Technische Universiteit Eindhoven University of Technology

Department of Mechanical Engineering
Dynamics and Control Group

Effect of Noise on Training a Network of Electronic Neurons

Internship Report

Jian Jin

DC 2017.021

Supervisor:
Prof. Henk Nijmeijer
Coach:
Dr. Erik Steur

Eindhoven, January 2017

Abstract

Complex networks are ubiquitous in our world. Synchronization in complex networked systems are especially interesting, with examples from nature to our society. From an engineering point of view, how to design coupling functions and network structures that lead to (optimal) synchronization of interconnected systems are of crucial importance. Such a design paradigm is called controlled synchronization paradigm, and we try to incorporate it into the spiking neural networks used for an autonomous navigation task. The reason for inclusion of this paradigm is that the spiking neural network has electronic Hindmarsh-Rose (HR) neurons as its components, which are heterogeneous in parameters and have different input-out mapping behaviours. This may result in a bias when the robot is advancing without any detection of obstacles. To train the sensor neurons to produce same outputs with respect to same inputs, Vromen et al. [1] considers clusters of neurons instead of single neurons to self-adjust each cluster's collective frequency and develops a training algorithm that can achieve frequency synchronization of all clusters. The work of this internship is to introduce this algorithm and investigate its robustness with respect to noise.

The first part of this report introduces the Hindmarsh-Rose neuron and discusses practical synchronization of HR neurons that interact via diffusive coupling. Simulations are presented to show that heterogeneous neurons can reach practical synchronization by strong coupling and their collective frequency can be influenced by changing the mutual adaptation parameter.

The second part of the report introduces the training scheme developed by Vromen et al. [1] and investigates its robustness with respect to noise. This training algorithm is first increasing the overall coupling gains to practically synchronize all neurons in the cluster, then adjusting the mutual adaptation parameters to coordinate the collective frequency. Based on the training scheme by Vromen et al. [1], this report investigates effect of channel noise on the algorithm and the limitations of it. In particular, for fixed noise power the algorithm will fail to train a cluster into a collective reference frequency with large number of neurons in that cluster. Besides, the more neurons to train, the less noise power tolerable for training.

The third part of the report studies the effect of noise on synchronization of diffusively coupled HR neurons in undirected netowrks. It is investigated for each type of subgraph the relation between coupling gain and practical synchronization error when the noise power is fixed. The spectral properties of each subgraph are computed to correlate with their ability of robustness to noise.

Preface

This report presents the result of my internship at Dynamics and Control Group in Technische Universiteit Eindhoven. The main task is to reproduce the work of Vromen et al's papaer [1] and study the effect of noise on it. I would like to thank Dr. Steur for spending uncountable time in supervising and advising me. If I would further my study in synchronization, Dr. Steur is the one who leads me through this field. I am also grateful to Prof. Nijmeijer for the fruitful discussions in our monthly meetings. Prof. Nijmeijer has encouraged me to take this project during our first meeting and enlightens me to gain a systematic approach towards possible challenges in this project. I would also like to thank my fellow students and friends for their company.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter the notion of synchronization in networks is introduced along with their engineering applications and real-world examples. We also give a brief literature review on neuronal networks for autonomous navigation and introduce the spiking neural networks used for the navigation problem. In the end the structure of the report is presented.

1.1 Synchronization in Complex Networked Systems

Complex networked systems are ubiquitous in our world. They consist of interacting agents communicating by some coordination laws. Typical examples include food web [2], metabolic networks [3], World Wide Web [4], etc. To mathematically model these complex dynamics three essential ingredients are needed: (i) a dynamical description of the evolution of each agent in the network; (ii) coupling laws used by each agent to coordinate with each other; (iii) network topology describing the connectivity relations between all agents (see for example [5]). The reason for combining all these aspects is that real world networks often evolve its topology in time, according to specific dynamical rules; each node is also a evolving subsystem, possibly *heterogeneous* with different dynamics and the coupling laws may also be time-dependent. A notable example is the neuronal network in our brain, which is known to adapt and evolve their structure as well as the strength of their synaptic connections to perform different functions [6].

Synchronization is a kind of collective behaviour in networked systems. It can be defined as a process in which "events" keep happening simultaneously for an extended period of time [7]. The scientific interest in synchronization dates back to Christiaan Huygen's work *"an odd kind of sympathy"* between pendulum clocks, cf. [8]. From then on, synchronous behavior among coupled dynamical systems has been focal interest and keeps being investigated, from fireflies synchronous flashing in Amazonia [9], synchronized motion of bird flocks and fish schools [10], to circadian rhythms with 24-hour day-night cycle [11] and synchronized pacemaker neurons regulating our heartbeats [12]. The state where all agents show simultaneous behaviour without any difference is called *full synchrony* and most of the aforementioned phenomena are *periodic synchrony* where various limit-cycle systems have the same period (frequency synchronization) or even the same phase (*phase-locking* [8]). In case of full synchrony, interactions where oscillators influence each other are called *mutual interaction* and interactions where one subsystem dominates the other are called *master-slave* configuration. Both types may subject to external forces (*external synchronization*) which may lead to bifurcation transitions of nonlinear systems [13] or adjust their periodic motions [14].

The topic "*synchronizability*" is mainly studied in physics community, investigating how structure properties (weight distribution, assortativity, connectivity distribution, etc.) will influence the whole system's ability to synchronize [15, 16, 17]. In these contexts the systems are often large-scale random graphs (scale-free, small-world, etc. [18]) and a large amount of analysis are based on the *Master Stability Function* (MSF) approach [19], which provides conditions for local stability of the linear variational synchronization manifold.

Applications of synchronization technique can be found in myriad domains. Examples are parallel image processing [20, 21], secure communication [19, 22], pattern recognition [23], etc. More engineering applications are synchronization of robot manipulators [24], power grid networks [25], unmanned aerial vehicles (UAVs) [26], consensus of mobile agents [27] and so on. When considering these applications not only should we guarantee synchronization of complex networks in analysis but also design coupling functions and structures, which we refer as *controlled synchronization*. Related works are discussed in, for instance, [28, 29].

1.2 Autonomous Navigation and Neuronal Networks

Autonomous navigation of mobile robots has a wide range of applications, from multivehicle cooperative driving (platoon) [30], multi-agent mobile robots in warehouses [31] to ummanned exploration of dangerous regions [32]. Roughly speaking, autonomous navigation can make use of the complete knowledge of environment and apply path planning. However, such detailed information is often not available in real world. For collision-avoidance mobile robots, a more feasible solution is to map sensory inputs to desirable motor actions [33].

Neural networks are a computing system made up of a number of interconnected nodes, which process information by their dynamic state response to external inputs [34]. The third generation of neural networks is spiking neural networks (SNNs). The spiking neurons use pulse coding to incorporate spatial-temporal information in communication and computation [35], like real neurons do, making them a good candidate for control of autonomous robots in real-time environment. Here we make a distinction between neural networks and *neuronal networks*. The former consists of "artificial" nodes that are static input-output maps connected via weighted coupling, while the latter consists of nodes that are dynamical systems, like SNNs.

Multiple examples of neuronal networks for control of obstacle-avoidance mobile robots can be found in literature. For instance, Floreano et al. [36] proposed using integrate-and-fire (IAF) neurons to construct neuronal networks. In [37] FPGA approach was used to implement spiking neural networks on real-time mobile robots. Wang et al. [38] use Hebbian learning algorithms to train a three-layer SNN for the behavior controller. Johnston et al. [39] presented an evolving spiking neural network (eSNN) paradigm that consists of Spike Time Dependent Plasticity (STDP) mechanism and Genetic Algorithm (GA) to implement on FPGA.

1.3 Spiking Neural Networks for Control of a Mobile Robot

The task of the robot is to maneuver an (unknown) environment and avoid collisions with all objects. The mobile robot used in this project is the e-puck mobile robot [40], which is shown in Figure 1.1.

The e-puck has translational velocity $v(t)$ and steering velocity $\omega(t)$, which defines the position and orientation of robot in a two-dimensional Cartesian space (x, y) with origin O:

$$\begin{aligned} \dot{x}(t) &= v(t)\cos(\theta(t)), \\ \dot{y}(t) &= v(t)\sin(\theta(t)), \\ \dot{\theta}(t) &= \omega(t). \end{aligned} \tag{1.1}$$

Denote ω_l and ω_r the angular velocities of the left and right wheel, b half the distance between the wheels and r the radius of the wheels. Then the translational velocity $v(t)$ and steering velocity $\omega(t)$ satisfy

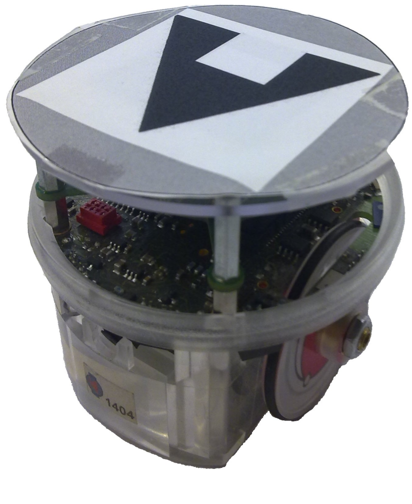

Figure 1.1: The e-puck mobile robot.

$$\begin{aligned} v(t) &= \frac{r}{2}(\omega_r(t) + \omega_l(t)), \\ \omega(t) &= \frac{r}{2b}(\omega_r(t) - \omega_l(t)). \end{aligned} \tag{1.2}$$

The angular velocities $\omega_r(t)$ and $\omega_l(t)$ are determined by the output of two motor neurons. Note because of a non-slip condition (non-holonomic constraint, c.f. [41]) on the wheels, the robot cannot move sideways, i.e. perpendicular to the translational direction.

The e-puck is equipped with nine sensors to detect distances to obstacles in each direction. All 9 sensors can then be divided into three groups S_{g1}, S_{g2}, S_{g3}, where sensor in group 2 detects distance to obstacles in the translational direction, sensors in group 1 and 3 detect obstacles to the robot's left and right direction. The distance to an object measured by sensor k in group j is denoted by s_{jk}. The maximal and minimal detectable distance is denoted by x_{max} and x_{min} respectively. An illustration of such sensor layouts are shown in Figure 1.2. The distance to an obstacle detected by sensor group j is given by

$$d_{gj} = \min_k(s_{jk}). \tag{1.3}$$

The neuronal controller we choose uses a two-layer network structure as shown in Figure 1.3, which is a simplified version from Wang et al. [38]. The nodes in the input layer (sensor neurons) receive external inputs I_{gj}, $j = 1, 2, 3$, which are chosen to be (exponentially) proportional to the distances d_{gj}, according to the (saturated) function

$$I_{gj} = \begin{cases} I_{\min}, & \text{if } d_{gj} \leq x_{\min}, \\ I_{\min}\left(\frac{I_{\max}}{I_{\min}}\right)^{\frac{d_{gj}-x_{\min}}{x_{\max}-x_{\min}}}, & \text{if } x_{\min} < d_{gj} < x_{\max}, \\ I_{\max}, & \text{if } d_{gj} \geq x_{\max}. \end{cases} \tag{1.4}$$

Here I_{max} and I_{min} denote the maximum and minimum input voltage. The outputs of the neuronal controller are the angular velocities ω_l and ω_r, which are proportional to the firing rate of motor neuron M_{n1} and M_{n2} respectively. For simplicity, we only assume positive angular velocities. This means that when detection of an obstacle right in front of the robot, it will stand still instead of turning around its center. Motor neuron M_{n1} and M_{n2} receives both excitatory

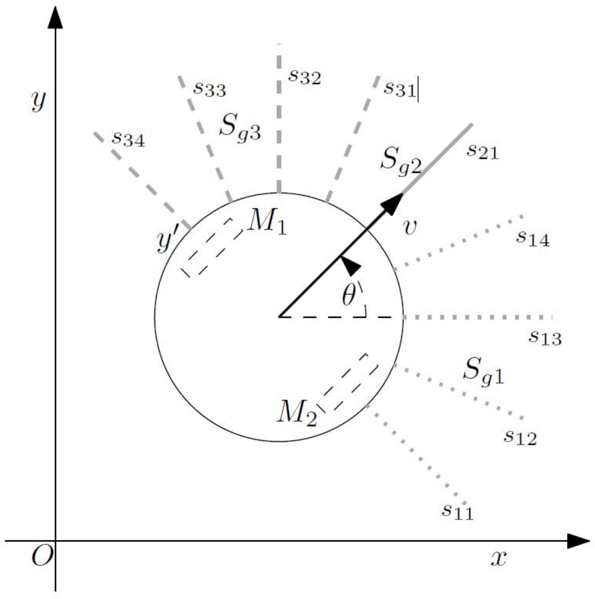

Figure 1.2: (adopted from [1]) The e-puck and sensor layout. The dashed rectangles are the wheels of the robot.

