filters, we have all functionalities needed for the smooth pursuit controller. Figure 6.6 shows the neural network designed to function as the smooth pursuit controller suggested by Churchland and Lisberger.

To complete this section, it should be mentioned that to make sure that the neural controller behaves as close as possible to the Churchland and Lisburger control model, two delay nodes, have been added to the neural model. These delay nodes introduce a pure time delay between when the pre-synaptic population produces an output and when the post-synaptic population receives it (these delays may correspond to axon conduction times).

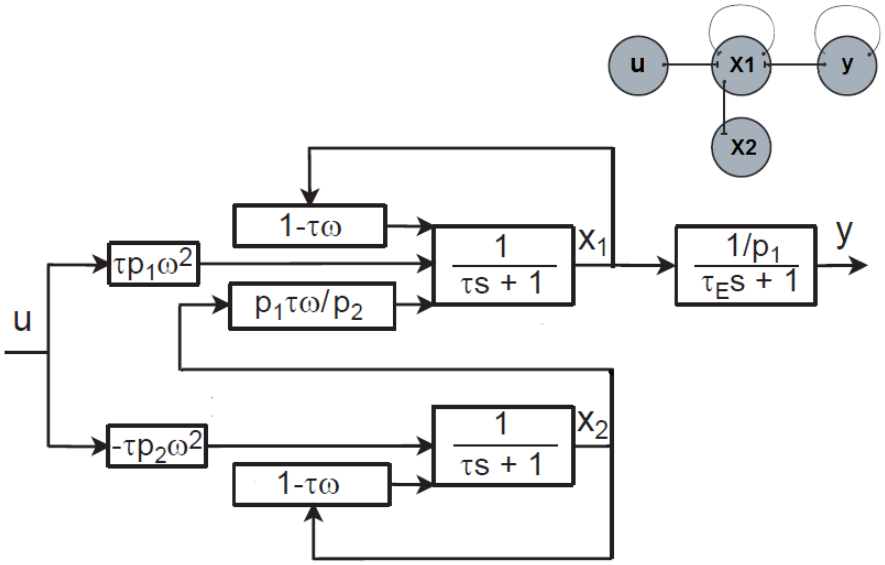



Figure 6.5: Block diagram of a neural feedback network which acts as a second-order band-pass filter, from [80] with modification. Inset shows the neural network structure with 4 populations, one for encoding the input, $u$, one for encoding the output, $y$, and two for encoding the state variables, $x_1$ and $x_2$, in the band-pass filter.

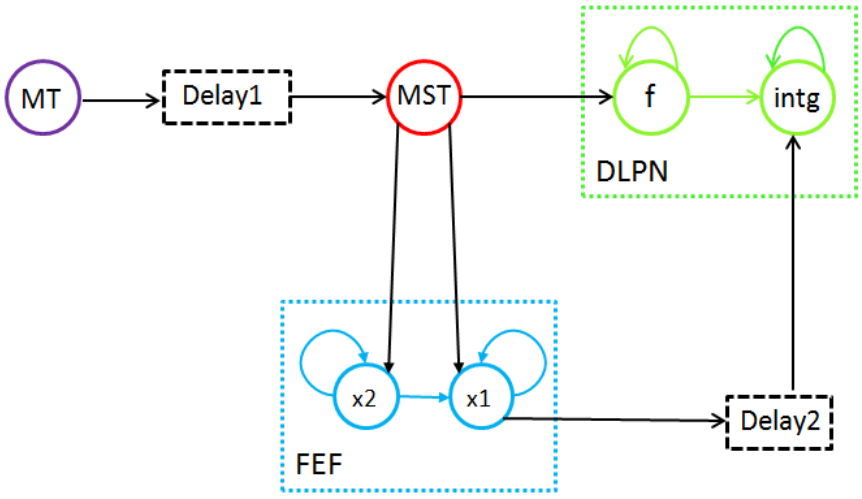



Figure 6.6: Illustration of the smooth pursuit neural controller (see figure 6.4 for comparison). MT = middle temporal area of the visual cortex; MST = medial superior temporal area; FEF = frontal eye fields; DLPN = dorsolateral pontine nucleus. MT represents the retinal velocity associated with the target (i.e., moving object). Two non-linear functions are decoded from MST. FEF is subdivided into two populations: x1 and x2 encode different state variables in a band-pass filter; besides, a non-linear function is decoded from x1. DLPN is also subdivided into two populations: f encodes the state of a low-pass filter; intg encodes the state of an integrator (see table 6.1).

## 6.3 Simulation and Experiment Results

### 6.3.1 Nengo Simulations

This section describes the simulation results of the neural smooth pursuit (SP) controller along with its comparison to Churchland and Lisberger's SP controller. Note that for these simulations a simple model of eye dynamics (i.e., a low-pass filter) was created and then a negative feedback from the output of the eye model was added to the MT population to resemble the attachment of the retina to the moving eyeball (see figure 3.1 in chapter 3).

Table 6.2 shows the parameters of Churchland and Lisberger's SP controller. The neural model was implemented in a way to have the same low-pass and band-pass filter time constants and the same non-linear saturations (see section 6.2.4). However, there are two major differences between these two models. The first difference is the neural model has one extra pole. This extra pole corresponds to synaptic dynamics in the connection between MT and MST populations. The second difference is related to delays. In Churchland and Lisberger's controller, there is a pure delay before the first non-linearity in each pathway (i.e., image velocity and image acceleration pathways in figure 6.4). In the neural SP controller, where both of these non-linearities were decoded from the MST population, a delay equal to the smaller delay in Churchland and Lisberger's model was added between MT and MST populations. Moreover, a delay equal to the difference between larger and smaller delays in Churchland and Lisberger's model was added between FEF and DLPN populations to create the same delay in the acceleration pathway of the neural model. This was done so that the MST component of each pathway would receive signals at the same time, so that a single MST population could be used. One can think of these delays in the neural SP controller as axon conduction delay.

