**NTNU – Trondheim**
Norwegian University of
Science and Technology

# Dopamine modulated STDP and reinforcement learning in an embodied context

Lars Andersen
Tormund Sandve Haus

Master of Science in Computer Science
Submission date: June 2013
Supervisor: Keith Downing, IDI

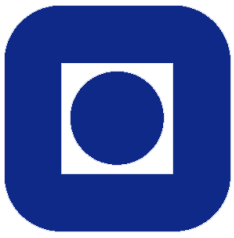
Norwegian University of Science and Technology
Department of Computer and Information Science

**Lars Andersen, Tormund S. Haus**

# Dopamine modulated STDP and reinforcement learning in an embodied context

Master thesis, spring 2013

Artificial Intelligence Group
Department of Computer and Information Science
Faculty of Information Technology, Mathematics and Electrical Engineering



**Abstract**

In recent years artificial neural networks have become increasing popular. New methods and ever increasing computational resources are turning second generation artificial neural networks into powerful tools.

Most of the work done with second generation artificial neuron networks do, however, at one point or another involve a phase of supervised learning. Supervised learning methods are inherently limited by the need for labeled training examples.

One way of solving this scaling problem is to rely on reinforcement learning, which is a form of unsupervised learning. The more biologically plausible third generation of artificial neural networks have recently been shown capable of tackling the distal reward problem that is at the core of reinforcement learning.

Using dopamine modulated spike-timing-dependent plasticity in a spiking neural network, we successfully demonstrate classical conditioning, instrumental conditioning, extinction and second order conditioning in an embodied context.

# Preface

This report serves as the master thesis for the authors. It was written in the spring of 2013 at the Department of Computer and Information Science at the Norwegian University of Science and Technology.

The work was supervised by professor Keith L. Downing.

Lars Andersen Tormund S. Haus

Trondheim, June 20, 2013

# Contents

# List of Figures

# List of Tables

# Acronyms

**ANN** Artificial Neural Network.

**ANNs** Artificial Neural Networks.

**BNNs** Biological Neural Networks.

**CS** Conditioned stimulus.

**SNc** Substantia nigra.

**SPNN** Spiking neural network.

**STDP** Spike-timing-dependent plasticity.

**TD** Temporal-difference.

**UR** Unconditioned response.

**US** Unconditioned stimulus.

**VTA** Ventral tegmental area.

# Chapter 1

# Introduction

This chapter gives a brief overview of the work done in our thesis. Section 1.1 will give a very short recap of the project background and motivation. Section 1.2 presents our goal and the resulting research questions. Then, in section 5.3, we outline our intended contribution to the field of artificial intelligence. In section 1.4 we briefly review how we intend to answer the research questions. Finally, in section 1.5, we present the structure of the rest of the thesis.

## 1.1 Background and motivation

It makes sense to speak of three generations of Artificial Neural Networks (ANNs)[12]. Networks of the second generation are presently the most popular, but the third generation of ANNs, based on spiking neurons, has some interesting properties[12] not found in second generation networks. They are more biologically plausible[23], and more computationally effective[12] for certain problems.

Spiking neurons come in many different flavors. The Hodgkin-Huxley model closely models biological neurons [7], at great computational expense[10]. The leaky-integrate-and-fire model aims at only modelling the spiking dynamic itself, but does so with a very low computational cost [10], and can easily be implemented in hardware[8]. Izhikevich's "simple neuron model"[9] strikes a good balance between being able to emulate a wide variety of neurons and the computational cost[10] of doing so.

In reinforcement learning an agent tries to maximize the notion of some

cumulative reward, by taking actions that transition the agent between states in the environment[21]. There are various forms of reinforcement learning. Operant, also called instrumental, conditioning occurs when an agent alters its behavior, making actions that trigger a reward occur more frequently. In classical conditioning a new stimulus comes to trigger an innate response–or a reflex–when the agent learns to associate the new stimulus with the subsequent occurrence of the familiar stimulus. For example, in Pavlov's famous experiment a new stimulus, the sound of a bell, predicted the dispensing of food in the dog's environment and caused the dogs to salivate–an innate response previously only associated with food presentation. In second order conditioning the level of indirection is increased further, by for example letting a light precede the sound of the bell. Pavlov's dogs would then start salivating once the light appeared.

Spike-timing-dependent plasticity (STDP) is a conceptually simple, but powerful mechanism, to selectively strengthen certain synapses and achieve Hebbian learning[20]. Synapses that have the firing pattern pre-then-post-synaptic are strengthened and neurons with the opposite firing pattern are weakened.

Dopamine appears to play a key role in reinforcement learning in humans, and other primates[17]. By linking STDP with dopamine it is possible to demonstrate reinforcement learning in an Artificial Neural Network (ANN) of spiking neurons, a Spiking neural network (SPNN)[11].

Using an ANN in a situated and embodied context is not always straight forward, but reinforcement learning was demonstrated in a virtual setting in [2] and with an iCub robot in [19].

Our aim is to do something similar to Chorley et al. in [2] and demonstrate reinforcement learning in a virtual robot. Our robot model and environment will be simpler, allowing us to make due with a much smaller SPNN. They show operant conditioning, and Soltottio et al. show classical conditioning in [19]. We aim to show second order conditioning as well, which to our knowledge has not been done before.

## 1.2 Goal and research questions

### 1.2.1 Goal

Investigate whether dopamine modulated STDP can be used to solve the distal reward problem, in a situated and embodied agent, by using reinforcement learning to change the behavior of a virtual robot.

### 1.2.2 Research questions

#### Research question 1

Can we demonstrate classical conditioning, in a situated and embodied agent, using dopamine modulated STDP?

#### Research question 2

Can we demonstrate instrumental conditioning, in a situated and embodied agent, using dopamine modulated STDP?

#### Research question 3

Can we demonstrate extinction of aquired behavior, in a situated and embodied agent, using dopamine modulated STDP?

#### Research question 4

Can we demonstrate second order conditioning, in a situated and embodied agent, using dopamine modulated STDP?

## 1.3 Contributions

Just like Chorley et al.[2] we are creating a virtual world, with an inhabiting robot, to solve the distal reward problem in a situated and embodied context. Our virtual robot will be simpler, in terms of its sensor array, and our network will be 40% the size of the network used by Chorley et al. We intend to calibrate our system on the work done by Izhikevich in [9]. Chorley

et al. demonstrate instrumental condition. In addition to instrumental conditioning, we plan to show classical and second order conditioning. To our knowledge, nobody has shown second order conditioning in a similar setting.

## 1.4 Research method

In order to answer our research questions we create a system with three components: a **virtual robot** controlled by an SPNN in a simulated **environment**.

The **environment** is created with the following constraints:

- The world is a *torus*, which means there are no walls; the world wraps around seamlessly.
- It is discrete, made up of square tiles.
- Each tile in the world is either
  - Empty
  - Green
  - Red
  - Blue

The **robot** has the following features:

- A light that can be turned on or off.
- A forward motor that can be turned on or off.
- If the motor is off, the robot will move randomly using the following rules:
  - Move forward with a probability of 0.5
  - Turn to the right with probability of 0.25
  - Turn to the left with probability of 0.25
- Ground sensors indicating whether or not the robot is on a red, green or blue tile.
- Distance sensors capable of detecting the colors red, green and blue in front of the robot.

The SPNN has the following characteristics:

- A total of 400 neurons, based on Izhikevich's spiking neuron model [9]
- 80% excitatory neurons
- 20% inhibitory neurons
- A network connectivity of 10%.

In addition to the above, the brain of the virtual robot has the following neuroanatomy:

- Neuron groups linked to the distance sensors for each color.
- Neuron groups linked to the ground sensors for each color.
- A Ventral tegmental area (VTA) region, with dopaminergic neurons, whose firings result in production of dopamine.
- A neuron group controlling the activation of the forward motor.
- A neuron group controlling the activation of the robot's light.
- Explicit connections between sensor groups and motor and light group.

By watching the activity of the virtual robot in the environment we can answer our research questions, and evaluate whether or not any learning has taken place. The 4 research questions result in 4 different experiments. Each experiment will be run 16 times, 8 times where the patches move after the robot enters it and 8 times when the patches are stationary.

## 1.5 Thesis structure

The remaining thesis has the following structure:

Chapter 2 presents the background information relevant to understanding our work, and place it in a larger context.

Chapter 3 presents the methods we are using to answer our research questions. This will include details about the SPNN we are constructing and the artificial environment and its virtual robot inhabitant.

Chapter 4 contains information about all our experiments and their results.

Finally, chapter 5 holds our evaluation and conclusion based on the experiments done. We will also touch on the contributions of this thesis and discuss our ideas for future work.

# Chapter 2

# Background theory and motivation

## 2.1 Background theory

In this section we will provide the background information necessary to understand our experiments and place them in a larger context.

### 2.1.1 Artificial Neural Networks

An ANN is a computational abstraction based on how the brain works. The neurons found in the human brain, and those of other animals, are incredibly complex in their own right. ANNs make no attempt at simulating the processes that afford neuronal activation and adaptation. Instead, by less complex means, the functional workings of a neuron is simulated. At present, the standard abstraction used in ANNs is as follows:

1. nodes
2. weighted connections between nodes
3. an integration function. The most common integration function sums the weighted output of all upstream neighbors
4. an activation function, which transforms the integrated input to an activation level for the neuron. This activation level will be serve as the output for the node, and as input to its downstream neighbors.

In terms of the biological neural networks the conceptual mapping is as follows:

- nodes → neurons
- connections between nodes → synapses
- integration function → total depolarization of cell soma is the sum of depolarization as caused by each synapse.
- Activation function → average firing rate of the neuron.

Figure 2.1 shows a typical ANN with the neurons placed in three layers. This type of network architecture is called *feed-forward*, because of the clear direction in signal propagation: from the input layer, through the hidden layer, and to the output layer. Other network structures are possible, e.g. with *recurrent connections*, but the *feed-forward* networks are popular because they are easy to train.

The basic computational loops is as follows:

- The input to the network is applied to the designated input neurons.
- The integration and activation function is applied to find an output value for each neuron.
- The output from each neuron is propagated to its downstream neighbors, to serve as input . The output from the network itself is thus just the output from some neurons designated as output neurons.

### Three generations of artificial neural networks

In [12] ANNs are said to belong to one of three classes, based on innovations in their constituent computational units. The first generation of ANNs have computational units that are called perceptrons or threshold gates. Threshold gate is the more descriptive name, because this class of artificial neurons *fire* when their total input reaches, or surpasses, some threshold value. These neurons are only capable of digital output (they either fire or they do not), but they are also *universal* for computations with digital input and output. They are *universal* in the sense that any boolean function can be approximated by some multi-layer perceptron with a single hidden layer[1].

[1]Such a network would have a structure similar to the one depicted in figure 2.1

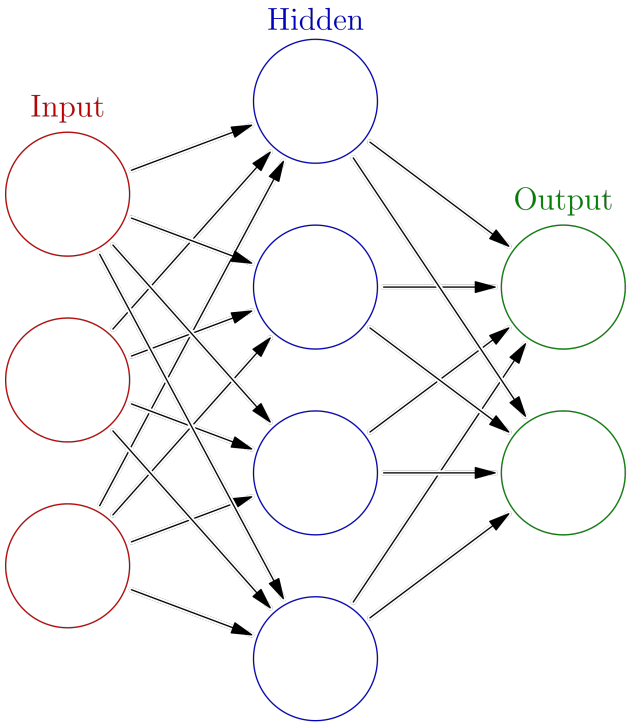



Figure 2.1: A fully connected feed-forward artificial neural network with an input, hidden and output layer.

The second generation ANNs are those equipped with an *activation function*. An *activation function* is a function of type $f\colon \mathbb{R} \to \mathbb{R}$, mapping the inputs of a neuron to a continuous output value. A common activation function choice is the *logistic function* $S(t) = \frac{1}{1+e^{-t}}$, sometimes called the *sigmoid function*[2].

The neural networks of this generation have the same capabilities as those of the first generation, but it has also been shown that they can approximate certain boolean functions using fewer computational units than networks of the first generation.

Second generation ANNs are also *universal* for analog computations: they can approximate, arbitrarily well, any continuous function with a compact domain and range, using only a single hidden layer. Also important is the

[2]The sigmoid functions are actually a family of functions, that are shaped like an "s". The *hyperbolictangent* and *arctangent* functions are also members of this family.

support for learning algorithms based on gradient descent[3] making it easier to perform supervised training in this generation of networks.

The scheme used to encode information in the second generation networks is called *rate coding* because the output of one the sigmoidal units can be interpreted as the average firing rate of a biological neuron. Experiments have shown, however, that the time scale of visual processing, in humans[22] and in macaque monkeys[16], is so short that any mechanisms relying on averaging is highly improbable. There is also increasing experimental evidence that many biological neural systems rely on the timing of individual action potentials to encode information.

Third generation ANNs are built up of *spiking neurons*. Spiking neurons are so named because they use *pulse coding*: they transmit and receive individual pulses. Neurons of the third generation are thus more biologically plausible, than those of the second generation, and are transmitting more information by also relying on the time dimension.

### 2.1.2 Spiking neurons

There are many models of spiking neurons, but they all aim to emulate the action potential seen in figure 2.2, with varying degrees of fidelity. The interior of all neurons have a slight negative charge, with respect to their surroundings. This is reflected in the negative value of the resting potential in figure 2.2. When the neuron receives excitatory stimuli the neuron is depolarized until it reaches a *threshold* value. When the threshold value is reached a chain reaction starts which causes the neuron to fire. After the neuron has fired it enters a short *refractory period*, the undershoot phase shown in figure 2.2, where the neuron is harder to excite than normal.

In [10] Izhikevich does a thorough review of various spiking neuron models. 11 models for spiking neurons are evaluated on their ability to reproduce the 20 most prominent features of biological neurons and the model's computational cost in doing so.

At one end of the spectrum we find the Hodgkin-Huxley model[7], derived from studies of giant squid neurons. This model is incredibly detailed, it consists of 4 equations, describing the membrane potential and the currents of $Na^+$ and $K^+$ ions, using ten parameters. It is one of the most important models of neuroscience. This level of detail, however, comes at a

[3]The famous backpropagation algorithm relies on this method.

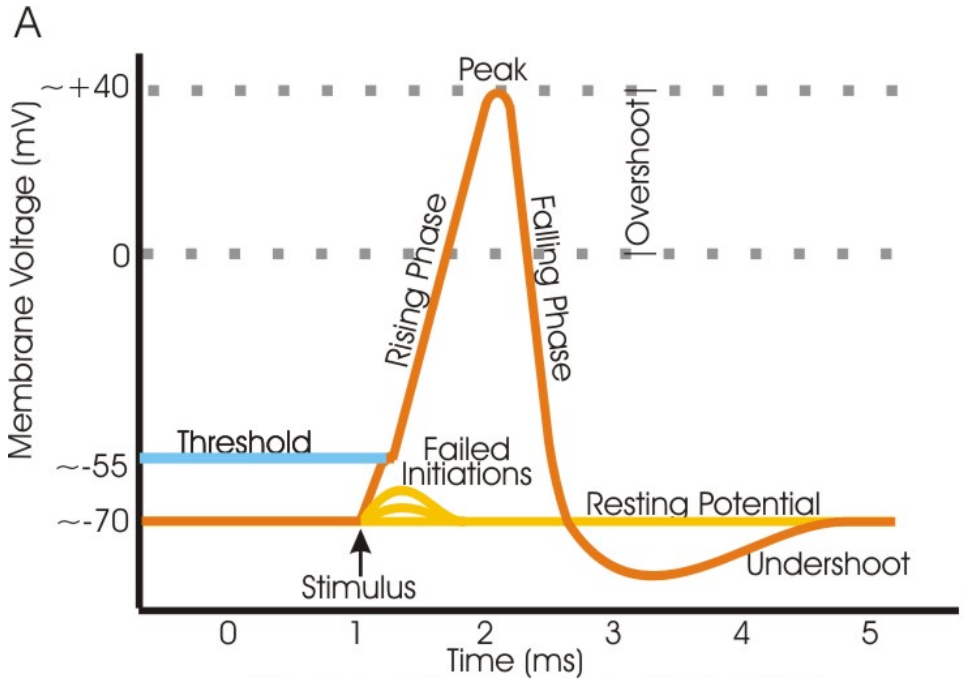



Figure 2.2: A schemetic overview of an action potential in a biological neuron.

cost: updating the state of a single neuron requires 1200 FLOPS.

At the other end of the spectrum we find one of the most commonly used models of spiking neurons: the leaky integrate-and-fire model. This model is very limited when it is evaluated on its ability to simulate biological neurons, but updating each neuron costs only 5 FLOPS, an attractive value compared to the Hodgkin-Huxley model.

