University of Hertfordshire

Evolving Spiking Neural Networks for Animat Foraging and Pattern Recognition

Chama Bensmail

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Department of Computer Science

Supervisors:
Prof. Borys Wróbel
Prof. Volker Steuber

July 2025

Abstract

The ability to process temporal information is essential for both biological and artificial systems, especially in noisy or constantly changing environments. This work explores how evolving spiking neural network (SNN) controllers can solve nontrivial pattern recognition tasks while foraging in a 2-D world. Using a genetic algorithm and adaptive exponential integrate and fire neurones, animats evolved to respond effectively in real-time and make accurate decisions under varying levels of neural noise.

The research is organised in three experimental phases: (1) Evolution of foraging animats that incorporate basic temporal signals paired with associated rewards/punishments. (2) Evolution under variable input intervals that requires some form of temporal memory. (3) Evolution with internal noise on voltage to evaluate robustness. Performance under varying conditions was evaluated by analysing the existence of neurotopological characteristics such as self-excitatory and recurrent motifs that emerged as a result of the temporal processing mechanisms of the animats.

The results support the hypothesis that neuroevolution can help uncover biologically plausible topological features that increase precision in timing, decisional robustness, and generalisation. The results also highlight the extent to which internal noise in networks affects the topology that maintains optimal temporal recognition.

This study helps to understand how complex temporal recognition tasks can be accomplished using SNN architectures. It also provides some methodological suggestions on robust design principles for neuromorphic and autonomous systems.

Acknowledgements

I would like to thank my supervisors, Volker Steuber and Borys Wróbel, and Adam Mickiewicz University for developing the GReaNs platform, which shaped the core of this research.

This work would not have seen the light of day without the support of my family and friends. I thank my mother, my brother, and my close friends Carl Asprey and Waleed Hamra for their constant presence.

During the two years leading up to my viva, I faced significant personal adversity. I am grateful to Rotherham Metropolitan Borough Council for helping me navigate homelessness while caring for my autistic son. I especially thank the Early Help service, and in particular Joanne McLaughlin. I also want to acknowledge Sarah Reed, whose support was instrumental in helping me leave domestic abuse; Vie Clerc, who advocated for me at critical moments; Michelle Cleary from Happy Kids, who ensured my son could remain in nursery despite financial hardship; and Mrs. Webster and Mrs Platts, my son's teachers. My thanks also go to The Aire Centre, and especially Florence Powell, for guiding me through the legal aspects of my immigration situation.

I am grateful for the charities that helped me navigate extreme circumstances: Rotherham Rise, Changing Lives and Women's Aid.

My deepest gratitude goes to my son Jade, whose resilience, joy, and presence have been a constant source of motivation.

I am also thankful to my examiners for their careful reading and thoughtful discussion during the viva. Their insights have strengthened both this thesis and the way I think about my own work.

I am also grateful to the School of Physics, Engineering and Computer Science at the University of Hertfordshire for providing the academic environment in which this work could develop, as well as Student Wellbeing for their support through therapy.

Contents

Chapter 1

Introduction

1.1 Background and Context

In nature, biological organisms perform difficult tasks such as recognition of temporal patterns with ease, even when working under complex and noisy conditions (Jones and Teeling 2006; Maass 1997). From the perspective of neuroscience and AI, the question of how a nervous system exhibits such consistent performance is fascinating. Unlike biological neural networks, traditional ANNs can outperform biological systems in static pattern recognition, but due to difficulties with temporal processing, noise handling, and reliance on rate-based coding and nonspiking architectures, they fail to deliver in dynamic environments (Maass 1997; Goodfellow et al. 2016).

Using pulses to convey information makes Spiking Neural Networks (SNNs) more closely resembling biological neurons and, therefore, features greater potential (Izhikevich 2003; Gerstner and Kistler 2002; Ponulak and Kasiński 2011; Naud et al. 2011). The event-based computation that occurs within SNNs, along with the inherent temporal dynamics, equips them with the capability needed for tasks that require precise timing, past input memory, resource conservation, and strong temporal dynamics (Indiveri and Horiuchi 2011; Brette and Gerstner 2005).

On another front, attention has been drawn to the use of evolutionary algorithms to generate the parameters and structure of a neural network, particularly for problems that are difficult to solve using manually designed architectures or hand-tuned parameters (Yao 1999; Floreano and Mattiussi 2008; Stanley and Miikkulainen 2002). Through neuroevolution, networks are equipped with the ability to autonomously organise topologies and adapt their behaviours, sometimes yielding innovative but solid solutions.

This work examines the evolution of spiking neural network controllers for animats trained to forage based on sequential temporal patterns (Bensmail, Steuber and Wróbel 2018; Abdelmotaleb et al. 2014). The focus is on the robustness of the internal mechanisms such as noise filtering and self-induced excitatory dynamics, as the system works within varying environmental conditions through a temporal input sequence.

Precise temporal processing is important in a number of biological systems because they need to operate in dynamically and perturbed environments (Jones and Teeling 2006; Carr and Konishi 1990). Consider, for example, echolocating bats that have to differentiate between rapidly

repeating echoes to locate prey while contending with cluttered environments and internal processing noise. Social communication and mating behaviours in songbirds are also dependent on the accurate temporal decoding of patterns in the soundscape (Singer 1999; Laurent 1999a). In both situations, the ability to capture relevant temporal information in the presence of stochastic disturbances is critical for survival. Understanding how similar abilities can develop in artificial spiking systems not only enhances understanding of biological cognition but also helps in the design of more sophisticated autonomous agents to function in unpredictable real-world environments (Yaqoob et al. 2023; Yao 1999; Bensmail, Steuber and Wróbel 2018).

1.2 Problem Statement

Even after the advancement of neuroevolution and spiking neural networks, the role of emergent topological structures in relation to temporal cognition in embodied systems has to be sufficiently resolved (Yao 1999; Floreano and Mattiussi 2008). Other researchers focus on static tasks that involve classification or locomotion strategies in less demanding spatial environments that require minimal memory and timing resources (Stanley and Miikkulainen 2002; Jin et al. 2010; Izquierdo et al. 2008). The sophisticated demands of temporal pattern recognition, especially in noisy and ever-changing scenarios, have not yet been fully investigated in the context of evolved spiking controllers (Bensmail, Steuber and Wróbel 2018; Abdelmotaleb et al. 2014; Yaqoob et al. 2019).

While some recurrent and self-excitatory features are known to exist in biological networks (Markram et al. 1997; Turrigiano and Nelson 2000), their purpose and function as artificial structures within spiking networks evolved for embodiment tasks remains largely unexplored (Yaqoob et al. 2023). In particular, the combination of network topology and decision making that merges uncertainty and noise internally has not been studied thoroughly when focussing on temporal foraging behaviour.

This work attempts to resolve these issues by evolving spiking neural networks that control animats using a series of temporally separated sensory signals. The goal here is to understand how some topological motifs, like self-excitatory loops and recurrent pathways, aid in temporal decision making during real-time interactions of an embodied agent by gradually increasing the complexity of the internal-noise-conditions task.

Previous attempts have mostly focused on spatial navigation tasks or static classification benchmarks while overlooking temporal decision making that involves uncertainty. For example, evolved spiking neural networks (SNNs) have been effectively implemented in locomotion tasks such as maze navigation and obstacle avoidance, where timing demands are low, but spatial memory is essential. To my knowledge, few studies have directly addressed the evolution of embodied agents' ability to temporally discriminate or their robustness to internal neural noise.

1.3 Research Questions and Objectives

This thesis has been framed around the following research questions:

1. RQ1: In what ways do spiking neural networks with their decision-making capabilities incorporated into animat bodies encode temporal patterns for intricate temporal decision-making processes?

2. RQ2: What topological forms result from the evolutionary processes associated with the development of SNNs designed for temporal pattern recognition?

3. RQ3: In what way do self-excitatory loops and other topologically emergent features affect the robustness of animat decision making under the influence of high internal noise levels, task complexity, and other built-in factors?

4. RQ4: How well can SNNs evolved through SNN controllers perform across varying varying conditions such as variable silent intervals and different levels of noise?

The corresponding objectives of this work are as follows:

- Evolve spiking neural network controllers that can perform foraging tasks with defined time windows in changing environments.

- Conduct detailed studies on the evolved networks on their topological features and how these relate to the defined performance indicators.

- Determine to what degree evolved controllers may be subjected to internal neural noise and externally induced changes.

- Study the development of structures that exhibit biological realism and that enable flexible temporal reasoning.

1.4 Methodological Approach

In an effort to answer the specific research questions listed, this thesis applies a neuroevolutionary strategy that uses genetic algorithms to evolve spiking neural networks (SNN) with The Adaptive Exponential Integrate-and-Fire (AdEx) models (Brette and Gerstner 2005; Naud et al. 2008). The biological evolution optimally designs the synaptic values in conjunction with the network topology so that more sophisticated and biologically plausible forms can arise.

We preferred AdEx neurons over simpler models like leaky integrate-and-fire (LIF) units due to their more intricate intrinsic dynamics (Izhikevich 2003; Naud et al. 2011). While our networks were not explicitly driven with constant input to explore bursting or adaptation regimes, the potential of AdEx to exhibit such biologically relevant behaviours (e.g., spike-frequency adaptation and bursting) made it an attractive choice (Brette and Gerstner 2005; Naud et al. 2008). While we did not make a direct comparison to LIF, other work (Naud et al. 2008; Brette and Gerstner 2005) assumed these dynamic properties might enable the formation of robust internal states and temporal strategies under evolutionary pressure . Evolved networks were able to resolve non-trivial tasks that involved temporally patterned stimuli when we employed AdEx neurons, which

attests to the model's credibility for our aims (Bensmail, Steuber and Wróbel 2018; Yaqoob et al. 2023).

The animats are placed in 2D worlds where they must collect food items using temporally distinct sensor cues (Bensmail, Steuber, Davey et al. 2017; Wróbel and Joachimczak 2011). Each animat has a set of discrete sensors and motor outputs. The entire behaviour of the animat is autonomously driven by an internal spiking network. The controller has to manage the temporal resolution of signals and requires making optimal prey capture plans in real-time, often with significant temporal uncertainty.

Evolutionary studies are stratified by difficulty with progressive increases in the levels of each task.

- Simple Temporal Pattern Recognition: Animats have to learn to match a single stimulus to a temporally shifted reward (building on the work in the IEEE 2017) (Abdelmotaleb et al. 2014).

- Complex Signal Separation Tasks: Challenges animats' ability to resolve input sensory signals with varying gaps using their temporal ordering (expanding the ICANN 2018 framework) (Yaqoob et al. 2019).

- Resilience With Respect to Internal Noise: Additional spiking neuron activities incorporate internal randomness to the implemented models, assessing the resilience and adaptability of evolved topologies (Faisal et al. 2008; Stein et al. 2005; Destexhe and Contreras 2006)(See Chapter 6).

The performance assessment is carried out by evaluating foraging success, timing precision, and overall resilience to noise.

Emergent network topologies are analysed after evolution using graph theory clustering methods to measure their functionality against self-excitation/inhibition and recurrence: Unsupervised clustering, path length, and correlation analysis.

With this range of empirical studies provided, this thesis focusses on the evolutionary forces and processes that enable temporal recognition in the context of embodied agents driven by spiking neural networks.

1.5 Contributions

Below are the primary contributions of this thesis that focus on the areas of spiking neural networks, neuroevolution, and embodied artificial intelligence:

- Evolved Temporal Recognition Mechanisms in SNNs:

  This thesis provides a physical illustration of temporal recognition in embodied agents with the results from evolving spiking neural networks for foraging tasks. The results allow for a reference for other models of the timing elements of decision making in automated systems,

especially in dynamic and uncertain environments (Bensmail, Steuber and Wróbel 2018; Abdelmotaleb et al. 2014; Yaqoob et al. 2019).

- Study of the Self-Exciting/Inhibiting Features of Topology and Their Functional Characterisation: This thesis analyses the graphs that emerged with self-excitatory/inhibitory loops and the topological features of the evolved networks. These features were found to improve the precision and reliability of temporal recognition after functional evaluation of time accuracy and noise resistance (Kashtan and Alon 2005; Sporns et al. 2004).

- Cross-Platform Deployment and Embodied Validation: This thesis shows the cross-simulation back-end deployment of evolved SNN controllers into real-world settings, including physical validation with a robot, as well as using GPU-accelerated simulations with GeNN. These findings emphasise the adaptability and biological significance of evolved networks, thus validating their functionality in both computational and embodied scenarios.

- Assessing Generalisation of Learning Over Time: We evaluated the broad adaptability and flexibility of the networks against training constraints by applying trained controllers to different temporal gaps and novel signal shapes. Such results demonstrate the evolutionary routes through which spiking networks could reach to form modifiable temporal strategies.

- Neuroevolutionary Methodology for Temporal Recognition: The focus of this methodological contribution is the evolution of synaptic connections and the network topology simultaneously with AdEx neurones. This method makes possible the emergence of temporal recognition strategies without any designed spatial structure, enabling networks to self-organise and address complex problems without externally imposed architectural constraints.

These undertakings help explain the evolution of spiking network architectures evolved to achieve advanced tasks in temporal recognition while simultaneously enabling the development of resilient systems that function in the presence of biologically significant rates of noise and changes.

1.6 Thesis Structure

The dissertation consists of seven chapters that will be discussed in the following format.

Chapter 1: Introduction Outlines the problem questions of the research, the objectives, methodology, and results.

Chapter 2: Review of Related Literature Covers spiking neural networks and neuroevolution methods with some focus on their application to embodied agents concerning temporal recognition and decision-making processes under noise.

Chapter 3: Methodology Details the experimental framework, which consists of the animat environment, the model of neurons, genetic algorithm, the criteria of fitness testing, and topological analysis techniques.

Chapter 4: Evolution of Basic Temporal Foraging Animats Describes the experiments conducted on simple time-dependent foraging tasks with animats and discusses the corresponding results achieved during the study to outline a benchmark for temporal decision-making ability.

Chapter 5: Evolution of Animats with Variable Temporal Intervals Studies the evolution of SNN controllers in relation to the increasing difficulty of the task where silent intervals between the sensory signal are set to vary during evolution.

Chapter 6: Robustness and Topological Analysis under Internal Noise The effects of internal neural noise on the evolved controllers are studied, the resulting network is analysed with respect to the structure, and the degree of generalisation is examined across various spatial and noise conditions.

Chapter 7: Cross-Platform Validation and Physical Robot Deployment of SNNs Presents the results of cross-platform simulation and the deployment of evolved SNN controllers in a physical robot, demonstrating real-world applicability and consistency across computational environments.

Chapter 8: Conclusion and Future Work Provides a comprehensive overview of the work done, challenges encountered, and insights for further research.

Chapter 2

Literature review

2.1 Introduction to Spiking Neural Networks (SNNs)

Spiking Neural Networks (SNNs) represent a more advanced type of neural network, based on the electrophysiological characteristics of biological neurons. Unlike conventional artificial neural networks (ANNs) and their complex subcategory, deep neural networks (DNNs), SNNs seek to emulate real-life nervous system dynamics, such as information processing through time-dependent spikes rather than the continuous activation functions in ANNs. Although not applicable in this study, SNNs can implement biologically inspired learning mechanisms, for example spike-timing-dependent plasticity (STDP). These features make SNNs capable of recognising temporal patterns during real-time processing, which is of increasing interest in both neuroscience and advanced computational models.

2.1.1 Fundamental principles

An important distinction is that SNNs are capable of reproducing various tonic spiking, bursting, and chattering neural firing patterns observed in biology. The adaptive exponential integrate-and-fire (AdEx) model, which is used in many SNN models, is capable of reproducing all of these behaviours and more, depending on how its parameters are set (Brette and Gerstner 2005; Naud et al. 2008). This flexibility is in stark contrast to the simpler perceptron models which do not possess temporal dynamics at all. This could explain why SNNs are most useful for simulating processes involving rhythm and timing such as motor control and sensory processing.

2.1.2 Advantages of SNNs

Compared to conventional artificial neural networks, SNNs are advantageous for handling temporal dynamics, biological realism, and real-world sensory data.

- **Processing of Temporal Information:** As a result of the spike timing property, SNNs are more effective at processing temporal patterns. They are ideal for performing tasks that involve time-dependent signals, for example, speech and visual processing (Thorpe and Gautrais 1998).

- **Energy Efficiency:** SNNs event-driven and sparse structure allow them to be more energy-efficient. Unlike traditional networks that are continuously activated, SNNs emit spikes only when necessary. This makes them ideal for neuromorphic hardware implementations (Indiveri and Horiuchi 2011).

- **Biological Realism:** SNNs capability to model wide-ranging temporal dynamics and neural firing patterns provides more biologically realistic simulations of sensory and motor functions.

- **Noise Robustness:** Spiking neurons, combined with dynamic thresholding, show strong resilience to input noise (Maass 1997; Brette and Gerstner 2005), making SNNs more robust in real-world conditions.

These benefits enhance SNNs abilities, pushing them further beyond traditional ANNs for simulating biological processes and solving tasks that are temporal, noisy, or energy constrained.

2.1.3 Categories of Spiking Neuron Models

Spiking neuron models vary in complexity, biological plausibility, and computational efficiency. The choice of model often depends on the intended application; from biological realism to hardware implementation.

- **Leaky Integrate-and-Fire (LIF):** One of the simplest and most widely used models. It integrates input currents over time and fires a spike when the membrane potential crosses a fixed threshold, then resets. While efficient and easy to simulate, it lacks key biological dynamics such as adaptation (Gerstner and Kistler 2002).

- **Hodgkin–Huxley model:** A biophysically detailed model based on ion channel kinetics. It accurately reproduces the generation and propagation of action potentials but is computationally expensive and less suited for large-scale simulations or real-time applications (Hodgkin and Huxley 1952).

- **Izhikevich model:** A two-dimensional model that deliver a balance between biological realism and computational efficiency. It reproduces a wide variety of spiking behaviours with low computational cost, making it suitable for large-scale neural simulations (Izhikevich 2003).

- **Adaptive Exponential Integrate-and-Fire (AdEx):** Also a two-dimensional model, AdEx captures essential neuronal dynamics such as spike-frequency adaptation and burst firing. Compared to Izhikevich, AdEx is not dependent on a fixed spike-detection threshold and has been shown to closely match actual neuronal recordings. It is also ideal for neuromorphic hardware implementation and could be more so than both Izhikevich and LIF models in some platforms (Brette and Gerstner 2005).

2.1.4 Comparison with Traditional Neural Network Models

SNNs approach of encoding and processing data makes them distinct from Artificial Neural Networks (ANNs) and Recurrent Neural Networks (RNNs). These traditional categories rely on continued activation function and weighted summations, whereas SNNs communicate via discrete spikes; hence they are more biologically realistic and better suited for processing temporal pattern inputs.

The high precision of SNNs in precisely encoding temporal patterns makes them better than ANNs. This is particularly useful in applications like voice recognition or sensory processing in robots that use dynamical input. The precise timing of spikes of SNNs helps to identify temporal dimensions (or axes) along which complex input patterns can be recognised (Maass 1997). ANNs on the other hand, are less effective in handling scenarios where the stationary input-output applications require real-time adaptation or temporal patten recognition.

For RNNs that can handle a sequential input thanks to their memory component and complex gating systems such as Long Short-Term Memory (LSTM) networks or Gated Recurrent Units (GRUs), this makes them more expensive energy-wise, in comparison to SNNs that can achieve the same results thanks to their ability to detect temporal sequences using spike timing, even across multi-timeframes, and therefore providing a better understanding of temporal dynamics (Roy et al. 2019).

SNNs are nondifferentiable, which means that they rely on discrete spike events that behave often like bursts and cannot be predicted following gradient-based optimisation techniques that are usually used for training ANNs and RNNs (Tavanaei et al. 2019). To overcome this challenging limitation, which is derived from the essence of SNNs, it is common to seek alternative optimisation methods, such as evolutionary algorithms or surrogate gradients.

The discrete event-driven feature of SNNs significantly reduces the energy use for neuromorphic hardware compared to ANNs and RNNs. Despite the challenges that SNNs face, the way they communicate through spikes ensures energy efficiency (Indiveri and Horiuchi 2011), making them more attractive for applications such as low power computing, edge computing, and mobile robotics.

In summary, traditional ANNs and RNNs are good at solving pattern recognition tasks and managing sequential data through continuous activations; however, SNNs offer a biologically inspired alternative capable of excelling in temporal information processing, which entail numerous advantages in energy efficiency and temporal precision.

2.2 Temporal Information Processing in Neural Systems

To manage difficult tasks such as sensory perception, motor control, and cognitive activities, biological brains rely primarily on temporal information processing to enable their neural network to encode, process, and store the information received from the environment.

