I don’t know how to start this shit, yo.. now

Summary

Write your summary here...

Preface

Write your preface here...

Table of Contents

List of Tables

List of Figures

Abbreviations

ANN	=	*Artificial* Neural Network
RC	=	Reservoir Computing

Chapter 1

Introduction

> They're trying to understand what space is. That's tough for them. They break distances down into concentrations of chemicals. For them, space is a range of taste intensities.
>
> Greg Bear, Blood Music

Countless manhours have been spent improving the design and manufacturing process of the digital computer, creating more and more complex architectures capable of operating at ever greater speeds. The brain is not subject to this top down design philosophy, yet through a process of self organization neurons are capable of forming highly complex networks capable of solving complex tasks, with far greater energy efficiency, robustness and parallellism than any designed processor. Inspired by work done in the field of material computing systems such as the mecobo platform of nascence [citation needed], a similar system has implemented using living neural networks grown from human stem cells. Neurons are grown *in vitro* [1] on *Micro Electrode Arrays* 1.1:A, C (MEA) are interfaced with a digital computer via voltage spike measurements 1.1:B, forming a hybrid neuro-digital system. This system utilizes the theoretical framework of *Reservoir Computing* to help translate between the digital and biological parts of the system, allowing it to solve simple tasks as a proof of concept. The ultimate goal of the cyborg is to further our understanding of the underlying principles that governing how nature computes.

1.1 Complexity

In the 50's and 60's there was much optimism in the burgeoning field of artificial intelligence. In 1965 H. A. Simon claimed "machines will be capable, within twenty years, of doing any work a man can do."[5] , while Marvin Minsky boldly claimed in 1967 that

[1] From latin, literally "in glass", as opposed to in the body.

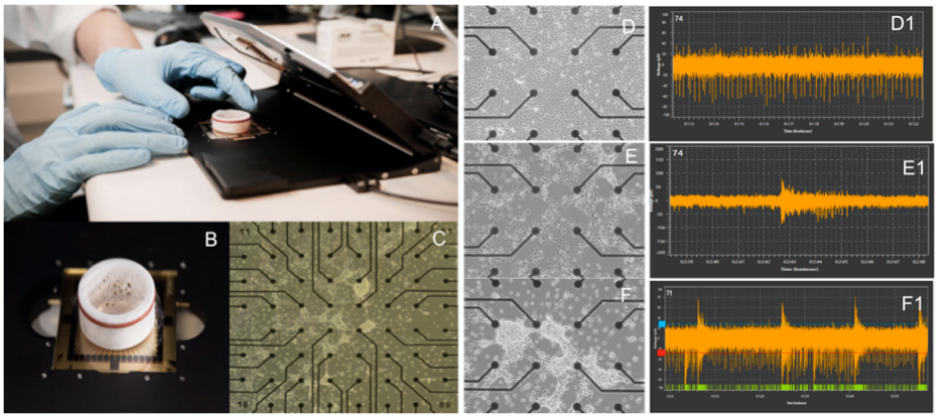

Figure 1.1: From socrates webpage

"Within a generation ... the problem of creating 'artificial intelligence' will substantially be solved." [1]. Had they chosen to predict any other field, such as logistics, information sharing or communications their statements would have been prophetic and visionary, so why did artificial intelligence turn out so differently? The researchers sought to make machines that could use logic similar to that of high level human thinking. Therefore, it followed that the machine had to be programmed with rules governing logic in order to reach sound conclusions. To represent the prior and deduced knowledge, the researchers designed programming languages such as lisp that could accurately describe these operations. In order to actually execute these lisp programs hardware had to be created, supporting the primitive operations such as addition, subtraction and loading from memory. Regardless of the underlying platform a lisp program did not change meaning, and the binary adder in the heart of the processor never interacted with the floating point adder. In short, each piece of the puzzle was self contained, and the *Complexity* of the systems behavior was similar to that of the sum of its parts, which made it feasible to build large systems. Nature, on the other hand, applies a completely different method. Complex structures appear with no blueprint, arising from a process of *self-organization* driven by a set of growth rules. This self organizing process is capable of producing incredibly complex, robust and diverse structures whose functionality arises not from specialized components working in isolation, but from the interaction of many components. Applying the superposition principle to the processor makes sense, it allows us to study each individual component in isolation before investigating how they interact. On the other hand, applying the superposition principle to nature leaves us blind to the fact that the purpose of most components is to interact rather than performing a specific function.
It seems the reason the researchers failed was their underestimation of the role played by complexity, believing it was an incidental product of evolution rather than a necessity. TODO: En setning her som motiverer alt over. A la "Derfor valgte va lage en cyborg i stedet for dille med n-te ordens logikk i LISP."

1.2 Computation

The invention of the digital computer will be remembered as one of, if not the most significant technological advances of mankind. With this neatly designed clockwork machine the concept of computation seemed rather straight forward, where each executed machine instruction corresponded to some unit of computation. This idea of computation is very convenient, and it fails utterly when applied to biological systems. In a recent example of this from 2005 [Cite The Singularity Is Near: When Humans Transcend Biology] Ray Kurzweil claimed that computers would be as powerful as a mouse brain by 2020, a prospect that looks rather unlikely in 2018. Other than in the head of computer scientists, neurons have no notion of floating point operations or branching logic, and measuring the computational capabilities of a brain in FLOPS is grossly underestimating what computing *can* be. In spite of the digital computers shortcoming compared to the human brain, the conventional digital approach to computing has been hugely successful in solving problems that humans are bad at, and is so ubiquitus that other approaches have been dubbed *Unconventional Computing*. Unconvential computing, as implied by the name, comes in many forms such as buckets of water [2], or blobs of carbon nanotubes [cite nascence]. In these unconventional approaches it becomes harder and harder to pin down exactly what computation is and what distinguishes it from any ordinary physical process.