The model we are going to use is presented by Izhikevich in [9] and strikes a good balance between simulation detail and computational costs. Izhikevich's model is able to reproduce all the 20 features evaluated in [10], can be described with only two equations and four parameters, and updating the state of a single neuron comes at the computational cost of 13 FLOPS.

### Izhikevich's spiking neuron model

Izhikevich's spiking neuron model can be described with the following equations:

$$\dot{v} = 0.04v^2 + 5v + 140 - u + I \tag{2.1}$$

$$\dot{u} = a(bv - u) \tag{2.2}$$

and the following equation for after-spike resetting:

$$\text{If v } \geq \text{ 30mV, then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \tag{2.3}$$

Where $a$, $b$, $c$ and $d$ are dimensionless parameters, and $\dot{v} = dv/dt$ where $t$ is time. The variable $v$ represents the membrane potential of the neuron and $u$ is a variable describing the recovery of the membrane potential with respect to the resting potential. $I$ represents the input to the neuron from other neurons.

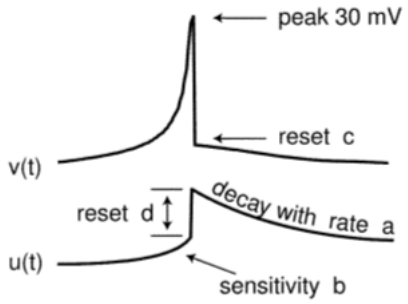



Figure 2.3: Effects of the various parameters on the shape of the pulse and recovery of membrane potential.

In figure 2.3 we can get a sense for what the various parameters does to the shape of the spike and its recovery.

- The parameter $a$ affects the time scale of the recovery variable $u$, lower values yields slower recovery.

- Larger values for $b$ gives a higher coupling between $v$ and $u$ making it possible to have subthreshold oscillations.
- $c$ describes the after-spike reset value.
- $d$ is the after-spike reset value for $u$.

### 2.1.3 Spike-timing-dependent synaptic plasticity

In order for an ANN to learn, a mechanism for altering the weights in the network is needed. Usually this is done through some kind of *Hebbian learning*. *Hebbian learning* is based on the following observation, made by Donald Hebb in 1949:

> When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.

Even though Hebb made his observation with respect to biological neural networks, Hebbian learning has served well as a guiding principle in ANNs as well. That said, it is clearly not enough to just say that "neurons that fire together wire together", we also need some way to create competition among synapses or the synapse strength will keep increasing for all neurons, because there is no mechanism in play to reduce the strength of a synapse.

STDP is an algorithm based on Hebbian learning, where the strengthening, or weakening, of a synapse is based on the correlations of spike timings between the pre- and postsynaptic neuron.

If we let $\Delta w_j$ denote the weight change of a synapse from a presynaptic neuron $j$ to a postsynaptic neuron $i$. This weight change will depend on the relative timing between the presynaptic spike arrivals and the postsynaptic spikes. Further, let us denote the presynaptic arrival times, at synapse $j$, as $t_j^f$ where $f = 1, 2, 3, ...$ counts the presynaptic spikes, and similarly $t^n$ with $n = 1, 2, 3, ...$ for the postsynaptic neuron. Then the total weight change induced by the spike train is[4]:

$$\Delta w_j = \sum_{f=1}^{N} \sum_{n=1}^{N} W(t_i^n - t_j^f) \tag{2.4}$$

where $W(t)$ denotes an STDP function. We are using the STDP function found in [20], which is a popular choice:

$$W(t) = A_+ \exp(-x/\tau_+) \qquad if \quad \Delta t < 0 \tag{2.5}$$
$$W(t) = -A_- \exp(x/\tau_-) \qquad if \quad \Delta t > 0 \tag{2.6}$$

Here $A_+$ is the magnitude of the weight change when we have a positive correlation, i.e. the presynaptic neuron fired prior to the postsynaptic neuron and $A_-$ is the magnitude when the correlation is negative. $\tau_+$ and $\tau_-$ are time constants affecting the length of the timing window for positive and negative weight changes, respectively.

Looking at figure 2.4 $\tau_+$ and $\tau_-$ affect the slope of the two curves and $A_+$ and $A_-$ affect the point of intersection with the Y-axis.

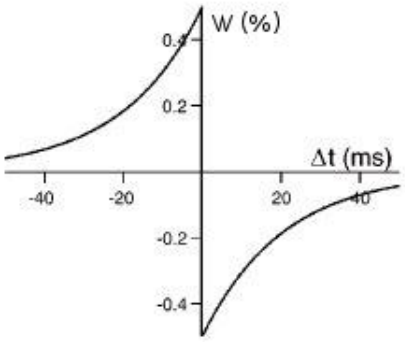



Figure 2.4: The STDP modification function.

The key takeaway from these equations is that, within the timing window for synaptic plasticity:

- If the *presynaptic* neuron fires before the postsynaptic neuron, the weight change will be positive.
- If the *postsynaptic* neuron fires before the presynaptic neuron, the weight change will be negative.

In order to get competition between the synapses we have to require that the integral over $W(t)$ is negative, i.e. that $A_-\tau_- > A_+\tau_+$. In terms of

figure 2.4 the absolute value of the area below the curve for $\Delta t > 0$ has to be greater than the area below the curve for $\Delta t < 0$. The effect of this competition among synapses is that if we only have random activity in our network all the weights will move toward 0.

### 2.1.4 Conditioning

There are various types of conditioning, but we will not review all of them. Instead, drawing on [14] we will give a very brief overview of the kinds of conditioning relevant for our work and the associated terminology.

#### Classical conditioning

Classical conditioning is perhaps better known by the name Pavlovian conditioning. In experiments involving classical conditioning the experimenter manipulates two events, known as the Conditioned stimulus (CS) and Unconditioned stimulus (US) where the US elicits some Unconditioned response (UR). The goal is to demonstrate learning in the subject, when CS comes to predict the US.

US is typically a biologically significant stimulus: in Pavlov's original experiments food was used. UR is the innate response to US: the food caused Pavlov's dogs to salivate. CS is some kind of stimulus which, initially, causes no response in the subjects.. Pavlov used the sound of a bell as the CS.

While training the subject the CS will be consistently presented prior to US. Eventually, this will lead to a shift of UR from US and to CS. In terms of Pavlov's experiments: the dogs started salivating at the sound of the bell, without any food in sight.

#### Operant conditioning

*Operant conditioning*, also called *instrumental conditioning*, is a form of learning where the subject's behavior is altered by the consequences of the subject's actions. The subject's behavior is altered through *reinforcement* and *punishment*:

- Reinforcement will cause the subject to exhibit the behavior with a greater frequency.

- Punishment will cause the subject to exhibit the behavior with a lower frequency.

In addition to *punishment* and *reinforcement*, an important term is *extinction*: when a behavior no longer carries any consequences it will occur less frequently.

The term operant conditioning was coined by B.F. Skinner, who also invented the eponymous Skinner box. This box had a lever which when pressed would deliver a food reward. When a rat was placed in the Skinner box it would move around randomly until it accidentally pressed the lever and a reward was dispensed. When the rat had the lever-pressing-behavior reinforced by the food reward, the frequency of lever presses increased dramatically. The rat had learned that it could get a pellet of food if it pressed the lever.

### 2.1.5 Reinforcement learning

Reinforcement learning is concerned with studying what *actions* an *agent* should take in some *environment* in order to maximize some cumulative *reward*. Drawing on [21] we can give the following, more formal, definition of reinforcement learning.

The agent and the environment interact at discrete time steps, $t = 0, 1, 2, 3....$ At each time step $t$, the agent is able to sense the state of the environment: $s_t \in S$ where $S$ is the set of all possible states. In state $s_t$ the agent selects an action $a_t \in A(s_t)$, from the set of actions afforded by this state, $A(s_t)$. In the next time step the agent will receive a reward based on the action taken. The reward, such as it is, is a numerical reward $r_{t+1} \in R$. To choose from the set of possible actions the agent uses a policy which is a probabilistic mapping from states to actions. The policy $\pi_t$ is the policy in play at the time step $t$ and $\pi_t(s, a)$ is the probability that $a_t = a$ if $s_t = s$.

#### Temporal-difference learning

Temporal-difference (TD) learning is a prediction method. It aims to combine some of the best ideas from Monte Carlo and dynamic programming. The main similarity with Monte Carlo methods is that learning is done through sampling. Like dynamic programming methods, TD is capable of bootstrapping: starting off at random and iteratively improving an estimate of the

target value. The simplest TD method, $TD(0)$, is:

$$V(s_t) \leftarrow V(s_t) + \alpha\big[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\big] \tag{2.7}$$

$V$ is a *value function*, and $V(s_t)$ represents the agent's best estimate for the value of being in state $s_t$. In equation 2.7 the estimate for the value of being in state $s_t$ is updated based on the current value, making bootstrapping possible. $\alpha$ can be thought of as a *learning rate*, determining how large of an impact the new information will have on the existing estimate. $r_{t+1}$ is the reward, for the action taken in the previous time step, and $V(s_t)$ is the estimate of that reward, so $r_{t+1} - V(s_t)$ is the difference between the *actual* and *expected* reward. $\gamma$ is a discount factor used to discount expected future value to present value: $\gamma V(s_{t+1})$ is thus the discounted expected value of the next state.

In actor-critic models of TD the policy is represented independently of the value function. The unit responsible for the policy is called the *actor* and the unit responsible for the value-function is called the *critic*. The role of the *critic* is naturally to provide feedback by criticizing the actions taken by the *actor*. This critique takes the form of a TD error, which is just the bracketed part from equation 2.7:

$$\text{TD error} = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{2.8}$$

Looking at this formula we can learn the following:

1. TD error $> 0 \rightarrow$ A larger than expected rewarded was received and the previous action should be encouraged.

2. TD error $< 0 \rightarrow$ A smaller than expected rewarded was received and the previous action should be avoided, to a certain degree, the next time the situation arises.

TD algorithms have received attention from neuroscientists because it has been discovered that the firing rate of dopaminergic neurons in the VTA and Substantia nigra (SNc) seem to play a role similar to the error function in the actor-critic model of TD. The experiments, done with monkeys in [17], showed that:

- when an unexpected reward was received the firing rate was high.

- when an expected reward was received the firing rate remained unchanged.
- when an expected reward was withheld the firing rate fell below normal.

The changes in firing rate of these dopaminergic neurons lead to a change in dopamine concentration. The neurotransmitter dopamine has been showed to be important for reward-driven learning and the above-mentioned experiment indicates that the information about a reward is propagated through the network as a change in the dopamine concentration.

TD learning can be extended to use eligibility traces. When an input is received it can be used to update the estimate for the previous state, but that new estimate can in turn be used to update the estimate for the state prior to that one and so forth. Eligibility traces can thus extend TD learning to update a collection of many earlier predictions for each step. Eligibility traces are usually implemented with exponential decay and the decay parameter is called $\lambda$. The various choices for lambda generates a family of TD algorithms, $TD(\lambda)$, $0 \leq \lambda \leq 1$, with $TD(0)$ as explained above, updating only the immediately preceding prediction, and $TD(1)$ updating all of them. In [13] the authors study dopamine cells in live rats to shed light on the TD algorithm. They find that a small value for $\alpha$ is required as well as the use of $TD(\lambda)$ instead of $TD(0)$ or $TD(1)$.

### Solving the distal reward problem in artificial neural networks

In experiments involving Pavlovian and instrumental conditioning the reward is typically delivered some time after the behavior that is being rewarded. This gives rise to the *distal reward problem*: how does the brain know which neurons and synapses are responsible for bringing about the reward? The firing patterns are obviously long gone, and the neurons and synapses involved might have seen significant activity in the interim period, between action and reward.

In [11] the distal reward problem is tackled by combining STDP and dopamine. STDP is used to provide the sign and magnitude of the changes to the active synapses. Recall that STDP will strengthen synapses where the previous activity was of form pre-then-post, and weaken synapses with the opposite firing pattern, post-then-pre.

A flood of dopamine is released in the network when it receives a reward. The effect of the dopamine is to increase the plasticity of the eligible synapses.

Since all eligible synapses have been active in the period between action and reward, they were either working against bringing about the reward, toward it, or the activity was unrelated to the reward. If they systematically worked against the behavior being rewarded STDP will weaken these synapses. If they were working toward the goal, the effect of STDP will be to strengthen them. Any uncorrelated activity will either lead to a synapse strengthening or weakening, but on average the net change in synapse strength will be small.

Izhikevich uses the following equation to simulate the concentration of extracellular dopamine in the network:

$$\dot{d} = -d/\tau_d + DA(t) \tag{2.9}$$

Where $\dot{d}$ is the derivative with respect to the time, $t$. $\tau_d$ is a time constant of dopamine uptake, reducing the total amount of available dopamine. $DA(t)$ models the production of dopamine by dopaminergic neurons in the midbrain, specifically in the areas VTA and SNc. It is important to note that $DA(t) > 0$ at all times due the spontaneous firing of the dopaminergic neurons.

The effect of this dopamine can be seen in the following equations. The first one simulates the activity of some enzyme important for neuroplasticity and the second one provides the rule for altering synapse strength:

$$\dot{c} = -c/\tau_c + STDP(\tau)\delta(t - t_{pre/post}) \tag{2.10}$$

$$\dot{s} = cd \tag{2.11}$$

In both equations the derivative is with respect to the time. $\delta(t)$ is the Dirac delta function which step-increases the variable $c$. The effect of the Dirac delta function is to only update the value of $c$ using STDP if one of the neurons, at either end of the synapse, fired in the current simulation time step.

Eq 2.10 is used to create an *eligibility trace*. The value of $c$ decays to $c = 0$ in an exponential manner, and $\tau_c$ is the variable manipulating the rate of decay. The effect of changing $\tau_c$ is thus to increase or decrease the sensitivity of plasticity to delayed rewards. A typical value for $\tau_c = 1$ s means that the synaptic plasticity will be negligible about 5 seconds after the STDP event.

Eq 2.11 is used to update the strength of synapses. $d$ is the dopamine concentration, leading to a situation where we have larger changes in synaptic

strengths when there is a lot of dopamine available in the system. In machine learning terms we can view this as increasing the *learning rate* of the system, upon encountering situations that are deemed especially important. We can say these situations are *important* because they give rise to a reward, which in turn created the spike in dopamine.

**Putting it all together**

In terms of doing reinforcement learning in a simulated robot, these orthogonal ideas come together in the following manner:

1. An ANN of spiking neurons forms the computational substrate, doing all the raw calculations.
2. STDP create a notion of *causality*, where neurons either contribute to or inhibit the firing of other neurons.
3. A reward triggers a flood of dopamine in the network and with eligibility traces, created by STDP, the system can correlate neuronal activity with reward-inducing behavior.
4. Rewarded behavior, learned through reinforcement learning, is encoded as changes in synaptic weights.

## 2.2 Motivation

Prior to the great AI winter the major focus of the field was symbolic AI. The leading approach, working toward creating a general AI, was to create systems with an inference engine and then input the necessary facts. The core idea was that if the system had enough key facts, it would be able to deduce the rest. The problem with this approach was that it didn't scale very well. The system needed a ton of facts, which had to be input by hand, and the problem of knowledge representation is also non-trivial.

This lead to a rising interest in sub-symbolic AI, which did not depend on an initial knowledge transfer, from human to machine. System based on ANNs belong to this category. These types of systems excelled at tasks which were previously very hard, like face or speech recognition. Instead of doing image pre-processing and relying on a system with a great amount of domain knowledge, to do rule-based comparison using the features extracted

from the image, an ANN could be automatically trained to do the same job, given only enough computational resources and a training set of sufficient size.

An important characteristic of the field of sub-symbolic AI is that its most successful methods rely on *supervised learning*. Second generation ANNs are so useful precisely because they are easy to train, using supervised training algorithms, like the gradient-descent based backpropagation.

Supervised learning relies on labeled data. E.g. if the tasks is classification then the system has to be presented with examples of the various classes along with the correct classification in order to learn. Creating such labeled data sets to train on can be difficult, and one often have to rely on humans to label the data with which the machine will train. The situation is quite analogue to the one encountered with symbolic AI, but instead of a direct knowledge transfer from human to machine, it is now indirect through the labeling of the data sets. This has lead to an increasing interest in combining labeled and unlabeled data sets[6], as the former is costly but the latter both cheap and abundant.

Reinforcement learning is an alternative to supervised learning, where the learning agent has to explore the environment and learning happens as the agent tries to maximize some cumulative reward. In [11] Izhikevich demonstrates one way to do reinforcement learning in a spiking neuron network. He does this by combining STDP and dopamine signalling. Izhikevich argues that the precise firing patterns, which are only possible in spiking neuron networks, are essential. Building on this work, Soltoggio et al. show that the key factor, making learning possible, is *rarely correlating* neural activity. Izhikevich creates rare correlations using STDP, but Soltoggio et al. are able to reproduce the results, in a second generation neural network, using a mechanism they call *rarely correlating Hebbian plasticity* (RCHP). With RCHP the rare events are created by comparing the activation level of the neurons with a threshold value. By adjusting this threshold value, they show that learning is impossible unless the correlating activity is rare enough. This key insight, along with the experiments done on the effect of varying the various STDP parameters in [1], should be quite helpful in tweaking the parameters of our model.