2.2.1 Importance of temporal coding in biological neural networks

In biological brain, the time-based approach of representing information by the timing of spikes rather than the firing rate (how often the spikes occur) is essential because it allows efficient and fast processing in a changing environment and time varying stimuli (Dayan and Abbott 2001). This type of coding, called temporal coding, is important in tasks requiring fast and accurate responses such as vision and sound, or coordinating movement.

Research has shown that neurons can encode information in different ways, notably phase coding, interval coding or spike-timing-based coding. These temporal coding strategies allow biological systems to communicate information at high speed, making them more effective than rate-based or purely analogue, in solving complex and fast-changing environments (Singer 1999).

2.2.2 Temporal pattern recognition in different sensory modalities

Biological neural networks are very efficient in solving pattern recognition tasks in different sensory modalities such as visual, auditory, and olfactory processing.

Auditory processing

The auditory system relies primarily on temporal coding to process input sounds, especially in tasks involving speech recognition or music processing. For example, neurons in the cochlear nucleus and auditory cortex can detect precise timing differences between sound waves, pinpointing the location and rhythms of the sound (Carr and Konishi 1990). In other words, recognising tiny differences in the arrival of sound waves (between two ears) or within their own sequence allows the brain to distinguish where the sound is coming from.

Olfactory processing

The olfactory bulb is equipped with neural networks that use timing patterns to identify what an odour is and its concentration. Research has found that when neural networks are rhythmic and synchronised, it helps in understanding and encoding smell information (Laurent 1999b). This time-driven way of processing the odour allows the brain to quickly adjust and react to changes in scents.

Visual processing

To detect motion, recognise objects, and perceive changing scenes, temporal coding is indispensable in the visual system; visual cortex and retina both contain neurons that respond appropriately to temporal changes in light intensity, thus allowing detection of moving objects and changes in the visual field (Meister and Berry 1999).

2.2.3 Mechanisms for temporal information storage and processing

Some of the main sophisticated mechanisms developed by biological neural networks to store and process temporal information include:

- Spike Timing-Dependent Plasticity (STDP): Based on the precise timing of the pre and post synaptic spikes, the weight of the connections in the neural network is adjusted accordingly. This enables neural networks to store temporal patterns and adapt to changing inputs, which is a core principle in learning and memory (Markram et al. 1997).

- Reverberating Circuits: When a stimulus enters these types of circuit, it may keep looping through it several times before fading away. This leads to a self-sustaining activity of the neural system that can continue despite the initial stimulus stopping. This echo-like behaviour is why they are called 'reverberating'. These circuits are observed to be essential for maintaining short-term memory and working memory and help coordinate any periodic activity such as breathing and walking (Hebb 1949). So, temporal information is managed through these circuits because when a temporal pattern enters the circuits, it is held for a certain time, transient memory, allowing the brain to remember the pattern or sequence long enough for tasks that need short memory or attention.

- Intrinsic neural dynamics: neurons can have properties that are inherent and allow them to retain temporal information. Most known is the after-hyperpolarisation, where the membrane potential drops below the resting threshold, making it harder to trigger another spike for a short period. This helps regulate spike timing. Bursting behaviour, which manifests in rapid clusters of action potentials followed by quiescent periods, also contributes to temporal coding. These features help neurons encode and maintain patterns that depend on time precision and produce rhythmic activities. For motor functions, temporal processing tasks, and neural synchronisation, such properties are crucial (Gerstner and Kistler 2002). Using these inherent properties, neurons support the processing of temporal patterns within large-scale neural networks, as discussed by Izhikevich 2007.

2.3 Evolutionary Approaches to Neural Network Design

It is particularly challenging to optimise and design spiking neural networks because of their complex dynamic and multidimensional search spaces. Evolutionary algorithms (like the one used in this work) are a promising tool to address this challenge by using biological evolution inspired principles to optimise SNNs topologies and parameters.

2.3.1 Overview of Neuroevolution Techniques and Applications to SNNs

Traditional training methods in rate-based artificial neural networks (ANNs) use gradient-based optimisation such as backpropagation in contrast to neuroevolution that employs a set of processes such as selection, mutation, recombination, and crossover to evolve networks (Yao 1999). These

techniques often use an error/fitness function that is nondifferentiable and does not need to go through the gradient-based process to be optimised; hence, the advantage of applicability to a broader range of tasks.

Topology evolution is the process in which we evolve the graph of the network, such as the number of nodes (neurons), edges (connections) and layers (interneurons). Algorithms such as NEAT (NeuroEvolution of Augmenting Topologies) dynamically adjust both the topology and the weights of networks during the evolutionary process, allowing the discovery of more efficient network topologies (Stanley and Miikkulainen 2002).

Neuroevolution can also be used to optimise network parameters such as weights and activation thresholds while keeping the topology fixed to fine-tune the network for better performance.

Evolutionary algorithms have been used to optimise the structure and synaptic weights of SNNs, as well as the neuronal type. NEAT, for example, has been adapted to SNNs: it starts with a simple topology and gradually increases its complexity until a more complex and specialised network is found (Floreano, Dürr et al. 2008). In addition, these methods can help fine-tune crucial parameters, such as firing thresholds, synaptic gains/weights, and time constants. Since SNNs are dependent on precise timing, algorithms such as GA (Genetic Algorithm) and particle swarm optimisation (PSO) have been shown to be successful in evolving such timing-sensitive parameters (Izquierdo et al. 2008).

2.3.2 Applications of evolutionary algorithms in SNN design

Neuroevolution has been successfully applied to design SNNs capable of performing complex tasks that require temporal processing and adaptation. The evolved SNNs were integrated to enable robot control systems, allowing networks to adapt to rapid changes in environments and to learn sensorimotor coordination (Ijspeert et al. 2007), which is why evolved SNNs are most appropriate for real-time adaptation tasks, rather than ANNs that struggle to detect temporal dependencies (Jin et al. 2010).

Furthermore, evolutionary algorithms have been successful in optimising SNNs for neuromorphic hardware, where energy efficiency and adaptability are of major interest for real-world application in embedded systems (Indiveri and Liu 2015).

2.4 Animat Research and Embodied Cognition

Embodied cognition is a concept based on the idea that cognition essentially emerges from the interaction between an agent's body and its environment. The interdisciplinary field of studying and developing such agents, often referred to as animats (artificial agents designed to mimic behaviours observed in animal studies), offers significant insight into embodied cognition.

2.4.1 Concept and history of animats

The autonomous agents designed to display real-life behaviours were introduced as "animats" (a fusion between "animals" and "automation") (Wilson 1985). This field was created following an increased interest in understanding how simple behaviours emerge from the interplay of an agent and its environment without necessitating complex internal modelling.

Braitenberg vehicles (Braitenberg 1984) were used in the early stages of animat studies and showed how simple sensorimotor connections, surprisingly, lead to complex behaviours . With time, the inclusion of more sophisticated neural network models allowed researchers to discover topics such as learning, adaptation, and decision-making in dynamic environments.

2.4.2 Simulated environments and embodied cognition

To study animats, a simulated environment is essential, one that allows control of settings and where animats can interact with virtual worlds while solving structured tasks, receiving sensory inputs, and producing feedback. Such an environment is what facilitates the study of embodied cognition where we observe how an animat's body (sensors, actuators, and forces) impacts the cognitive process while interacting with a changing environment (Pfeifer and Bongard 2007). Examples of simulation platforms include Wbots, where animats are equipped with neural networks and evolved over time (Michel 2004), and Neuro-Evolving Robotic Operatives (NERO), a game-based platform for training animats in real time through neuro-evolutionary methods (Stanley, Bryant et al. 2005).

2.4.3 Animat control using neural networks

Artificial neural networks both spiking and non-spiking became central to animat control as they allow for adaptability, flexibility, and autonomous behaviour. The incorporation of these neural networks to be the brain of the animat, allowed learning from experience, adapting to changing environments and performing difficult decision-making tasks (Floreano and Mondada 1996). Evolutionary algorithms are commonly used to evolve neural networks that govern animat controllers, which lead to complex behaviours without manual tuning typically required in gradient-based optimisation methods. Such behaviours were explored by training these networks to solve tasks such as obstacle avoidance, goal-directed behaviour to reach specific targets, and adaptive behaviours that adjust to environmental changes such as shifting terrain or moving targets. This combination of neural networks with animat studies allowed us to advance our exploration and understanding of embodied cognition, which has direct applications in robotics, artificial intelligence, and neuroscience (Floreano, Dürr et al. 2008).

2.5 Biologically Inspired Robotics

Biologically inspired robotics uses principles that we observe in real organisms to create controllers for robotic systems capable of being adaptive, flexible, and behave efficiently. This area of research

tends to incorporate neural networks into controllers hoping to mimic biological nervous systems, this way robots are more likely to move and react with minimal reality gap.

2.5.1 Neural controllers for robotic systems

In the robotics field, neural controllers are inspired from biological nervous systems and often use SNNs to model the dynamic interactions of a robot with its surroundings. SNNs are particularly suitable thanks to the efficiency with which they process temporal patterns embedded in the perceived sensory information in real time (Ijspeert et al. 2007).

Central pattern generators (CPGs) are neural circuits that produce rhythmic patterns without the need for sensory feedback; they are a common method used in biologically inspired robotics. These circuits have been used to control robot locomotion, mimicking walking, swimming, or crawling behaviours observed in animals (Ijspeert et al. 2007).

2.5.2 Case studies in bio-inspired robotics

RoboLobster: this robot uses sensors inspired by biological chemoreceptors, similar to the sensory mechanisms of lobsters. This allowed Robolobster to navigate underwater and locate chemical sources (Ayers 2004). Such bioinspired robots were designed to help understand how lobsters process sensory information for navigation and orientation.

Bipedal robots with neural controllers: equipped with neural networks as controllers, these robots could walk while maintaining balance and adapting to changing terrain. Researchers have used neuroevolutionary algorithms to teach bipedal robots ways of walking resembling human walking patterns (Reil and Husbands 2002).

2.5.3 Neuromorphic Hardware and SNN Implementation

Neuromorphic hardware has a unique ability to mimic the structure and mechanisms of biological neural systems at the hardware level, which is what SNNs need for their implementation. Similarly to the way brain neural networks process information, neuromorphic chips run in large-scale parallel and event-driven manner (Indiveri and Liu 2015), making them completely different from GPUs and CPUs.

Chips such as IBM's TrueNorth, Intel's Loihi, and SpiNNaker have proven a great ability to use lower power compared to traditional hardware. This is one of the primary advantages of using neuromorphic hardware for SNNs. The TrueNorth for instance, can simulate more than a million neurons and consume only a few watts, making it capable of handling real-time processing in robotics and sensory networks (Merolla et al. 2014). On the other hand, GPUs use energy that is orders of magnitude higher to run deep learning models.

GPUs and CPUs would struggle with scalability since speed and performance drop as the size or complexity of the SNNs increases. This is mainly due to their sequential processing nature or limited parallelism. In contrast, the inherently parallel nature of neuromorphic chips allows them to handle multiple computations simultaneously, like a human brain, without degradation

in performance when processing large SNNs (Furber et al. 2014). This is particularly important in tasks that rely on real-time adaptation, such as sensory processing or autonomous navigation.

Intel's Loihi chip was developed to run SNNs capable of performing gesture recognition with low latency and energy consumption. This is because neuromorphic hardware platforms only compute when spikes are triggered, making processing sparse and efficient (Davies et al. 2018).

The challenges with the integration of SNNs in neuromorphic hardware are still significant despite all the great advantages, and this is primarily due to the lack of a standard software for developing and training SNNs, thus limiting the adoption of these chips by a wide community. Moreover, neuromorphic hardware is still an in-progress technology, and researchers are still trying to optimise the interaction between SNN algorithms and hardware for more efficient and scalable implementations (Schuman et al. 2017). These challenges do not diminish the promising future that aims at developing more energy-efficient and biologically inspired computing systems. There is growing interest and it is expected that neuromorphic hardware will be a pillar in advancing SNNs and their application from robotics to intelligent sensory networks.

2.5.4 Case Studies and Real-World Applications

SNNs have shown significant success in handling various and complex real-world tasks in fields such as robotics, sensory processing, and autonomous systems.

- Robotics: A well known example is the implementation of SNNs as controllers of legged robots which allows for generating rhythmic locomotion patterns. In a work by Ijspeert et al. 2007, SNNs were used as controllers of an animat that looked like a salamander, and could effectively switch between walking and swimming based on sensory input. This animat showed adaptation to its gait as a reaction to changes in the environment, which confirms again the effectiveness of SNNs in real-time motor control.

- Autonomous Driving: Sensory data processing SNNs have been used for autonomous driving tasks to help vehicles monitor their changing surroundings in real-time. Researchers have used SNNs to process data from event-based cameras (these cameras mimic the human eye), so they do not capture all the frames like normal cameras, but instead detect changes in the environment (when something moves). In a study by Lagorce et al. 2017, an SNN used visual data from an even-based camera to detect obstacles in autonomous vehicles and was found to be faster while consuming less energy than traditional cameras. This makes SNNs suitable for processing data in real time while driving in changing environments.

- Temporal Pattern Recognition: SNNs outperformed conventional neural networks in speech recognition. Temporal Spike Pattern Learning Rule (TSLR) was used to train SNNs with the objective of recognising spoken digits with high precision in noisy environments (Mozafari et al. 2019). This shows that SNNs are more suitable for applications that require robust temporal pattern recognition such as real-time speech-to-text systems.

- Sensory feedback integration: The use of SNNs in neuroprosthetics has shown significant success in enabling natural control of limbs thanks to sensory feedback loops. For instance, Spüler et al. 2015 has shown that using SNNs to process and evaluate tactile feedback signals from a prosthetic hand allowed users to sense different levels of pressure applied to the prosthetic fingers. Processing spikes in real time made the reactions of control system to be more intuitive and effective, this could potentially advance the functionality of prosthetic devices.

- Energy-Efficient Sensor Networks: In a study by Taubner et al. 2019, SNN was implemented in a method aiming to detect and classify seismic events in real-time, and thanks to the event-driven nature of SNNs, this allowed for low power consumption and thus suitable for use in remote areas where energy is scarce.

These case studies demonstrate how SNNs are capable of efficiently handling various tasks across different domains, where temporal information processing is required, along with adaptability and lower-cost energy. These successful applications in real-world setups highlight SNNs potential to solve challenges in robotics, sensory processing, autonomous systems, and other fields.

2.6 Noise in Neural Systems

Noise exists in biological and artificial neural systems, it affects how information is passed and processed, thus it is important to understand how neural networks handle noise in order to design resilient and efficient models.

2.6.1 Types of noise in biological and artificial neural networks

Noise can arise from various sources in biological and artificial neural networks:

- Intrinsic Noise: This type of noise results from the stochastic nature of biological mechanisms — notably the randomness in the opening and closing of ion channels, membrane voltage fluctuations, and variability in synaptic transmission. In particular, synaptic noise, arising from differences in neurotransmitter concentration and receptor sensitivity, introduces variability in signal propagation (Destexhe and Contreras 2006). In artificial systems, intrinsic noise can also result from hardware variability or random fluctuations in synaptic weights (Faisal et al. 2008).

- Extrinsic Noise: This noise comes from external factors such as changes in the environment, sensory inputs or interactions with other neurons (Stein et al. 2005).

2.6.2 Effects of noise on neural information processing

Auditory interference, commonly referred to as noise, has the potential to influence neural information processing in both advantageous and disadvantageous ways. It can impact the precision in

signal transmission, which may consequently lead to errors in information processing that affect decision-making abilities. Elevated noise levels have the potential to induce instability within the overall neural activity, as observed in Gollo et al. 2012. Interestingly, under certain circumstances, noise may facilitate information processing by amplifying weak stimuli. This occurrence is associated with the concept of stochastic resonance, wherein the presence of a context-specific optimal noise level enhances the detection and transmission of weak signals, as highlighted in Mos 2004. Consequently, it indicates that noise could play a significant role in neural systems by increasing their sensitivity to relatively minor stimuli.

2.6.3 Noise-robust learning and computation in SNNs

SNNs exhibit notable robustness against noise and incorporate various mechanisms to support the acquisition of knowledge in difficult and noisy conditions. One such mechanism is homeostatic plasticity, which allows SNNs to adjust their synaptic weights and firing thresholds. This compensatory adjustment process serves to help SNNs adapt to a noisy environment (Turrigiano 2008). Spike Timing Dependent Plasticity (STDP) can adjust the spike-dependent weights in spiking neural networks (SNNs). With this form of plasticity, the network can strengthen robust sequences of spikes while weakening unnecessary noisy spiking, increasing its ability and efficiency in discerning temporal patterns in noisy environments (Caporale and Dan 2008). Additionally, when employing large populations of spiking neurons for information storage, SNNs can use population coding to average out noise and enable effective population-level signal processing within fully spiking architectures (Pouget et al. 2000).

2.7 Network Topology and Information Processing

The topology of a neural network directly influences its computational functions and efficiency. To understand how biological and artificial systems solve complex tasks, it is necessary to examine how different network structures shape information processing.

2.7.1 Importance of network structure in neural computation

The graph of a neural network, which includes neurons as nodes and synaptic connections as edges, plays a central role in how information is passed and processed. In biological systems, network topology affects connectivity, integration, specialisation, modularity, and feature extraction (Sporns et al. 2004). For instance, dense subgraphs of neurons allow more for a local processing whereas long edges between distant neurons allow the integration and passing of information to different regions.

2.7.2 Small-world networks and their properties

Small-world networks are distinguished by having high clustering coefficient and short average path lengths, which is necessary to efficiently process information both locally in the neuronal

populations (hubs) and globally across the whole network. Such topology is commonly found in biological systems, including neural ones, and contributes to efficient communication and robustness (Watts and Strogatz 1998).

In small-world networks, local clustering combined with long-range connections increases coherence in neural oscillations while maintaining an efficient processing system for both local and global stimuli. High clustering allows for efficient synchronous oscillations, and signal propagation is faster due to the short path lengths (Lago-Fernández et al. 2000).

2.7.3 Self-excitatory and self-inhibitory connections in neural circuits

Self-excitatory and self-inhibitory neurons are neurons that either excite or inhibit themselves; typically via a direct axon (loop) or an intermediate neuron (feedback loop). These neurons are either excitatory or inhibitory, not both, and their self-connections help stabilize network activity, regulate firing, or generate rhythmic patterns (Wang 2010).

Self-excitatory connections and feedback loops may amplify the signal so that the firing rates stay within a functional range, preventing it from becoming quiescent, whereas inhibitory feedback loops help against runaway excitation thereby preventing the whole network from becoming overly active. These topological features are essential to maintain a balanced network capable of efficiently processing information (Turrigiano and Nelson 2000).

2.8 Temporal Pattern Recognition Tasks

Temporal pattern recognition is the identification of patterns that occur over time, which makes it a complex and difficult task for both biological and artificial neural networks. This process is vital for speech recognition, gesture identification, and in general, time-series prediction.

2.8.1 Overview of temporal pattern recognition problems

Temporal pattern recognition tasks are those in which sequences of data are analysed and grouped, considering their temporal order. The idea behind these tasks is to identify certain patterns, rhythms, or trends over a time period, which makes them more difficult compared to static pattern recognition (Graves et al. 2013).

As seen in human understanding of speech or in their ability to anticipate the movement of objects, biological systems are good at identifying temporal patterns. For example, SNNs and RNNs are designed to perform these tasks by encoding temporal information and processing continuous inputs.

2.8.2 Applications in various domains

Recognizing temporal patterns is an important task that can be applied in many different disciplines. In the area of speech recognition, listening comprehension involves extracting time-sensitive

patterns from the audio signal. In combination with deep neural networks, the use of deep architectures and recurrent structures has enhanced the effectiveness of automatic speech recognition systems (Hinton et al. 2012). In gesture recognition, it is essential to discern the order and timing of movements, as they must be performed in a specific sequence (Hinton et al. 2012). In finance, pattern recognition is performed to forecast stock prices, detect shifts in the market, and uncover abnormal activities which help in decision making, risk assessment, and risk mitigation (Fischer and Krauss 2018).

2.8.3 Challenges in temporal pattern recognition

Over the years, achieving the recognition of patterns occurring within a specific time frame has not been without its challenges. One of the more persistent problems is the long-term dependencies, earlier time steps may get lost or decay with time. A neural network model, which attempts to tackle this problem, is Long Short Term Memory (LSTM) network. It can be argued, however, that even LSTMs do not perform well when it comes to very long sequences (Hochreiter and Schmidhuber 1997). Another problem is distinguishing the relevant patterns to be recognised from the considerable randomness in temporal data (Bishop 2006). Lastly, problems related to temporal information are still fundamental when dealing with large datasets due to their multimodality, continuity, as well as their complexity (Chung et al. 2014).