Chapter 2

Background

Creating a cyborg is a massive cross disciplinary effort, and if a scope is not clearly defined the background runs the risk of becoming equally massive. The goal of the thesis is to create a hybrid neuro-digital system capable of controlling a simple robot, and by extension to create a "bridge" between neural and digital. Each section in the background shares this bridge as a red thread, and consequently topics that are not directly related to the thesis' goal are discarded. The background is laid out as follows: First *Complex Systems* are introduced as a framework to discuss the computational capabilities in a wide range of systems that exhibit system dynamics similar to that of neurons. Modeling neurons as a complex systems is a first step towards establishing a common "language" between neural activity and digital logic. Next, *Evolution in Materio*, EiM for short, introduces computation done in unstructured matter through the process of evolution. The goals of EiM are closely aligned to the goal of this thesis, as both study massively parallel computation happening in physical matter shaped by the process of evolution. Next section, *Neurons As Computers* introduces neurons with a strong focus on their computational capabilities. Building on the EiM section, this section motivates a necessary reduction of scope, proposing a simplified model of the computing neuron and a medium of communication between neuron and digital. Finally, *Reservoir Computing* is introduced, tying together the previous sections by introducing the framework of reservoir computing in order to establish a common "language" between neural cultures and digital.

2.1 Complex Systems

When designing systems in a top down manner it is imperative to control the complexity of the system, or we are unable to reason about what it does and how it does so. By carefully managing complexity it is possible to make systems that are *complicated* in their makeup, yet with a simple, understandable behavior. Larger systems can be assembled by combining smaller systems, and since each subcomponent acts in a simple, predictable way, then so does the system as a whole. The effort necessary to understand the behavior of system as a whole is the same as the sum of the effort required to understand each of

its subsystems, and so forth. When studying most systems that occur naturally however, such as economical systems, ecosystems, materials and how neurons together make up the mind, this does not hold. This is because the behavior we would like to model and predict cannot be traced back to the individual agents of the system. Although the individual parts of these *Complex Systems* can be just as complicated as in human engineered ones, it is the *Non-linear* dependence between these parts that decides the behavior of the system as a whole. It is these complex interactions that allows systems to spontaneously *Self Organization* and adapt to change. The "force" that drives the self organization of complex systems is feedback. Positive feedback loops can cause a chain reaction that completely alter the dynamics of a system, while negative feedback acts as a dampener, allowing the system to have some measure of stability. Systems with mostly negative feedback are ordered and predictable, while systems with mostly positive feedback are chaotic and unpredictable, seemingly random. It is at *The edge of chaos* where positive and negative feedback are evenly matched that systems can spontaneously self organize without being chaotic. In the context of evolution seems necessary for systems to exist at this edge of chaos. For a plant it is essential to be able to respond to changes in light conditions, but it is equally important to maintain integrity. No two pines are alike, but they all have bark on the outside and they all grow upwards. In a swiss watch everything stops working if a single spring or cog is replaced with a slightly different one, yet in nature no two animals or plants are alike. The *robustness* of complex systems goes beyond mere survival, it's what makes the incremental changes of evolution possible in the first place! While the neurons that make up neural networks are both complex and complicated, it is in the interactions between them that computation happens, and this complex interaction can be modeled in a much simpler system.

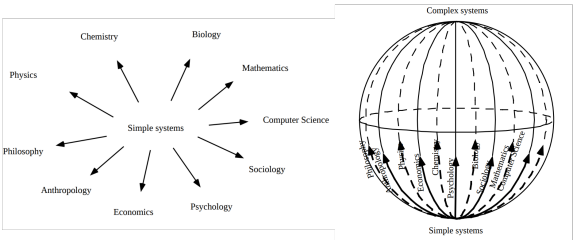

Figure 2.1: From Bar Yam book.

Cellular Automata

A *Cellular Automaton*[1], CA in short, is a simple discrete model of a single cell which changes between a discrete set of states based only on its immediate neighbors. Figure 2.2 shows an example of a *rule set*, or *transition table* for a cellular automaton with only two states, and an initial configuration, or "seed", which over several steps forms a pattern. One of the pioneers in the field of CAs, Stephen Wolfram suggested that under certain conditions CAs were capable of computation [cite Wolfram: Universality and

[1] Singular: Automaton, Plural: Automata

complexity in cellular automata]. Cellular automatas have been shown capable of solving global problems such as contour-extraction [4] where the problem is expressed as the seed and a global solution emerges from the local interaction of cells. Wolframs hypothesis of universal computation has also been proven correct, in fact even a simple 1D cellular automata such as in 2.2 has been proven sufficient to simulate a turing machine [Cite Matthew Cook], but as Sipper puts it: "This is perhaps the quintessential example of a slow bullet train: embedding a sequential universal Turing machine within the highly parallel cellular-automaton model."

In figure [figur av class 1 - IV CA] four cellular automatas are shown, each classified per Wolframs CA classification. The fourth class which "yields complex patterns of localized structures" shows complex behavior which Wolfram suggested was the sort of behavior necessary for computation to occur spontaneously. As a model for neurons it is sufficient to know that they are capable of computation and under which circumstances rather than knowing exactly what they are computing. In Langton's pioneering paper *Computation on the Edge of Chaos* [3] the space of possible transition tables for one dimensional cellular automata with two states, such as 2.2 is explored. What Langton discovered was that the ratio between states leading to life and death (black and white cells respectively) was analogous to how temperature causes phase changes in materials. This is not suprising, as one would expect rulesets where most transitions lead to death will lead to a stationary system analogous to ice, while a transition table favoring life and death equally leads to a chaotic system analogous to a gas. What was suprising however, was that at the edge between periodic and chaotic there existed a *Critical Phase*, shown in 2.3. This effect also happens in materials in what is called a 2nd order phase change, in which the material is neither gas, liquid or solid, but exhibits all three in local substructures that can be found on all scales, from macroscopic to microscopic. The takeaway from the model of cellular computation is the idea that there are certain criterias that enable computation to occur spontaneously, and that neurons actively attempt to reach this critical phase.