In [15] the results produced by Izhikevich in [11] are reproduced and dopamine modulated conditioning is demonstrated. However, in their experiments all firing patterns are reinforced during training, and the mechanism suggested in Izhikevich–which would leave only the firing patterns respon-

sible for bringing about the reward reinforced–is claimed to be ineffectual. They conclude that the random post-then-pre firings, responsible for weakening the synapses unrelated to bringing about the reward, happens too rarely to prevent the average weights from increasing. Their conclusion is right, in the sense that the rarity of these events is very important, but the reason they are seeing the average weight increase is likely to be because of their choice of parameters. Recall that in order to achieve competition between the synapses of a neuron, using STDP, the following condition has to be true: $A_-\tau_- > A_+\tau_+$. In [15] this is not the case, they use $A_- = A_+$ and $\tau_- = \tau_+$.

In [11] Izhikevich has 20% inhibitory neurons in his model. While conducting our literature review we noticed that this seems to be a widely used practice. In [18] evolutionary algorithms is used to show that ANNs which include inhibitory neurons dominate those without, for reinforcement learning tasks. Because the results in [18] are especially relevant, as they pertain to reinforcement learning, and it being such a widespread practice to include 20% inhibitory neurons, we do not plan to differentiate ourselves by changing this parameter of the model.

Knowing that the mechanisms were in place for doing reinforcement learning in a spiking neuronal network Chorley et al. investigate, in [2], if they can show conditioning in an embodied context, using a robot in a virtual environment. Their work is based on what Izhikevich did in [11] and they successfully demonstrate reinforcement learning, specifically instrumental conditioning. They had to predispose their agent to perform exploratory behavior, and they also had to use gated stimuli. Gated stimuli in this context means that the input from the sensors are sent synchronously to the network, giving rise to "sensing frames", as opposed to independent continuous streams of sensor readings.

Soltoggio et al. follow up on the work done in [18] and show that their work, using rare correlating Hebbian plasticity also works well in an embodied context. A neural network using the RCHP rule is used to train an iCub robot. They successfully show both classical and operant conditioning, as well as extinction.

We aim to follow in the footsteps of Chorley et al., and investigate the distal reward problem in an embodied context. We intend to use a less complex sensor array than they use, but keep the gated sensor input which they found to be important. They hardwired their robot to perform exploratory behavior by explicitly linking the sensors on the left side of the array to the to the motor neurons controlling the right wheel, and similarly for the right

side. Our approach will be different and simpler. Our robot behavior is inspired by organisms like the C. elegans, which moves randomly around for the most part but is capable of moving straight ahead in response to certain chemicals in the environment. Our robot has no wheels and lives in a torus shaped grid world, unlike the continuous world of Chorley et al.

This is not to say that we find any faults with the approach taken in Chorley et al., but rather we want to avoid any incidental complexity. Incidental in the sense that our main goal is to show that we can use spiking neural networks, with dopamine modulated STDP, to solve the distal reward problem. By for example having walls, our agent would have to learn obstacle avoidance, which is a distraction in terms of the thesis goal, and worse, such ancillary tasks might lead to:

- a need for a more complex network, with a higher capacity for learning. A larger network would be–perhaps prohibitively so–more computationally demanding.
- higher variance, making it difficult to reproduce our results, when the agent sometimes fails to learn the ancillary tasks, like obstacle avoidance.
- higher variance when the randomly created neural network has a topology such that learning the ancillary tasks are impossible.

# Chapter 3

# Methodology

This chapter contains an overview of the system we created to answers the research questions from section 1.2. In particular we will presents the details of the SPNN that affords learning in the virtual robot, about the environment the robot interacts with and about the virtual robot itself.

## 3.1 The SPNN

The SPNN is very similar to the one detailed in [11]: it uses Izhikevich's spiking neuron model, as well has his mechanisms for dopamine modulated STDP. A time step in the simulation is 1 ms, and sensing, using sensory gating, happens every 100 ms. Like Chorley et al. in [2] we found that sensory gating is absolutely necessary. Without sensory gating the sensory neurons are frequently being tasked with firing every time step, but the neuron model itself does not support firing frequencies anywhere near 1000 Hz (due to the refractory period). Firing rates this high are also not biologically plausible, indicating that we might be better off seeking alternatives rather than forcing the issue, e.g. by changing the spiking neuron model to one allowing a higher firing rate. Evidence for sensory gating can be found in [3], and in [5] where the experiments were carried out on human subjects.

The ANN needs to have some background activity, or it will be quiescent during the period when the network is not receiving sensory input. Another reason to have some background activity is to avoid a situation where all activity is essentially correlated because it is entirely driven by the sensory input. Izhikevich refers to this driver of background activity as "random

thalamic input" [11]. The thalamus is responsible for relaying sensor and motor signals to the cortex, but it is just as well to think of this as random input from other brain regions. We know from [18] that this background activity should result in a spontaneous firing frequency of around 1 Hz, in order to be able to solve the distal reward problem.

To get to 1 Hz we go through every excitatory neuron in the network and inject them with a small, random DC current. This will cause some neurons to fire, some of the time. The size of this DC current has to be found through trial and error. Thankfully Izhikevich published code related to his work in [9] making it possible for us to use the same values as Izhikevich.

Rewards are delivered by injecting a DC current to a group of neurons designated as dopaminergic, in effect simulating the VTA. In total there are 20 dopaminergic neurons.

The conduction delay between all the neurons is set to 1 ms. The model is made easier by this simplification, and we believe the task is solvable without the added complexity of varying conduction delays.

### 3.1.1 Neuroanatomy

The ANN consists of 400 neurons, with 80% being excitatory neurons and 20% inhibitory. Only synapses between excitatory neurons are plastic, i.e. amenable to change. The role of the inhibitory neurons is to modulate activity in the network, specifically to keep the spontaneous firing rate low enough for learning to occur. Thus, by not making these weights amenable to change, we increase the likelihood that the spontaneous firing rate remains around 1 Hz.

Figure 3.1 shows how the neurons are assigned to various functional groups. Unlike Chorley et al. in [2] we have quite a few unassigned neurons, even though our network is smaller than theirs.

Each sensor is represented with a neuron group consisting of 20 neurons. The robot detects 3 colors, with distance and ground sensors, resulting in $20 * 2 * 3 = 120$ sensory neurons.

The robot also has a group of 20 motor neurons, which are connected to an *integrator neuron*[10]. The role of the integrator neuron is to integrate the activity of the 20 motor neurons and fire if a sufficient number of them fire within some time span. This neuron helps bridge the gap between the two time scales in the simulation: the robot moves and senses every 100 ms, but the neural network state is updated every 1 ms. We do this by integrating


Figure 3.1: Assignment of neurons to various functional areas. Neuron numbers along the left edge. Each neuron group, except the last two, contain 20 neurons.

the output from the motor neuron group in the period between moves. If the integrator neuron fires in this period, the robot moves forward in the following time step.

Similarly, an integrator neuron is used to integrate the activity of the 20 neurons controlling the robot's light. If this integrator neuron fires, due to activity in the light neuron group, during the 100 ms between the update of the robot's state, then the robot will turn on its light in the following time step.

Every neuron is connected to any other neuron with a probability of 10% The one exception is that connections between two inhibitory neurons are disallowed. By wiring up the network in this manner a certain degree of randomness is introduced, and we risk running experiments where critical connections are missing. To avoid this we manually add a few extra synapses to connect the following "brain regions".

- distance sensor neurons $\rightarrow$ motor neurons
- distance sensor neurons $\rightarrow$ light neurons
- ground sensor neurons $\rightarrow$ light neurons

For each neuron in the groups that are to be explicitly connected, we randomly connect it to 4 neurons in the other group. This leads to a total of $20*4 = 80$ additional synapses per group pairing, and overall $80*(3+3+3) = 720$ additional synapses in the network.

With a 10% connection probability and a few explicitly created connections the total number of synapses has an expected value of $16000 + 720 = 16720$.

Figure 3.2 shows a schematic view of the robot's neuroanatomy. Only the explicitly created connections are shown.

## 3.2 The Environment

The geometry of the arena in which the robot exists is resembles the surface of a torus. As can be seen in figure 3.3 the surface of a torus is entirely continuous, and without any edges. In regards to the robot this means there are no walls to collide with. When the robot exits on any side side of the arena it immediately appears on the opposite side.

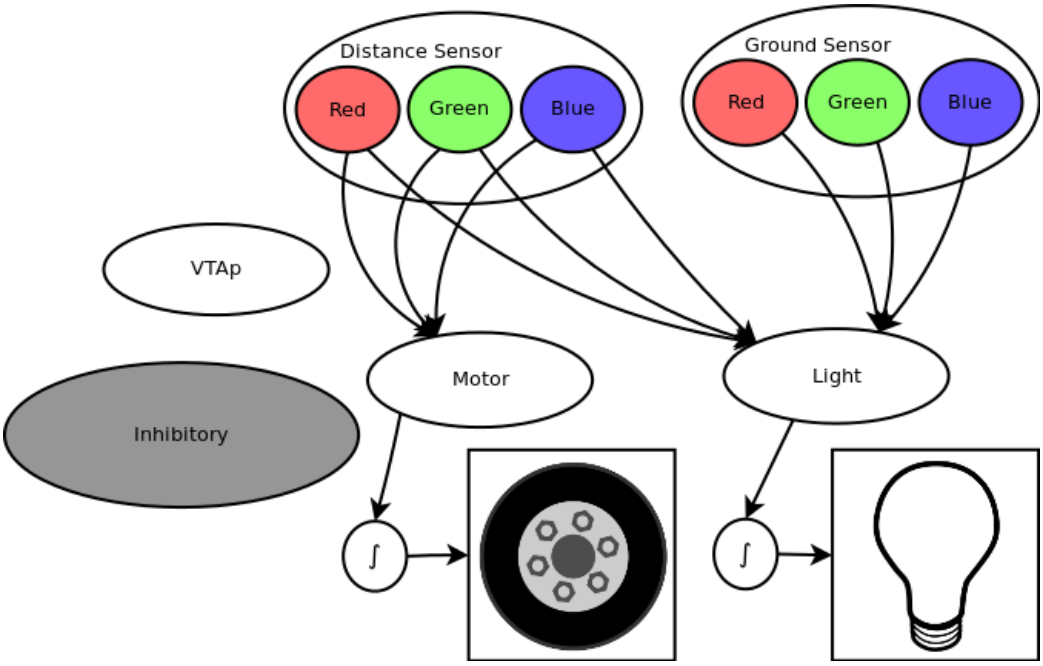



Figure 3.2: Neuroanatomy of the virtual robot, showing the various "brain regions". The $\int$ label marks the integrator neurons controlling forward movement and the robot light.

The surface of the world is divided into discrete tiles, making up a grid. Each tile in the grid is either:

- Empty
- Red
- Blue
- Green

The color of the tile hold no meaning other than what is assigned in our experiments. Except for the colors, they are identical in every regard.

The colored tiles are placed in patches, making up a square of four tiles. The resolution of the robot sensors is very poor. By using patches, instead of individual tiles, the robot can more easily seek out colored tiles by relying on its sensors.

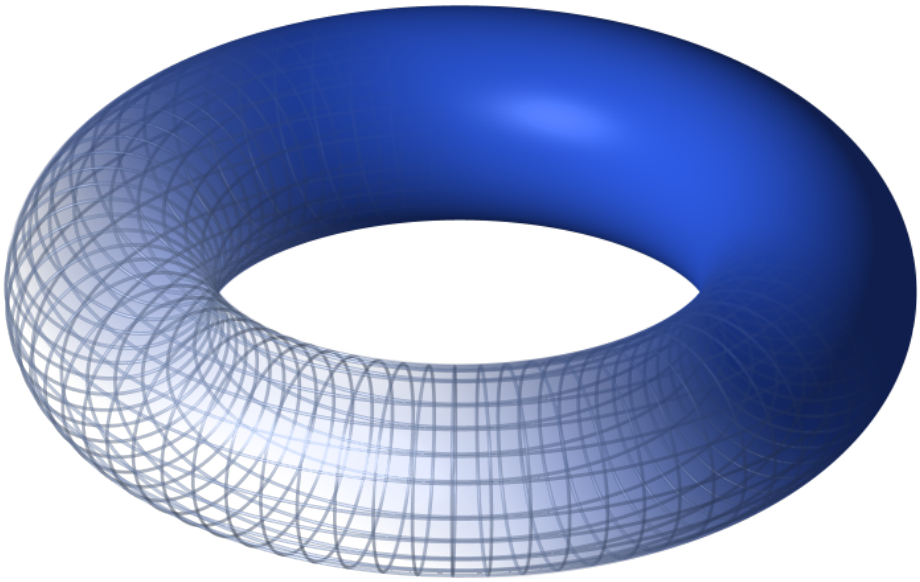


Figure 3.3: A picture of a torus. The geometry of our robot environment has the same properties as the surface of a torus.

When the robot enters a patch of colored tiles, that patch will move to some random location in the environment.

The robot's position in the environment is updated every 100 ms.

## 3.3 The Virtual Robot

Figure 3.4 shows the robot, the robot's cone of vision, and patches of red and green tiles in the environment. The black circle indicating the robot will turn yellow when the robot's light is turned on.

The world can contain tiles of three colors and the robot has ground and distance sensors capable of detecting each one. The ground sensors are active when the robot is standing on a tile with the matching color.

The distance sensors map to a single scalar indicating the amount of a given color within the robot's visual field. The mapping is done using the matrix in 3.5, which creates a cone of vision in the direction the robot is facing and where the contribution of each individual colored tile is inversely

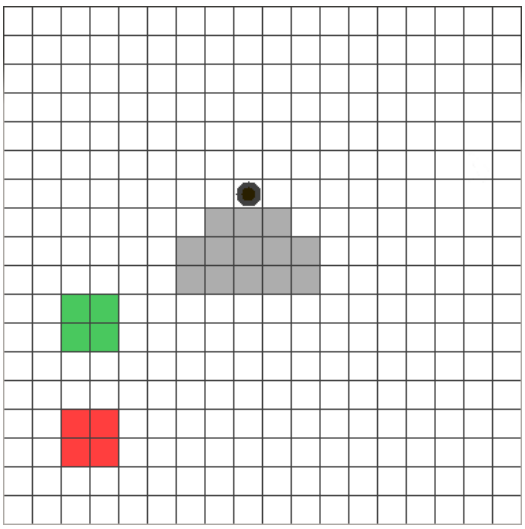


Figure 3.4: The robot in the environment. The grey colored area represents the robot's cone of vision.

proportional with its distance from the robot.

$$\begin{pmatrix} 0 & 0.1 & 0.1 & 0.2 & 0.1 & 0.1 & 0.0 \\ 0 & 0.1 & 0.2 & 0.4 & 0.2 & 0.1 & 0.0 \\ 0 & 0.0 & 0.5 & 1.0 & 0.5 & 0.0 & 0.0 \\ 0 & 0.0 & 0.0 & \text{robot} & 0.0 & 0.0 & 0.0 \\ 0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$$

Figure 3.5: Matrix representation of the robot's field of vision. The matrix element with the text "robot" indicates the position of the robot in the grid. The robot is looking upward in this example. The matrix is rotated when the robot turns.

The input to the network from the sensor is thus as follows:

$$I_c = \sum c_{ij} V_{ij} \tag{3.1}$$

Where $I_c$ is the sensor input for color $c$, $V_{ij}$ is a matrix element of 3.5 and $c_{ij}$ is 1 if the color is present at the matching tile in the world and 0

otherwise. If $I_C > 0.3$ the neurons in the sensor group will fire.

Without any external stimulus the robot follows the following non-deterministic movement rules:

- move forward with $P = 0.5$
- turn to the left with $P = 0.25$
- turn to the right with $P = 0.25$

The SPNN powering the robot can cause it to disregard the above-mentioned movement rules and instead move forward for one time step. The SPNN guide movement in experiments 2 and 3 whereas in experiments 1 and 4 the robot simply moves forward with $P = 1$ when the rewarded color is visible.

# Chapter 4

# Experiments and results

This chapter begins with our experimental plan, detailing how we plan to answer our research questions. The following experimental setup section aims to contain sufficient information to reproduce our experiments, including parameter values for the various methods used. The chapter concludes with the results from our experiments.

## 4.1 Experimental Plan

### 4.1.1 Runs

We plan to run all experiments 16 times. The experiments are divided into two sub-experiments: in 8 runs the patches are moved, to a random location, once the robot has entered them, and in 8 runs the patches remain stationary. Each of the sub-experiments we will run four times with a reward on red patches, and then four times with reward on green patches, to eliminate any potential source of bias. We will run each simulation until we reach one hour of simulated time.

### 4.1.2 Experiment 1

Experiment 1 aims to answer research question 1, which is about classical conditioning.

In Pavlov's experiments the UR was salivation upon the presentation of dog food, the US. Our equivalent UR is when the robot turns on the light, and our US is standing in a tile of the rewarded color.

In Pavlov's experiments the UR was shifted, so the dogs started salivating when they heard the bell ring, the CS. In our experiment we want to shift the UR so the robot turns on its light when it sees the rewarded color with its distance sensor, the CS. Just like the sound of the bell could be used to predict food in the dog's environment, the distance sensor can be used to predict the robot's subsequently entering a rewarded tile. In the dog's environment the bell predicted the food with certainty, but in the robot environment the prediction is less than perfect.

#### Goal

The goal in this experiment is to demonstrate classical conditioning. This means shifting an UR so it is triggered by a CS.

#### Evaluation

To demonstrate classical conditioning we have to observe the following:

> The robot's light (UR) has to be turned on when the rewarded patches are visible (CS), after some learning period.

In terms of the underlying SPNN we should observe that the weights going from the distance sensor group, for the rewarded color, and to the neuron group controlling the light is strengthened during the experiment.