2.8.4 Emergent Autaptic Structures for Temporal Encoding

The dynamics of spiking neural networks (SNNs) has been studied in the context of autaptic connections, also known as self-synapses, focusing on their role in regulation and temporal pattern recognition (Yaqoob et al. 2023). Minimal SNNs evolved for specific predefined sequence detection, such as ABC pattern detection, tend to develop self-excitatory loops; enabling short term memory by maintaining neuronal output activities long after the stimuli stop.

The work conducted in this thesis differs from others mainly in modelling approaches, thus expanding this perspective, Yaqoob et al. 2023 formalised such dynamics with finite-state transducers, thereby capturing the role of autapses in the preservation of internal states during silence with state transitions, solidifying their functioning as intrinsic memory buffers in recurrent spiking systems.

2.9 Robustness and Adaptability in Neural Systems

Different kinds of neural networks, biological or man-made, are set to change and grow depending on the context they are put in. Understanding such mechanisms can allow us to formulate better and more adaptable neural networks and systems.

2.9.1 Mechanisms for maintaining performance under varying conditions

Neural systems undergo exposure to a range of environmental conditions, but tend to have consistent and reliable performance. This robustness is achieved by several mechanisms that work together:

- Homeostatic Plasticity: This allows neurons to remain active at the desired levels, which corresponds to alterations in network activity or synaptic structures (Turrigiano 2011). For example, in dealing with long-lasting changes in input, neurons can change their level of responsiveness to maintain activity.

- Redundancy and Degeneracy: Biological neural networks frequently demonstrate redundancy, multiple neurons integrating the same functions, and in most cases there is degeneracy, different components producing the same outcome (Tononi et al. 1999). These features allow the system to continue to operate even in the presence of damage or disorder.

- Noise Tolerance: Neural systems have learned to deal with noise by taking it as a feature rather than a liability. As mentioned earlier in the case of stochastic resonance, certain levels of noise can help detect and process signals by providing a degree of amplification for weak triggers (McDonnell and Ward 2011). This form of noise tolerance improves the overall reliability of biological systems in the presence of different and unpredictable environmental factors.

2.9.2 Adaptive behaviours in biological and artificial systems

Adaptability describes how a system can modify its operations and workflows based on changes in the environment. In living systems, organisms need to develop adaptive strategies and resourcefulness to survive, learn, defend themselves, and use various ecological niches. This type of dynamic modification is obtained primarily through learning and plasticity. For example, certain factors influencing synaptic plasticity, for instance long-term potentiation (LTP) and spike-timing dependent plasticity (STDP), allow for experience-dependent reorganisation of neural circuits (Abbott and Nelson 2000).

In machines, adaptive behaviour results from the application of training algorithms that modify their internal parameters according to the input data. Such systems can recognize and improve their behaviour as a result of feedback through reinforcement learning and evolutionary algorithms (Mnih et al. 2015).

2.9.3 Trade-off between specialisation and generalisation

Neural systems must often decide between specialisation or efficient performance at particular tasks and generalisation. Although bat echolocation is an achievement of precision, it is largely limited to a narrow range of conditions (Jones and Teeling 2006). In contrast, generalist systems

embrace a greater variety of tasks, but perform below the level of specialist systems in any single task. As with many artificial neural networks, there is an attempt to reach a balance between these two extremes by generalising without rigid overfitting to training data (Goodfellow et al. 2016).

2.10 Current Challenges and Future Directions

Although SNNs have shown great application in several fields, there are challenges that need to be resolved to optimally employ SNNs. This part explains the main concerns and the future scope of SNN research.

2.10.1 Scalability and Training Efficiency

One of the main concerns of SNNs is their scalability. Training large SNNs that contain numerous neurons and synapses tends to be resource intensive. It is often much more expensive than traditional ANNs. Event-driven SNNs are ideal for inference tasks, but training them is more difficult because their discrete spike-based communication is not compatible with traditional gradient-based approaches. There has been progress with methods such as surrogate gradient learning and spike-based backpropagation (Neftci et al. 2019), however, there is still more work to be done to create training frameworks for large networks which are less costly.

2.10.2 Transfer Learning in SNNs

As with any new technique, transfer learning development has been particularly effective within a deep learning context of ANNs. The application of transfer to SNNs is challenging given the difference in the processing of information across networks. Developing these concepts into real-world applications would improve the usage of SNNs and reduce the effort needed to retrain these networks for new tasks. Efforts are being made to adapt or fine-tune pre-trained SNN modules for different applications, which may produce more advanced models that are easier to use (Rueckauer et al. 2017).

2.10.3 Interpretability of Evolved SNNs

As SNNs are applied to more complex tasks, it is essential to gain insight into their information processing methods. For a long time, artificial neural networks, particularly deep learning models, have been considered black boxes. For a long time, neural network vision models have been the focus of deep learning research, which led to the development of novel techniques such as activation atlases and feature visualization that revealed the role of each neuron in pattern and object recognition. Advanced techniques have shown that individual neurons within deeper layers responded to distinct concepts. With large language models, studies focusing on mechanistic interpretability recently uncovered how certain neurons track syntax and resolve pronouns and encode abstract features such as sentiment and world knowledge (Olah et al. 2020; Nanda et al.

2023). Such efforts bring novel changes towards DNNs, gradually transforming the perception from black-box systems that lack interpretability to systems with components that can be analysed and partially grasped.

However, the combination of the temporal dynamics and discrete spiking behaviour of SNNs makes their representations difficult to interpret. This lack of clarity poses a major difficulty in applying SNNs to domains that need to be explainable, such as medical decision making. Currently, there are ongoing initiatives that aim to improve the interpretable frameworks of the systems by spatial visualization and interpretation of spike-based activity (Yin et al. 2020).

2.10.4 Integration with Cognitive Architectures

The incorporation of SNNs into cognitive architectures aims to improve the level of how animats and robotic systems can replicate human-like intelligence. Cognitive architectures such as ACT-R or Soar offer models for constructing human cognition that form the basis of perception, decision making, and motor output (Anderson 2004). Using these frameworks, researchers can use SNNs as peripheral devices and simulate cognitive and motor behaviours based on sensory input and physiological output. This development could be beneficial to autonomous robotics because it can foster the development of systems that are more adaptive and flexible to the actions of humans in the surrounding world.

2.10.5 Neuromorphic Hardware and Real-World Deployment

The adoption of neuromorphic devices is one of the most exciting avenues for SNN implementation; however, challenges remain when it comes to bringing research prototypes to deployment. One primary issue is the integration of software with hardware, as well as effective neuromorphic device training, in addition to the creation of standardized frameworks. More effort should be directed towards resolving the SNN algorithm and neuromorphic hardware interoperability issues with the goal of making them suitable for edge computing, robotics, and sensor network applications.

2.10.6 Towards Lifelong Learning and Adaptation

An important feature of biological neural systems is their ability to learn and adapt at any point in their life. However, some modern SNN models are often trained to perform specific tasks and do not have the ability to change their training environments or adapt to new ones without retraining. An interesting future direction is the identification of plasticity mechanisms for learning and adaptation, such as online learning algorithms that allow SNNs to modify synaptic weights based on novelty (Zenke et al. 2017).

Addressing these challenges will be essential to progress in the field of SNNs. Resolving the integration of scalability, transfer learning, interpretability, and cognitive architecture will make SNNs more effective in building adaptive systems that function efficiently in complex real-world settings.

Chapter 3

Methodology

3.1 Foundational Framework

3.1.1 Overview of the GReaNs Platform

This study uses the Genetic Regulatory Evolving Artificial Networks (GReaNs) platform (Abdelmotaleb et al. 2014): a software infrastructure designed for the evolution of gene regulatory networks through genetic algorithms, and subsequently adapted to spiking neural networks (SNNs) for temporal pattern recognition. The flexibility of SNN controllers to adapt in terms of the number of nodes and their connectivity as task difficulty increases is facilitated by GReaNs genome-to-network encoding, which maps network topologies from linear strings of coding and regulatory units.

GReaNs is unlike neuroevolution approaches such as NeuroEvolution of Augmenting Topologies (NEAT) (Stanley and Miikkulainen 2002) which add complexity to fixed structure neural networks, GReaNs employs fully symbolic linear-genome representation that allows the addition, removal or modification of regulatory units (Wróbel and Joachimczak 2011).

By avoiding biologically-inspired rigid regulatory structures, GReaNs allows networks to expand or contract their topology and connection density throughout evolution dynamically. These traits are essential for reliable performance under noisy and unpredictable environmental conditions (Floreano and Mattiussi 2008). Our selection of GReaNs was due to its unique features, which include adaptive capability, symbolic genome structure, and compatibility with our lab's computational infrastructure.

3.1.2 Structure and Encoding of the Genome

In the GReaNs platform (Abdelmotaleb et al. 2014), each neural network is encoded as a linear genome—a symbolic sequence of elements that can be divided into functional components. These elements include promoters (P), genes (G), and special elements (S), which are grouped into simple regulatory units. Each regulatory unit consists of exactly one promoter and one gene, forming a simplified encoding structure from which network nodes and connections can be inferred.

Although this encoding draws some inspiration from gene regulatory networks, it does not

implement a developmental mapping process. Instead, it provides a direct and interpretable mapping between the genome structure and the resulting network topology. In practice, one could reconstruct the network manually by parsing the genome, although this is automated in simulation.

Each element in the genome is associated with a set of attributes used in the genome-to-network mapping:

- **Type:** Specifies whether the element is a promoter (P), a gene (G), or a special node (S), such as an input or output.
- **Coordinates:** Defines the position of the element in a continuous 2D affinity space, which determines the potential interactions between promoters and genes.

Although excitatory and inhibitory edges can be represented in network visualisation, there is no sign attribute in the genome or simulation code. The inhibitory or excitatory nature of a synapse is defined by the conductance channel applied during the simulation step (E or I), following the AdEx model used.

3.1.3 Genome-to-Network Mapping

Regulatory units were directly mapped to spiking neurons (as shown in Figure 3.1), and synaptic connections were established based on spatial relationships in a 2D affinity space. Specifically, a synapse is created between a gene and a promoter if their Euclidean distance d is below a threshold (set to 5 units). The affinity space is bounded, with coordinates typically initialised in a 0–100 range.

$$w = \frac{1}{10d + 1} \tag{3.1}$$

Rationale: Because weights decay with distance, the network avoids unnecessarily strong long-range edges. Importantly, this constraint does not prescribe recurrence or autapses; those appear (or not) depending entirely on evolution.

This simple distance-based mapping allows for sparse connectivity with recurrent loops, producing networks with varying topologies, including self-loops and feedback motifs. The flexibility of the genome encoding supports the emergence of non-uniform and functionally diverse network structures without requiring predefined wiring patterns.

3.1.4 Neuron and Synapse Model

We implement neural dynamics at the level of the evolved networks using the Adaptive Exponential Integrate-and-Fire (AdEx) model. It offers a biologically plausible yet computationally efficient framework, capturing key neuronal behaviours such as spike-frequency adaptation—an important mechanism for short-term memory and temporal pattern recognition in spiking networks.

The dynamics of the membrane potential of each neurone V and the adaptation current W are described by the following differential equations:

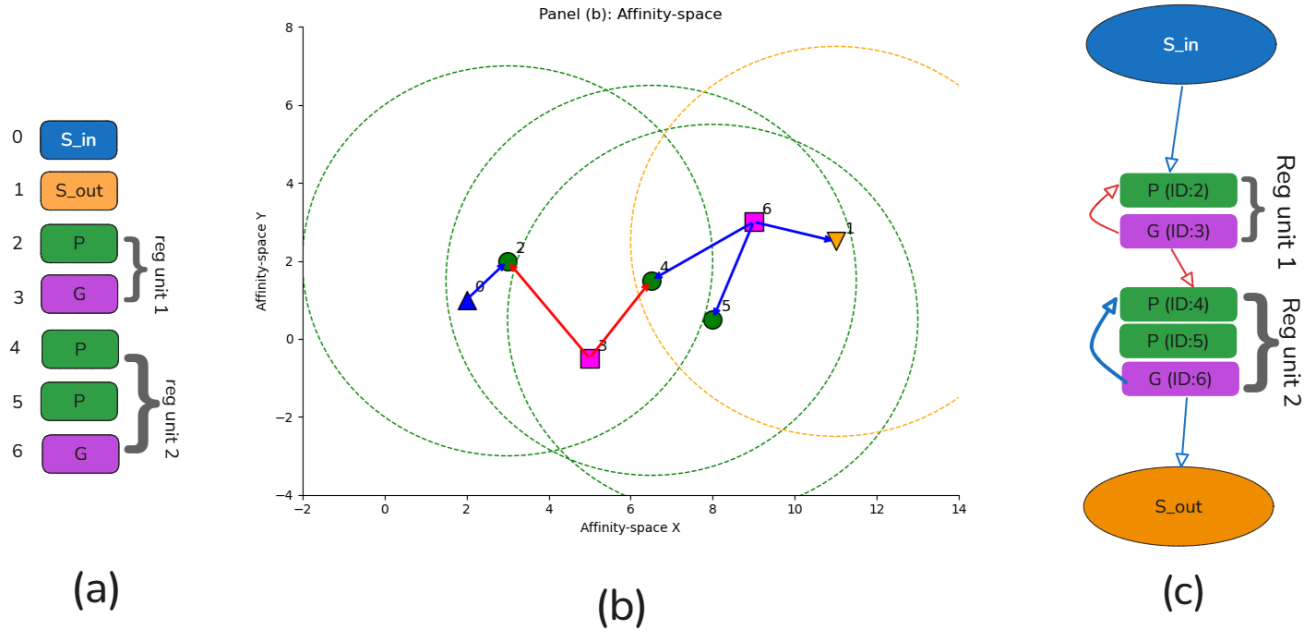


Figure 3.1: **Mapping and Final Network Structure in GReaNs.** (a) Layout of the genome, consisting of two regulatory units: *Reg unit 1* (elements 2–3) and *Reg unit 2* (elements 4–5–6), each composed of one or more promoters (P) and a single gene (G) acting as a transcription factor. Input and output nodes are represented by S_in (ID: 0) and S_out (ID: 1), respectively. The genome is sign-free; polarity is assigned at simulation via conductance channels. (b) Spatial distribution of genetic elements in 2D affinity space. Dashed circles (radius = 5 units) around each promoter and the output node indicate binding capture regions. Directed arrows represent potential regulatory bindings from gene to promoter (or output): red for excitatory-channel and blue for inhibitory-channel couplings used in simulation. Multiple arrows targeting promoters of the same regulatory unit collapse into a single synapse in the resulting SNN. (c) Final spiking neural network structure after mapping. Each regulatory unit becomes a neuron. Arrows denote the flow of influence through the network: from input node (`S_in`), through regulatory units, and finally to the output node (`S_out`).

$$C\frac{dV}{dt} = g_L\,(E_L - V) + g_L\,\Delta_T \exp\!\left(\frac{V - V_T}{\Delta_T}\right) + g_E(t)\big[E_E - V\big] + g_I(t)\big[E_I - V\big] - W(t), \quad (3.2)$$

$$\tau_W \frac{dW}{dt} = a\,(V - E_L) - W, \quad (3.3)$$

$$\tau_E \frac{dg_E}{dt} = -g_E + \sum_j w_j^{(E)}\,\delta(t - t_j^{(E)}), \quad (3.4)$$

$$\tau_I \frac{dg_I}{dt} = -g_I + \sum_k w_k^{(I)}\,\delta(t - t_k^{(I)}), \quad (3.5)$$

where:

- C: membrane capacitance
- g_L: leak conductance

- E_L, E_E, E_I: reversal potentials for leak, excitatory, and inhibitory currents
- Δ_T, V_T: slope and location of exponential threshold
- W: adaptation current, governed by a and τ_W
- $g_E(t)$, $g_I(t)$: synaptic conductances, governed by τ_E, τ_I
- $w_j^{(E)}$, $w_k^{(I)}$: synaptic weights of presynaptic spikes

Spike Generation and Reset Rule A spike is emitted when $V(t) \geq V_{\text{det}}$, then the neuron's state is updated as follows:

$$\begin{cases} V \leftarrow V_r \\ W \leftarrow W + b \end{cases} \tag{3.6}$$

Here V_{det} is the numerical detection threshold used by GReaNs which is different from the AdEx threshold parameter V_T and V_r is the reset voltage.

Synaptic Input Each presynaptic spike increases the conductance of the target neuron by $g_{\text{gain}} \times w_{ij}$. Excitatory and inhibitory inputs are distinguished by the conductance channel they affect (not by the sign of weight, which remains positive). Both $g_E(t)$ and $g_I(t)$ decay exponentially, producing temporally rich synaptic currents that interact with AdEx equations to allow for precise and temporally sensitive signal integration.

All parameter values are within commonly used AdEx ranges (Naud et al. 2008).

3.1.5 Evolutionary Algorithm

To develop SNN controllers for more advanced tasks, we selected a genetic algorithm (GA) framework over alternatives such as gradient descent or reinforcement learning, due to its suitability for evolving discrete topologies, non-differentiable components, and asynchronous dynamics.

The GA operates through the standard loop of selection, variation, evaluation, and reproduction across multiple evolutionary generations (iterations of the algorithm). Note that "generation" here refers to population turnover, not to random generation of weights. This approach allows for open-ended, unstructured neuroevolution of SNN controllers, removing the need for hand-crafted network designs or explicit guidance.

The process is as follows:

1. **Initialisation:** A population of $n = 300$ genomes is initialised. Each genome encodes a minimal network consisting of two input neurons, two output neurons, and two interneurons. These units are randomly connected. Synaptic weights are sampled from a normal distribution $\mathcal{N}(\mu = 0, \sigma = 0.3)$ and cut to the range $[0, 1]$. Each synapse is assigned to an excitatory or inhibitory channel (E/I) during the simulation (weights remain positive). Each genome is limited to 100 elements and no more than 8 interneurons.

Table 3.1: AdEx parameters used in this work. The third column summarises the role of each setting in achieving regular spiking.

Parameter	**Value**	**Justification**
$\tau_{E/I}$ (exc/inh time constants)	5 ms	Short decay to keep transients brief and comparable across channels.
g_L (leak conductance)	10 nS	With C, sets τ_m in a stable and like a cortical system.
E_L (rest potential)	-70 mV	Standard resting level where the subthreshold baseline is below V_T.
E_I (inhibitory reversal)	-70 mV	Chosen to give primarily shunting inhibition.
E_E (excitatory reversal)	0 mV	Chosen for a robust depolarisation of excitatory postsynaptic potentials.
Δ_T (slope factor)	2 mV	Sharp spike while avoiding runaway bursting.
V_T (AdEx threshold)	-50 mV	Allows spiking for brief inputs.
C (capacitance)	0.2 nF	Membrane time constant suitable for patterns ranging tens of ms.
a (adaptation conductance)	2 nS	Mild subthreshold adaptation for decaying activity trace which briefly reduces excitability.
b (spike-triggered adaptation)	0 pA	Set to zero to target regular spiking.
τ_w (adaptation time constant)	30 ms	Adaptation covering the signal response interval.
V_r (reset voltage)	-58 mV	Quick return to subthreshold after spike.
V_{det} (detection threshold)	0 mV	GReaNs detection level which is separate from V_T.

2. **Testing:** Each network is decoded and tested within the animat task environment. A fitness score is calculated to reflect performance in temporal pattern recognition, following a directional smooth path, and avoidance of distractors. (Details are in Section 3.1.7.)

3. **Selection:** A maximum of five elite individuals are preserved unchanged through elitism (top 1–2% of the population). The remainder of the next generation is produced using tournament selection (tournament size = 2), which favours genomes with better (lower) fitness scores.

4. **Variation:**

   - *Mutation:* Each genome may undergo:
     - Local edits, including modification of synaptic weights and changes to unit type or sign. ('sign' refers to channel assignment - the weights are never negative).
     - Small coordinate displacements in affinity space, simulating gradual changes in network geometry. These perturbations are sampled from $\mathcal{N}(\mu = 0, \sigma = 0.1)$,

applied to gene positions in the continuous genome space.

– Structural mutations, including insertion, deletion, and duplication of genome segments. Duplication events are preferred over deletions (duplication probability is higher than deletion probability), and structural mutations occur at a typical rate of $\mathcal{O}(10^{-3})$ per locus per generation.