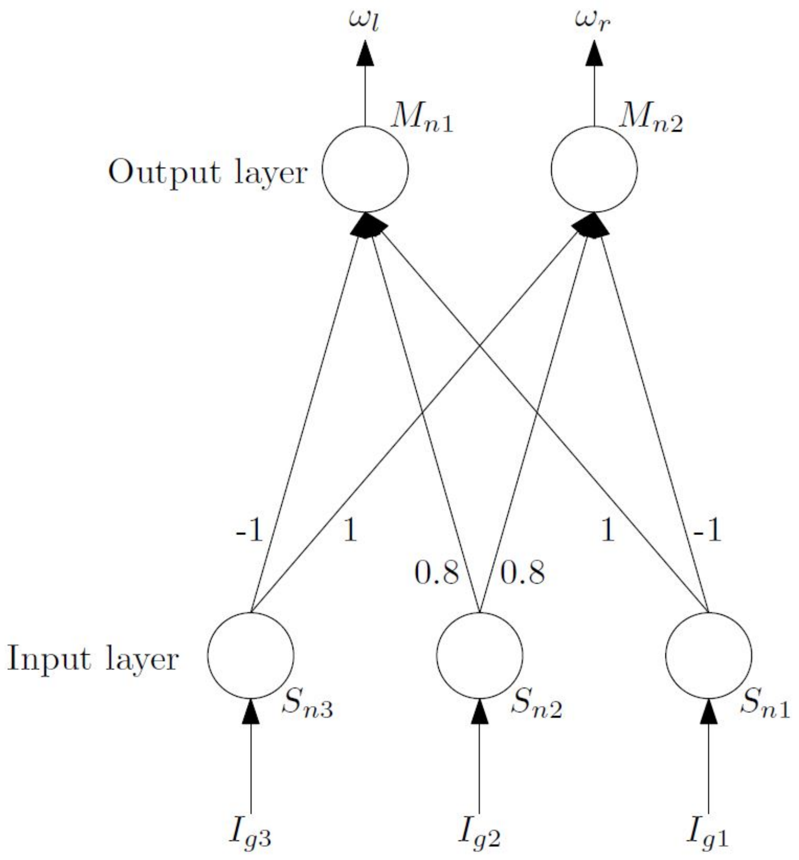

Figure 1.3: Network structure of neuronal controller, adopted from [1].

and inhibitary inputs from corresponding sensor neurons. The activating potential $U_i(t), i = 1, 2$, of the motor neurons is determined by the equation

$$U_i(t) = \sum_{j=1}^{3} \sum_{N_j^f} W_{Sn}(i,j)\psi(t - t_j^f). \tag{1.5}$$

with t_j^f the spike emitting time of the jth sensor neuron in a pre-defined time window and N_j^f the number of spikes in that window. According to Figure 1.3, the coupling gain matrix W_{Sn} from input to output layer is given by

$$W_{Sn} = \begin{bmatrix} 1 & 0.8 & -1 \\ -1 & 0.8 & 1 \end{bmatrix}. \tag{1.6}$$

The function $\psi(\cdot)$ is defined by

$$\psi(s) = \frac{s}{\tau_s}\exp(-\frac{s}{\tau_s}). \tag{1.7}$$

with τ_s a time constant. Here ψ serves as an "exponential forgetting" function to make recent spikes have a larger impact on increasing the activating potential than previous spikes. The generation of spikes for the motor neuron is determined by the membrane potential $V_i(t)$ which depends on the activating potential $U_i(t)$. Their relation is defined by

$$I_{gj} = \begin{cases} 2, & \text{if } U_i(t) \geq V_M, \\ U_i(t), & \text{if } U_i(t) \geq V_{\text{rest}} \text{ and } t - t_i^f \geq \delta_t, \\ V_{\text{rest}}, & \text{otherwise.} \end{cases} \tag{1.8}$$

where V_{rest} is the resting potential, t_i^f the spike emitting time of motor neuron i and δ_t the refractory period (the minimum time between two successive spikes). A spike is emitted if $V_i(t) = 2$ and the number of spikes n_i in a predefined time window is proportional to the translational or angular velocities of the wheels.

After introducing SNN controller, we stress that the nodes of the neuronal controller are *heterogeneous* Hindmarsh-Rose neurons. Because the input-out mapping of each sensor neuron is different, this may lead to a bias when the robot is advancing without any detection of obstacles. To eliminate such bias, three sensor neurons should produce same outputs with respect to same inputs, i.e. they are frequency synchronized. A solution developed by Vromen et al. [1] is to consider clusters of synchronized neurons instead of single neurons, which can self-adjust each cluster's own collective frequency to reach frequency synchrony. This motivates the goals of the training algorithms:

(i) if an obstacle is detected on the left, or if an obstacle on the left is closer to the robot than on the right, then the input to motor neuron M_{n1} is larger than the input to M_{n2} and the robot makes a turn to the right.

(ii) the robot will move straight ahead ($M_{n1} = M_{n2}$) if no obstacle is detected by any sensor neuron or if both S_{n1} and S_{n3} detect obstacles of the same distance to each side.

1.4 Contributions and Outline

This internship report introduces the training algorithms of spiking neural networks for autonomous navigation developed by Vromen et al. [1] and investigates the effect of noise on it. In Vromen's paper it is found that when training more than seven neurons the algorithm will not be accurate. This report proposed that training failure is due to channel noise. Another focus of this report is to study the effect of noise on synchronizing undirected networks of diffusively coupled HR neurons.

In Chapter 1, a brief literature review is given on synchronization of complex networks and neuronal networks for autonomous navigation. The problem of this internship is also formulated with a proposed neuronal controller.

Chapter 2 introduces the Hindmarsh-Rose neuron and discusses practical synchronization of them via diffusive coupling. Simulations are presented to show that heterogeneous neurons can reach practical synchronization by strong coupling and their collective frequency can be influenced by changing the mutual adaptation parameter.

Chapter 3 discusses the training scheme developed by Vromen et al. [1] and investigates the effect of noise on it. To frequency synchronize three sensor clusters, the algorithm first increases the overall coupling gains to practically synchronize all neurons in the cluster, then adjusts the mutual adaptation parameters to coordinate each cluster's frequency. Based on Vromen's algorithm, we add channel noise and investigate the effects of different noise power and different number of trained neurons. It is found that the more neurons to train, the less noise power tolerable for effective training.

Chapter 4 studies the effect of noise on synchronization of diffusively coupled HR neurons in some typical undirected networks. We investigate for each type of subgraph the relation between coupling gain and practical synchronization error when the noise power is fixed. The spectral properties of each subgraph is computed and it is found that the eigenratio $\frac{\lambda_2}{\lambda_n}$ is closely related to noise resistance capability of diffusively coupled systems.

Chapter 2

Synchronization of Hindmarsh-Rose Neurons

In this chapter we will introduce some important concepts and properties of neurons in computational neuroscience. We focus on the Hindmarsh-Rose model and introduce our modified model. Mathematical representations of diffusively coupled HR neurons are discussed. In the last section we present some simulation results regarding two coupled HR neurons.

2.1 Introduction

Single neurons are important functional units for the computational properties of the brain [42]. The most important physical variable in neural computation is the neuron's membrane potential, which can rapidly change over time [43]. Neurons can influence other connected neuron's membrane potential by *synapse*, a structure allowing neurons to transmit information. There are two types of synapses: *chemical* and *electrical* synapses. At the chemical synapses, the pre-synaptic neuron releases neurotransmitters and induce a current at the post-synaptic neuron. At the electrical synapses (also called *gap junctions*), there is a direct high conductance pathway connecting pre- and post-synaptic neurons [44]. According to [45], electrical synapses play an important role in synchronization of individual neurons.

Three types of neuron activities are often distinguished. (i) *resting*: the membrane potential is constant over time; (ii) *tonic spiking*: the neuron can fire *action potentials* at a constant rate. An action potential is an electrical impulse characterized by a rapid increase of the neuron's membrane potential followed by a sudden drop to a lower level; (iii) *bursting*: the neuron repeatedly fires discrete groups of spikes, each followed by a period of quiescence before the next burst occurs [46]. These distinct activities can be controlled by applying a certain input current, e.g. an external clamping current (see [43] for details) or other neuron activities.

To describe the neuron dynamics different models have been developed, from which the most important ones are models of Hodgkin-Huxley [47], Morris-Lecar [48], FitzHugh and Nagumo [49, 50], Hindmarsh and Rose [51]. Despite the differences in the range of behavior that these models are capable to produce, each of these models has finite amount of free energy (also called semipassive, c.f. [52]).

2.2 Hindmarsh-Rose Model

In 1982 Hindmarsh and Rose proposed a two-dimensional neuron model [53] based on the FitzHuge-Nagumo model [49]. In 1984 they made another modification on the *1982 model* and resulted in the so-called *1984 model* [51], which is given by the following equations

$$\begin{aligned}\dot{x} &= y - x^3 + 3x^2 + I(t),\\ \dot{y} &= 1 - 5x^2 - y.\end{aligned} \tag{2.1}$$

In order to mimic neuron's ability to produce *firing frequency adaption* [43](e.g. the cell *Lymnaea* does not fire with a steady frequency, but the firing slows down and is finally terminated) and bursting modes, a third state is added to (2.1). The full set of equations is now given by

$$\begin{aligned}\dot{x} &= y - x^3 + 3x^2 + I(t),\\ \dot{y} &= 1 - 5x^2 - y,\\ \dot{z} &= r\left(s(x - x_0) - z\right),\end{aligned} \tag{2.2}$$

with r, s positive constant parameters and x_0 the stable equilibrium point of state x when no input is applied. These equations are often used to study the synchronization patterns of coupled oscillators, see e.g. [54]. However, throughout this report a modification of the 1984 model will be made. This modified model comes from an affine coordinate transformation concerning the x-state of (2.2) (c.f. [55]) and an additional linear coordinate transformation [55] The reason for this modification is to avoid saturation of signals in the opetational amplifiers. The modified H-R models are now given by

$$\begin{aligned}\dot{y} &= -c_1 y^3 + c_2 y^2 + c_3 y + c_4 z_1 - c_5 z_2 - c_6 + c_7 I,\\ \dot{z}_1 &= -c_8 y^2 - c_9 y - c_{10} z_1,\\ \dot{z}_2 &= c_{11}\left(c_{12} y + c_{13} - z_2\right).\end{aligned} \tag{2.3}$$

where $\dot{} := \frac{\mathrm{d}}{\mathrm{d}t^*}$, $t^* = 1000t$ with t the time in seconds, y denoting the membrane potential of a neuron, which also serves as the natural output of the neuron, z_1, z_2 are internal variables and I the external input. The time scaling factor $\frac{t^*}{t} = 1000$ is in accordance with the electronic realization of the HR model as presented below. The parameters $c_{i|i=1,2,\cdots 13} \in \mathbb{R}_{\geq 0}$ are given by

$$\begin{aligned}&c_1 = 1, c_2 = 0, c_3 = 3, c_4 = 5, c_5 = 1, c_6 = 8, c_7 = 1,\\ &c_8 = 1, c_9 = 2, c_{10} = 1, c_{11} = 0.005, c_{12} = 4, c_{13} = 4.472.\end{aligned}$$

2.3 Simulations of Electronic HR Neuron

To construct more biological plausible neuronal controller introduced in Section 1.3, we use electronic Hindmarsh-Rose model neurons as nodes of the controller. The electronic circuit board realization of the HR neuron is shown in Figure 2.1. It consists of three integrating circuits, which integrate the three states of the HR model (2.3) and two multiplier circuits that generate the squared and cubic terms of the y-state. The parameters of 15 electronic HR neurons have been identified by Neefs [56] using extended Kalman filter. The table of these parameters can be found in Appendix B. Our simulation results are based on this table.

We first plot the output responses of three different electronic HR neurons, each with external input $I = 4.5$ $[V]$. The result is shown in Figure 2.2.

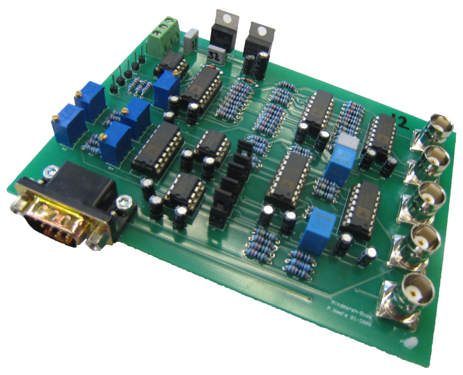
Figure 2.1: Electronic HR model neuron.

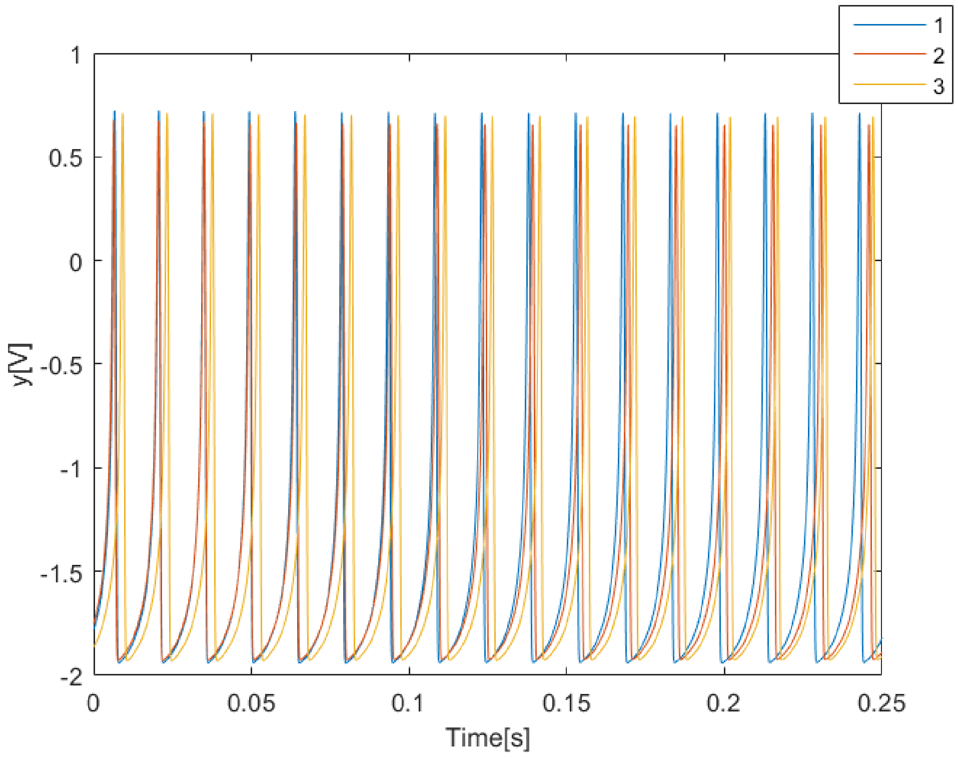

Figure 2.2: Responses of three different electronic HR neurons with external input $I = 4.5$ $[V]$.