In this experiment we do not make use of the motor neuron group and the SPNN to guide movement. Instead the robot simply moves forward, in the following time step, if the rewarded color is visible.

### 4.1.3 Experiment 2

Experiment 2 aims to answer research question 2, which is about instrumental conditioning.

#### Goal

The goal for this experiment is to demonstrate instrumental conditioning. Recall from chapter 2 that instrumental conditioning occurs when the subject alters its behavior in the course of the experiment, to receive larger or more frequent rewards.

**Evaluation**

To demonstrate instrumental conditioning in the robot we have to observe the following:

> The robot has to enter patches of the rewarded color more frequently than the unrewarded color, after some learning period.

In terms of the SPNN we should observe that the weights going from the distance sensor, detecting the rewarded color, and to the motor neurons have increased in strength. This change would cause the robot to move straight ahead when spotting the rewarded color, and this is the only mechanism by which the robot can increase the reward frequency.

### 4.1.4 Experiment 3

Experiment 3 aims to answer research question 3, which is about extinction of learned behavior.

In extinction experiments the subject has to learn behavior $B_1$ and then later learn $B_2$, while forgetting $B_1$.

**Goal**

The goal for this experiment is to demonstrate extinction of learned behavior.

**Evaluation**

Let the rewarded patches have color $C_1$ in the first half of the experiment and then $C_2$ in the second half. Then, to demonstrate extinction we have to observe the following:

1. The robot learns to seek out patches of color $C_1$, after some learning period.
2. The rewards are switched from $C_1$ to $C_2$.
3. The robot learns to seek out patches of color $C_2$, after some learning period.

In terms of the underlying SPNN we should see the following:

1. The weights going from the distance sensors detecting $C_1$ to the motor neurons **increase** in strength.

2. (a) The weights going from the distance sensors detecting $C_2$ to the motor neurons **increase** in strength.

   (b) The weights going from the distance sensors detecting $C_1$ to the motor neurons **decrease** in strength.

### 4.1.5 Experiment 4

Experiment 4 aims to answer research question 4, which is about second order conditioning.

Recall from chapter 2 that second order conditioning is similar to classical conditioning but we have two CS, and CS2 predicts the consequent occurrence of CS1.

In experiment 1 having the rewarded color, $C_1$, in the visual field comes to predict entering a tile of this color. To add another level of indirection we surround the rewarded color, $C_1$ with tiles of another color $C_2$. That way, seeing $C_2$ comes to predict seeing $C_1$ which in turns predicts entering a cell of color $C_1$ and receiving a reward.

#### Goal

The goal is to demonstrate second order conditioning.

#### Evaluation

To demonstrate second order conditioning we have show the following:

> The robot's lights turn on (UR) when tiles of color $C_2$ are visible (CS2), after some learning period.

In terms of the underlying SPNN we should see that the weights going from the distance sensors detecting $C_2$ and to the neuron group controlling the light are strengthened.

In this experiment we do not make use of the motor neuron group and the SPNN to guide movement. Instead the robot simply moves forward, in the following time step, if the rewarded color is visible.

## 4.2 Experimental Setup

This section details the setup used in our experiments. It aims to provide enough information to reproduce our experiments.

### 4.2.1 The environment

In all our experiments the environment consists of 18x18 tiles. Given the topology of the environment the number of tiles is not very important. What is important is the density of colored patches within the environment. The density has to be such that the frequency of rewards is high enough that depression, due to random activity, does not undo whatever we are trying to learn. The density cannot be too high, however, or the dopamine levels, due to frequent rewards, will increase without bound.

We have found a size of 18x18, using patches of size 2x2 to be a good compromise. When the robot enters a patch, the patch will move to some random location on the board. This randomization increases the likelihood that we learn general behavior.

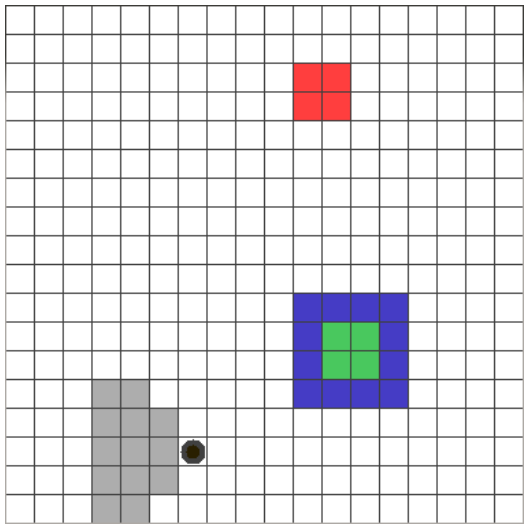


Figure 4.1: Screenshot showing the robot in the environment for the second order conditioning experiment. This is the only experiment where we make use of blue tiles, which are used to surround tiles of the rewarded color.

In figure 4.1 we can see the environment used for the second order condi-

tioning experiment. In this experiment the rewarded color is surrounded by blue tiles.

### 4.2.2 Model parameters

The model parameters are shared between all experiments.

### 4.2.3 Izhikevich's neuron model

Parameters related to Izhikevich "simple neuron model" [9] and equations 2.1, 2.2 and 2.3:

Excitatory neurons:
$a = 0.02$
$b = 0.2$
$c = -65$
$d = 8$

Inhibitory neurons:
$a = 0.1$
$b = 0.2$
$c = -65$
$d = 2$

Integrator neurons:
$a = 0.02$
$b = -0.1$
$c = -55$
$d = 6$

#### STDP

Parameters used for STDP in equation 2.5.
$A_+ = 0.1$
$A_- = -0.3$
$\tau_+ = \tau_- = 20\text{ms}$

#### Dopamine modulation

The parameters related to equations 2.9 and 2.10 are taken from [11]:
$\tau_d = 0.2$s
$DA(t) = 0.002$
$\tau_c = 1s$

## 4.3 Experimental Results

This section contains the results obtained while running our experiments. In addition we have included some of the experiments we ran to convince ourselves that the system worked correctly.

### 4.3.1 Calibrating the system

To convince ourselves that the system was implemented correctly we reproduced some of the experiments Izhikevich did in [11].

#### Reinforcing a synapse

The network consist of 1000 neurons and 100 000 synapses. We randomly choose a synapse that connects two neurons and set the synapse strength to 0 mV. The random activity in the network causes every neuron to fire with a frequency of around 1 Hz. Every time, across the chosen synapse, when the pre-synaptic neuron fires within 10 ms of the post-synaptic neuron we trigger a reward in the system, with a random delay between 1 and 3 s.

Figure 4.2 shows a plot of how the synapse strength (in red) changes throughout the experiment. The green line is the average synapse strength in the network. The strength of the synapse increases gradually until it reaches the maximum allowable value of 4 mV, where it remains. After an hour of simulated time the difference between the average weight of synapses in the network and the rewarded synapse is significant. The changes in synapse strength is quite steep. This might seem surprising at first, but remember that the resolution of the simulation is 1 ms.

Figure 4.3 shows how the dopamine in the system varies over time. This plot is not very interesting, but it shows that the spiky nature of dopamine release, and shows that the dopamine does not accumulate in the system. The plot looks as expected.

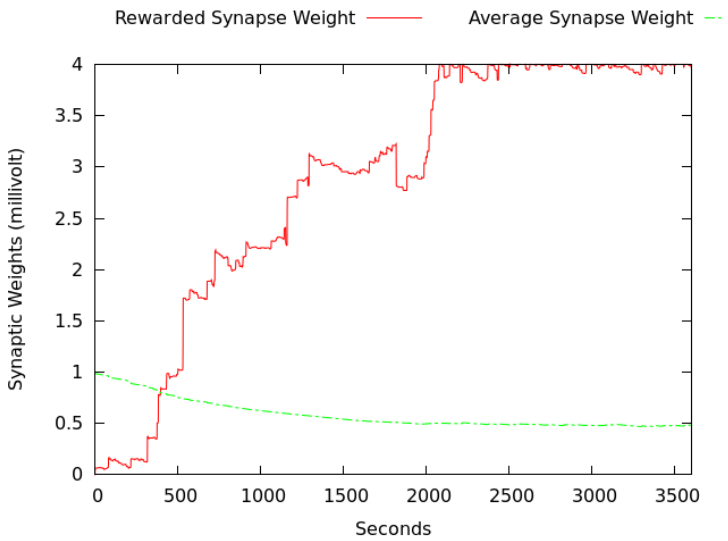



Figure 4.2: Instrumental conditioning of a single synapse. Whenever there is activity across the synapse a reward is triggered in the system. The strengthened synapse is in red, the average synapse strength is in green.

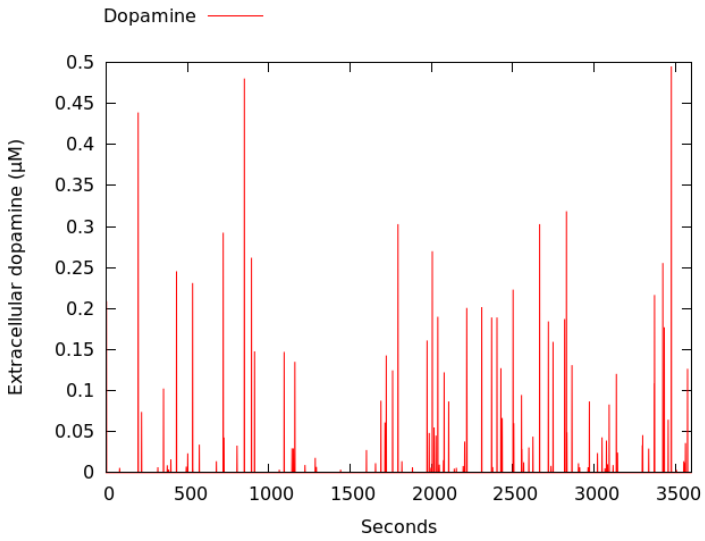



Figure 4.3: Dopamine over time in the system.

### Classical conditioning

The second experiment is also in a network of 1000 neurons and 100 000 synapses. We randomly choose 100 sets, $S_0, S_1, ..., S_{99}$ of 50 neurons each to represent some kind of stimuli. The sets are at least partially overlapping, and no effort is made toward manipulating this fact. A single neuron can potentially be a part of all 100 sets.

To give the system the stimulus represented by $S_i$ we inject a 1 ms DC-current into all 50 neurons of $S_i$, this will cause the nearly coincidental firing of all neurons in $S_i$.

Every 100-300 ms we give the system a stimulus by activating at random one of the groups $S_0 - S_{99}$, resulting in an average of 5 stimulus per second. Every time the system receives the stimulus represented by $S_0$ a reward is triggered in the system, with a random delay of up to 1 s.

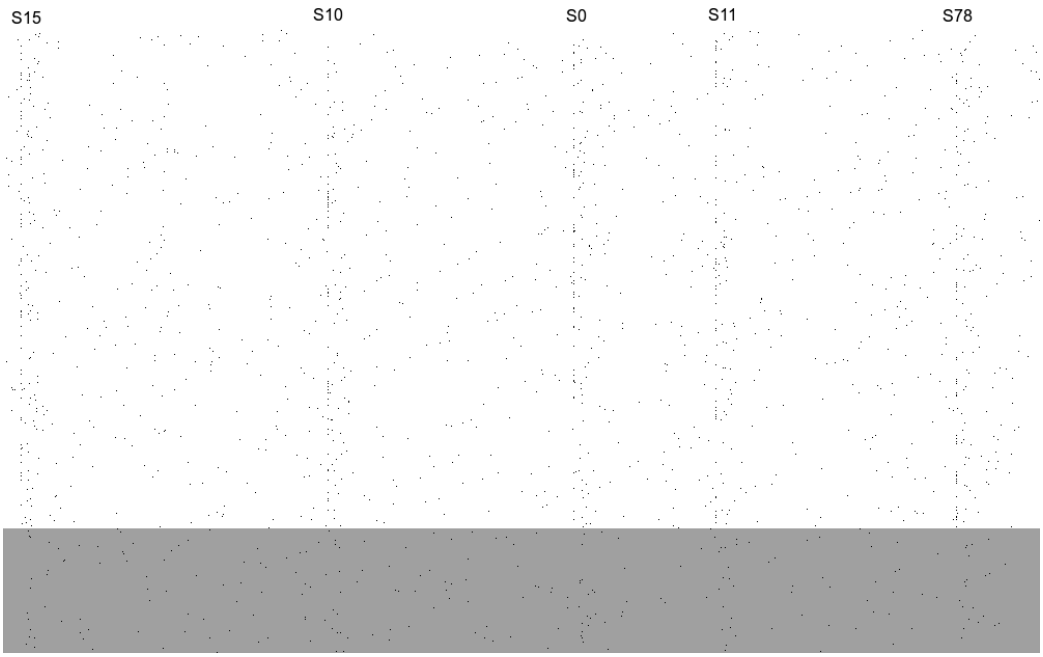



Figure 4.4: The response of the network to various input stimuli. The y-axes is neuron number and the x-axis is time. A dot indicates that a neuron has fired. Notice that the effect of the stimulus $S_0$ is nothing special. The shaded area indicates that the neurons are inhibitory.

In figure 4.4 we can see the effect the various stimuli has on the network.

Clearly, prior to training, the effect of $S_0$ is very similar to any other stimuli.

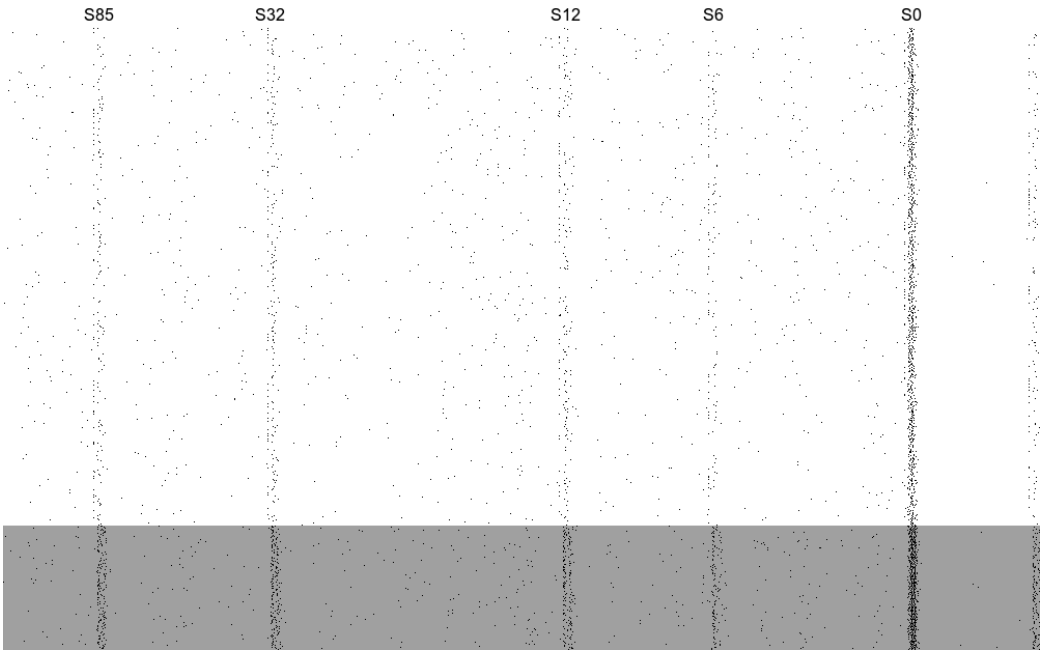



Figure 4.5: The response of the network to various input stimuli after an hour of simulated time. The y-axis is neuron number and the x-axis is time. A dot indicates that a neuron has fired. After training the system exhibits a much greater response when it receives the stimuli represented by $S_0$

Figure 4.5 shows the effect stimulus $S_0$ has on the network after around one hour of simulated time (we plot the last firing $S_0$ had, after simulating one hour.) The effect $S_0$ has on the system is now clearly larger than that of any of the other shown stimuli.

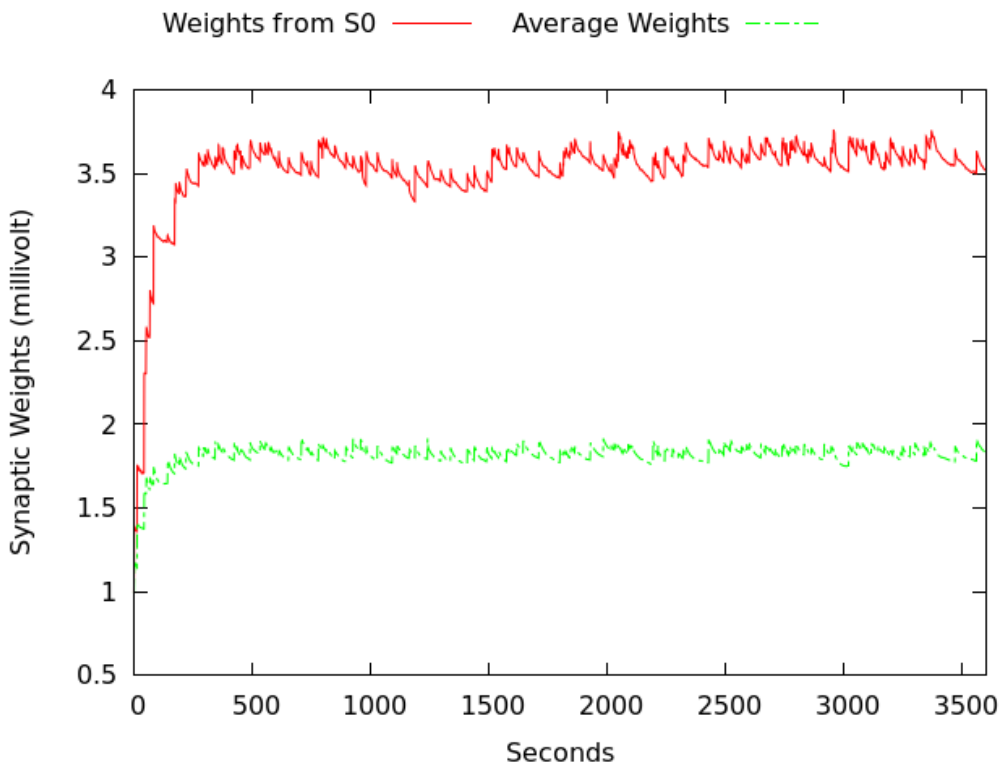



Figure 4.6: The average weight of neurons leaving neuron group $S_0$ has clearly been strengthened to a much greater degree than the average synapse in the network.