Design choices **Population size ($n = 300$).** Embodied evaluation is noisy (randomness in object positions, orientations, and input sequences), leading to greater fitness variance per individual. A population of 300 individuals balances exploration of different topologies and a stable selection per generation at manageable computational cost: a smaller n led to early convergence in the first runs whereas a larger n increased wall-clock with no meaningful gains. The final setting is in accordance with the reproducibility tables (See Appendix A).

Selection pressure (2-tournament + elitism of 5). A tournament size of 2 puts mild pressure (appropriate under high fitness noise) helps avoid a loss of diversity, while having the top $\approx 1.7\%$(5 elites) passed to the next generation protects novelty from drift. This combination successfully and reliably maintained the behavioural variation and produced repeatable champions in our setup.

Mutation operators and rates. We combine frequent, fine-grained edits with rare structural changes: local synaptic/neuronal edits at $\approx 0.5\%$ per locus and coordinate jitters ($\mathcal{N}(0, 0.1)$) help incremental adaptation, whereas low-rate structural edits ($\mathcal{O}(10^{-3})$ per locus) with a duplication bias facilitate the emergence of recurrent motifs without destabilising the incumbents. This mixture yielded a gradual refinement interspersed with occasional topology jumps, as reflected in the evolved self-excitatory loops of the top champions.

AB frequency during evolution (30% AB and later 50%). During evolution, AB constituted $\sim$30% of emissions (the remainder was uniformly split across AA/BA/BB) to keep the correct pattern present but not dominant —discouraging trivial strategies. Sequences were uniformly permuted via [1] to preserve marginal counts without positional bias. For post-evolution analyses, we also used a 50% AB regime for fair comparisons.

Each evolutionary run is conducted for 2000 to 2500 generations, or less if a suitable and stable high performing solution is found and maintained for 50 generations. The networks are permitted to expand in size over the generations to a maximum of 8 interneurons, given that duplication-biassed mutation strategies encourage the evolution of recurrent and richly structured topologies.

3.1.6 Animat Body and Sensors

As part of this study, aimed at evolving SNN controllers with temporal recognition and adaptive behaviour, SNNs are embedded into a 2D virtual world as animats - simulated agents with basic actuators and sensors.

An animat (whose behaviour is a phenotype) is evaluated using a genetic algorithm (GA). Each candidate genome is processed as an SNN controller within an animat. This is integrated

[1] Fisher-Yates (`std::shuffle`) using a uniform random-bit generator.

with an evaluative time-based simulation. Although the animat is tested alone during each trial, the conditions are not fixed. The environment is randomly regenerated for each testing phase. This includes random object placements, varied input sequences, and different initial orientations of the animat. This guarantees robustness to environmental variability. The fitness of the animat is calculated on the basis of its behaviour performance during the several simulation trials. The computational cost of the simulation trial is high. For example, running the trials over many generations with 300 individuals and dynamic, time-varying environments is resource-intensive. Compared to other neuroevolution studies that rely on static datasets for signal processing or classification, our approach evaluates networks in an embodied setting that requires temporal adaptation and interactive control. This enables the emergence of dynamic behavioural strategies within SNNs. For instance, the consistent evolution of excitatory autapses in high-performing networks suggests a mechanism for sustained internal activity or short-term memory, which would not typically arise in static, feedforward tasks.

An animat engages with its surroundings using two actuators modelled as wheels on the left and right of its body. Each actuator is controlled by a corresponding output neuron in the spiking neural network (SNN) of the animat. When an output neuron fires, it generates a thrust force that is applied on the corresponding side (AL -> left wheel, AR -> right wheel). The model accounts for basic Newtonian physics, such as motion, inertia and drag. Firing both output neurons simultaneously and equally allows the animat to move forward in a straight line. In contrast, asymmetric activity, i.e., one output neuron is firing more than the other, results in turning. If no thrust is applied, it means no output neuron fires, the animat slows down and eventually stops due to friction.

To obtain sensory information, the animat's environment contains two types of signals "A" and "B", which require four physical sensors in total (two for A, two for B). These sensors receive the range distance, decreasing with the distance from the source with intensity $I(d)$ (see Fig. 3.2).

Signals vs. physical sensors. Objects emit sequences of two pulses, which can either form the target pattern AB, or one of the distractor patterns, which are AA, BA, and BB. A and B are abstract signal channels, and the colours of the figures are illustrative (the animat does not see colours). At every step in the simulation, a pattern which could be either AB or one of the distractors comes in from the left or right (rarely seen from front/back). For the controller, the discriminative cue is primarily of a temporal order of the pulses comprising AB and the other patterns, along with lateralisation of left and right. The two physical sensors of a channel type, A or B, measure distance-based intensity and are combined into a single scalar variable called Diff (Eq. 3.7). For each pattern, we round Diff to two decimals and keep that value fixed for the entire presentation. This prevents tiny within-presentation fluctuations from acting as unintended temporal cues. Neutral objects (N) do not emit informative signals, and we report T/N and D/N along with T/D to disentangle the active avoidance of distractors from random collisions.

$$\text{Diff} = \frac{1}{1 + \exp\left[-10\,(S_L - S_R)\right]} \in [0, 1] \tag{3.7}$$

This produces a directional scalar: 0.5 indicates straight forward, values > 0.5 for left, and < 0.5

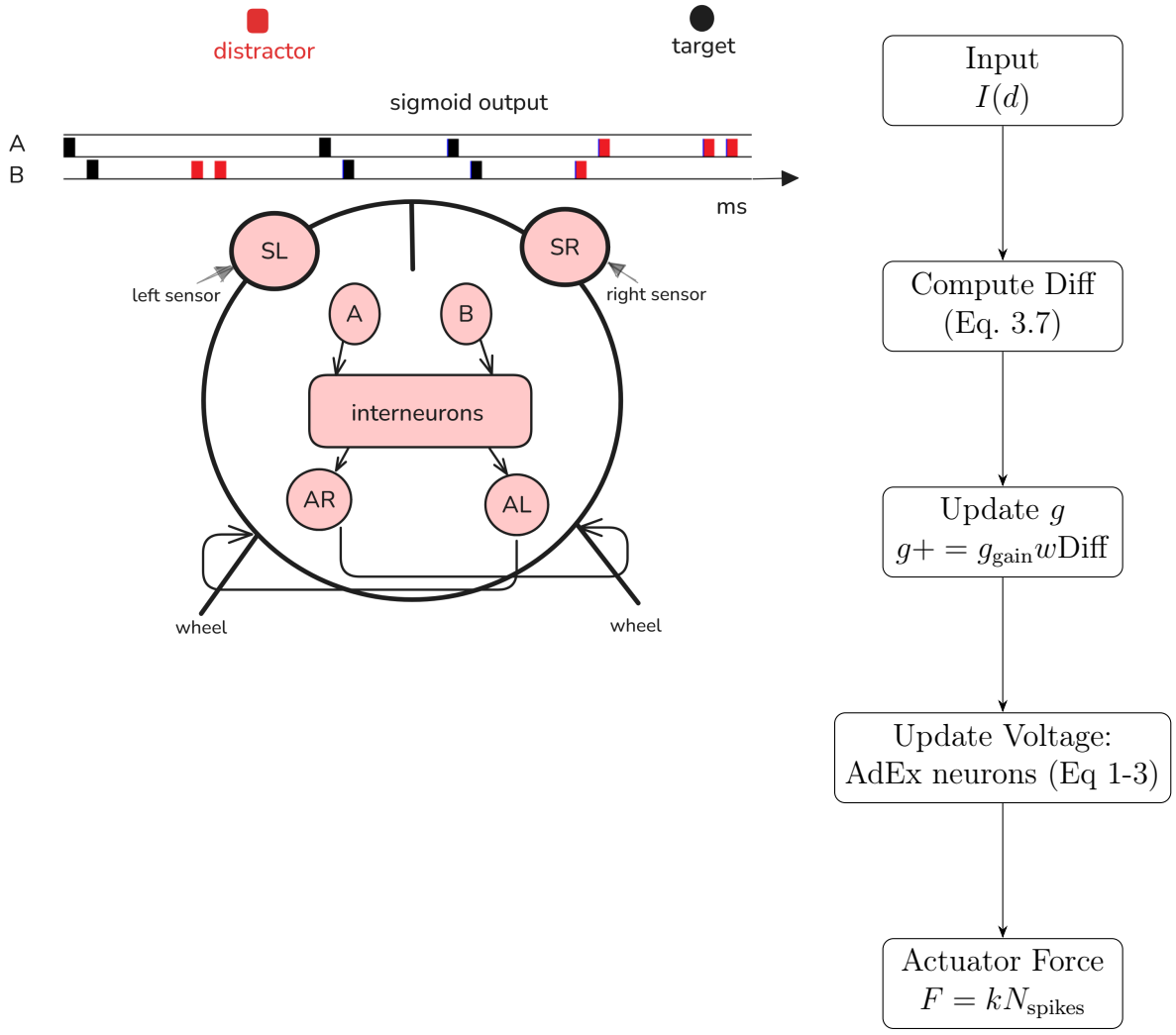


Figure 3.2: Animat's sensory-to-motor processing pipeline with emphasis on: gradient input, computation of Diff, modulation of conductance, and calculation of actuator force.

for right. We round Diff to two decimals and keep it constant during each pattern.

Design decisions (body & sensors) In order to maintain unambiguous contact and stabilise the left-right gradient during the final approach while minimising "grazing" events, the animat body was set larger than the objects (twice the object size). Distances are defined in the animat radii R for scale-free comparison. The physical sensors for each channel (A and B) are located in fixed orientations of $+60°$ and $-60°$ relative to the forward axis (i.e. $60°$ and $300°$ in body coordinates; $120°$ separation). Sensor positions are as follows given by Eq. 3.8:

$$\mathbf{p} = \mathbf{c} + \rho\, R\, \big[\cos(\theta + \phi),\, \sin(\theta + \phi)\big], \tag{3.8}$$

where R is the animat radius, ρ a distance multiplier, θ the body heading (radians) and $\phi \in \{+60°, -60°\}$. Therefore, when the signal arrives in a straight way (Diff $= 0.5$), all connected interneurons receive the synaptic conductance corresponding to roughly half of a full presynaptic spike. If, on the other hand, the signal comes strongly from one side, Diff tends towards one of the extremes: for a strong signal on the left (Diff $\to 1$ is also very rare in our champions), the injected

conductance approaches that of a full presynaptic spike, whereas for a strong signal on the right (Diff $\to$ 0) the injected conductance becomes very small, representing the opposite directional extreme. This strategy of encoding compresses the directional information into a single value, thus making the interface between the network and the inputs more straightforward.

Actuators and physics. The animat has two thrusters (left/right wheels) where each is driven by an output neuron AL/AR for the right/left wheel (force on the right wheel pushes the animat left). At each simulation step, the AL/AR spike count is converted into positive activations a_L, a_R that grow linearly with the spike count. Each thruster gives a push in its own fixed body direction (ϕ_i); push size = $a_i \eta_i$ as defined in Eq. 3.9:

$$\mathbf{F} \leftarrow \mathbf{F} - \sum_{i \in \{L,R\}} a_i \, \eta_i \begin{bmatrix} \cos(\theta + \phi_i) \\ \sin(\theta + \phi_i) \end{bmatrix}, \qquad \tau \leftarrow \tau - \sum_{i \in \{L,R\}} a_i \, \eta_i \, \sin(\phi_i), \tag{3.9}$$

where θ is the animat's body orientation. The two thrusters are mounted symmetrically with body-frame headings $\phi_L = 225°$ and $\phi_R = 135°$ (actuator placements/orientations set at initialisation).[2] Linear viscous drag is applied to translation and rotation,

$$\mathbf{F} \leftarrow \mathbf{F} - \lambda \, m \, \mathbf{v}, \qquad \tau \leftarrow \tau - 2\lambda \, m \, \omega,$$

which damps motion in the absence of thrust. Balanced AL/AR activations produce forward motion; imbalanced activations produce turning; zero activation leads to a stop under drag. Default actuation uses per-side multipliers η_i ("force multipliers") with an effective thrust baseline ≈ 2 in our test matrix.

3.1.7 Task and Environment

Arena size, density, and boundaries The arena is a continuous square space with coordinates $[-S, S] \times [-S, S]$, where $S =$ `orbWorldSize` is the *half-side* in simulator units. The sides for which initial placements are sampled are $[-(S-10), +(S-10)] \times [-(S-10), +(S-10)]$ to prevent animats from spawning on the boundaries. Internally, the same bound is exposed as `worldMaxX` and `worldMaxY` (See Appendix D). In toroidal mode, positions wrap at $\pm S$ on each axis; otherwise the arena is open. For scale-free reporting, distances are expressed in animat radii R; for example, with $S = 45R$ the arena is $90R$ across—about 45 animat diameters per side.

Placement bounds and scale We measure distance in animat radii R (customisable in the GUI). We initialise object placements at 10–45 R from the start position (for $R = 2$ SU, $\approx$ 20–90 SU), avoiding trivial near or far starts: this reduces random early collisions during immature behaviour and ensures successful runs require directional navigation and correct temporal discrimination.

[2] Left actuator: `orientation`=270°, `direction`=225°; right actuator: `orientation`=90°, `direction`=135°.

In all three experimental setups explored in this thesis, the animats had a task to distinguish a "correct" stimulus pattern from "wrong" patterns and move toward targets while avoiding distractors. The environment is a two-dimensional continuous space containing both target and distractor objects. Each object emits a temporal sequence of signals (which can be thought of as flashes of light with two different colours, or sounds with two different frequencies). The target always emits a specific two-signal sequence (the correct pattern), denoted "AB" (signal A followed by signal B). Distractors emit incorrect patterns, consisting of the other possible two-letter combinations using signals A and B (e.g., "AA", "BB", or "BA").

Pattern timing Patterns comprise two brief pulses separated by a gap, followed by a silence before the next pattern. Typical values: 8-10 ms pulses; IEEE: 3 ms gap; ICANN: gap drawn Poisson(λ=30 ms); inter-pattern silence 80–150 ms depending on experiment. Targets and distractors emit in alternating, non-overlapping phases ("chorus"), with AB ≈30% during evolution and 50% AB in testing.

Whenever the animat comes into physical contact with an object, it is counted as 'collected'. A target collection is a success; a distractor collection is a failure. The animat's fitness in each trial is determined by the ability to collect targets while avoiding distractors. To reflect a strong punishment for recognising other wrong patterns, distractor penalties are weighted more heavily (by a factor of two), we also include a penalty for inefficient behaviour, such as losing an actuator or input node and circling to favour more directional navigation.

Thus, animats are scored higher for moving directly toward targets and penalised for erratic locomotion, even if they eventually succeed. The exact form of the fitness function varied across the three experimental setups explored, but the underlying principle remained consistent. A baseline fitness score was calculated as shown in Eq. 3.10:

$$f_{\text{base}} \propto \frac{T - P\,D}{N_T}, \tag{3.10}$$

where T is the number of targets collected, D the number of distractors collected, N_T the maximum possible targets in a trial, and $P = 2$ the distractor penalty multiplier. Path efficiency penalties were then added to f_{base}.

We set $P = 2$ to reflect findings in biological and behavioural literature suggesting that negative outcomes (e.g., collecting distractors) have a stronger impact on evolution than positive ones (Nolfi and Floreano 2000; Sutton and Barto 2018). Individuals that collided with distractors early in a trial were assigned a very high values of the fitness function (high values mean low fitness) to ensure strong selection pressure against such behaviours.

This choice of penalty structure encourages the evolution of precise pattern discrimination over exploratory or erratic behaviours. Initial trials also revealed that excessive trajectory changes diluted correct recognition performance, prompting the inclusion of path-efficiency terms in the fitness function (Floreano and Mondada 1996).

3.2 Results 1: IEEE (2017) - Unique Methodological Elements

This study Bensmail, Steuber, Davey et al. 2017, introduced the general task and evolutionary approach, establishing the baseline framework described above. The following specific methodological choices were made and distinguish it from following studies.

- **Environment configuration:** During evolution, the world was set to be *toroidal* (wrap-around boundaries) so that the animat was never far from objects, and after leaving the arena, the animat re-entered from the opposite side. This was to ensure the animat had enough exposure to target and to avoid the boundaries effect. In each testing, there were initially 4 objects present simultaneously (2 targets and 2 distractors placed at randomly - we chose this to speed up the computation). Whenever the animat collected an object, a new object of the same type would immediately reappear at a random location, maintaining a constant number of available targets and distractors. During testing, the environment was made non-toroidal (an effectively infinite plane) to examine the champions' performance without boundary wrapping.

- **Temporal pattern parameters:** In evolutionary trials, each target emitted the correct 'AB' pattern and each distractor emitted one of the three wrong patterns ("AA", 'BB' or 'BA'). To simplify the evolutionary process and enable clearer behavioural analysis, we limited training to a single target pattern (correct pattern 'AB') during evolution. This approach is parallel to classical conditioning; for example, Pavlov notes that the sound of a bell does not elicit a response until food is paired with its ringing (Pavlov 1927). In the same way, reduction to one constant target was intended to encourage the development of distinct and clear behavioural patterns in the animats, much like the specific responses observed in conditioned behaviours.

  The signals were very short, only 8ms in duration each, with a 3ms gap between the two signals in a pattern (so an entire two-letter word lasted 19ms). A 20ms silence was imposed between successive pattern presentations. As previously described, all targets emitted their AB patterns in synchrony (e.g., if two targets were present, both emitted AB at the same time), and distractors likewise emitted their respective patterns in unison. These emissions occurred in alternating phases - target and distractor signals never overlapped. The relative frequency of the correct pattern versus wrong patterns in the sensory input was controlled but varied across testing scenarios to promote robust recognition (approximately 30% of signals were from the target on average).

- **Fitness testing:** Each individual's fitness was measured by averaging its output over multiple worlds presenting different pattern sequences and objects placements. In the early generations, fitness was evaluated across 13 distinct trials (worlds). In 10 of those trials, all three types of wrong patterns (AA, BB, BA) were emitted by distractors with roughly equal probability, and in the remaining 3 trials a single wrong pattern (e.g., AA only)

was emitted to constitute the majority of distractor signals (to test the animat's ability to handle a most frequent distractor pattern). The animat's behaviour was simulated for 3900 ms per world (allowing on the order of 100 pattern presentations per trial). If in any trial the animat happened to collect a distractor before collecting a target, a large fitness penalty was assigned (effectively setting the the value of the fitness function above 1 for that individual to remove it from selection contention). The fitness function in this study took the form of an 'error' measure minimised as given by Eq. 3.11:

$$f = 1 - \Big(\frac{T - P\,D}{N_T} - 0.2\,c\Big), \tag{3.11}$$

where T is the number of targets collected, D is the number of distractors collected, $P = 2$ is the distractor penalty multiplier, and N_T is the maximum attainable targets (here $N_T = 1$ per trial, since at most one target could be collected within the time window). The term c is a flag indicating whether the animat changed direction during the trial ($c = 1$ if a turning manoeuvre occurred, 0 if the animat maintained a straight trajectory throughout). Thus, if the animat moved in a straight line ($c = 0$), it avoided a penalty; but if it turned at any point ($c = 1$), a penalty of 0.2 was added (making the fitness worse). This simple term was introduced to encourage efficient, roughly straight navigation towards targets and to discourage perpetual circling. Lower f indicates better performance, with $f = 0$ corresponding to perfect behaviour (collecting the target and no distractors with no unnecessary turning).

- **Evolutionary run parameters:** Evolution was run for a maximum of 2500 generations (with elitism and tournament selection, as already described). Tournament selection (size 2) was applied to the remaining population as described in the framework. The mutation rates were relatively high in this initial study: the probability was 0.01 per genome per generation, while the deletion rate was 0.001 per genome. This ten-fold higher rate of duplication vs deletion biased the search toward increasing network size. In fact, networks often grew beyond the initial 2 interneurons; some evolved individuals had on the order of 4 to 6 interneurons and rich recurrent connectivity. We performed 50 independent evolutionary runs under these conditions. To progressively challenge evolving networks with harder temporal memory demands, the testing conditions were made more stringent in later generations: after generation 1900, the number of objects in each testing world was reduced to 2 (1 target and 1 distractor), and the silence interval between pattern presentations was increased (to 50 ms between patterns from generations 1900–2000, and further to 80 ms after generation 2000). By generation 2100 and beyond, each trial presented 150 patterns with a 100ms interpattern silence. These phased adjustments forced networks to rely on internal state maintenance (memory of the first signal) for longer periods and with fewer concurrent stimuli, ensuring that only those controllers with genuine temporal pattern recognition capability would continue to improve. If an evolving champion's fitness dropped below a threshold (indicative of near-perfect performance) before 2500 generations, the run was terminated early (after an additional 100 generations to confirm stability); otherwise,

all runs ended in generation 2500.