We can see that all three neurons are tonic spiking. However, because of their parameter mismatch, they don't have exactly same output response. In fact, according to the experimental results of [56], when the external input is approximatly between 4.5 $[V]$ and 10 $[V]$, electronic HR neurons are tonic spiking with a fixed period time. This relation is illustrated in Figure 2.3.

As mentioned in Section 1.3, we want to practically synchronize heterogeneous neurons in each cluster to form a collective frequency. Here we introduce the definition of practical synchronization.

Definition 2.1. (Practical synchronization, [57]). Consider k systems (A.9) with output $y_i(t) \in \mathbb{R}^m$ defined on an interval $[t_0, t_2)$. The interconnected systems are said to *practically synchronized* with bound ϵ if there is a $t_1(\epsilon)$, $t_0 \leq t_1(\epsilon) < t_2$, such that $|y_i(t) - y_j(t)| < \epsilon$ for all $i, j \in \mathcal{V}$ and $t \in [t_1, t_2)$.

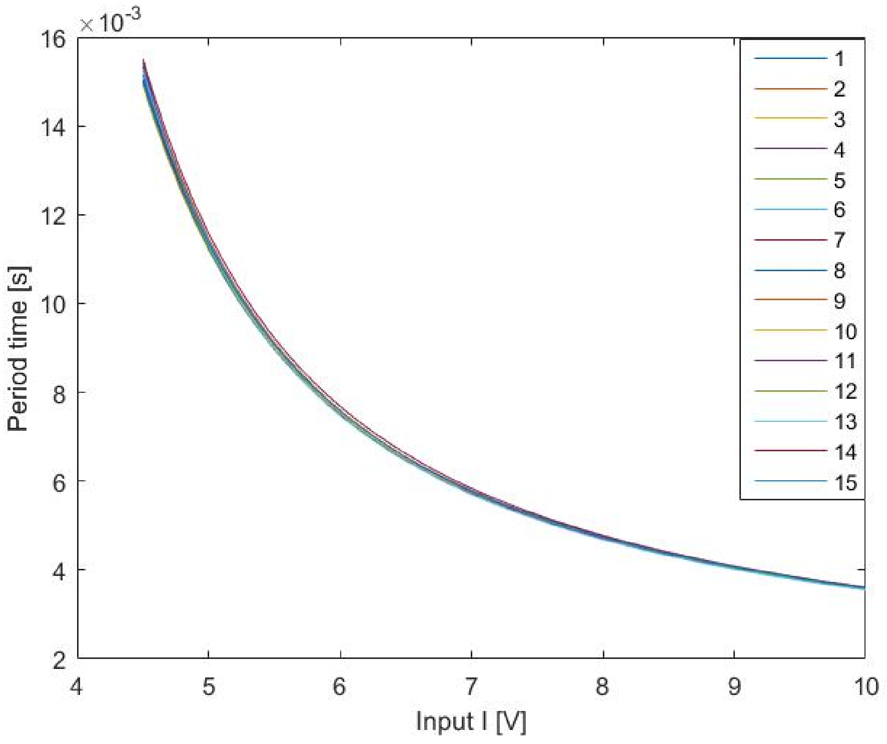

Figure 2.3: Period of all 15 HR neurons as function of external input I.

For convenience we fix the value ϵ and refer to practical synchronization with bound ϵ as practical synchronization. A feasible type of interaction that can lead to neuronal synchronization is *diffusive coupling*, which is a coupling defined as the weighted difference of the output responses of coupled systems. The mathematical representation of diffusively coupled HR neurons is shown in equation (2.4):

$$\begin{aligned}\dot{y}_i &= -c_{i,1}y_i^3 + c_{i,2}y_i^2 + c_{i,3}y_i + c_{i,4}z_{i,1} - c_{i,5}z_{i,2} - c_{i,6} + c_{i,7}I_i + u_i,\\ \dot{z}_{i,1} &= -c_{i,8}y_i^2 - c_{i,9}y_i - c_{i,10}z_{i,1},\\ \dot{z}_{i,2} &= c_{i,11}\left(c_{i,12}y_i + c_{i,13} - z_{i,2}\right),\end{aligned} \tag{2.4}$$

where the term u_i is the diffusive coupling term. It is the weighted output error between connected systems, defined as

$$u_i = -\sum_{j=1,i\neq j}^{n} \gamma_{ij}\left(y_i - y_j\right), \tag{2.5}$$

with γ_{ij} representing the directed coupling strength from neuron j to i. The whole network of neurons can also be expressed in matrix form shown in equation (2.6):

$$\begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix} = -\Gamma \begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix} = -\begin{pmatrix} \sum_{j=2}^{k}\gamma_{1j} & -\gamma_{12} & \cdots & -\gamma_{1k} \\ -\gamma_{21} & \sum_{j=1,j\neq 2}^{k}\gamma_{2j} & \cdots & -\gamma_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ -\gamma_{k1} & -\gamma_{k2} & \cdots & \sum_{j=1}^{k-1}\gamma_{kj} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix}. \tag{2.6}$$

Note that Γ is the Laplacian matrix of the network, which can characterize the complete connectivity and coupling strength relations in a graph. For sufficiently strong couplings between

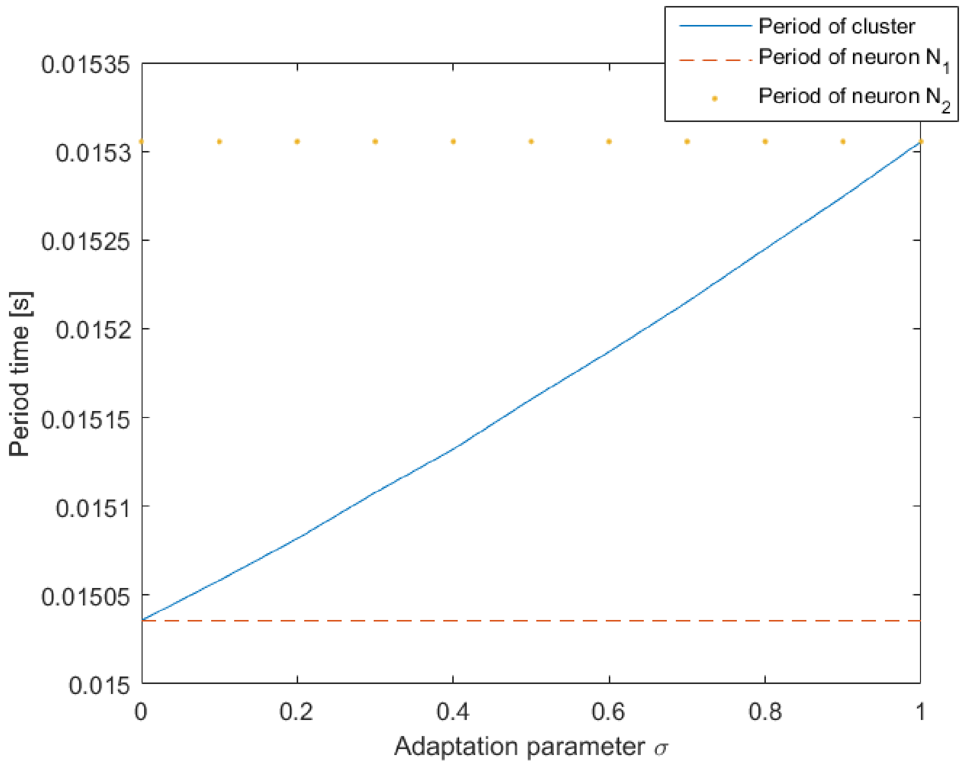

Figure 2.4: Relation between adaptation parameter and collective period.

heterogeneous oscillators, it is proven in [58] that practical synchronization can be achieved (for details see Appendix A).

After achieving practical synchronization of a cluster, we are also interested in how to adjust the collective frequency of it. We start by considering two coupled HR neurons in equation (2.7):

$$\begin{aligned} u_1 &= \gamma\sigma(y_2 - y_1), \\ u_2 &= \gamma(1-\sigma)(y_1 - y_2). \end{aligned} \tag{2.7}$$

The reason for introducing both overall coupling gain γ and mutual adaptation parameter σ is that based on the truth that sufficiently large γ will always lead to practical synchronization of two coupled neurons (with $\sigma = \frac{1}{2}$ initially), varying σ in range $[0, 1]$ will still preserve neurons' synchronization while at the same time changes the collective frequency [1]. Such a relation between σ and the collective period of the cluster is shown is shown in Figure 2.4:

From Figure 2.4 we can see that the relation between mutual adaptation parameter σ and collective period are monotonic and approximately linear. The end points of y axis corresponds to $\sigma = 0$ or $\sigma = 1$. These two extreme cases are called master-slave configuration, which means that one neuron is totally enslaved by the other. The simulation results of master-slave configuration are shown in Figure 2.5. The results in Figure 2.4 also imply that parameter σ is an interpolation parameter that allows the collective period of two coupled neurons to be anywhere between the period times of the uncoupled neurons. This relation can be used as the principle for training two couples neurons into a desired frequency.

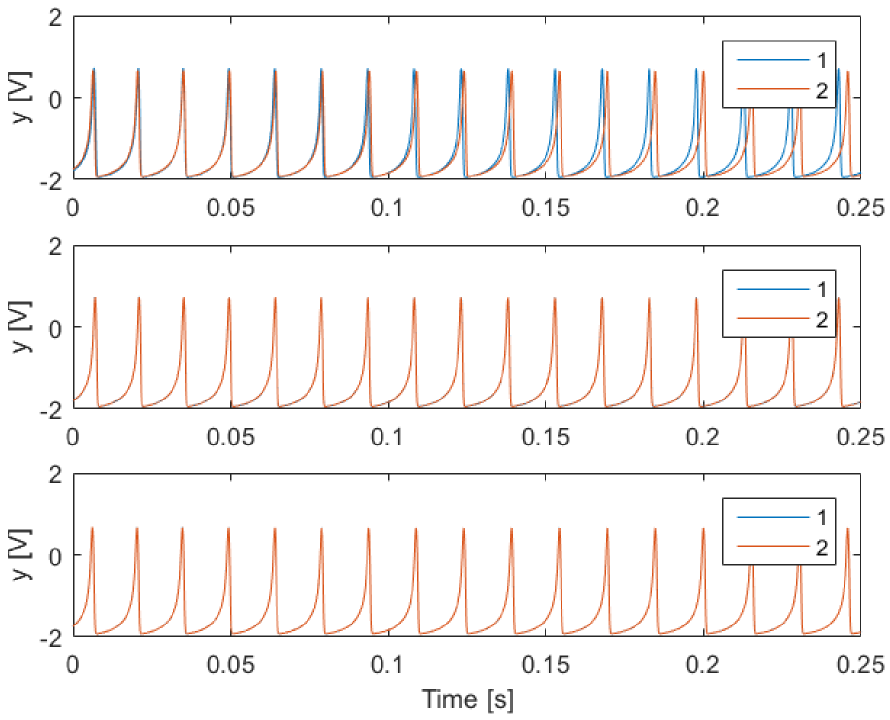

Figure 2.5: Output responses of two neurons coupled by master-slave configuration. The first plot is their uncoupled responses.

2.4 Summary

In this chapter synchronization of Hindmarsh-Rose neurons are discussed. First some biological properties of neurons are reviewed. Then different models of neuron dynamics are introduced, focusing on the modified Hindmarsh-Rose model. Simulations of heterogeneous uncoupled neurons are presented. For two coupled HR neurons we investigate the overall coupling gain and mutual adaptation parameter: the former can be used to practically synchronize two neurons and the latter can be used to adjust their collective frequency. Such results can be seen as principles for training neurons in a cluster to achieve a desired collective frequency.

Chapter 3

Training Procedure and the Effect of Noise on it

In this chapter we will introduce a training scheme of all-to-all coupling, which is developed by Vromen [1]. The training algorithm is first increasing the overall coupling gains to practically synchronize all neurons in the cluster, then adjusting the mutual adaptation parameters to coordinate the collective frequency. Based on this training algorithm, we investigate its robustness with respect to channel noise. In particular, for fixed noise power the algorithm fails to train a cluster into a collective reference frequency with large numbers of neurons in that cluster. Besides, we also investigate for different number of neurons in training the maximal tolerable noise power.

3.1 Motivation

Inspired by strong spatio-temporal pattern recognition capabilities of biological brain, a natural candidate for this project implementation is to use bio-mimic electronic brain consisting of spiking neural networks. The SNNs typically have spiking neurons that interact via synaptic connections. These synapses can be trained by various learning methods as discussed in Chapter 1. In the controlled synchronization paradigm, we use the electronic HR neurons presented in Section 2.3. Meanwhile, the synaptic connections between neurons are softwired so that they can be freely changed in the training scheme.

