

Imperial College London  
Department of Computing

---

# A NEURALLY CONTROLLED ROBOT THAT LEARNS

---

by

Benjamin Walther Büel (B.W.)

Submitted in partial fulfilment of the requirements for  
the MSc Degree in Computing Science / Artificial Intelligence of Imperial College London





## Abstract

Spike time dependent plasticity (STDP) has been subject to much research in recent years and is understood as one of the causes for changes in synaptic efficiency. Reward modulated STDP via dopamine is a prime candidate to explain how the non-declarative long term memory in the mammalian brain learns. We will explore the advances in this field and apply reward modulated STDP in an embodied context of a simple differential drive robot.

Up to date relatively little work has been done to apply spiking neural network simulators to a robot environment, and there are no studies we are aware of that successfully demonstrated the learning via reward modulated STDP in a wheeled robot. We apply a functional approach to model motor sensor interactions and focus on the learning in the neural controller of the robot. With instrumental conditioning we teach the robot to learn different behaviours such as obstacle avoidance, searching for food or combinations of the two.

We can demonstrate under what conditions reward modulated learning is possible and show that such a robot over time has a similar performance as when we impose neuroanatomical constraints to wire the behaviour fix into the controller.



## Acknowledgements

Firstly, I would like to say thanks to Prof Murray Shanahan as the supervisor of this project who has made the project possible and has helped me throughout, with good advice and valuable feedback. I also thank Dr Andreas Fidjeland for his assistance with NeMo and many other aspects of the project.

# Content

<b>1</b>	<b>INTRODUCTION .....</b>	<b>1</b>
1.1	MOTIVATION .....	1
1.2	PROJECT AIMS .....	2
1.3	PROJECT CONTRIBUTIONS .....	3
1.4	REPORT STRUCTURE .....	4
<b>2</b>	<b>BACKGROUND .....</b>	<b>5</b>
2.1	BIOLOGICAL FOUNDATIONS .....	5
2.1.1	<i>Neurons and Synapses.....</i>	<i>5</i>
2.1.2	<i>Spike-time dependent plasticity (STDP).....</i>	<i>7</i>
2.1.3	<i>Homeostatic plasticity: synaptic scaling and intrinsic plasticity.....</i>	<i>9</i>
2.1.4	<i>Reward modulated STDP .....</i>	<i>10</i>
2.1.5	<i>Sensori-motor loops in the brain .....</i>	<i>12</i>
2.2	NEURON MODELS.....	12
2.2.1	<i>Hudgkin-Huxley model .....</i>	<i>13</i>
2.2.2	<i>Integrate-and-Fire models.....</i>	<i>14</i>
2.2.3	<i>Izhikevich model .....</i>	<i>14</i>
2.3	SPIKING NEURAL NETWORKS.....	16
2.4	NEURAL CODING .....	17
2.5	REINFORCEMENT LEARNING .....	19
2.5.1	<i>Temporal difference learning .....</i>	<i>20</i>
2.5.2	<i>Reward structure .....</i>	<i>22</i>
2.5.3	<i>Exploration versus exploitation .....</i>	<i>23</i>
2.6	SPIKING CONTROLLERS FOR MOBILE ROBOTS .....	23
2.6.1	<i>Braitenberg vehicles .....</i>	<i>23</i>
2.6.2	<i>Existing spiking controllers for mobile agents .....</i>	<i>24</i>
<b>3</b>	<b>MODELLING .....</b>	<b>25</b>
3.1	ROBOT.....	25
3.1.1	<i>Sonar configuration .....</i>	<i>26</i>
3.1.2	<i>Robot Environment.....</i>	<i>27</i>
3.2	NEURAL CONTROLLER.....	27
3.3	SENSOR AND MOTOR ENCODING .....	29
3.3.1	<i>Sensor readings to spike patterns .....</i>	<i>29</i>
3.3.2	<i>Spike patterns to motor commands .....</i>	<i>31</i>
3.4	LEARNING MODEL .....	31
3.5	TRAINING AND TESTING .....	34
3.5.1	<i>Single object training.....</i>	<i>35</i>
3.5.2	<i>Random walk training .....</i>	<i>36</i>
3.5.3	<i>Defining a stop training condition for synaptic weights.....</i>	<i>37</i>
<b>4</b>	<b>EXPERIMENTS .....</b>	<b>39</b>
4.1	ENFORCING A CONNECTION BY REWARDING RELATIVE SPIKE COUNTS .....	39
4.1.1	<i>Methods .....</i>	<i>39</i>
4.1.2	<i>Results .....</i>	<i>41</i>
4.2	ATTRACTOR BEHAVIOUR .....	43
4.2.1	<i>Effects of neural noise and sonar angle.....</i>	<i>44</i>
4.2.2	<i>Inhibitory effects.....</i>	<i>47</i>
4.2.3	<i>Dynamic training .....</i>	<i>49</i>
4.2.4	<i>Activity dependent scaling.....</i>	<i>51</i>
4.2.5	<i>Network size .....</i>	<i>53</i>
4.2.6	<i>STDP Learning rate .....</i>	<i>55</i>
4.2.7	<i>An optimal controller configuration for learning .....</i>	<i>56</i>

4.2.8 Autonomous training using random walk.....	60
4.2.9 Discussion.....	63
4.3 REPULSIVE BEHAVIOUR .....	63
4.4 ATTRACTOR AND REPULSIVE BEHAVIOUR COMBINED .....	65
4.4.1 Autonomous training using random walk.....	67
<b>5 CONCLUSION AND FUTURE WORK .....</b>	<b>70</b>
5.1 PROJECT REVIEW .....	70
5.2 FUTURE WORK .....	71
<b>6 APPENDIX.....</b>	<b>72</b>
6.1 IMPLEMENTATION AND TOOLKITS .....	72
<b>7 BIBLIOGRAPHY .....</b>	<b>73</b>



## List of figures

FIGURE 2-1 SCHEMATIC NEURON. CREDITS: WIKIMEDIA COMMONS .....	5
FIGURE 2-2 SYNAPTIC CURRENTS.....	5
FIGURE 2-3 TYPES OF NEURONS: A: UNIPOLAR NEURON OF SENSORY GANGLION. B: BIPOLAR NEURON. C: INTERNEURON. D: PYRAMIDAL CELL (CEREBRAL CORTEX) E: MOTOR NEURON [CREDITS: UNIVERSITY OF WESTERN ONTARIO] .....	6
FIGURE 2-4 NEURON ACTION POTENTIAL .....	7
FIGURE 2-5 PRE-POST SYNAPTIC SPIKE PAIRS A: PRE SPIKES BEFORE POST AND CAUSES LTP B: PRE SPIKES AFTER POST AND CAUSES LTD .	8
FIGURE 2-6 ASYMMETRICAL STDP CURVE VERY CLOSE FIRINGS HAVE STRONG EFFECTS ON THE SYNAPTIC WEIGHTS $\Delta s$ . THE DIRECTION OF THE WEIGHT CHANGE DEPENDS ON THE EXACT SPIKE TIMINGS. ....	8
FIGURE 2-7 SYMMETRICAL STDP FUNCTION .....	8
FIGURE 2-8 DOPAMINERGIC PATHWAYS. ADAPTED FROM WIKIPEDIA.ORG .....	10
FIGURE 2-9 DA MODULATION .....	11
FIGURE 2-10 COMPARING NEURON MODELS. FROM [25] .....	14
FIGURE 2-11 PARAMETER EFFECTS AND SIMULATION OF REGULAR SPIKING (EXCITATORY) NEURONS AND FAST SPIKING (INHIBITORY) NEURONS USING THE IZHIKEVICH MODE. PARAMETER EFFECTS FROM [26] .....	15
FIGURE 2-12 ARTIFICIAL NEURON AND SAMPLE ACTIVATION FUNCTIONS. CREDITS: WIKIMEDIA COMMONS.....	16
FIGURE 2-13 ANN ALL-TO-ALL CONNECTIVITY.....	17
FIGURE 2-14 SNN WITH A MODULAR TOPOLOGY. FROM [29].....	17
FIGURE 2-15 SNN WITH SPARSE CONNECTIVITY.....	17
FIGURE 2-16 AGENT ENVIRONMENT INTERFACE. FROM [34] .....	19
FIGURE 2-17 ELIGIBILITY TRACES. FROM [34] .....	20
FIGURE 2-18 BASIC IDEA OF TD LEARNING IN SIMPLE NEURON MODEL. SINGLE NEURON $v$ WITH TWO INPUTS $x_0$ (CS) AND $x_1$ (US) AND SYNAPTIC WEIGHTS $w_0$ AND $w_1$ . INITIALLY THE WEIGHT $w_0$ IS VERY LOW SO THAT THE NEURON ONLY SPIKES WHEN US OCCURS. OVER TIME THROUGH THE ELIGIBILITY TRACE $e$ -TRACE HOWEVER THE CORRELATION $\otimes$ BETWEEN THE RESPONSE $v(t)$ AND CS RESULTS IN A WEIGHT INCREASE SO THAT $v(t)$ SPIKES ALSO AS A RESULT OF A CS INPUT. FROM [35]. ....	21
FIGURE 2-19 ACTOR-CRITIC CONTROL SYSTEM. ADAPTED FROM [38] .....	22
FIGURE 2-20 CIRCUIT IN BASAL GANGLIA SNR: SUBSTANTIA NIGRA PARS RETICULATE, STN: SUBTHALAMIC NUCLEUS, GPe: GLOBUS PALLIDUS PARS EXTERNA .....	22
FIGURE 2-21 BAITENBERG VEHICLES. ADAPTED FROM [41].....	23
FIGURE 3-1 ROBOT MODEL.....	25
FIGURE 3-2 THE ROBOT .....	26
FIGURE 3-3 SONAR CONFIGURATION .....	26
FIGURE 3-4 ROBOT WORLD WITH FOOD SOURCES .....	27
FIGURE 3-5 NEURAL CONTROLLER .....	28
FIGURE 3-6 BENCHMARK TOPOLOGY CONFIGURATION .....	29
FIGURE 3-7 DIFFERENT FORAGER TRAJECTORIES.....	30
FIGURE 3-8 FIRING RATES FOR TRAJECTORY B.....	30
FIGURE 3-9 DIFFERENT $U_{\min}/U_{\max}$ RELATIONS: FROM LEFT TO RIGHT: 5/6, 4/5, 3/4, 2/3.....	31
FIGURE 3-10 EQUILIBRIUM DISTRIBUTION OF SYNAPTIC WEIGHTS.....	32
FIGURE 3-11 STDP AND ELIGIBILITY TRACES A PRE-BEFORE POST SPIKE PAIR GENERATES A POSITIVE ELIGIBILITY TRACE DURING WHICH THE SYNAPSE IS ELIGIBLE FOR AN WEIGHT INCREASE INDUCED BY A REWARD SIGNAL. TAKEN FROM [14]. ....	34
FIGURE 3-12 OVERSHOOTING A TURN .....	35
FIGURE 3-13 RANDOM WALK TRAJECTORY.....	37
FIGURE 3-14 GOAL AND MAXIMUM TRAJECTORY .....	37
FIGURE 3-15 MINIMUM STEERING ANGLE .....	38
FIGURE 3-16 ORIENTATION CHANGE TO MFR DIFFERENCE PER ROBOT ITERATION .....	38
FIGURE 4-1 REWARDING RELATIVE SPIKE COUNTS.....	41
FIGURE 4-2 SPIKE PATTERN OF FIRST 40MS OF LAST TRIAL.....	42
FIGURE 4-3 WEIGHTS $> 3.8$ .....	42
FIGURE 4-4 ADS WITH EQUATION (4-3) .....	43
FIGURE 4-5 ADS WITH EQUATION (4-4) .....	43
FIGURE 4-6 EXPERIMENT 6 RUN FOR 60MIN .....	46
FIGURE 4-7 WEIGHT DISTRIBUTION EXPERIMENT 6.....	47
FIGURE 4-8 CHANGING CONNECTION STRENGTHS IN NEURAL CONTROLLER. THE THICKNESS OF THE LINE REPRESENTS THE STRENGTH OF THE CONNECTION. INHIBITORY CONNECTIONS ARE NOT PLASTIC. ....	47
FIGURE 4-9 WITHOUT MOTOR-INHIBITORY PLASTICITY .....	48

FIGURE 4-10 WITH MOTOR-INHIBITORY PLASTICITY .....	48
FIGURE 4-11 TRAINING WITH MODIFIED INHIBITORY NEURONS.....	48
FIGURE 4-12 FIRING RATES DURING TRAININGS.....	49
FIGURE 4-13 EFFECTS OF TOO LITTLE TRAINING CURRENT .....	50
FIGURE 4-14 WEIGHT DISTRIBUTION START (LEFT) AND END (RIGHT) OF TRAINING .....	50
FIGURE 4-15 CHANGING WEIGHTS AT DYNAMIC TRAINING.....	51
FIGURE 4-16 TRAINING WITH ADS .....	52
FIGURE 4-17 FIRING PATTERNS FOR ADS TRAINING .....	52
FIGURE 4-18 WEIGHT DISTRIBUTION FOR ADS TRAINING .....	52
FIGURE 4-19 NEURON ALLOCATION IN ORIGINAL NETWORK (LEFT) AND 50% SCALED DOWN NETWORK (RIGHT) .....	53
FIGURE 4-20 STABLE LEARNING WITH SMALLER LEARNING RATE AND LESS LTD ASYMMETRY .....	55
FIGURE 4-21 WEIGHT DISTRIBUTION FOR SMALLER LEARNING RATES .....	55
FIGURE 4-22 SWITCHING REWARD FROM POSITIVE TO NEGATIVE AND BACK .....	56
FIGURE 4-23 SPIKE PATTERN OF 1 TRIAL IN EXPERIMENT 6.....	58
FIGURE 4-24 SPIKE PATTERN OF 1 TRIAL IN EXPERIMENT 6 – DETAILS.....	58
FIGURE 4-25 SPIKE PATTERNS OF 1 TRIAL IN THE VERSION WITH LOW INHIBITORY SPIKING.....	59
FIGURE 4-26 SPIKE PATTERNS OF 1 TRIAL IN THE VERSION WITH LOW INHIBITORY SPIKING – DETAILS.....	59
FIGURE 4-27 NETWORK DYNAMICS WITH CURIOSITY BEHAVIOUR. A,B CURIOSITY CS:10%, C:20% .....	63
FIGURE 4-28 RANDOM WALK TRAJECTORY .....	63
FIGURE 4-29 EFFECTS OF LTD TO LTP ASYMMETRY IN AVOIDANCE LEARNING .....	64
FIGURE 4-30 REPULSIVE TRAINING.....	65
FIGURE 4-31 NEURON ALLOCATION.....	66
FIGURE 4-32 TRAJECTORY IN ATTRACTOR AND REPULSIVE BEHAVIOUR .....	67
FIGURE 4-33 TRAINING COMBINED BEHAVIOUR USING RANDOM WALK – 10MINUTES .....	68
FIGURE 4-34 TRAINING COMBINED BEHAVIOUR USING RANDOM WALK - 4 HOURS .....	69

# 1 Introduction

*“What seems astonishing is that a mere three-pound object, made of the same atoms that constitute everything else under the sun, is capable of directing virtually everything that humans have done: flying to the moon and hitting seventy home runs, writing Hamlet and building the Taj Mahal — even unlocking the secrets of the brain itself.”*

-- Joel Havemann, *A Life Shaken*, 2002

## 1.1 Motivation

The brain is the most complex biological structure known to mankind. It can solve difficult problems within a blink of second, controls all the organ systems of the body, and operates 24x7. It accepts and processes a flood of information about the world from its various senses, handles and coordinates physical movement and enables thoughts, dreams, and experience of emotions. The brain together with the spinal cord and the peripheral nerve regulates all facets of conscious and unconscious life. The enormous tasks handled by the brain are reflected in its complex structure of neurons and synapses. The human brain has around 100 billion neurons, 10 billion alone in the cerebral cortex. Each neuron has up to 10'000 connections to other neurons.

Brain complexity can be described at several levels of scale. Looking at the microscale, there are hundreds of types of neurons with a variety in morphology and function. Over 300 distinct electrical behaviours of neurons, i.e. ion channels, and over 100 different neurotransmitters have been identified[1]. Just to accurately simulate a single synapse one would need some 10'000 of differential equations. Synapses are also extremely complex molecular machines that would themselves require thousands of differential equations to simulate just one. There are hundreds of different types of synapses and neurotransmitters and many synapses have more than one neurotransmitter.

Going to larger scale, different neuronal structures and connectivity in the brain give specific regions particular capabilities. The visual cortex, for example, processes input from our eyes, the cerebellum coordinates motor movement. Every area of the brain is specialized for something but they often work together to do a task.

With this amazing complex structure humans are able to make equally amazing complex decisions in the matter of a fraction of a second. For example humans need about 0.1s to visually recognize their mother [2].

Because of its ability to transmit electrochemical signals neurons are often compared to wires and gates of a computer. When we abstract from everything but the electrochemical transmission of signals the brain still has some fundamental differences to a computer in how it operates. It is digital and analog, it is asynchronous, it is fault tolerant, it is reconfigurable, but maybe most importantly in the brain there is no clear separation between memory and processing unit<sup>1</sup>. Instead, processing unit and memory are uniformly distributed and tied together. There seems to be no communication bus bottleneck. Instead,

---

<sup>1</sup> Modern computers are built on the principles of John von Neuman from the 1940s: Memory and processing unit are separated but connected via communication bus.

conduction delays between neurons play an integral part in spike timed communication that would otherwise not be possible.

Given the sheer complexity and differences in its architecture, it seems obvious that up to date or in the near future we have no computer that even remotely can emulate the capabilities of the brain. Dharmendra S. Modha - Principal Investigator from the DARPA program SyNAPSE to reverse engineer the brain's computational power - said in 2009 on the IEEE's 125<sup>th</sup> anniversary [3]:

*"A computer comparable to the human brain would need to be able to perform more than 38 thousand trillion operations per second and hold about 3,584 terabytes of memory."*

As of 2011 the world's fastest supercomputer, the K-Computer, runs at around 8 petaflops [4]. Even if we will eventually be able to match this power, there is still the nature of memory, let alone consciousness which there is nothing comparable built from engineers yet.

That being said, the brain has been subject to intense study through most of the modern times. For much of the early history of neuroscience, this was done by visually examining its anatomy. Neuroscientists have identified and categorised regions and mapped functions to structure. Later computational models were invented and most notable breakthroughs have been obtained by looking at the basic building blocks of the brain, namely the neurons and synapses. When confronted with a system like the mammalian brain with such immense complexity it is paramount to have the right level of abstraction in the model. Here biological realistic neural networks have played an important role to find the right balance between abstraction and biological feasibility.

The fundamental output of the mammalian brain is motor action. The fundamental input comes from sensors that enable us to interact with our environment. The brain processes the sensory information, and takes appropriate actions if necessary. A single sensor-motor loop involves usually several regions of the brain. One instance of such a loop is for example spinal cord, dorsal column, thalamus and neocortex. In order to understand the principles instead of looking at those individual areas, a functional approach is often more appropriate.

Plasticity is one of the most important features of the mammalian brain. The brain is able to continually adapt and rewire itself. If we would wake up one day and the world around us would be upside down first we would be distracted and not able to move in this new environment. After a few hours however the brain would have learned to cope with the new sensor readings and we would be quite capable of acting almost normal. With some exercise we would even be able to master complex actions like riding a bike or skiing down a hill. This might sound like science fiction, but is actually the result of a classical experiment made famous by T. Erismann and I. Kohler in the first half of the 20th century where volunteers were instructed to wear prism goggles during several weeks.

## 1.2 Project aims

Sensor-motor interaction and plasticity of neural networks set the context of this thesis. Spike time dependent plasticity (STDP) has been subject to much research in recent years and is understood as primary cause for changes in synaptic efficiency. Reward modulated STDP via dopamine (DA) is a prime candidate to explain how the non-declarative long term memory in mammalian brain learns.

The approach we follow in this project is to have a simple but realistic enough setting of a motor sensor loop where we can study reward modulated STDP. The motor cortex output will control a simulated wheeled robot, abstracting from the more complex kinematics of a humanoid robot. The robot and the controller operate in a closed feedback loop. First the robot senses new information from the environment and transfers it to the neural network for processing. The neural network processed the

information and then sends a sequence of motor commands back to the robot. The movement of the robot will result in new information that can be gathered by the sensors. This will be a fairly controlled environment and a setup that allows us to concentrate on the learning aspects in the neural controller.

Our main goal is to understand if and how learning processes driven by dopamine modulated STDP can enable the specific behaviours through controlled motor movements. The concrete behaviours are obstacle avoidance and food attraction. In this context we want to examine the dynamics involved in the synaptic modifications, effects of network architecture, neuron modelling, and neuronal coding. Besides STDP we consider other plasticity mechanism especially “synaptic scaling”, which is a regulative process not limited to a single neuron. Further we want to investigate in methods of encoding that happen between the robot and the controller as it does between senses, the spinal cord and the brain, and the rest of the body.

### 1.3 Project contributions

In this project we set out to model a biological inspired neural controller for a simulated robot that is able to learn certain behaviours. The contributions from this work are the following:

- We implement a neural controller in a novel way that is able to successfully learn autonomously in a previously unknown environment. This has been a challenging problem addressed in the literature (e.g. see the paper from Chorley (2008) [5]). Other approaches are using genetic algorithms [6], [7] or different STDP models [8] to learn synaptic connection that control motor movements.
- We show how the easier task of a dedicated training phase can help the robot to quickly adapt synaptic connections that maximise rewards. A dedicated training phase that is commonly used in machine learning problems can enable a neural controller in the robot to quickly learn new behaviours. The advantage of a training phase is that it reduces the noise and randomness of an open environment.
- Relatively little is known about the precise computational roles of different plasticity mechanisms. They often share receptors and are very difficult to measure in a biological setting. Computational models hold therefore a great promise to advance the knowledge in the interplay between different mechanisms of plasticity. We contribute to this area by showing how synaptic scaling influences the dynamics of a learning task. Concretely we implement activity dependent scaling and show how this is regulating the balance between excitation and inhibition during learning.
- We analyse the effects of STDP parameters especially the learning rate on the stability and ability to learn. We find that only with a low learning rate the neural controller can distinguish between random inputs and effective correlations. With a higher learning rate the network is strengthening random connections which then are subject to a higher probability to be further increased due to the competitive nature of STDP.
- We show how the sensor and motor encoding has significant effects on the learning performance of the robot.

## 1.4 Report Structure

We begin this report with a background chapter. Due to the interdisciplinary nature of the thesis between computational neurodynamics, robotics, and reinforcement learning we touch the relevant aspects of each of these fields. In chapter 3 we detail the model and implementation of our neural controller as well as the training and testing methods applied. Chapter 4 covers the experiments we carried out including the results obtained and an evaluation of those. We conclude the report in chapter 5 where we summarize the results and give suggestions of possible extension for future work that could be added to this project.

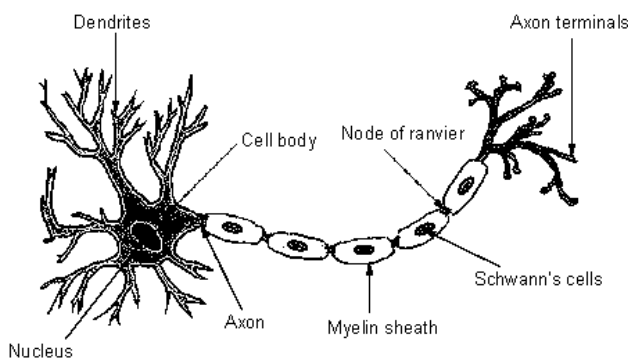
## 2 Background

This chapter briefly lies out the foundations for the following chapters. It reflects the highly interdisciplinary nature of the project including neuroscience, computational neurodynamics, machine learning and robotics. We first briefly go over some of the biological foundations and then proceed to computational models of the project. Next we outline how reinforcement learning gives us an interpretation of the learning process in the brain and finish the chapter with an overview of the mobile agent that we will use.

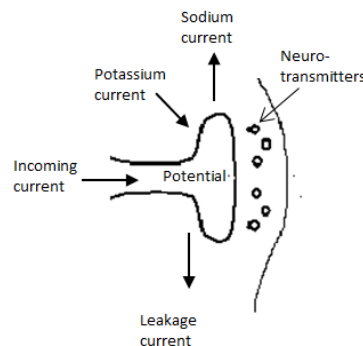
### 2.1 Biological foundations

In modelling we need to decide on a certain abstraction level to investigate the question at hand. There is always a trade-off between biological accuracy and computational feasibility. This section outlines some basic biological principles of neurons, synapses and neural networks relevant for the study.

#### 2.1.1 Neurons and Synapses



**Figure 2-1 Schematic Neuron.** Credits: Wikimedia Commons



**Figure 2-2 Synaptic currents**

Neurons are the basic structural components of the brain. They distinguish themselves from other cells by dendrites and axons. Figure 2-1 shows the structure of a (multipolar) neuron: highly branched dendrites, the cell body or soma, the long axon with myelin sheath that minimises the conduction delay and the axon terminals. The electrical potential that flows through a neuron is generated by the difference of ion concentration between the inside and surrounding liquid of a cell. Electrical signals are transmitted within a neuron from dendrites to axons where they are transformed to chemical signals. A chemical process<sup>2</sup> via neurotransmitters transmits the pulse between synapses of different neurons. For terminology we call the cell that sends signals presynaptic and the cell that receives signals postsynaptic neuron.

Figure 2-2 shows the currents at the synaptic level. Incoming current arrives from the axon. Simultaneously Potassium current is entering into the synapse while Sodium Leakage current is exiting the synapse. The neurotransmitters that are released bind to receptors which determine the effect on

---

<sup>2</sup> Besides chemical synapses but less common exist electrical synapses. There the signal is transferred via a small gap between pre and postsynaptic neurons known as gap junction.

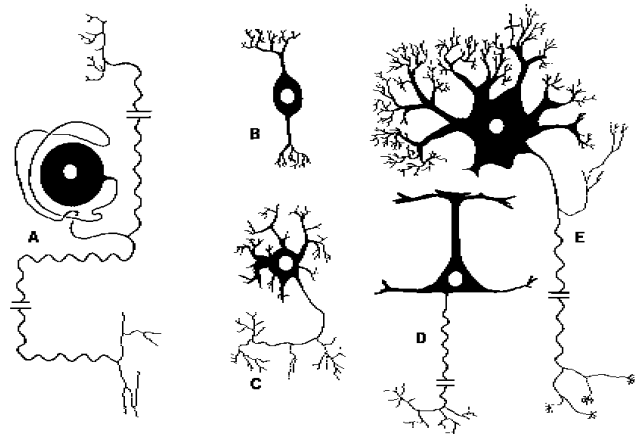
the next neuron. The effects are mainly excitatory or inhibitory<sup>3</sup>. Excitatory neurons increase the activity of the postsynaptic neurons, inhibitory decrease the activity of the neurons they are connected to.

Neurons come in many different shapes and sizes. Neurons also vary in their function and can be classified by the direction, effects on other neurons or spiking behaviour.

Figure 2-3 shows some basic types of neurons. The classification in terms of structure (or polarity) is as follows:

- A: Unipolar: Contains only a single neurite; the dendrite and the axon come from the same process.
- B: Bipolar: Axon and dendrite are on the opposite sides of the soma.
- C-E: Multipolar: Contains more than two dendrites and at least one axon.

Sensory neurons are unipolar and have dendrites on both ends. Motor neurons that control muscle contractions are multipolar and have a cell body on one end, a long axon in the middle and dendrites on the other end. Bipolar neurons are interneurons, or associative neurons that carry information between motor and sensory neurons. Pyramidal cell neurons are also multipolar as are Purkinje cells as other typical example.



**Figure 2-3 Types of neurons:**

A: Unipolar neuron of sensory ganglion. B: Bipolar neuron. C: Interneuron. D: Pyramidal cell (cerebral cortex) E: Motor neuron [Credits: University of Western Ontario]

Directional classification:

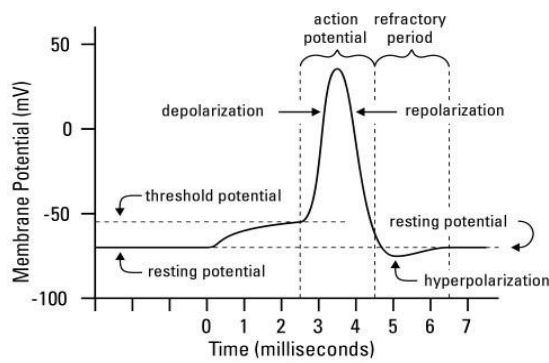
- Sensory neurons (afferent neurons): carry signals from the outer parts of the body (periphery) into the central nervous system.
- Motor neurons (efferent neurons): convey information from the central nervous system to the outer parts (muscles, skin, etc.) of the body.
- Interneurons: connect various neurons within specific regions of the central nervous system.

In terms of electrophysiological properties, three behaviours can be discriminated: regular, phasic or bursting, and fast spiking. More details on this see chapter 2.2.

Figure 2-4 shows the typical dynamics in the membrane potential of a neuron. Through incoming current the voltage of synapses increases. When a certain threshold, the threshold potential, is reached the neuron spikes and sends the signal via neurotransmitters to the neighbouring neuron. This voltage pulse has only a duration of 1-2 milliseconds. The neuron then repolarizes and typically undershoots its resting potential. The neuron has to recover during a refractory period before it can fire again. When no incoming current is present, the neuron tends towards a stable resting potential. A series of action potentials coming from a single neuron is called a spike train.

<sup>3</sup> About 90% of the neurons in the brain release either glutamate or GABA neurotransmitters. Glutamate and GABA have largely consistent effects on their receptors: Most of the glutamate receptors have excitatory and most of the GABA receptors have inhibitory effects.