Input Simplification Analysis: To investigate the role of sensory input granularity in evolved behaviour, we designed a separate testing stage where each champion was only given two fixed values to each sensors, 0.33 (0.67) if the target/distractor was on the right (left). This simplification eliminated the gradient or the change in distance throughout the input, forcing the system to use a reduced spatial representation. This analysis aimed to facilitate the interpretation of network dynamics by reducing the complexity of input variations during network decoding.

3.3 Results 2: ICANN (2018) – Unique Methodological Elements

The second study, Bensmail, Steuber and Wróbel 2018, built on the Bensmail, Steuber, Davey et al. 2017 framework but introduced several key methodological changes aimed at investigating the robustness of the controllers and the requirement of internal memory. Notable unique elements in this study include:

- **Environment and object placement:** In contrast to the toroidal world used previously, here the evolution and testing were conducted in an *open* world (large, effectively unbounded 2D arena) without toroidal. To ensure the animat was neither too close nor too far from objects at the start, each trial placed one target and one distractor at random positions constrained to be at least 10 animat radii away and at most 45 animat radii away from the animat's starting position (the animat's radius was 2 units, so this corresponds to roughly 20 to 90 units distance). In order to minimize the chances for random collisions during the early evolution stage when behaviour is not refined, objects were placed at least 10 animat radii away from the starting position. This allows for ensured collection of objects through active navigation as opposed to mere coincidence, thereby enhancing selection pressure on significant movement and discrimination techniques. Once an object was collected by the animat, it was *not* replaced during that trial (each object could be collected at most once). Thus, in any given testing scenario there was a maximum of one target and one distractor available. These changes meant that the animat had to navigate a larger space to find objects and could not rely on objects reappearing nearby; it had to correctly identify the target in a single approach opportunity.

- **Temporal pattern:** The signal patterns remained two-letter words (two sequential pulses), but a critical difference was that the interval between the two signals in a pattern was no longer fixed. Instead, the duration of the silence separating the first and second signal of the pattern was randomly drawn from a Poisson distribution with mean $\lambda = 30$ms for each occurrence. In other words, each time a pattern "AB" (or a wrong pattern like "AA", "BB", etc.) was emitted, the gap between the two signals (A and B) could vary, with an average of 30ms. The length of each signal pulse was fixed at 10ms, and the silence after each two-signal

pattern (before the next pattern begins) was fixed at 150ms. By introducing variability in the inter-signal interval, this setup encouraged the evolution of networks capable of genuine short-term memory: the network could not rely on a precise, fixed timing; it needed to handle cases where the second signal might come a bit earlier or later. During evolution the target always emitted pattern AB and the distractor emitted randomly one of the other patterns (AA, BA, or BB) in each pattern presentation, with all patterns alternating in time (no overlap).

- **Signal intensity handling:** An additional methodological tweak was made to the way signal intensity was presented across the two signals of a pattern. In this study, the intensity of the second signal was held constant and equal to that of the first signal in the pattern, as measured at the start of the pattern. In other words, even if the animat moved closer or farther from the source during the interval between the first and second pulse, the intensity of the second pulse was not updated to the new distance. This was done to remove a potential cue — if intensity were allowed to change, the network might infer the temporal relationship between the two signals from the change in signal strength. By fixing the intensity, the task of recognizing the pattern had to rely purely on temporal order rather than any within-pattern intensity differences. (This modification made the task more challenging for the animat, but it simplifies analysis by decoupling the pattern recognition from continuous distance feedback during a single pattern.) The directional encoding of sensor input (the Diff value computation) remained the same as in the foundational framework (Eq. 3.7), and the patterns still conveyed directional information to the network indicating where the source was. However, within a single pattern, the relative intensities of A vs B were locked to the initial sensor reading at the start of the pattern and did not update during its duration. This ensured that any changes in movement during the pattern would not be reflected in the input - in other words, the network received no additional information about positional change after the pattern began.

- **Fitness function and path efficiency term:** Bensmail, Steuber and Wróbel 2018 employed a fitness function similar in spirit to Bensmail, Steuber, Davey et al. 2017, but refined to incorporate a continuous measure of path efficiency. The fitness to be minimized was defined by Eq. 3.12:

$$f = 1 - \Big(\frac{T - 2D}{N_T} + c\frac{\alpha}{\beta}\Big), \tag{3.12}$$

where T, D, N_T, and the penalty multiplier (2 for distractors) have the same meaning as before. The new term $c\frac{\alpha}{\beta}$ represents a graded turning penalty based on the efficiency of the animat's trajectory: α is the straight-line distance from the animat's start position to the target's position, and β is the actual path length travelled by the animat (up to the point of target collection or the end of the trial). Thus $\alpha/\beta \in (0, 1]$ measures how direct the animat's route to the target was (1.0 for a perfectly straight path, smaller values for longer, indirect paths). The factor c was set to 0 for the first 100 generations and then increased to 0.5 for the remaining generations of evolution. By generation 101 onward, the fitness function

effectively became $1 - \left(\frac{T-2D}{N_T} + 0.5\frac{\alpha}{\beta}\right)$. This term penalizes meandering: an animat that drives straight toward the target gets a smaller penalty (since $\alpha/\beta \approx 1$) than one that takes a circuitous route (where α/β might be much lower). The inclusion of this continuous path factor (as opposed to the binary c term in Bensmail, Steuber, Davey et al. 2017) provided a smoother gradient for evolution to prefer straighter trajectories. With only one target per trial ($N_T = 1$) and one distractor, the best possible fitness in this formulation was -0.5 (if $T = 1$, $D = 0$, and $\alpha/\beta = 1$ yielding $1 - (1 - 0)/1 + 0.5 = 1 - (1 + 0.5) = -0.5$), while the worst (collecting the distractor and not the target with a poor path) could reach $f = 3$.

- **Evolutionary parameters:** We conducted 200 independent runs in this study (a larger number of replicates to gather more data on variability). The genetic algorithm used the same population size (300) and tournament selection (size 2) as before, but notably *no elitism* was applied (each generation's population was entirely replaced by offspring of selected parents, which prevents stagnation but can risk losing good solutions). Mutation rates for network structure were adjusted from the previous work: the duplication rate was set to 0.002 per genome per generation, and the deletion rate to 0.001 per genome. These rates (duplication slightly higher than deletion) still encouraged growth, but they were lower than in Bensmail, Steuber, Davey et al. 2017, reflecting a more conservative structural evolution. In practice, even though we allowed an unlimited number of interneurons in principle, the relatively short evolutionary duration and these mutation rates meant that networks did not grow excessively large (most evolved networks had on the order of 3–5 interneurons). Each run proceeded for up to 2500 generations. An early stopping criterion was implemented such that if an animat achieving $f < -0.15$ (a very high-performing individual) was evolved, the run would be continued for only 50 additional generations and then terminated, under the assumption that a near-optimal solution had been found. Fitness evaluation for each individual in a generation consisted of multiple trials: specifically, 18 total worlds were simulated (15 worlds with randomized object positions and a mixture of wrong patterns, followed by 3 worlds where the distractor consistently emitted only one type of wrong pattern, e.g. always "AA"). In each world, 200 pattern presentations were delivered (with target patterns comprising 30% of them in the mixed-pattern trials, and 50% in the single-pattern trials). The fitness values from those 18 scenarios were averaged to obtain the final fitness for the individual during current generation . This protocol ensured that the evolved networks were exposed to a variety of conditions, including different distractor pattern frequencies, further enforcing robust temporal pattern recognition.

3.4 Results 3: Chapter 6 - Unique Methodological Elements

This chapter examines the evolution of SNN controllers under internal noise and how this affects robustness and network topology. The basic evolution parameters were kept from Bensmail, Steuber and Wróbel 2018, with the following unique changes and additions:

- **Accidental Constant-Noise Evolution:**

Due to an unexpected problem with parameter modification in the MPI implementation in the HPC computing cluster, networks were evolved using a constant and low internal noise level (SD = 0.1 mV) for the entire evolutionary run, as opposed to an incremental noise schedule. In each simulation step (1 ms), each neuron's membrane potential received additional Gaussian noise (mean = 0 mV, SD = 0.1 mV). Although this was not our goal, the constant low-noise condition resulted in obtaining 50 independently evolved champion networks, which robustly provided important information regarding topological features and their role in temporal pattern recognition.

- **Network Topology and Evolutionary Parameters:** The network architecture and initial conditions were consistent with our previous study Bensmail, Steuber and Wróbel 2018, using two input neurons, two output neurons, and two interneurons. All possible connections could evolve freely, including recurrent and self-connections. However, the structural mutation parameters were adjusted in this study to emphasize stability: duplication mutations occurred at a rate of 0.001 per genome per generation, while deletion mutations were set at a much lower rate of 0.00001. The evolutionary runs were executed for 1000 generations, followed by an additional evaluation phase of 50 generations without further mutation to confirm performance stability.

- **Performance Metrics:** Two primary metrics were defined and applied:

  Rate of Curvature Change (ROCC): ROCC quantifies how quickly, on average, the animat trajectory is turning, and is inversely related to how smooth or straight the path is. For the purpose of this metric, the animat path coordinates are collected at every simulation time step. Denote the coordinates (x_i, y_i) for the position of the animat at the i^{th} step. The instantaneous turning angle θ_i at step i is formulated as:

$$\theta_i = \arctan 2(y_i - y_{i-1}, x_i - x_{i-1}) - \arctan 2(y_{i+1} - y_i, x_{i+1} - x_i)$$

  The absolute change in this heading angle between successive steps is:

$$\Delta\Theta_i = |\theta_{i+1} - \theta_i|$$

  The ROCC is computed as the average of these curvature changes over the trajectory, as shown in Eq. 3.13:

$$\text{ROCC} = \frac{1}{n-3} \sum_{i=1}^{n-3} \Delta\Theta_i \tag{3.13}$$

  ROCC has units of "radians per time-step," with lower values indicating smoother trajectories.

  Targets/Distractors Index (T/D): The Target-to-Distractor (T/D) ratio quantifies pattern discrimination as the ratio of correct target identifications to erroneous distractor

activations, with higher values indicating better segregation of relevant inputs.

These metrics were computed exactly as described and applied systematically across all testing conditions.

- **Unsupervised Clustering and Structural Analysis:**

  The seven topological features — excitation weight sum, inhibition weight sum, number of: excitatory self-loops, inhibitory self-loops and their total, interneuron connectivity (measured as the proportion of connections between interneurons), and sparseness (defined as the proportion of possible connections that are absent) — were extracted, standardised, and subjected to comprehensive unsupervised clustering (Elbow, Gap Statistic, Hierarchical, Consensus Multi-metric, Bootstrap Stability analyses). These features were selected to capture the balance of excitation/inhibition, the presence of recurrent loops (self-connections), the role of interneurons in network integration, and the overall density of connections (sparseness). To determine the relationship between topology and behaviour or performance, unsupervised clustering was applied to the seven extracted features, followed by standard clustering and correlation analyses (details in Chapter 6).

- **Robustness testing under Noise:** Each network was assessed with no noise (baseline), moderate noise at 0.5 mV, and high noise at 1 mV. Performance retention ratios were calculated to evaluate every cluster's robustness under noise.

- **Evolution with Incremental Noise (Experiment 2):** In the subsequent experiment, once we solved the MPI parameter issue, the incremental noise evolution began at 0.1 mV and increased by 0.05 mV every 25 generations, reaching a maximum of 1.4 mV by the 700th generation (a range consistent with biological membrane noise Faisal et al. 2008). Due to time constraints, evolutionary runs were terminated at 700 generations, irrespective of convergence status. Among the 40 independent runs, one strikingly robust minimalist network emerged which generalised performance at 2 mV noise and contained only two interneurons with strong self-reinforcing excitatory loops (> 5 units for weight). Additional work is necessary to guarantee reproducibility and determine more robust champions.

Chapter 4

Basic Pattern Recognition and Foraging (IEEE paper)

4.1 Experiment Setup

The evolutionary algorithm, network encoding, AdEx model, animat embodiment, and fitness function used in this experiment are described in Chapter 3. We followed the standard setup with a few adjustments:

- Evolution run for 50 independent trials using a population of 300 genomes for up to 2500 generations (or terminated early if fitness fell below 0.1 for 100 consecutive generations).
- Each animat was evaluated in 13 random worlds per generation per animat.
- The correct temporal pattern was always AB while the distractors emitted one of AA, BB, or BA at a time. For each pattern, we have two 8 ms signals separated by 3 ms silence and followed by 100 ms silence between patterns.
- Fitness penalised early distractor collection with a worst-case score to enforce robust discrimination. A penalty for circling ($0.2 \times c$) was applied to promote directional movement.

4.2 Robustness of Champions

Of the 50 evolutionary runs, 48 were successful (fitness values below 0.1, indicating strong task performance), while only 2 runs failed to evolve a sufficient solution (their champions had fitness values of 0.2 and 0.35). In successful runs, the evolved animats learnt to approach targets and avoid distractors: for example, Figure 4.1, shows a typical trajectory of a champion animat, which actively foraged to collect the target (black circles) while steering away from the distractor (red square). To assess how robust these evolved controllers were, the top 48 champions were systematically tested under various conditions beyond their evolution settings. We first ranked the champions by performance (the T/D ratio of targets (T) to distractors (D) collected in a standard test with 1 target and 1 distractor, over 1000 random worlds) and selected the top 10

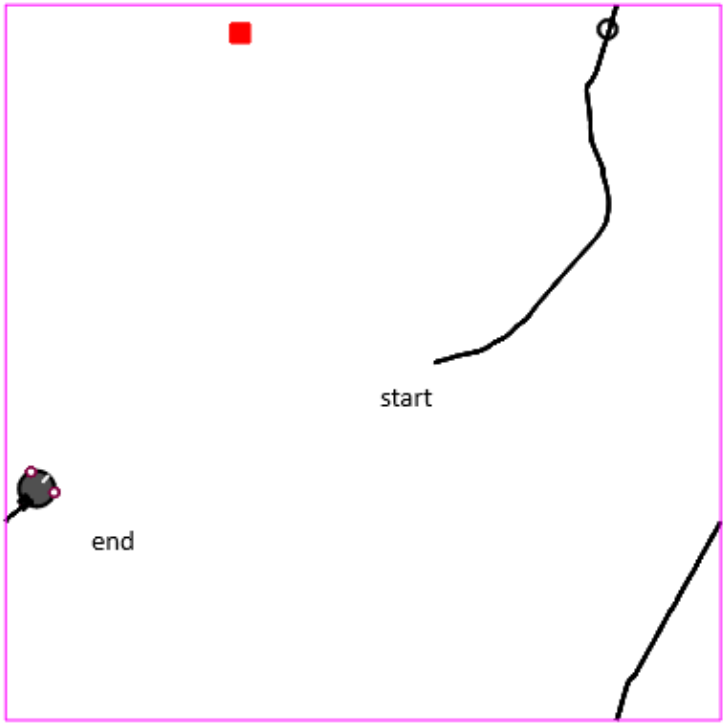


Figure 4.1: The trajectory of one of the champions obtained in a successful run, tested over 11 900 ms, which is equivalent to 100 patterns spaced by 100 ms of silence ($100\times(8+3+8+100) = 11\,900$), where AB is present 30% of the time. At any given time, there is only one target (black circle) and one distractor (red square); the empty circles show the positions of the targets collected during the evaluation.

champions for detailed analysis. Notably, all of these top 10 networks had 3 interneurons in their final architecture. Robustness tests examined whether these champions still prefer targets over distractors when certain aspects of the task or environment were altered.

The key findings on robustness are summarised below:

Varying target signal frequency: the rate at which the target emits the AB pattern (versus distractor patterns) was varied from 0% to 100%. All 10 top champions required a minimum frequency of the target pattern of about 20% to reliably outperform distractors. When the AB signal occurred with very low frequency (such as 10% or 0%), most champions collected as many targets as distractors or neutrals (objects that emit no signals; no information in the input stream). For example, at 0% AB (no correct target signals), champions collected targets as often as neutral objects; essentially random foraging. However, once the frequency of the target pattern exceeded 20%, the champions consistently achieved a T/D greater than 1, and their performance increased as the percentage of AB increased (see Figure 4.2). Some champions were exceptional: one champion (5) could still maintain a slight target preference even at 10% AB (for example, T/D $\approx$ 2.6 at 10% for that champion), suggesting a trade-off where it had evolved some tolerance for highly infrequent AB signals at the cost of a lower performance when the correct pattern occurred more frequently.

Distractor signal types (BA is the hardest): The champions were tested to see which specific distractor patterns (AA, BB, or BA) were most often mistaken for the target pattern

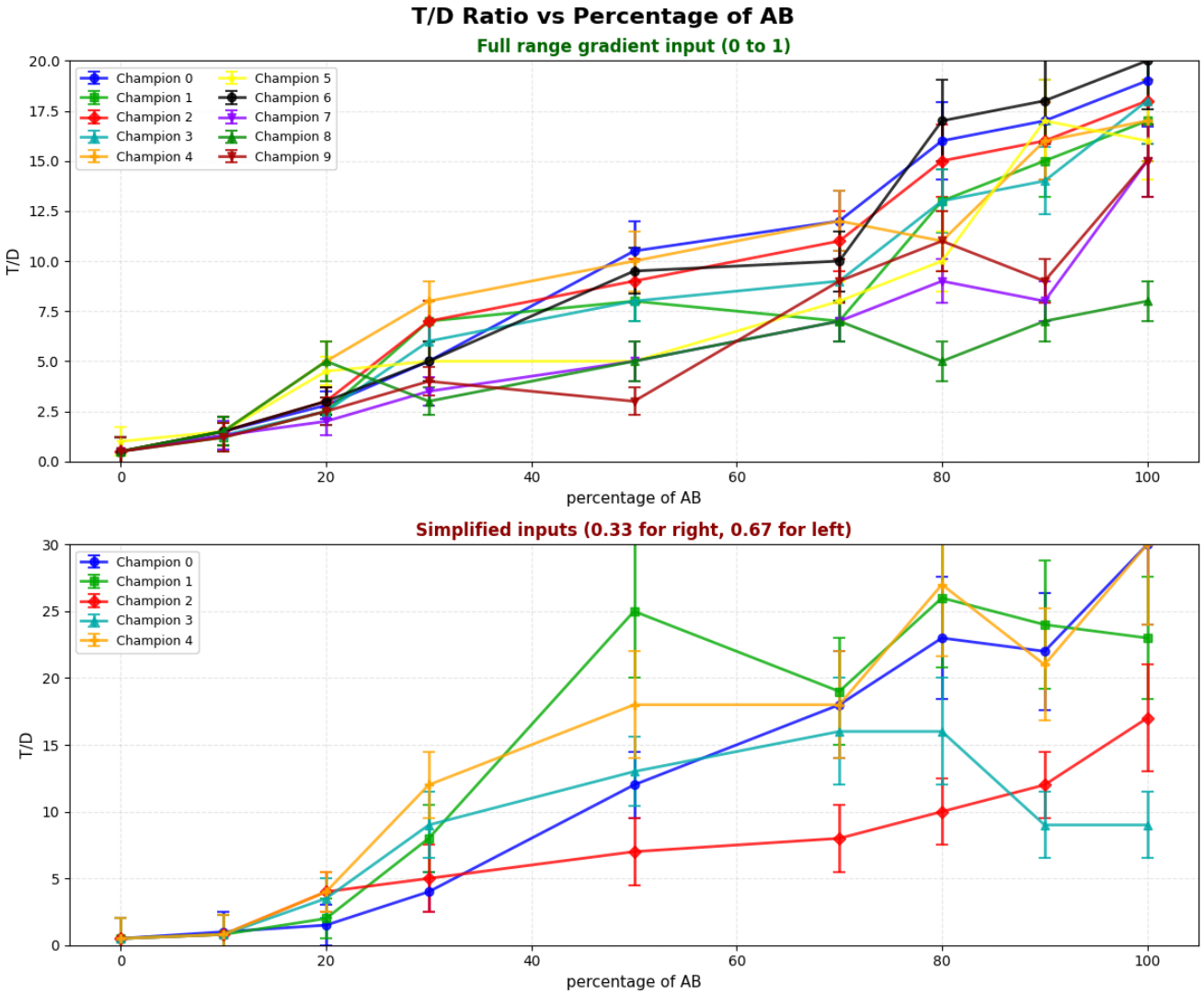


Figure 4.2: The ratio between the number of targets and the number of distractors (T/D ratio) collected on 1000 worlds for top 10 champions for various percentages of AB in the sensory input with full-range gradient input based on the sigmoid transformation of distance (top). To facilitate the analysis, each champion was tested with two simplified input intensities: 0.33 for right-side stimuli and 0.67 for left-side stimuli (bottom). For the zoom-in analysis, we focus on Champion 0, which is the best of the top 10 champions.