| Component | Time/Time constant (ms) |
|---|---|
| Delay | 72 |
| | |
| Low-pass filter | 55 |

Image velocity pathway

| Component | Time/Time constant (ms) |
|---|---|
| Delay | 77 |
| Differentiator | 4 |
| Low-pass filter | 4 |

Image acceleration pathway

Table 6.2: Parameters of Churchland and Lisberger's model (see figure 6.4).

| Parameter | Value |
|---|---|
| $\tau_{ref}$ | .002 |
| RC | .02 |
| $V_{thresh}$ | 1 |

Table 6.3: LIF parameters

Table 6.3 shows the parameters of the LIF neurons in the simulation. Table 6.4 summarizes the number and the representation range of neurons in each population. Table 6.5 presents post-synapatice time constants.

| Population | Number | Range |
|---|---|---|
| MT | 1000 | ±20 |
| MST | 1000 | ±20 |
| f | 2000 | ±70 |
| x1 | 2000 | ±40 |
| x2 | 2000 | ±90 |
| intg | 1000 | ±20 |

Table 6.4: Number and representation range of neurons in each population

| Connection | Time constant (ms) |
|---|---|
| MT-MST | 5 |
| MST-f | 10 |
| MST-x1 | 10 |
| MST-x2 | 10 |
| x1-x1 | 10 |
| x2-x1 | 10 |
| x2-x2 | 10 |
| x1-intg | 100 |
| f-f | 10 |
| f-intg | 100 |
| intg-intg | 100 |

Table 6.5: Post-synaptic time constants

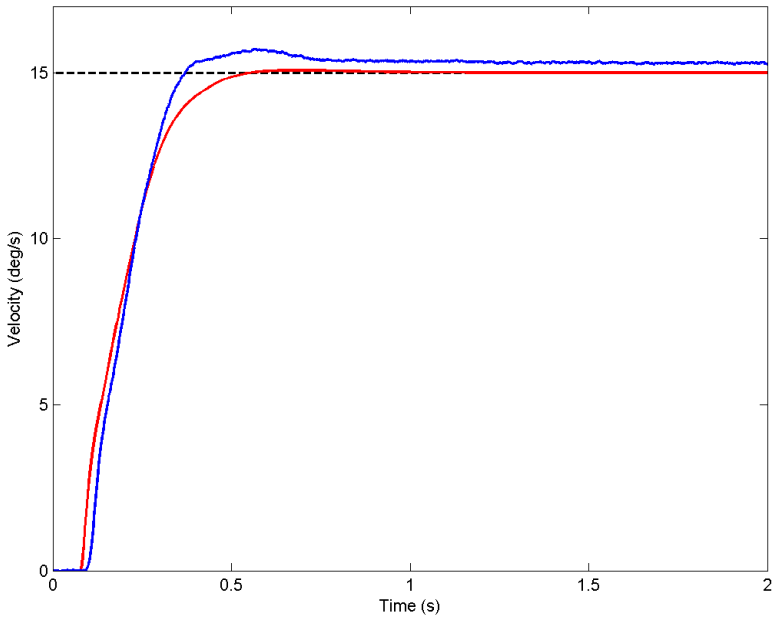



Figure 6.7: Response comparison between Churchland and Lisberger's control model and the neural model to 15 $deg/s$ step input. Blue line is the output of the neural model, red line is the output of the control model and black dash line is the input. See 6.3.1 for explanation.

### Step Velocity Input

Figure 6.7 shows the responses of both controllers to 15 $deg/s$ step velocity input. Two major differences exist between the models' responses. First, the response of the neural controller exhibits a larger time lag due to the extra pole that connects MT and MST populations. Second, there is a steady state error in the neural model as a consequence of point attractors in the neural network (see section 6.3.1).

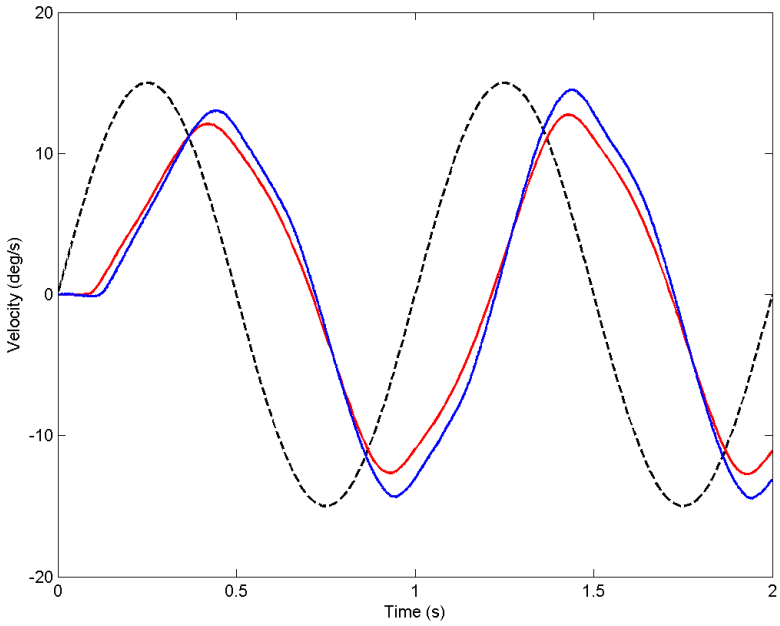



Figure 6.8: Response comparison between Churchland and Lisberger's control model and neural model to a sine wave with large amplitude. Blue line is the output of the neural model, red line is the output of the control model, and black dash line is the input. See section 6.3.1 for explanation.

### Sine Wave Velocity Input with the Amplitude of 15 $deg/s$

Figure 6.8 depicts the responses of the models to a sine wave with the amplitude of 15 $deg/s$ and the frequency of 1Hz. The main differences between the two models are greater phase lag as a result of the extra pole in the neural model and difference in amplitude due to the existence of distortion in the neural model (see section 6.3.1). However, the neural and control model responses are quite similar.