As shown in Chapter 2, all the electronic neurons are non-identical. An undesirable effect is that the mobile robot may have deviation from the middle (either turning left or right) when no obstacle detected. Such a scenario is experimented by Vromen [1], as shown in Figure 3.1.

An ingredient to eliminate such undesirable effects is to make all three sensor neurons have the same output for external input $I_{\min}$, which will guarantee that the mobile robot drives at full speed in a straight line. To make all three sensor neurons frequency synchronized without directly coupling them, we propose to use clusters instead of single neurons, which can self-adjust each cluster's own collective frequency to reach frequency synchrony. Note that we want such unbiased property of motor neurons also subject to multiple external input I.

The training procedure should fulfill two goals: (i) for every external input I, the outputs of each neuron in a cluster should be practically synchronized; (ii) the frequency or period of each synchronized cluster should be equal to the reference. Likewise, two adaptation algorithms are needed. Firstly, synchronize all neurons in the cluster by increasing the coupling gain. Secondly, change the mutual adaptation gain to make the collective cluster have the same frequency as the reference neuron. Note that because we consider cluster as a whole, so coupling between the new comer and each existing neuron in the cluster is the same.

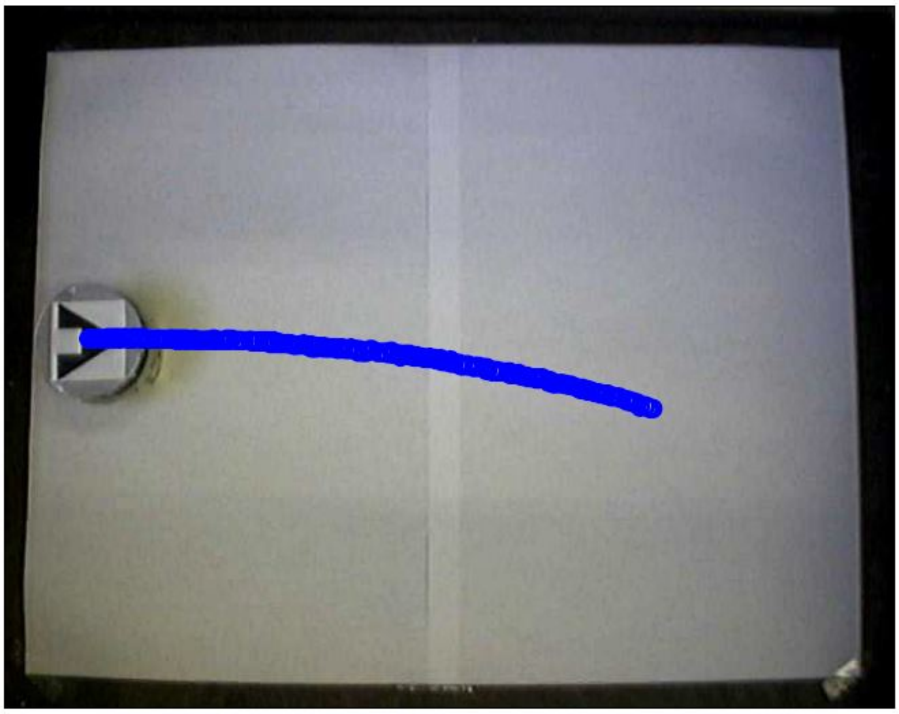

Figure 3.1: Experimental result of heterogeneous neurons producing bias, adopted from [1].

3.2 Training Procedure

Recall in the last section that the goal of our training is to achieve both synchronization within each cluster and frequency synchronization between clusters. We start illustrating the training procedure by showing a simple example.

Example 3.1. Consider two mutually coupled neurons in a cluster. The diffusive coupling term of node N_1 influenced by node N_2 is $\sigma_1\gamma_1(y_2-y_1)$, and $(1-\sigma_1)\,\gamma_1(y_1-y_2)$ the other way around. The coupling law is defined by

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = -\Gamma_2(\sigma_1,\gamma_1)\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = -\begin{pmatrix} \gamma_1\sigma_1 & -\gamma_1\sigma_1 \\ -\gamma_1(1-\sigma_1) & \gamma_1(1-\sigma_1) \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}.$$

where Γ_n is the Laplacian matrix of n coupled systems.

Step 1: increase $\gamma=\gamma_1$ by interval $\Delta\gamma=\frac{\alpha}{m-1}$ until N_1 and N_2 practically synchronized ($\sup_{\tau\in[t_1,t_2]}|y_i(\tau)-y_j(\tau)|<\epsilon$, ϵ is the practical synchronization bound and $[t_1,t_2]$ is chosen after the transients have died out). Here $\alpha>0$ is a parameter that controls the adaptation speed of coupling gain and $m\geq 2$ is the current number of neurons in the cluster. At this stage $\sigma=\sigma_1=\frac{1}{2}$ is fixed.

Step 2: adjust adaptation parameter σ_1 by amount $\Delta\sigma=\pm\frac{\alpha_\tau}{\gamma}|\Delta\tau|$ to make the collective period of synchronized cluster be as close as the reference period (within a sufficiently small bound ϵ_τ). Here $\alpha_\tau>0$ is a constant, $\Delta\tau=T_{\text{ref}}-T_{\text{cluster}}$ is the period difference between reference neuron and cluster, and $\Delta\sigma$ should be constrained in $[0,1]$. The sign $\pm$ is needed because it's uncertain whether the optimal adaptation parameter σ will lie in the interval $[0,0.5]$ or $[0.5,1]$. Thus the sign of $\Delta\sigma$ should be reversed if the adaptation pushes the cluster period away from the reference period.

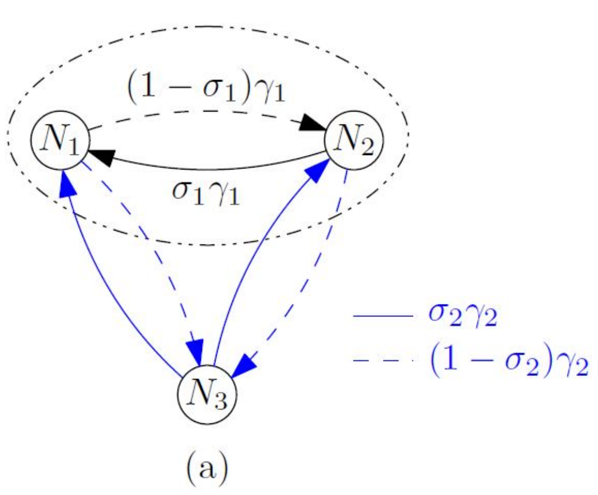

Figure 3.2: (adopted from [1]) Coupling law of three neurons: (a) synchronized cluster of two neurons bi-directionally coupled with neuron N_3; (b) each neuron in the cluster has the same inter-cluster coupling relations with outside neurons.

Step 3: fix σ_1 and γ_1 and add a third neuron to the cluster according to the law

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = -\Gamma_3(\sigma_1,\sigma_2,\gamma_1,\gamma_2)\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = -\left(\begin{array}{cc|c} \multicolumn{2}{c|}{\multirow{2}{*}{$\Gamma_2(\sigma_1,\gamma_1)+\sigma_2\gamma_2\mathbf{I}_2$}} & -\gamma_2\sigma_2 \\ & & -\gamma_2\sigma_2 \\ \hline -\gamma_2(1-\sigma_2) & -\gamma_2(1-\sigma_2) & 2\gamma_2(1-\sigma_2) \end{array}\right)\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}.$$

This coupling law comes from the fact that the coupling gain between the new comer and each existing neuron in the cluster is the same. The adaptation of γ_2 and σ_2 follows the same procedure as Step 1 and 2, but now the criteria for practical synchronization or frequency synchronization should consider the whole cluster. The schematic representation of three neuron coupling law is illustrated in Figure 3.2.

From this example one can easily derive the coupling law between neuron $n+1$ and the cluster of n synchronized neurons:

$$\left(\begin{array}{c} u_1 \\ \vdots \\ u_n \\ \hline u_{n+1} \end{array}\right) = -\Gamma_{n+1}\left(\begin{array}{c} y_1 \\ \vdots \\ y_n \\ \hline y_{n+1} \end{array}\right) = -\left(\begin{array}{ccc|c} & & & -\gamma_n\sigma_n \\ & \Gamma_n+\sigma_n\gamma_n\mathbf{I}_n & & \vdots \\ & & & -\gamma_n\sigma_n \\ \hline -\gamma_n(1-\sigma_n) & \cdots & -\gamma_n(1-\sigma_n) & n\gamma_n(1-\sigma_n) \end{array}\right)\left(\begin{array}{c} y_1 \\ \vdots \\ y_n \\ \hline y_{n+1} \end{array}\right).$$

with Γ_{n+1} the Laplacian matrix of all $n+1$ neurons, which is uniquely defined by the combinations of σ_1,γ_1,σ_2,γ_2,$\cdots$,σ_n,γ_n.

The complete training procedure is almost same with example 3.1, with further steps just a repetition of step 3, which is adding new neuron to the cluster and do training. A more detailed illustration of the frequency adaptation is shown in Figure 3.3 [1]. Here if $\sigma \leq 0$ or $\sigma \geq 1$, we should set $\sigma = 0$ or $\sigma = 1$ respectively to ensure best convergence possible. The parameter k is the adaptation loop index and $k = 1$ means the first adaptation attempt.

The parameters for the training algorithm are chosen as follows:

$$\epsilon = 0.2\ [V], \quad \epsilon_\tau = 7e^{-6}\ [s], \quad \alpha = 0.3125, \quad \alpha_\tau = 2500.$$

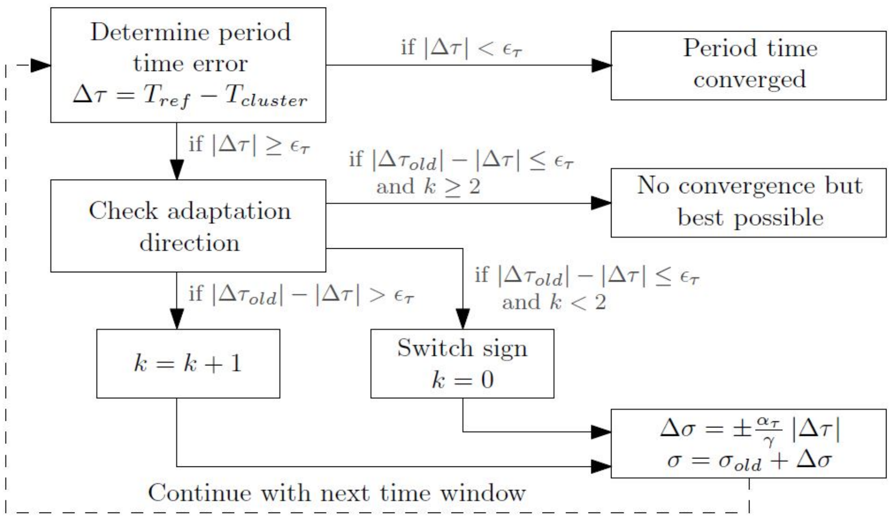
Determine period time error $\Delta\tau = T_{ref} - T_{cluster}$
if $|\Delta\tau| < \epsilon_\tau$
Period time converged
if $|\Delta\tau| \geq \epsilon_\tau$
Check adaptation direction
if $|\Delta\tau_{old}| - |\Delta\tau| \leq \epsilon_\tau$ and $k \geq 2$
No convergence but best possible
if $|\Delta\tau_{old}| - |\Delta\tau| > \epsilon_\tau$
if $|\Delta\tau_{old}| - |\Delta\tau| \leq \epsilon_\tau$ and $k < 2$
$k = k + 1$
Switch sign $k = 0$
$\Delta\sigma = \pm\frac{\alpha_\tau}{\gamma}|\Delta\tau|$ $\sigma = \sigma_{old} + \Delta\sigma$
Continue with next time window

Figure 3.3: Flowchart of frequency adaptation procedure, adopted from [1].

3.3 Simulation Results

In the simulation we try to train nine neurons using the all-to-all coupling scheme. The external input is $I_{\text{min}} = 4.5$ and reference period is chosen as $T_{\text{ref}} = 0.0151s$. This parameter is chosen to be in the middle of most points from Figure 2.3 at input $I = 4.5$. The output of all nine neurons are shown in Figure 3.4. We use solver ode23 and chose initial condition $[-2, -0.2, -0.3]$ for neuron $N1$ and add 0.01 on all three states for the next index neuron.