AB. Performance was generally worst when the distractor emitted the pattern BA (the inverse order of the target's signals). In fact, when the roles were reversed (i.e., the target emitted BA and the distractors emitted the other patterns), the ' performance of the champions decreased significantly: even at high frequencies of BA, they struggled to collect more than 1 target on average (indicating that they often could not find the target in most trials). Analysis showed that in this scenario the animats frequently failed to move towards the BA target, often remaining nearly stationary (in one test, the champion did not move in 80% of trials when the target emitted BA, whereas it would move in 90% of trials for the original AB-target setup). Additionally, it was observed that when the target emits BA, champions only succeed if that target happens to be on a particular side of the animat (the side corresponding to how the network has wired its response, e.g. only when the BA target was on the right side). This indicates a bias in the evolved

strategy: the network was tuned to recognise AB in a directional manner and finds BA much harder to recognise, especially from one side. The pattern BA (when coming from a distractor as per the original AB target setup) was also problematic: champions sometimes treated a BA distractor on one side as if it were a target (approaching it), but not if a BA-emitting distractor was on the opposite side (see the analysis below for why this side-specific confusion occurs). On the other hand, distractors emitting repeated identical signals (AA or BB) were easier for the animats to ignore; champions rarely moved towards those unless targets were extremely rare. We also briefly tested whether champions could recognise longer temporal patterns beyond two signals: for example, six possible triplet patterns (such as AAB, ABB, etc.). In those tests, none of the champions showed a reliable preference for targets emitting a longer 3-signal pattern (their T/D ratios never exceeded 1 for any of the novel triplet patterns), indicating that the networks evolved were specialised to recognise the exact 2-signal pattern and did not generalise to longer sequences.

Changes in object number and environment size: the champions evolved in a scenario with a certain object density (2 targets + 2 distractors in a toroidal world). To test robustness to different densities, the number of objects was varied. When the number of objects was increased (e.g., 3 targets and 3 distractors present simultaneously) without increasing the world size, performance dropped (the number of targets collected by Champion 0 dropped from approximately 0.98 to 0.35 per 1000 trials). A similar decrease in efficiency happened when switching to an open (non-toroidal) arena of the same size with more objects (the animat sometimes wandered without finding targets in the larger space). However, if the size of the environment was increased proportionally to keep object density the same as seen during evolution, the champions remained effective. Champion 0, for example, maintained mean target-collection rates between $T \approx 0.85$ and $T \approx 1.20$ and mean distractor-collection rates between $D \approx 0.12$ and $D \approx 0.18$ across configurations with up to six targets and six distractors in a larger world (so that $T/D > 1$ in all cases).

Even when the distractors outnumbered the targets (up to four times more distractors), the champions still preferentially collected the targets (keeping T/D>1). Conversely, if targets greatly outnumbered distractors (e.g. 50 targets vs 1 distractor), champions simply collected more targets roughly in proportion to their availability (e.g. tens of targets per trial) with only a few distractor hits. These tests show that the controllers are robust to increased object number, provided that the arena is not drastically more dense than during evolution. This experiment was conducted on a balanced evolution setup of 2 targets and 2 distractors. Generalisation was evaluated by scaling the object count and arena size around this baseline. Other density configurations were tested but did not yield additional insights; hence, they were not documented. This analysis includes only combinations with a complete and reproducible experiment. It remains possible that other target/distractor ratios could reveal further generalisation behaviours in these champions.

Actuator strength robustness: we attempted to generalise different movement speeds by modifying actuator strength (thrust per spike). The baseline actuator force used during evolution was 2.0; here, all multipliers (e.g., 1×, 2×, 3×) refer to scaling relative to this baseline. As shown

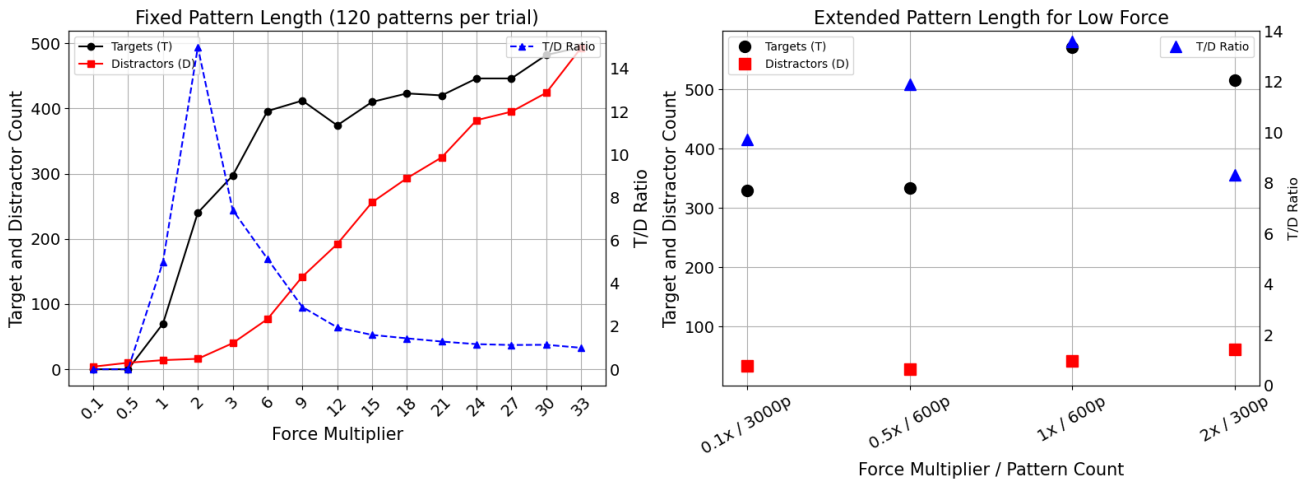


Figure 4.3: Animat performance across varying actuator force multipliers under two testing setups. **Left: Fixed Pattern Length (120 patterns per trial)**; X-axis: force multiplier (relative to baseline thrust); left Y-axis: number of targets (black circles) and distractors (red squares) collected; right Y-axis: T/D ratio (blue dashed line). T/D peaks at 2× force. At higher forces, distractor collisions increase and T/D ratio decreases. **Right: Performance at Lower Force Multipliers with Longer Trials**; Performance improves when slower animats are allowed longer simulation time. For example, in "0.1× / 3000 p", the actuator force is 0.1× baseline and the animat is given 3000 patterns. These longer trials allow even slow-moving animats to sense and forage effectively, resulting in higher T/D ratios.

in Figure 4.3, the performance of the animat was strongly influenced by the actuator force. With force multiplier lower than 1; target collection was very small because the animat could not reach the objects in time. There was a sharp performance improvement from 1× to 3× with a peak around where the animat collected the most targets with a strong preference for targets over distractors (T/D $>$ 7). However, at higher forces (above 3×) there was a significant increase in distractor collisions, leading to a decrease in the T/D ratio despite the fact that target collection remained high. A force multiplier of 2x was set for all subsequent experiments to optimise target collection efficiency while minimising distractor collisions. We kept the standard simulation length of 300 patterns in later experiments to reduce computational cost. Although some tests showed that excellent performance could be achieved at lower force levels and longer input sequence durations; allowing slower animats more time to sense and react; we chose to set the force multiplier to 2. This decision maintained efficiency while still demonstrating the robustness of the evolved networks across different actuator strengths and input dynamics. Note that the networks evolved under fixed conditions (force multiplier = 2); only post-evolution tests involved varying the actuator strength.

Sensor sensitivity and neural gains: The baseline networks (Figure 4.5) have excitatory and inhibitory synapses from sensory nodes to interneurons with certain gains (weights). We effectively 'lesioned' the networks by removing or reducing these inputs. Removing the single inhibitory connection from the sensory layer (setting its gain to 0) did not completely ruin the behaviour. The animat still showed directed movement toward the target, even with lower efficiency ($T ≈ 0.32$, $D ≈ 0.14 for champion 0). Reducing the excitatory sensory input gain from

3.0 to 2.5 nS caused the animat to move much more slowly, but it still managed to find targets reliably (in a longer simulation time, it could show performance nearly as good as shown for the gain it was evolved for). However, if the sensory gain was reduced below a certain threshold, the network would cease to produce all movement. In summary, the champions do rely on a minimum strength of sensory input to trigger their turning behaviour, but they do not require finely tuned precise values - some reduction can be compensated (by slower movement or longer search time), and even loss of an inhibitory input can be tolerated, demonstrating a degree of fault-tolerance in the network. This behaviour was observed consistently across the top champions, indicating shared robustness properties that do not depend on a narrow range of input strengths or exact circuit tuning. These findings tie into broader themes of resilience observed in evolved networks and will be further supported by the robustness tests discussed below.

Temporal signal parameters: the evolved networks were tested for robustness to changes in the timing of the input signals. During evolution, the two signals in a pattern were separated by a 3 ms gap. The tests showed that champions could handle even no gap (0 ms, signals presented consecutively without interruption) or shorter gaps (1-2 ms) between A and B; they still preferred to collect targets in those cases. They also handled moderately longer gaps: increasing the inter-signals gap to 4, 5 and 6 ms had little effect on T/D (remained above 1). However, when the gap was 7-9 ms; comparable to or exceeding the duration of the signal (8 ms); the performance started to degrade (e.g., at 9 ms the champion 0 had T/D decreased but remained slightly above 1). Very large gaps beyond 10 ms caused the network to lose its ability to discriminate targets from distractors. Thus, the controllers have some tolerance for variation in the spacing of the two signals within the same pattern, but only up to the duration of the signal.

Introducing random gaps within a signal (e.g., turning off the signal for 1 ms out of the 8 ms) was also highly disruptive - a single 1 ms interruption in each 8 ms signal led to almost no target collection (T/D fell to almost 1 or below). We note that this is expected because such a noise effectively reduces the total energy of the stimulus (7 ms of actual signal instead of 8 ms). Interestingly, if the total active duration per signal was kept the same (8 ms) but spread over a longer time window (for example, 8 ms of active signal within a 12 ms window, i.e. making the signal sparser but not shorter in total), the network was much more robust. In these tests, even with signal slots extended to 12-16 ms (with only 8 ms of the time window in which the signal was present), the champions still showed strong target preference. This indicates that the network's neurons responded more to the total integrated stimulus than to the precise timing, as long as the overall amount of stimulation per signal was unchanged. We also found that intolerance to shorter signals (7 ms) is partially alleviated by removing the inhibitory sensory link, suggesting that the particular connection made the timing more rigid.

During evolution, a long silence (100 ms) was set between each two successive patterns to avoid interference. To test whether the champions could handle faster or continuous streams of patterns, the gap between patterns was reduced. With 50 ms interpattern intervals, champions performed about the same as with 100 ms. Even at only 3 ms between patterns (meaning patterns follow almost immediately one after another), champion 0 still showed a slight but non-

random preference for targets (collecting more targets than distractors). In this extreme case, the performance was much lower (T/D only slightly over 1) and interestingly the champion would only reliably move towards the target if it appeared on a specific side (left side). This suggests some residual capacity (for champion 0) to recognise patterns in a continuous stream, although not robustly. However, increasing the silence between patterns to a very high value (300 ms) did not affect performance; in fact, the champion still performed well and could even improve its success rate when the signals of the target pattern were frequent. Figure 4.4 shows boxplots of target and distractor collection counts (relative to neutral objects -those that emitted no signal) for champion 0 under a 300 ms pattern gap, confirming that it strongly favours targets across 1000 trials.

Overall, the champions demonstrated a high degree of robustness to many variations: they forage in larger, more populated arenas (if density is similar), can handle moving faster, and are relatively tolerant to noise in sensors and moderate timing changes. They are less robust to large changes in the temporal pattern structure (reordering or substantially lengthening/shortening signals) and to very low prevalence of the target pattern. As noted in the paper (Bensmail, Steuber, Davey et al. 2017), the evolved controllers remained robust to changes in environment, actuator strength, and pattern intervals, but showed limitations when sensor input strengths were reduced or signal timing was heavily altered.

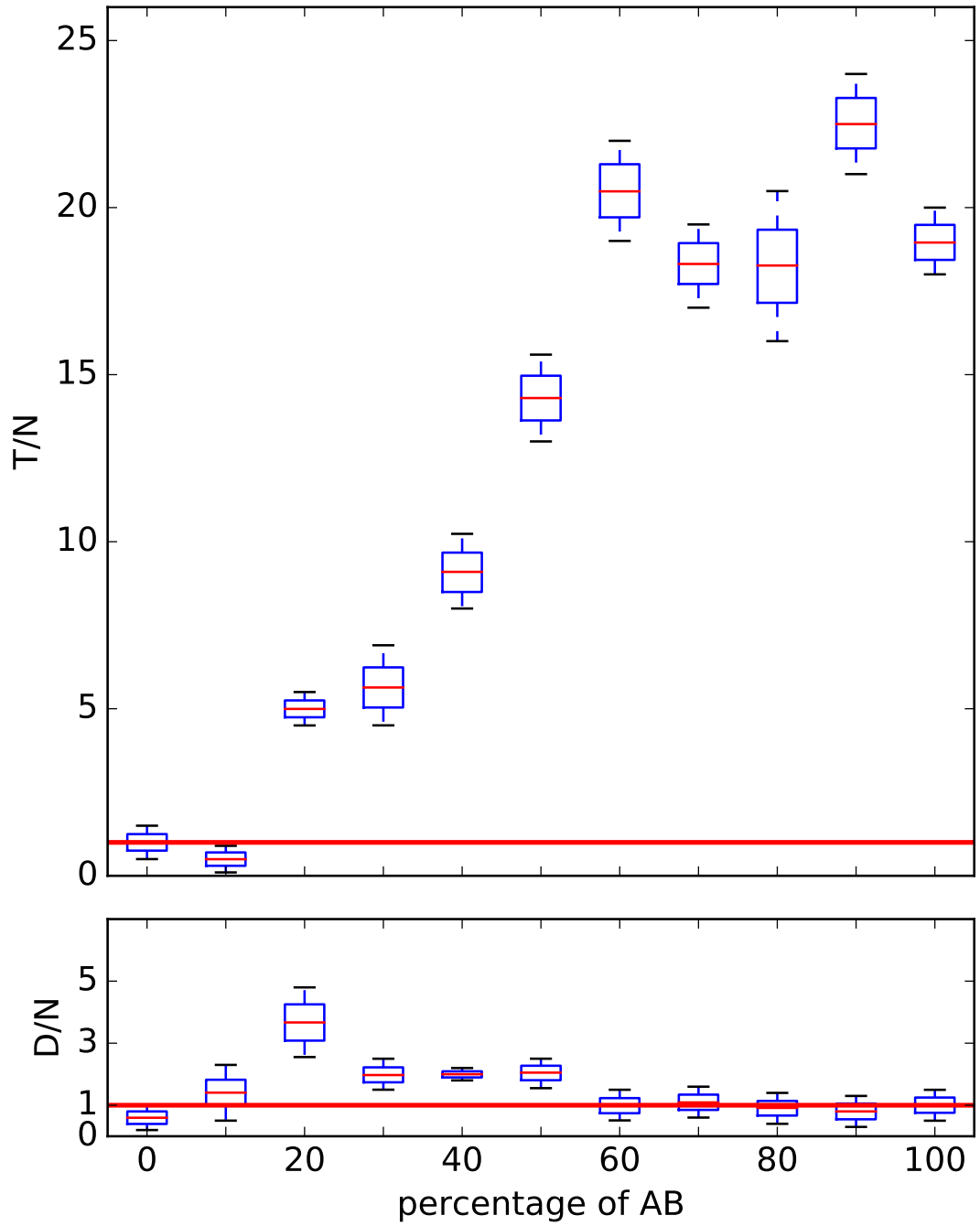


Figure 4.4: The ratios between the number of targets or distractors collected over 1000 worlds, compared with neutral objects (T/N or D/N, respectively) for champion 0 with a 2-level sensory nodes and 300 ms silence between patterns. The box sizes indicate the interquartile regions (with medians marked in red), the whiskers mark the lowest and highest data point still within the 1.5 interquartile region of the lower and upper quartile. Ratios above 1 (red horizontal line) indicate preference for target or distractor; values below 1 indicate failure to recognise target pattern over distractors.

4.3 Analysis of the Best Champion

To understand *how* these evolved networks make decisions, we performed a detailed analysis of champion 0 (the network with the best performance). All top champions behaved similarly in robustness tests, so champion 0 is representative. This network had 3 interneurons (denoted N1, N2, N3) in addition to the 2 sensory neurons (A, B) and 2 motor outputs (A_L for the left actuator, A_R for the right actuator). The analysis was facilitated by simplifying the sensory inputs: instead of a continuous range of sensory intensity values ([0,1] interval output of the sigmoid normalisation of the distance gradient), we discretised the sensory node output into just two levels (representing signal source on the left vs on the right). Under this 2-level input setup, there are only $2^2 = 4$ possible signal combinations for a single time step (each of A and B high or low), and since a pattern consists of two signals in sequence, there are 8 possible distinct input patterns (two signals A or B with each being emitted from left or right). This greatly simplified the analysis because the response of the network can be examined for each of these 8 cases. We verified that the behaviour of the network with 2-level sensors was consistent with the full sigmoid range setup (the discretization did not change the performance T/D), so insights from this reduced scenario should carry over to the real continuous-input setup.

- **Target vs distractor motor response:** The evolved network exhibits a clear pattern in its motor output to achieve pattern recognition. For the target pattern AB, the network triggers an *asymmetric* response in the two output neurones depending on the location of the target. Specifically, when the target (emitting AB) is on the left side of the animat, the right output neuron (AR) fires more strongly than the left (AL), causing the animat to turn left towards the target. In contrast, if the target is on the right, AL fires more than AR, steering the animat right toward that target. Figure 4.5 shows the network topology of champion 0 and confirms this behaviour by plotting the voltage traces of all neurons for each of the 8 stimulus patterns: In fact, in the AB cases, the output corresponding to the opposite side of the source has a higher spiking activity. In contrast, for all distractor patterns (incorrect patterns AA, BB, or BA), the two outputs fire roughly equally or not at all, resulting in no directed turning. Essentially, the network ignores distractors by not creating a left-right imbalance in its output spikes. The only exception was the BA pattern (correct pattern signals reversed) coming from a specific side: in this case, one output neuron fires, causing the animat to turn toward a BA source on that side. This explains the earlier observation that distractors emitting BA were sometimes collected when on one particular side. But for BA on the opposite side, or other patterns, the outputs remained equal (or silent), and the animat does not directionally approach them. For example, when the distractor emitted AA (repeated A), the left and right outputs of champion 0 produced the same number of spikes, which means that the animat moved forward rather than turning towards the source AA. This forward movement in the presence of AA is effectively a neutral behaviour (no attraction or active avoidance), and if the animat hits an AA distractor, it is mainly by chance.