**Figure 2-4 Neuron Action Potential**

One important aspect of a synapse is its ability to form memory by changing the strength of the connection. The best known form of synaptic plasticity is long-term potentiation (LTP) [9]. LTP was discovered in the hippocampus in 1973 [10] and has been subsequently studied in a variety of species, and a number of different synapses in other regions of the brain. What makes LTP particularly interesting is that it causes the long-term strengthening of the synapses when two neurons are activated simultaneously. At the molecular level the process as we know of today is as follows: The neurotransmitter glutamate is released into the synaptic cleft and binds to AMPA and NMDA receptors. The AMPA receptor is paired with an ion channel and lets sodium ions enter the postsynaptic neuron. The NMDA receptor on the other hand enables calcium ions. When the postsynaptic neuron is at its resting potential this channel is blocked by magnesium and nothing can enter. Only if the dendrite of the postsynaptic neuron depolarizes, the channel is opened for calcium influx. The entry of calcium initiates a second molecular process that ultimately leads to an increase in the number of receptors in the target cell which increases the synaptic efficiency.

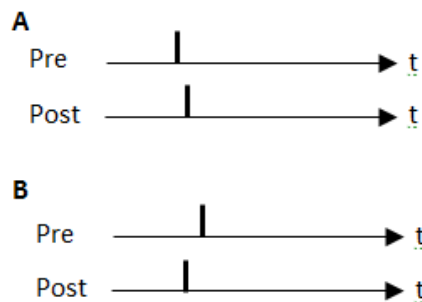
So in order to let calcium enter the postsynaptic dendrite, both neurons have to be activated simultaneously. Learning by rewards, as we will discuss later, is an example of LTP where this simultaneous activation is conditioned by dopamine. Long term depression (LTD) is the opposite of LTP and results in a long-lasting decrease of synaptic efficiency. It is thought to occur via long trains of low-frequency (1 Hz) stimulation [11], or mismatching of pre- and postsynaptic action potentials [12].

Plasticity enables the brain to adapt to changes and learn. LTP/LTD is a form of Hebbian learning [13], which involves associative activity-dependent changes in the impact of a presynaptic spike on the postsynaptic firing probability. The Hebbian learning rule states that *“neurons that wire together fire together”*.

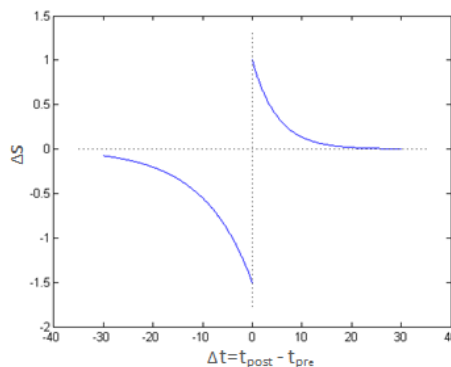
## 2.1.2 Spike-time dependent plasticity (STDP)

There is a variety of plasticity mechanisms affecting synaptic efficiency of excitatory and inhibitory synapses. Spike-time dependent plasticity (STDP) builds on associativity between spike patterns of neurons. It is a form of Hebbian plasticity. On the other hand, synaptic scaling and intrinsic plasticity are a form of homeostatic plasticity that interplays with STDP to form the plasticity in the brain. Here we shall focus on STDP. Homeostatic plasticity will be discussed further in the modelling chapter.

The exact form of STDP varies between the different types of synapses. In its most common form spike timed dependent plasticity suggests that a presynaptic spike just before a postsynaptic spike (pre-post pattern) causes synaptic weights increase; and a presynaptic spike just after the post neuron is spiking (post-pre pattern) causing a long term depression on the synaptic strength between neurons. It is now widely accepted that memory and learning is strongly correlated with STDP (see [14], [15]).



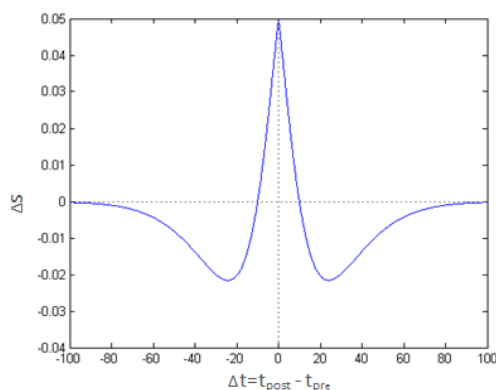
**Figure 2-5 Pre-post synaptic spike pairs**  
A: Pre spikes before post and causes LTP  
B: Pre spikes after post and causes LTD



**Figure 2-6 Asymmetrical STDP Curve**  
Very close firings have strong effects on the synaptic weights  $\Delta s$ . The direction of the weight change depends on the exact spike timings.

Figure 2-5 show the temporal dynamics in the spike pairs: A pre-post spike pair causes LTP, whereas a post-pre spike pair is causing LTD. The next figure, figure 2-6, illustrates the temporal asymmetry between the integral of LTD and LTP. This asymmetry prevents uncontrolled growth, strong destabilizing self-excitatory loops and leads to stable dynamics.

An alternative somewhat less common, symmetrical form for STDP has been found at GABAergic synapses. GABA is the primary inhibitory neurotransmitter. An example of a symmetrical STDP curve is shown below.



**Figure 2-7 Symmetrical STDP function**

Neurons compete among each other for the control of the timing of postsynaptic action potentials. This competitive nature of STDP promotes early selectivity of connections between causal related neurons. The temporal connections that are there first and/or the strongest will become even stronger, whereas the weights of the weaker connection will tend to zero. Such a shifting response to the earlier predictors can be seen for example when animals learn to react to the earliest sign of prey or danger.

Suppose we have a single neuron that receives inputs from a number of presynaptic neurons that fire sequentially. This could be the result of a moving stimulus. At the beginning all synapses have the same weight. Let's assume the postsynaptic neuron fires at some time  $T$  during the sequence of firings from the presynaptic inputs. Now the weights from the synapses that have been activated prior to the firing will strengthen, and the connections to the neurons that have been firing after the postsynaptic spike will weaken. This leads to a temporal shifting of the response so that over time the postsynaptic synapse spikes at an earlier time with STDP.

Some studies showed that LTP occurs if postsynaptic spikes are repeatedly paired with presynaptic spikes that precede them up to 50ms. LTD on the other hand has a time window of 50ms or more depending on the synapse (see [15]).

Song and Abbott [15] showed that this asymmetry allows selective groups of neurons “the teacher group” with correlated firing patterns to direct the development of other non-selective neurons “the student group” with more random firing patterns. This result is shown in simulations with feedforward and recurrent connections and matches experimental evidence in vivo: dynamics show temporally increasing recurrent connections that drive feedforward connections to become stronger. The intracortical pathways can be interpreted as the recurrent neurons that are temporarily increased and then weakened, once strong feedforward pathways become established.

This has important implications e.g. for cortical maps that can be reorganized in the brain as a result of an injury. Neurons that receive sensor information (input neurons) are connected to network neurons – neurons in the cerebral cortex. A loss of input correlation because of lesioned input neurons causes a rapid strengthen of recurrent connections of nearby network neurons. The strengthened recurrent neurons provide an instructive signal that guides the strengthening of new feedforward inputs to the network neurons. It allows a group of neurons that become selective early in the process to guide other neurons to become similar selective.

### 2.1.3 Homeostatic plasticity: synaptic scaling and intrinsic plasticity

Homeostatic plasticity is another form of plasticity. It acts as a negative feedback loop for changes in the neuronal activity and stabilizes activity in a network. STDP as a form of correlation based Hebbian plasticity, fosters competition and has a tendency to destabilize network activity. Once an input is strengthened, its correlation with the postsynaptic activity increases. This leads to additional potentiation. Similarly when a synapse is depressed this will reduce its correlation and the synaptic strength will quickly approach zero. To prevent indefinite growth we need to impose constraints on the strength of synapses. In STDP this is usually done by adding a weight maximum, and not allowing a synapse to switch sign, i.e. for an excitatory synapse to become inhibitory and vice versa.

Modelling homeostatic plasticity is another way to compensate some of the chronic changes in neural activity. This can be achieved via several methods.

- Keeping the sum of the weights constant
- Adjust the threshold for potentiation depending on postsynaptic activity
- Regulate neuronal activity depending on postsynaptic activity
- Scale all synaptic weights depending on postsynaptic activity

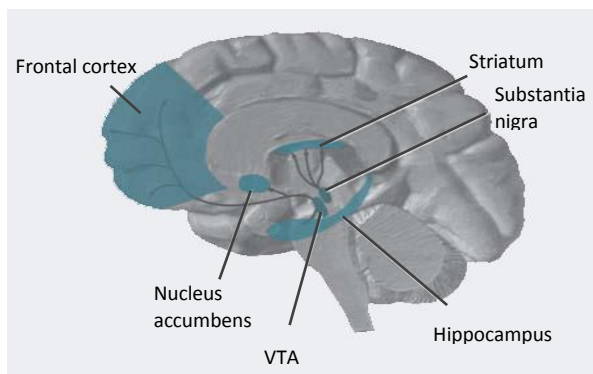
The first and last are forms of synaptic scaling. The other two are cases of intrinsic plasticity (IP). IP changes the excitability of the individual neuron (its transfer function in ANN terms) depending on the firing patterns. In the literature the focus of research has been largely on Hebbian plasticity. Only recently some computational implications of IP have been analysed [16]. The concrete function of IP however remains unknown.

Synaptic scaling is an somewhat more researched field [17], [18] and has been applied to computational models more often. Functionally it is important for the refinements of cortical circuits especially during development. This is the phase when the number and strength of synapses changes as result of experience. The idea is quite simple: The postsynaptic neuron scales the strength of its inputs as a function of a target activity. If the postsynaptic activity is above the target firing rate, excitatory synapses are scaled down, if below the synapses are scaled up. The relative strength of synapses within a neuron is preserved, but the amplitude normalised in order to achieve a certain firing rate.

## 2.1.4 Reward modulated STDP

### 2.1.4.1 Dopaminergic neurons

Motivated behaviour is aimed at achieving a specific outcome and driven by internal states or external events. An internal state might be a physiological need like hunger; an external event might be a threat. Any stimulus that increases the chances that a goal directed response occurs is called a positive reinforcer or reward. Natural rewards include food, water, sex, and fighting. Food for example is a powerful stimulus for a hungry person. The effect of the reward depends on the internal state, specifically satiation or deprivation. One principle of satiation is to maintain the organism's homeostasis. With satiety the positive reinforcing quality changes, but can still be sufficient if its novel, or especially delicious. A reinforcer by contrast is said to be negative if the subject works to avoid the stimulus.



**Figure 2-8 Dopaminergic pathways.** Adapted from Wikipedia.org

In the human brain dopaminergic (DA) neurons are responsible for triggering motivated behaviour. DA neurons are neurons whose primary neurotransmitter is dopamine. They receive inputs from many parts of the brain and the nature of the input defines what we find rewarding.

The primarily region where DA neurons are found is the midbrain in the ventral tegmental area (VTA) and the substantia nigra. From there they project axons to the ventral striatum, the mesolimbic system, the frontal cortex, and the mesocortical system where they release dopamine (see Figure 2-8).

Dopamine regulates attention, motivation, movement, reward, and learning. In small doses dopamine activates D2 receptors which control ongoing thoughts and movements. If something unexpected or rewarding happens large amounts of dopamine are released which activates D1 receptors. D1 receptors stimulate learning and formation of new connections between neurons. Dopamine neurons are also thought to encode the temporal difference prediction error in reinforcement learning. This theory holds especially for positive prediction errors, i.e. when the difference between actual reward minus the expected reward is positive. For the negative case there is currently no agreement in a computational model to explain dopamine in reporting punishment. The key for a dopamine increase seems to be the prediction error. Dopaminergic neurons only fire rewards early in the training (when unexpected). When the animal has learned to expect a reward the motivational effects diminish and dopaminergic neurons cease to fire. The prediction error at a time of the cue is proportional to the predicted probability of a reward (see [19]).

### 2.1.4.2 Conditioning

#### Classical conditioning:

Classical conditioning (also Pavlovian conditioning) is a form of associative learning. In behavioural psychology a natural inherent reflex can be associated with a new reflex, added through learning.

Unconditioned stimulus (US) refers to the stimulus that triggers a certain reaction (unconditioned reaction) without preceded learning. A neutral stimulus that through learning leads to a certain conditioned reaction is called conditioned stimulus (CS). Pavlov discovered in 1927 that when he consistently rang a bell before feeding his dog, the dog started salivating simply in response to the bell. The most effective classical conditioning is when there is only a short delay between CS and US. This type is called forward conditioning. The opposite is backward conditioning. Here the CS acts as a signal that the US has ended. Backwards conditioning tends to have inhibitory effects.

#### Instrumental conditioning

In instrumental conditioning an agent learns a stimulus-response pattern through rewards. It modifies behaviour in anticipation of a reward. The actions of the agent influence the delivery of the reward. This is in contrast to classical conditioning where the reward delivery is independent of the chosen actions of an agent.

There are four possibilities how a certain behaviour can be influenced:

- *Positive reinforcement*: A certain behaviour is increased because favourable events/outcomes are presented afterwards
- *Negative reinforcement*: Unfavourable events/outcomes are removed after behaviour, also leading to an increase of the behaviour. Example: avoid the aversive stimulation of having lights in one's eye by shielding the eyes when in the sunlight.
- *Positive punishment*: An unfavourable event is presented after the behaviour.
- *Negative punishment*: A favourable event is removed after the behaviour.

### 2.1.4.3 Dopamine modulated STDP

In Pavlovian and instrumental conditioning rewards typically come seconds after the reward-triggering actions. This is known as “distal reward” or “credit assignment problem” where the brain has to find out which of the firing patterns were responsible for the rewarding actions. When the reward arrives, the patterns that caused the rewards are no longer there. During the period until the reward arrives the synapses are typically not silent.

Global extracellular release of dopamine enhances LTP and LTD if arrived within 15-25 seconds after the spike pairs have fired. This slow synaptic process determines the eligibility trace for correlated spikes: a short-term memory process that decays slowly over time. This process marks the synapse as eligible for learning. If an unexpected good or bad event occurs while the trace is non-zero, then the synapse is assigned credit accordingly and an enhancement of LTP will happen, see [14].

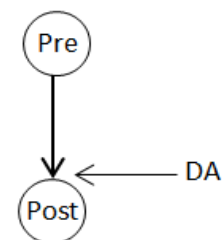


Figure 2-9 DA modulation

Naturally there are many other pairs of neurons that fire nearly coincident spikes by chance just before the reward so that the corresponding synapses are also modified. However, the order of the firings of these neurons is random, and will cancel out in the long run, actually resulting in a net decrease of the synaptic weight because the  $LTD > LTP$ . The synaptic tags are therefore insensitive to random ongoing activity during the waiting period. Izhikevich has demonstrated this in his

study on DA modulated STDP and precise spike timings [14]. One main functional role of noise is to maintain a suitable level of spontaneous firing, since if a neuron does not fire, it cannot find out whether it will be rewarded. The noise should vary from trial to trial in order to explore which firing patterns should be rewarded. On the other hand the noise must not be too dominant to give the real correlations a chance to dominate the noise: If the neuron primary fires because of the noise, STDP does not have the required effect [20].

### 2.1.5 Sensori-motor loops in the brain

The brain is part of the body that has evolved to control its movement. Bodily movement is the fundamental function. To interact with the environment the body has a range of senses. Images, for example, are encoded in the retina to the optical nerve which transfers it the visual cortex. There the brain begins to reconstitute the image from the receptive fields of the cells of the retina. Many of areas involved in visual processing are reciprocal connected, forming feedback loops that demonstrate the highly interlinked nature of the brain.

For the brain to execute goal directed movements it must constantly evaluate the environment and internal state of the body. The sequences of force, amplitude, speed and contractions of every muscle concerned in doing a seemingly simple movement are highly complex. The motor cortex is the most involved area for body movements. Coming from the motor cortex - together with the sensory and prefrontal cortex acting as “consulting capacity” -, the direction of movements is organised in a hierarchical fashion. The hierarchy involves basal ganglia, the pons, cerebellum, the spinal cord, and finally the target muscles where neurons connected directly to the muscles cause them to contract and release.

## 2.2 Neuron Models

A good theoretical model of a complex system should emphasize the most important features and ignore inessential details. Up to today it remains unclear what level of single-cell modelling is appropriate in order to understand the dynamics and computations that are carried out in such a complex system as the mammalian brain. For this study we are looking for a model suitable to describe the dynamics and plasticity changes in a moderately heterogeneous small network of few layers of neurons. To do this we must decide for a model that is a good compromise between biological plausibility and computational feasibility. For a detailed discussion see [21], [22] where this chapter has been largely based on.

The *cable theory* of neurons models dendrites and axons as cylinders composed of segments with capacitances resistances combined in parallel. It uses partially differential equations to describe the current flow in a continuous passive (i.e. membrane resistance being voltage-independent) dendritic tree. In classical cable theory there are constraints on the branching of the tree and several other behavioural constraints outside the scope of this chapter. The problem is that the model becomes complex and computationally expensive when those idealistic constraints on the structure and behaviour are lifted.

*Compartmental models* don't have such morphological representation constraints. They can model any desired topology with arbitrary branches and lengths. Instead of using continuous differential equations it divides the system into small segments. Nonuniformity in physical properties occurs between compartments rather than within. The model consists of a set of ordinary differential equations that require numerical integration.

*Single-compartmental, conductance based models* are a simplification of detailed compartmental models and focus on the different ionic currents and voltage traces. They are itself a large class ranging from the detailed Hodgkin-Huxley, to the phenomenological Leaky-Integrate-and-Fire and the Izhikevich neuron model. Single-compartmental models can produce many different spike patterns such as phasic, chattering, tonic and adaptive spiking and have been simulated from 100'000 (Hudgkin-Huxley) up to  $10^{11}$  neurons (Izhikevich) [23].

## 2.2.1 Hudgkin-Huxley model

For the goal of modelling the transmission of action potentials in neurons the Hudgkin-Huxley model [24] is the most prominent in terms of biological accuracy. It consists of the following four differential equations with biophysically meaningful and measurable parameters describing the membrane potential, activation of Na and K currents and the inactivation of Cl current.

$$C \frac{dv}{dt} = - \sum_k I_k + I \quad (2-1)$$

where C is the capacitance of the neuron. The sum over sodium, potassium and leakage currents is given by

$$\sum_k I_k = g_{Na} m^3 h (v - E_{Na}) + g_K n^4 (v - E_K) + (v - E_L)$$

with the parameters constants  $E_{Na}, E_K, E_L, g_{Na}, g_K, g_L$ . The remaining terms m, h, and n are defined by:

$$\frac{dm}{dt} = \alpha_m(v)(1 - m) - \beta_m(v)m \quad (2-2)$$

$$\frac{dn}{dt} = \alpha_n(v)(1 - n) - \beta_n(v)n \quad (2-3)$$

$$\frac{dh}{dt} = \alpha_h(v)(1 - h) - \beta_h(v)h \quad (2-4)$$

with parameters:

$$\alpha_m = (2.5 - 0.1v)/(e^{(2.5-0.2v)} - 1)$$

$$\beta_m = 4e^{-v/18}$$

$$\alpha_n = (0.1 - 0.1v)/(e^{1-0.1v} - 1)$$

$$\beta_n = 0.125e^{-v/80}$$

$$\alpha_h = 0.07e^{-v/30}$$

$$\beta_h = 1/(e^{(3-0.1v)} + 1)$$

The accuracy of the Hudgkin-Huxley model comes at a price of high computational costs. Four differential equations must be solved for every simulation step of the model resulting in 1200 operations per millisecond [25]. Also for most applications intracellular concentrations of sodium, potassium or chloride is irrelevant. Thus, at least for larger networks there is a need for a computational simplified model.

## 2.2.2 Integrate-and-Fire models

Integrate and fire models are the most simplified models in the single-compartment class. They abstract away much of the biological details. The before spike sub-threshold is approximated by a differential equation and the action potential is set explicitly after the membrane potential reaches a threshold. After that, the neuron is explicitly reset to a refractory value. Integrate and fire models come in various versions. The most commonly used are the leaky integrate-and-fire (LIF) and the quadratic integrate-and-fire (QIF) model. Let's look at the LIF model. It is defined by the following equation:

$$\tau \frac{dv}{dt} = v_r - v + RI \quad (2-5)$$

Where  $\tau$  and  $R$  are constants,  $v_r$  is the resting potential, and  $I$  is the dendritic current. The reset rule is given by

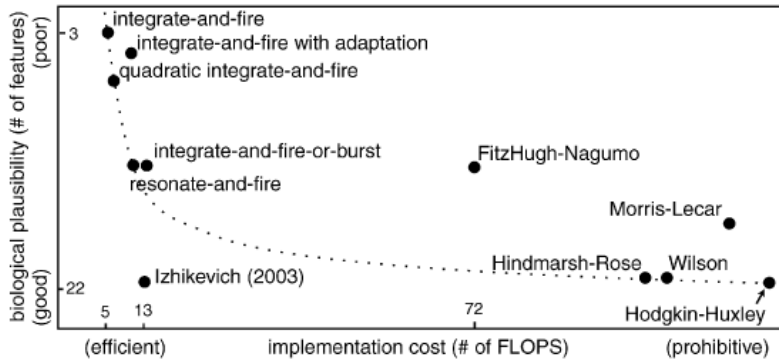
$$\text{if } v \geq \vartheta \text{ then } v \leftarrow v_r \quad (2-6)$$

A common value for  $\vartheta$  is -50mV.

The QIF model describes the sub-threshold profile more accurately and contains quadratic terms. Therefore the QIF model is computational more expensive than the LIF model.

## 2.2.3 Izhikevich model

As mentioned, the main issue with the Hodgkin-Huxley model is that this it is only computational feasible in small simulations. Alternatively integrate-and-fire neurons require less computation, but fail to reproduce the rich dynamics shown by cortical neurons. Figure 2-10, taken from [25], compares those two and other popular neuron models in terms of biological plausibility and implementation costs measured by number of FLOPS. FLOPS is an approximated number of mathematical operations required in the model to simulate one unit of time.



**Figure 2-10 Comparing Neuron Models.** From [25]

The Izhikevich neuron model is a phenomenological model. The parameters do not have a biological interpretation. Instead the focus is in reproducing the richness and complexity of several of spike patterns encountered in the mammalian brain. In [25] Izhikevich demonstrates how his model can reproduce 20 of the most frequently encountered neuro-computational features of cortical neurons. He achieves this by different parameter settings. Interestingly for the biological more accurate Hodgkin-Huxley model parameters to reproduce 3 out of the 20 patterns could not be found in his study, although the model should theoretically be able to reproduce all of them.



In the Izhikevich model there is only one non-linear term in the two differential equations defining the model:

$$\dot{v} = 0.004v^2 + 5v + 140 - u + I \quad (2-7)$$

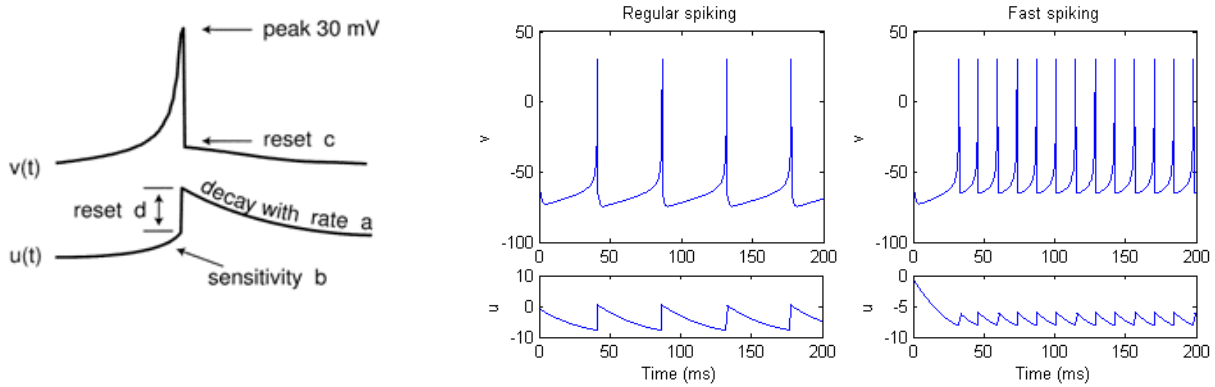
$$\dot{u} = a(bv - u) \quad (2-8)$$

The reset rule after a spike occurred is given by:

$$\text{if } v \geq 30 \text{ then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \quad (2-9)$$

where  $v$  denotes the membrane potential,  $\dot{v}$  the change in the membrane potential over time,  $I$  is the incoming current and  $u$  the recovery variable. The recovery variable accounts for the activation of K<sup>+</sup> (potassium) and inactivation of Na<sup>+</sup> (sodium) currents, providing negative feedback to  $v$ . With the parameters  $a$ ,  $b$ ,  $c$ , and  $d$  a wide variety of realistic neuron behaviours can be simulated. Each parameter has specific effects on the dynamics of the neuron, see figure 2-11. The parameter  $a$  represents the speed of recovery of the membrane after the spike. Smaller values result in slower recovery. The parameter  $b$  determines the sensibility of the neuron to the fluctuation of the voltage. Greater values couple the two terms  $v$  and  $u$  stronger resulting in low threshold spiking dynamics. Finally  $c$  and  $d$  denote the after-spike reset values for  $v$  and  $u$  respectively. This model can simulate regular spiking, bursting or fast spiking neurons. Although mammalian brain neurons can be classified into some additional types it is often sufficient to model excitatory neurons as regular spiking and inhibitory as fast spiking neurons [26].

Typical parameter values for regular spiking neurons are  $a = 0.02$ ,  $b = 0.2$ ,  $c = -65\text{mV}$  and  $d = 8$ , and for fast spiking neurons  $a = 0.1$ ,  $d = 2$  while the other parameters remain the same as for regular spiking.



**Figure 2-11** Parameter effects and simulation of regular spiking (excitatory) neurons and fast spiking (inhibitory) neurons using the Izhikevich model. Parameter effects from [26]

### 2.2.3.1 Numerical integration

The ordinary differential equations (ODE) (2-7) and (2-8) are an example of an initial value problem which can be solved using numerical methods. In his original model [26] Izhikevich uses the Euler method for numerical integration of the ODEs. Given an initial value of  $y(t)$  and time step  $\delta t$  a differential  $\dot{y}$  is approximated by

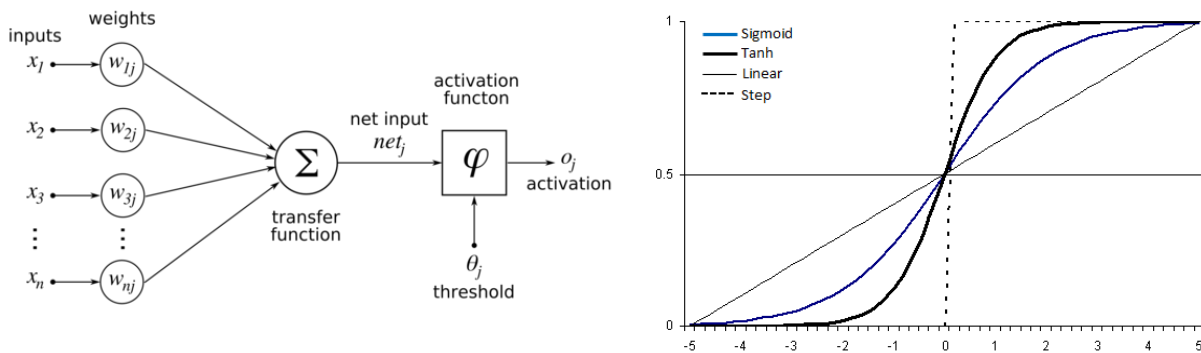
$$y(t + \delta t) \approx y(t) + \delta t \dot{y}(t) \quad (2-10)$$

The Euler method is computational efficient requiring to compute each equation only once per time step. The drawback is that the Euler method is less accurate than other numerical methods, such as the Runge-Kutta method. Recent studies [27] found that for regular spiking neurons the Euler method provides an efficient way to solve the Izhikevich model. The study assesses efficiency in terms of the accuracy/speed trade-off by comparing the Euler method with a so called zero-order hold approximation (ZOH) also used to solve Hodgkin-Huxley type models [28]. For fast spiking neurons however, the ZOH model provided better efficiency.

In this thesis we will use Izhikevich neuron model. Since most of the neurons will be regular spiking excitatory neurons we will use the Euler method for numerical integration. We calculate the neuron variables  $v$  and  $u$  each millisecond and use an Euler step size of 0.25ms for integration.

## 2.3 Spiking Neural Networks

Spiking neural networks (SNN) are the third generation of neural networks in an evolution models with increasing realism of neural simulation. The first generation was based on McCulloch-Pitts threshold neurons, also referred to as perceptrons or threshold-gates. Hopfield nets and Boltzmann machines are neural network models of this generation. The second generation applied continuous activation functions, such as a sigmoid function to their inputs. Recurrent and feedforward nets as well as networks of radial basis function units are examples of this second generation of neural nets. In the literature these networks are referred to artificial neural networks (ANN). ANN's have a very simplistic activation function compared to real synapses. Their primary goal is to solve artificial intelligence problems. Biological plausibility is of secondary nature for ANN's.



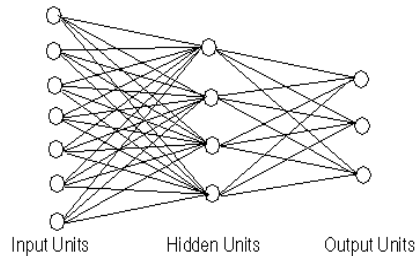
**Figure 2-12 Artificial Neuron and sample activation functions.**

Credits: Wikimedia commons.