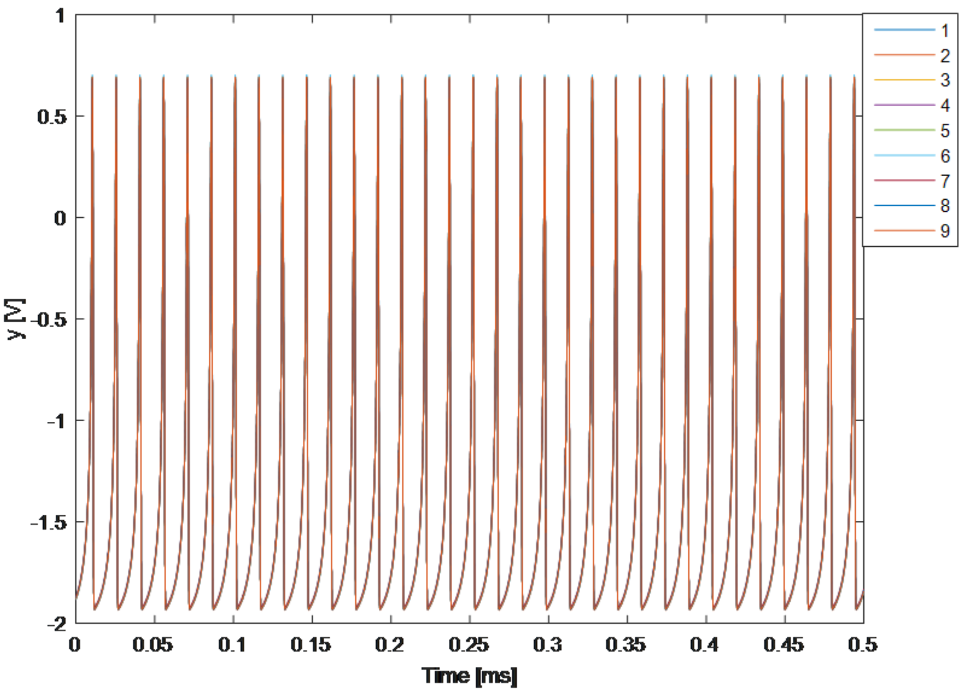

Figure 3.4: The output of nine neurons trained by all-to-all coupling scheme.

From this graph we can see that all neurons are practically synchronized. Meanwhile in the workspace we find that each of them has the same period 0.0151s. The all-to-all training scheme is successful, and ideally should work for even more neurons.

We further investigate the adaptation parameters for the training of 9 neurons, presented in Table 3.1:

Table 3.1: Adaptation parameters γ and σ after training.

γ_1	γ_2	γ_3	γ_4	γ_5	γ_6	γ_7	γ_8
1.2500	0.7813	0.5208	0.4688	0.3750	0.3125	0.2679	0.2344
σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8
0.3029	0.1856	0.5000	0.1209	0.3230	0.5000	0.0181	0.0407

$$\Gamma_9 = \begin{pmatrix} 1.1325 & -0.3787 & -0.1450 & -0.2604 & -0.0567 & -0.1211 & -0.1563 & -0.0048 & -0.0095 \\ -0.8713 & 1.6251 & -0.1450 & -0.2604 & -0.0567 & -0.1211 & -0.1563 & -0.0048 & -0.0095 \\ -0.6363 & -0.6363 & 1.8814 & -0.2604 & -0.0567 & -0.1211 & -0.1563 & -0.0048 & -0.0095 \\ -0.2604 & -0.2604 & -0.2604 & 1.1297 & -0.0567 & -0.1211 & -0.1563 & -0.0048 & -0.0095 \\ -0.4121 & -0.4121 & -0.4121 & -0.4121 & 1.9401 & -0.1211 & -0.1563 & -0.0048 & -0.0095 \\ -0.2539 & -0.2539 & -0.2539 & -0.2539 & -0.2539 & 1.4401 & -0.1563 & -0.0048 & -0.0095 \\ -0.1563 & -0.1563 & -0.1563 & -0.1563 & -0.1563 & -0.1563 & 0.9519 & -0.0048 & -0.0095 \\ -0.2630 & -0.2630 & -0.2630 & -0.2630 & -0.2630 & -0.2630 & -0.2630 & 1.8506 & -0.0095 \\ -0.2248 & -0.2248 & -0.2248 & -0.2248 & -0.2248 & -0.2248 & -0.2248 & -0.2248 & 1.7987 \end{pmatrix}.$$

From Table 3.1 we can observe that coupling gain from γ_1 to γ_8 is decreasing. The reason is that during Step 1 the new comer is equally coupled with each neuron in the cluster, thus the more neurons in the cluster, the less coupling gain needed for practical synchronization. Another observation is that the mutual adaptation parameter σ_n are all located at $[0, 0.5]$ (σ_7 and σ_8 are even close to 0). This is because in case the previous trained clusters have period time close to the reference (no σ_i is located at 0 and 1 thus we can claim this), the new comer should be dominated by the original cluster to reach reference period. Except this obvious solution, there may be other local solutions to reach desired period as the interaction effect of the new comer will influence the intrinsic collective period of original cluster, deviating the end points of Figure 2.4. Thus we can expect mutual adaptation parameters σ_i located in $[0, 0.5]$.

To further justify the aforementioned arguments, we compare Table 3.1 with the intrinsic period of all neurons (with external input $I = 4.5$) in Table 3.2:

Table 3.2: Intrinsic frequencies of all 15 neurons at input $I = 4.5$.

T_1	T_2	T_3	T_4	T_5
0.015044	0.015301	0.015186	0.014999	0.014928
T_6	T_7	T_8	T_9	T_{10}
0.015199	0.015157	0.015426	0.015315	0.015183
T_{11}	T_{12}	T_{13}	T_{14}	T_{15}
0.015345	0.015259	0.015274	0.015511	0.015143

From these two tables we can observe that $T_4 = 0.014999$ and $T_7 = 0.015157$ already have intrinsic period close to $T_{\text{ref}} = 0.0151$, thus even without frequency adaptation ($\sigma_3 = 0.5$ and $\sigma_6 = 0.5$) desired collective period can be achieved within an accuracy bound. Another observation is that $T_8 = 0.015426$ and $T_9 = 0.015315$ are very far away from reference period, thus needs a lot of effort (almost dominated by original trained clusters as $\sigma_7 = 0.0181$ and $\sigma_8 = 0.0407$) to train.

3.4 Effect of Noise on Training Scheme

In this section we consider the training scheme with channel noise for all interconnected neurons. The spike train of individual neurons recorded from *vivo* are found to have some degree of irregularity, far from being periodic [59]. The origin of such irregularity is still unknown, but in spiking neuron models such as integrate-and-fire model, noise is often added to mimic such unpredictability. Two types of noises are often distinguished: *intrinsic noise* that appears on the level of neuronal dynamics and *extrinsic noise* that arises from effects of synaptic transmission [60]. We also refer to the latter type of noise as *channel noise*.

The mathematical representation of k diffusively coupled HR neurons on a simple strongly connected graph are shown in equation (3.1):

$$\begin{aligned} \dot{x}_i &= F_i(x_i) + Bu_i + n_{1,i}, \\ y_i &= Cx_i + n_{2,i}. \end{aligned} \tag{3.1}$$

Compared with the original expression in equation (2.4), we add the intrinsic noise term $n_{1,i}$ and the channel noise term $n_{2,i}$ for neuron i. Full state $x_i = (y_i \;\; z_{i,1} \;\; z_{i,2})^{\intercal}$ includes both output and zero dynamics. $B = (1 \;\; 0 \;\; 0)^{\intercal}$, $C = (1 \;\; 0 \;\; 0)$. Replacing the diffusive coupling term u_i by weighted output errors between coupled neurons, the full state dynamics of neuron i is expressed in equation (3.2):

$$\begin{aligned} \dot{x}_i &= F_i(x_i) + n_{1,i} - B \sum_{j=1,i\neq j}^{k} \gamma_{ij} \left(y_i - y_j\right) \\ &= F_i(x_i) + n_{1,i} - BC \sum_{j=1,i\neq j}^{k} \gamma_{ij}(x_i - x_j) - B \sum_{j=1,i\neq j}^{k} \gamma_{ij}\omega_i. \end{aligned} \tag{3.2}$$

For simplicity we will only investigate the effect of channel noise and ignore intrinsic noise, which means that $n_{1,i} = 0$. The term $\omega_i = n_{2,i} - n_{2,j}$ is the channel noise for neuron i in the network (both ω_i and $n_{2,i}$ are white noise term and we do not further distinguish them). In matrix form the full state dynamics of the whole system is shown in equation (3.3):

$$\begin{pmatrix} \dot{x_1} \\ \vdots \\ \dot{x_k} \end{pmatrix} = \begin{pmatrix} F_1(x_1) \\ \vdots \\ F_k(x_k) \end{pmatrix} - \Gamma \begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix} \otimes B = \begin{pmatrix} F_1(x_1) \\ \vdots \\ F_k(x_k) \end{pmatrix} - (\Gamma \otimes BC) \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} - \Gamma \begin{pmatrix} \omega_1 \\ \vdots \\ \omega_k \end{pmatrix} \otimes B. \tag{3.3}$$

From equation (3.3) we can see that output dynamics are perturbed by channel noise and the coupling gain of the network will also influence the noise intensity.

We now investigate the effect of channel noise on the training scheme. For fixed noise power 0.00001 we simulate the responses of training different number of neurons (from 3 to 6) in a cluster. The initial conditions for each state are all uniformly distributed random number in interval $(0, 1)$. The solver is ode23. Because of the stochastic terms, each simulation may produce different responses for unsuccessful trainings. The results are shown in Figure 3.5.

By checking the adaptation parameters from the simulation of Figure 3.5, we will find that $\sigma_4 = 0$ for $i = 5$ and $\sigma_5 = 0$ for $i = 6$ respectively, and their period time are both far from reference. The overall coupling gain γ_i ($i = 3, 4, 5, 6$) is almost same as Table 3.1 (except γ_3 has become a

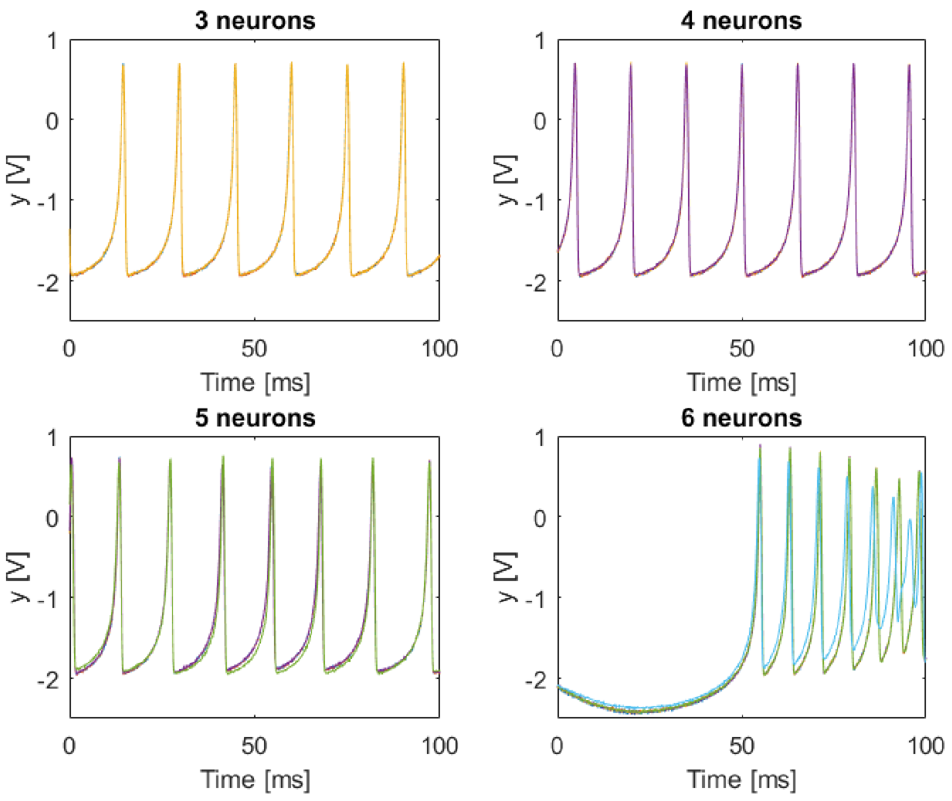

Figure 3.5: Output responses of training different number of neurons for noise power 0.00001.

bit larger), which means that training failure happens mostly in the frequency adaptation process. We also do simulations for training different number of neurons and check their responses around the critical noise threshold as presented in Figure 3.6. From Figure 3.6 we can observe that the more neurons to train, the lower the noise threshold will be for effective training. Moreover, the desynchronization are more likely to happen between the last added neuron and existing cluster. Note that the noise threshold is not tight and may vary in a small neighbourhood of the provided critical values for different simulations.

The underling reason behind training failure is that the output dynamics of neuron i is perturbed by the ith row of Laplacian matrix multiplied by k-dimensional white noise term, as can be seen from equation (3.3). Based on the analysis of mutual adaptation parameter in Section 3.3, the newly added neuron will tend to be enslaved by existing cluster, leading to larger coupling gains in the last row of the Laplacian matrix. This adds to the randomness of the dynamics of this new comer, resulting in more unpredictable training.

In the end, we plot the noise threshold for training different number of neurons. We also switch the neuron index to see whether it will have any effect on the training process. The results are shown in Figure 3.7. Note that this bound is only approximate and may vary for different simulations.

In conclusion, the all-to-all coupling scheme will amplify noise, making it more difficult to do training. Thus in real implementation the number of neurons for training is limited due to the channel noise effect. Meanwhile this scheme is very sensitive to the coupling sequence of neurons we coupled.