### Sine Wave Velocity Input with the Amplitude of 2 $deg/s$

Figure 6.9 shows the responses of the models to a sine wave with the amplitude of 2 $deg/s$ and frequency of 1Hz in two different simulations. The same differences as explained in 6.3.1 can be seen between the responses of the control model and the neural model. Furthermore, the response of the neural model is different in two simulations due to different initialization of some neuron parameters (e.g., maximum firing rates and preferred direction vectors). One simulation (left figure in 6.9) depicts a fairly good tracking by the neural controller, while the other simulation (right in figure 6.9) depicts a worse performance. Generally, a poor performance of the neural model was more common to happen for smaller inputs since the effect of point attractors in the network became more dominant (see section 6.3.1).

### Point Attractors

Some of the differences we saw between the responses of two models were mainly created as a consequence of point attractors in the neural network. In neural networks with feedback, point attractors emerge as a result of distortion in the estimation of state variables. To illustrate this phenomenon with an example, we can consider the neural integrator which was discussed in section 5.5. We saw that the dynamics of a neural integrator in Laplace domain can be written as:

$$\hat{x}(s) = \frac{1}{1+\tau\, s}(A'\hat{x}(s) + B'u(s)) \tag{6.9}$$

where dynamics matrix, $A'$, is equal to one and the input matrix, $B'$, is equal to post-synaptic time constant, $\tau$.

Taking the inverse Laplace transform, we can see from (6.9) that in each time step,

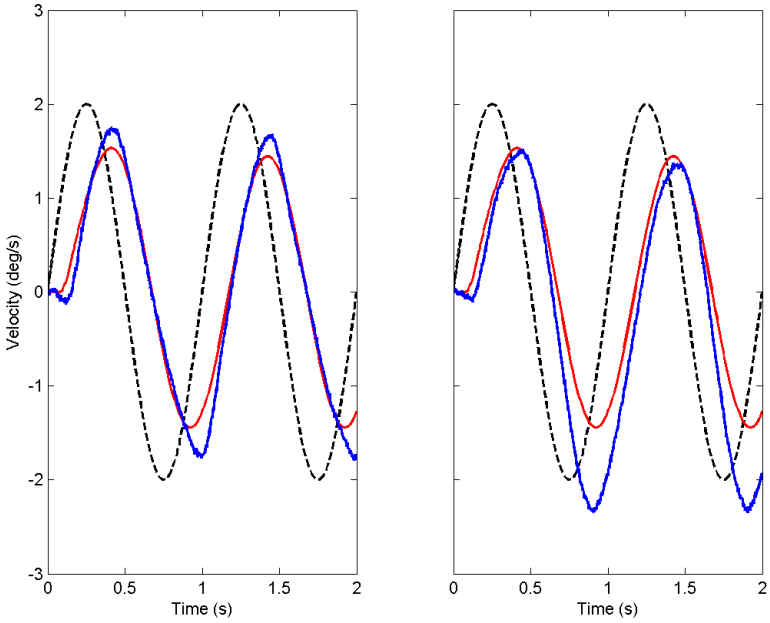



Figure 6.9: Response comparison between Churchland and Lisberger's control model and the neural model to a sine wave with the amplitude of 2 $deg/s$. Left figure shows a fairly good performance by the neural model, while right figure shows a worse performance of the neural model. The performance differences are due to differences in the population distortion functions (see figure 6.10 for examples of distortion function) in different instantiations of the model. Note that the same number of neurons was used in both cases and distortion differences are due to neuron parameters that are chosen at random from realistic ranges. Blue lines are the output of the neural model, red lines are the output of the control model and black dash lines are the input. See section 6.3.1 for explanation.

$\hat{x}(t)$ is a weighted sum of $\frac{1}{\tau}e^{\frac{-t}{\tau}}\hat{x}(t)$ and input $\frac{1}{\tau}e^{\frac{-t}{\tau}}u(t)$ . $\hat{x}(t)$, which is the estimate of the state variable $x(t)$, is calculated over a specific range of $x(t)$ with distortion. Intersections between dashed line and gray line in 6.10 where the dashed line has a lower slope are point attractors in the network. These points are dynamically attractive [22] because the value of $\hat{x}(t)$ tends to approach to them over time. For smaller inputs, $u(t)$, the effect of point attractors is more dominant (see equation 6.9). Besides, having more neurons in the network leads to a better estimation (i.e., $\hat{x}$ approaches $x$) and for a given value of the input, point attractors have less effect (see figure 6.10).