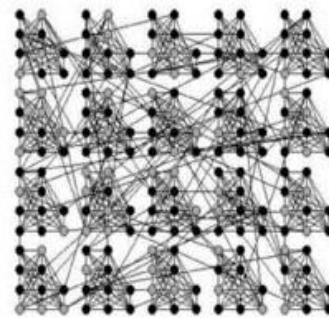
In contrast to ANN's SNN incorporate spatial-temporal timing into their model. The equivalent to an activation function in ANN's is the neuron model which strives to model the actual neuron behaviour. SNN's can carry out analog computation not only under a rate code but also under temporal codes. It has been showed that SNN can simulate arbitrary feedforward sigmoidal neural nets and thus approximate any continuous function. So besides representing a biological more realistic model than the classical networks, SNN's also exhibit a more powerful computational model.

In ANN the topology of the network is primarily problem specific. The network of a spiking neural network however, is driven by an abstracted representation of actual brain structures. The images below shows a typical topology for an ANN with one input, one hidden and one output layer versus a two SNN topologies. The SNN topologies are a modular network and a network with an excitatory and inhibitory layer having layer specific sparse connectivity. In ANN the connectivity is modelled all-to-all. The weights of the connections will be determined during the training of the network. In SNN the

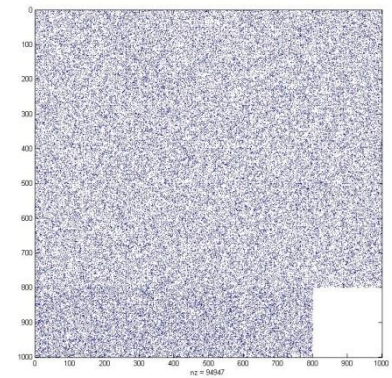
connectivity is a design choice that depends on biological relevance. Human brain generally resemble to a modular network structure that is highly intra-connected within modules (or say brain regions) but sparsely interconnected between modules.



**Figure 2-13 ANN all-to-all connectivity**



**Figure 2-14 SNN with a modular topology.** From [29]



**Figure 2-15 SNN with sparse connectivity**

Apart from the neuron model, the resulting spike timing effects and the topology there are several other discriminating factors between artificial neural networks and spiking neural networks. The most important features that discriminate SNN from ANN in the context of this study are as follows:

- Synaptic weights can change via plasticity mechanisms.
- Neurons and neuron layers (mostly excitatory and inhibitory) have heterogeneous signalling characteristics. This is because neuron populations in real life depict rich dynamics and variety.
- In SNN usually external noise current has to be taken into account. Biological neural networks always have a certain amount of activity independent of concrete/modelled input. The source of this activity might come from feedback loops or other complex dynamics that are abstracted out of the model. This external current is modelled as noise current fed into the various layers.
- Conduction delays: The speed of communication between real neurons depends on the distance and myelination of the involved axons. The conduction delay is the time it takes for an action potential to propagate from the cell body along the axon to the synapse, crossing the synaptic gap onto the dendrites and cell body of the neighbouring neuron. SNN define conduction delays that might be different per connection.
- Scaling factors: Usually it is not practicable to simulate a realistic amount of neurons. To emulate a larger number of pre-synaptic neurons – for example for a sensory layer - the incoming dendritic current might be scaled by a constant factor. Individual neurons can then be interpreted as the average responses of a much larger set of neurons.

## 2.4 Neural coding

Even though action potentials between neurons can vary in duration, amplitude and shape, they are usually treated as uniform events. A series of action potentials, or spike trains is therefore characterised by its temporal pattern and spike rate. Neural coding deals with the question how the mammalian brain encodes stimuli from the environment into such spike trains and decodes them after processing into motor actions or other outputs.

The question of neural coding is a very active research field and there is no agreement yet on a method how to transform an analog stimulus input into a digital spike pattern and vice versa. We list the most commonly used schemes below:

- *Rate coding*: Rate based models assume that the information in the neural network is contained largely in the average firing rate of a neuron measured over a specific period of time. Neural responses are treated statistically and the exact time patterns are ignored. The rate can refer to either the average firing rate over time, the average spiking rate over several repetitions of an experiment, or the average over a population of neurons. Rate models are highly robust with respect to noise. Rate based models were the quasi standard for neural modelling up to about 1990. A common critique for rate based models is that they require a considerable amount of (simulation) time and amount of neurons to obtain a statistically accurate mean value. According to Gautrais et al. to encode a single analogue value to an accuracy of  $\pm 10$  Hz one would need no less than 281 independent neurons [30]. Another issue is that rate coding ignores any temporal structure in the firing patterns. Time-dependent firing rate is an experimental procedure that addresses this concern to some extent by including relative spike timings: the average number of spikes (during trials) appearing within a short interval is divided by the interval length.
- *Temporal coding*: These are models where exact spike timings are important, for example in STDP. Each spike is treated independently of all other spikes. The time dependent spiking defines the temporal coding.
- *Rank order coding*: Rank order coding has been proposed as a potential encoding scheme how the retina encodes images [31]. Instead of an exact latency coding scheme, rank order coding is based on relative arrival times of incoming spikes at a target neuron. Each order is assumed to encode a specific value related to the original stimulus. The amount of information that can be transmitted with such a code grows factorial, e.g. when we have 10 input neurons we can encode  $10! = 3'628'800$  values. An advantage of this method is that it is relatively resistant to noise. This has advantages for example in winner takes all scenarios where we are just interested in the most active neuron/stimulus.
- *Population coding*: Stimulus is represented by the activity of a group of neurons. For some set of inputs each neuron has a distribution of responses and the responses of many neurons combined can encode the value of the inputs. Experimental studies have shown that this encoding scheme is used in some sensor and motor area of the brain [32], [33]. A common approach for position coding such as joint or eye positions is to assign for each neuron position values to a firing rate. In an experiment by Georgopoulos et al. [32], a monkey was trained to move a joystick in a number of directions. He showed that motor cortex neurons responded the most during movements in their preferred direction. If each neuron represents a movement towards a certain direction the vector sum of all neurons points in the direction of the motion. This encoding scheme is referred to as population vector and is an example of simple averaging.

The two key problems in any coding scheme are:

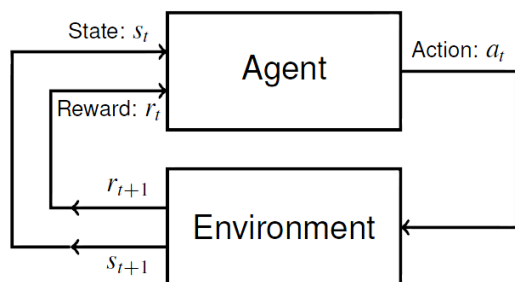
- a) Noisy neural responses: The same stimulus can produce different neuronal responses given the stochastic nature of the neuron model.
- b) Noisy stimulus: When we see an obstacle in bright sunlight or at dawn, the same object produces different visual stimulus but eventually has to result in the same motor action.

Population encodings deal with noisy responses by treating the neural activity as a stochastic variable. A typical population code represents a population of neurons with a Gaussian tuning curve that describes the mean firing rates. The mean of the tuning curve indicates the neuron with the highest sensitivity for any given input value. When neuronal activity patterns are assumed to represent a single value per variable at any given time, decoding can be done with maximum likelihood estimation or a Bayesian approach. Alternatively a population can be thought of encoding a probability distribution over a variable of interest. This would allow extracting both the variable and its uncertainty from the neural activity.

## 2.5 Reinforcement Learning

Many events encountered by an agent in an environment are linked in a temporal causal way. For example foraging precedes smell, which itself precedes taste of a caught prey. Or, as discussed in chapter 2.1.4.2, the bell in Pavlo's classical conditioning experiment precedes the feeding. The list of casual events or actions that occur in a correlated temporal fashion during the lifetime of an agent is infinite. Clearly agents that have the ability to response earlier to events have an evolutionary advantage. An agent has therefore a major incentive to learn to correctly interpret a certain temporal sequence and deduce appropriate actions. The decision-making process by which animals and humans select actions in the face of rewards is one of the fundamental questions in behavioural neuroscience. Behavioural psychology investigates this subject by conditioning. In the neural computation field, reinforcement learning provides a computational framework for such conditioned behaviour.

Reinforcement learning is learning by interacting with the environment. Instead of explicitly taught, an agent learns from consequences of its actions, which are delivered as rewards (that can be positive or negative). The goal of the agent is to maximise the expected future rewards it receives over its lifetime. The action, the states and the rewards do not need to be deterministic and the model of the environment does not need to be a complete one. The figure below shows a schematic view of the agent environment interface.



**Figure 2-16 Agent environment interface.** From [34]

At each time step,  $t$ , the environment is in some state,  $s_t$ . The agent perceives state  $s_t$  at time  $t$  and chooses action  $a_t$ . Up to the next timestep  $t+1$  the world changes to state  $s_{t+1}$ . At time  $t+1$  the agent perceives state  $s_{t+1}$  and gets reward  $r_{t+1}$ . A policy function  $\pi(s, a)$  - which can be deterministic or probabilistic - determines the action for a certain state. Given an action and a current state, a transition function  $t(s, a, s')$  defines the next state. Again, this function can be probabilistic or deterministic. In this setup, reinforcement learning tries to solve the following two related problems:

- The prediction problem: The agent needs to learn the value function for a certain state.
- The control problem: Finding an optimal policy that maximises the reward for each state.

There are three algorithmic approaches to solve these problems: dynamic programming, monte carlo methods, and temporal difference learning. Temporal-difference (TD) learning combines aspects of dynamic programming and monte carlo methods and also takes timing of different events into account. TD learning is currently one of the most actively researched areas within reinforcement learning. The TD learning models can be implemented in neuronal models and many of its ideas base on neuroscience evidence.

### 2.5.1 Temporal difference learning

In temporal-difference (TD) learning changes in predictions over successive time steps drive the learning process. The following is a summary of different aspects of TD learning in the context of rewards and motivation, largely based on [34-37].

Let's consider a sequence of states  $\tau = s_t, r_t, s_{t+1}, r_{t+1}, \dots, s_N, r_N$ . The total return to be expected in the future from state  $s_t$  is thus  $R_t = \sum_{k=1}^N \gamma^k r_k$ , where  $\gamma$  is the discount factor, normally set as the geometric discount  $\gamma_k = \gamma^k$ , with  $0 < \gamma < 1$  (future rewards are less valuable). The value of a state is equivalent to its expected return and is given by

$$V^\pi(s) = E\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid \pi, s_t = s\right) \quad (2-11)$$

where  $s_t$  is the state at time  $t$ ,  $\pi$  the policy,  $r_t$  the reward at time  $t$ ,  $\gamma$  the discount factor, and  $E$  the expectation function. The value of state  $s$  can iteratively be updated by

$$\hat{V}'(s_t) \leftarrow \hat{V}(s_t) + \alpha[R_t - \hat{V}(s_t)] \quad (2-12)$$

where  $R_t$  is the trace of aggregated rewards up to time  $t$  and  $\alpha$  is the learning rate  $0 < \alpha \leq 1$ . If the running estimation  $\hat{V}(s_t)$  correctly estimates the reward sample  $R_t$  the update in equation (2-12) is zero. This method requires to wait until we reach a terminal state. Instead of waiting we can estimate  $R_t$  by

$$E(R_t) = E(r_{t+1}) + \gamma \hat{V}(s_{t+1}) \quad (2-13)$$

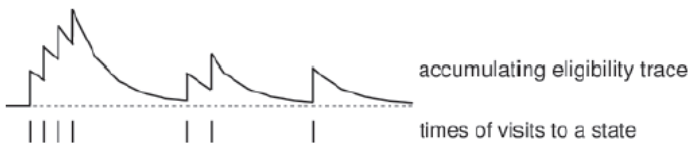
giving the following temporal difference TD(0) procedure equation:

$$\hat{V}'(s_t) \leftarrow \hat{V}(s_t) + \alpha[r_{t+1} + \gamma \hat{V}(s_{t+1}) - \hat{V}(s_t)] \quad (2-14)$$

The term in the square brackets of equation (2-14) is the temporal difference prediction error denoted as:

$$\delta_t = [r_{t+1} + \gamma \hat{V}(s_{t+1}) - \hat{V}(s_t)] \quad (2-15)$$

The prediction error indicates how far the current prediction deviates from condition (2-13). In equation (2-14) no previous visited states are considered by evaluating state  $s_t$ . We can fix this by introducing eligibility traces. Each time a state is visited, a short-term memory process or a trace is initiated which decays gradually over time.



**Figure 2-17 Eligibility traces.** From [34]

Suppose the eligibility trace at the current visited state is given  $E_t = 1$  and decays gradually along the next states visited. This provides us with the so-called TD( $\lambda$ ) algorithm with  $0 \leq \lambda \leq 1$ . If  $\lambda = 0$  only the current state is considered, if  $\lambda = 1$  all previous predictions are updated equally.

$$\hat{V}'(s_{t+j}) \leftarrow \hat{V}(s_{t+j}) + \alpha \delta_j E_{t+j} \quad (2-16)$$

An eligibility trace therefore marks which states have been visited recently. It indicates how much a state is eligible for learning. The state is assigned credit if an unexpectedly good or bad event occurs when the trace is not zero.

In recent years reinforcement learning models such as the TD( $\lambda$ ) algorithms have been applied to a wide range of biological and behavioural data. It emerged reward-based learning can be explained with neuron models and do not contradict correlation based learning models, like STDP.

To define the TD-formalism in a neuronal way we replace “states” with “time-steps”. Then we introduce a “prediction neuron”  $v_t$  that replaces the valuation of state equation (2-13) as follows

$$v_t \sim r_{t+1} + \gamma v_{t+1} \quad (2-17)$$

The TD(0) equation now becomes

$$v_t \leftarrow v_t + \alpha [r_{t+1} + \gamma v_{t+1} - v_t] \quad (2-18)$$

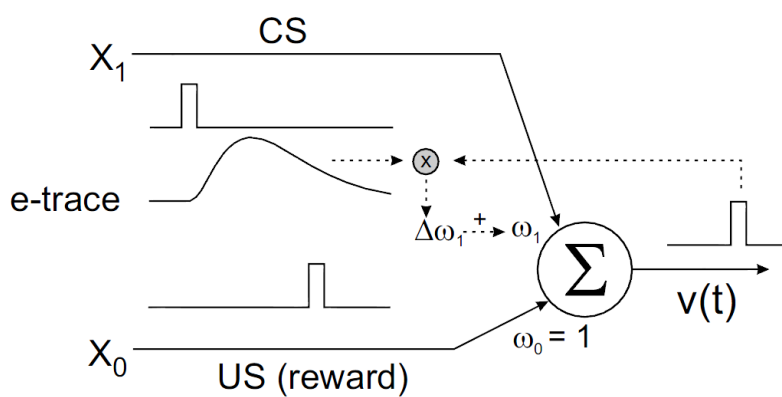
and instead of the value update iteration we define an iterative synaptic weight update function

$$w_i \leftarrow w_i + \alpha \delta_j E_{i,t+j} \quad (2-19)$$

where  $w_i$  is the weight of synapse  $i$  and  $E_{i,t+j}$  is the eligibility trace associated with stimulus  $x_i$ .

This formalism (2-19) can be applied to the classical conditioning problem of behavioural psychology. There we have the unconditioned stimulus (US) that is preceded by the conditioned stimulus (CS). The CS – a bell in Pavlo’s dog experiment – after some training time predicts the US (food) and the dog starts salivating already in response to the CS. Since the action of salivating does not influence the presentation of the US, this is an open-loop architecture.

The figure below illustrates a simple neuron  $v$  with a CS and US stimulus input and an output. The CS precedes the reward and strengthens its synapse with the output neuron during learning. Through the eligibility trace, e-trace, which is non-zero at the time when US occurs it is possible to correlate the two inputs  $x_0$  and  $x_1$  and increase the synaptic weight by  $\Delta w_1$ . After learning the neuron reacts to the stimulus CS as well as US.



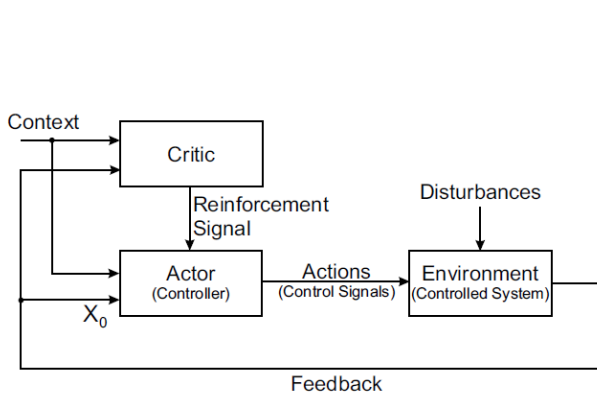
**Figure 2-18 Basic Idea of TD learning in simple neuron model.**

Single neuron  $v$  with two inputs  $X_0$  (CS) and  $X_1$  (US) and synaptic weights  $w_0$  and  $w_1$ . Initially the weight  $w_0$  is very low so that the neuron only spikes when US occurs. Over time through the eligibility trace e-trace however the correlation  $\otimes$  between the response  $v(t)$  and CS results in a weight increase so that  $v(t)$  spikes also as a result of a CS input. From [35].

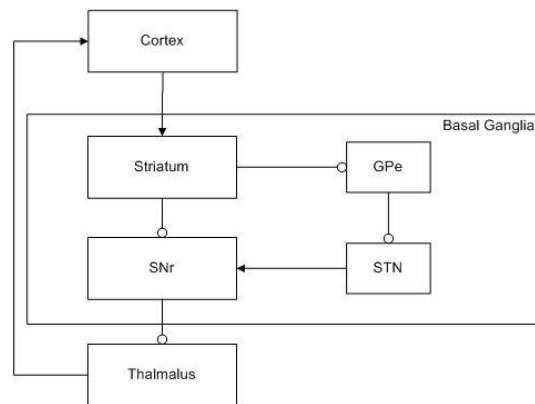
In most cases actions of a biological actor do influence its sensor inputs. When we think of pain and pleasure as rewards this can happen directly at the sensor level. Instrumental conditioning is such a



closed-loop architecture. In a conventional feedback control system a controller provides control signals to a system which is influenced by disturbances. Feedback from the system allows the controller to adjust its signals. In an actor-critic system additionally there is a critic that evaluates the actor providing a reinforcement signal, the TD-error. If the reinforcement signal is positive the actions should be strengthened if its negative actions should be weakened. Figure 2-19 shows the system layout.



**Figure 2-19 Actor-Critic control system.** Adapted from [38]



**Figure 2-20 Circuit in Basal Ganglia**

SNr: Substantia nigra pars reticulata, STN: subthalamic nucleus, GPe: globus pallidus pars externa

Several attempts have been made to map this actor-critic architecture to the mammalian brain structure. There is now broad evidence that the phasic firing of dopaminergic neurons relate to reward prediction errors [14], [34], [39]. The input received by dopaminergic neurons seems to contain information about current rewards and the predictive value of the current state  $V(s_t)$ . The dopaminergic circuit uses this information to then compute a temporal difference reward prediction error. Dopamine provides target neurons with signals appropriate for learning and reward optimizing actions. The dopamine system could therefore be the critic that issues the reinforcement signal. Most attempts to map this structure have been focused on basal ganglia and the cortex. Basal ganglia is a richly interconnected nuclei situated at the base of the forebrain. Its functions are voluntary motor control, procedural learning related to routing behaviours, eye movements, cognitive and emotional functions, action selection, and involvement in the reward system. Figure 2-20 shows the basal ganglia circuit with excitatory connections depicted as arrows and inhibitory connections as rounded end. There is no agreement in the literature neither about the exact mapping of actor and critic nor on the pathways forming the connections.

## 2.5.2 Reward structure

In a supervised learning context a teacher might evaluate the actions of an agent. If feedback only comes from the environment at the sensors of an actor, the feedback is called non-evaluative. Note that the environment does per se not provide any evaluation. In reinforcement learning however feedback is evaluative. Rewards can come from an external or an internal source. External evaluative feedback refers to a teacher that judges certain actions. It requires detailed knowledge of the world of the agent. In Pavlo's dog classical conditioning experiment rewards are delivered externally. Generally this setting is a very limited and clearly does not apply to most biological actors.

With Internal feedback the actor itself provides the value of the rewards. The actor is therefore critique as well as actor. The question is how does the critique know how to criticise? In simple settings like food is rewarding and pain is not this might not be a problem. But when looking at slightly more complex situations with for example conflicting goals this becomes challenging. A designer must define rules for all those situations. If the world-model does not match the model of the agent this becomes a problem



(this is called “Frame problem”, see [40]). A possible solution to this employs an evolutionary approach. In this approach the actor bootstraps his own reward function.

### 2.5.3 Exploration versus exploitation

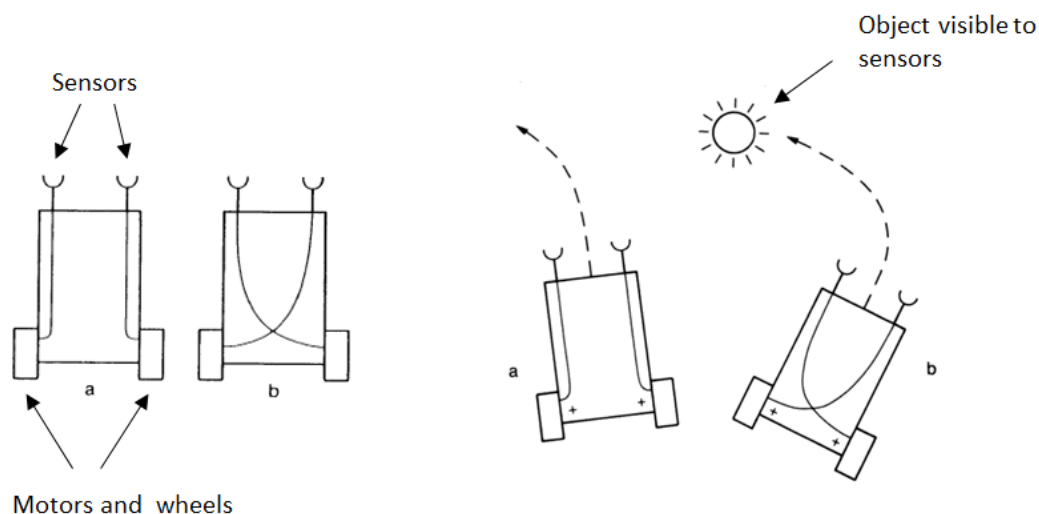
At each state an actor essentially faces the choice between exploring the environment or exploiting existing knowledge. Agents explore the environment to assess the reward structure. After some time the agent might have enough information to evaluate it and choose the most rewarding actions. The question is at what state should the agent exploit existing knowledge and when should it explore new states. If the agent does only little exploring, it might get stuck in a local minima, if it explores too much it has bad converging properties.

## 2.6 Spiking controllers for mobile robots

Traditionally artificial neural networks have been used in mobile robots to cope with unknown or ever changing environments. SNNs on the other hand have been proved useful in neuroscience and it is relatively easy to model its dynamics. Applying a spiking neural network to an embodied context reveals crucial insights on the path of understanding biological intelligence and the first step is to control the sensor-motor loop. This is traditional tested with obstacle avoidance and object attraction for a terrestrial robot with wheels or legs.

### 2.6.1 Braitenberg vehicles

In his famous book “Vehicles: Experiments in Synthetic Psychology” [41] Valentino Braitenberg describes a series of thought experiments in which vehicles that have a basic internal structure behave in unexpectedly complex ways. Braitenberg vehicles are two wheeled mobile robots with a differential drive configuration. They operate without internal memory, map of the environment and without inference. The controller or brain in these vehicles is a simple neural network that acts like a clocked digital circuit driving the motors based on sensor input.



**Figure 2-21 Baitenberg vehicles.** Adapted from [41].

Among other behaviours Braitenberg describes how obstacle avoidance and attractor behaviour can be implemented by just wiring sensors to motors differently. In the configuration a) shown in the figure above, the robot steers away from obstacles. Once it spotted the obstacle with its left sensor its left wheel begins to turn faster turning the agent away from the object. In the second configuration b) the connections are crossed from left sensor to right motor and vice versa. A sensor reading on the right sensor stimulates the left motors making the vehicle turning towards the object. The left motors will only turn faster as long the object is closer to the left sensor. When the object lays straight ahead both sensors will stimulate the motors with equal strength, making the robot driving straight ahead

This basic setup can be extended by the introduction of additional sensors that react to different stimulus in the environment enabling richer behaviours and dynamics. For example an agent might be attracted by light. Another might avoid high temperatures. A third agent might have combinations of all those behaviours. Instead of having a monotonic relationship between sensor stimulus and motor output we can alternatively introduce threshold gated functions. A common application for threshold functions is to require a minimum of sensor stimulus to overcome the initial friction of a robot, as well as a maximum possible level for the motor speed. One could think of other non-linear relationships between stimulus and motor output: For example a bell-shaped curve in which a certain level of stimulus intensity produces the maximum velocity of the wheels. Intensity above that level would result in declining velocity.

In this study we build on Braitenberg's ideas to build a spiking neural network to control a robot. In the controller we do not hardwire rules but model learning using STDP. The goal is to be able to learn to avoid obstacles and approach food without imposing unrealistic neuroanatomical or environmental constraints. The focus is on biological approximate modelling of neuronal plasticity processes but limited to control a single aspect in the form of a sensor-motor controller inspired by the Braitenberg vehicle.

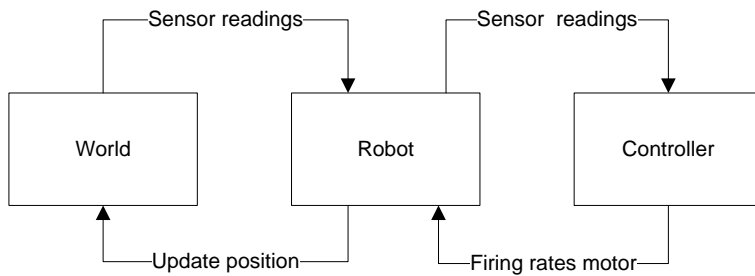
### 2.6.2 Existing spiking controllers for mobile agents

There currently exist few implementations of mobile agent controllers based on spiking neural networks. In order to deal with unforeseen problems and environments mostly artificial neural controllers have been employed. Our motivation is different as we try to gain a better understanding how biological processes work, as opposed to solve a concrete AI problem.

Our aim is to design a plastic neural controller that features biological inspired learning and behaviour in the context of a mobile wheeled robot. There are currently no studies the author is aware of that have demonstrated such a model. Existing work on spiking neural controllers in a embodied context often place neuroanatomical constraints on the network [5]. Other studies artificially require the neural encoding to satisfy certain inequalities [42], are applied to a different context [8], or learn the network topology using an evolutionary approach, i.e. a genetic algorithm [6], [7], [43].

### 3 Modelling

Our model consists of a robot, the world, and a neural controller. The robot is a simple differential drive mobile robot with no physics model. The robot moves within a world that is a continuous torus and might contain obstacles and food sources. The figure below illustrates the basic setup. In the following sections we will explain in detail the different components involved especially focussing on the learning that happens in the neural controller.



**Figure 3-1 Robot model**

#### 3.1 Robot

In this chapter we describe the Braitenberg inspired sensor-motor controller that is the base for our learning experiments. We decided to use a simple simulated robot for our task. Using a real robot would introduce additional uncertainty and variables to control. Instead we want to limit the complexity of the robot in order to be able to concentrate on the neurodynamics and plasticity features of the controller.

Our differential drive wheeled robot has the following features:

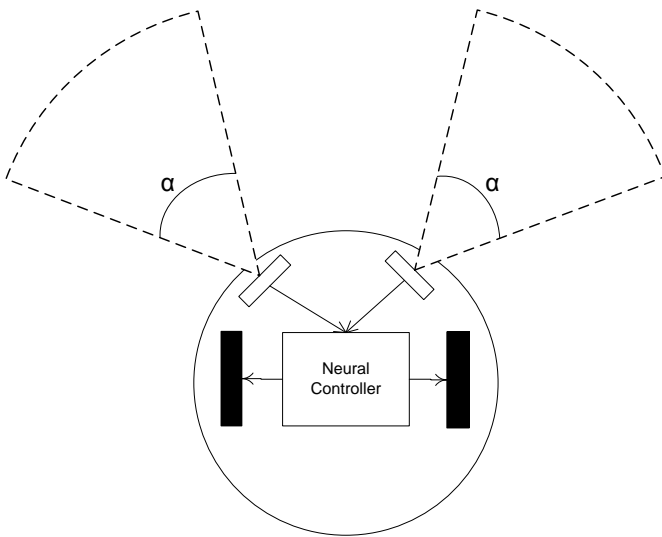
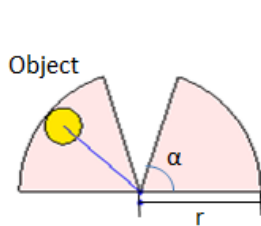


Figure 3-2 The Robot

- Two forward pointing sonar sensors. They can detect the exact distance from an object to centre of the robot.
- The range of the sensors is constrained and subject to the experimental setup.
- The angle  $\alpha$  where sensor readings are available is constrained and again subject to experimental setup. Usually the robot cannot see objects directly in front of him and the angle does not extend  $\alpha=90^\circ$ .
- Objects behind, on and generally out of the range of sonar sensors, are not visible for the robot
- The wheel velocity is constrained to be within  $U_{\min}$  and  $U_{\max}$ .

### 3.1.1 Sonar configuration



If we abstract from the robots diameter and mass we can represent the sensors with the following diagram. We use this abstraction in most of our experiments if not mentioned otherwise, so that the robots body only ever occupies a single coordinate. Both sensors touch each other at this centre point. Left and right sensors are oriented symmetrical and are defined by the angle  $\alpha$  and range  $r$ .