Figure 4.6 shows the average synapse weight out from $S_0$ and the average weights of all synapses in the network. As expected the weights leaving $S_0$ are larger, and it is clear that the system has learned to pay special attention to the activity of the neurons making up neuron group $S_0$.

We have successfully reproduced two of Izhikevich's experiments from [11] and feel confident that the system works as it should. Izhikevich has four experiments in [11]. It takes time to implement and run these experiments and in terms of increasing our confidence in our own implementation the returns are diminishing. Other researchers, among them [15], have reproduced Izhikevich's results, leaving us with no doubt that the methods presented in [11] are sound.

### 4.3.2 Experiment 1

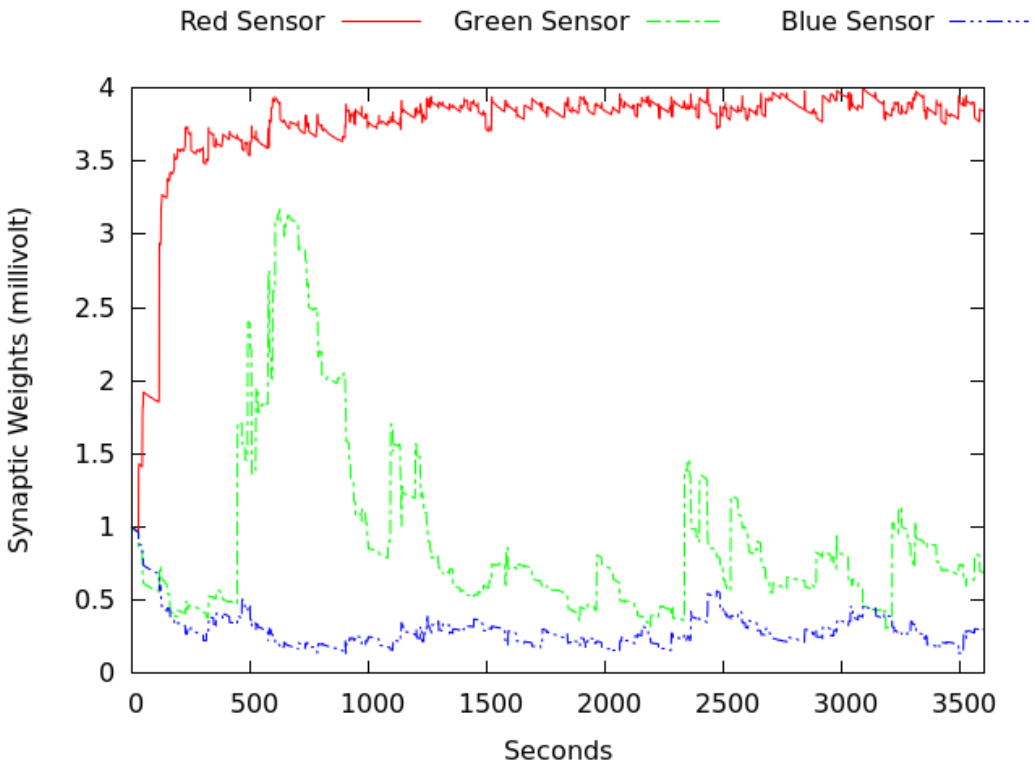



Figure 4.7: Classical conditioning experiment where red is the rewarded color. Plots of the average synapse weights between distance sensor and light neurons. In this experiment the patches move.

Figure 4.7 shows the synapse weights going from the distance sensors and to the neuron group controlling the robot light. The synapse weights corresponding to the rewarded color quickly rises to a high value where they remain. There are no blue tiles in the environment, only red and green. The patches are moving in this experiment, when the robot enters a patch of some color the patch will disappear and reappear at a random position in the environment.

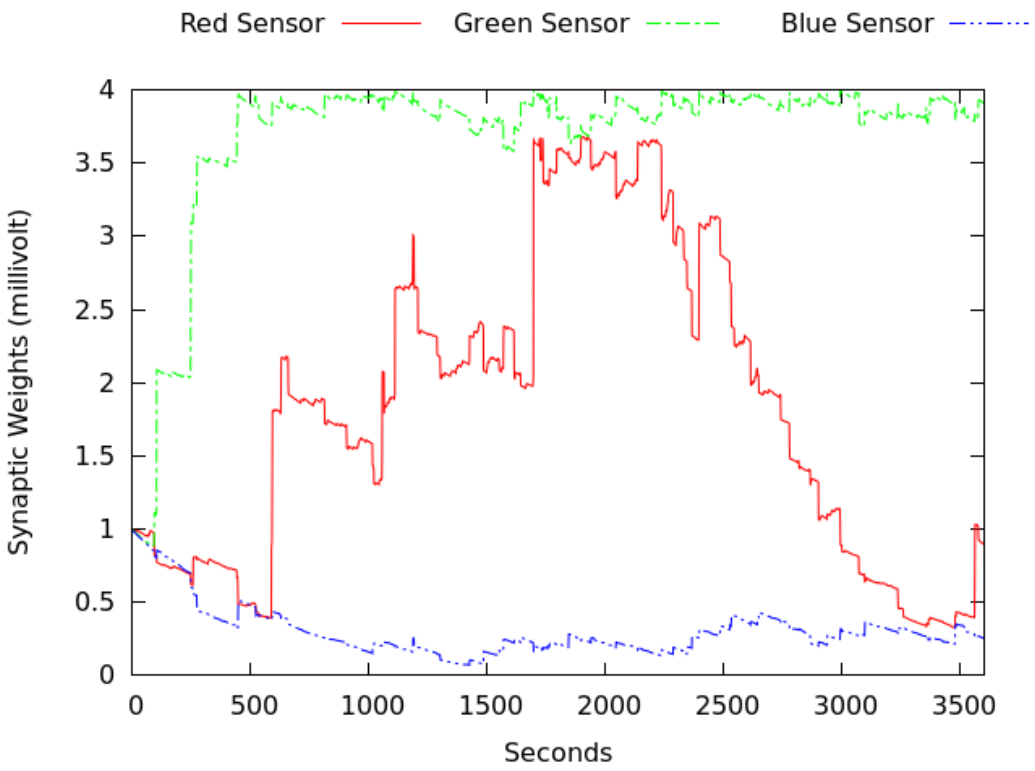



Figure 4.8: Classical conditioning experiment where green is the rewarded color. Plots of the average synapse weights between distance sensor and light neurons. In this experiment the patches move.

Figure 4.8 shows the same experiment with the colors reversed: now green patches are rewarded and the red patches are not. There are still no blue tiles in the environment and the patches are moving. The plots in figures 4.7 and 4.8 are very similar in nature, indicating that the learning mechanism works independently of which color we are rewarding.

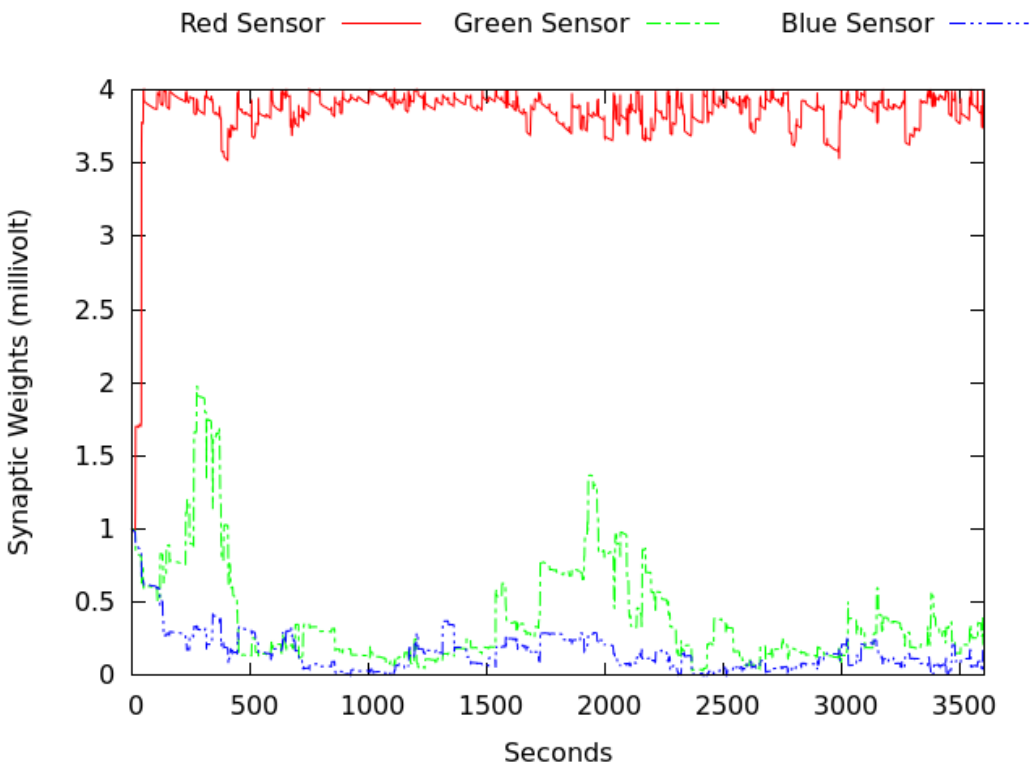



Figure 4.9: Classical conditioning experiment where red is the rewarded color. Plots of the average synapse weights between distance sensor and light neurons. In this experiment the patches do not move.

In figure 4.9 red is the rewarded color and we can see the effect the movement of the patches has on the experiment. Here the patches are stationary at all times, leading to a plot that looks a bit different. The absolute values and variance in the green line, showing the synapse weights out from the distance sensor of the unrewarded color, is much lower.

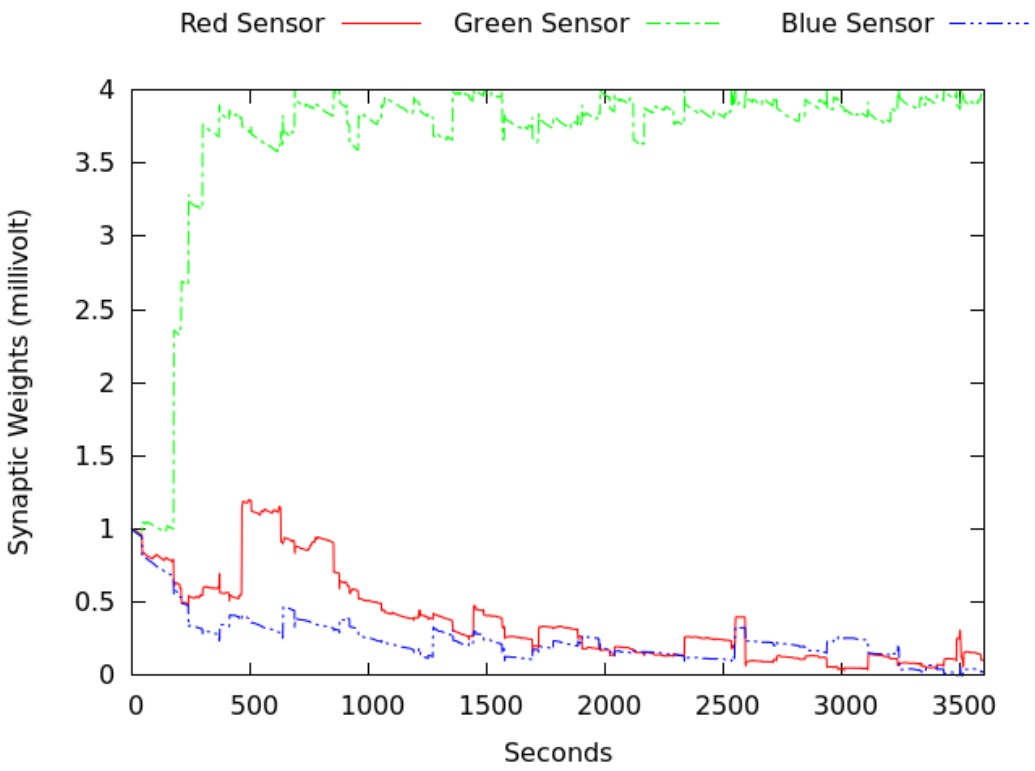



Figure 4.10: Classical conditioning experiment where green is the rewarded color. Plots of the average synapse weights between distance sensor and light neurons. In this experiment the patches do not move.

Figure 4.10 shows the classical conditioning experiment with green being the rewarded color and patches that are stationary at all times. Again we observe less variance and lower absolute variance in terms of the synapse weights of the distance sensor for the unrewarded color. In the case of the stationary patches we also observe that the learning mechanism works just as well when we reward the other color.

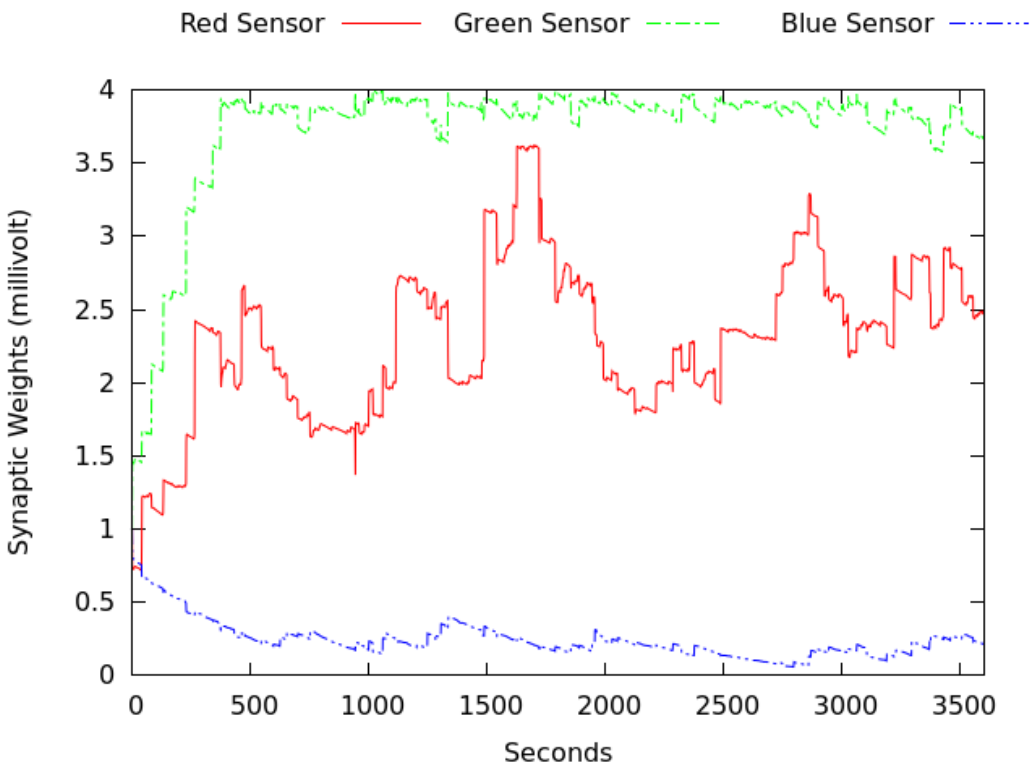



Figure 4.11: Classical conditioning experiment where green is the rewarded color. Plots of the average synapse weights between distance sensor and light neurons. In this experiment the patches are moving.

Figure 4.11 shows the run with highest variance and absolute values for the synapse weights out from the distance sensor group for the unrewarded color. This plot is of a run where green is rewarded and the patches are moving.

The robot is consistently able to learn that seeing the rewarded color with its distance is a reward predictor. It does, however, at times also react to the unrewarded color. In practice this means that it will turn on its lights, during certain phases of the experiment, when it sees patches of the unrewarded color with its distance sensor.

This behavior is not stable throughout the experiments, because it is not consistently reinforced. The reason this behavior arises in the first place is that sometimes patches of the two colors will appear next to one another. The robot will pass through them both, within some short time span, a reward will be delivered, and the robot will be unable to determine exactly what it did to bring about the reward.

This explanation is confirmed when we observe that the robot does not turn on its lights, upon seeing the unrewarded color, in the experiments with the stationary patches. Or, equivalently, that that the synapse weights going form the distance sensors for the unrewarded color and to the light sensors remain small. In the experiments with the stationary patches the distance between the rewarded patch and the unrewarded patch is large enough that the robot is unlikely to pass through them both in such a short time span that ambiguity in terms of cause of reward can occur.