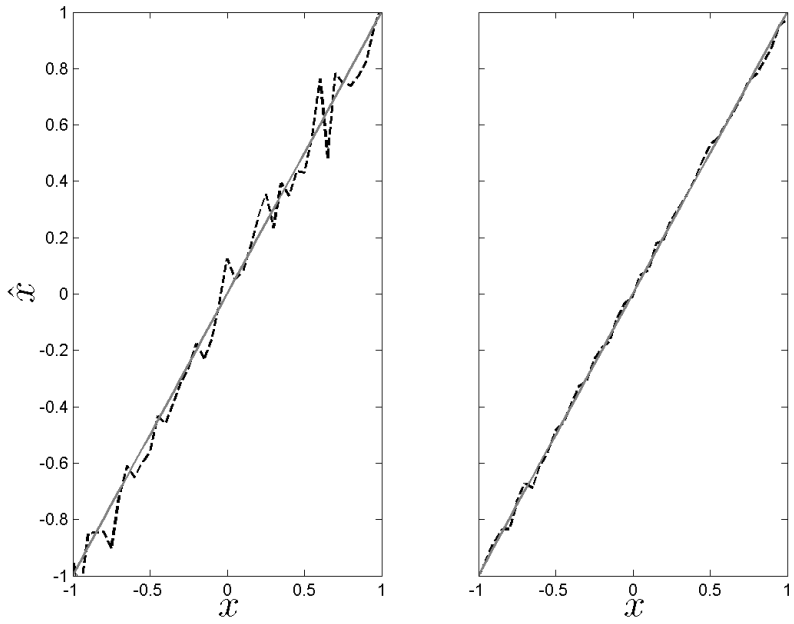


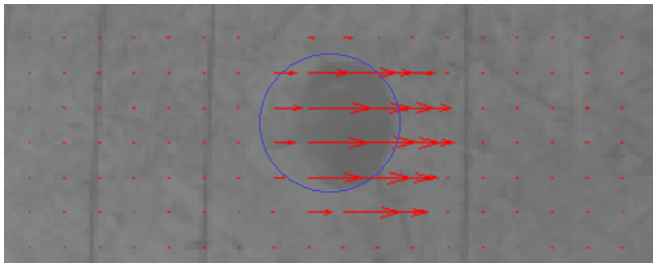

Figure 6.10: Distortion in state variable estimation from neural populations. Dash lines are neural estimates and gray solid lines are ideal estimation. Left, estimate from a population of 100 neurons; right, estimate from a population of 1000 neurons. More neurons in a population reduce distortion, and allow even small inputs to overcome point attractors.

### 6.3.2 Tracking a Pendulum with Churchland and Lisberger's SP Controller

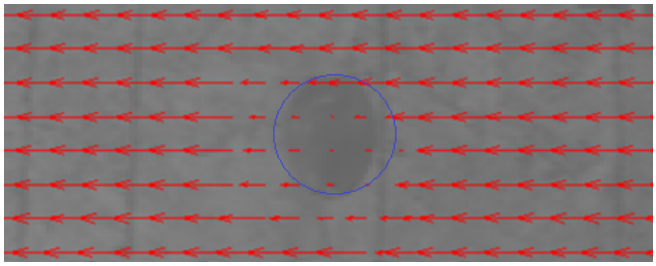
One of the main purposes of this thesis was to test whether the existing control models of smooth pursuit eye movement will operate in interaction with the real world. For this purpose, first, Churchland and Lisberger's controller (which is one of the best known smooth pursuit controllers) was used instead of the neural controller that was derived from it, to make sure that the possible limitations of the neural controller would not affect the experiment results.

For tracking, a pendulum (i.e., a ball attached to a rod) with the length of two meters was used as the target. Figure 6.11 depicts two frames captured from this pendulum during the time of tracking. Figure 6.11a shows the time in which the camera was stationary and pendulum had started to move. The optical flow components in the blue circle (which has a diameter equal to sigma of the Gaussian average (see section 6.2.2) were averaged and then fed to the controller. Figure 6.11b, which is three frames after 6.11a, shows that the camera started to track the pendulum and the optical flow components related to the pendulum became approximately zero, but background was moving relative to the camera.

Figure 6.12 shows the result of tracking the pendulum with a fairly small angular velocity. We can see that since the image of the pendulum did not go out of the centre, the camera tracked the pendulum fairly well. On the other hand, figure 6.13 shows that when the pendulum moved faster, the camera was not able to track the pendulum because the image of the target quickly went out of the the centre of the frame and the Gaussian average no longer averaged the optical flow components related to the target. This observation suggested that in order to smoothly track an object, there should be a mechanism to selectively concentrate on it. In other words, selective attention seems essential to maintain



(a)



(b)

Figure 6.11: Illustration of optical flow components during tracking of pendulum.

the smooth pursuit eye movement.

To test if adding selective attention would lead to a better performance, the target was tracked within the image using the MATLAB Computer Vision Library (specifically *HistogramBasedTracker*). This class found the portion of the image associated with the target in each frame. Then optical flow components on this part were averaged to extract the object velocity (instead of averaging on the centre). Figure 6.14 illustrates how the target was found in each frame.

Figure 6.15 depicts the result of tracking after the averaging on the centre was replaced with selective averaging. We can see that even though the pendulum movement was leading in time and its image was not at the centre (i.e., on the fovea), tracking was successful.

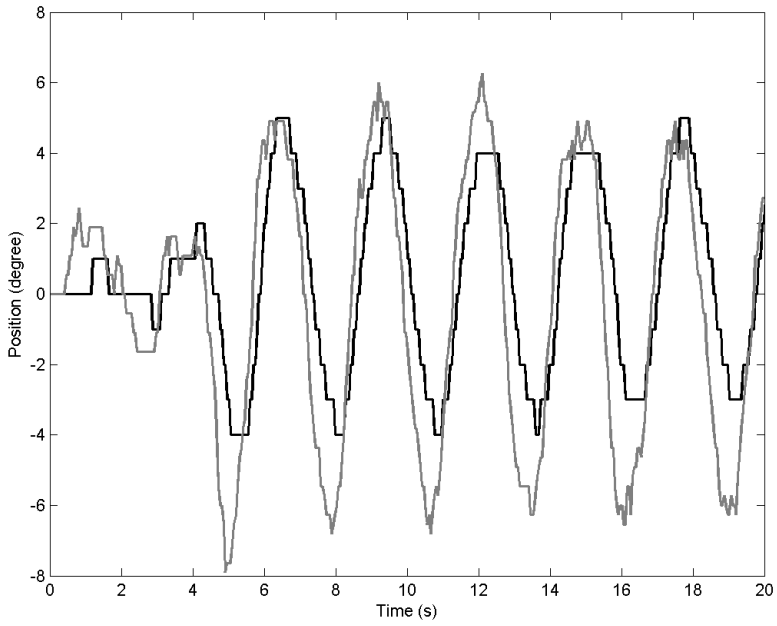



Figure 6.12: Result of tracking the pendulum by averaging on centre (small angular velocity). Black line: angular position of the camera over time. Gray line: angular position of the pendulum over time. Position of the camera obtained by saving the position of the servo in each frame during the course of tracking. Position of the pendulum was calculated by first finding the pixel distance between the centre of the ball and centre of the frame in each frame (MATLAB Computer Vision Library found the target and then the results checked manually) and then multiplying by the pixel-to-degree coefficient. If the image of the pendulum remained inside the Gaussian average, tracking was successful.