Figure 3-3 Sonar configuration

#### 3.1.1.1 Range

The sonar range determines how far away the robot can recognise objects. The larger the range, the more objects come into the view. In our world we have objects that are typically within 50 discrete robot steps. We therefore decide to make the sonar range rather small (within 20 steps) to limit the number of signals the robot has to process. Many signals make it more difficult for the robot to focus and lead to oscillations between left and right turns. This in turn leads to more complex learning.

#### 3.1.1.2 Angle

The angle  $\alpha$  defines from where objects can be seen. In experiments we found that it's best to have 30-40 degrees in front of the robot that is out of reach for the left and the right sensor. This is to have some flexibility on the angle when approaching an object. A bigger angle makes the robot overshoot its ideal orientation towards the object which he then has to correct.

### 3.1.2 Robot Environment

Our world is a torus without borders. On the torus we place a certain amount of objects, which can be food sources or obstacles depending on the experiment.

The robot is able to move freely around the environment. By making contact with an object it either collects a resource or bumps off the obstacle. Once a food source is collected or obstacle is hit it is removed and another resource is immediately created in some random location within the environment.

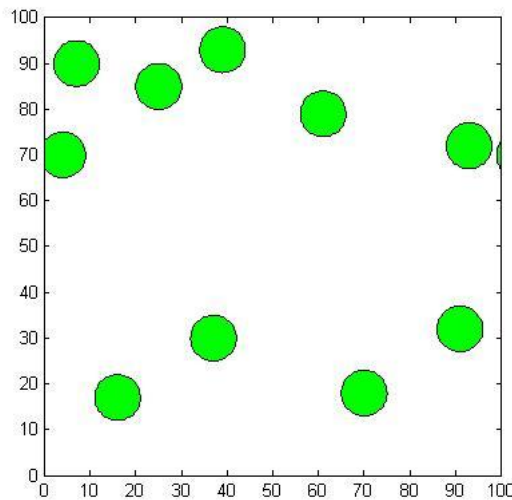


Figure 3-4 Robot World with Food Sources

### 3.2 Neural Controller

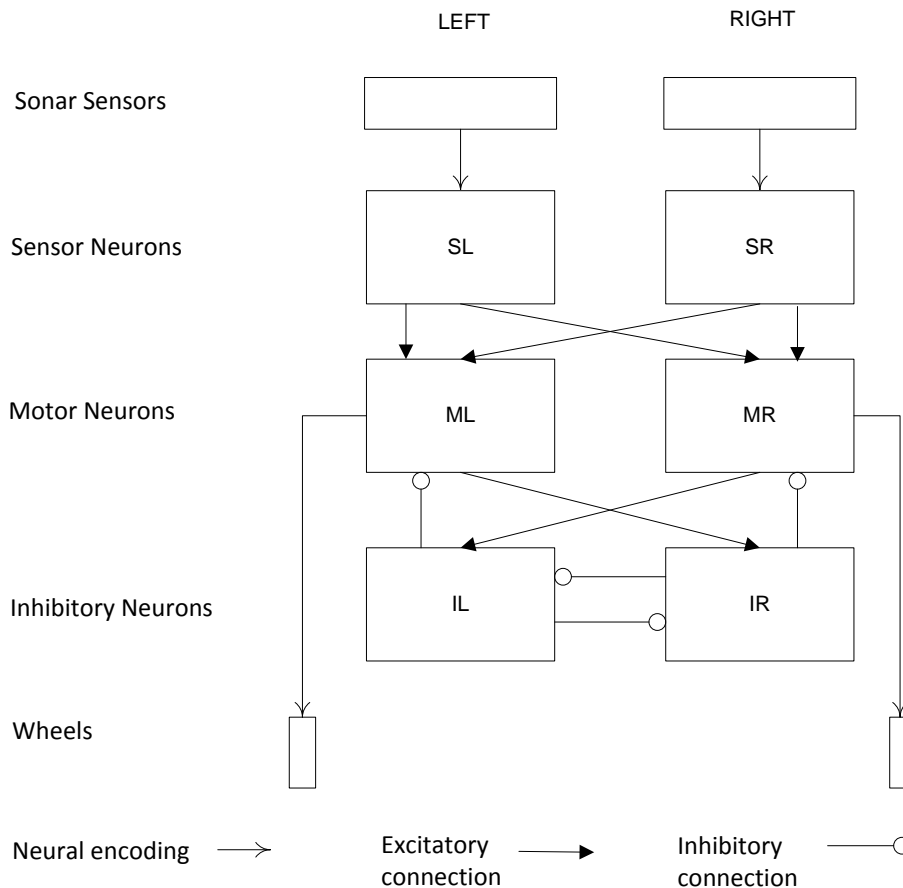
Our neural controller consists of a spiking neural network with neuronal layers for sensors and motors. Figure 3-5 shows the morphology. Sonar sensor readings are encoded into spike inputs for the sensor neurons left (SL) and right (SR). The sensor neurons excite motor neurons left (ML) and motor neurons right (MR). Connections from either sensor neuron population are going to both motor neuron layers ML and MR. Instead of hardwiring attractor or avoidance rules we let the learning mechanism evolve the correct connectivity patterns.

For attractor behaviour, the connections SL to MR and SR to ML must dominate SL to ML and SR to MR. For obstacle avoidance behaviour the connections SL to ML and SR to MR must dominate SL to MR and SR to ML.

For the neuron model we use the Izhikevich model as described in 2.2.3. Excitatory neurons are modelled as regular spiking neurons with the following parameters:  $a = 0.02$ ,  $b = 0.2$ ,  $c = -65mV$  and  $d = 8$ . In order to introduced some heterogeneity we randomize the  $c$  and  $b$  parameters as suggested in [26]. Inhibitory neurons are modelled as fast spiking neurons The parameters for these neurons are  $a = 0.1$ ,  $b = 0.25$ ,  $c = -65mV$ , and  $d = 2$ .

Basic Braitenberg vehicles do not know whether to turn left or right when there are sensor readings of the same strength from both sensors. Instead such a vehicle would just drive straight with higher speed. If we are learning attractor behaviour this is an issue. The neural controller must be able to decide which side to turn. This can be solved with a winner takes all (WTA) architecture. WTA is realised with mutual inhibition of the motor neurons. The left (right) motor layer has excitatory connections to the right (left)

inhibitory population. In the presence of two competing sensor inputs this configuration has a stable state when either the left or right motor layer population is active. The state in which both neuron layers are active is not stable. For mutual inhibition we need a certain amount of neurons in each inhibitory layer. With too little neurons the variance of the mean firing rate increases and makes it more likely that the rival population recovers too quickly to have any clear winner during a certain amount of time.

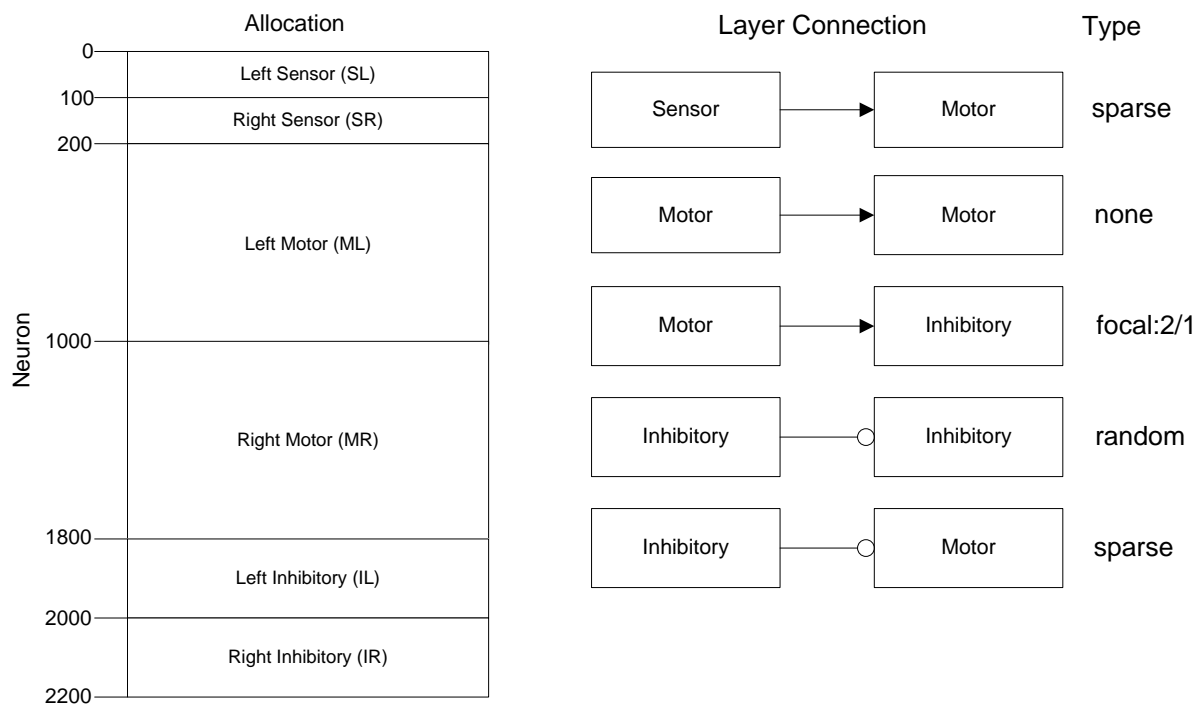


**Figure 3-5 Neural Controller**

When there is one resource type to recognize we use initially 2200 neurons in total: 800 neurons for each motor layer, 200 for each inhibitory layer and 100 for each sensor layer. Recurrent connections within the motor layer (e.g. to model drift) did not improve results in the training and we therefore have no recurrent connections within the motor layers. The number of synapses per sensor neuron is 400, for the motor layer 100 and for the inhibitory layer again 400. This setup implies that a sensor neuron connects on average to 200 right and left motor neurons. Every two motor neurons connect to an inhibitory neuron and between inhibitory populations there is a random all-to-all connectivity. Several considerations were made when choosing this topology:

- The number of neurons has to be significantly large to be able to represent diverse behaviours. This is because we need statistical properties of the model to drive the behaviour of the random components within the model.
- The relation between sensor and motor neurons and its projections has to be within a certain range. Sensor neurons must be able to influence the mean firing rate of the motor neurons and the motor neurons should be resilient to sensor noise.
- The number of motor neurons must be able to sustain a certain amount of firing to control the robot wheel speeds.

- The Inhibitory neurons must be able to temporally prevent a majority of the motor neurons from firing. Only if this is the case a WTA mechanism can take control.



**Figure 3-6 Benchmark topology configuration**

Figure 3-6 shows our benchmark configuration for one resource type. In the case where there are two resource types, obstacles and food, we have two sensor layers more per side and the number of sensor neurons increases to 400 neurons (200 neurons per resource).

Synaptic plasticity is constrained to excitatory synapses. In general the mechanisms behind the plasticity of inhibitory synapses are much more diverse and less understood than that of excitatory synapses.

### 3.3 Sensor and motor encoding

In order to process an emulated analog sensor signal we have to encode it into a neuron spike pattern. On the other end a neuron spike pattern has to be decoded into an executable motor command. Here we provide a high level overview of this conversion process.

#### 3.3.1 Sensor readings to spike patterns

The sensor reading  $S$  is a value between 0 and 1 and is defined by (3-3). Sensor reading is converted to a mean firing rate fed as Poisson input stream to the sensor neurons. Poisson processes are often used to simulate composite activity of neurons. This is however only a rough approximation. At the core the spike of a neuron depends substantially on its recent firing history. Firing statistics of individual neurons can greatly affect the behaviour of a network when temporal coding models such as STDP are used. Furthermore synchronous responses are likely different from asynchronous responses of a neuron as has been shown in the literature [21]. Since in our case STDP and synchrony at the sensor encoding level is irrelevant we can ignore this retention.

We defined the spike input pattern for the sensor neurons  $P$  as

$$P = pois(\lambda) \quad (3-1)$$

where  $P$  is the spike input pattern and  $\lambda$  is the mean firing rate calculated by

$$\lambda = Sw \quad (3-2)$$

where

$$S = \frac{sonar\ range - dist}{sonar\ range} \quad (3-3)$$

and  $w$  is a scaling weight. When tuning this encoding its helpful to consider different forager trajectories. “A” in Figure 3-7 shows an ideal path in terms of learning: A left sensor reading is encoded into spike sources for the neural controller. The stimulus causes the right motor neuron population to fire. The motor firings are translated to motor commands which result into a sharp turn. After that there are no more sensor readings. The robot can therefore drive straight towards the object. Note that the sensor angle of  $72^\circ$  (from  $90^\circ$  to  $18^\circ$ ) gives the robot some flexibility in terms of the approaching the object. In “B” and “C” the robot turns too much at the first sensor reading and must correct its orientation later. This makes it more difficult for the neural controller to learn a correct correlation between sensor readings and motor activity.

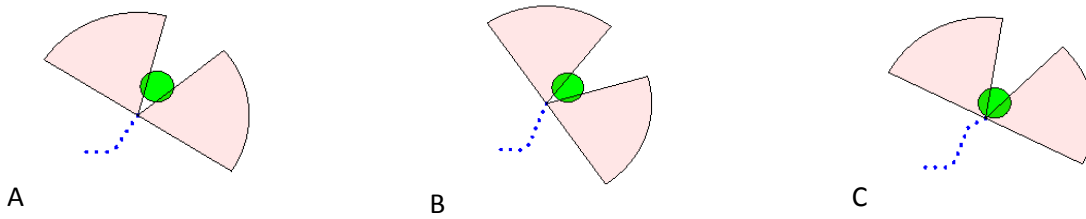


Figure 3-7 Different forager trajectories

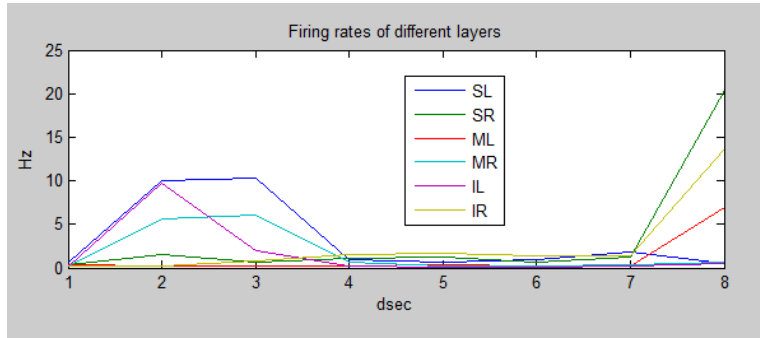


Figure 3-8 Firing rates for trajectory B

The closer we come to the object, the stronger becomes the sensor signal and therefore the greater the probability of overshooting a directional change because of sharp turns. To mitigate this we adapt equation (3-3) by thresholding it to a maximum value. This was found to smooth sensor reactions when already very close the target object.

$$S = \max(0.8, \left( \frac{sonar\ range - dist}{sonar\ range} \right)) \quad (3-4)$$



### 3.3.2 Spike patterns to motor commands

Left or right motor velocity are encoded by the mean firing rate ( $MFR$ ) of the associated motor layer. The formula for the velocity  $v$  for a wheel can be written as

$$v = U_{min} + r(U_{min} - U_{max}) \quad (3-5)$$

where  $r$  is given by

$$r = MFR / MFR_{max} \quad (3-6)$$

and  $MFR_{max}$  is the estimated maximum mean firing rate. The estimation of this maximal firing rate is somewhat critical. If we set it too low, the robot will turn very quickly and generally be very agile. On the other hand a too high value makes the robot sluggish. Furthermore during learning the firing rates typically changes. To accommodate this change we update  $MFR_{max}$  at ever timestep as follows

$$MFR_{t,max} = \max(MFR_{t-1}, MFR_{t-2}, \dots, MFR_{t-50}) \quad (3-7)$$

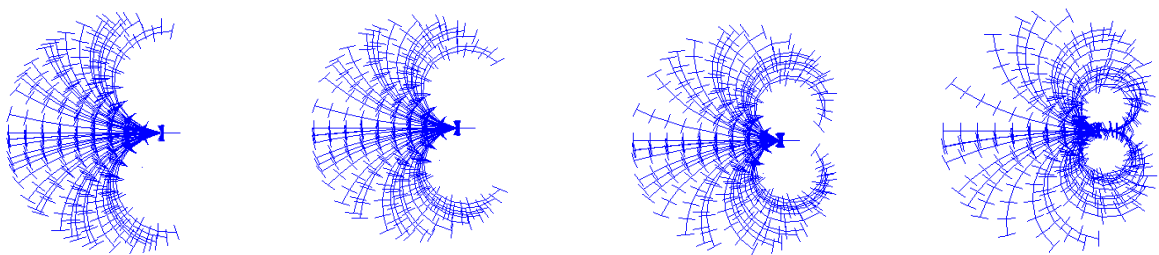
where  $t$  is a robot iteration time step (100ms). During the first 5s of a trial the MFR is kept at a constant value.

Another constraint we had to impose to avoid unnatural steering angles is to cap the velocity to be between  $U_{min}$  and  $U_{max}$ . This is because the firing rate is a stochastic variable and there is always the potential of outliers, that don't make sense to include.

#### 3.3.2.1 Defining a steering angle

Motor commands are extracted in a fixed interval of 100ms from the neural controller. Velocities coming from (3-5) are therefore average velocities. The distance and orientation are calculated in 100ms intervals as are the motor firing rates.

The maximal steering angle depends on the relation  $U_{min}/U_{max}$ , where the estimated maximal firing rate  $MFR_{max}$  defines the elasticity between the MFR and the velocity. Again we need a good compromise between agility and damping effects of outliers. In our configurations we set  $U_{max}$  within 100-200% of  $U_{min}$ .



**Figure 3-9** Different  $U_{min}/U_{max}$  relations: from left to right: 5/6, 4/5, 3/4, 2/3.

The ideal behaviour for learning and exploring is a balance between agility and constant movement. The robot should make some turns in order to learn the association between sensor readings and motor firings but not too many so that left and right sensors fire sequentially.

## 3.4 Learning model

One recent computational model [14] demonstrates how DA-STDP can be used to explain the credit assignment problem leveraging precise spike timings. In this model correlated pre-post spike pairs form synaptic eligibility traces. These processes that decay exponentially form a window of opportunity for dopamine to enhance LTP, or LTD for that matter.

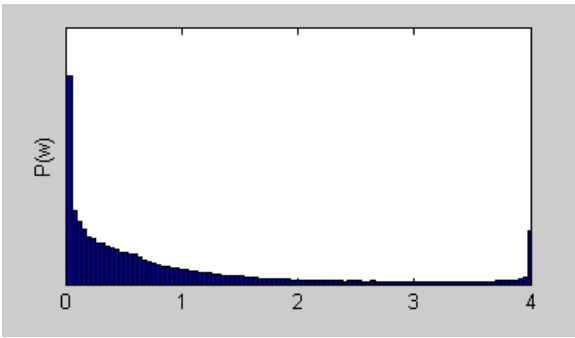
Our learning model builds on this model and includes elements from Song et al (2000) [44], Song and Abbott (2001) [15], and Izhikevich (2007) [14]. It takes into account multiple spike interactions between a pre and postsynaptic pair of neurons.

In the asymmetric version the STDP function is given by the following function:

$$\Delta w = \begin{cases} A_+ \exp(-\Delta t/\tau^+) & \text{if } \Delta t > 0 \\ A_- \exp(\Delta t/\tau^-) & \text{if } \Delta t < 0 \end{cases} \quad (3-8)$$

where  $\Delta t = t_{post} - t_{pre}$ , the post and pre spike timings. If  $\Delta t > 0$  the weight change is positive, the synapse is subject to LTP. On the other hand if  $\Delta t < 0$  LTD is in effect and the synaptic weight decreases.  $A_+$  and  $A_-$  respectively are scaling constants defining the size of the LTP and LTD windows,  $\tau^+$  and  $\tau^-$  are the defined decay rates of the two windows. Initially we work with the following values for the STDP function:  $A_+ = 0.1, A_- = -0.15, \tau^+ = \tau^- = 20ms$ .

An implication of the STDP rule is that there is a linear relationship between the strength of a synapse and the probability of a postsynaptic spike: The stronger the weight the more likely it is that the next neuron will spike. Thus, once a synapse is potentiated its chances for subsequent potentiation increase. Using equation (3-8) there is no steady state for the weight distribution in a neuron. In biology however, weights cannot grow arbitrary large. We therefore enforce this by putting a limit of size (see [15]) on excitatory synapse between 0 and 4mV and 0 and -1 (see [14]) on inhibitory synapses. Enforcing such hard bounds on synapses introduces competition between weights and the resulting weight distribution is bimodal. The figure below shows a sample equilibrium distribution, where  $P(w)$  denotes the probability of a weight  $w$ .



**Figure 3-10 Equilibrium distribution of synaptic weights**

Alternatively to hard bounds soft bounds have been suggested (see [18]). With soft bounds a different STDP rule is applied where potentiation slows down with increasing weight. Depression does not slow down, so that the weaker a synapse the stronger the fluctuations. The equilibrium of the soft bounded STDP rule in [18] is unimodal, with a to the left side skewed bell-shaped distribution. In our neural controller competition between synaptic weights is important. We start with a uniformly connected sensor to motor layer and aim to learn the connections where a maximal amount of rewards is collected. For this reason we use a model with hard bounds and reject a STDP rule with soft bounds.

Synapses also can never switch signs and become inhibitory if excitatory or vice versa. Synapses can thus only be deactivated or reach a maximum. The balance of potentiation and depression is controlled by the ratio  $A_- \tau^- / A_+ \tau^+ > 1$ . This temporal asymmetry prevents uncontrolled growth and leads to stable

dynamics. To determine the spike pair at any given point of time  $t$  for a synapse for STDP we choose the nearest post-pre and pre-post pair.

For reward modulated STDP the weight change  $w_{ji}$  at synapse  $ji$  from neuron  $i$  to neuron  $j$  can be written as:

$$\Delta w_{ji}(t) = c_{ji}(t)d(t) \quad (3-9)$$

with  $c_{ji}$  denoting the eligibility trace from neuron  $i$  to neuron  $j$  at time  $t$ , and  $d(t)$  the reward function. The eligibility trace function  $c_{ji}$  is given by:

$$\dot{c}_{ji} = -c_{ji}/\tau_c + STDP(t_{post} - t_{pre}) \quad (3-10)$$

where  $STDP(t_{post} - t_{pre})$  is the synaptic weight change according to the STDP learning rule (3-8) and  $\tau_c$  is the time constant indicating the exponential decay rate of the eligibility trace function. Following [14] we set the decay at 1% per millisecond, so that 5 seconds after the STDP triggering event the effect of DA can be ignored. Alternatively - to model insensitivity to rewards that come too early - one could model the eligibility function as a gradual increase instead of a step increase, see [20]. For simplicity we do not consider this case further.

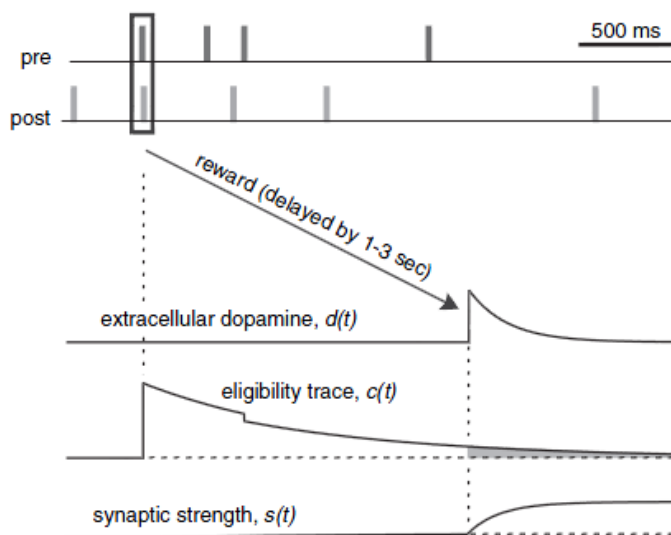
DA is modelled by a baseline and a rate of change. After a reward DA step increases and then decays exponentially over time to the base level. We can write

$$\dot{d} = -d/\tau_d + DA(t) \quad (3-11)$$

$$DA(t) = \begin{cases} 0.01 \mu M/s + 0.5 \mu M & \text{if reward issued} \\ 0.01 \mu M/s & \text{else} \end{cases} \quad (3-12)$$

where  $d$  is the concentration of extracellular DA (in  $\mu M$ ),  $\tau_d$  the time constant of DA uptake and  $DA(t)$  the tonic source of DA, giving a baseline of 2nM, following [14]. This baseline ensures that the synaptic weight does not undergo significant modification unless a reward is issued. Reward is modelled by step increasing the concentration of DA by  $0.5 \mu M$ . If the reward is negative we reduce DA by  $0.2 \mu M$ . This value ensures that the DA concentration recovers to its baseline after 1s which is important to guarantee stable learning. If the DA taken off the baseline is too large the weights would rapidly converge to zero. To model a 5%/ms decay in DA concentration we set  $\tau_d = 0.2s$ . With this setting, significant extracellular dopamine remains about 1s in the system. This limits the probability of rewarding uncorrelated spike pairs that occur *after* the reward is issued.

Figure 3-11 illustrates the dynamics of the model. LTP and LTD can be enhanced by a reward that is issued within the time of an active eligibility trail.



**Figure 3-11 STDP and eligibility traces**

A pre-before post spike pair generates a positive eligibility trace during which the synapse is eligible for a weight increase induced by a reward signal. Taken from [14].

A nearly coincident firing of a pre and postsynaptic neuron makes the synapse eligible for a reward. If the reward is issued after some delay this might strengthen other not reward-triggering connections. This problem is overcome by low firing frequency of neurons and the induction of noise. The low firing frequency of around 1Hz per neuron ensures that the probability that an uncorrelated spike falls into the STDP time window by chance is relatively low. Through random firings such coincidental connections cancel each other out over time and moreover through the asymmetry of LTP vs. LTD result in a net decrease of such synapses.

As a result over many trials, rewards will potentiate only the reward triggering connections.

### 3.5 Training and Testing

If we have uniform distributed connectivity between each sides of sensor and motor neurons with no external input current into the neural controller the robot will drive just straight through the world. If an object appears say on the left, the left sensor neurons will excite equally left and right motor neurons. Since all connections have the same weight this will simply result in an increased velocity of both wheels. Turns however will rarely happen.

In order to induce learning we need an external teacher to direct the robot at the beginning. Looking at how humans learn, a training phase with external intervention from a teacher is a quite realistic approach. For example parents guide their baby to learn to walk or show them how to eat food. On the other hand there are equally many situations where humans need to learn themselves. For example running into a wall hurts and triggers a negative reward in terms of pain. We consequently learn to control our motor movements and stop early when approaching an obstacle.

In our experiments the training can either be supervised, where we inject external spike streams, or unsupervised with the robot performing a random walk coincidentally bumping into objects. The random walk is realised with an external spike stream into the motor neurons.

The training phase stops when either a time limit or a learning goal is reached. We define the learning goal by the synaptic weight discrimination between connections from sensor to motor layers. After the training the robot is tested in a world with several objects. Learning continues during this phase. If the

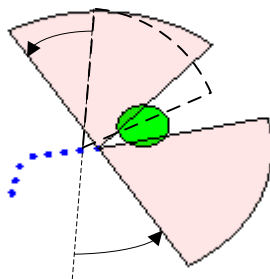
configuration is stable, weights between layers in the neural controller will stay constant or continue to diverge as during the training. Performance is measured by comparing the number of rewards the robot is collecting with the number of rewards collected by a robot with a hardwired connectivity and a robot that just moves around the environment in a random walk without any STDP and uniformly connected neurons.

### 3.5.1 Single object training

In this setup the robot is presented with one object per trial. The goal of the training is that the robot is able to turn into the correct directions to then later be able to further improve its agility autonomously. Via external current into the motor neurons the robot is directed to drive towards the object and touch it. The duration of each trial is limited to 1-1.5s and trials are separated by 1-1.5s. The object can be to the left or right of the robot and must be within the reach of one sensor. During a sonar reading, independent of the magnitude, the left or right motor neurons are stimulated with a Poisson distributed input current. The stimulation is as strong to manifest in a significant mean firing rate change which eventually will orient the robot perpendicular to the object.

#### 3.5.1.1 Dynamic training

In our robot experiments we basically reward a turn in the right direction. If the robot completes the turn without training current, additional rewards cannot increase the connection strength difference in the long run. If they do in the short run, the robot will turn too much and cause additionally the “wrong” layer correlations being rewarded.



**Figure 3-12 Overshooting a turn**

In order to further increase the connection strength difference we can gradually reduce the training current. The process that is changing the weights is stochastic however. Instead of trying to find a function that fits the weight changes related to the external current, we implement a feedback loop. The training current is either decreased when the robot is overshooting a turn, or increased if the robot is turning too little and is missing the target. The magnitude is dependent on the history of the left and right turn behaviour and exponentially increases or decays if previous modification were done in the same direction.

### 3.5.2 Random walk training

We define the following random walk algorithm that generates input currents for the left and / or right motor neurons.