2.2 Evolution In Materio

Modeling neurons as cellular automata gives an idea of how neurons organize themselves into computationally capable networks and how this computation may look like, but the question of how to make sense of these dynamics and utilize them. Before tackling neurons it is necessary to extend our model to the physical realm and develop a model that can be extended to perform computation. A very natural extension of the CA model is *material computing* in which the atoms that make up matter locally interact in similar ways to CAs. Two pioneers in this field were the british duo Pask and Beer which studied how unstructured matter could be used to perform computational tasks. In one experiment [cite ???] the duo used silver in an acidic solution which would form short-lived silver filaments when subjected to electric currents. By tuning the various knobs controlling parameters such as voltage this solution could be made to perform the task of tone discrimination, proving that unstructured matter could in fact perform computation. In Langtons CAs the goal was not to compute, but to explore what conditions were necessary for computation to occur, while Gordon and Pask sought to actually compute by observing how the matter reacted to sound and tuning the parameters in response. Tommaso Toffoli argues that

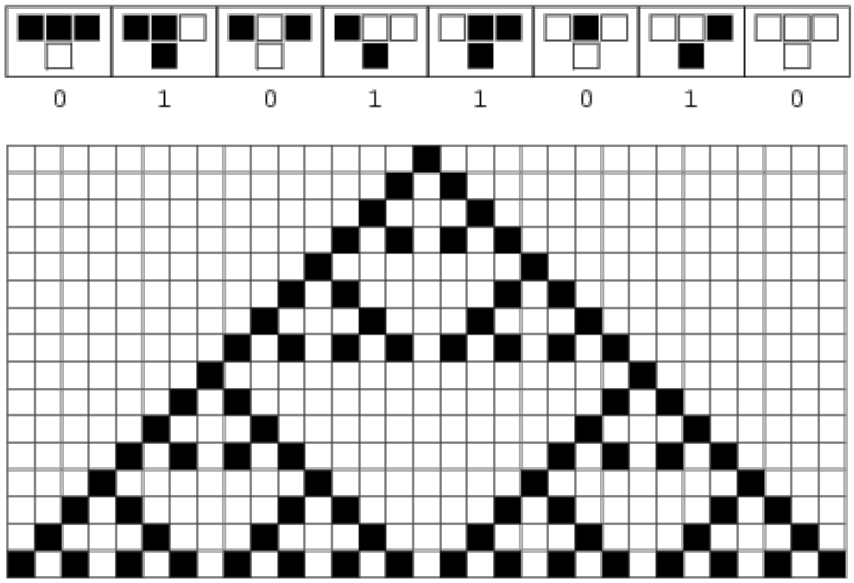

Figure 2.2: Should add a time axis to drive the point home that these are 1D

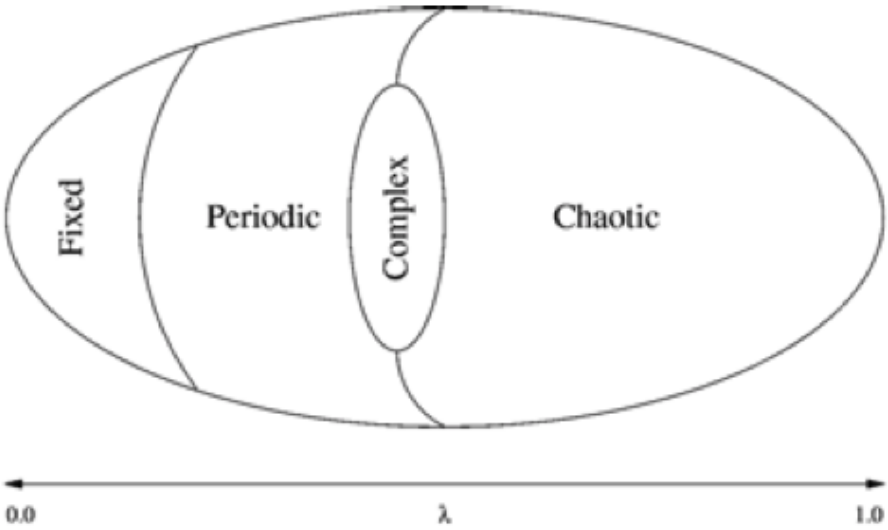

Figure 2.3: Langtons egg

"Nothing makes sense in computing except in the light of evolution" in his eponymous paper [cite TT], which separates Langtons compute capable CAs and the tone discrimination which had been evolved by repeatedly observing its performance and altering the parame-

ters accordingly. A corollary of the turing completeness of cellular automatas is that CAs can be "tuned" in a rather heavy handed way to compute, but a more interesting conjecture is that a CA with a random seed will eventually exhibit local self-replicating behavior, spontaneously giving rise to what Toffoli would regard as computation, or possibly even as life.

Whereas the pioneers in material computing sought to perform specific tasks with their experiments more recent work has been made for general computation. One approach is *Evolution In Materio* where various materials have their parameters tuned algorithmically to perform a transformation between input and output. A recent effort is the NASCENCE project which has developed a physical device, "mecobo" that can host a variety of materials which can then be interacted with using an array of electrodes. By using the same basic loop as the early material computing reasearchers the Mecobo board has been used to create XOR logic gates using unstructured carbon nano-tubes [Cite Odd paper 1].

TODO: mer her? fikse citations.

2.3 Neurons As Computers

It is tempting to reuse the same methodology employed in EiM for neurons but this line of thinking neglects the fact that neurons have already been shaped by the process of evolution over billions of years. Neurons are not unstructured matter, that can be coaxed into computing by tweaking parameters, rather they are the result of an EiM billion years in the making. While directly modifying the property of matter a la EiM is not a good fit for neural cultures there is still many clues on how to approach neurons: When Pask and Beer tweaked their silver solution to discriminate tones, they did viewed the solution as a black box, and only its performance on the task at hand was evaluated. Just like Pask and Beer did not consider the exact inner workings of silver filaments, the monumentally complex chemical pathways and genetic regulation that govern the growth and functionality of neural networks can be abstracted away. On a more theoretical level, if we imagine that the neuron were the result of an EiM experiment, what sort of behavior did the experimenters target when varying the parameters? While this is not the case it does...