In figure 4.11 it almost looks like the robot has been trained to react to both red and green patches. As mentioned earlier, the robot learns to react to the unrewarded patches by chance. This is fine, in the sense that there is real ambiguity in terms of what brought about a reward. The reason the reaction to the unrewarded color persists is that nothing is done to unlearn this behavior, except weakening through STDP caused by random post-then-pre firing patterns in the network. As we can see here, this mechanism might not be powerful enough, the network learns something by chance after around 400 s, and then never unlearns the behavior because it is coincidentally reinforced a few more times.

### 4.3.3 Experiment 2

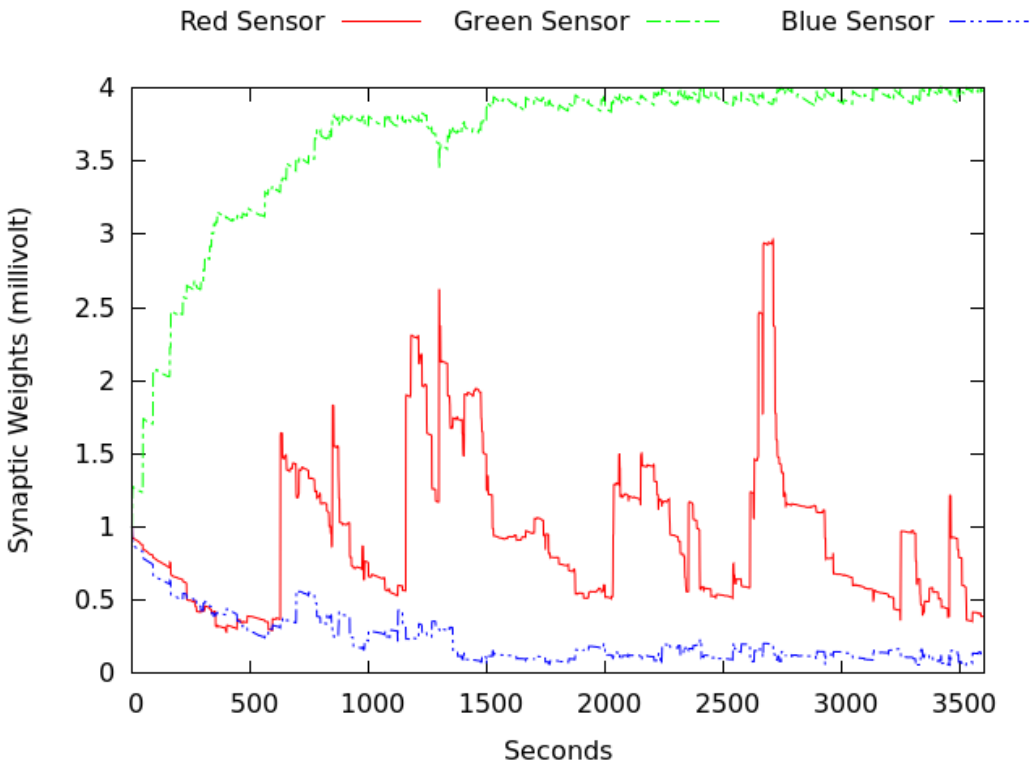



Figure 4.12: Instrumental conditioning experiment where green is the rewarded color. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

Figure 4.12 shows the synapse weights between the distance sensors group and the motor neuron group. The rewarded color is green and the patches are moving. The weights coming out of the green distance sensor group quickly rises to the maximum value, where they remain. Just as in experiment 1, we see quite a bit of variance in the average synapse weight out from the neuron group of the distance sensor for the unrewarded color.

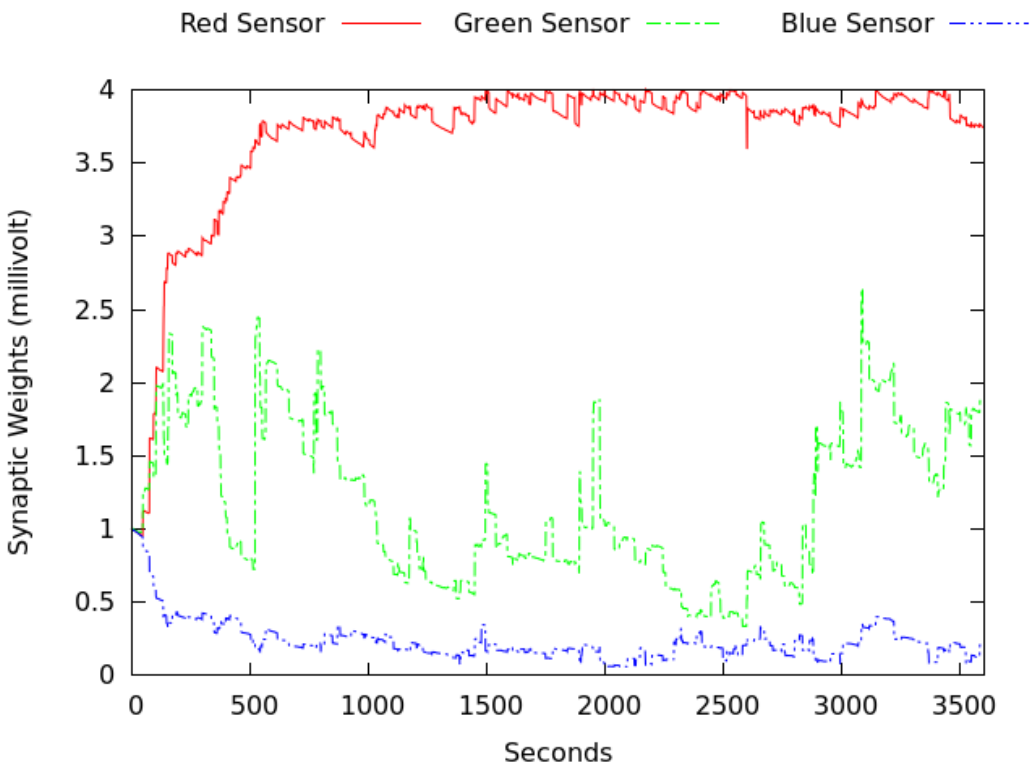



Figure 4.13: Instrumental conditioning experiment where red is the rewarded color. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

Figure 4.13 shows a very similar result when the colors are reversed, further indicating that the robot can learn what it should regardless of which color is rewarded.

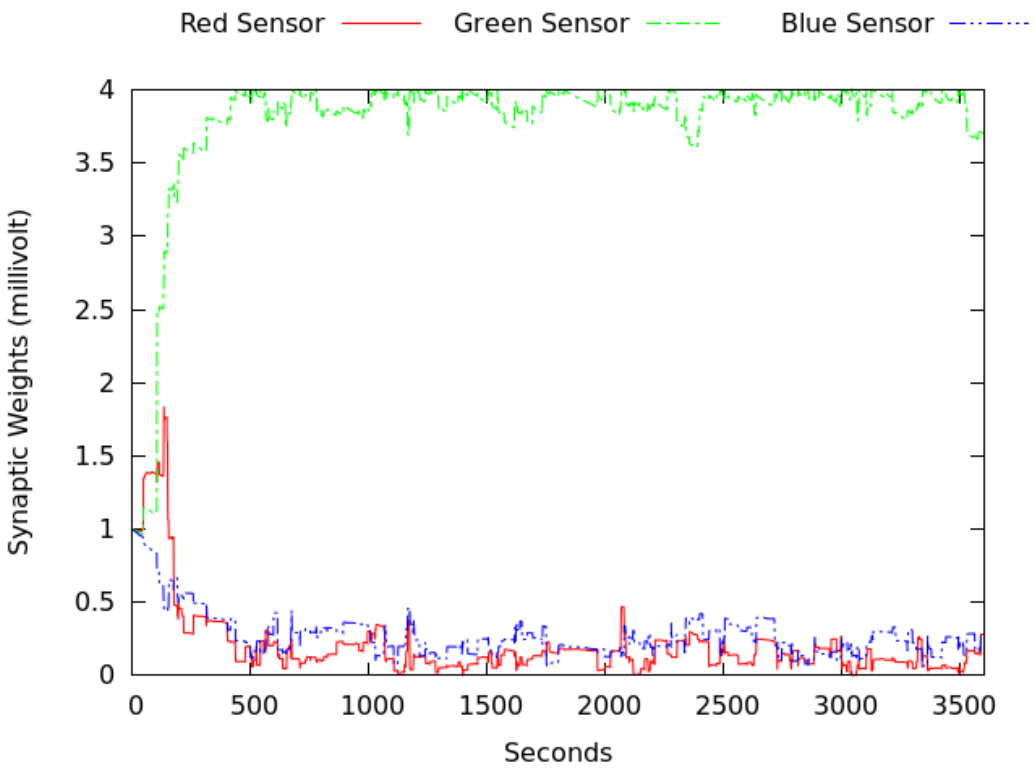



Figure 4.14: Instrumental conditioning experiment where green is the rewarded color. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are stationary.

In figure 4.14 we can see the same plot when the patches are stationary. In light of the results of experiment 1 the effect of moving and stationary patches on the experiment is as expected: lower variance and absolute values for the average synapse weight out of the neuron group corresponding to the distance sensors for the unrewarded color when the patches are stationary and an appropriate distance apart from one another.

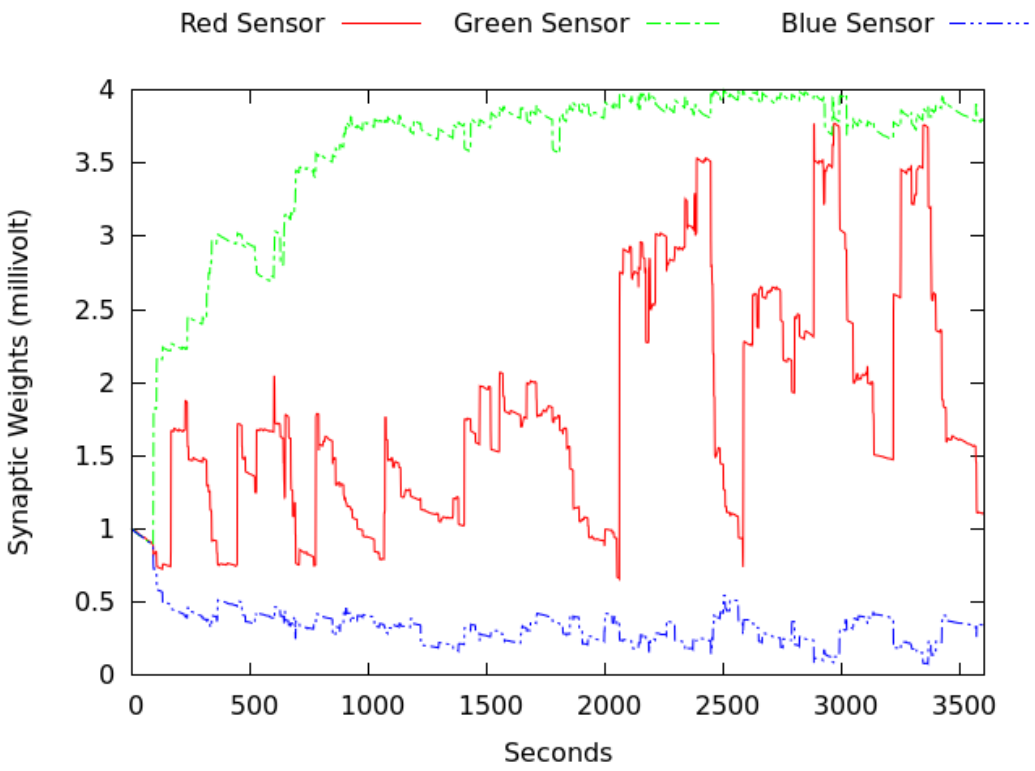



Figure 4.15: Instrumental conditioning experiment where green is the rewarded color. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

Figure 4.15 shows the run with the highest variance and absolute values in the average weight out from the neuron group corresponding to the distance sensor for the unrewarded color. Again the results are in line with what was seen in experiment 1: sometimes the robot is by chance taught to show interest in patches of a color that is not rewarded, and this behavior can be hard to unlearn.

| Run | Rewarded Color | Moving Patches | Percent on red | Percent on green |
|---|---|---|---|---|
| 1 | green | yes | 36 | 64 |
| 2 | green | yes | 46 | 54 |
| 3 | green | yes | 30 | 70 |
| 4 | green | yes | 44 | 56 |
| 5 | red | yes | 67 | 33 |
| 6 | red | yes | 60 | 40 |
| 7 | red | yes | 64 | 36 |
| 8 | red | yes | 55 | 45 |
| 9 | green | no | 22 | 78 |
| 10 | green | no | 21 | 79 |
| 11 | green | no | 29 | 71 |
| 12 | green | no | 29 | 71 |
| 13 | red | no | 77 | 23 |
| 14 | red | no | 60 | 40 |
| 15 | red | no | 74 | 26 |
| 16 | red | no | 73 | 27 |

Table 4.1: Results from all 16 instrumental conditioning experiments comparing the percentage of time steps spent on green and red patches.

Table 4.1 shows an overview of the results where we can see if the behavior of the virtual robot is in any way altered to favor green over red patches. We can see that the result is invariant to the color being rewarded, but whether or not the patches are moving or not has a large effect on the result.

To evaluate the results of the robot in this experiment we ran the experiment again with a robot that is hardcoded to move forward whenever a tile of the rewarded color is within its visual field. Since the robot's only way of increasing the amount of reward it receives is to react to rewarded tiles in front of it, this robot represents the optimal performance, given the robots other constraints.

Table 4.2 show the performance of this perfect robot in its environment. Compared to the performance of the SPNN controlled robot the maximum values, in terms of percent time steps on the rewarded color, is larger, and there seems to be less variance.

| Run | Rewarded Color | Moving Patches | Percent on red | Percent on green |
|---|---|---|---|---|
| 1 | green | yes | 34 | 66 |
| 2 | green | yes | 30 | 70 |
| 3 | green | yes | 30 | 70 |
| 4 | green | yes | 25 | 75 |
| 5 | red | yes | 73 | 27 |
| 6 | red | yes | 76 | 24 |
| 7 | red | yes | 75 | 25 |
| 8 | red | yes | 72 | 78 |
| 9 | green | no | 33 | 67 |
| 10 | green | no | 17 | 83 |
| 11 | green | no | 19 | 81 |
| 12 | green | no | 16 | 84 |
| 13 | red | no | 69 | 31 |
| 14 | red | no | 83 | 17 |
| 15 | red | no | 80 | 20 |
| 16 | red | no | 81 | 19 |

Table 4.2: Performance of a robot hardcoded to move forward when the rewarded color is in its visual field. This represents the best possible performance the robot can achieve in terms of maximizing its reward.

| Run | Rewarded Color | Moving Patches | Percent on red | Percent on green |
|---|---|---|---|---|
| 1 | green | yes | 52 | 48 |
| 2 | green | yes | 55 | 45 |
| 3 | green | yes | 48 | 52 |
| 4 | green | yes | 49 | 51 |
| 5 | red | yes | 49 | 51 |
| 6 | red | yes | 47 | 53 |
| 7 | red | yes | 50 | 50 |
| 8 | red | yes | 47 | 57 |
| 9 | green | no | 57 | 43 |
| 10 | green | no | 54 | 46 |
| 11 | green | no | 52 | 48 |
| 12 | green | no | 50 | 50 |
| 13 | red | no | 48 | 52 |
| 14 | red | no | 48 | 52 |
| 15 | red | no | 50 | 50 |
| 16 | red | no | 53 | 47 |

Table 4.3: Performance of a robot that does not use its distance sensors to guide movement at all. This robot represents the performance that is achieved by only relying on the movement rules for random movement

Table 4.3 shows the performance of a robot which does not use its distance sensors at all to guide movement. It only follows the movement rules for random movement. There is very little variance in this results and it is quite clear that it does not seem to favor either of the colors.

| Movement Rule | Moving Tiles | Mean Percent on Reward | Standard Deviation |
|---|---|---|---|
| Random | yes | 49.88 | 3.64 |
| | no | 48.25 | 2.96 |
| Optimal | yes | 72.13 | 3.36 |
| | no | 78.5 | 6.63 |
| SPNN | yes | 61.25 | 5.92 |
| | no | 72.88 | 6.03 |

Table 4.4: Means and standard deviations of the time steps the various robots spend on the rewarded patches.

In table 4.4 we can see a more succinct comparison of the three robots. It is clear that the SPNN controlled robot has learned something. It does quite a bit better than the robot that is moving about randomly.

In all the runs the robot is able to consistently learn that it can increase its cumulative reward by seeking out the rewarded patches. However, in the experiments where the patches are moving we also observe that the robot is more or less unsure about the unrewarded color. We often see that it will learn to move toward patches of the unrewarded color as well.

This happens when the two patches coincidentally appear in close proximity to one another, due to their random movements. The robot will enter both, in a short time period, receive a delayed reward, and be unsure about what caused the reward. This intuition is confirmed when we run the experiment with stationary patches, where the patches are separated by some distance, and observe that the robot does not learn to seek out the unrewarded color.