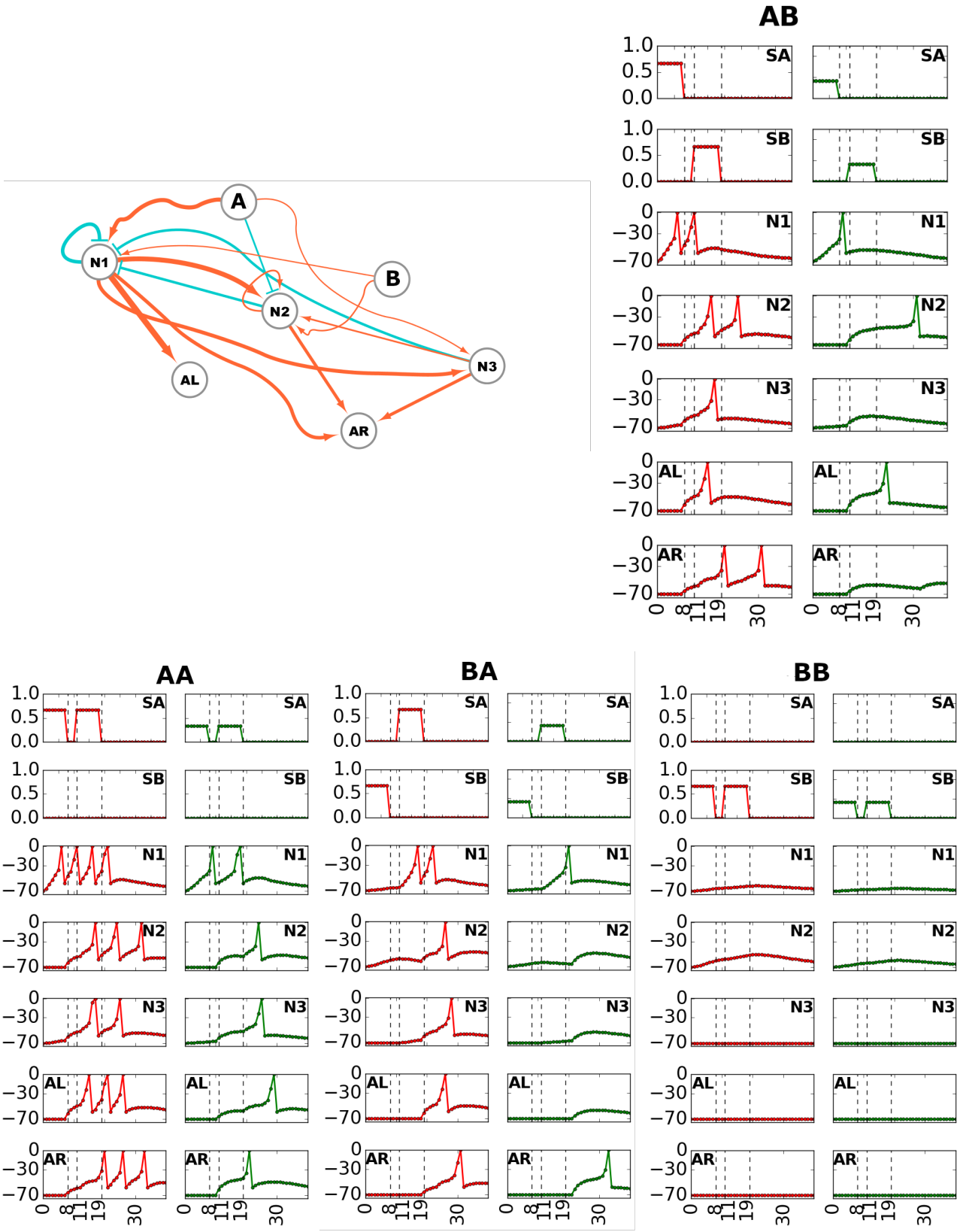


Figure 4.5: The network topology of champion 0 and its response to the 8 possible stimuli for 2-level sensory nodes. The voltage traces (vertical axes: voltage in mV, horizontal axes: time in ms) are red (green) for the source on the signals on the left (right). Excitatory links are shown with arrow heads (orange), inhibitory are bar-headed (cyan).

- **Internal network dynamics:** the champion's three interneurons (N1, N2, N3) have

evolved specialised roles in processing the temporal pattern. In general, the network forms a feedforward structure where A (sensor for signal A) is the dominant input driving the initial reactions. Interneuron N1 receives a strong excitatory input from A (and a weaker one from B). As a result, whenever an A signal is sensed, N1 spikes immediately (especially if A is at high intensity, i.e. coming from the closer left side). N1 is always the first interneuron to fire for any pattern starting with A, acting as a fast A detector. This spike from N1 is directly passed to the motor outputs: notably, AL (the left actuator neuron) has an excitatory synapse triggered primarily by N1. Therefore, a single spike in N1 (triggered by sensing A) can cause AL to fire, which would turn the animat right (towards where the A was emitted from). This is exactly what happens for the first signal of the target pattern AB: if A is detected, the animat immediately starts turning toward it, which is a correct response since A is the first part of the target pattern. The other output neuron, AR, does not receive such a strong direct input; instead, it only fires after it accumulates input from multiple interneurons. It has moderate excitatory inputs from N1, N2, and N3. AR effectively integrates input only when the interneuron layer is activated together. For AR to spike, typically N1 must fire multiple times or N1 together with N2 (or N3) must fire. This design ensures that AR (which triggers left-turning) only fires during a full AB sequence. For instance, in response to the complete AB pattern coming from the left side, N1 will fire (due to A), then as B arrives, N2 and N3 fire as well (since B plus the prior A activity push their membrane potentials over threshold), and together their inputs cause AR to fire, resulting in a left turn. If the pattern is not AB (say just an A), AR might not get enough input to spike, whereas AL might fire one spike from N1 input but that alone does not drive the animat far on the path. Interneuron N2 in champion 0 receives a mix of inputs: a significant inhibitory input from A and excitatory inputs from N1, B, and N3. N2 often fires after a short delay, acting as a coincidence detector and timeout mechanism. For example, when AB is received, N1 rapid spikes followed by B input cause N2 to fire a second spike shortly after, reinforcing AR activation. If A was emitted without a B (as in AA pattern), N2 might be suppressed due to the absence of B and the inhibitory A input, firing only once or not at all. Interneuron N3 has a high threshold, needing strong or repeated input to fire: it receives a strong excitatory input from N1 and a weaker one from A. In many cases N3 will only spike if N1 fires multiple times in high rate (which happens for a long A stimulus or repeated A signals). In the AB scenario, N3 might contribute a spike if A was strong or if A and B together trigger a postsynaptic response. For distractor patterns, N3 usually remains quiescent (it fails to respond to wrong patterns). In summary, N1 is immediately activated by A, N2 adds a delayed reinforcement especially when B follows A, and N3 responds only to sustained input (which could correspond to the repeated A in AA or a very strong A). The combined effect is that only the correct sequence A followed by B (emitted from the corresponding direction) produces the distinctive pattern of interneuron firing that leads AR (or AL) to fire strongly, pushing the animat toward the target. Other patterns either produce a symmetric output activation (forward movement) or insufficient

output spikes to make a turn.

- **Verification under continuous input:** to confirm these findings under more realistic conditions, we also analysed champion 0 responses with the original evolution setup: full range of sigmoid outputs and without the large between-patterns gap. We recorded the spiking of the network while presenting various patterns. The same behaviour was observed. For example, when the target pattern AB was emitted from the right side, AL fired one spike shortly after activation (the animat turned slightly to the right), and then, as the pattern was completed (B received), AR fired more, causing a left turn. This occurred because the animat's initial rightward turn changed its orientation, so that the same stationary target was now received from the left side. If the target was very strong (close, from left side), both outputs fired multiple spikes but still with AL spiking fewer than AR, resulting in an overall turn towards the target's location. For a distractor such as AA, the network tended to fire both outputs equally (or not at all) across a range of signal intensities, so the animat would move mostly straightforward, and only increasing its speed as the stimulus increased (because both wheels have equal input from actuator forces). In some cases with very high stimulation, a slight turning away from the AA source was observed, but generally no strong reaction. For the BA distractor, if the distractor was on the right, the network sometimes produced the pattern of spikes that would turn it right (mistakenly approaching it); but when the distractor was on the left, the outputs difference used for turning did not occur except at very high intensity ranges. These detailed observations align with the simplified analysis: champion 0 neural circuit is tuned to accurately forage towards a target emitting AB, and this mimicked a phonotaxis-like behaviour (turning toward a calling target) while consistently ignoring other unwanted stimuli. Minor asymmetries, such as the one-sided response to BA, are a result of the network imbalanced in how connections are distributed - in particular, its strong reliance on input A and the subsequent influence of N1 on a single output neuron.

In conclusion, the best champion strategy can be summarised as a form of temporal pattern recognition: the network uses a fast feedforward mechanism to respond to the presence of signal A, but only performs a complete turning movement if that A is followed by a B (hence, forming the correct pattern AB) at the right timing. The interneuron layer creates a short transient memory of A and requires subsequent B to actively trigger the second output neuron. In this way, random or incorrect patterns do not trigger a turning response, whereas the correct one does. The analysis of champion 0 spiking network provides insights into how a very small SNN (with just 3 interneurons) can solve a non-trivial temporal recognition and navigation task. We note that the simplifications in the task (e.g., discrete signals and enforced pattern gaps) were specifically chosen to make such an analysis feasible, and future work would involve extending these principles to tasks involving more continuous input (no gaps between patterns/signals).

Chapter 5

Pattern Recognition with Variable Silences (ICANN paper)

5.1 Experiment Setup

This experiment builds on the temporal pattern recognition task from Chapter 4 by adding variable silent gaps between stimuli. The purpose is to see if the evolved animat controllers can maintain discrimination performance when the regularity in time is removed. Animats depended on fixed intervals to identify the AB target pattern in the preceding setup, likely resulting in a significant reliance on internal timing mechanisms. In order to counter this, we systematically altered the silence durations on a per-trial basis between two signals of the same pattern. The animat's body structure, environment, and signal types remained the same. Moreover, the internal controller still employs a spiking neural network of AdEx neurons evolved with the same genetic algorithm.

In this chapter we distinguish: (i) the **intra-pattern gap**: the silence **between the two signals within one pattern**, and (ii) the **inter-patterns gap**: the silence **between successive patterns** in the input sequence. Unless stated otherwise, references to a "gap between two signals" mean the intra-pattern gap.

5.2 Pattern Discrimination Efficiency of the Evolved Animat

Following 200 independent evolutionary runs, 20 runs produced champion networks that produced a final fitness lower than the set threshold (fitness < -0.25). Of these 20 top performers, more scrutiny was conducted to check their ability to collect target items while avoiding distractors. The network that performed best out of all champions demonstrated perfect pattern recognition abilities, as it collected $T = 1000$ targets and made no distractor errors $D = 0$ out of a possible 1000 test world. This best-performing network (winner), having 4 interneurons in its architecture, uses simple goal-directed navigation without circling (Figure 5.1 illustrates its trajectory

as a smooth sweeping line towards targets). In contrast, other champions had network sizes ranging from 3 to 12 interneurons and displayed more complex behaviours with more circling or staying still in specific worlds. This four-neuron winner was chosen for closer analysis because its consistent non-circling trajectory makes the results easier to interpret and it reached maximum performance.

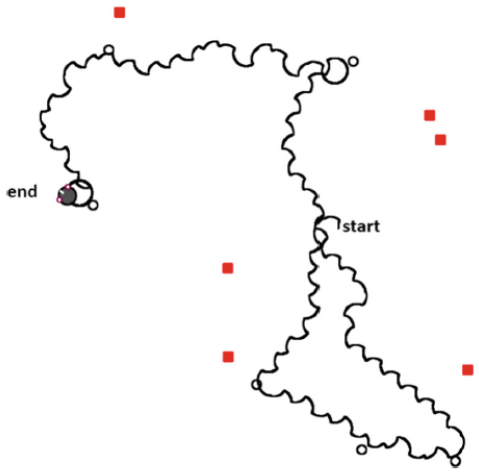


Figure 5.1: Visualisation of the performance of the winner. The test world has 6 targets (black circles) and 6 distractors (red squares), the black sweeping path is the movement trajectory of the animat.

The winner's performance was then tested with different pattern distributions and timing to evaluate its efficiency in distinguishing the correct AB pattern from distractors. Even when the desired target (AB) pattern was overwhelmingly difficult to find (constituting only 1% of the input patterns), the winner was still able to get to the target in 700 out of 1000 worlds (70% success rate). The success rate increased significantly as the correct pattern became more common: With the correct pattern present only in 10% (90% of the patterns being distractors), the animat was able to collect more than 960 to 1000 targets per 1000 attempts (>96% success rate). After the correct pattern appeared more than 10% of the time, the performance was almost perfect. This demonstrates the ability of the network to cope with an extremely large number of distracting patterns while still locating the target, although performance is negatively affected by extremely low-frequency targets.

We also explored how changing the temporal features of the patterns, particularly the silence gap between two consecutive signals , would affect performance. In evolution, the gap between two signals of the same pattern followed a Poisson distribution with a mean of 30 ms; the winner network has adapted to this distribution. During the tests, when we set the silence gap between 17 and 43 ms (which is close to the evolutionary mean of 30 ms), the animat performance remained nearly perfect (almost all targets were collected and almost no distractors were hit). However, the performance decreased sharply when the gap between the two signals of the same pattern was extended: at a 45-ms silence, the winner network failed to collect any targets. This sharp cut-off indicates a cap to the state memory of the network -if the gap is too long, the network can no longer remember the two signals. Notably, the winner's performance also decreased, although

more gradually, for very short gaps.

Even with a 0 ms gap, the winner animat was still able to collect some targets, 346 out of 1000. The degradation in success for gaps under roughly 17 milliseconds is caused by spike coincidence detection, suggesting that the winner requires a spike from the interneuron N0, which is triggered by A, to overlap in time with the spike fired by N1, which is activated by the input B signal. If B is emitted earlier, the two spikes may no longer arrive within the coincidence detection window, and the required joint activation might stop occurring. Thus, there is an effective temporal gap range: too long a gap, the memory of A is lost, and the signals merge so that the network logic can no longer be retained.

Next, we investigate the effect of silence intervals between consecutive patterns (i.e., the time separating the end of one pattern and the start of the next in the continuous input stream), see Figure 5.2. In particular, long silences between patterns (on the order of hundreds of milliseconds, up to 1000 ms tested) did not affect animat's performance–the winner still collected far more targets than distractors, maintaining a high average T/D ratio above 50 in all cases. This indicates that the network maintains its pattern recognition capability despite extended silence between patterns. On the other hand, when the silence between patterns was very short (less than 40 ms), the number of targets collected by the animat decreased (showing that it had some difficulty when patterns arrive in rapid succession). Importantly, even under rapid stimuli input, the animat consistently avoided distractors, although it failed to respond to some targets, yielding a lower count of total targets but still zero distractor hits. This suggests that under extremely high frequency patterns in the input sequence (patterns coming one after another with minimal gap), the controller sometimes fails to react to every target in time, but it still reliably avoids wrong patterns; in other words, it trades off sensitivity for specificity when overwhelmed by highly frequent stimuli.

Finally, we tested the impact of changing the duration of each signal in the pattern. The default signal length in evolution was 10 ms. Shortening the signal pulses below this (e.g., 5 ms) caused the animat to circle in place or not move at all, indicating that the signals became too brief for reliable network processing, making the patterns undetectable. Slightly lengthening of the signals, on the other hand, was tolerable: for example, increasing the duration of each signal to 13 ms still allowed the animat to perform a fair recognition of the pattern, achieving a T/D of $997/128 \approx 7.79$ (that is, it collected almost 1000 targets while hitting 128 distractors over 1000 trials, a substantial, though not perfect, discrimination ability). This demonstrates a degree of robustness in timing: The network can handle some variation in stimulus length, but there are limits beyond which performance degrades.

Neutral objects: Where neutral, non-emitting objects were included as a control, targets were still preferentially collected, and neutrals were not actively approached nor avoided (qualitatively consistent with Chapter 4); distractor avoidance behaviour persisted. When neutrals were not part of a given ICANN test configuration, results are reported against distractors to ensure consistency with that setup.

5.3 Robustness of the Winner to Environmental Changes

The winner's performance was evaluated with respect to its generalisability and we conducted a series of robustness tests to determine how it would perform in evaluations beyond the environment experienced during evolution. One of the tests involved increasing the number of objects present in the arena. The original setting had a balance of targets and distractors (6 of each in Figure 5.1). We increased this to 10 distractors and 10 targets per world. The winner maintained a high discrimination ability in these denser worlds, with T/D ratios of 12-18 (originally ∞). This indicates that even when the environment was cluttered with objects, the winner was still able, on average, to collect 12-18 times more targets than the distractors, which shows scalable performance.

Next, we analysed an extreme case of imbalance: just 1 target and 20 distractors in every world. At this level, the animat faced substantial efficiency challenges due to a large imbalance in target-to-distractor ratio. Interestingly, it was able to collect 999 out of 1000 targets (99.9% of targets) over 1000 trials while collecting only 170 out of 20,000 distractors (0.85% of distractors) encountered. This gives approximately a T/D ratio of 5.88. While the ratio is lower than in the balanced case (due to the absolute number of distractor hits increasing because there were so many opportunities), the point that matters is that the animats hardly ever miss a target in the presence of numerous distractors and continue to collect targets at a high rate. Practically, this demonstrates the network's ability to effectively locate rare targets within environments densely populated by distractors.

We also analysed how changes to the animat's actuator strength impacts the performance of the system. The actuator power of 2 produces the smooth trajectory seen in Figure 5.1, this is the power under which the evolution run was conducted. When doubling both actuator forces, the animat managed to achieve $T = 1000$ with $D = 44$ in the trials (T/D ≈ 22.73). During even more extreme and rapid changes, where we increased the actuator force by a factor of 33, the network still had its ability to discriminate: It obtained 889 of 1000 targets and 85 distractors (T/D ≈ 10.46). Although it is true that the success ratio dropped with more extreme forces, it is still significant that the animat was able to perform the task with such high actuator strengths. It is intuitive to assume that increased actuator strength might lead to uncontrolled movements and increased errors, but the neural controller modified its output (spiking patterns) to ensure that the behaviour remained effective. This means that the motor control is robust: the mechanism for orienting and moving toward signals' sources is not very dependent on the precise motor calibration.

The network was subsequently tested with more complex patterns than those presented during evolution. In particular, we presented the AAB three-signal pattern as the target pattern, with distractors producing BA, BB, and AA two-signal patterns. The animat was having difficulty: a T/D less than 1, meaning it was getting fewer targets than distractors. This failure indicates that the evolved network was not performing well because the internal decision logic of the network was tuned to the AB signal pattern. It is difficult to extend to a longer sequence that includes a repeated signal, which is AA followed by B. The additional A in front is likely to confuse the

internal state mechanism.

However, the network performed better with patterns that consisted of a sequence of Bs placed after an A. As long as any pattern that started with A (and at least one B followed it) was marked as the target, the network could still accurately recognise patterns to some degree by achieving T/D ratios of 6 to 19 depending on the Bs. In these cases, if the pattern begins with an A the initial state of the network is set correctly and it is usually possible for the network to understand the chain of Bs as though they are the single B seen in evolution, responding with the correct action at least for the first B and at times continuing to approach if many B signals are received. The performance does degrade as the pattern lengthens (T/D closer to 6 for longer sequences), but the fact that it remains above one signals generalisation: the controller employs a strategy that is not confined to just one B.

Finally, we evaluated how robust the winner's controller is to changes in its internal parameters. We systematically varied synaptic strengths and neuron parameters to find the maximum amount of perturbation the network can sustain without becoming dysfunctional. When we reduced all inhibitory synaptic conductances to half of their original value (0.5 nS, as opposed to 1.0 nS), the network worked quite well, yielding $T = 1000, ; D = 77$ ($T/D \approx 13$) doubling the inhibitory synaptic strength to 6 nS also kept the performance high: $T = 1000, ; D = 101$ ($T/D \approx 9.90$). These results indicate some robustness in synaptic inhibition, demonstrating that the logic of the network is not based on a very precise inhibitory gain. It can operate with both weaker and stronger inhibition without significant decay (some more distractors are being hit as inhibition increases, presumably because the network has become slower to spike in response to the correct pattern, giving distractors more opportunity to be hit). However, in sharp contrast, the performance was not very robust to excitatory synaptic gains. Any small adjustment to the excitatory synapse strength caused T/D to drop to roughly one. The excitatory connections in the network are obviously very optimised, since even small changes create a condition where the timing or the activation requirements of the pattern recognition sequence are bypassed and the animat collects as many distractors as targets (failing to discriminate). Most changes in intrinsic neuronal parameters could not be made without compromising the controller's behaviour.

Early qualitative checks showed a simple trend: when we shorten the synaptic or membrane time constants (e.g. τ_E, τ_I, τ_m), the "A then B" coupling becomes more tight-windowed. The network then tolerates far less variability in the gap between A and B. When these constants are longer, that window opens up, but at the cost of catching the wrong coincidences from distractors. Similarly, increasing the adaptation time constant (τ_w) or making the reset stronger slows the rate of recovery of the circuit, so patterns that sit close together in time become harder to separate. Slightly reducing those values does the reverse.

The sole exception was the specific membrane reset potential (V_r). Even when tuning V_r by $\pm$3mV for all neurons (from -58mV shifted to -55mV or -61mV), performance was above 1: within these constraints, the animat collected more than two times more targets relative to the distractors (T / D $\approx$ 2.65). This, combined with collecting far fewer targets than optimal, suggests that there is some tolerance to changes in neuronal excitability and that the network

showed a weak preference for correct patterns when the neuron reset voltage was slightly higher or lower. Performance dropped to nearly zero (T/D $\approx$ 1 or worse) almost instantly with other adjusted parameters such as the firing threshold and the membrane time constants.

In summary, the winner's controller showed resistance to many external and some internal changes (object density, actuator strength, moderate pattern changes, and considerable shift in inhibitory strength or reset voltage); however, it is shaped by the finely tuned excitatory synaptic weights and neuron parameters it uses for its core pattern recognition logic.