Neurons

The neuron, or nerve cell, is the basic building block of both the human brain and the nerve system. There are many types of neurons in the human body, but to reduce the scope the focus of this thesis is a simplified model of the neurons that make up the brain. The model neuron, shown in fig [a figure of a neuron] consists of three parts: The body, *Soma*, the *Dendritic network* and an *Axon*. The dendritic network acts as a receptor sensing electrical activity around the neuron, while the axon transmits electric pulses to neighboring cells. The connection between two neurons is called a *Synapse*. Axons and dendritic networks are themselves vastly complex, and viewing them as simple electrical transmitters neglects the importance of neurotransmitters. However, given the strong correlation between neurotransmitter concentrations and electrical activity is considered part of the black box. When the neuron is sufficiently excited by electrical activity it will fire an electrical pulse that travels along the axon, which in turn stimulate other neurons. Together neurons form

networks in a complex interplay between topology and behavior: The behavior of the network decides its behavior, and the behavior in turn causes some synapses to wither, and others to form in a process that is not well understood. The model used for the neuron for the rest of this paper is a node in an *Adaptive Network* which communicates through electrical pulses. The missing part in our model is the underlying rules that dictates the growth of the network.

2.4 Reservoir Computing

So far we have arrived at a model of the neuron as a complex adaptive network which can be interfaced with using electrical signals, but the fundamental issue of actually utilizing neurons for computing remains. The final piece of the puzzle comes in the form of *Reservoir Computing*, a technique developed to exploit the dynamics of complex systems. If the fundamental rule governing the neuron is to create networks exhibiting the same complexity behavior as Langton's automatas then harnessing the computational capabilities of these simple models is a first step towards interfacing and understanding neurons. In reservoir computing, a complex systems is used as a *reservoir* [**?**] which "acts as a complex nonlinear dynamic filter that transforms the input signals using a high-dimensional temporal map, not unlike the operation of an explicit, temporal kernel function."
In order to explain, schrauwen makes a comparison to the the machine learning technique of source vector machines work, as shown in fig [rm 1]: The reservoir acts as a kernel, projecting input into a high-dimensional feature space. Figure [rm 1] shows this technique, note that the regression performed upon the feauture space is a simple linear regression, an important point both in SVMs and reservoir computing. Figure [rm 2] shows a typical reservoir computing setup which follows a similar method of operation as the SVM in figure [rm 1] The reservoir serves as the high dimensional feature space, while the output layer is only capable of linearly separating the resulting dynamics. Schrauwen points out two major differences between SVMs and RCs. First, SVMs only implicitly expands the input to high dimensional space in order to make the problem tractable, while reservoirs do not. Secondly, kernels are not capable of handling temporal signals. The second difference is very important, it is what allows reservoirs to implicitly encode temporal signals in their dynamics, making reservoirs a natural fit for tasks such as speech recognition. In other terms, the properties that make complex systems so hard to work with such as sensitivity to initial conditions also allow them to discern very subtle nuances in input, and their complex behavioral patterns causes the systems to change their behavior to new input based on previous input. In light of this, asking how to build a computer using Langton's automatons is the wrong question, instead the focus should be on how exploit the computation that is already occuring.
There are many examples of reservoirs which have been successfully exploited: In [**?**] an *echo state network* is utilized to solve classification problems. More esoteric reservoirs have been used, for instance in [**?**] the idea of reservoir computing is taken quite literally using a bucket of water as a reservoir.

2.4.1 Linear and nonlinear output layers

TODO: Elaborate on the use of linear vs nonlin classifiers.

Chapter 3

Making Of A Cyborg

> What is real? How do you define 'real'? If you're talking about what you can feel, what you can smell, what you can taste and see, then 'real' is simply electrical signals interpreted by your brain.
>
> Morpheus to Neo - The Matrix

The word cyborg is portmanteau of cybernetic organism, a hybrid between the biological and mechanical. TODO: Write good text here. In [Cite AHDNN] researchers created

3.1 Concept

In figure 5.1 the conceptual cyborg is shown. This conceptual cyborg is comprised of three main components: An *MEA*, short for *Micro Electrode Array* in which a biological neural network is grown. A *neural interface*, allowing two-way communication between the neural network and the outside world. A robotic body, responding to movement commands and equipped with a sensor allowing it to perceive its environment. In this concept neural readouts are transformed into a Left and Right signal by a feed forward neural network, controlling the direction of the robot. Simultaneously data retrieved from the sensors of the robot are processed in a feedback processor and fed back to the neural network. The conceptual cyborg is a closed loop system: The only input to the system is what the sensors perceive which the cyborg must act upon.

3.2 Architecture

As shown in fig 3.2, the actual cyborg adds a lot of parts compared to the initial concept. This overview focuses on the main data loop, depicted as red and green lines, with red

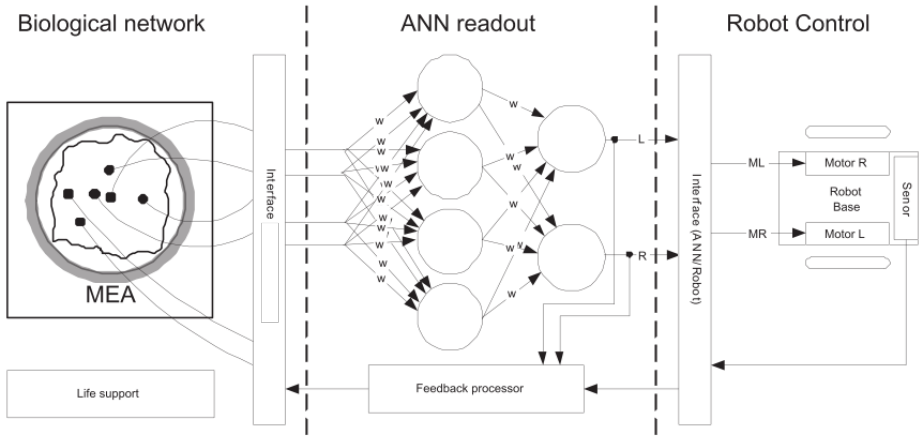

Figure 3.1: A simple conceptual cyborg.

denoting the data pipeline from the reservoir to the reservoir computer, and green from reservoir computer to reservoir. This design expands on the initial concept, fleshing out the "interface" box in 5.1 and substituting ANN with the more general notions of reservoir output filter. The expanded architecture reveals a very important detail in the final design, which is that the data loop between the reservoir computer and the reservoir extends over network. This design choice allows the neural cultures to be safely stored in a laboratory, which has ramications both in a practical and philosophical sense. By decoupling the reservoir and reservior computer, the neurons are not restricted to a single robot. In fact, the robot does not have to be a physical robot, it can be fully virtual without the neurons noticing any difference. Additionally, the reservoir computer can interface with any reservoir as long as the interfaces are adhered to, allowing other reservoirs to be used in place of the neurons, both for testing purposes, and to compare the capabilities of different reservoirs.