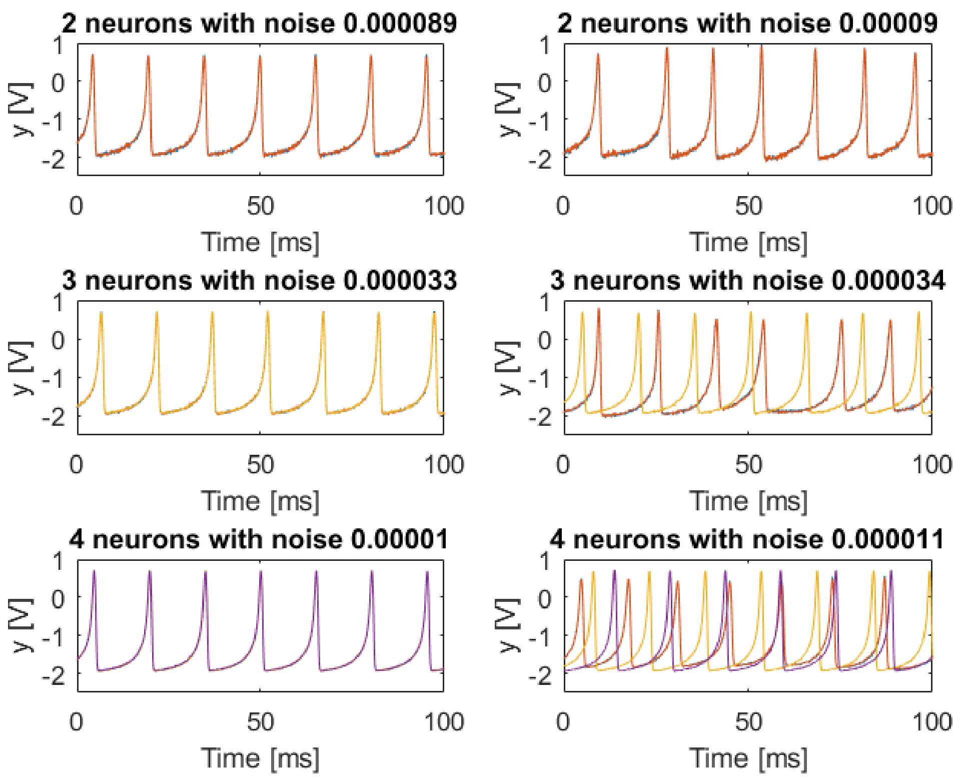

Figure 3.6: Output responses of training different number of neurons for critical noise power

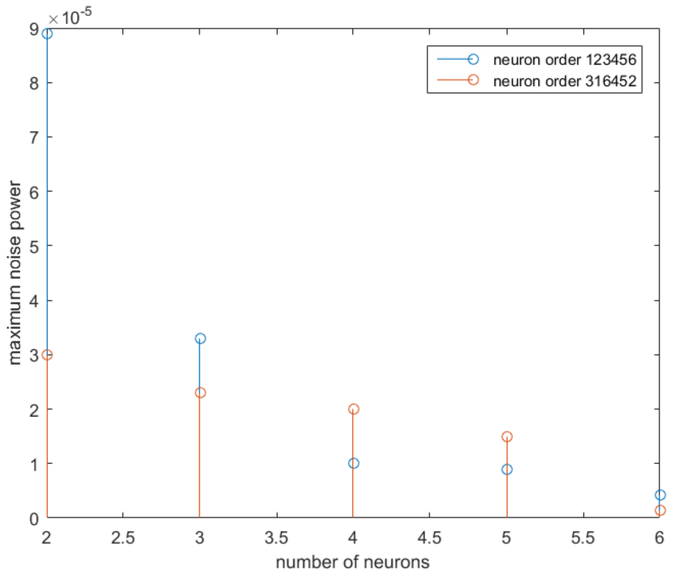

Figure 3.7: Maximum noise power allowed for different amount of neurons (from 2 to 6) to synchronize.

3.5 Discussion

In this chapter we introduce the all-to-all coupling scheme developed by [1]. The main goal of this scheme is to train the neuronal network such that for the same input ($I_{\min}$) all three sensor neurons will produce same output. The training algorithm is first increasing the overall coupling gains and then adjusting the mutual adaptation parameters. The simulation result of training 9 neurons are investigated, focusing on the adaptation parameter σ_n and γ_n. We also include the channel noise effect on the training scheme. It is found that for the scheme to do successful training, the noise power should be below some threshold, and the larger the amount of neurons for training, the lower threshold will be. The simulation of noise effect shows some limitations of this scheme.

Chapter 4

Noise Effect on Synchronizing Diffusively Coupled Systems

In this chapter we will investigate the effect of noise on synchronization of diffusively coupled systems. It is shown that for each type of subgraph in presence of channel noise, the practical synchronization error will first decrease and then increase with the change of coupling gain, until eventually lose synchrony. Such a range of coupling gain for practical synchronization is closely related to the eigenratio $\frac{\lambda_2}{\lambda_n}$. Our simulation will focus on noisy diffusively coupled systems with symmetric coupling.

4.1 Experimental Synchronization of Diffusively Coupled HR Neurons with Channel Noise

Consider typical subgraphs shown in Figure 4.1. Each edge is symmetric and has uniform strength 1. We increase the coupling gain γ and plot corresponding synchronization errors (the maximum output error between coupled neurons at all time instants after transients). The noise power is 0.0001 in each simulation. The results are shown in Figure 4.2, where the dashed line represents the practical synchronization bound 0.2. A summary of non-zero eigenvalue λ_2, λ_n and the coupling gain range γ for practical synchronization is provided in Table 4.1.

Table 4.1: Comparison of non-zero eigenvalue λ_2, λ_n and coupling threshold $\gamma_{\min}$, $\gamma_{\max}$ for different motifs.

Subgraph	λ_2	λ_n	$\frac{\lambda_2}{\lambda_n}$	$\gamma_{\min}$	$\gamma_{\max}$	$\frac{\gamma_{\max}}{\gamma_{\min}}$
G1	2	2	1	0.71	5.54	7.80
G2	1	3	1/3	1.00	3.28	3.28
G3	3	3	1	0.49	4.20	8.57
G4	0.59	3.41	0.17	1.72	4.58	2.66
G5	2	4	1/2	0.58	4.67	8.05
G6	2	4	1/2	0.65	5.87	9.03
G7	4	4	1	0.30	3.65	12.17

From Table 4.1 we can see that for fixed amount of diffusively coupled subsystems, the larger the eigenratio $\frac{\lambda_2}{\lambda_n}$ is, the wider the synchronization bound $[\gamma_{\min}, \gamma_{\max}]$ of the whole interconnec-

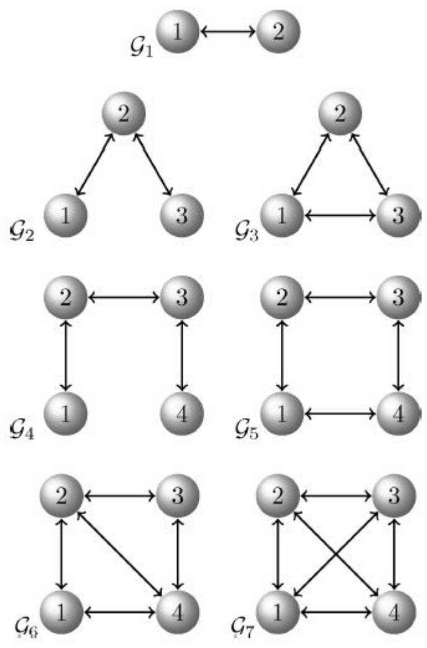

Figure 4.1: (adopted from [61]) Different subgraphs of diffusively coupled systems $\mathcal{G}_e|_{e=1,2,\cdots,7}$.

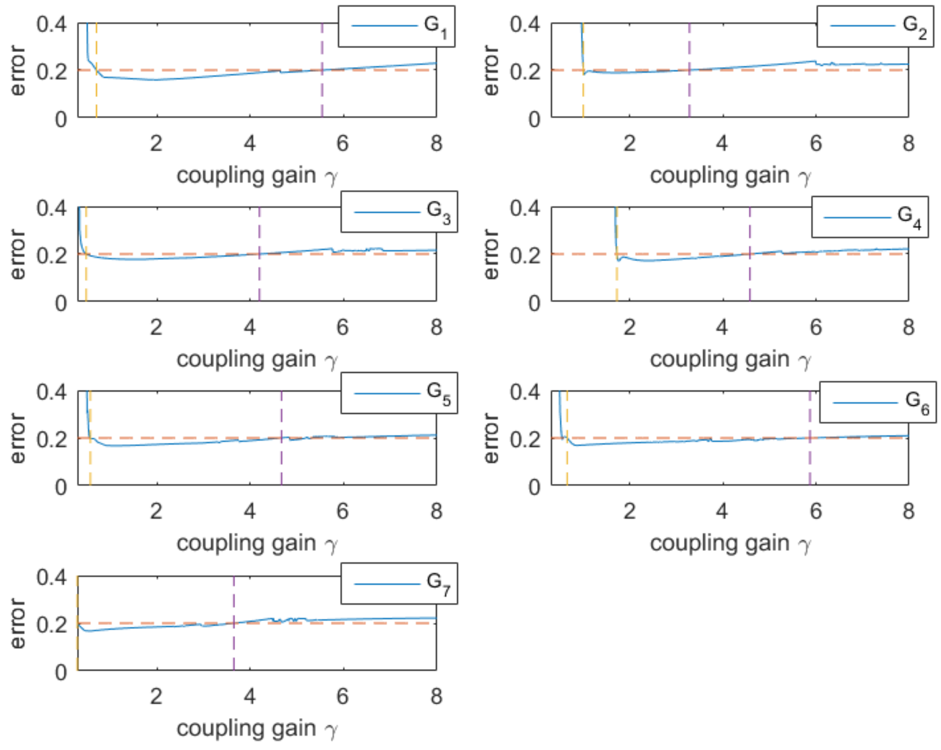

Figure 4.2: Plots of different subgraphs' coupling threshold for practical synchronization.

ted system will be in a noisy channel environment. This synchronization bound $[\gamma_{\min}, \gamma_{\max}]$ also reflects the whole system's vulnerability to noise, with $\frac{\gamma_{\max}}{\gamma_{\min}} = 1$ corresponding to maximal synchronizability and robustness to noise. Here subgraph $\mathcal{G}_3$ and $\mathcal{G}_7$ are both all-to-all symmetrically coupled with uniform strength, which will result in optimal eigenratio. Another observation is that algebraic connectivity λ_2 relates to $\gamma_{\min}$. For fixed amount of diffusively coupled HR neurons in a connected network, the coupling gain required to synchronize all neurons $\gamma_{\min}$ monotonically decreases with the increase of the smallest nonzero eigenvalue λ_2 of the Laplacian matrix [62].

4.2 Discussion

Based on the simulation results in Figure 3.7, we investigate the eigenratio corresponding to different amount of HR neurons in diffusively coupled networks with maximal noise power (neuron index 1234567). The result is shown in Table 4.2.

Table 4.2: Eigenratio for training different number of neurons using all-to-all coupling scheme with noise channel.

Number	λ_2	λ_n	$\frac{\lambda_2}{\lambda_n}$	Noise Power
2	1.25	1.25	1	0.000089
3	1.20	1.62	0.74	0.000033
4	1.04	1.67	0.62	0.000010
5	1.32	1.77	0.75	0.000009
6	1.13	2.09	0.54	0.0000042
7	1.07	2.57	0.42	0.0000018

From Table 4.2 we can see that the all-to-all scheme will result in smaller eigenratio when training more neurons, leading to worse robustness to noise. Another observation is that when training 5 neurons the eigenratio actually becomes large again, resulting in almost same noise robustness ability as training 4 neurons.

4.3 Summary

In this chapter we consider the noise effect on synchronization of all-to-all diffusively coupled HR neurons on uniform undirected networks. It is found that in presence of channel noise, the practical synchronization error will first decrease and then increase again with the change of coupling gain γ. Such a range of coupling gain for practical synchronization reflects the network's ability of robustness to noise, which is related to the eigenratio $\frac{\lambda_2}{\lambda_n}$. The robustness to noise ability is best when eigenratio equals to 1.

Chapter 5

Conclusions and Recommendations

The final chapter summarizes the main results that are presented in this report. In addition, recommendations for future research are given.

5.1 Conclusions

In this report the problem of autonomous navigation of a mobile robot is treated using a training algorithm developed by Vromen [1]. The main contribution of this internship is to reproduce this algorithm and study its behavior in presence of channel noise.

In Chapter 2 the Hindmarsh-Rose neuron are introduced and practical synchronization of them via diffusive coupling are discussed. It is shown that heterogeneous neurons can be practically synchronized by strong coupling and their collective frequency can be influenced by changing the mutual adaptation parameter.

In Chapter 3 we introduce the training algorithm developed by [1] and study the effect of noise on it. Based on Vromen's algorithm, we add channel noise and investigate for fixed noise power how the responses of training different number of neurons will change. It is found that the more neurons to train, the lower noise power are tolerable for effective training.

In Chapter 4, we study the effect of noise on synchronization of diffusively coupled HR neurons in undirected networks. It is investigated for each type of subgraph the relation between coupling gain and practical synchronization error when noise power is fixed. The spectral properties of each subgraph are computed to correlate with their ability of robustness to noise.

5.2 Recommendations

The training algorithm developed by Vromen [1] has some limitations. Firstly, the choice of reference neuron is not random and preferably requires its intrinsic frequency located in the middle of most other neurons' frequencies in the cluster for each external input. More intelligent and automatic choice of reference neuron or equilibrium states are preferable. Secondly, the all-to-all network structure will amplify noise for training large number of neurons in a cluster. To weaken the effect of noise, new algorithms should consider more flexible and less conservative structure. Thirdly, it should be more biologically plausible to train the sensor clusters in a distributed manner.

Moreover, it is expected to design robust, highly synchronizable network structure that can achieve desired collective frequency simply by using the information of weight distributions of different neurons. Such optimization-based perspective could give us insights about how to design decentralized training algorithms to achieve local or even global optimal solutions.

Bibliography

[1] T. G. M. Vromen, E. Steur, and H. Nijmeijer, “Training a network of electronic neurons for control of a mobile robot,” *International Journal of Bifurcation and Chaos*, vol. 26, no. 12, 2016.