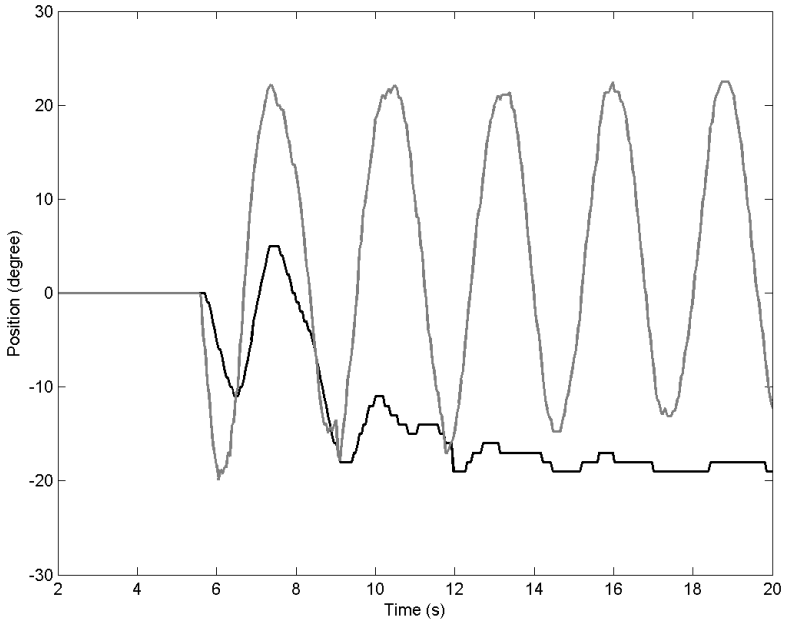



Figure 6.13: Result of tracking the pendulum by averaging on centre (larger angular velocity compared to figure 6.12). Black line: angular position of the camera over time. Gray line: angular position of the pendulum over time. Whenever the image of the pendulum went out of the Gaussian average, tracking failed.

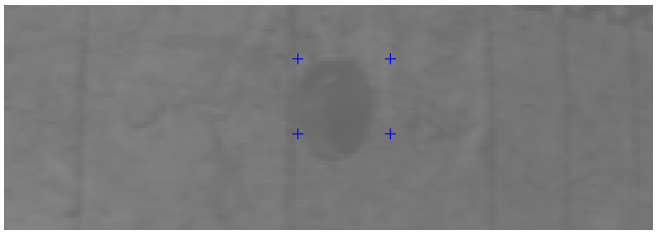


Figure 6.14: Illustration of using MATLAB Computer Vision Library to find the target in each frame.

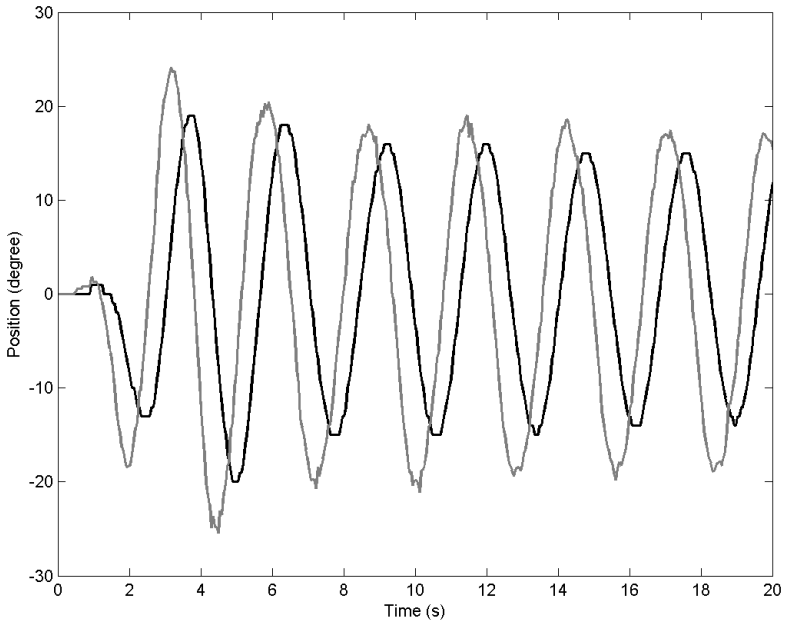



Figure 6.15: Result of tracking the pendulum by selective averaging. Black line: angular position of the camera over time. Gray line: angular position of the pendulum over time. Even if the image of the pendulum was not at centre for some frames, the tracking was successful.

### 6.3.3 Tracking a Pendulum with the Neural SP Controller

We saw in section 6.3.2 that adding selective attention led to robust tracking of the pendulum. To make sure that minor differences between the neural model and Churchland and Lisberger's control model would not affect this robust tracking, control model was replaced by the neural model. Figure 6.16 shows the result of tracking the pendulum using the neural SP controller along with selective attention. After replacing the models, tracking remained successful.

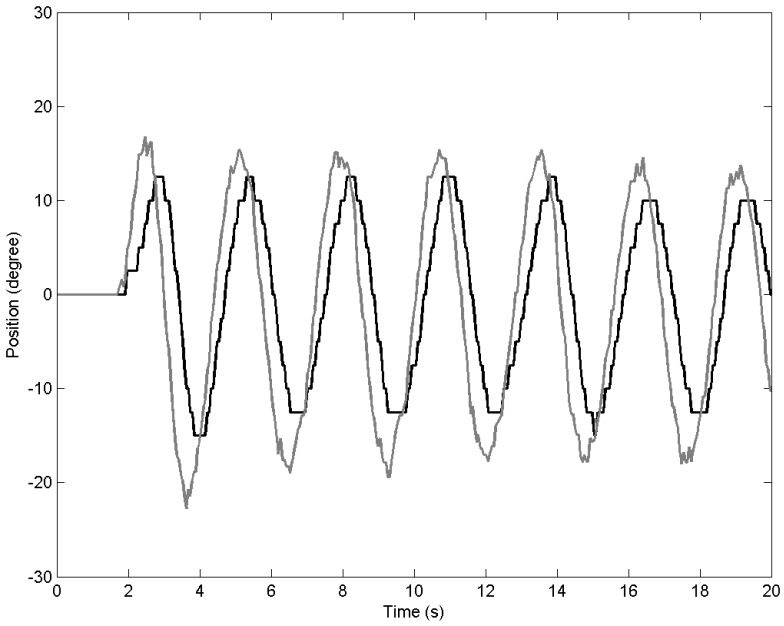



Figure 6.16: Result of tracking the pendulum with neural controller and using selective attention. Black line: angular position of the camera over time. Gray line: angular position of the pendulum over time. After replacing the control model by the neural model, tracking was still successful.