Since the two behaviors are mutually exclusive, in the sense that a robot seeking out unrewarded patches is not spending time seeking out rewarded patches, this recurring interest in the unrewarded patches goes a long way to explain the difference between the results for the robot with optimal behavior and the one controlled by the SPNN. This is also an explanation for the performance difference in terms of time spent on rewarded patches and unrewarded patches between experiments with moving and static patches in table 4.4

When comparing the results of the robot powered by the SPNN and the hardcoded robot using table 4.4 we see that there is a noticeable performance gap. Part of the explanation is the robot's propensity to seek out the unre-

warded patches, but another contributing factor is that the table is created using data based from the entire run. We have not excluded the learning phase of the experiment. The length of this learning phase varies, but as figure 4.15 shows the learning period can be significant. In that particular experiment it takes 30% of the experiment time to increase the weights, from sensor to motor neurons, to their maximum value.

### 4.3.4 Experiment 3

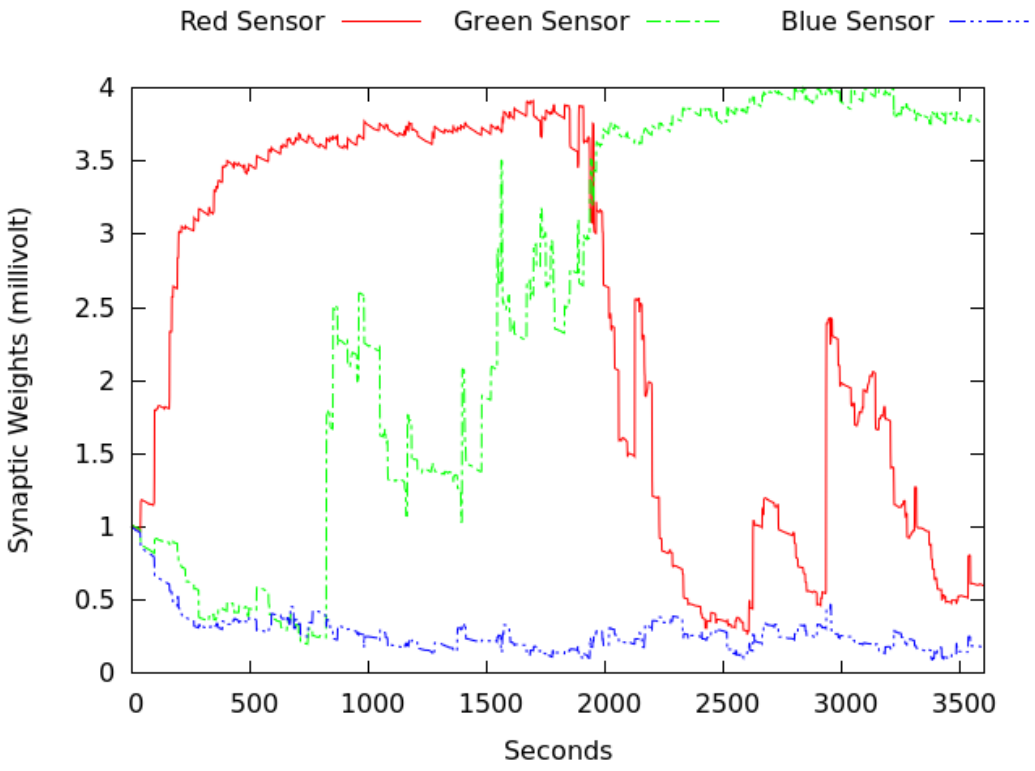



Figure 4.16: Extinction experiment where red is rewarded in the first half of the experiment and green in the last half. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

In figure 4.16 a typical run of our extinction experiment is showed. Initially red is rewarded and at the half way point, after simulating 1800 seconds,

the switch is made to reward green patches. In this experiment the patches are moving.

Just like in the previous experiments there is some interest in the patches of the unrewarded color. Nevertheless the switch occurring at around 1800 s of simulated time is very noticeable.

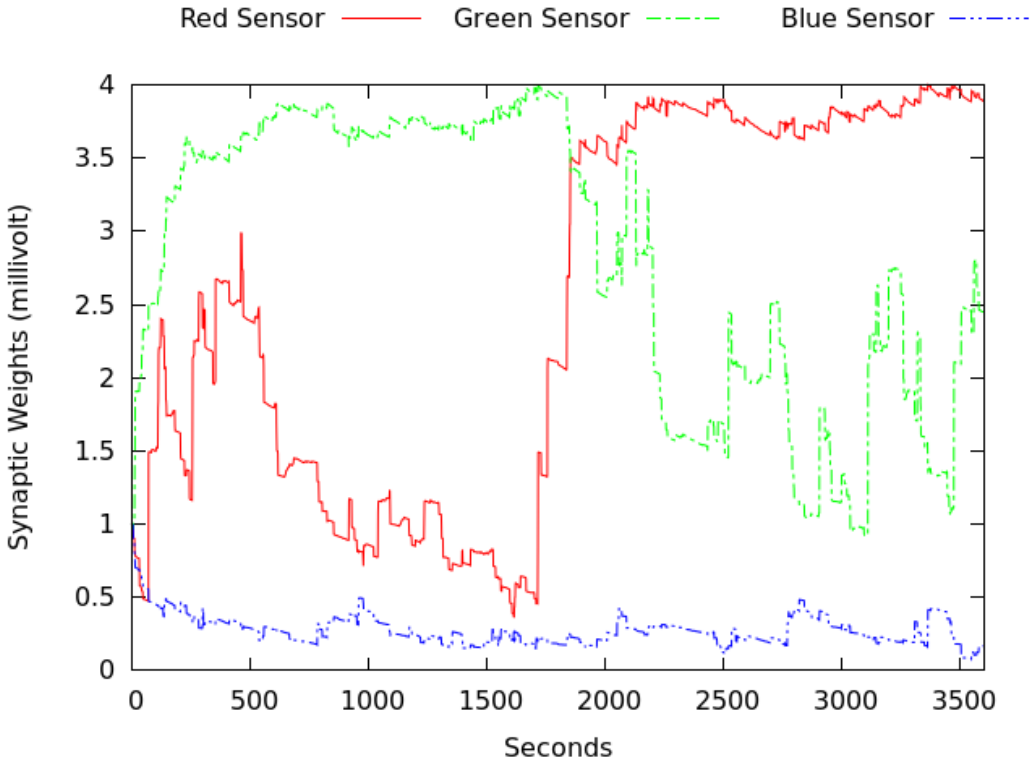



Figure 4.17: Extinction experiment where green is rewarded in the first half of the experiment and red in the last half. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

Figure 4.17 shows the same experiment with the rewarded colors reversed, initially green is rewarded, with red being rewarded in the latter half of the experiments. The patches are moving. In this experiment, as well, we observe that the robot is just as capable of learning what it should when the rewarded color is changed.

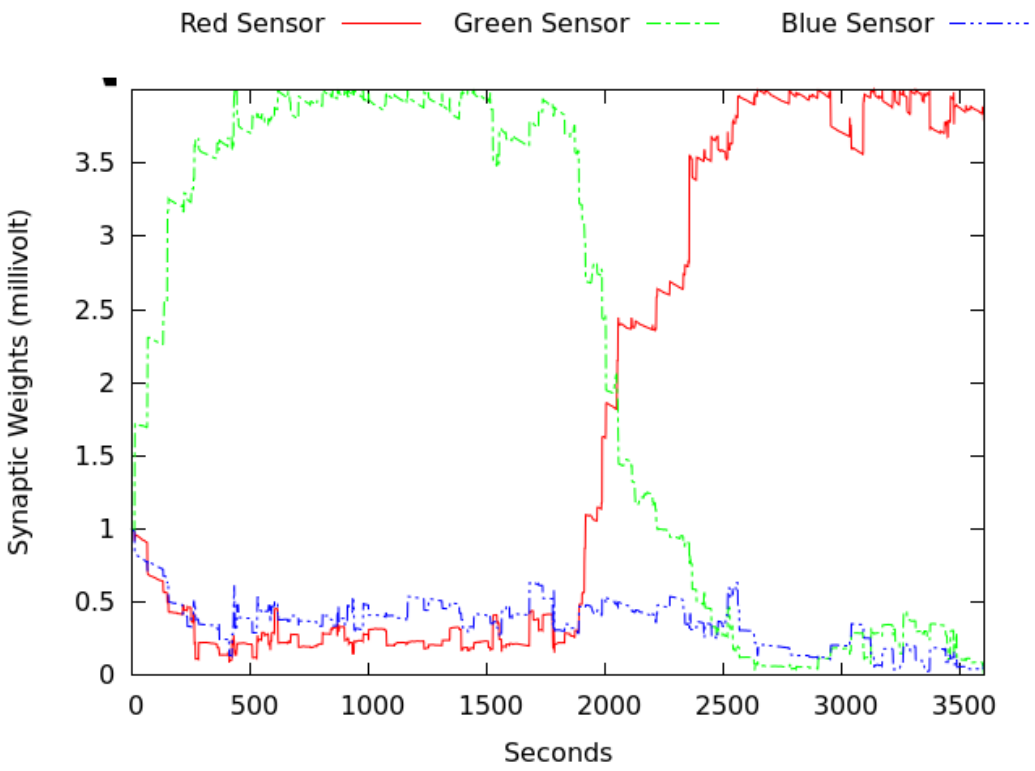



Figure 4.18: Extinction experiment where green is rewarded in the first half of the experiment and red in the last half. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are stationary.

Figure 4.18 shows an extinction experiment where the patches are stationary. Green is rewarded in the first half of the experiment and red in the last half.

Overall extinction seems to work very well. Just like in the previous two experiments, and for the same reason, there is quite a bit of variance in terms of the average synapse weights out form the distance sensor group detecting the unrewarded color, in the runs with moving patches, but this does not prevent the robot from learning what it should.

### 4.3.5 Experiment 4

Second order conditioning experiments have three phases. The first phase is just like a classical conditioning experiment where a conditioned stimulus (CS1) is followed by an unconditioned stimulus (US). In the second phase, a second-order conditioned stimulus (CS2) is presented along with CS1. Finally, in the third phase CS2 is presented on its own and at that point CS2 should be followed by US.

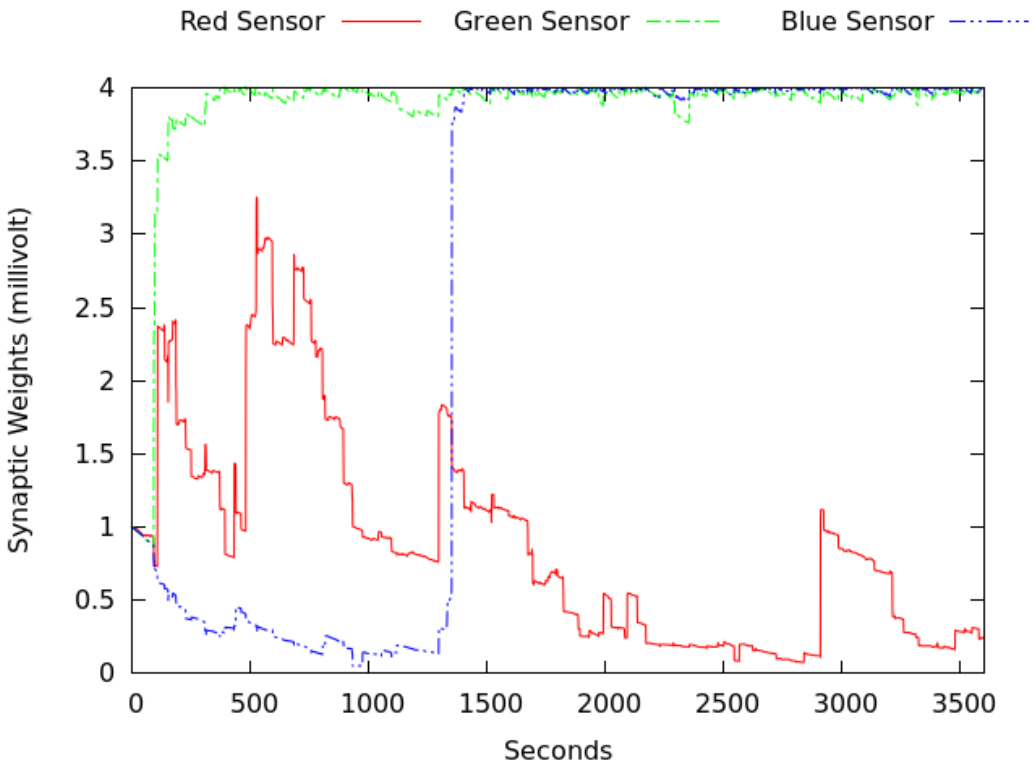



Figure 4.19: Second order conditioning experiment where green is the rewarded color. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

Figure 4.19 shows the plots of the weights between distance sensor and motor neurons. The rewarded color is green and the patches are moving. In the first phase of the experiment the robot successfully learns to react to the rewarded color and in the second phase it learns–very rapidly–that blue tiles also serve as a reward predictor indirectly. The second phase of the

experiment starts at 1200 s of simulated time; the robot's reaction is almost immediate. By the end of the run seeing blue tiles is just as likely to activate the robot's light as seeing the green tiles.

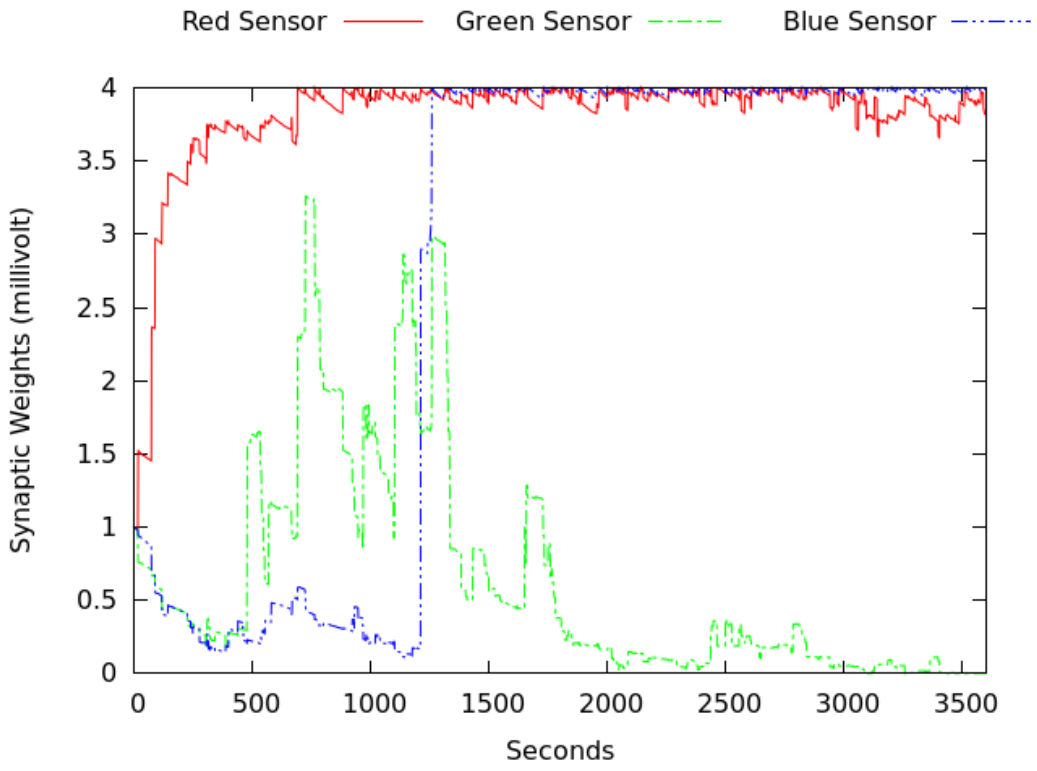



Figure 4.20: Second order conditioning experiment where red is the rewarded color and all green patches are surrounded by blue ones. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are moving.

Figure 4.20 shows an identical outcome when the rewarded color is changed from green to red. In both figure 4.20 and in figure 4.19 we can also see that introducing the blue tiles around the rewarded patch seems to have an effect on the variance in terms of the robot's interest in the unrewarded tiles. This phenomenon occurs in all the runs we are not showing as well.

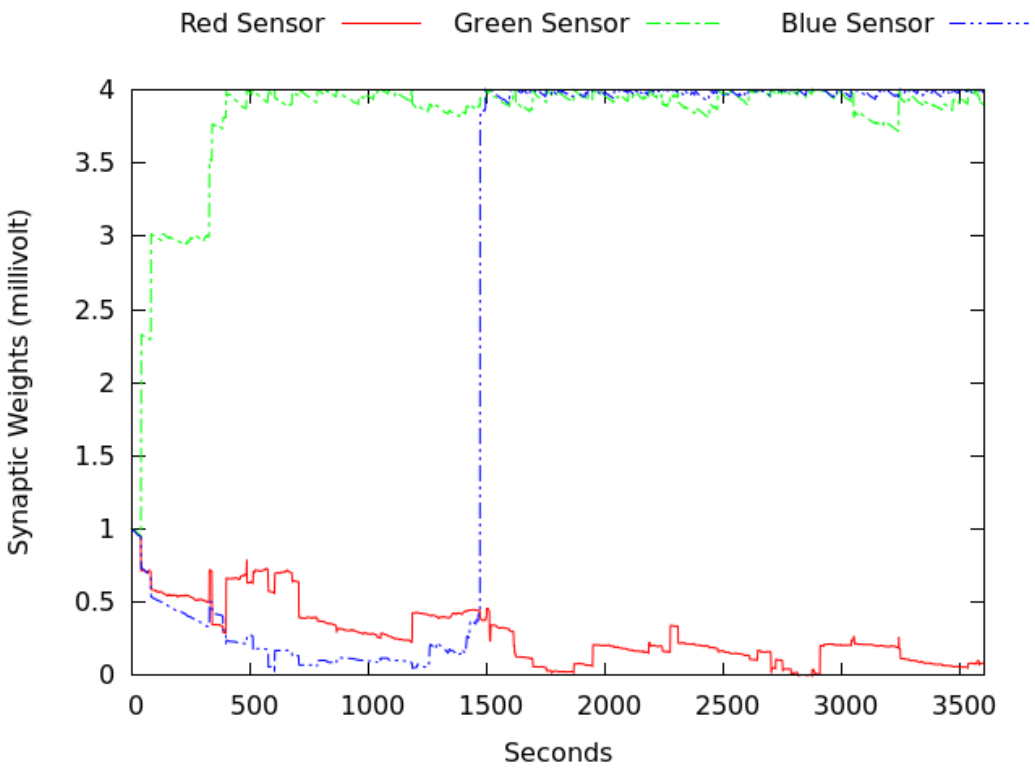



Figure 4.21: Second order conditioning experiment where green is the rewarded color, and all green patches are surrounded by blue ones. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are stationary.