3.3 Wetware

As opposed to hardware and software, the term wetware describes system components of biological origin, i.e "wet" components. The wetware of the cyborg is thus the neural networks which are being grown in MEAs at the department of neuroscience located at St.Olavs research hospital. The MEAs are seeded with neural stem cells of either human or rat origin which then spontaneously form networks. At seeding there is no network at all, only a "soup" of dissociated neurons which over the course of several weeks start forming networks. The activity from these so-called pacemaker neurons can be seen in **??**. In the figure each cell in the grid corresponds to one of the electrodes as seen in **??**, however at this stage the monotonic spiking activity tends to be transient, starting and stopping randomly.

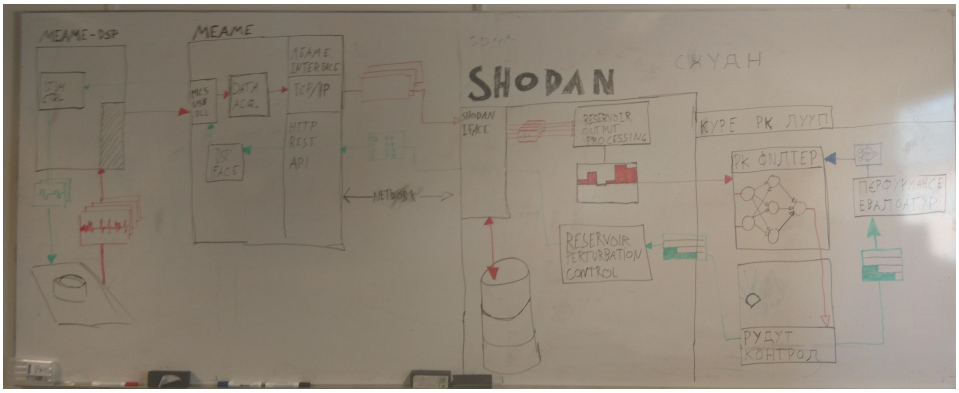

Figure 3.2: Rough sketch. Should indicate the use of TCP/IP. Should show user interface. Final version will be done in inkscape.

3.3.1 Neural Interface

The *Neural Interface* is the bridge between the biological and digital, responsible for providing a convenient API for reading out digital waveform data and inducing stimuli. Although not intended for use in close loop systems, the *MEA2100* system features the necessary hardware to implement a neural interface with the necessary functionality for a closed loop system. The MEA2100 is built to conduct in-vitro experiments electrically active cell cultures such as neurons contained in micro electrode arrays by measuring and and inducing voltages with its high precision electrodes. In addition to voltage and current the MEA2100 system comes with a life support module, allowing neural cultures to survive for prolonged experiments for up to 30 minutes. Figure **??**[rm 12] shows a physical overview of the components that make up the neural interface, which is also marked in 3.2.

Micro Electrode Array

Shown in **??**, an MEA containing a neural culture is slotted into the headstage for measurements. Not shown however is that the MEA can easily be removed, and is in fact one of many MEAs which are kept in an incubator when not experimented on. These MEAs feature 60 electrodes, which when accounting for the reference electrode provides 59 individual measurement and stimuli points, evenly spaced out in a grid. Each MEA is seeded with a neural culture, meaning that once seeded an MEA will be the host of a single culture, each capable of living for over a year, like the culture shown in **??**.

Headstage

Equipped with 60 high precision electrodes, shown in 3.3, the headstage can connect to an MEA when inserted into the measuerement bay of the headstage. Thanks to the headstage each MEA is relatively expendable as they need only provide electrodes, leaving the measuerement to the headstage.

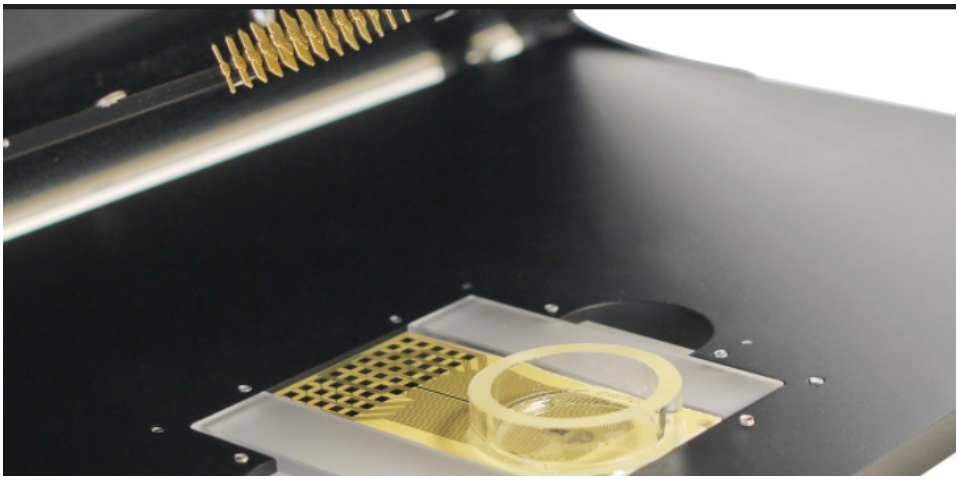

Figure 3.3: A placeholder for an open headstage showing the electrodes.