The results depended somewhat on the background texture. For these experiments three different backgrounds were used. A textured wooden background, a wrapping paper and a blackboard. The results which have been shown here were all related to the wooden background. Using the wrapping paper did not lead to acceptable results probably because it had a repeating pattern that made the optical flow of the background ambiguous. The results of the blackboard as the background were as good as wooden texture because the optical flow algorithm grouped the background with the foreground, so that a large region of the image appeared to have the target velocity.

# Chapter 7

# Discussion and Future Work

This thesis has presented two major contributions. First, it has demonstrated for the first time that the abstract control models of smooth pursuit control can be successfully implemented in spiking neurons. Second, it has shown that selective attention is essential for the smooth pursuit system to function properly. None of the previous control models of smooth pursuit directly accounted for attention in their models.

## 7.1 Discussion

The importance of attention in smooth pursuit has been indicated in behavioral experiments with humans [39]. Furthermore, studies have shown that lesions in middle temporal visual area (MT) cause retinotopic deficits in smooth pursuit, while lesions in medial superior temporal area (MST) cause direction deficits [21, 20]. This suggests an attention-mediated averaging of velocity over the target should occur in the projection from area MT to area MST.

The main purpose of mapping different components of the control model onto the brain areas was to indicate that spiking neurons are capable of performing the functions that have been attributed to the neurological system in control models. An ongoing debate still exists over the role of each part of the brain during smooth pursuit [45]. For example, in this thesis the role of the frontal eye fields (FEF) in smooth pursuit was considered to be encoding retinal slip acceleration associated with the target. However, FEF may contribute to target selection as well [25].

This thesis focused on smooth pursuit in horizontal direction. However, none of the components in the model (namely the neural controller) are limited to one direction and the model can be conveniently extended into two dimensions. An important concern is that the brain mechanisms and circuitry for horizontal and vertical pursuit are different [68, 29], and further work should be done to find the overlaps and differences between the two.

Adding selective attention to the smooth pursuit model led to more robust tracking. However, the model was not necessarily able to stabilize the image of the target on the centre (i.e., fovea), which is crucial for accurate vision. In general, even smoothly moving targets can evoke a combination of smooth and saccadic eye movements [46]. Therefore, adding corrective saccades to the model will lead to robust foveated tracking.

## 7.2 Future Work

This thesis can be continued in four major directions. First, corrective saccades can be added to the model for robust foveated tracking. Second, a vertical tracking component can be added to the system in order to make smooth tracking possible in all directions. Third, an advanced implementation of the motion energy model, which is able to solve the

aperture problem, can replace the existing optical flow method in order to have a model that corresponds more closely to the neurological system. Fourth, more detailed models of specific brain areas in the smooth pursuit neural controller can be developed. Also, a neural implementation of other brain areas can be added to the model in order to perform functions such as integration of eye velocity signal over time with spiking neurons.

# References


[1] Edward H Adelson and James R Bergen. Spatiotemporal energy models for the perception of motion. *J. Opt. Soc. Am. A*, 2(2):284–299, 1985.

[2] J. Alonso and Y. Chen. Receptive field. 4(1):5393, 2009.

[3] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM Computing Surveys (CSUR)*, 27(3):433–466, 1995.

[4] Richard Ernest Bellman and Kenneth L Cooke. Differential-difference equations. 1963.

[5] Richard T Born and David C Bradley. Structure and function of visual area mt. *Annu. Rev. Neurosci.*, 28:157–189, 2005.

[6] Gary R Bradski. Real time face and object tracking as a component of a perceptual user interface. In *Applications of Computer Vision, 1998. WACV'98. Proceedings., Fourth IEEE Workshop on*, pages 214–219. IEEE, 1998.

[7] Bill Christmas. Designing complex gabor filters. *November 16th*, 2007.

[8] Mark M Churchland and Stephen G Lisberger. Experimental and computational analysis of monkey smooth pursuit eye movements. *Journal of neurophysiology*, 86(2):741–759, 2001.

[9] JD Crawford, W Cadera, and T Vilis. Generation of torsional and vertical eye position signals by the interstitial nucleus of cajal. *Science*, 252(5012):1551–1553, 1991.

[10] PM Daniel and D Whitteridge. The representation of the visual field on the cerebral cortex in monkeys. *The Journal of physiology*, 159(2):203–221, 1961.

[11] John G Daugman et al. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Optical Society of America, Journal, A: Optics and Image Science*, 2(7):1160–1169, 1985.

[12] Peter Dayan, Laurence F Abbott, and L Abbott. Theoretical neuroscience: Computational and mathematical modeling of neural systems. 2001.

[13] Simon Denman, Vinod Chandran, and Sridha Sridharan. An adaptive optical flow technique for person tracking systems. *Pattern recognition letters*, 28(10):1232–1239, 2007.

[14] Alain Destexhe, Zachary F Mainen, and Terrence J Sejnowski. Kinetic models of synaptic transmission. *Methods in neuronal modeling*, 2:1–25, 1998.

[15] Allan Dobbins, Steven W Zucker, and Max S Cynader. Endstopped neurons in the visual cortex as a substrate for calculating curvature. *Nature*, 329(6138):438–441, 1987.