[2] R. J. Williams and N. D. Martinez, “Simple rules yield complex food webs,” *Nature*, vol. 404, pp. 180–183, Mar 2000.

[3] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási, “The large-scale organization of metabolic networks,” *Nature*, vol. 407, no. 6804, pp. 651–654, 2000.

[4] R. Albert, H. Jeong, and A.-L. Barabási, “Internet: Diameter of the World-Wide Web,” *Nature*, vol. 401, pp. 130–131, Sep 1999.

[5] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, “Complex networks: Structure and dynamics,” *Physics Reports*, vol. 424, no. 45, pp. 175 – 308, 2006.

[6] G. Zamora-López, C. S. Zhou, and J. Kurths, “Cortical hubs form a module for multisensory integration on top of the hierarchy of cortical networks,” *Frontiers in Neuroinformatics*, 2010.

[7] S. H. Strogatz, *Sync: the emerging science of spontaneous order.* New York: Theia, 2003.

[8] A. Pikovsky, M. Rosenblum, and J. Kurths, *Synchronization: a universal concept in nonlinear sciences.* The Cambridge nonlinear science series, Cambridge: Cambridge University Press, 2001.

[9] J. Buck, “Synchronous rhythmic flashing of fireflies. II.,” *The Quarterly Review of Biology*, vol. 63, no. 3, pp. 265–289, 1988.

[10] I. D. Couzin, J. Krause, N. R. Franks, and S. A. Levin, “Effective leadership and decision-making in animal groups on the move,” *Nature*, vol. 433, pp. 513–516, Feb 2005.

[11] C. A. Czeisler, E. Weitzman, M. C. Moore-Ede, J. C. Zimmerman, and R. S. Knauer, “Human sleep: its duration and organization depend on its circadian phase,” *Science*, vol. 210, p. 1264, Dec 1980.

[12] C. S. Peskin, *Mathematical aspects of heart physiology.* New York, NY, USA: Courant Institute of Mathematical Sciences, New York University, 1975.

[13] V. S. Afraimovich, N. N. Verichev, and M. I. Rabinovich, “Stochastic synchronization of oscillation in dissipative systems,” *Radiophysics and Quantum Electronics*, vol. 29, no. 9, pp. 795–803, 1986.

[14] S. Postnova, K. Voigt, and H. A. Braun, “Neural synchronization at tonic-to-bursting transitions,” *Journal of Biological Physics*, vol. 33, pp. 129–143, Apr 2007. 9048[PII].

[15] A. E. Motter, C. S. Zhou, and J. Kurths, “Network synchronization, diffusion, and the paradox of heterogeneity,” *Physical Review E*, vol. 71, p. 016116, Jan 2005.

[16] M. E. J. Newman, “Assortative mixing in networks,” *Physical Review Letters*, vol. 89, p. 208701, Oct 2002.

[17] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[18] M. E. J. Newman, “The structure and function of complex networks,” *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[19] L. M. Pecora and T. L. Carroll, “Synchronization in chaotic systems,” *Physical Review Letters*, vol. 64, pp. 821–824, Feb 1990.

[20] N. G. Rambidi, K. E. Shamayaev, and G. Peshkov, “Image processing using light-sensitive chemical waves,” *Physics Letters A*, vol. 298, no. 56, pp. 375 – 382, 2002.

[21] V. I. Krinsky, V. N. Biktashev, and I. R. Efimov, “Autowave principles for parallel image processing,” *Physica D: Nonlinear Phenomena*, vol. 49, no. 1, pp. 247 – 253, 1991.

[22] H. J. C. Huijberts, H. Nijmeijer, and R. M. A. Willems, “A control perspective on communication using chaotic systems,” in *Proceedings of the 37th IEEE Conference on Decision and Control, 1998*, (Piscataway, NJ, USA), pp. 1957–1962, IEEE, 1998.

[23] F. C. Hoppensteadt and E. M. Izhikevich, “Pattern recognition via synchronization in phase-locked loop neural networks,” *IEEE Transactions on Neural Networks*, vol. 11, pp. 734–738, May 2000.

[24] K. Y. Pettersen, J. T. Gravdahl, and H. Nijmeijer, *Group coordination and cooperative control*, vol. 336. Berlin: Springer, 2006.

[25] F. Dörfler, M. Chertkov, and F. Bullo, “Synchronization in complex oscillator networks and smart grids,” *Proceedings of the National Academy of Sciences*, vol. 110, no. 6, pp. 2005–2010, 2013.

[26] W. Ren and R. W. Beard, *Distributed consensus in multi-vehicle cooperative control: theory and applications*. London: Springer-Verlag London Limited, 2008;2007;.

[27] F. Bullo, J. Cortés, and S. Martínez, *Distributed control of robotic networks*. Applied Mathematics Series, Princeton University Press, 2009.

[28] S. J. Chung and J. J. E. Slotine, “Cooperative robot control and synchronization of Lagrangian systems,” in *Decision and Control, 2007 46th IEEE Conference on*, pp. 2504–2509, IEEE, 2007.

[29] S. J. Chung and J. J. E. Slotine, “Cooperative robot control and concurrent synchronization of Lagrangian systems,” *IEEE Transactions on Robotics*, vol. 25, pp. 686–700, June 2009.

[30] W. Ren, R. Beard, and E. Atkins, “Information consensus in multivehicle cooperative control,” *IEEE Control Systems Magazine*, vol. 27, no. 2, pp. 71–82, 2007.

[31] P. R. Wurman, R. D’Andrea, and M. Mountz, “Coordinating hundreds of cooperative, autonomous vehicles in warehouses,” *AI Magazine*, vol. 29, no. 1, pp. 9–19, 2008.

[32] J. Guivant, E. Nebot, and S. Baiker, “Autonomous navigation and map building using laser range sensors in outdoor applications,” *Journal of Robotic Systems*, vol. 17, pp. 3817–3822, 2000.

[33] R. Chatterjee and F. Matsuno, “Use of single side reflex for autonomous navigation of mobile robots in unknown environments,” *Robotics and Autonomous Systems*, vol. 35, no. 2, pp. 77–96, 2001.

[34] M. Caudill, "Neural networks primer, part i," *AI Expert*, vol. 2, pp. 46–52, Dec. 1987.

[35] J. Vreeken, "Spiking neural networks, an introduction," tech. rep., Utrecht University, 2003.

[36] D. Floreano, Y. Epars, J. C. Zufferey, and C. Mattiussi, "Evolution of spiking neural circuits in autonomous mobile robots," *International Journal of Intelligent Systems*, vol. 21, no. 9, pp. 1005–1024, 2006.

[37] M. J. Pearson, A. G. Pipe, B. Mitchinson, K. Gurney, C. Melhuish, I. Gilhespy, and M. Nibouche, "Implementing spiking neural networks for real-time signal-processing and control applications: a model-validated fpga approach," *IEEE Transactions on Neural Networks*, vol. 18, no. 5, pp. 1472–1487, 2007.

[38] X. Q. Wang, Z. G. Hou, A. M. Zou, M. Tan, and L. Cheng, "A behavior controller based on spiking neural networks for mobile robots," *Neurocomputing*, vol. 71, no. 4, pp. 655–666, 2008.

[39] S. P. Johnston, G. Prasad, L. Maguire, and T. M. Mcginnity, "An FPGA hardware/software co-design towards evolvable spiking neural networks for robotics application," *International Journal of Neural Systems*, vol. 20, no. 6, pp. 447–461, 2010.

[40] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J. C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education in engineering," in *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions* (P. J. Gonalves, P. J. Torres, and C. M. Alves, eds.), vol. 1, (Portugal), pp. 59–65, IPCB: Instituto Politcnico de Castelo Branco, 2009.

[41] A. M. Bloch, P. S. Krishnaprasad, and R. M. Murray, *Nonholonomic mechanics and control.* New York NY: Springer New York, 2nd 2015. ed., 2015.

[42] T. H. Bullock, M. V. L. Bennett, D. Johnston, R. Josephson, E. Marder, and R. D. Fields, "The Neuron Doctrine, Redux," *Science*, vol. 310, no. 5749, pp. 791–793, 2005.

[43] C. Koch, *Biophysics of computation: information processing in single neurons.* Oxford university press, 2004.

[44] B. W. Connors and M. A. Long, "Electrical synapses in the mammalian brain," *Annual Review of Neuroscience*, vol. 27, no. 1, pp. 393–418, 2004.

[45] M. V. L. Bennett and R. Zukin, "Electrical coupling and neuronal synchronization in the Mammalian Brain," *Neuron*, vol. 41, no. 4, pp. 495 – 511, 2004.

[46] E. M. Izhikevich, *Dynamical systems in neuroscience : the geometry of excitability and bursting.* Computational neuroscience, Cambridge, Mass., London: MIT Press, 2007.

[47] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of Physiology*, vol. 117, pp. 500–544, Aug 1952. 12991237.

[48] C. Morris and H. Lecar, "Voltage oscillations in the barnacle giant muscle fiber," *Biophysical Journal*, vol. 35, pp. 193–213, Jul 1981. 7260316.

[49] F. Richard, "Impulses and physiological states in theoretical models of nerve membrane," *Biophysical Journal*, vol. 1, no. 6, pp. 445 – 466, 1961.

[50] J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon," *Proceedings of the IRE*, vol. 50, pp. 2061–2070, 1962.

[51] J. L. Hindmarsh and R. M. Rose, "A model of neuronal bursting using tree coupled first order differential equations," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. B221, pp. 87–102, 1984.

[52] Editor, "Focus issue on control and synchronization of chaos," *Chaos*, vol. 7, no. 4, 1997.

[53] J. L. Hindmarsh and R. M. Rose, "A model of the nerve impulse using two first-order differential equations," *Nature*, vol. 296, pp. 162–164, Mar 1982.

[54] W. T. Oud, I. Tyukin, and H. Nijmeijer, "Sufficient conditions for synchronization in an ensemble of Hindmarsh and Rose neurons: passivity-based approach," *Proceedings NOLCOS 2004*, 2004.

[55] E. Steur, "On synchronization of electromechanical Hindmarsh-Rose oscillators," *Master thesis*, 2007.

[56] P. J. Neefs, "Experimental synchronization of Hindmarsh-Rose neurons in complex networks," *Master thesis*, 2009.

[57] E. Steur, C. Murguia, R. H. Fey, and H. Nijmeijer, "Synchronization and partial synchronization experiments with networks of time-delay coupled hindmarsh–rose neurons," *International Journal of Bifurcation and Chaos*, vol. 26, no. 07, p. 1650111, 2016.

[58] E. Panteley and A. Loria, "On practical synchronisation and collective behaviour of networked heterogeneous oscillators," *IFAC-PapersOnLine*, vol. 48, no. 18, pp. 25–30, 2015.

[59] W. Gerstner and W. M. Kistler, *Spiking neuron models : single neurons, populations, plasticity.* Cambridge, New York, Melbourne: Cambridge University Press, 2002. Autre tirage : 2006, 2008 (4e).

[60] A. Manwani and C. Koch, "Detecting and estimating signals in noisy cable structures, i: neuronal noise sources," *Neural computation*, vol. 11, no. 8, pp. 1797–1829, 1999.

[61] P. J. Neefs, E. Steur, and H. Nijmeijer, "Network complexity and synchronous behavior: an experimental approach," *International Journal of Neural Systems*, vol. 20, no. 03, pp. 233–247, 2010.

[62] C. W. Wu and L. O. Chua, "On a conjecture regarding the synchronization in an array of linearly coupled dynamical systems," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 43, no. 2, pp. 161–165, 1996.

[63] H. K. Khalil, *Nonlinear systems.* London: Pearson Education Limited, third pearson new international ed., 2014.

[64] A. Y. Pogromski, T. Glad, and H. Nijmeijer, "On diffusion driven oscillations in coupled dynamical systems," *International Journal of Bifurcation and Chaos*, vol. 9, no. 4, pp. 629–644, 1999.

[65] A. Pogromsky and H. Nijmeijer, "Cooperative oscillatory behavior of mutually coupled dynamical systems," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 2, pp. 152–162, 2001.

[66] B. P. Demidovich, "Lectures on stability theory," 1967.

[67] A. V. Pavlov, N. v. d. Wouw, and H. Nijmeijer, *Uniform output regulation of nonlinear systems : a convergent dynamics approach.* Boston: Birkhuser, 1 ed., 2006.

[68] A. Pavlov, A. Pogromsky, N. v. d. Wouw, and H. Nijmeijer, "Convergent dynamics, a tribute to Boris Pavlovich Demidovich," *Systems and Control Letters*, pp. 257–261, 2004.

[69] B. Bollobas, *Modern graph theory.* Springer, 1998.

[70] R. Diestel, *Graph theory.* Springer, August 2005.

[71] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, vol. 23, no. 2, pp. 298–305, 1973.

Appendix A

Preliminaries

In this appendix the notation, we present some definitions and mathematical framework. The notions of *ultimate boundedness, semipassivity, convergent systems* and some basic terminology of *graph theory* are discussed. Mathematical framework for synchronization are also presented.

A.1 Notation

The symbol $\mathbb{R}$ denotes the real numbers and $\mathbb{C}$ denotes the complex numbers. The symbol $\mathbb{R}_{>}$ ($\mathbb{R}_{\geqslant 0}$) stands for positive (non-negative) real numbers and $\mathbb{C}_{>}$ ($\mathbb{C}_{\geqslant 0}$) stands for complex numbers with positive (non-negative) real parts. The Euclidian norm in $\mathbb{R}^n$ is denoted as $|\cdot|$, $|x|^2 := x^\mathsf{T} x$, where x^T denotes the transpose of x. The $n \times n$ identity matrix is denoted by I_n or simply I if no confusion can arise.