Interface board

The interface board connects to up to two head-stages and is responsible for interfacing with the data acquisition computer, as well as auxiliary equipment such as temperature controls. For this project, the most interesting feature of the IFB is a user programmable digital signal processor, which can access the raw stream of data from the headstage and issue stimuli.

3.4 Software

Creating a cyborg is a massive undertaking, thus a quite extensive software suite has been created to make research feasible. As the system architecture overview 3.2 hints, the software is the main body of work performed for this thesis. Another important detail is the strong emphasis on the dataflow which is what connect the individual components. What goes on inside a component is less relevant compared to the transformation of data, visible from the outside. Finally, the figure shows a division between two systems, MEAME and SHODAN. MEAME runs close to the neural cultures, residing on the DSP and lab computer, and is specialized to interface with the MEA2100 system specifically. SHODAN on the other hand resides on the other side of the "network chasm", and is closer to a framework, capable of handling any reservoir as long as the necessary transformations are implemented.

3.4.1 MEAME

An overview of MEAME is shown in 3.4, revealing that MEAME internally consists of two distinct subprojects exposed by a unified REST interface. While typically REST interfaces imply that a service is public, the MEAME REST interface is only intended to

be used by SHODAN. Behind the REST interface there are two modules, data acquisition and DSP interface. The data acquisition module is responsible for configuring recording parameters such as samplerate and starting or stopping recordings, while the DSP interface provides a very thin layer of abstraction for writing to the DSP memory. Both these modules are built on top of a very thin API provided the equipment vendor, making it clear that the closed loop cyborg system is pushing the equipment further than the vendor intended, for better or worse. In addition to the REST interface MEAME exposes raw TCP sockets, outputting the raw waveform data once the lab equipment has been configured. The format of the raw output is shown in figure 3.5, which must be demultiplexed by the receiver in order to separate data into individual electrode readouts, referred to as *channels*.

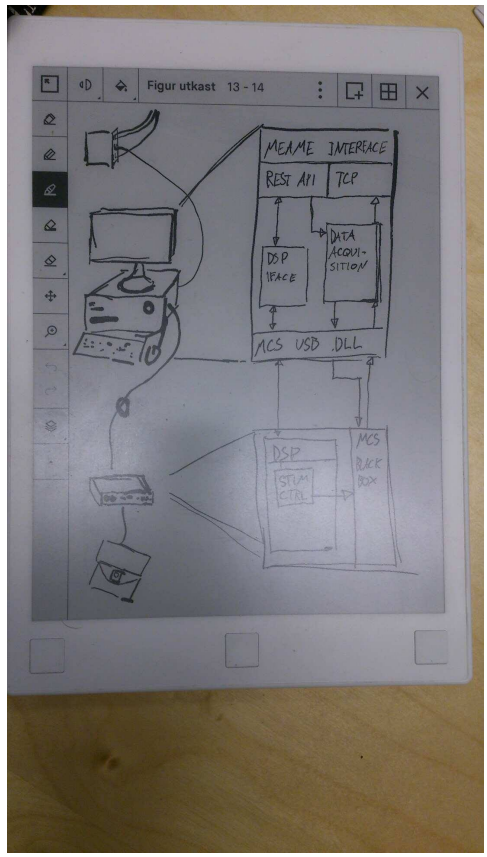

Figure 3.4: Rough sketch. nice MEAME TODO: Flesh out the DSP box with RPC handler a la MCS USB .dll

MEAME-DSP

Housed on the IFB, the DSP is not in use during normal operation and thus fully user programmable. Communication between MEAMEs DSP interface and the DSP itself is

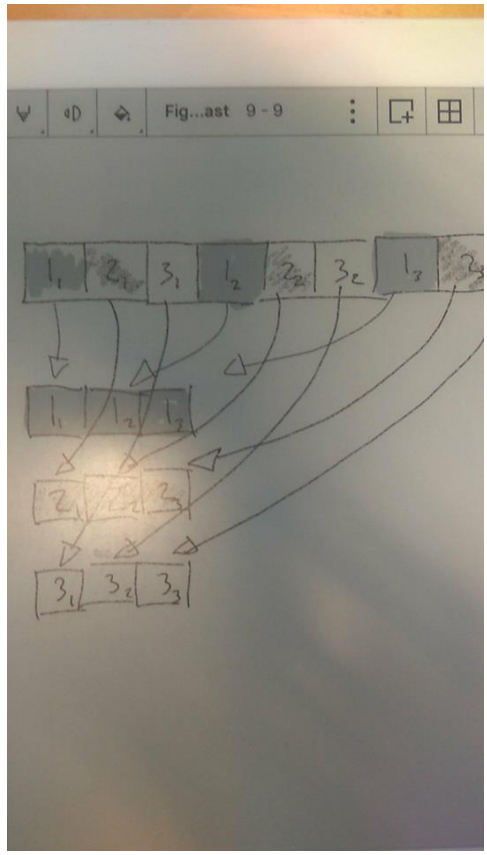

Figure 3.5: Rough sketch. Shows how data is transmitted.

done with a primitive remote procedure call system, implemented using shared memory on the DSP. When SHODAN wishes to execute a DSP procedure this is done by submitting a list of memory writes via the REST interface, and then reading a special register that is incremented when the procedure call has been executed by the DSP, indicating that a new call can be executed, and that return data is valid and can be safely read. This rather thin API is admissable because of the fact that MEAME is only accessed by SHODAN which internally translates procedure calls to memory writes and reads. The stim control module maintains a list of *stim groups* which contain a list of electrode numbers and a desired frequency. Consequentially, stimulating neurons is done by specifying which electrodes should be stimulated, and at which frequency, rather than explicitly sending a signal whenever stimuli should be applied. A visual representation is given in 3.8 showing two stimulus groups and the resulting stimulus applied. The stimulus group configuration also specifies a pattern for each group, sine and square respectively. These patterns can be uploaded by directly writing to memory from SHODAN, but for this project simple square waves are adequate.