[16] Daniel D Doyle, Alan L Jennings, and Jonathan T Black. Optical flow background estimation for real-time pan/tilt camera object tracking. *Measurement*, 2013.

[17] Marius Drulea and Sergiu Nedevschi. Motion estimation using the correlation transform. 2013.

[18] R Dubner and SM Zeki. Response properties and receptive fields of cells in an anatomically defined region of the superior temporal sulcus in the monkey. *Brain research*, 35(2):528–532, 1971.

[19] Charles J Duffy and Robert H Wurtz. Sensitivity of mst neurons to optic flow stimuli. i. a continuum of response selectivity to large-field stimuli. *Journal of Neurophysiology*, 65(6):1329–1345, 1991.

[20] MR Dursteler and ROBERT H Wurtz. Pursuit and optokinetic deficits following chemical lesions of cortical areas mt and mst. *Journal of Neurophysiology*, 60(3):940–965, 1988.

[21] MR Dursteler, ROBERT H Wurtz, and WILLIAM T Newsome. Directional pursuit deficits following lesions of the foveal representation within the superior temporal sulcus of the macaque monkey. *Journal of Neurophysiology*, 57(5):1262–1287, 1987.

[22] Chris Eliasmith and C Charles H Anderson. *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT Press, 2004.

[23] Michael William Eysenck and Mark T Keane. *Cognitive psychology: A student's handbook*. Taylor & Francis, 2005.

[24] David H Hubel MD John Franklin et al. *Brain and Visual Perception: The Story of a 25-Year Collaboration: The Story of a 25-Year Collaboration*. Oxford University Press, 2004.

[25] Kikuro Fukushima. Frontal cortical control of smooth-pursuit. *Current opinion in neurobiology*, 13(6):647–654, 2003.

[26] Dennis Gabor. Theory of communication. part 1: The analysis of information. *Electrical Engineers-Part III: Radio and Communication Engineering, Journal of the Institution of*, 93(26):429–441, 1946.

[27] D Goldreich, RJ Krauzlis, and SG Lisberger. Effect of changing feedback delay on spontaneous oscillations in smooth pursuit eye movements of monkeys. *Journal of neurophysiology*, 67(3):625–638, 1992.

[28] Melvyn A Goodale and A David Milner. Separate visual pathways for perception and action. *Trends in neurosciences*, 15(1):20–25, 1992.

[29] Helena Grönqvist, Gustaf Gredebäck, and Claes von Hofsten. Developmental asymmetries between horizontal and vertical tracking. *Vision Research*, 46(11):1754–1761, 2006.

[30] David J Heeger. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1(4):279–302, 1988.

[31] Ellen C Hildreth and Shimon Ullman. The measurement of visual motion. 1982.

[32] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1):185–203, 1981.

[33] David H Hubel. *Eye, brain, and vision.* Scientific American Library/Scientific American Books, 1995.

[34] David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591, 1959.

[35] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[36] Quasar Jarosz. Neuorn hand-tuned.

[37] Renaud Jolivet, Felix Schürmann, Thomas K Berger, Richard Naud, Wulfram Gerstner, and Arnd Roth. The quantitative single-neuron modeling competition. *Biological cybernetics*, 99(4-5):417–426, 2008.

[38] Eric R Kandel, James H Schwartz, Thomas M Jessell, et al. *Principles of neural science*, volume 4. McGraw-Hill New York, 2000.

[39] Beena Khurana and Eileen Kowler. Shared attentional control of smooth eye movement and perception. *Vision research*, 27(9):1603–1618, 1987.

[40] Helga Kolb. Simple anatomy of the retina.

[41] Eileen Kowler. Eye movements: The past 25years. *Vision research*, 51(13):1457–1483, 2011.

[42] RJ Krauzlis and SG Lisberger. A control systems model of smooth pursuit eye movements with realistic emergent properties. *Neural Computation*, 1(1):116–122, 1989.

[43] Lieven Lagae, Hugo Maes, Steven Raiguel, DK Xiao, and GA Orban. Responses of macaque sts neurons to optic flow components: a comparison of areas mt and mst. *Journal of Neurophysiology*, 71(5):1597–1626, 1994.

[44] Rebekka Lencer and Peter Trillenberg. Neurophysiology and neuroanatomy of smooth pursuit in humans. *Brain and cognition*, 68(3):219–228, 2008.

[45] Stephen G Lisberger. Visual guidance of smooth-pursuit eye movements: sensation, action, and what happens in between. *Neuron*, 66(4):477–491, 2010.

[46] Stephen G Lisberger, EJ Morris, and L Tychsen. Visual motion processing and sensory-motor integration for smooth pursuit eye movements. *Annual review of neuroscience*, 10(1):97–129, 1987.

[47] Stephen G Lisberger and J Anthony Movshon. Visual motion analysis for pursuit eye movements in area mt of macaque monkeys. *The Journal of Neuroscience*, 19(6):2224–2246, 1999.

[48] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981.

[49] Henry Markram, Maria Toledo-Rodriguez, Yun Wang, Anirudh Gupta, Gilad Silberberg, and Caizhi Wu. Interneurons of the neocortical inhibitory system. *Nature Reviews Neuroscience*, 5(10):793–807, 2004.

[50] Julien Marzat, Yann Dumortier, André Ducrot, et al. Real-time dense and accurate parallel optical flow using cuda. In *7th International Conference WSCG*, 2009.

[51] John HR Maunsell and David 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 Comparative Neurology*, 266(4):535–555, 1987.

[52] Michael J Mustari, Seiji Ono, and Vallabh E Das. Signal processing and distribution in cortical-brainstem pathways for smooth pursuit eye movements. *Annals of the New York Academy of Sciences*, 1164(1):147–154, 2009.