#### MATLAB PSEUDOCODE: RANDOM WALK GENERATOR

```
function IExt=randomWalkMotorCurrent(IExt,ML,MR)
% input:  IExt Vector of input current per neuron
%         ML  Indices of left motor neurons
%         MR  Indices of right motor neurons
% output: IExt Updated vector of input currents

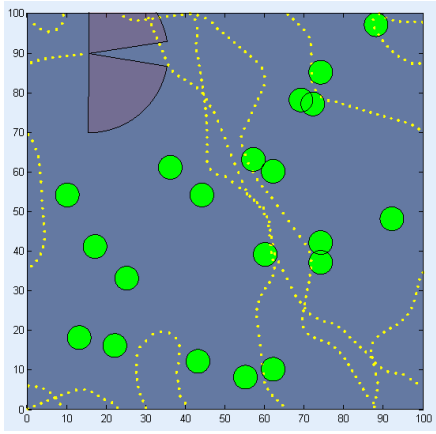
persistent countr vL vR waitingPeriod;
baseCurrent=0.3;
if isempty(countr)
    countr=0;
    vL=baseCurrent;
    vR=baseCurrent;
    waitingPeriod=10;
end
countr=countr+1;

if mod(countr,waitingPeriod) == 0
    waitingPeriod=min(lognormal random number,20);
    if rand > 0.4
        if rand > 0.5
            vL=baseCurrent +rand;
            vR=baseCurrent;
        else
            vL=baseCurrent;
            vR=baseCurrent +rand;
        end
    else
        vL=baseCurrent;
        vR=baseCurrent;
    end
end
% generate a poisson random stream with  $\lambda=vL$  and length ML
IExt(ML)=poisson random number list(vL,length(ML));
IExt(MR)=poisson random number list(vR,length(MR));

end
```

Figure 3-13 shows a random walk trajectory on the torous, when we did not remove any touched objects (normally we would immediately replace it by another object at some random location, but this might paint over the trajectory). The random walk needs to be tuned to have a good variation between turns and straight movements, so that the robot gets a chance for rewarding turns.

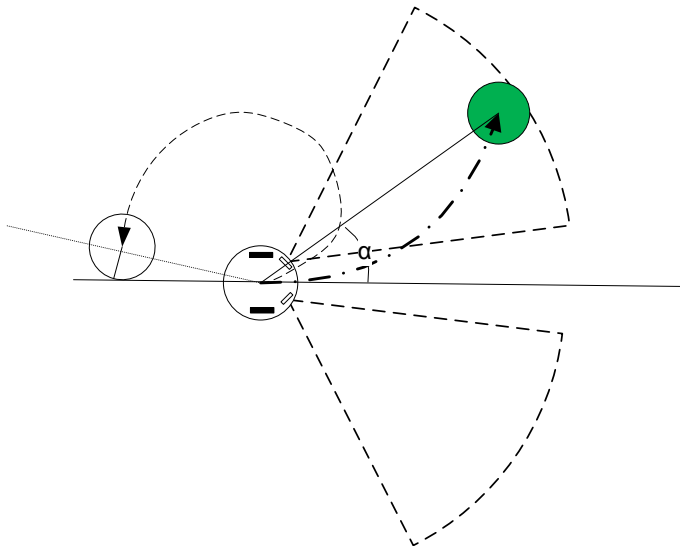
When using a random walk for training we need also a balance on the number of objects present in the environment. Too many objects lead to coincidental collisions with few chances to reward explicit turning beforehand.



**Figure 3-13 Random walk trajectory**

### 3.5.3 Defining a stop training condition for synaptic weights

In 0 we defined a linear relation between the difference in mean firing rates and the change in orientation of the robot. In order to be able to at least make soft turns in response to recognised objects, we set required minimum steering angles for different stimulus strengths so that the robot is able make contact with an object  $\alpha=45$  degrees away close to its sonar range border as depicted below:



**Figure 3-14 Goal and maximum trajectory**

At every step the robot moves forward without changing its orientation,  $\alpha$  increases. The number of steps required to gap the distance to the object depends on the velocity and trajectory. In our experiments it's typically between 5-9 iterations. When the velocity difference between left and right wheel is at its maximum, the robot is able to change its orientation by about 150 degrees within 10 iterations. This is when the maximal velocity is set to  $U_{\max}=U_{\min}+U_{\min}/2$ . Since the stimulus strength is linearly increasing when approaching, the steering angle typically increases during the trajectory. Of course if the first stimulus is so strong that the robot turns perpendicular to the object there are no more orientation changes needed. Otherwise the angle between the robot and the object changes at every iteration. Thus, the required accumulated change in orientation to make for the robot from the original position until contact is made with the object is significantly larger than  $45^\circ$ .

Increasing the complexity, the mean firing rates depend on stochastic variables of the neuron model, the selected topology, the inhibitory neurons, background firings and the calculated maximum mean firing rate. This leads to the fact that we would have to tune the stop condition per experiment.

External background firing is set via a Poisson distributed random input stream so that sensor neurons spike around 2Hz, motor neurons 5Hz and inhibitory neurons 10Hz. Figure 3-16 shows the linear relationship between the orientation change per mean firing rate difference for the setting  $U_{\max}=U_{\min}+U_{\min}/2$ . For an aggregated 90° orientation change in 8 iterations the average mean firing rate differences needs to be about 3, given both rates are below  $MFR_{\max}$ . We need to consider however only the 2-3 iterations, since turning does not happen continuously. The increasing stimulus strength causes a rapid orientation changes once the object becomes close. During each of these 2-3 steps the steering angle of the robot needs to be at least 15 degrees. The aggregated orientation change is then around 90 degrees that will be completed within 8 iterations.

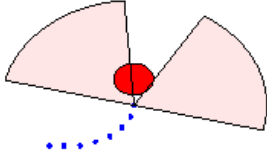


Figure 3-15 Minimum steering angle

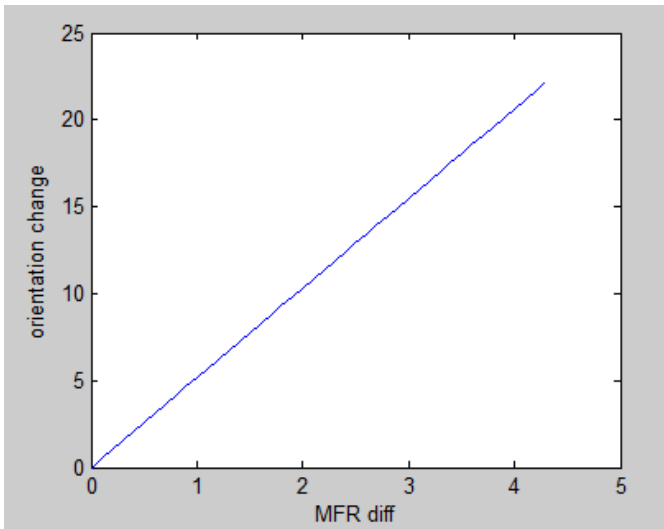


Figure 3-16 Orientation change to MFR difference per robot iteration

The next step is to relate the difference in mean firing rates to the difference in synaptic weights between the two competing connections. This is the following two rules must hold for the training to terminate

$$\begin{aligned} \text{mean}(s(SLMR)) &> \lambda * \text{mean}(s(SLML)) \\ \text{mean}(s(SRML)) &> \lambda * \text{mean}(s(SRMR)) \end{aligned} \tag{3-13}$$

where  $s$  denotes the synaptic weights, SL the sensor neuron left, SR the sensor neuron right, ML the motor neuron left and MR the motor neuron right.  $\lambda$  is the connection strength factor. We will use this lambda factor as means of measuring the separability of motor responses.



## 4 Experiments

### 4.1 Enforcing a connection by rewarding relative spike counts

In this preliminary experiment we follow Izhikevich (2007) [14] and apply the instrumental conditioning experiment to our neural controller with WTA architecture. At this point there is no robot involved.

#### 4.1.1 Methods

##### Basic setup

We start with an all-to-all synaptic connectivity between the sensor and motor neurons as described in the preceding chapter. All the weights are randomized between 0 and 1. As in [14] we choose a group to represent the input stimulus S and two nonoverlapping groups of neurons A and B as motor neurons where to measure the response. For the stimulus group we choose SL and for the two motor neurons groups A and B we choose ML and MR. At the start of every trial we deliver a strong 1ms pulse of current into SL. Then during a 20ms window of time after that stimulation we measure the response of the motor neurons A and B. Clearly the number of spikes fired in A or B depends on the strength of the connection from the sensor neurons. If the response of  $|A| > |B|$  then we deliver reward in form of a delayed dopamine release. The delay is inversely proportional to the difference in the spike count between A and B and limited to 1s.

The experiment is conducted in several trials separated by 10 s. After every trial we measure the mean weights between S and A, and S and B. We try to find out if we can reinforce a particular response in our neural controller and gain insights on how to tune our STDP model.

##### Progressive learning

We have seen how temporal difference learning can be applied to instrumental conditioning. In this framework dopaminergic activity is encoding the reward prediction error. We incorporate this into our experiment by steadily increasing requirements for a reward. Instead of having a fixed rule stating  $|A| > |B|$ , we depend the requirements for a reward on past performance. Doing this we attempt to get better response discrimination. In our experiments we found that a progressive reward rule should only be applied after the weight difference reached a certain threshold. The rule we found best performing is given by:

$$\delta(r) = \begin{cases} 1 & \text{if } |A| > |B| \text{ AND } sratio_{t-1} \leq 1.15 \\ 1 & \text{if } sratio_{t-1} > 1.15 \text{ AND } |A| > (sratio_{t-1} + 0.2)|B| \\ 0 & \text{else} \end{cases} \quad (4-1)$$

where  $\delta(r)$  is the dirac delta function that is multiplied with the DA release at equation (3-12), so that it works as a case based rule, and

$$sratio_{t-1} = |A|_{t-1} / |B|_{t-1} \quad (4-2)$$

is the response ratio from time t-1.

### Activity dependent scaling (ADS)

We implement synaptic scaling as accommodating, regulating mechanism to STDP. Activity dependent scaling (ADS) modulates the firing activity to a target rate. The relative synaptic strengths within a neuron are kept constant, but scaled up or down. We follow [18] and modify excitatory synapses with

$$\frac{dw_{ij}}{dt} = \tau_{ADS} w_{ij} (z_{goal} - z_j) \quad (4-3)$$

Where  $\tau_{ADS}$  is the timescale over which ADS is adjusting. This parameter has not been measured, but is expected to be slow. We use 100s, the same value as in [18].

As variant, instead of modifying the activity per synapse, synaptic scaling can be applied to populations of neurons. This fits our experiment in that we try to extract a mechanism for a good discriminator between the responses A and B. We define a scaling factor for all the incoming currents for our motor layer (group A and B). Every 10ms we multiply each incoming synapse of group A and B by

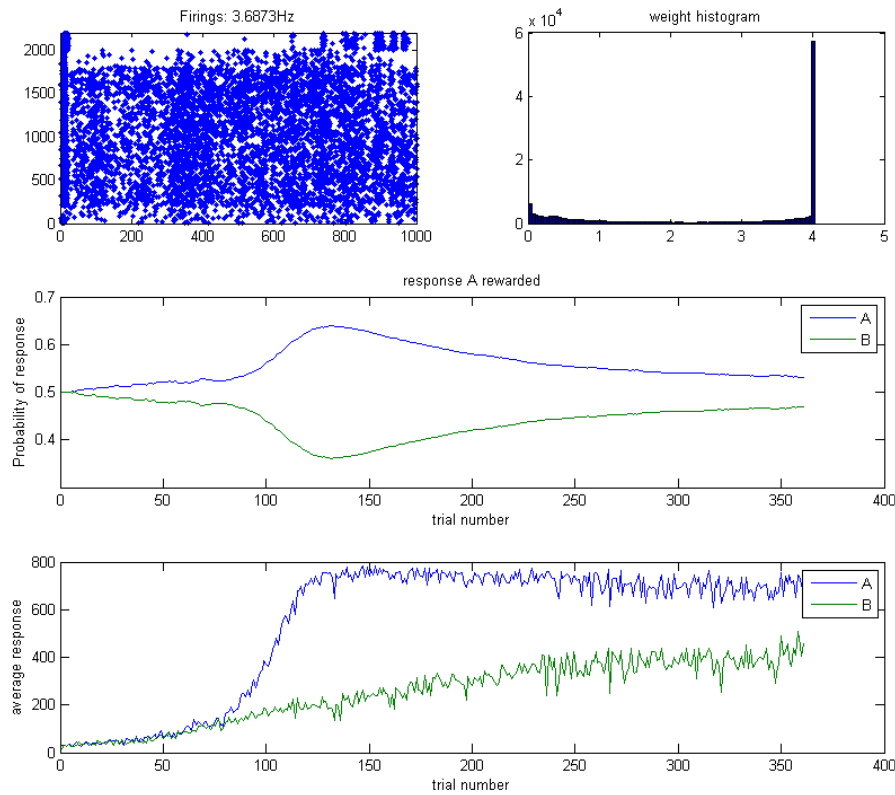
$$\frac{dw_{ij}}{dt} = w_{ij} \frac{w_{TOT}}{\sum_i w_i} \quad (4-4)$$

where  $w_{TOT}$  is the fixed weight sum for both groups. Basically this is an arbitrary value. For convenience we set this value to the total sum of all weights in group A and B.

$$w_{TOT} = \sum_i w_i \quad (4-5)$$

## 4.1.2 Results

### 4.1.2.1 Basic setup



**Figure 4-1** Rewarding relative spike counts

Figure 4-1 shows the results of the experiment after 360 trials where group A is rewarded. In the first row of the figure we show the firing patterns of the last trial and the synaptic weight distribution over all synapses. Below this we depict the probability of a response of group A or B measured by the average strength of the synaptic input from S. The bottom row shows the average number of spikes fired by A and B per trial.

Rewarding group A reinforces the connections into that group. The average response increases for both groups. This is because we usually have responses from both group A and B to a stimulus, and therefore are rewarding both connections, but reward connection A more significantly. The average response from A and B stays the same up to about trial number 70. Interestingly the mean synaptic weights at that trial number already are quite different, but this doesn't seem to have significant effects on the responses. After trial number 70 however, the average response difference between group A and B increases steeply until almost all of the neurons of group A are spiking.

It may at the first glance be surprising why the weights are converging again after trial 120. The reason for this is twofold. First, group A spikes are at maximum rate which cannot further increase. This is also validated by the following plot showing the firings of first 40 ms of the last trial in the experiment.

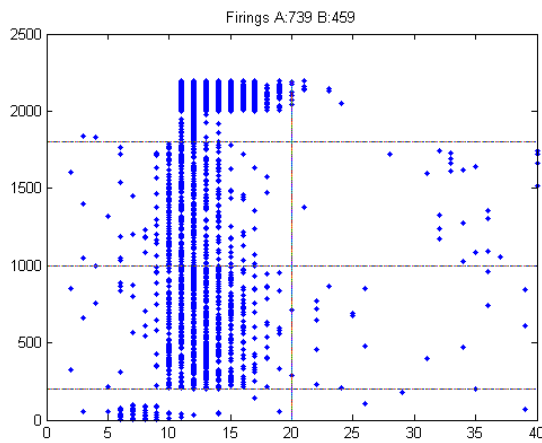


Figure 4-2 Spike pattern of first 40ms of last trial

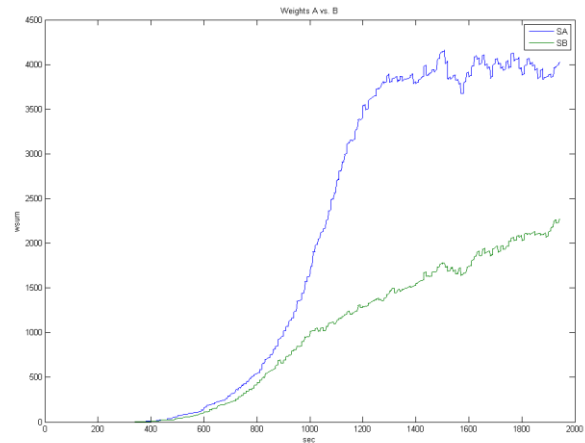
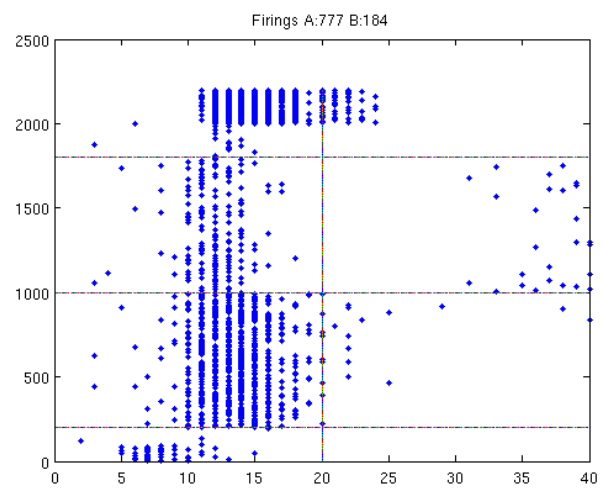
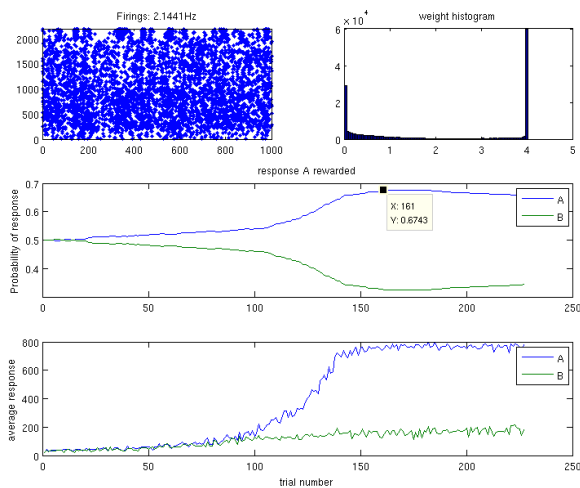


Figure 4-3 Weights > 3.8

Secondly, almost all of the synapses leading into group A are at maximum. Figure 4-3 shows that after 1200 sec (after the 120th trial) the number of weights in group A that are bigger than 3.8 float around 4000. The total number of synapses connecting from S (100 neurons) to A (800 neurons) is 5000. Given that some connections likely have been extinguished due to coincidental negative STDP correlations it is fair to assume that most connections from S to A reached their maximum strength after trial number 120. Since these weights cannot further increase rewards increase connections between S and B and therefore the converging pattern is observed.

#### 4.1.2.2 Progressive learning

Progressive learning has the effect that the network learns slower and the response performance can be increased slightly. The issue of overtraining of the response A however remains because of the reasons discussed earlier. It is clear that we would need additional rules to prevent an overtraining.



#### 4.1.2.3 ADS

We let the experiment run for simulated time of 60minutes. During this time 360 trials were executed.

Both versions of ADS show a slower learning in compare the trials without ADS. Scaling for a target firing rate based on (4-3) reaches about at trial 200 the maximum weight separation between group A and B.

As can be seen from the average response subplot after this trial basically all neurons fire in response to the stimulus. We see then as before a converging of the response probabilities, although much slower. For our second version of ADS we have a quite different picture. First the mean weights between group A and B have diverged much more, giving a response probability of group A of more than 80% compared to 72% for (4-3), 67% for progressive learning and 65% for the basic setup. Secondly the response probabilities do not converge anymore after having reached a maximum. The ADS modulation seems to work in favour of a maximal response discriminator.

Another striking difference between our ADS versions is the weight distribution. In the second version there are more synapses between 1 and 0 and less at the maximum.

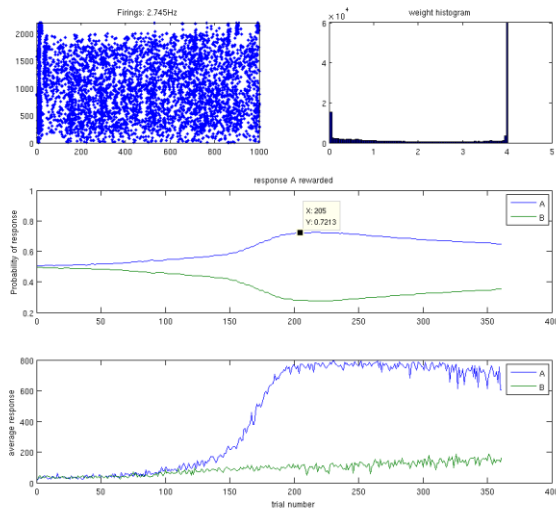


Figure 4-4 ADS with equation (4-3)

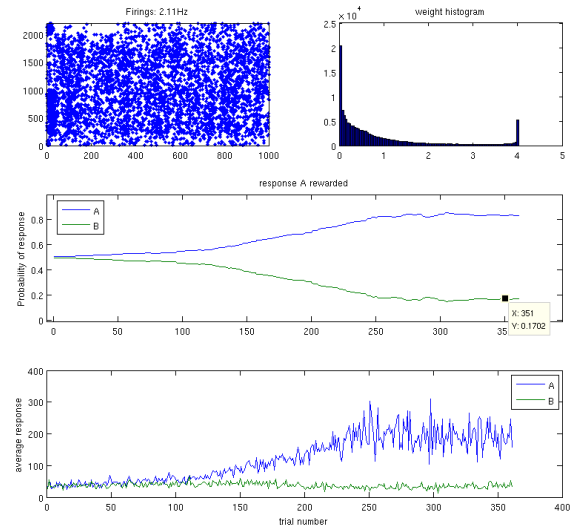


Figure 4-5 ADS with equation (4-4)

## 4.2 Attractor behaviour

For attractor behaviour we must find a stable condition where there is significant separation between the connections from left (right) sensor neurons to right (left) motor neurons and left (right) sensor neurons to left (right) motor neurons, so that the robot is able to turn autonomously later on. The tuning of the parameters of the model turns out to be quite tricky. This is because many different parameter settings such as neuron model parameters, network connectivity, neuronal encoding, training current and robot morphology all have profound sometimes dramatic effects on the learning patterns.

After the training we let the robot drive around the environment containing resources. The quality of the training is measured by the number of rewards collected relative to the rewards a random walk would collect. Learning continues during the testing but will be more random. The learnt behaviour should not deteriorate but ideally be further enforced. We start with attractor behaviour training because in compare to repulsive behaviour it is easier to tune. For training repulsive behaviour we face the additional difficulty to model negative rewards and to prevent weights to not converge to zero too quickly.

In the following sections we study the effects of different artefacts of the model as defined in the previous chapter. We apply the training procedure as described in the section 3.5. Neural noise is introduced into the network by perturbing all layers with a random thalamic input in the range  $[-6.5, 6.5]$ . Additionally we induce noise into the motor neurons in order to have a constant motor firing defining the speed of the robot. This noise causes the sensor layer to spike at a frequency of 0.5Hz, the

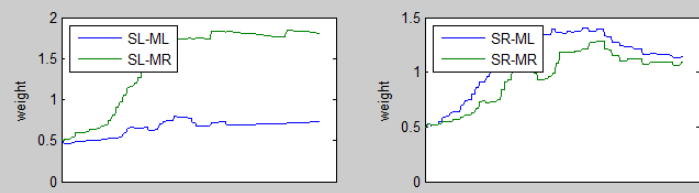
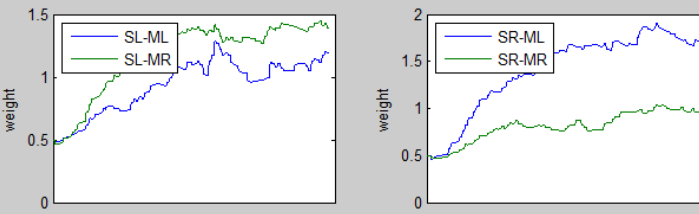
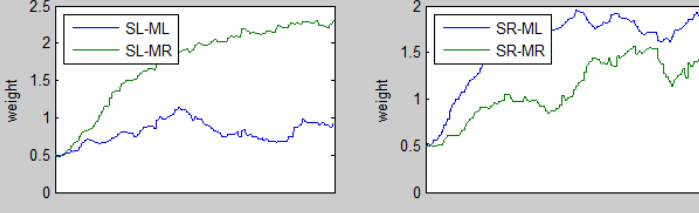
motor layer at 1Hz, and the inhibitory layer at 0.5Hz when there is no other stimulus. The training current is delivered into a motor layer when there is a corresponding sensor input current. The strength of the sensor input current depends on the distance to the object and is encoded as described in chapter 3.3.1. As stop training condition we use either the lambda factor as defined in chapter 3.5.3 (3-13) as well as a maximum time duration.

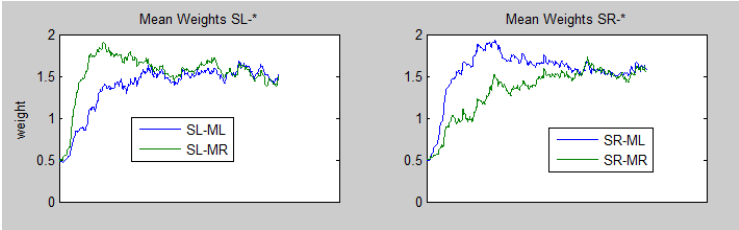
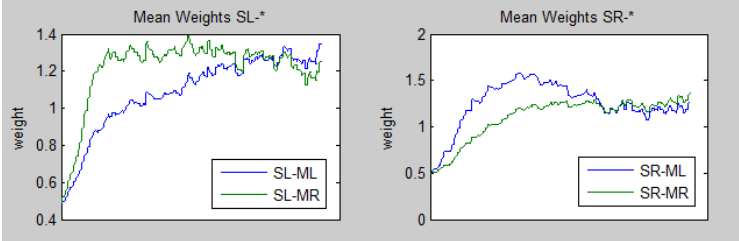
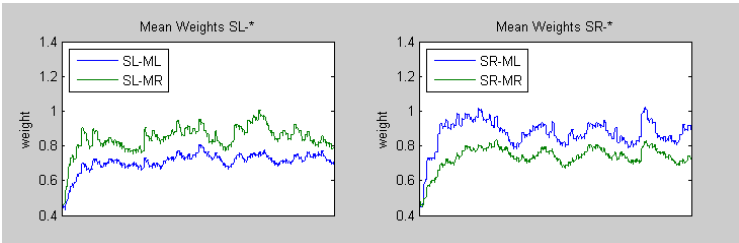
#### 4.2.1 Effects of neural noise and sonar angle

Experiments 1-4 illustrate the effect of a large sonar angle  $\alpha$  (see 3.1.1.2) of 85.5° from 90° to 4.5° during a training period of 3 minutes. During that time each side, left and right, has been trained in 100 trials. The mean of the Poisson random input current for the motor layer is set to 0.3.

In experiment 1.1 the connections between left sensor and right motor neurons are trained well, whereas the connections between right sensor neurons initially diverge and then converge after a few trials. The small sonar angle introduced oscillations in many trials. This means that the robot turned too much so that both left and right sensor neurons were activated during a trial. The learning process becomes highly random and mutual inhibition randomly enforced one connection over another. In 1.1 the WTA mechanism enforced the selection of the left side and made the weights between right motor and left inhibitory neurons stronger, in 1.2 it's just the opposite. Depending on how much one side is winning there are also cases where the mean weight curves for left and right sensors developed similar during some time, shown in 1.3.

In order to limit the mutual inhibition on the dynamics several mechanism were tried. In Experiment 2 we switched off plasticity towards the inhibitory neurons and in 3 we limited the connections between motor and inhibitory neurons, so that the inhibitory neurons receive less input current. Both modifications did not yield satisfactory results.

Weight figure	Configuration and mean firing rate (MFR)	
1.1		Small sonar angle MFR: IL:30, IR 10, Sensors:1.5, Motors:1
1.2		Same as 1 MFR: IL:10, IR 30, Sensors:1.5, Motors:1
1.3		Same as 1 MFR: Inhibitory: 10, Motors:1, Sensors:1.5

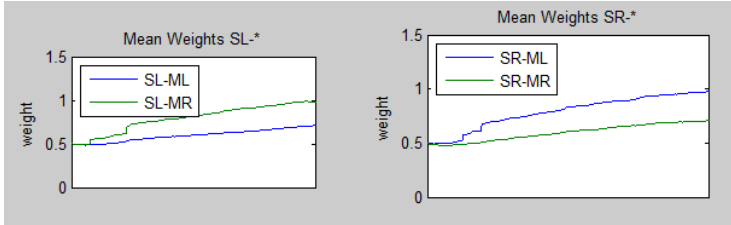
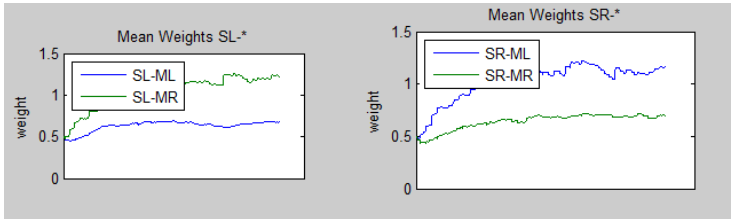
2		As 1, but without plasticity between motor and inhibitory layers to mitigate increase in inhibitory firings. MRF: Inhibitory: 10, Motors:1, Sensors:1.5
3		Same as in 2 but with sparse random 50% connectivity between motor and inhibitory layer. MRF: IL:2, IR:2, Sensors: 1.5, Motors:1
4		Same as 3 but with additional sensor and motor layer noise MRF: Inhibitory: 10, Motors:5, Sensors:3

**Table Neural Controller Training, Experiments 1-4.**

It becomes clear that having such a large sonar angle and thereby enforce the robot to approach a food source in a very specific angle has adverse effects on the learning dynamics that cannot be mitigated easily. Out of 1-4, experiment 4 where we introduced additional external noise shows the most stable dynamics. Why might the introduction of noise contribute to stable dynamics?