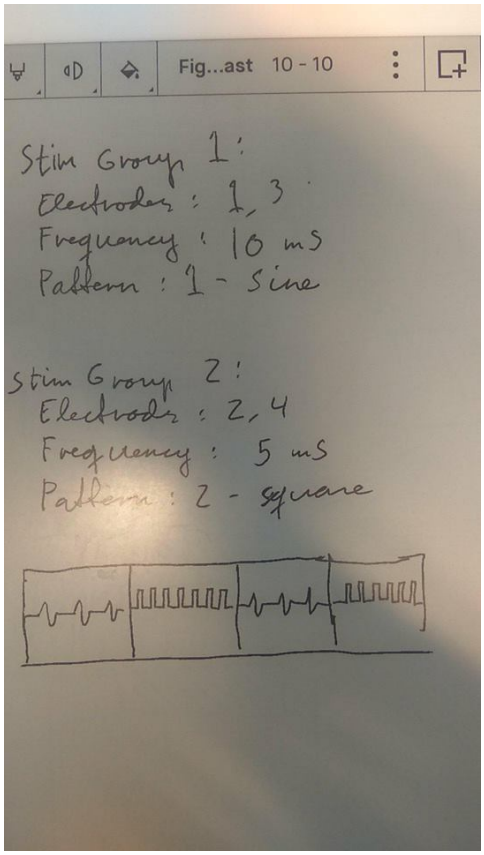

Figure 3.6: Rough sketch. Some wavez

3.4.2 SHODAN

As shown in 3.2, SHODAN is responsible for storage, filtering, generating perturbations and maintaining the core RC loop. During an experiment SHODAN can be divided into two parts, reservoir interfacing shown in 3.7, and and the core RC loop shown in **??**. The former can in turn be subdivided into input and output processing with regards to the main RC loop.

Input Processing

The MEAME comms interfacing module mirrors the corresponding interface exposed by MEAME, consuming raw waveform data over a TCP connection and configuring experiment parameters and issuing stimuli requests using a HTTP client. The channel demultiplexer splits the raw datastream from the TCP sockets into individual channels as described in 3.8 before entering the filtering module. The filtering module is responsible for translating the raw waveforms to a format that can be utilized by the main RC loop.

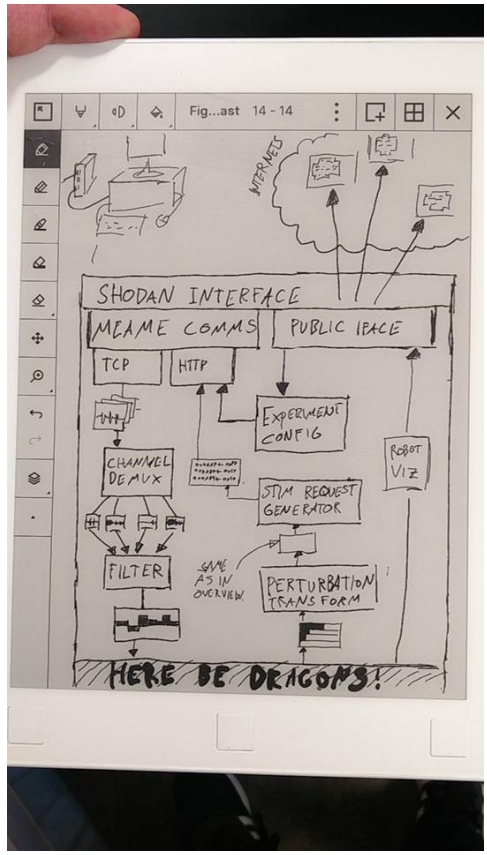

Figure 3.7: Rough sketch. Some wavez

The currently implemented filter does a basic "spike detection" transform as shown in **??**. When a certain voltage threshold is reached a spike is recorded and the detector is disabled for a short cooldown period before it can detect the next spike. The spikes are then input into a moving average filter, smoothing out the staccato spike/no spike filter output to the amount of spikes that has happened between t and $t - \Delta t$ where Δt is on the order of 100ms to 1000ms. The final input to the core RC loop is a vector of values, one for each channel that can be periodically sampled. As the data crosses the boundary to the core RC loop, its shape reveals little facts about its origin. For any other reservoir it would be just as natural to translate the output dynamics to a vector of scalars, thus for any reservoir the filter module is what translates from specific to generic.

Output Processing

Being the dual of the Input processor, the output processor does the same thing in reverse, transforming data from generic to specific. In the figure 3.7 the output of the main RC loop is shown as a list of distances, but that is merely the interpretation of data, the underlying

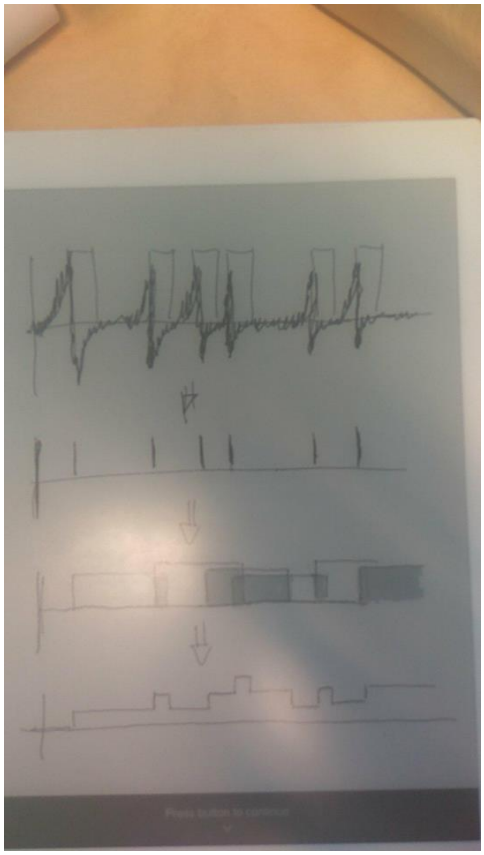

Figure 3.8: Rough sketch. Some wavez

shape is just the same as the output of the filter from the input processor: a vector of scalars. Since the goal of the project is to create a cyborg the focus is naturally on robotics, virtual or not, but the main RC loop can just as well handle other tasks such as classifying images, and the resulting output would still be expressed as a vector of scalars. Figure 3.9 shows this duality between the input and output processor: Just as the input processor masks the identity of the reservoir from the core RC loop, the output processor masks the identity of the task being performed!