[53] Kazuhiro Nakadai, Ken-ichi Hidai, Hiroshi Mizoguchi, Hiroshi G Okuno, and Hiroaki Kitano. Real-time auditory and visual multiple-object tracking for humanoids. In *Proceedings of the 17th international joint conference on Artificial intelligence-Volume 2*, pages 1425–1432. Morgan Kaufmann Publishers Inc., 2001.

[54] GA Orban, Lieven Lagae, A Verri, Steven Raiguel, D Xiao, Hugo Maes, and V Torre. First-order analysis of optical flow in monkey brain. *Proceedings of the National Academy of Sciences*, 89(7):2595–2599, 1992.

[55] Christopher Pack, Stephen Grossberg, and Ennio Mingolla. A neural model of smooth pursuit control and motion perception by cortical area mst. *Journal of Cognitive Neuroscience*, 13(1):102–120, 2001.

[56] Christopher C Pack and Richard T Born. Temporal dynamics of a neural solution to the aperture problem in visual area mt of macaque brain. *Nature*, 409(6823):1040–1042, 2001.

[57] Karl Pauwels and Marc M Van Hulle. Optic flow from unstable sequences through local velocity constancy maximization. *Image and Vision Computing*, 27(5):579–587, 2009.

[58] János Attila Perge. *Cortical mechanisms of visual motion integration*. Utrecht University, 2005.

[59] Nicholas J Priebe, Carlos R Cassanello, and Stephen G Lisberger. The neural representation of speed in macaque area mt/v5. *The Journal of neuroscience*, 23(13):5650–5661, 2003.

[60] C1 Rashbass. The relationship between saccadic and smooth tracking eye movements. *The Journal of Physiology*, 159(2):326, 1961.

[61] F. Raudies. Optic flow. 8(7):30724, 2013.

[62] Werner Reichardt. Evaluation of optical motion information by movement detectors. *Journal of Comparative Physiology A*, 161(4):533–547, 1987.

[63] Dario L Ringach. A tachometerfeedback model of smooth pursuit eye movements. *Biological cybernetics*, 73(6):561–568, 1995.

[64] D Ar Robinson, JL Gordon, and SE Gordon. A model of the smooth pursuit eye movement system. *Biological cybernetics*, 55(1):43–57, 1986.

[65] DA Robinson. The mechanics of human saccadic eye movement. *The Journal of physiology*, 174(2):245–264, 1964.

[66] DA Robinson. The mechanics of human smooth pursuit eye movement. *The Journal of Physiology*, 180(3):569, 1965.

[67] David A Robinson. Integrating with neurons. *Annual review of neuroscience*, 12(1):33–45, 1989.

[68] Klaus G Rottach, Ari Z Zivotofsky, Vallabh E Das, LEA Averbuch-Heller, ALFRED O DISCENNA, ANUCHIT POONYATHALANG, and R Leigh. Comparison of horizontal, vertical and diagonal smooth pursuit eye movements in normal human subjects. *Vision research*, 36(14):2189–2195, 1996.

[69] Hide-aki Saito, Masao Yukie, Keiji Tanaka, Kazuo Hikosaka, Yoshiro Fukada, and E Iwai. Integration of direction signals of image motion in the superior temporal sulcus of the macaque monkey. *The Journal of Neuroscience*, 6(1):145–157, 1986.

[70] H Sakata, H Shibutani, Y Ito, and K Tsurugai. Parietal cortical neurons responding to rotary movement of visual stimulus in space. *Experimental Brain Research*, 61(3):658–663, 1986.

[71] Hideo Sakata, Hidetoshi Shibutani, Kenji Kawano, and Thomas L Harrington. Neural mechanisms of space vision in the parietal association cortex of the monkey. *Vision research*, 25(3):453–463, 1985.

[72] Robert J Schafer and Tirin Moore. Selective attention from voluntary control of neurons in prefrontal cortex. *Science*, 332(6037):1568–1571, 2011.

[73] Jeffrey D Schall, Doug P Hanes, Kirk G Thompson, and Dana J King. Saccade target selection in frontal eye field of macaque. i. visual and premovement activation. *The Journal of Neuroscience*, 15(10):6905–6918, 1995.

[74] Behzad Shahraray and Michael K Brown. Robust depth estimation from optical flow. In *Computer Vision., Second International Conference on*, pages 641–650. IEEE, 1988.

[75] Abhishek Singh and Narendra Ahuja. Exploiting ramp structures for improving optical flow estimation. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2504–2507. IEEE, 2012.

[76] Ritesh Singh, Manali Godse, and Tanaji Biradar. Video object tracking using particle filtering. *International Journal of Engineering Research and Technology*, 2(9):2987–2093, 2013.

[77] Masaki Tanaka and Stephen G Lisberger. Enhancement of multiple components of pursuit eye movement by microstimulation in the arcuate frontal pursuit area in monkeys. *Journal of neurophysiology*, 87(2):802–818, 2002.

[78] Michael Tao, Jiamin Bai, Pushmeet Kohli, and Sylvain Paris. Simpleflow: A non-iterative, sublinear optical flow algorithm. In *Computer Graphics Forum*, volume 31, pages 345–353. Wiley Online Library, 2012.

[79] Peter Thier and Uwe J Ilg. The neural basis of smooth-pursuit eye movements. *Current opinion in neurobiology*, 15(6):645–652, 2005.

[80] Bryan P Tripp and Chris Eliasmith. Population models of temporal differentiation. *Neural computation*, 22(3):621–659, 2010.

[81] Leslie G Ungerleider and Robert Desimone. Cortical connections of visual area mt in the macaque. *Journal of Comparative Neurology*, 248(2):190–222, 1986.

[82] LG Ungerleider and M Mishkin. Analysis of visual behavior, eds ingle dj, goodale ma, mansfield rjw, 1982.

[83] Brian A Wandell. *Foundations of vision.* Sinauer Associates, 1995.

[84] HongJiang Zhang, John YA Wang, and Yucel Altunbasak. Content-based video retrieval and compression: A unified solution. In *Image Processing, 1997. Proceedings., International Conference on*, volume 1, pages 13–16. IEEE, 1997.