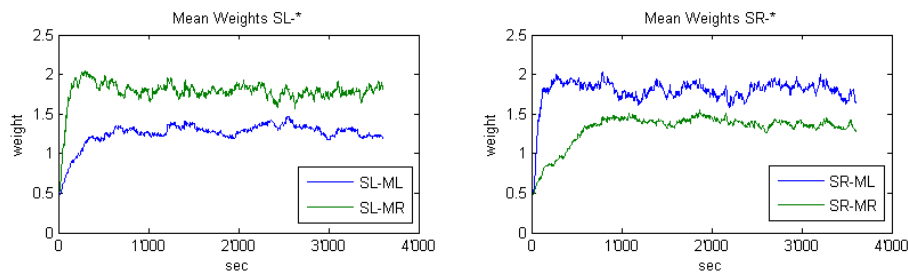
In STDP the amount of noise affects the learning process. Naturally there are many pairs of neurons that fire nearly coincident spikes by chance just before we deliver a reward. These connections are therefore also rewarded. However since the order of firing of those neurons is random, over time they cancel each other out. Due to the asymmetry of the STDP function (LTD>LTP) in fact it results in a net decrease of the synaptic weights. At the beginning synaptic connectivity between sensor and motor neurons is uniform. A sensor and motor layer is stimulated by an external spike source during the period when an object is within the sensor range. The motor spike source is the training current; the sensor spike source comes from the sensor reading. Since the stimulated sensor layer projects to both motor layers equally strong, both layers are activated. Overall, the difference in spikes is defined by the magnitude of the training current. The training current must be significant to increase the chances for correlations of spike pairs between the “correct” layers. Further the background firing or noise that is injected to sensor and motor neurons must be strong enough to cancel out synaptic weight increases from spike pairs between the “wrong” layers. We therefore need to find a balance between training current, sensor input current and background noise to enable statistically rewarding the right correlations. In experiments 1-3 we keep the background noise low. In experiment 4 we increase the noise in the motor and sensor layer so that there is a considerably higher MFR in all layers. The weights still increase more or less parallel between the layers, but at a lower level. The lambda factor in experiment 4 is maximal 1.2 and still far from our target value of 3.

In the next experiments, shown below, we decrease the sonar angle from 90° to 72° and also increase the training current. Introducing a smaller sonar angle clearly stabilises the learning curves as shown in table 4-2. In all experiments the weights are initially diverging but start to consolidate or even converge later.

	Weight figure	Configuration and mean firing rate (MFR)
4		MRF: Inhibitory: 10, Motors:4, Sensors:3
5		MRF: Inhibitory: 10, Motors:5, Sensors:3

**Table 4-2 Neural Controller Training, Experiments 4, 5**

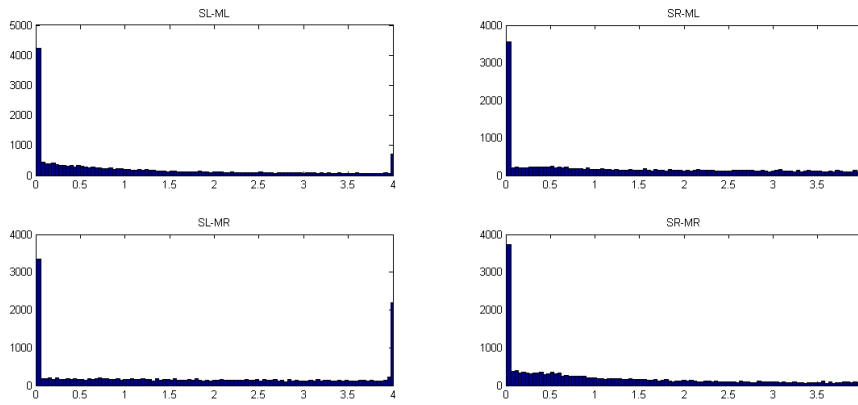
The mean separation of weights is about 0.5 in experiment 5. We build on this result and conduct another experiment where we further increase the external background noise. To judge the quality of the model we run it 30 times for 60min. The resulting weight changes were consistent over all runs and displayed the following patterns similar to experiment 5. The mean separation between weights is again about 0.5. Mean firing rates are identical for motor and sensor neurons and about 11 Hz for inhibitory neurons.



**Figure 4-6 Experiment 6 run for 60min**

How can we explain this pattern? Initially we have an all-to-all connectivity. Due to noise and spill over effects we might increase the wrong connections at times, although these should cancel out over time. Due to the competitive nature of STDP we expect that certain synapses between the sensor and motor layer will gain its maximal strength rapidly and some that happened to be uncorrelated at the beginning will rapidly get depressed. The result should be a bistable weight distribution as discussed in 3.4. Figure 4-7 confirms this expectation only to some extent. There are clearly two extremes of zero weights and maximal strengths. But also there is a considerable amount of neurons with uniformly distributed weights between 0 and 4. We assume that there should be still room to improve this result since relatively few synapses reach the maximum strength  $s_{max}$  at the SR-ML and SL-MR layer connections.

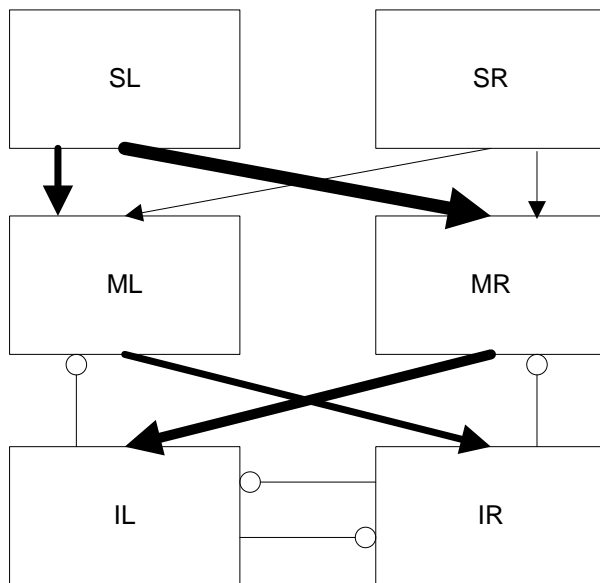




**Figure 4-7 Weight distribution experiment 6**

## 4.2.2 Inhibitory effects

In experiment 2 we switched off plasticity between excitatory and inhibitory neurons. We continued to experiment with and without plasticity in mutual inhibition and found that plasticity between motor and inhibitory layers plays an important modulating role in the training of the controller. Consider the slight but steady increase in connection strength between left sensor and left motor neurons as in experiments 4 and 5. This increases the spiking in the left motor layer and propagates to the right inhibitory neurons. If we apply STDP also the later connections will be enforced.



**Figure 4-8 Changing connection strengths in neural controller.**

The thickness of the line represents the strength of the connection. Inhibitory connections are not plastic.

The stronger inhibitory force will damp the increased membrane potential in the right motor layers but the left motor layer inhibition will be stronger as seen in the figure above. In fact the WTA mechanism will shut down activity in the right inhibitory population. If there is no plasticity towards the inhibitory neurons, the motor response from a left sensor stimulus will be less clear. Figure 4-9 and figure 4-10 illustrate this fact further. They both show firing patterns after some learning when the robot turns just in front of the object, receiving a strong stimulus. Figure 4-9 shows the pattern without plasticity between motor and inhibitory layer, figure 4-10 shows the same situation but with plasticity between motor and inhibitory layer.

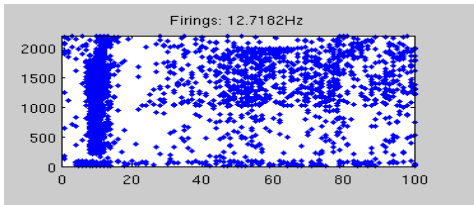


Figure 4-9 Without Motor-Inhibitory Plasticity

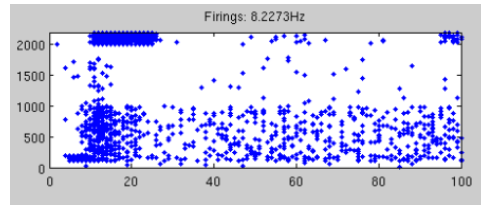


Figure 4-10 With Motor-Inhibitory Plasticity

We already discussed the stability issues in learning that occur with higher spike frequencies. Inhibitory neurons are fast spiking neurons and spike with the highest frequencies of the neurons in the controller.

In the following experiment we decrease the excitability of the inhibitory neurons. This is done by decreasing the parameter “a” in the neuron model from its original value 0.1 to 0.001. This value is biological not realistic, as it is believed to be within the range of [0.02,0.1], see [26]. This modification increases the refractory variable “u” and limits the firing. To keep a certain amount of mutual inhibition still in effect we increase the initial synaptic weights to postsynaptic neurons from  $\sim U(-1,0)$  to  $\sim U(-2,-1)$ . To test for stability we let the experiment run for 60 min simulation time, which results in about 2400 trials, given each trial lasts for 1s on average (the steps per trial are limited to 12, whereas each step is 100ms, see 3.5). At the same time we also reduce the background firing on the motor neurons so that all populations spike below 1Hz when there is no stimulus. Figure 4-11 shows the usual statistics for this configuration.

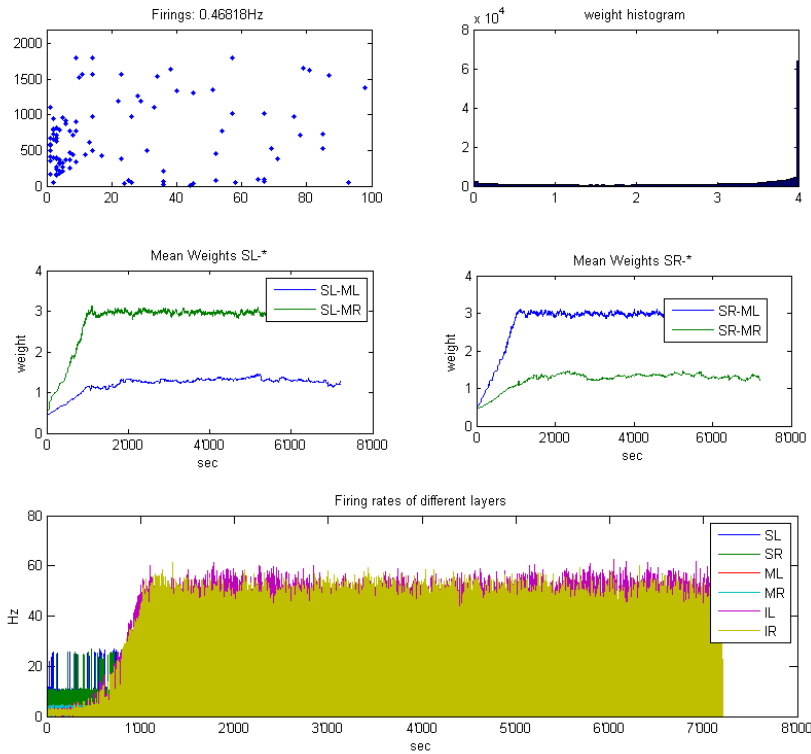
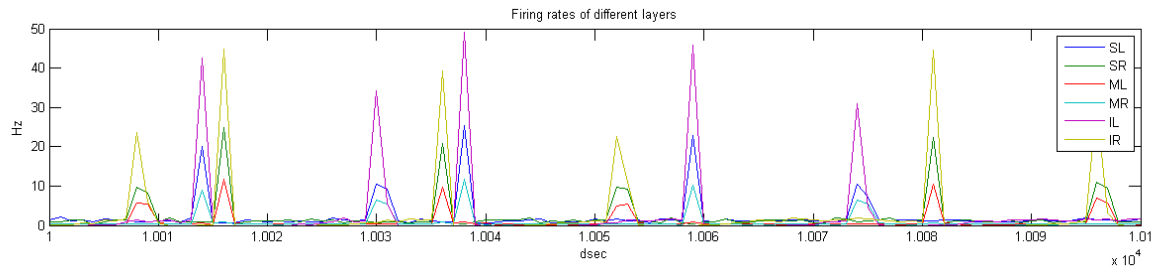


Figure 4-11 Training with modified inhibitory neurons

Separation of connection strength lambda is at 3 after the training. This is the best value we saw so far. The weights on the diagonal connections increase constantly during about 15minutes and then remain constant at about 3mV. The parallel connections also increase but only slightly to about 1mV and stay at that level after that. The bottom row of figure 4-11 shows the change in mean firing rates of the individual layers. The mean firing rates increase in correspondence to the weight changes. To see those spiking patterns more detailed figure 4-12 shows an excerpt of 10s after 1000s of learning. When there

is a stimulus, inhibitory layers spike the most at a gamma rhythm of 20-50Hz. Sensor layers spike 6-20Hz and motor layers spike generally up to 10Hz. During the quiet periods all layers spike below 1 Hz. This low frequency of regular spiking makes the probability that a coincidental spike pair is rewarded during training very low. Spiking in case when there is a stimulus is always at least to a factor of 3 higher between correlated populations.



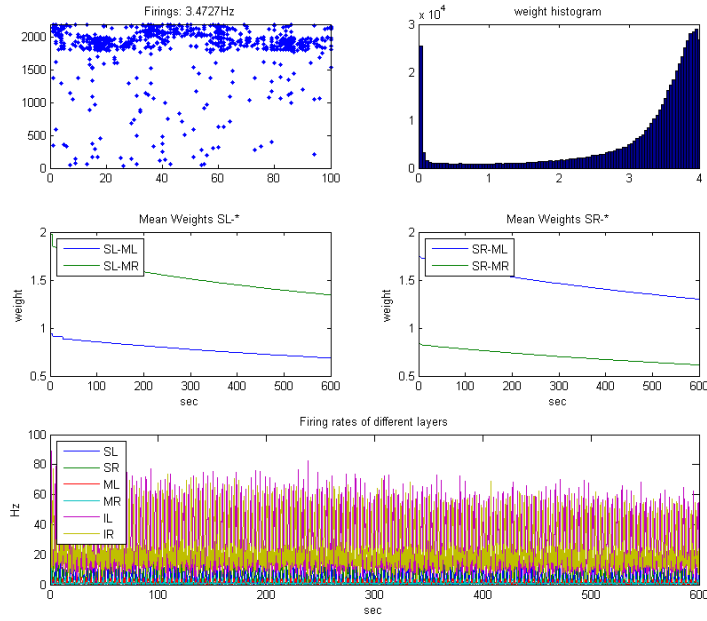
**Figure 4-12 Firing Rates during trainings**

### 4.2.3 Dynamic training

Next we apply the dynamic training as described in chapter 3.5.1.1. The idea is basically the same as with what discussed in 4.1.2.2: We want to keep the difficulty of getting a reward constant.

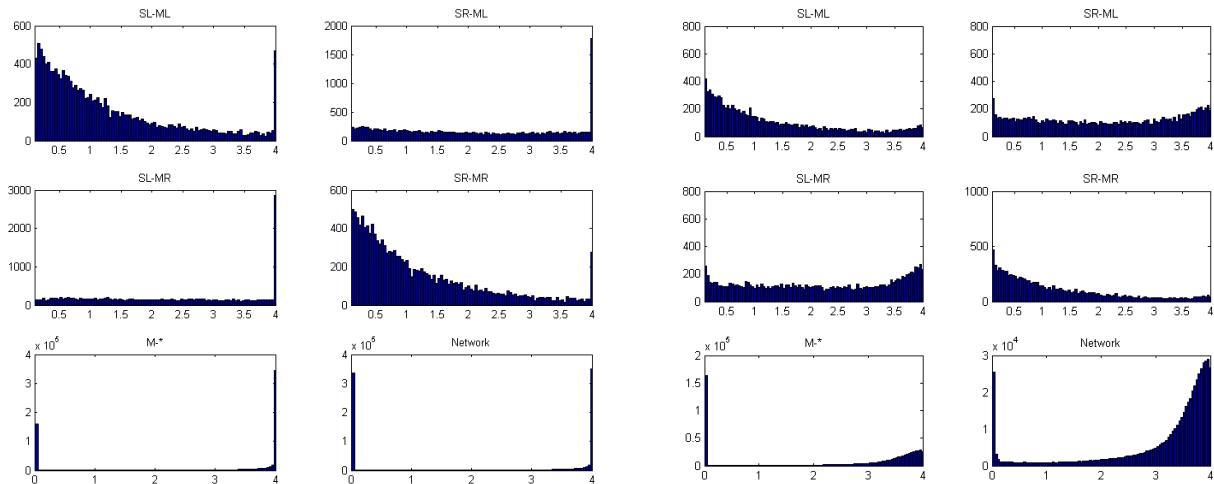
To illustrate the dynamics figure 4-13 shows the result of a too small training current applied to an already pre-trained network. In this setup the robot will only occasionally hit the object during training because the spike count difference between the left and right motor neurons is too small. In the first row of the figure 4-13 we show the firing patterns of the last trial and the synaptic weight distribution over all synapses. As defined in 3.2 neurons are allocated to layers by their number. Neuron number 1-100 are left sensor neurons (SL), 101-200 are right sensor neurons (SR), 201-1000 left motor neurons (ML), 1001-1800 right motor neurons (MR), 1801-2000 left inhibitory neurons (IL), and finally 2001-2200 right inhibitory neurons (IR).

The effect on the mean weights is a small but steady decline. We argue that this decay is the result of a too weak training supplied to a network that still has considerable parallel connections between sensor and motor neurons. The difference between the parallel and diagonal firing correlations i.e. SL-ML (SR-MR) vs. SL-MR (SR-ML) is not significant enough for the STDP mechanism. Accordingly to the weight decline the firing frequencies decrease also slowly until the end of the training after 60min.



**Figure 4-13 Effects of too little training current**

Figure 4-14 shows the weight distribution between the layers for this experiment. At the beginning most of the diagonal connections are the maximum and dominate the rest of the synapses that is evenly distributed along the axis. 16% of SL-MR (2.8% of SL-ML) and 10% of the SR-ML (1.7% of SR-MR) are at its maximal strength. The weight distributions of the parallel connections on the other hand have a bistable shape with a decaying number of neurons along the weight axis. After the training the distributions changed dramatically. The mean weights for all the connections decreased by about 0.5 and only 4% of SL-MR (and 1% of SL-ML) are at its maximum. Similar for SR-ML only 3% and SR-MR only 1% are at 4mV. Besides the sensor to motor connections strengths also the motor to inhibitory connection strengths decreased dramatically. Where at the beginning most of the synapses between motor and inhibitory neurons were at the maximum of 4, after the training most of the synapses are zero.



**Figure 4-14 Weight distribution start (left) and end (right) of training**

We now change our training mechanism to include a feedback loop which adapts dynamically the external training current as discussed in 3.5.1.1. The thick marks in figure 4-15 indicate the point of time where training current has been in- or decreased. For example at  $t=137$  training current into the right motor neurons - after a stimulus to the left sensor neurons - is increased. This has the effects that the

mean weight of the SL-MR synapses increases dramatically. The same effect is observable at various points of modification. However our feedback loop does not succeed at increasing the separability between the two competing layer connections. Every decrease in training current results in either a rapid convergence or does not have any effect at all. It would be interesting to see how much (left or right) motor spiking is caused by noise versus stimulus during (right or left) turns. This would maybe allow us to build a more elaborate feedback controller. We could reduce the noise inversely proportional to its share in causing motor neurons to spike. Unfortunately due to limited time this was out of the scope of this project.

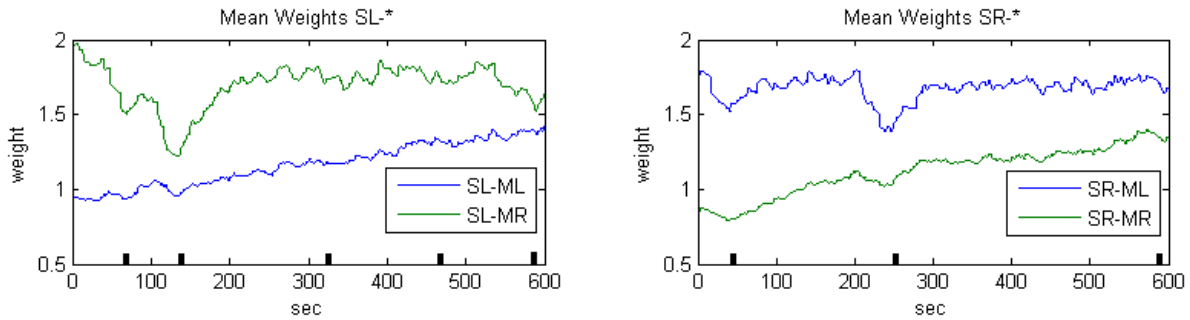


Figure 4-15 Changing weights at dynamic training

#### 4.2.4 Activity dependent scaling

In 4.1.1 we introduced activity dependent scaling (ADS) as a biological inspired model way to modulate neuronal excitability. ADS can be applied as a mechanism that is working on the individual neuron or on populations of neurons. In the preliminary experiment the population scale version performed best. We therefore apply equation (4-4) to the whole motor population and set  $w_{TOT}$  to

$$w_{TOT} = \sum_l \sum_i w_{l,i}$$

where the summations are over the motor layers (left and right). For the rest we use the neuronal controller configuration as described in the modelling chapter with the following specifics:

##### Encoding settings

- $U_{max}=U_{min}+U_{min}/3$
- Sonar range of 20, sonar angle  $\alpha=72^\circ$
- $R_{max}=4\text{Hz}$  initially and is dynamically updated during training as described in 0

##### Training settings

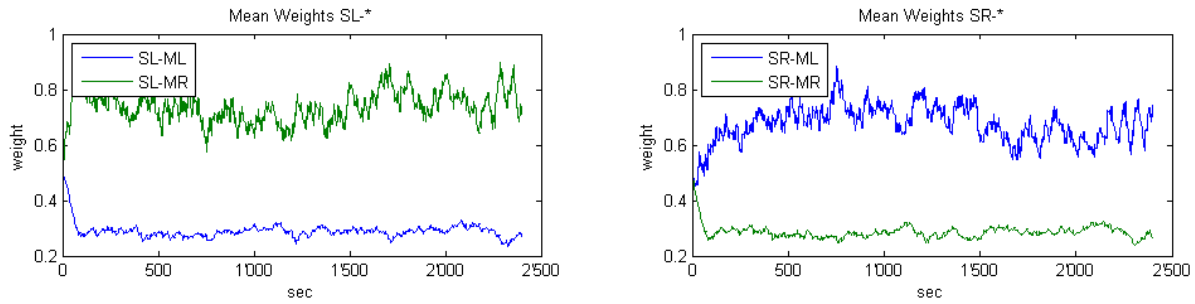
- Training current is kept constant as Poisson random input with mean 0.7 to the left or right motor neurons
- Training stop condition at  $\lambda=2$

##### Background noise

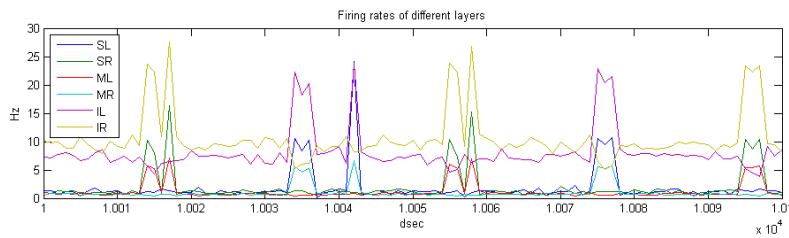
- Constant background noise as uniform distributed random number to all neurons in the range of  $[-6.5, 6.5]$
- Additionally external motor noise current as a Poisson input stream with mean 0.2

Results are shown in figure 4-16, figure 4-17, and figure 4-18. The training period was 60min, during which  $\approx 2400$  trials were executed. The mean weights of diagonal layer connections become  $\lambda=2.5$  times bigger than the parallel connections. After the initial divergence that happens within the first 150s the mean weights stay approximately the same until the training ends. Spike frequencies of motor and sensor layer when there is no stimulus is around 1Hz. When the robot is approaching an obstacle this increases up to 10Hz in the sensor layer and up to 6 Hz in the motor layer. For the inhibitory population the steady rate is 8-10Hz and when activated through motor spiking this goes up to 30Hz.

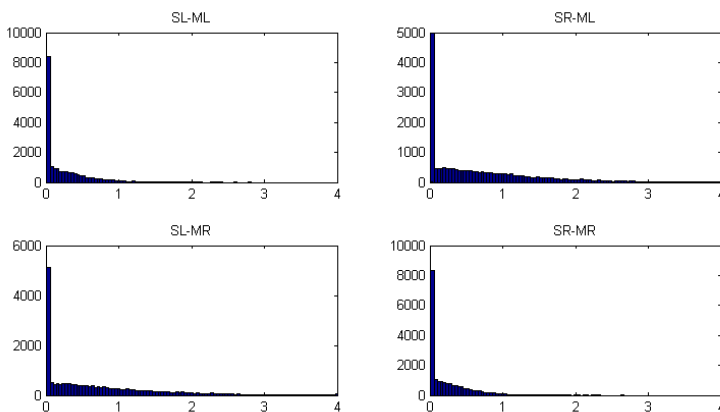
The weight distribution is depicted in figure 4-18. ADS induces unipolar stability into the weight distribution. It dampens the weight increase so that the maximum synaptic weight rule does not have to be applied with most of the weights since they stay below the 4mV.



**Figure 4-16 Training with ADS**



**Figure 4-17 Firing patterns for ADS training**



**Figure 4-18 Weight distribution for ADS training**

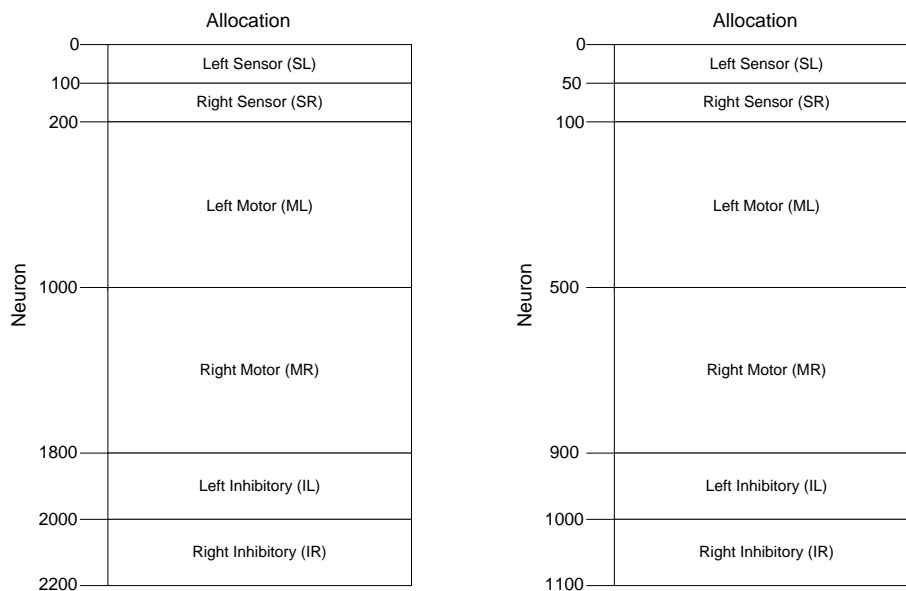
## 4.2.5 Network size

Experiments so far have been conducted with the neural network of 2200 neurons. The number of neurons was chosen to be able to represent some of the basic functions of the motor sensor interactions. We wanted to test if the results so far also hold with a simpler, half as big model. Reducing the model size has apart from computational other advantages. Occam's razor suggests to only make a model as complicated as necessary. When we reduce the model size we can gain insights into its dynamics more easily. We tried several scalings of our neuron model using 10%, 30%, and 50% of the original neuron population. We rerun experiment 6 for each scaling for a simulated 60minutes and compared the results after the training. The external noise injected to the motor layer was reduced to achieve the same spike frequency as in the original. As measurement we note the mean and variance of the following variables for the 20 trials conducted: mean synaptic weights and mean firing rates for the several layers.

Network	$MFR_{ML}$	$MFR_{MR}$	$MFR_{IL}$	$MFR_{IR}$	$SSLML$	$SSLMR$	$SSRML$	$SSRMR$
Original	5.0/1.62	5.0/1.9	11.3/4.3	12/4.2	1.21/0.22	1.8/0.11	1.70/0.10	1.25/0.15
10%	5.5/3.63	5.4/3.3	13.6/4.5	13.9/3.2	2.09/0.51	2.18/0.42	2.52/0.61	2.35/0.34
30%	5.3/2.23	5.2/1.9	12.6/3.7	12.4/2.2	1.97/0.34	2.01/0.10	2.14/0.28	2.25/0.45
50%	5.1/1.16	5.0/2.0	10.9/4.2	1.2/4.9	1.17/0.14	1.72/0.20	1.71/0.12	1.15/0.19

**Table 4-3 Effects of network scalings**

From table 4-3 we see that the results between the 50% scaled network and original network are largely consistent in terms of mean firing rate and synaptic weights. Smaller networks however show different behaviours. Using only 10% of the neurons introduces a greater variability in the learning. The mean weight differences between the parallel and diagonal layers are less significant in the 10% as well as 30% scaled network. We further test this result of reducing the network 50% by reproducing the changes for ADS and inhibitory effects as discussed previously and get similar results. We will therefore for new experiments use the smaller 1100 neuron sized network as default.