The datapath of the output processor is rather similar to the input processor with the perturbation transform module serving the same purpose as the filter in reverse, turning scalars into frequency ranges. In the figure an additional module is shown, transforming a stim request into a set of memory writes for the DSP as described in the MEAME section, but this is just a practical matter added for completeness.

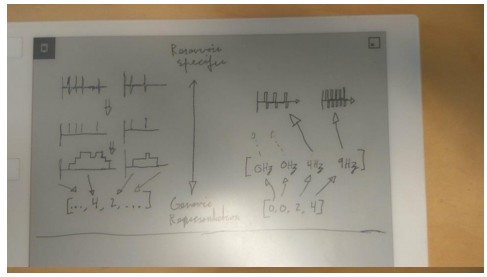

Figure 3.9: Rough sketch.

Core RC Loop

In the conceptual cyborg 5.1 all of the software described previously can be fit into the interface between the ANN and MEA. The core of the reservoir computing system is the ANN readout and the robot control module, but the concept leaves out a very important piece of the puzzle: How can the correct ANN readout layer be chosen? The Core RC loop consists of three components, the robot control, the RC filter, and the reservoir performance evaluator. Just as the robot control does not have to be an actual robot, the RC filter can employ any filter, it does not need to be an ANN. The last module is what separates SHODAN from a classical reservoir computer. In classical reservoir computing the reservoirs themselves are fairly static, exhibiting little changes in behaviour over time. A reservoir computer using a random boolean network can store an RBN in memory and access this data to reconstruct the experiment at any time. This simplifies the task of training the RC output layer since the reservoir will not change between runs. Unlike digital RBNs, neural cultures have no reset button, their behaviour changes both long term and short term. Long term, a cluster of neurons might "emigrate" away from an electrode, new connections form and old connections wither. This process happens over days and weeks, but change also happens on a short basis, both due to natural fluctuation, and because of the fact that no two experiments are truly independendent, there are no ways to make neurons forget and revert to previous behaviour. The solution to this problem is to back link the robot control to the RC filter via a performance evaluation module, creating a feedback loop capable of coping with the moving target presented by a neural culture.

3.4.3 Storage And User Interface

This section will feature the UI and a description of the database system and the sort of offline analysis that can be done. It's not that interesting, but it is significant.

3.5 Current State

MEAME and SHODAN are both still in development and are far from complete. All functionality described in this chapter has been implemented and is in a running state, but the stimuli control module that is responsible for applying voltage to the electrodes does not work correctly. This means that the closed loop system does not work at all until the

issue can be fixed. This frustrating issue reflects how the closed loop system goes far beyond the intended use for the lab equipment, which is marred by poor software and nonsensical documentation, meaning development is often a process of reverse-engineering. While this flaw means the system does not fulfill its purpose, it's enough to verify that the remaining modules work. Recordings can be made and played back, which means that MEAME/SHODAN is already able to provide a better storage solution than the vendor provided solution.

Chapter 4

Experimental Setup

Because of the issues with stimuli generation the experiment setup has not been employed on live neural tissue. As a consequence, the current experimental setup is designed to be easy to modify whenever testing becomes possible and to verify functionality of the rest of the system. Unlike traditional reservoir computers there are two orthogonal sets of parameters to explore which can be divided into neuron-specific and RC-specific as shown in figure **??**. While it is likely not the case, in order to reduce complexity these two paratemer sets are viewed as independent which lets us fix one set of parameters while exploring the other.

4.1 Tasks

Central to each experiment is the task, consisting of a problem for the reservoir computer to solve and an evaluation function to gauge how well the problem was solved. The main task chosen for the reservoir computer is a simple wall avoiding game, shown in figure **??**

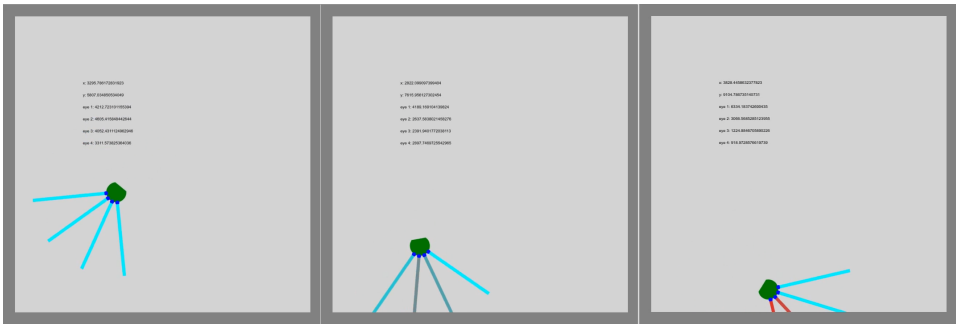

Figure 4.1: A simple conceptual cyborg.

Chapter 5

Conclusion

In short, it says you talk like a fag and your shit's all retarded.

Figure 5.1: Says here you talk like a fag and your shits all retarded.

Bibliography

[1] , ???? Marvin Minsky - Wikiquote.
URL https://en.wikiquote.org/wiki/Marvin_Minsky

[2] Fernando, C., Sojakka, S., Sep. 2003. Pattern Recognition in a Bucket. In: Advances in Artificial Life. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 588–597.
URL https://link.springer.com/chapter/10.1007/978-3-540-39432-7_63

[3] Langton, C. G., Jun. 1990. Computation at the edge of chaos: Phase transitions and emergent computation. Physica D: Nonlinear Phenomena 42 (1), 12–37.
URL http://www.sciencedirect.com/science/article/pii/016727899090064V

[4] Sipper, M., Jul. 1999. The emergence of cellular computing. Computer 32 (7), 18–26.

[5] Vardi, M. Y., ???? Artificial Intelligence: Past and Future.
URL https://cacm.acm.org/magazines/2012/1/144824-artificial-intelligence-past-and-future/fulltext

Appendix

My appendix was surgically removed. Take that, evolution.