As can be seen in figure 4.21 the outcome is the same when the patches are stationary, but the rise to the maximum value is slightly quicker and the variance in the weights leaving the sensor group for the unrewarded color, in this case red, is smaller.

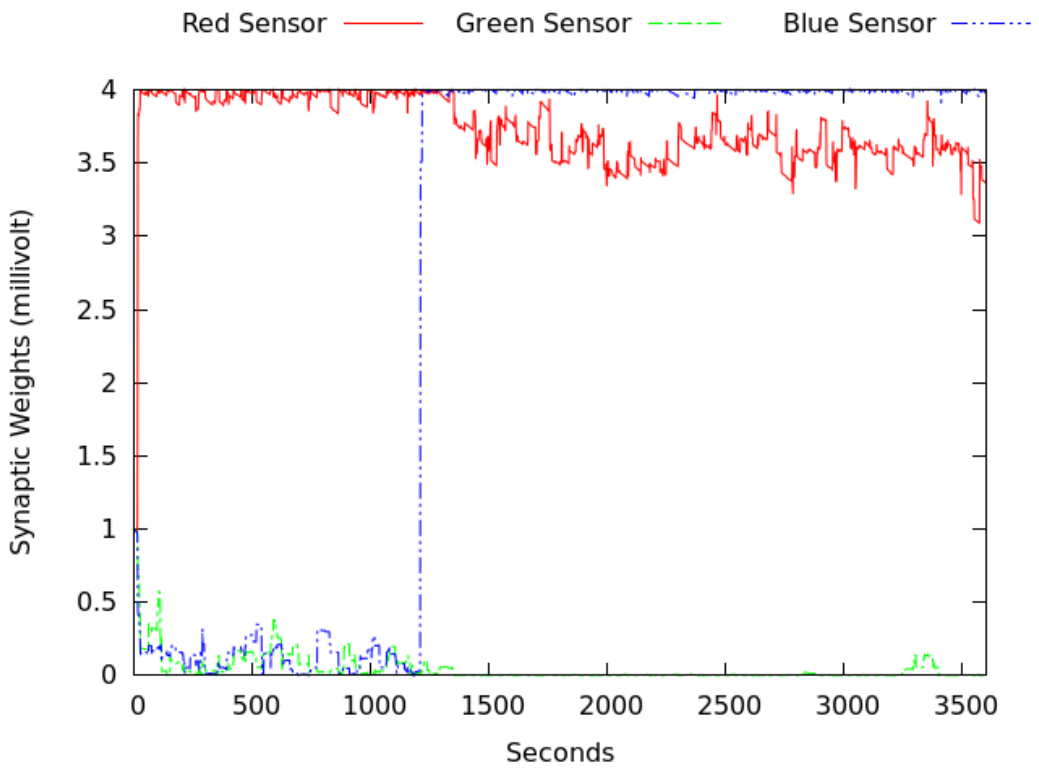



Figure 4.22: Second order conditioning experiment where green is the rewarded color, and all green patches are surrounded by blue ones. Plots of the average weights between distance sensor and motor neurons. In this experiment the patches are stationary.

Figure 4.22 shows a plot for weights going from the distance sensors to the motor neurons when the rewarded color is changed from red to green with stationary patches. Again we see that there is little variance in the weights between the distance sensor group and the motor neurons for the unrewarded group.

The results for the second order experiments are quite interesting. All the plots of synapse weights are very similar to that of figure 4.20 with a very rapid rise to the maximum allowable value for both of the distance sensors able to predict the future rewards. What is most interesting, however, is how the experiments seem to exhibit less variance than seen in the other experiments with regard to the average synapse weights out from the distance sensor for the unrewarded color. The robot never pays less attention to patches of an unrewarded color in these experiments, even when the unrewarded and rewarded patches appear in close proximity to one another, which they are wont to do, by chance, with a certain frequency.

We speculate that the reason the robot is able to ignore the irrelevant color, is because the rewarded patches are surrounded by blue ones, but the unrewarded patches are not. This means that the robot has more information it can use to separate the two situations from one another, which makes it easier to learn which actions and stimuli precede the reward.

A more mundane explanation is the effect the blue tiles have on the geometry of the experiment: when the blue tiles surround the rewarded patch the unrewarded and rewarded patch cannot appear as close to one another as they otherwise can. This decreases the likelihood that the robot passes through both patches in a short time span and becomes confused as to exactly what triggered a reward.

# Chapter 5

# Evaluation and Conclusion

This chapter concludes the report. In section 5.1 the results presented in chapter 4 is evaluated in light of the research questions presented in section 1.2. Section 5.2 contains our discussion of the results of chapter 4. In section 5.3 we review our contribution to the field of artificial intelligence. And section 5.4 contains our suggestions for future research.

## 5.1 Evaluation

In section 1.2 the research goal was stated as follows:

> Investigate whether dopamine modulated STDP can be used to solve the distal reward problem, in a situated and embodied agent, by using reinforcement learning to change the behavior of a virtual robot.

From this broad goal we derived the following, concrete, research questions:

**Research question 1**: Can we demonstrate classical conditioning, in a situated and embodied agent, using dopamine modulated STDP?

**Research question 2**: Can we demonstrate instrumental conditioning, in a situated and embodied agent, using dopamine modulated STDP?

**Research question 3**: Does our situated and embodied agent, using dopamine modulated STDP, exhibit extinction of acquired behavior?

**Research question 4**: Can we demonstrate second order conditioning, in a situated and embodied agent, using dopamine modulated STDP?

### 5.1.1 Research question 1

The results presented in section 4.3.2 shows that classical conditioning is possible with our setup. The unconditioned response, turning on the light, has now been moved from only being triggered by the ground sensor (the unconditioned stimulus) to also be triggered by the distance sensor, in other words there is now a conditioned response.

The robot consistently learns the conditioned response in our experiments. The robot is, however, at times unsure whether observing the unrewarded color also predicts an upcoming reward causing it to turn on its lights in situations where it should not.

### 5.1.2 Research question 2

The results shown in 4.3.3 shows that the robot exhibits a clear preference for the rewarded color. In fact, the best results are very close to the performance of the control robot with optimal performance.

The robot is consistently better at collecting rewards after a training period.

### 5.1.3 Research question 3

The results from section 4.3.4 make a convincing case that extinction of learned behavior is possible in our system.

The robot is consistently able to learn one behavior and unlearn it to favor another. There is quite a bit of variance in the degree with which the two behaviors are mutually exclusive in the experiments with the moving patches. In the experiment with the stationary patches the extinction behavior is very clear.

### 5.1.4 Research question 4

The results in section 4.3.5 show that second order conditioning is possible in our system. The robot comes to learn that seeing the blue patches predicts the later occurrence of a reward.

Second order conditioning is consistently achieved in the system. In fact there is less variance in terms of the robot reacting to the unrewarded color in these experiments.

## 5.2 Discussion

The system performance is in line with what was expected based on work done by other researchers [2] [11] [19]. In general, we believe that the methods used in this thesis are sound, and worthy of further investigation, but that that we might still be missing a few pieces in terms of achieving truly effective reinforcement learning.

### 5.2.1 Superstition

In the instrumental conditioning experiment we observed that the robot sometimes became superstitious and believed that the unrewarded patches were worth seeking out. Since seeking out the two types of colored patches are mutually exclusive in terms of time management the system failed to achieve optimal performance. The only mechanism we had to combat this behavior was to increase the level of depression due to random post-then-pre firings in the network, causing the robot to forget the irrelevant behavior more quickly.

In an environment like the one in our experiment this strategy, of increasing the effect of random depression, is not very effective because the events causing the unwanted behavior to be acquired are so frequent.

An alternative would be to punish the robot when it entered the irrelevant patches, but this is not a good option either. We want the robot to figure out what is irrelevant on its own; labeling things explicitly as "bad" does not scale. We also believe that it is important that things that are *irrelevant* and *bad* or *painful* can be treated in different ways by the system.

### 5.2.2 On reducing interest in the unrewarded patches

The problems we have seen in regard to the robot's interest in the patches of the unrewarded type are actually quite reasonable. Since the reward is given after a certain amount of time, the robot cannot know exactly what was rewarded. In order to find out what caused the reward the robot would have to compare the current situation with other reward-inducing situations and conclude that the common element was entering a rewarded patch and that entering the unrewarded patch was incidental.

The work done by Schultz et al. in [17] illuminates one possible way in which we could alter this system to reduce interest in the unrewarded patches. In these experiments it was shown that when the monkeys did not receive an expected reward the dopamine concentration dropped below baseline. In terms of our robot this means that it would, correctly, learn that both the rewarded and unrewarded patches might cause a reward when they appear in close proximity with one another. However, consequently when the robot encounters an unrewarded patch alone, and does not receive a reward, it will be "disappointed" and the previously learned behavior will be extinguished to a certain degree. This additional way of reducing the strength of the synapses is likely to complement random depression through STDP in a manner mirroring the effectiveness of the dopamine modulated strengthening.

## 5.3 Contributions

We have demonstrated classical conditioning, instrumental conditioning and extinction as well as second order conditioning in an embodied context. The mechanisms used was dopamine modulated STDP, based on the work of Izhikevich in [11]. Compared to Chorley et al. in [2], who also based their work on [11], we are able to get away with a network 40% smaller, using simpler robot sensors and environment. Chorley et al. also use varying conduction delays, in the interval [1, 10] ms in the neural network. We managed with a uniform conduction delay of 1 ms, showing that the added complexity is unnecessary. To our knowledge our demonstration of second-order conditioning, using dopamine modulated STDP is novel.

## 5.4 Future work

We suggest that investigating "disappointment" in networks using dopamine modulated STDP would be interesting. The motivation behind this is to have a complementary mechanism for depression, based on the agent's previous experiences.

Our experiments are done with a conduction delay of 1 ms between all neurons. It would also be interesting to investigate the effect of varying and non-uniform conduction delays.

# Appendix A

# Literature review protocol

# 1 Structured Literature Review

The purpose of this document is to aid us in performing a literature review. The goal of a structured literature review is to document the process used in obtaining, and selecting, the works that our own will build upon.

# 2 Background

Our thesis is in the domain of artificial neural networks. Specifically, we will concern ourselves with spiking neurons and reinforcement learning. Spiking neurons have been shown to be biologically plausible models and the recent theoretical work is promising. However, the model has not found many practical applications as of yet. We want to investigate if we can use a model based on spiking neurons, in a virtual robot, to show examples of reinforcement learning, in the form of classical conditioning. The goal of the literature review will be to find:

1. Papers on spiking neurons and reinforcement learning.
2. Papers where spiking neuron models are put to practical use.

# 3 Research Questions

- RQ1: Is it possible to do reinforcement learning with spiking neurons and neuromodulators?
- RQ2: Has anyone else used spiking neuron models as the brain of a robot?
- RQ3: How does a spiking neuron model work? What are the limitations and benefits of the model?

# 4 Search Process

## 4.1 Sources

In previous literature searches we have opted to cast a very wide net. What we found was that here–as in so many cases–the Pareto principle applies: 20% of the sources searched gave 80% of the relevant resources, or more. We are therefore opting to focus on a few key sources, which have yielded good results in the past.

- IEEE Xplore
- CiteSeer
- SpringerLink
- Google Scholar

## 4.2 Search Terms

| Group 1 | Group 2 | Group 3 | Group 4 |
|---|---|---|---|
| Spiking neuron | Model | Robot | Reinforcement learning |
| | Network | Controller | Conditioning |
| | System | Agent | Learning |
| | Neuromodulator | | |

Table 1: Table showing the search terms we used.

The search terms we used to get at our research questions are shown in table 1. A search string is created by combining the search term from group 1 with one term from one or more of the other 3 groups.

# 5 Study Selection Process

Using these search terms we sometimes uncovered far more papers than we could reasonably sift through. In these cases, initial screening was based on the title of the paper and the paper preview displayed in the search engine. We processed the first 200 hits deemed "most relevant" by the search engine. This first step of the selection process, and the next step where we read the paper abstract to determine relevance, was guided by our inclusion criteria.

## 5.1 Study Inclusion Criteria

At least inclusion criteria (IC) 1 has to be met, along with either IC2 or IC3:

- IC1: The study's main concern is spiking neurons of the Izhikevich type.
- IC2: The study involves the use of a neuromodulator.
- IC3: The study involves an agent controlled by a network of spiking neurons.

## 5.2 Study Quality Criteria

Once a paper was included, we used the following quality criteria (QC) to give it a score. A hit on each QC gave the paper an additional point. Papers with higher scores were given priority in terms of scheduling thorough readings. We did not set a predefined cap i.e. to only read papers with a score above **x**.

- The study involves spiking neurons based on Izhikevich's work.
- The study involves an *agent* or *robot* being controlled by a spiking neuron based "brain".
- The study includes empirical results.
- The study is placed in a proper context of other studies.
- The system used is thoroughly explained.
- The methods used are thoroughly explained.
- The results are good, or *interesting*, for some interpretation of *interesting*.

# Bibliography


[1] Daniel Bush, Andrew Philippides, Phil Husbands, and Michael O'Shea. Investigating stdp and ltp in a spiking neural network. In *From Animals to Animats 9*, pages 323–334. Springer, 2006.

[2] Paul Chorley and Anil K Seth. Closing the sensory-motor loop on dopamine signalled reinforcement learning. In *From Animals to Animats 10*, pages 280–290. Springer, 2008.

[3] R Christopher Decharms and Anthony Zador. Neural representation and the cortical code. *Annual review of neuroscience*, 23(1):613–647, 2000.

[4] Wulfram Gerstner, Richard Kempter, J Leo van Hemmen, Hermann Wagner, et al. A neuronal learning rule for sub-millisecond temporal coding. *Nature*, 383(6595):76–78, 1996.

[5] M Gho and FJ Varela. A quantitative assessment of the dependency of the visual temporal frame upon the cortical rhythm. *Journal de physiologie*, 83(2):95, 1988.

[6] Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[7] Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.

[8] Giacomo Indiveri, Elisabetta Chicca, and Rodney Douglas. A vlsi array of low-power spiking neurons and bistable synapses with spike-timing de-

pendent plasticity. *Neural Networks, IEEE Transactions on*, 17(1):211–221, 2006.

[9] Eugene M Izhikevich. Simple model of spiking neurons. *Neural Networks, IEEE Transactions on*, 14(6):1569–1572, 2003.

[10] Eugene M Izhikevich. Which model to use for cortical spiking neurons? *Neural Networks, IEEE Transactions on*, 15(5):1063–1070, 2004.

[11] Eugene M Izhikevich. Solving the distal reward problem through linkage of stdp and dopamine signaling. *Cerebral Cortex*, 17(10):2443–2452, 2007.

[12] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural networks*, 10(9):1659–1671, 1997.

[13] Wei-Xing Pan, Robert Schmidt, Jeffery R Wickens, and Brian I Hyland. Dopamine cells respond to predicted events during classical conditioning: evidence for eligibility traces in the reward-learning network. *The Journal of neuroscience*, 25(26):6235–6242, 2005.

[14] Hal Pashler and Randy Gallistel. *Stevens' Handbook of Experimental Psychology, Methodology in Experimental Psychology*, volume 3. Wiley, 2002.

[15] Kira Radinsky and Michael Goldish. Pavlovian and instrumental conditioning in spiking neural networks.

[16] Edmund T Rolls, Martin J Tovee, et al. Processing speed in the cerebral cortex and the neurophysiology of visual masking. *Proceedings of the Royal Society of London-B-Biological Sciences*, 257(1348):9–16, 1994.

[17] Wolfram Schultz. Predictive reward signal of dopamine neurons. *Journal of neurophysiology*, 80(1):1–27, 1998.

[18] Andrea Soltoggio, P Durr, Claudio Mattiussi, and Dario Floreano. Evolving neuromodulatory topologies for reinforcement learning-like problems. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 2471–2478. IEEE, 2007.

[19] Andrea Soltoggio, Andre Lemme, Felix Reinhart, and Jochen J Steil. Rare neural correlations implement robotic conditioning with delayed rewards and disturbances. *Frontiers in Neurorobotics*, 7:6.

[20] Sen Song, Kenneth D Miller, and Larry F Abbott. Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature neuroscience*, 3(9):919–926, 2000.

[21] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

[22] Simon J Thorpe and Michel Imbert. Biological constraints on connectionist modelling. *Connectionism in perspective*, pages 63–92, 1989.

[23] Jilles Vreeken. Spiking neural networks, an introduction. *Technical Report UU-CS*, (2003-008):1–5, 2003.