**Figure 4-19 Neuron allocation in original network (left) and 50% scaled down network (right)**





## 4.2.6 STDP Learning rate

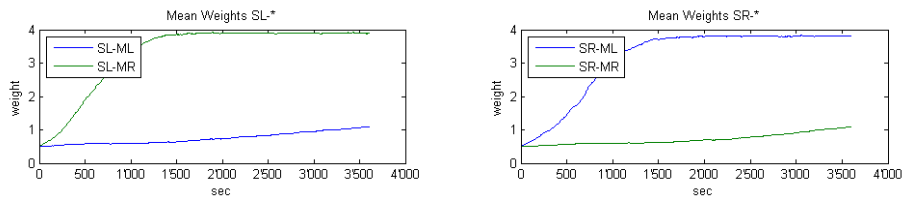
In the STDP learning rule

$$\Delta w = \begin{cases} A_+ \exp(-\Delta t/\tau^+) & \text{if } \Delta t > 0 \\ A_- \exp(\Delta t/\tau^-) & \text{if } \Delta t < 0 \end{cases}$$

we were using  $A_+ = 0.1, A_- = -0.15, \tau^+ = \tau^- = 20ms$  following Izhikevich (2007) [14]. The magnitude of the STDP change can be interpreted as learning rate. In the following experiment we try different learning rates. We try  $A_+ = 0.005, A_- = -0.0053$  and leave  $\tau^+ = \tau^- = 20ms$  following [44],  $A_+ = 1.03 \cdot 10^{-4}, A_- = 1 - 0.51 \cdot 10^{-4}, \tau^+ = 12ms, \tau^- = 38ms$  following [45], and  $A_+ = 0.01, A_- = -0.011, \tau^+ = \tau^- = 20ms$ . The results of these changes dramatically change the outcomes from the previous experiments conducted. The decrease in the learning rate turns out to enable a constant and stable weight increase for diagonal layer connections. The decrease of the asymmetry  $A_-/A_+$  prevents the parallel connections SR-MR and SL-ML to increase. Both changes result in a stronger separability between the motor responses.

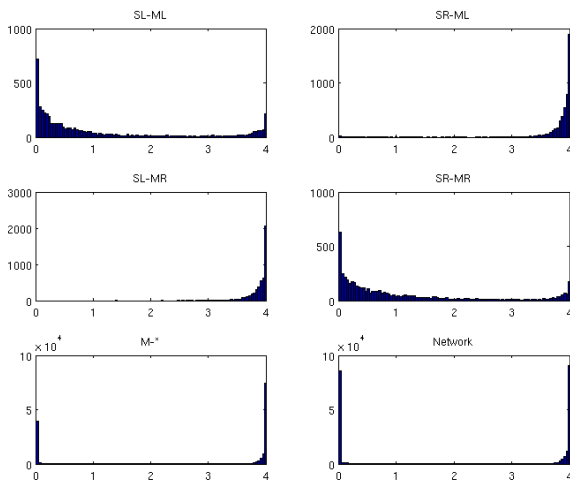
Among the three mentioned learning rate changes we found however no significant difference. All do perform equally well and learn stable attractor behaviour. The mean weights between the diagonal layer connections become more than twice as big as the parallel connections in less than 5 minutes of training. To confirm the significance of the result we let the trainings run 30 times. During these runs we see only a small 10% variability in the mean weights learnt.

The following graph shows the learning for a period of 60 minutes.



**Figure 4-20 Stable learning with smaller learning rate and less LTD asymmetry**

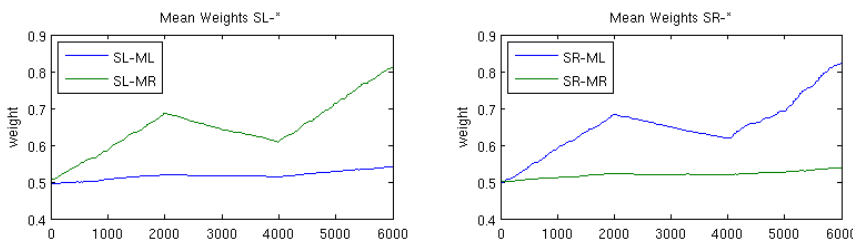
The weight histograms after 60 minutes depict a bistable shape for the parallel connections. For the diagonal connections most synapses are at their maximum.



**Figure 4-21 Weight distribution for smaller learning rates**

An implication of the STDP rule is that there is a linear relationship between the strength of a synapse and the probability of a postsynaptic spike: The stronger the weight the more likely it is that the next neuron will spike. Thus, once a synapse is potentiated its chances for subsequent potentiation increases. When using a big learning rate, the network seems to be more affected by random movements which will potentiate also the parallel SL-ML and SR-MR connections. That in turn prevents a more accentuated response from the diagonal connections SL-MR and SR-ML because of the competitive nature of STDP.

To check the robustness of this finding we switch the reward into a punishment after 200s. A punishment is implemented by negative dopamine as we will discuss in chapter 4.3. This causes the network to learn a repulsive behaviour. Once the reward structure is switched, the network quickly starts to learn the new behaviour and the diagonal mean weights deteriorate at about the same rate as they have been increasing. We repeat this experiment 20 times, selecting random time delays after which we switch the reward. In all cases the network learns the new behaviour. It is interesting that it does not seem to matter if the synapses were at a minimum or maximum weight, the network consistently learns the behaviour.



**Figure 4-22 Switching reward from positive to negative and back**

#### 4.2.7 An optimal controller configuration for learning

Overall our experimental procedure indicates favourable effects of three important tuning changes:

- Reducing the sonar angle
- Reducing the learning rate
- Introduce homeostatic plasticity in the form of activity dependent scaling
- Limiting inhibitory layer effects on the network paired with decreasing the background noise
- Modulating/increasing the background noise

Out of these modifications reducing the learning rate had by far the most favourable effect. The last two changes are mutual exclusive. The best performing models in terms of separability of motor responses have been with a small learning rate, a smaller sonar angle, and an increased background noise that causes the network to spike at just over 1Hz when there is no other external spike source.

To get a better understanding on optimal configurations we investigated the differences in firing patterns of a successful training trial between the version with low inhibitory spiking and experiment 6. The snapshot of the spike patterns is taken after 15minutes of training. For the version with the low inhibitory spiking the window is 1s, for experiment 6 it's 0.8s. For both models left-turn training is snapshotted, where a stimulus is presented to the right sensor neurons. As defined in 0 the time step to calculate the mean firing rates in the controller is 100ms. Every 100ms the robot updates its sonar readings as well as its position. The results of experiment 6 are shown in figure 4-23 and figure 4-24. The results of the low inhibitory spiking model are shown in figure 4-25 and figure 4-26. Figure 4-23 and figure 4-25 show the neuron firings during the whole time window of 1s. Figure 4-24 and figure 4-26 show the firing patterns that occurred short before the robot hits the object. Layer firings are

distinguishable by neuron number as defined in the neural controller topology. For this experiment we use the original network size of 2200 neurons in total.

In both figures the firing of the right motor neurons dominate as expected. Most action potentials are fired in intervals of 100ms. The peak in figure 4-23 is after 800ms, in figure 4-25 the peak is after 600ms. This is the time when the robot is just in front of the object and receives a last stimulus.

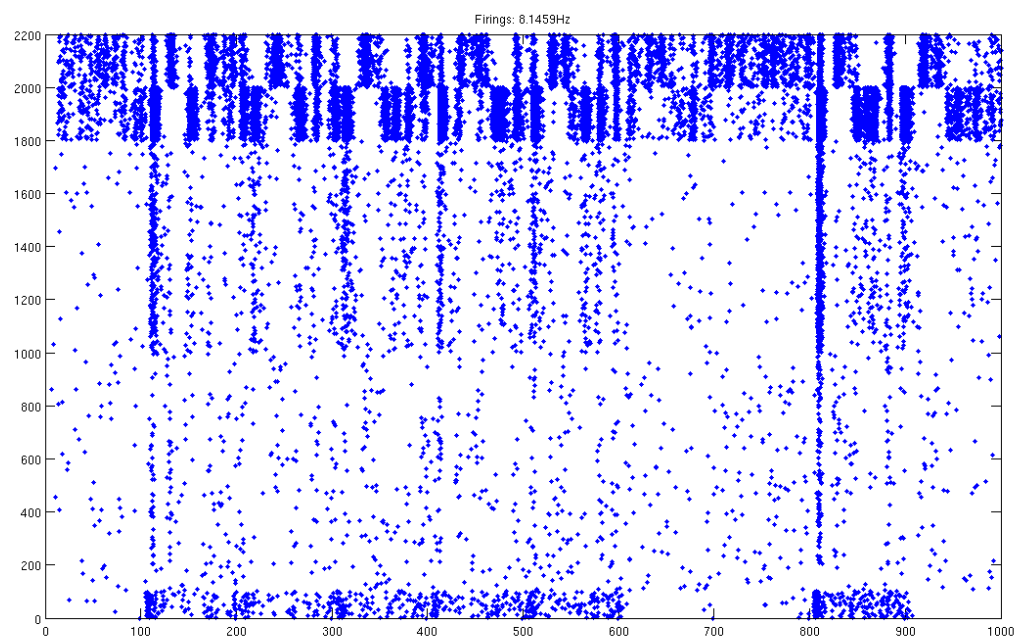
Maybe the most distinctive features are the different firing patterns of the inhibitory neurons. In both versions left and right inhibitory population fire alternately. In experiment 6 the inhibitory neurons spike fast and the winner in mutual inhibition is changing several times during the 100ms. In figure 4-25 the inhibitory firing pattern is different. Even though mutual inhibition is clearly observable the slow recovery causes the right inhibitory neurons to stay quiet after the first burst at  $t=100\text{ms}$ . Only after a new stimulus is arriving short before the robot is touching the object at  $t=600\text{ms}$  the inhibitory neurons fire again. The average spike frequency during the trail is  $\approx 2\text{Hz}$  and therefore four times lower than in figure 4-23.

This low spike frequency causes the probability of coincidental spike pairs that are rewarded with STDP to be very low. Looking at the detail window we see that the firing of experiment 6 is more intense for both neurons. Interestingly the mean weights between the parallel layer connections are for both models about 1mV. The diagonal connections between SL-MR and SR-MR however are with 3mV (see figure 4-11) much stronger in the second model where the value is about 1.8mV (see figure 4-6). Counting the relative number of spikes in each experiment for the motor layer gives the following statistics:

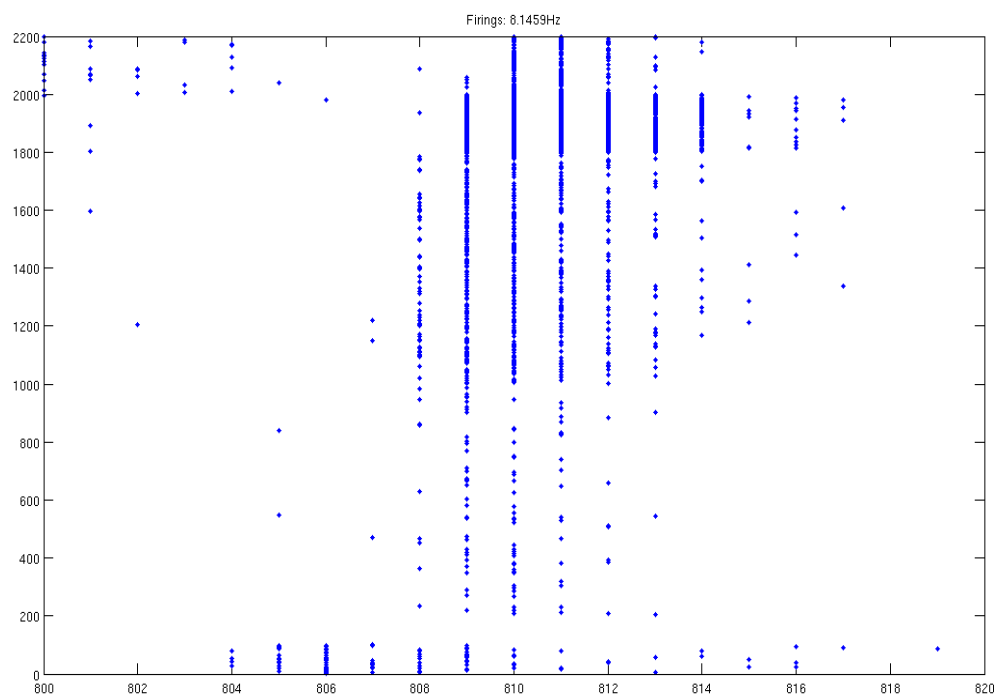
Experiment	Time window	$MFR_{ML}$	$MFR_{MR}$	$MFR_{ML}/MFR_{MR}$
Experiment 6	808-812	104	534	0.19
Experiment with low inhibitory spiking	609-617	66	255	0.26

**Table 4-4 Comparing mean firing rates between two experiments**

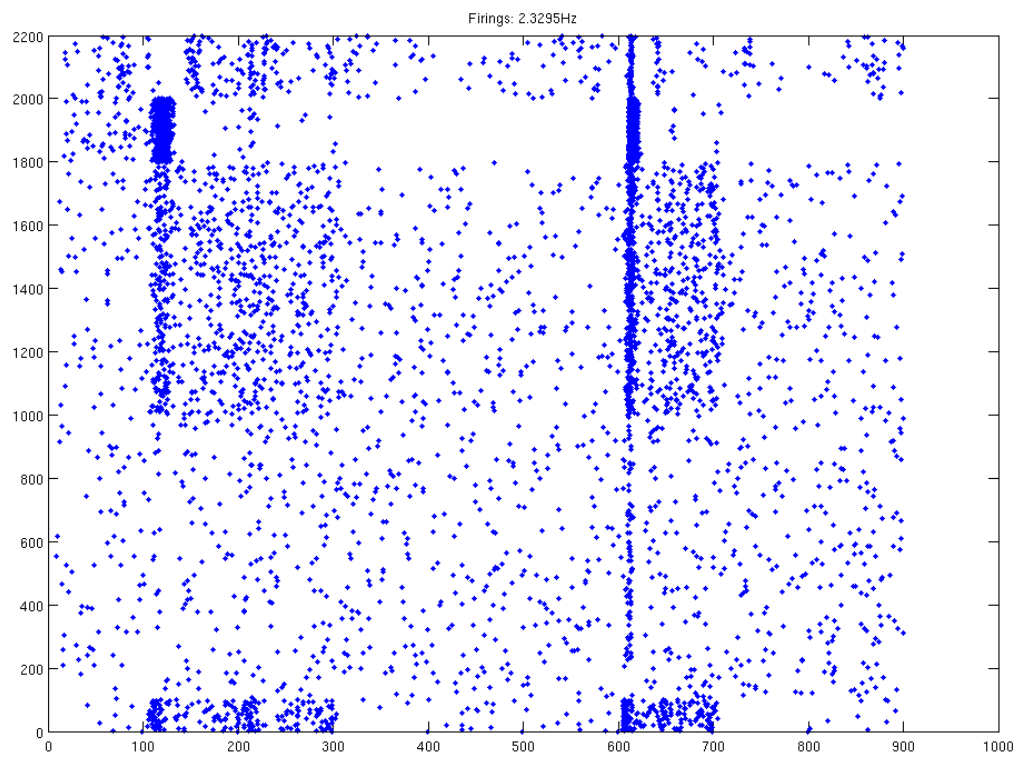
This difference in the relative mean firing rates might be the reason of the different learning behaviours. It is caused on one hand by a more distinguished motor firing, and on the other hand by a clear dominance of right inhibitory activity that prevents most of the firing in the left motor neurons in figure 4-26. There seems to be a “sweet spot” in the amount of firing in the system where learning will be possible.



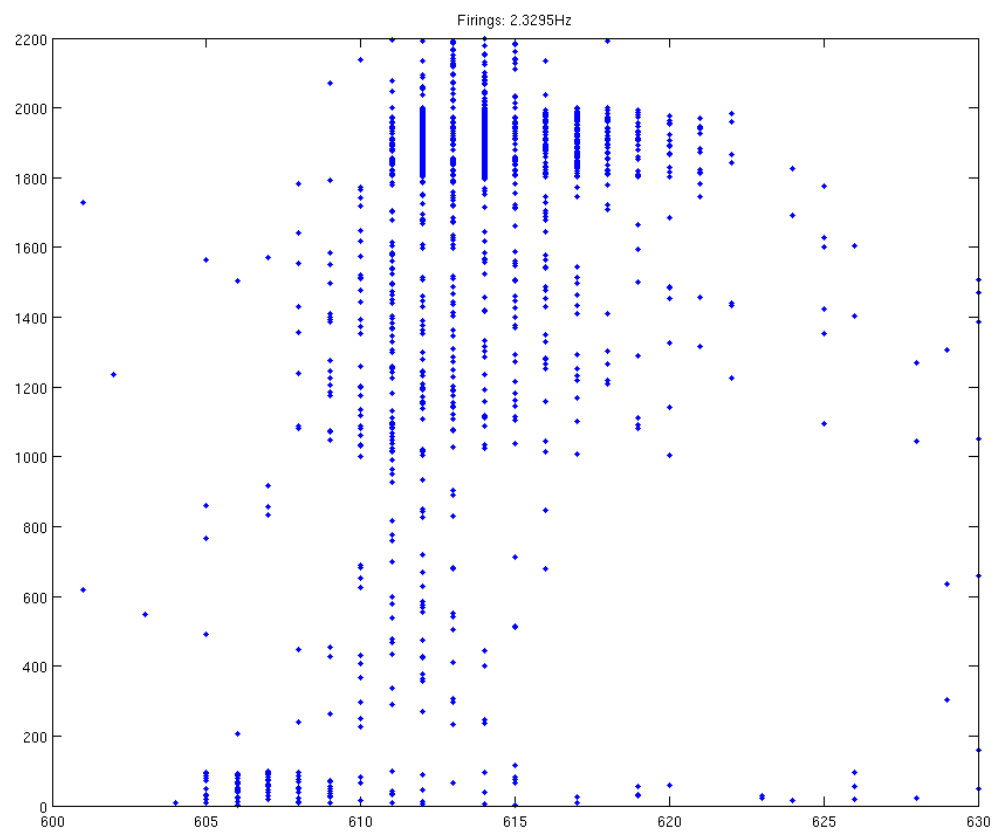
**Figure 4-23 Spike pattern of 1 trial in experiment 6**



**Figure 4-24 Spike pattern of 1 trial in experiment 6 – details**



**Figure 4-25 Spike patterns of 1 trial in the version with low inhibitory spiking**



**Figure 4-26 Spike patterns of 1 trial in the version with low inhibitory spiking – details**

#### 4.2.7.1 Evaluation of the relative mean weight differences

We are interested in the relative difference between the two motor layers responses. The magnitude of the mean firing rate is less important since this is accounted for in the encoder, see 3.5.1. The “Ref” column in table 4-5 refers to the following neural controller configurations: 1: experiment 6 (figure 4-6), 2: experiment 6 with ADS (figure 4-16), 3=slow inhibitory spiking (figure 4-11), and 4: Slow learning rate (figure 4-20). We denote mean weight of diagonal connections coming from left sensor  $s_{Ldiag}$ , and parallel connections as  $s_{Lparallel}$ . Analogous notation is used for the right side. Minimal and maximal mean weights,  $ms_{min}$  and  $ms_{max}$ , indicate the weight spread measured during the whole training period over the four layer connections between sensor and motor population. The bigger the number in  $s_{Ldiag}/s_{Lparallel}$  and  $s_{Rdiag}/s_{Rparallel}$  the bigger the separability, and the more likely it is that the robot shows the intended behaviour. The  $\lambda$  factor (defined in (3-13)) is just the lowest common dominator of  $s_{Ldiag}/s_{Lparallel}$  and  $s_{Rdiag}/s_{Rparallel}$ .

Ref	$s_{Ldiag}/s_{Lparallel}$	$s_{Rdiag}/s_{Rparallel}$	$\lambda$ factor	$ms_{min}, ms_{max}$
1	1.359	1.364	1.359	1.3-1.87
2	2.132	2.058	2.058	0.3-0.66
3	2.151	2.169	2.151	0.138-3.0
4	3.545	3.522	3.522	1.08-3.9

**Table 4-5 Evaluation of mean weight separability**

Clearly version 4 delivers the best separability.

#### 4.2.7.2 Testing

After we have trained the neural controller we test the performance in the environment. We preselect the best performing model that reached the greatest separability and conduct 10 trials each running a simulated 10 minutes. The environment has 15 food resources at random locations. Every time the robot touches one food source the food source is removed from the torus and recreated at a random location. We compare the performance of version 4 against a random walk without STDP, and a benchmark with neuroanatomical constraints. The latter is a robot with no parallel connections, i.e. a robot with only SL-MR and SR-ML connectivity in its neural controller.

The result confirms our expectation. The trained robot collects the same amount of rewards as the benchmark.

c	mean	std.dev
Random walk	99.5	10.3
Benchmark	371.2	7.63
Version 4	369.1	8.87

**Table 4-6 Testing attraction behaviour**

Performance in terms of collected mean rewards over 10 runs including standard deviation (std.dev).

#### 4.2.8 Autonomous training using random walk

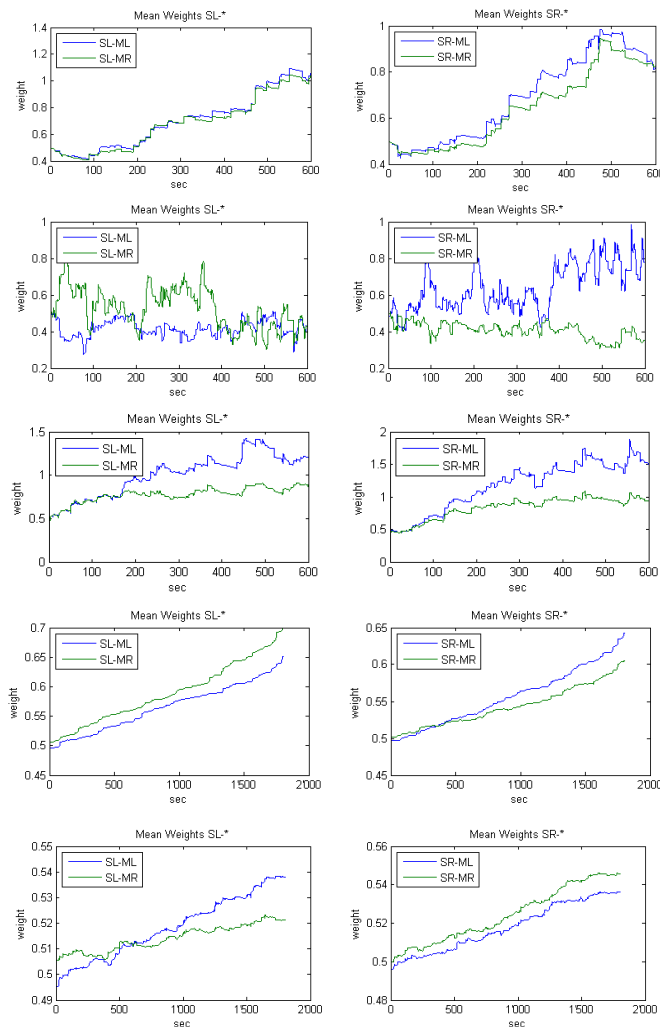
As a next step we try to train attraction behaviour by letting the robot run in autonomous mode. For this we employ the random walk training as described in section 3.5.2. Figure 4-28 shows a typical trajectory. We compare the four configurations as discussed before:

- 1: Experiment 6 (figure 4-6)
- 2: Experiment 6 with ADS (figure 4-16)

- 3: Slow inhibitory spiking (figure 4-11)
- 4: Slow learning rate (figure 4-20)

Only the robot with the small learning rate successfully learns to discriminate motor responses. The  $\lambda$  factor as defined in (3-13) states how much bigger the mean weights of the diagonal connections in compare to the mean weight of the parallel connections are. A value of NA means the network could not learn any distinctive behaviour.

#### Weight graph



#### $\lambda$ factor

NA

NA

NA

1.078

1.02

#### Configuration

Version 1, attractor  
Training: 10 min.

Version 2, attractor  
Training: 10 min.

Version 3, attractor  
Training: 10 min.

Version 4, attractor  
Training: 30 min.

Version 4, repulsive  
Training: 30 min.

**Table 4-7 Random walk training**

Version 4 is the only configuration that learnt the correct behaviour. We will therefore dismiss version 1 to 3 from further experiments and solely continue with version 4. Even though after 30 minutes of training the correct behaviour emerges the  $\lambda$  factor is still quite low.

We test the performance of this network by comparing the amount of rewards collected with an untrained randomly connected network and a benchmark. Since the dynamics are slower and the robot continues learning we let the experiment run for 120 instead of 30 minutes. As before the benchmark is a robot with neuroanatomical constraints. The benchmark as well as the random walk configuration does not have any built-in learning (no STDP). The controller configuration version 4 learns via DA-modulated STDP.

As before the environment contains 15 randomly distributed food sources. Once the robot makes contact with a food source, a reward is delivered to the network and the food source is recreated at a random location.

	mean	std.dev
Random walk	398.2	15.8
Benchmark	1484.5	24.63
Version 4	1207.2	20.55

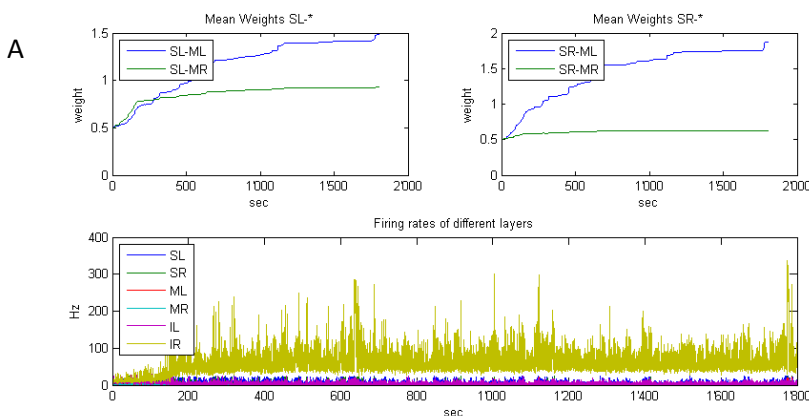
**Table 4-8 Testing random walk training**

The results in table 4-8 show that the robot learns even when not trained in a controlled manner as we did in the single object training. Through STDP the network is flexible enough to filter out the noise and learn the sensor-motor combinations that correlated the most with the rewards. The STDP trained robot collects only 20% less rewards than the optimal configured benchmark does, and three times more than in a random walk. The longer the training goes the better the network will adapt to the reward structure. Rewards collected per minute increase from 3 at the beginning where the robot conducts a pure random walk up to 12 at the 120<sup>th</sup> minute.

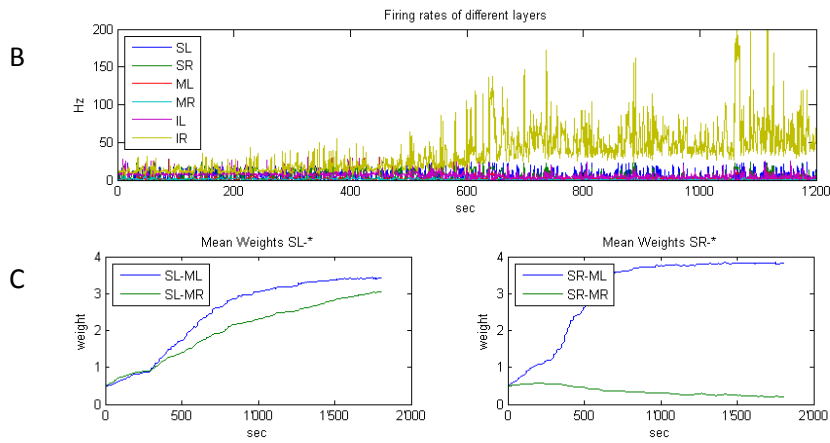
To see if we can speed up the learning we implement built-in “curiosity”: When an object is within certain proximity, measured as percentage of the robots sonar sight, the robot will steer towards the object. We define the curiosity value  $cs$  as percentage of the sonar distance and run experiments with  $cs = 10, 20, 30, 40$  and 50%. With 40 and 50% the robot learns the same sort of lambda separation as in the single object training. With less than  $cs = 30\%$  the system starts to get unstable. Figure 4-27 shows the dynamics in a run of 30 minutes simulation time for  $cs = 10\%$  (chart A and B), and  $cs = 20\%$  (chart B).

The network in chart A learns to turn away from food sources for left sensor stimulus, whereas the right sensor connections learn the correct behaviour. The firing rates in chart A of the different neuron populations depict a noticeable increase in mean firing rates of the right inhibitory layer after 150s. B shows the changes at a larger scale. The increase must be related to synaptic weight changes between the motor and inhibitory layer since synapses going out from the inhibitory layer are not subject to STDP in our model. Although the causes of the increase in weights are unknown, it is clear that such a dominant firing distorts the dynamics. The right inhibitory layer inhibits the right motor layer. Actually we can see that at about the same point of time when the inhibitory layer started firing at a rate of 40-90Hz, the mean weights SL-ML overtook the mean weights SL-MR and continued to increase. Since the right motor was strongly inhibited it could not dominate the left motor spikes anymore preventing any left turns. The same but less pronounced phenomena occurred at the second experiment with 20% curiosity shown in chart C.

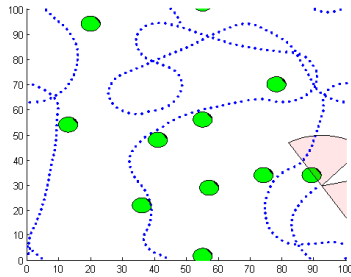
In a WTA architecture with STDP such behaviour is not unlikely to occur. A remedy would be to implement ADS as homeostatic measure. ADS modulates the firing activity to a target rate. The relative synaptic strengths within a neuron are kept constant, but scaled up or down. This could prevent such an unhealthy random dominance of inhibitory neurons.







**Figure 4-27 Network dynamics with curiosity behaviour. A,B curiosity cs:10%, C:20%**



**Figure 4-28 Random walk trajectory**

## 4.2.9 Discussion

In this chapter we analysed how the model we designed can be applied to learn attractor behaviour in a simple continuous environment. We analysed effects of parameter settings and adopted our original model to yield a nearly optimal learning performance within the scope set. Important conclusions in this respect is the importance of a low learning rate, the amount of background noise, and a small solar angle to prevent oscillation when the robot is approaching the food source.