The symbol $\otimes$ stands for Kronecker product, i.e. given any two matrices A and B,

$$A \otimes B = \begin{bmatrix} A_{11}B & A_{12}B & \cdots & A_{1n}B \\ A_{21}B & A_{22}B & \cdots & A_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1}B & A_{n2}B & \cdots & A_{nn}B \end{bmatrix}$$

where A_{ij} denotes the ij^{th} entry of matrix A.

Let $\mathcal{X} \in \mathbb{R}^n$ and $\mathcal{Y} \in \mathbb{R}^m$ be the *n-dimensional* and *m-dimensional* manifold, respectively. We denote by $\mathcal{C}^r$ the space of continuous functions from $\mathcal{X}$ to $\mathcal{Y}$ that are at least r times continuously differentiable. Let $\mathcal{L}_\infty(\mathcal{X}, \mathcal{Y})$ denote the space of essentially bounded functions that map elements of $\mathcal{X}$ into elements of $\mathcal{Y}$, i.e. it is the space of all measurable functions $f : \mathcal{X} \to \mathcal{Y}$ for which ess *sup* $|f| < \infty$.

A.2 Ultimate Boundedness and Semipassive Systems

Consider a system of ordinary differential equations,

$$\dot{x}(t) = f(t, x(t)) \tag{A.1}$$

with state $x \in \mathbb{R}^n$ and $f : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ being piecewise continuous in t and locally Lipschitz continuous in x for all $t \geq t_0$. The assumptions on f guarantee existence and uniqueness of solutions.

For common Lyapunov stability concepts (stable, uniformly stable, asymptotically stable, exponentially stable, etc.) we refer the readers to [63]. Here we introduce some notions of boundedness for the system (A.1).

Definition A.1. (Lagrange stability and $\mathcal{L}$-dissipative, [64]). The system (A.1) is called

i. *Lagrange stable* if every solution is bounded in forward time;

ii. *$\mathcal{L}$-dissipative* if the system is Lagrange stable and there exists a constant $c > 0$ such that $\limsup_{t\to\infty} |x(t)| \leq c$ for any initial condition $x_0 \in \mathbb{R}^n$.

Remark A.1. The solution of a $\mathcal{L}$-dissipative system are *ultimately bounded*, which means that all solutions of any initial condition enter a compact set in finite time.

Consider the system

$$\begin{aligned} \dot{x} &= f(x) + Bu \\ y &= Cx \end{aligned} \tag{A.2}$$

with state $x \in \mathbb{R}^n$, input $u \in \mathbb{R}^m$, output $y \in \mathbb{R}^m$, and $f : \mathbb{R}^n \to \mathbb{R}^n$ is a $\mathcal{C}^1$ function and matrices B and C of appropriate dimensions.

Definition A.2. [65] The dynamical system (A.2) is called $\mathcal{C}^r$-semipassive if there exists a nonnegative function $V \in \mathcal{C}^r(\mathbb{R}^n, \mathbb{R}_{\geq 0})$, $x \mapsto V(x)$, called the storage function, such that

$$\dot{V}(x) = \frac{\partial V(x)}{\partial x}\left(f(x) + Bu\right) \leq y^{\intercal}u - H(x) \tag{A.3}$$

where the function $H \in \mathcal{C}^0(\mathbb{R}^n, \mathbb{R})$ is nonnegative outside some ball, i.e., $\exists\ \varphi > 0$ s.t. $|x| \geq \varphi \Rightarrow H(x) \geq \varrho(|x|)$, for some continuous nonnegative function $\varrho(\cdot)$ defined for $|x| \geq \varphi$. If the function $H(\cdot)$ is positive outside some ball, then the system (A.2) is said to be strictly $\mathcal{C}^r$-semipassive.

Remark A.2. System (A.2) is $\mathcal{C}^r$-passive (strictly $\mathcal{C}^r$-passive) if it is $\mathcal{C}^r$-semipassive (strictly $\mathcal{C}^r$-semipassive) with $H(\cdot)$ being positive semidefinite (positive definite).

Remark A.3. Solutions of strictly semi-passive systems with a radially unbounded storage function V are ultimately bounded if $y^{\intercal}u \leq 0$. The proof can be found in [64].

A.3 Convergent Systems

Definition A.3. (Convergent systems,[66, 67]). Consider the system

$$\dot{x}(t) = f(x(t), \omega(t)), \tag{A.4}$$

with state $x \in \mathbb{R}^n$, time-varying input $\omega(t) \in \mathbb{PC}(\mathbb{R}, \mathcal{W})$, that is, $\omega(t)$ is piecewise continuous in t and takes values in a compact set $\mathcal{W} \subset \mathbb{R}^m$. The function f is locally Lipschitz and $\mathcal{C}^1$ in x. The system (A.4) is called *convergent* if and only if for any bounded input $\omega(t) \in \mathbb{PC}(\mathbb{R}, \mathcal{W})$ defined on $\mathbb{R}$, there is a unique bounded globally asymptotically stable solution $\bar{x}_\omega(t)$ in the same interval and for any initial condition it holds that

$$\lim_{t\to\infty} |x(t) - \bar{x}_\omega(t)| = 0. \tag{A.5}$$

If the differences between $x(t)$ and $\bar{x}_\omega(t)$ exponentially decrease:

$$|x(t) - \bar{x}_\omega(t)| \leq Ce^{-\alpha(t-t_0)}, \tag{A.6}$$

where $C > 0$ and $\alpha > 0$ are the same for all solutions $x(t)$, then we call the system (A.4) *exponentially convergent.*

Proposition A.1. *(Demidovich Condition [68]). If there exists a symmetric positive definite matrix $P \in \mathbb{R}^{n\times n}$ such that all the eigenvalues $\lambda_i(Q)$ of the symmetric matrix*

$$Q(x, u) = \frac{1}{2}\left(P\left(\frac{\partial f}{\partial x}(x, \omega)\right) + \left(\frac{\partial f}{\partial x}(x, \omega)\right)^\intercal P\right) \tag{A.7}$$

are negative definite and separated from zero, i.e. there exists a constant $c \in \mathbb{R}_{>0}$ such that

$$\lambda_i(Q) \leq -c < 0 \tag{A.8}$$

for all $i \in 1, \cdots, n$, $u \in U$, and $x \in \mathbb{R}^n$, then the system (A.4) is exponentially convergent.

A.4 Elementary Graph Theory

In this section some basic terminology from graph theory is presented, with notation and terminology adopted from [69, 70]. A *graph* is a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V} = \{v_1, v_2, \cdots, v_k\}$ denoting the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges. A graph is called *undirected* if $\{x, y\} \in \mathcal{E}$ for each $\{y, x\} \in \mathcal{E}$. An undirected network is called *connected* if every two nodes are joined by some path. A directed edge from node v_i to node v_j is denoted by $(v_j, v_i) \in \mathcal{E}$, which means that node v_j can receive information from node v_i. For a weighted digraph (directed graph) we use $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ with A the weighted *adjacency matrix.* The neighbours of v_i is the set of directed edges to node v_i and is denoted by $\mathcal{E}_i$. If the graph does not contain self-loops and any two nodes are joined by maximally one edge for each direction, it is called simple. Assume that the network consists of k nodes, then the adjacency matrix $A \in \mathbb{R}^{k\times k} := a_{ij}$ with $a_{ij} > 0$ if $\{i, j\} \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. We also introduce the *degree matrix* $D \in \mathbb{R}^{k\times k} := \text{diag}\{d_1, \cdots, d_k\}$ with $d_i = \sum_{j\in\mathcal{E}_i} a_{ij}$, and $L := D - A$, which is called the *Laplacian matrix* of graph $\mathcal{G}$.

Definition A.4. A digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ is said to have a *spanning tree* if its set of nodes $\mathcal{V}$ contains at least one node from which information can propagate to all other nodes along paths in $\mathcal{G}$. It is called *strongly connected* if there are paths connecting any two nodes.

A.5 Mathematical Framework

This section introduces the class of systems to be considered in this report. In particular, the necessary input-output properties of each system in the network are specified.

Theorem A.1. *[65]. Consider k systems on a strongly connected graph:*

$$\dot{z}_i(t) = q(z_i(t), y_i(t)) \tag{A.9a}$$

$$\dot{y}_i(t) = a(y_i(t), z_i(t)) + Bu_i(t), \quad i \in \mathcal{I} := \{1, 2, \cdots, k\} \tag{A.9b}$$

where $y_i \in \mathbb{R}^m$, $z_i \in \mathbb{R}^p$,$u_i \in \mathcal{L}_\infty(\mathbb{R}, \mathbb{R}^m)$, sufficiently smooth function $q : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}^m$ and $a : \mathbb{R}^p \times \mathbb{R}^m \to \mathbb{R}^p$. The matrix $B \in \mathbb{R}^{m\times m}$ is positive definite. The systems (A.9) are diffusively coupled in the relation

$$u_i(t) = \sigma \sum_{j\in\mathcal{E}_i} a_{ij} \left(y_j(t) - y_i(t)\right) \tag{A.10}$$

with coupling strength $\sigma > 0$ and interconnection weights $a_{ij} = a_{ji} \geq 0$ for all $i, j \in \mathcal{I}$. We make the following assumptions:

***H2.1** each system (A.9) is strictly semipassive with respect to input u_i and output y_i with continuously differentiable and radially unbounded storage functions;*

***H2.2** the internal dynamics (A.9a) is an exponentially convergent system, i.e., there exists a positive definite matrix $P \in \mathbb{R}^{p\times p}$ such that the eigenvalues of the symmetric matrix*

$$Q(z_i, y_i) = \frac{1}{2}\left(P\left(\frac{\partial q}{\partial z_i}(z_i, y_i)\right) + \left(\frac{\partial q}{\partial z_i}(z_i, y_i)\right)^{\intercal} P\right) \tag{A.11}$$

are uniformly negative and bounded away from zero for all $z_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}^m$.

Then the solutions of the system (A.9) (A.10) are ultimately bounded and there exists a constant σ^* such that if $\sigma\lambda_2(L) \geq \sigma^*$, $\lambda_2(L)$ is the smallest nonzero eigenvalue of the symmetric Laplacian matrix, there exists a globally asymptotically stable subset of the diagonal set

$$\mathcal{M} := \left\{ col(z_1, \cdots, z_k, y_1, \cdots, y_k) \in \mathbb{R}^{k(p+m)} | y_i = y_j \text{ and } z_i = z_j \text{ for all } i, j \in \mathcal{I} \right\}. \tag{A.12}$$

Remark A.4. Assumption (H2.2) ensures that the zero-dynamics is convergent, uniformly in the passive outputs. Assumption (H2.1) and (H2.2) are independent of the network. This means that systems can synchronize for sufficiently large $\sigma\lambda_2(L)$ (λ_2 is *algebraic connectivity, c.f. [71]*).

Appendix B

Parameter Table of Hindmarsh-Rose Neurons

The parameters of the electronic HR neurons are taken from [56]. Note that the index has been change.

Neuron	c_1	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	$c_{11} \cdot 10^3$	c_{12}	c_{13}
1	0.9946	2.9925	4.9564	0.9880	7.8380	0.9874	1.0138	2.0271	1.0110	5.0279	3.9897	4.5348
2	0.9902	2.9861	4.9454	0.9829	7.8420	0.9846	1.0083	2.0305	1.0074	4.9914	4.0188	4.6579
3	1.0036	2.9826	4.9312	0.9946	8.0198	1.0009	1.0119	2.0174	0.9977	4.8884	4.1030	4.6043
4	0.9989	2.9737	4.9014	0.9941	7.9129	0.9989	1.0116	2.0161	0.9959	4.8677	4.0143	4.5275
5	0.9982	2.9915	4.9340	0.9860	7.8960	0.9884	1.0132	2.0216	1.0072	4.9686	4.0084	4.4629
6	1.0063	2.9905	4.9246	0.9888	7.9346	0.9949	1.0196	2.0255	1.0050	4.8458	4.0646	4.5447
7	1.0112	2.9898	4.9491	0.9841	7.8532	0.9916	1.0161	2.0262	1.0044	4.8643	4.0427	4.6245
8	0.9913	2.9581	4.9191	0.9981	7.9737	1.0031	0.9958	2.0039	0.9906	4.8605	4.0235	4.6363
9	1.0061	2.9999	4.9819	0.9908	7.8814	0.9935	1.0074	2.0152	1.0034	4.8442	4.0317	4.6456
10	1.0080	2.9734	4.9190	0.9872	7.9038	0.9994	1.0142	2.0159	0.9954	4.9425	4.0702	4.6286
11	1.0351	3.0026	4.9587	1.0023	7.9654	1.0029	1.0295	2.0119	1.0046	4.8333	4.0277	4.3949
12	0.9993	2.9829	4.9317	0.9914	7.9626	1.0036	1.0108	2.0138	1.0003	4.8000	4.0749	4.6086
13	1.0137	2.9841	4.9374	0.9989	7.9595	1.0015	1.0183	2.0326	1.0013	4.8833	4.0925	4.6788
14	0.9802	2.9825	4.9388	1.0024	8.0299	1.0059	1.0003	2.0100	1.0017	4.8252	4.0299	4.5863
15	1.0061	2.9891	4.9247	0.9867	7.8698	0.9972	1.0136	2.0191	0.9969	4.9159	4.1138	4.7313