## 4.3 Repulsive behaviour

Training to avoid obstacles is done in a similar way as attraction behaviour, but with a negative reward. The robot is directed into the obstacle. Once the object is hit, a punishment is issued in terms of negative dopamine. Negative values of dopamine can be interpreted as concentration below the baseline. In our model we use a very low baseline concentration of 2nM (see section 3.4). The dynamics between zero and the baseline are too weak to have any modulating effect on STDP and did not yield good results. We tried the following approaches to deal with punishment in our model:

- Increase the baseline of dopamine to a level so that a negative reward would reduce it simply to zero.
- Implement punishment by assuming DA controls the LTP part in the STDP curve. The absence of DA then results in a lower LTP relative to the LTD and an overall depression over time.
- Keep the low baseline at 2nM and have temporarily negative dopamine level in the system. The magnitude of DA is chosen so that the DA concentration recovers to its baseline after 1s.

In the first two approaches the network could not be tuned to learn the right connections. All connections were potentiated. To analyse this result we consult the equations that define the learning dynamics. The weight changes are modelled by  $\Delta w_{ji}(t) = c_{ji}(t)d(t)$  where  $c_{ji}(t)$  is the eligibility trace of a synapse and  $d(t)$  the dopamine concentration. When keeping dopamine constant the dynamics are defined by pure STDP, which enforces correlated spikes. When the robot is driving into the object there is a higher spike correlation between left (right) sensor and right (left) motor neurons than there is for parallel connections. A higher baseline concentration has therefore to be tuned to be able to modulate the STDP impact.

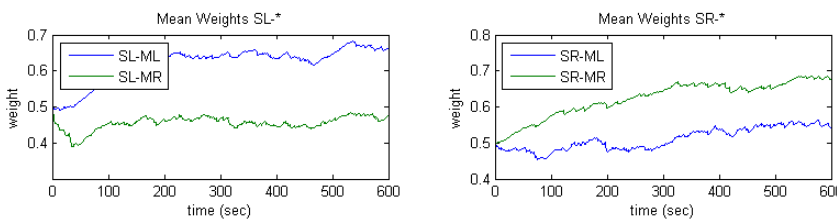
Negative dopamine was the only way we could achieve a desirable learning. Post-pre spike relations that are causing LTD will cause strengthening of synapses in this case while pre-post spike relations will cause a weakening. Coincidental LTD must not dominate the learning. This is dependent on the relation  $A_- \tau^- / A_+ \tau^+$ . What we strive for, is a configuration that depresses the causal correlated connections between diagonal layers in the neural controller. This should after learning increase the probability of the robot turning away from obstacles.

To examine this relation we switch back to the big learning rate of  $A_+ = 0.1$  and modify the relation  $A_- \tau^- / A_+ \tau^+$  under the constraint  $A_- \tau^- / A_+ \tau^+ \geq 1$ . The separation improves with increasing values for  $A_-$ . A lambda factor of 1 in table 4-9 means that the network did not improve separability, and NA means that the network learnt the wrong behaviour in at least one of the direction. We compare 3 different neural configurations that are referred to in the first column. 1: experiment 6 (figure 4-6), 2: experiment 6 with ADS (figure 4-16), and 3=slow inhibitory spiking (figure 4-11).

Ref	$A_- \tau^- / A_+ \tau^+$	$\lambda$ factor	$ms_{min}, ms_{max}$
1	1.5	1.1	1.13-1.3
2	1.5	NA	0.42-0.66
3	1.5	1	2.65-2.74
1	1.1	1.2	0.67-0.83
1	1.05	1.24	0.67-0.832
1	1.01	1.11	0.87-1.00
1	1.005	1.3	0.47-0.68
2	1.005	NA	0.46-0.506
3	1.005	1.085	0.98-1.1

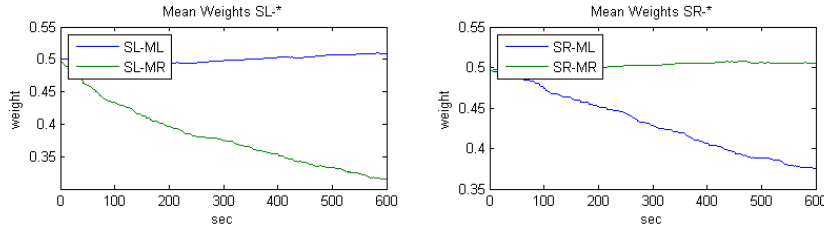
**Table 4-9 Parameter values to find consistent repulsive learning configuration.**

Below is the graph of the mean weight changes for the trial of version 1 with  $A_- \tau^- / A_+ \tau^+ = 1.005$  during a 10 minute learning run.



**Figure 4-29 Effects of LTD to LTP asymmetry in avoidance learning**

Next we use the model with the low learning rate to conduct the same experiment. Since we are punishing diagonal correlation we would expect these connections to be lowered in strength. Figure 4-30 confirms our expectation. As in attraction behaviour the model with the low learning rate performs smoothly and learns consistently the right behaviour.



**Figure 4-30 Repulsive training**

After we have trained the neural controller we again test the performance in the environment. As before the environment contains 15 food sources randomly distributed. Once the robot makes contact with a food source a reward is delivered to the network and the food source is recreated at a random location. We choose the model with low learning rate and conducted 10 trials each for 10 minutes of simulation time. We compare the robots performance against a benchmark with directed connectivity between sensor and motor layers and a random walk without STDP.

The trained robot shows roughly the same performance as the benchmark.

	mean	std.dev
Random walk	99.5	10.3
Benchmark	36.2	3.63
Version 4	37.2	3.8

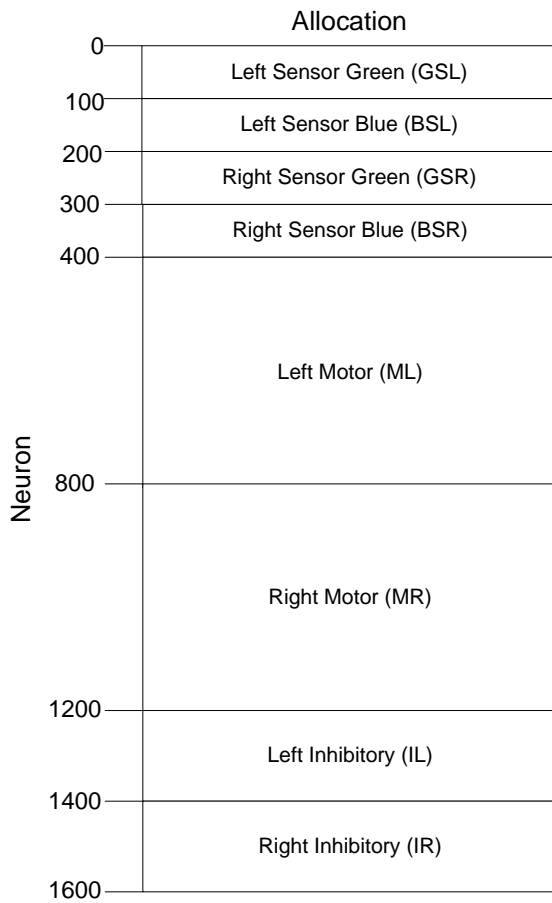
**Table 4-10 Testing for repulsive behaviour**

Performance in terms of collected mean rewards over 10 runs including standard deviation (std.dev).

#### 4.4 Attractor and repulsive behaviour combined

In this experiment we combine attractor and repulsive behaviour. For this we add two additional sensor neuron layers, giving four sensor populations in total. Two sensor layers react to food sources, represented as green objects and two to obstacles represented as blue objects. As before the two sensor layers for each resource type are to recognize left and right side stimulus. The difficulty to the separate training case is now, that the two behaviours might interfere each other. This can easily happen when two objects of different resource types are placed close to each other. The robot then does not know what to do. It might miss a food source or hit an obstacle while trying to collect the food source.

For our model we accommodate a small learning rate of  $A_+ = 0.01, A_- = -0.011$  as discussed in 0, a background noise to achieve a constant amount of spiking at 1Hz when there is no stimulus input, a sonar angle of  $\alpha=72^\circ$ , and a reduced network size of 1100 neurons. The allocation of the neurons is shown in figure 4-31.



**Figure 4-31 Neuron Allocation**

First we train the objects in a directed setting as outlined in 3.5.1. The connections between the sensor and motor layers are learnt to a lambda factor of 2 in less than 5 minutes. The speed of learning is the same as when we were training the behaviours separately. After one hour of training we have almost zero connection strength between the parallel layers for attraction behaviour and a maximum separability for repulsive behaviour. The differences in magnitude of the  $\lambda$  factors in table 4-11 are because we start with a synaptic weight that is uniformly distributed  $U \sim (0,1)$ . In obstacle avoidance training the robot learns from mistakes. The diagonal connections are reduced in strength, and can maximally reach zero, whereas the mean of the parallel connections stay about the same during training, see figure 4-30 for an illustration. In attractor training on the other hand the diagonal connections can increase up to 4mV coming from an average of 0.5mV, which is an 8-fold increase. The robot therefore learns almost the maximal possible separability within 60minutes of directed training.

To test the learnt behaviour we conduct the same procedure as when looking at the behaviours separately. We test the learnt behaviour by letting the robot drive in an environment with 10 food sources and 10 obstacles at any given point of time. Once the robot hits an obstacle a punishment is issued and the obstacle is recreated at a random location. Analogous, if the robot touches a food source a reward is issued and the food source is recreated at a random location. Figure 4-32 shows an example of the environment and trajectory. We compare the performance against a random walk without STDP, and a benchmark with neuroanatomical constraints, and check if there is clear attraction and avoidance behaviour. We execute 10 trials each running a simulated 5 minutes.

The results in table 4-12 show that the performance of the training is as good as the benchmark. Notice that the variance is a bit smaller for the trained robot. A robot with learning capabilities might be able to cope with conflicting goals, so that he learns to tune the motor responses.

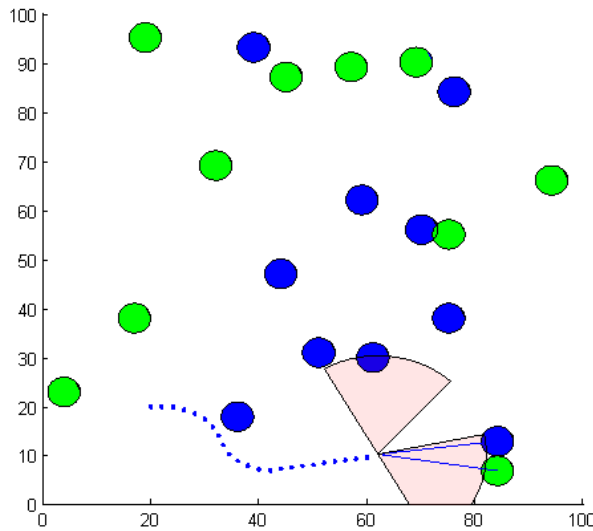
Behaviour	$\lambda$ factors of Learning
Attractor	$SLMR/SLML = 7.15$
Attractor	$SRML/SRMR = 7.08$
Repulsive	$SLML/SLMR = 2.15$
Repulsive	$SRMR/SRML = 2.44$

**Table 4-11 Training for attractor and repulsive behaviour**

	rewards		punishments	
	mean	std.dev	mean	std.dev
Random walk	32.0	9.8	31.5	10.6
Benchmark	123.1	8.63	6.6	3.4
Trained robot	122.1	6.24	5.9	2.8

**Table 4-12 Testing performance of combined behaviour**

Performance in terms of collected mean rewards and punishments over 10 runs including standard deviation (std.dev).



**Figure 4-32 Trajectory in attractor and repulsive behaviour**

#### 4.4.1 Autonomous training using random walk

As a final experiment we try to train the combined attractor and repulsive behaviour by letting the robot learn autonomously using a random walk. The random walk training is described in section 3.5.2. This is the most difficult task so far. The robot starts off with uniformly connected sensor to motor neurons and knows nothing about the environment. Only from correlations between rewards and spike patterns the robot has to learn to avoid certain objects and make contact with other ones. Objects of both types are distributed randomly in the environment. There is likely the situation when there are two objects close together and the correlation between a certain sensor-motor activity and a reward or punishment cannot be made, as illustrated by figure 4-32.

The number of objects in the environment influences how many potential conflicting signals the robot is sensing. We conduct the experiment once with 10 objects of each type and once with 15 objects. The results showed that with 15 objects for each resource, the learning performance was considerably worse and the environment seemingly too cluttered. We therefore decided to use 10 objects for each resource type instead.

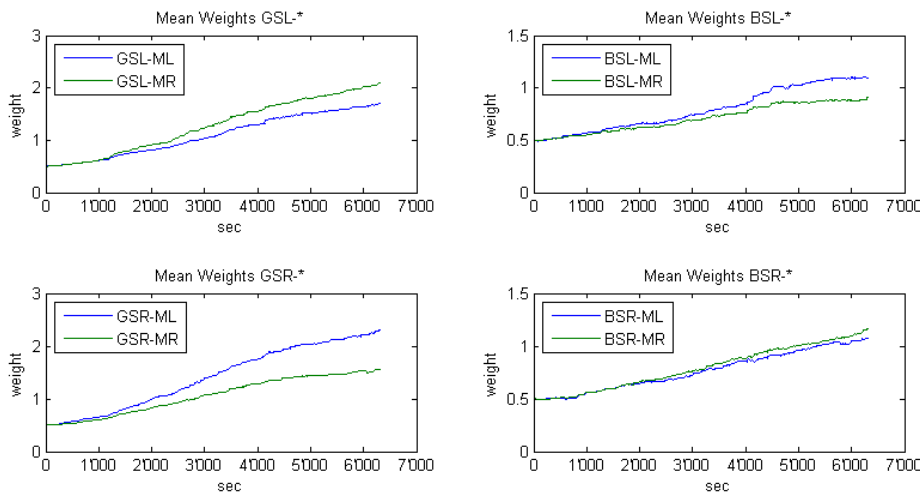
Figure 4-33 shows how the weights between the two sensors and the motor population evolved. The abbreviation G stands for the sensor sensing the green food sources and B for the sensor sensing the blue obstacles. The rest of the notation as before: SL stands for sensor left, SR for sensor right, ML for motor left, and MR for motor right.

The learning is more difficult as tests show. In only 70% of the cases we were able to learn the correct behaviour. In the other cases random influences were too strong. The amount of rewards collected and punishments received is shown in table 4-13. Although the rewards collected are significantly higher than in a random walk, the robot did not perform as well for obstacle avoidance. All mean weights did increase during the training as can be seen in figure 4-34. A greater weight between the parallel connections is generally less harmful for the performance of attractor behaviour than it is for obstacle avoidance.

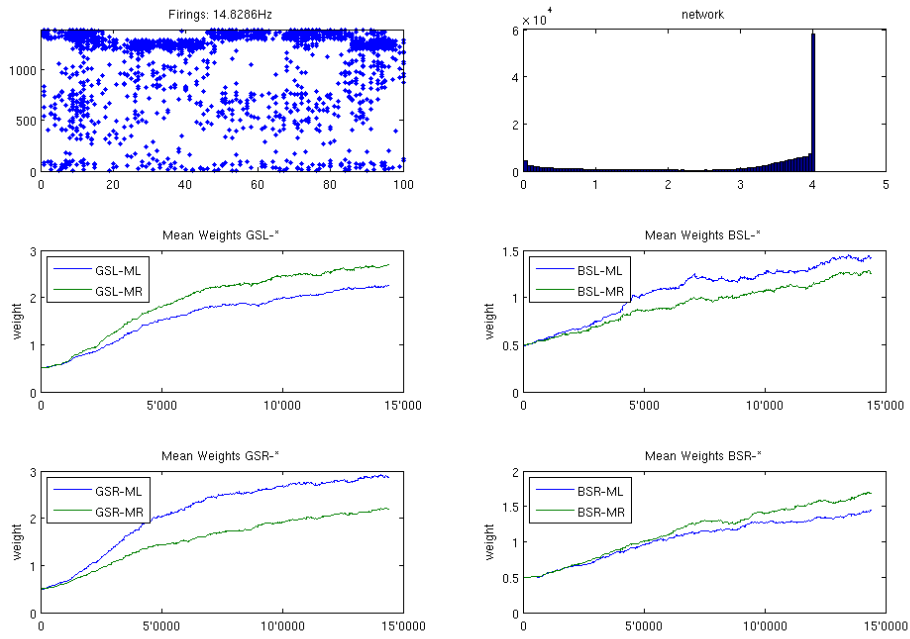
We also notice that in figure 4-34 many weights are at the maximum. This does influence the behaviour of the network substantial as we saw in previous experiments. However with the evidence gathered so far we are confident that there is room using modulating synaptic processes to improve the performance for the combined attractor and avoidance behaviour.

	Rewards		Punishments	
	mean	std.dev		
Random walk	99.5	10.3	102.4	11.2
Benchmark	243.2	31.89	40.8	5.6
Version 4	164.75	30.8	131.5	12.2

**Table 4-13 Reward and punishment statistics testing the combined behaviour during 30min each**



**Figure 4-33 Training combined behaviour using random walk – 10minutes**



**Figure 4-34 Training combined behaviour using random walk - 4 hours**

## 5 Conclusion and Future Work

### 5.1 Project review

We started this project with an evaluation of the state of research in computational modelling of neuronal processes that are involved in learning. Reward modulated STDP defines the core of our model. Reward triggering actions and rewards that come typically seconds after the actions can be correlated by the brain by slow moving synaptic processes called eligibility traces. We applied and extended this model to an embodied context in the form of a simulated mobile robot. Our neural controller that drives the robot has a winner takes all network architecture containing excitatory and inhibitory neurons. It encodes the sensor reading coming from the robot into spike trains and decodes spike patterns coming from the motor neuron population back to odometry commands. Rewards are delivered by events in the environment when the robot makes contact with a certain resource. A resource can be a food source or obstacle. This is a typical instrumental conditioning problem. We show that not only rewards but also punishments can be modelled using dopamine. This is when we interpret a negative reward as dopamine concentration below a certain baseline. The design of our robot was inspired by Braitenberg vehicles and we can show how manifold behaviours that have been described by Valentino Braitenberg can in fact be learned using reward modulated STDP. This is a novel result which has to our knowledge not been demonstrated yet. By a detailed investigation into the dynamics and parameters effects we find a constellation where a robot with random projections between sensor and motor neurons over time performs as good as when we impose neuroanatomical constraints to wire behaviours fix into the controller. In some versions of our neural controller we combine the competitive Hebbian learning mechanism with a modulating homeostatic learning mechanism. For this we apply activity dependent scaling to the controller. Studying the effects and interplay of various forms plasticity is an area where relatively little is known and has not been researched much. However in order to understand how learning works this seems to be an important aspect and computational models hold great promise for insights in this area.

The challenges in this project were largely concerned with parameter tuning and the random components in the model. For many of the components there is a range of possible parameter values that can be used. We approached this issue by systematically analysing the effects of biological relevant value ranges. This was done by first scaling down the network size to only few neurons and eliminating as many random variables as possible. We then gradually increased the complexity of our model only ever changing one parameter at a time to control the effects. This approach has proven beneficial and enabled us to control the model.

We also investigated into using a genetic algorithm to find an optimal parameter combination. We would define a number of parameters and value ranges of these parameters that are subject to genetic modifications. We could then judge the fitness of evolved neural controllers against each other. This approach however emerged quickly as computational unfeasible with the resources at our disposal. A single evaluation of the fitness function involves a learning and testing process that takes considerable CPU time because of the slow synaptic processes that are simulated. In terms of modelling we evaluated several existing neural simulators. Part of the software has been written using NeMo, a C++ library for high performance real-time simulation of spiking neural networks. Unfortunately several issues and limitation in the STDP model forced us to abandon this library and write our own implementation (see also Appendix for details).



Having a bidirectional communication between a robot and an intelligent plastic neural network enables a range of possibilities to create autonomous robots that can evolve its behaviours and rule set depending on the environment.

## 5.2 Future work

We demonstrated how a robot can learn attractor and repulsive behaviour. This could be extended to model all sorts of additional behaviours such as hunger or aggression. These could be dynamically changing for example hunger would be a temporary state until satiation occurs. Another possible extension is to incorporate novelty into rewards. For example it has been shown [19] that novel stimuli cause phasic burst in dopamine neurons of animals. The reason for this might be that animals have learned to expect that most often stimulus predict a reward and generalize initial response for a novel event. A novelty bonus could be added to the reward, depending on the state and type of an object in the environment. Such a bonus would foster exploration.

In our model we use only limited sensor capabilities for the simulated robot. As interesting extension one could add additional sensory capability like vision, location, or temperature. The information gathered could be used for the robot to create an internal representation of its surroundings which can be used to enable a particular behaviour. In our model we use a rate encoding model for the interface between the robot and the neural controller. For visual sensors, different encodings such as rank order or population encoding might be more appropriate.

An area where much research still needs to be done is the effects of different plasticity mechanisms. We only scratched the surface there by integrating synaptic scaling with STDP. Additional homeostatic plasticity such as intrinsic plasticity, as well as structural plasticity and meta-plasticity all play important roles in the highly dynamic connectivity of the brain.

## 6 Appendix

### 6.1 Implementation and Toolkits

The model of this thesis has been implemented in Matlab 7.11 (R2010b). Some of the files are implemented as Matlab class files using the object oriented feature of Matlab that has been introduced in R2008a. Some scripts are implemented in Matlab GUI and built using GUIDE. Originally we used NeMo as our neuron simulator. NeMo is a Spiking Neural Network (SNN) simulator. It is implemented as a C++ class library with APIs for Python, PyNN, Matlab, and pure C. Learning is supported through spike-timing dependant plasticity. Unfortunately some issues and short comings related to the implementation of the STDP functionality emerged with this library that forced us to abandon it and implement the simulator in vectorized Matlab scripts. Concretely in the version 0.7.10 that was available at the time of writing this report, there were no eligibility traces modelled in the STDP function as well as there was no decay of dopamine available. However, the main problem was that the appliance of STDP could not be controlled as desired because of a bug in the library.

## 7 Bibliography

- [1] E. R. Kandel, J. H. Schwartz, and T. M. Jessell, *Principles of Neural Science*, vol. 3. McGraw-Hill, 2000, p. 1414.
- [2] T. M. Mitchell, *Machine Learning*, vol. 4, no. 1. McGraw-Hill, 1997, pp. 417-433.
- [3] "Interview with Dharmendra S. Modha Principal Investigator from the DARPA program SyNAPSE." [Online]. Available: <http://www.scientificamerican.com/blog/post.cfm?id=computers-have-a-lot-to-learn-from-2009-03-10>. [Accessed: 2011].
- [4] "K Computer." [Online]. Available: <http://www.fujitsu.com/global/about/tech/k/>. [Accessed: 27-Jul-2011].
- [5] P. Chorley and A. K. Seth, "Closing the Sensory-Motor Loop on Dopamine Signalled Reinforcement Learning," *Sensors Peterborough NH*, vol. L, no. 9, pp. 280-290, 2008.
- [6] H. Hagaras, A. Pounds-Cornish, M. Colley, V. Callaghan, and G. Clarke, *Evolving spiking neural network controllers for autonomous robots*. IEEE, 2004, pp. 4620-4626 Vol.5.
- [7] D. Floreano, J.-C. Zufferey, and J.-D. Nicoud, "From wheels to wings with evolutionary spiking circuits.," *Artificial life*, vol. 11, no. 1-2, pp. 121-38, Jan. 2005.
- [8] A. Bouganis and M. Shanahan, "Training a Spiking Neural Network to Control a 4-DoF Robotic Arm based on Spike Timing-Dependent Plasticity," *Computational Intelligence*, pp. 18-23, 2010.
- [9] S. F. Cooke and T. V. P. Bliss, "Plasticity in the human central nervous system.," *Brain: A journal of neurology*, vol. 129, no. 7, pp. 1659-73, 2006.
- [10] T. V. P. Bliss and T. Lomo, "Long-lasting potentiation of synaptic transmission in the dentate area of anaesthetized rabbit following stimulation of the perforant path," *J Physiol*, vol. 232, pp. 351-356, 1973.
- [11] S. M. Dudek and M. F. Bear, "Homosynaptic long-term depression in area CA1 of hippocampus and effects of N-methyl-D-aspartate receptor blockade.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 89, no. 10, pp. 4363-4367, 1992.
- [12] H. Markram, J. Lübke, M. Frotscher, and B. Sakmann, "Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs.," *Science*, vol. 275, no. 5297, pp. 213-215, 1997.
- [13] D. O. Hebb, *The organisation of behaviour*. New York: Wiley, 1949, p. 355.
- [14] E. M. Izhikevich, "Solving the distal reward problem through linkage of STDP and dopamine signaling.," *Cerebral cortex (New York, N.Y. : 1991)*, vol. 17, no. 10, pp. 2443-52, Oct. 2007.
- [15] S. Song and L. F. Abbott, "Cortical development and remapping through spike timing-dependent plasticity.," *Neuron*, vol. 32, no. 2, pp. 339-50, 2001.

- [16] C. Savin, "Homeostatic plasticity - computational and clinical implications," Frankfurt Institute for Advanced Studies, 2010.
- [17] G. G. Turrigiano, "Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same.," *Trends in Neurosciences*, vol. 22, no. 5, pp. 221-227, 1999.
- [18] M. C. Van Rossum, G. Q. Bi, and G. G. Turrigiano, "Stable Hebbian learning from spike timing-dependent plasticity.," *Journal of Neuroscience*, vol. 20, no. 23, pp. 8812-8821, 2000.
- [19] W. Schultz, "Predictive reward signal of dopamine neurons," *Journal of Neurophysiology*, vol. 80, no. 1, pp. 1-27, 1998.
- [20] R. Legenstein, D. Pecevski, and W. Maass, "A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback.," *PLoS computational biology*, vol. 4, no. 10, p. e1000180, Oct. 2008.
- [21] F. G. Ashby, S. Helie, T. Maddox, D. R nger, and D. Worthy, "A Tutorial on Computational Cognitive Neuroscience : Modeling the Neurodynamics of Cognition," 2011.
- [22] C. Koch and I. Segev, *Methods in Neuronal Modeling: From Ions to Networks*, vol. 2. The MIT Press, 1998, pp. 1-26.
- [23] E. M. Izhikevich and G. M. Edelman, "Large-scale model of mammalian thalamocortical systems.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 9, pp. 3593-3598, 2008.
- [24] 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, no. 4, pp. 500-544, 1952.
- [25] E. M. Izhikevich, "Which model to use for cortical spiking neurons?," *IEEE Transactions on Neural Networks*, vol. 15, no. 5, pp. 1063-1070, 2004.
- [26] E. M. Izhikevich, "Simple model of spiking neurons," *Time*, vol. 14, no. 6, pp. 1569-1572, 2003.
- [27] M. D. Humphries and K. Gurney, "Solution methods for a new class of simple model neurons.," *Neural Computation*, vol. 19, no. 12, pp. 3216-3225, 2007.
- [28] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT Press, 2001, p. 476.
- [29] E. Fuchs, A. Ayali, E. Ben-Jacob, and S. Boccaletti, "The formation of synchronization cliques during the development of modular neural networks.," *Physical Biology*, vol. 6, no. 3, p. 036018, 2009.
- [30] J. Gautrais and S. Thorpe, "Rate coding versus temporal order coding: a theoretical approach.," *Bio Systems*, vol. 48, no. 1-3, pp. 57-65, 1998.
- [31] S. J. Thorpe, "Spike arrival times: A highly efficient coding scheme for neural networks," *Parallel processing in neural systems and computers*, pp. 91-94, 1990.

- [32] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neural population coding of movement direction," *Science*, vol. 233, pp. 1357-1460, 1986.
- [33] A. Pouget and P. E. Latham, "Population Codes," in *Handbook of Brain Theory*, 2002, pp. 1-10.
- [34] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 9, no. 5. MIT Press, 1998, pp. 1054-1054.
- [35] F. Wörgötter and B. Porr, "Temporal sequence learning, prediction, and control: a review of different models and their relation to biological mechanisms.," *Neural Computation*, vol. 17, no. 2, pp. 245-319, 2005.
- [36] Y. Niv, "Reinforcement learning in the brain," *Journal of Mathematical Psychology*, vol. 53, no. 3, pp. 139-154, Jun. 2009.
- [37] R. S. Sutton, "Learning to Predict by the Methods of Temporal Differences," *Machine Learning*, vol. 3, no. 1, pp. 9-44, Aug. 1988.
- [38] A. G. Barto, "Adaptive Critics and the Basal Ganglia," *Models of Information Processing in the Basal Ganglia*, no. 1994, pp. 215-232, 1995.
- [39] W. Schultz, "Multiple dopamine functions at different time courses.," *Annual review of neuroscience*, vol. 30, pp. 259-88, Jan. 2007.
- [40] D. C. Dennett, "Cognitive Wheels: The Frame Problem in Artificial Intelligence," in *The Robots Dilemma The Frame Problem in Artificial Intelligence*, Ablex, 1987.
- [41] V. Braitenberg, *Vehicles: Experiments in Synthetic Psychology*, vol. 19. MIT Press, 1984, p. 152.
- [42] X. WANG, Z. HOU, A. ZOU, M. TAN, and L. CHENG, "A behavior controller based on spiking neural networks for mobile robots," *Neurocomputing*, vol. 71, no. 4-6, pp. 655-666, Jan. 2008.
- [43] E. A. Di Paolo, "Evolving Spike-Timing Dependent Plasticity for Robot Control," *EPSRCBBSRC International Workshop Biologicallyinspired Robotics The Legacy of W Grey Walter WGW2002 HP Labs Bristol 14 16 August 2002*, 2002.
- [44] K. D. M. Sen Song, "Competitive Hebbian Learning through Spike-Timing-Dependent Synaptic Plasticity."
- [45] G. Q. Bi and M. M. Poo, "Synaptic modifications in cultured hippocampal neurons," *Journal of Neuroscience*, vol. 18, pp. 10464-10472, 1998.