5.4 Analysis of the Network

We carried out a detailed analysis to understand how this winner's spiking network achieves the behaviour. It consists of four interneurons (N0, N1, N2, N3) and two output motor neurons (AL and AR that command the left and right actuators, respectively). The best-performing network is shown in Figure 5.2, including spike trains for the four input patterns at two levels of sensory input (when the source is on the left and the source is on the right).The key element of this analysis is that this network has emerged as a mechanism of internal state: when an animat detects signal A from a given side (say the left side), N0 initially fires and perpetually self-excites, and, N0 goes into a sustained-low firing after A ceases. This sustained firing mechanism within N0 for A remembers that an A was recently received from that side. In case the next signal to be processed on that same side is signal B of the AB pattern, then the network is set to respond accordingly.

B will cause interneuron N1 to fire and since N0 is already active, together, these inputs trigger interneuron N2 to start spiking (its self-excitatory loop allows it to persist in firing once activated). N2 acts as an essential key neuron: provides an excitatory trigger to the motor neuron on the contralateral side (AR if the pattern comes from the left side or AL if from the right side). In our left-side case, when N2 starts spiking, The animat rotates left, which causes AR to be more active than AL and during that time the right actuator is firing, which is what is helping the animat in moving leftward. So N2 is indeed the main neuron that activates AR in this network; therefore, sustained activity of N2 results in a prolonged turn to the left. This accounts for the sweeping trajectory towards a left-source target: the animat constantly rotates left when N2 is on, and it keeps turning until the sensor orientation has changed so much that the pattern is no longer coming from the left.

In this scenario, the animat has to turn sufficiently, so that the appropriate pattern signals begin to activate the correct sensor (the lower level of the left input and the higher level of the right input). Once the pattern comes from the right side, the dynamics of the network remains the same, except that they are reversed: the sustained activity of N0 and the subsequent firing (1 spike) of N3 result in AL (the left actuator) spiking, which causes the animat to turn to the right. This is due to N2 remaining quiescent due to the low level of input, so AR was not activated. In essence, the network generates alternating turning commands to navigate towards targets by sending left or right signals alternately in a zigzag pattern, which gradually brings it closer to the

source. The trajectory displayed in Figure 5.1, demonstrates this behaviour.

The logic of the network is specifically tailored to the AB sequence. It will respond to A by entering the "A-received" state and anticipating B. If there is an unexpected sequence with more than one B in a row, the network is configured in such a way that it does not respond to an 'additional B' state and therefore will not change its actions in response to signals following the first B. This limitation has been observed in earlier robustness tests, where the network struggled with sequences like AAB or other unseen patterns during evolution. However, during the course of evolution, the network was never subjected to more than 1 B after an A, so having no response to multiple B did not affect performance in the settings of the task it was evolved to solve.

The network's responses to incorrect stimuli (distractors) also demonstrate its mechanism and how it avoids following inappropriate signals. For the distractor pattern AA (two A signals in succession), an interneuron N0 starts sustained spiking after the first A is perceived (just as in the AB case). N0 fires, forcing AL to spike. But because there was no single B, N2 is not active, which kept AR quiescent. The end result is that, regardless of where the 'AA' pattern comes from, AL spikes for a brief period, leading to the animat turning in a single (clockwise) direction. Basically, the animat makes a minimal turning (due to the single-sided motor activation) but does not receive the intermittent left-right guidance that would draw it to a source instead it ends up spinning in place, or close to the starting position. Therefore, the animat does not forge towards the source emitting AA.

For the case of BA, the order of events is reversed, and the system does not manage to enter the A state. For example, consider a source on the left (strong input). The B signal will cause N1 to spike (and N2 to spike once). Then A comes and activates N0. The problem is that with N0 activation coming after B in this case, it does not properly overlap with N1 activity to start N2 firing. In the case where the source is on the right (weak input), B does not trigger N1, meaning that N2 stays quiet. In either case, N2 is mostly silent for 'BA'. The motor outputs are the same as in AA because N0, when activated by A we see a spike in AL and AR is not firing, so the animat spins in place instead of moving toward the source.

In the absence of any A signal in the BB pattern, N0 never turns on. Therefore, half of the network, the part needed to trigger turning, remains dormant. This B signal alone potentially triggers N1 (and N3) for a very short period of time. However, without N0 firing, N2 cannot maintain spiking. Thus, there is no activity in either motor neuron in the case of BB. The animat, therefore, remains mostly still.

To summarise, the winning strategy for all incorrect patterns is either to continually turn in one direction (circular) or remain stationary, thus not moving in the direction to any distractor. This behaviour is consistent with the success of foraging: The animat completely disregards any input that is different from the AB pattern thanks to the simple, safe circling strategy or complete rest, which prevents it from hitting any distractor.

We examined network logic with low (0.33 – right side) and high (0.67 – left side) input levels to assess its behaviour. When the target is at the edge of the sensory range, the interneurons show some activity, but not enough to trigger a turn. In the absence of high to moderate input

level, the animat displays very low motor activity and tends to turn by default to the right. This is observed at low input levels (sigmoid outputs from 0.1 to 0.4), when the target is not collected. As the input level increases, the network becomes more active and begins foraging.

It was observed that when the right sensors received stronger input than the left, the animat turned toward the right. This occurs because when the source is located on the right side, the signals are weaker on the left sensor. Since B comes from the right, N1 does not fire, thus N2 remains silent and only with activation of N0 then AL spikes, causing a right turn. For larger inputs (> 0.5) that are associated with the source being located on the left, the above mechanism functions as: N0 and N1 fire, N2 holds, and thus AR is activated and dominated AL spikes that result in a left turn. The spiking trains of interneurons at various strengths matched the expected responses of different patterns (see Figure 5.2) . What is interesting about N0 is that it is the only interneuron that does not fire at low input. However, with increasing input, N0 becomes active and remains so until B arrives, contributing to the key state memory.

With higher input values, N2 is observed to spike even when N1 is not activated by A, due to the sustained activity of N0 without a coincidence in the timing of the spikes from other neurons (Figure 5.3). For example, at very large inputs and due to high N0 activity, the B signal on its own can trigger N2 much earlier than N1 is even beginning to spike. In this case, N2 starts to fire well before N0 and N1 are fired together and tends to maintain its firing rate. When N2 is activated by B, the spiking criterion of N0 and N1, which was essential before, is not relevant because the input B alone is enough to trigger N2. AR fires at a very high rate and easily outcomes AL because N2 is strongly activating it. Note that with higher inputs, the difference in firing rate of AR relative to AL being higher makes the animat respond to the target in a fast way. The animat moves faster and directionally towards targets, especially when they are within close proximity.

The raster plots in Figure 5.3 illustrate the timing of spikes in interneurons and output neurons for the AB pattern and for the AA pattern. For the correct pattern AB, we noticed the following: within the B time frame, we noted clear coordinated activity where certain interneurons fire in the described order (N0 fires then N1, activating N2 and then N3 fires) and both motor neurons fire. In particular, AR firing is dominant over AL when the source is to the left, indicating that the network is making a left turn to the target. For the incorrect AA pattern, the rasters show a completely different result. Only one of the motor neurons fires spikes, in this case AL, and this is independent of the location of the source being left or right. This is consistent with our earlier analysis for AA, where the animat behaviour is a single-direction turn (with the AL actuator constantly firing) regardless of the distractor location, which means that the motion is circular and not directional towards the distractor, in other words, no recognition of AA.

In this regard, Figure 5.3 supports the claim that the winner network uses both actuators exclusively for the correct AB pattern. Any other pattern results in one-sided or no asymmetrical activation that prevents the animat from moving towards distractor, and either stays put or spins.

The analysis of the winner's spiking neural controller shows that there is a simple state machine for temporal pattern recognition: a self-exciting loop on neuron N0 enables the network to capture

the occurrence of signal A for some time, and another loop on N2 guarantees a sustained turning command when a B following A is detected from the same side. This preservation of internal states is what allows the animat to deal with the delay between A and B and still respond correctly. The network works as a finite-state machine with short-memory: one state after receiving A (which is waiting for B) and a final state of 'turn' after B is received. Incorrect transitions do not provide enough information to trigger the correct state changes, so the animat chooses not to track distractor patterns. The robustness of this mechanism is reflected in the extensive testing, in which the animat is capable of handling significant variations over time and in the environment while performing well, as long as the input sequence including the pattern AB (one A followed by one B) and the essential synaptic parameters are preserved.

The evolved solution highlights the ways in which recurrent spiking neural circuits maintain reliable pattern recognition and decision processes, with self-excitation being the key to closing temporal gaps and self-inhibition (in this example, limited inhibitory influence) being less significant in this task.

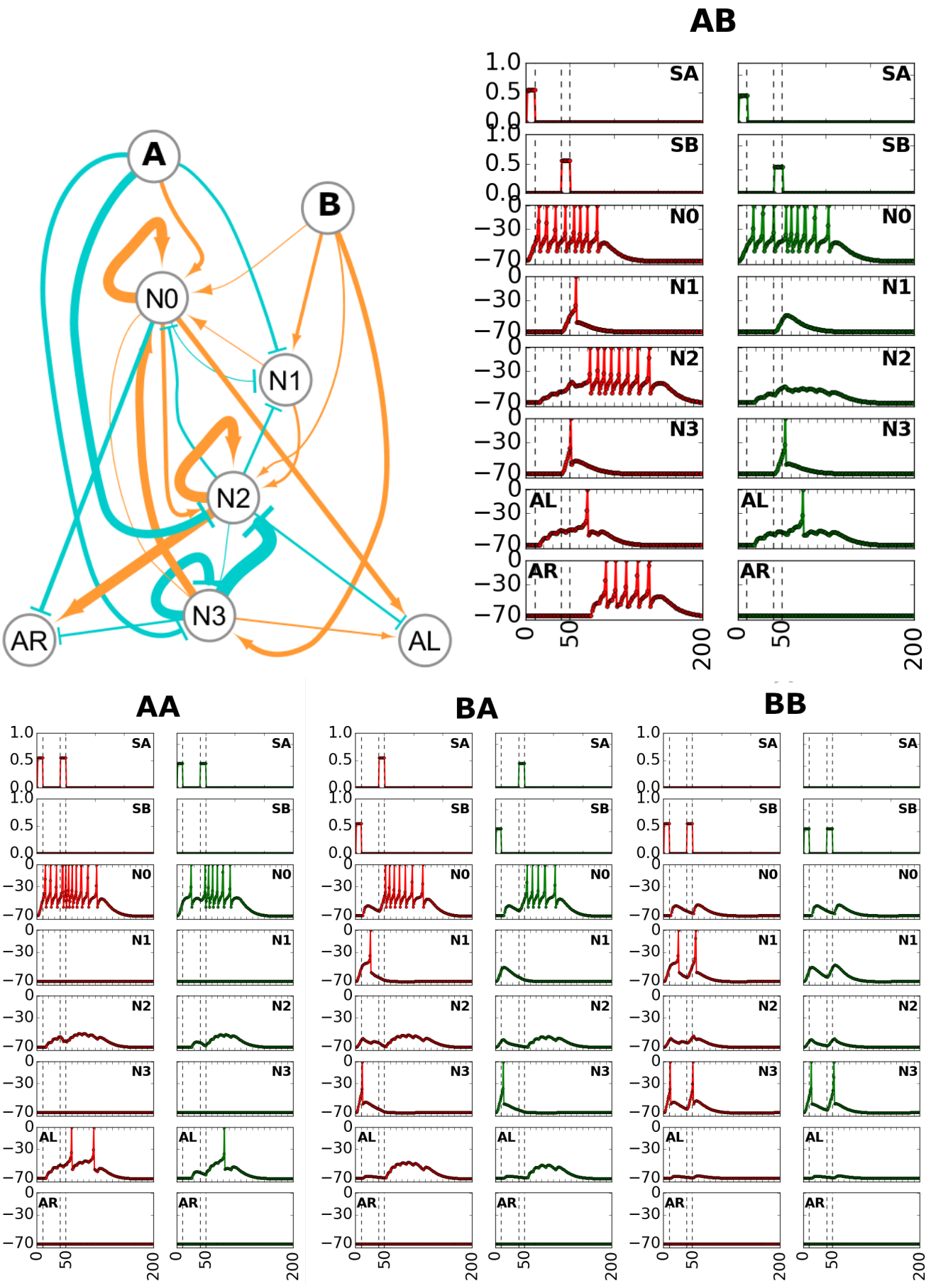


Figure 5.2: The network topology of the winner and spike trains of the four patterns for two-level sensory inputs. Excitatory links are shown with arrow heads (orange), inhibitory links are bar-headed (cyan). The voltage traces (vertical axes: voltage in mV, horizontal axes: time in ms) in green are the responses for signals coming from the right (at 0.45 sigmoid output), red traces for signals coming from the left (at 0.55 sigmoid output).

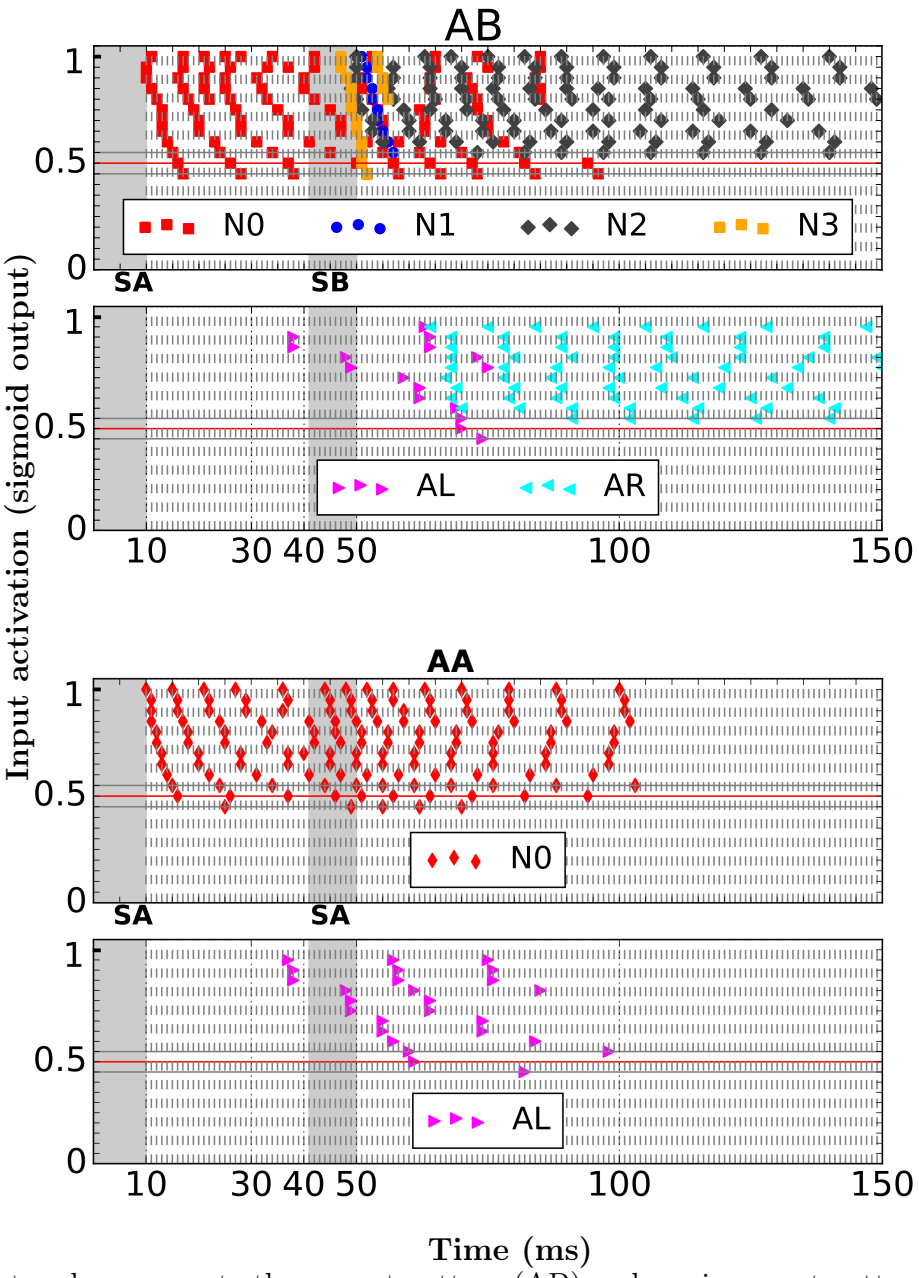


Figure 5.3: Network response to the correct pattern (AB) and one incorrect pattern (AA). The top panel shows spike raster plots for the interneurons (N0–N3), and the middle panel shows motor neuron spikes (AL, AR) for the AB pattern. The bottom two panels show the same for the AA pattern. The shaded gray regions along the time axis (labelled SA, SB) indicate the input windows for the sensory nodes. The vertical axis in the actuator panels represents the sigmoid-processed sensor input from the distance gradient sensor, with the red horizontal line at 0.5 indicating the midpoint of the sigmoid, and gray lines at 0.45 and 0.55 marking the perceptual threshold band. In the AA condition, actuator neurons show minimal spiking, and in other incorrect patterns (BA and BB, not shown), actuator spiking is absent. This indicates that inputs producing sigmoid activations below 0.45 fail to trigger motor output due to insufficient stimulation.

Chapter 6

Emergence of Topological Features in the Presence of Internal Noise

6.1 Unsupervised Learning

6.1.1 Feature Extraction

To understand the effect of network topology on performance and directional paths in the presence of noise, we first conducted an objective classification of the 50 evolved networks based exclusively on their topology (See 3.4). From each network matrix, we computed the following seven structural metrics:

- **Sum of excitatory weights** (`Sum_exc_w`): Sum of the weights of excitatory connections
- **Sum of inhibitory weights** (`Sum_inh_w`): Sum of the weights of inhibitory connections
- **Count of excitatory self-loops** (`n_exc_loops`): Sum of self-excitatory loops count
- **Count of inhibitory self-loops** (`n_inh_loops`): Sum of self-inhibitory loops count
- **Total loops** (`total_loops`): Sum of `n_exc_loops` and `n_inh_loops`.
- **Number of hidden layer edges** (`num_hidden_edges`): Number of edges connecting the interneurons among themselves.
- **Network sparseness**: The inverse of the total connectivity density.

In networks where inhibition is non-zero, the ratio of excitation to inhibition was defined as Sum_exc_w/Sum_inh_w. Each of the seven features was standardised using z-scores across the 50 networks to ensure comparability and prevent scale-related bias in the clustering algorithms. Standardisation was important here because the distributions of some features were heavy-tailed and it could otherwise introduce skewness and distort the clustering.

6.1.2 Metrics Consensus

To identify consistent topological groups within the evolved champions, we used six different unsupervised clustering algorithms in parallel, instead of assuming a fixed number of clusters. Each method captures different aspects of the structure within the data, and we aimed to achieve convergence across a variety of different criteria, not relying on a single one.

More specifically, we used: (i) the elbow method on within-cluster dispersion (WCSS); (ii) the gap statistic with bootstrap reference distributions; (iii) hierarchical clustering (Ward linkage) on z-scored features; (iv) a multi-metric internal-validation aggregate (silhouette, Calinski–Harabasz, Davies–Bouldin, plus cluster balance); (v) PCA purely for low-dimensional visualisation; and (vi) bootstrap stability quantified via the Adjusted Rand Index.

To each of the dimensions of cluster robustness, six of the approaches provided reliability, covering: compactness and separation, structure consistency without model assumption, agreement with multiple internal metrics, visual interpretability, and consistency across samples. The convergence of all these approaches provided evidence of $k = 3$.

6.1.3 Characterization of Clusters

The first two components accounted for 44.6% of the topological variance that visually separates the three clusters in lower-dimensional space, as seen in Figure 6.1(E). A partition of K-means with $k = 3$ and silhouette score = 0.293 showed that the networks for each group were distinctively represented.

Cluster 0 ($n = 14$): Inhibition-Dominant Networks

- Excitatory/Inhibitory (E/I) ratio: 0.10 (strongly inhibited, 10:1)
- Mean excitatory weight: 1.03 ± 0.52 (weak excitation)
- Mean inhibitory weight: 6.80 ± 3.21 (strong inhibition)
- Mean number of Excitatory loops: 0.47 ± 0.83 (low recurrence)
- Mean number of Hidden edges: 7.20 ± 2.15 (dense distributed interneuron connections)
- Topology: Dominated by strong inhibition.

Cluster 1 ($n = 24$): Balanced-Sparse Networks

- E/I ratio: 1.15 (approximately balanced)
- Mean excitatory weight: 2.17 ± 1.12 (moderate excitation)
- Mean inhibitory weight: 1.89 ± 1.54 (moderate inhibition)
- Mean of number of Excitatory loops: 0.70 ± 1.02 (low recurrence)
- Mean of number of Hidden edges: 3.35 ± 1.87 (sparse interneuron connections)

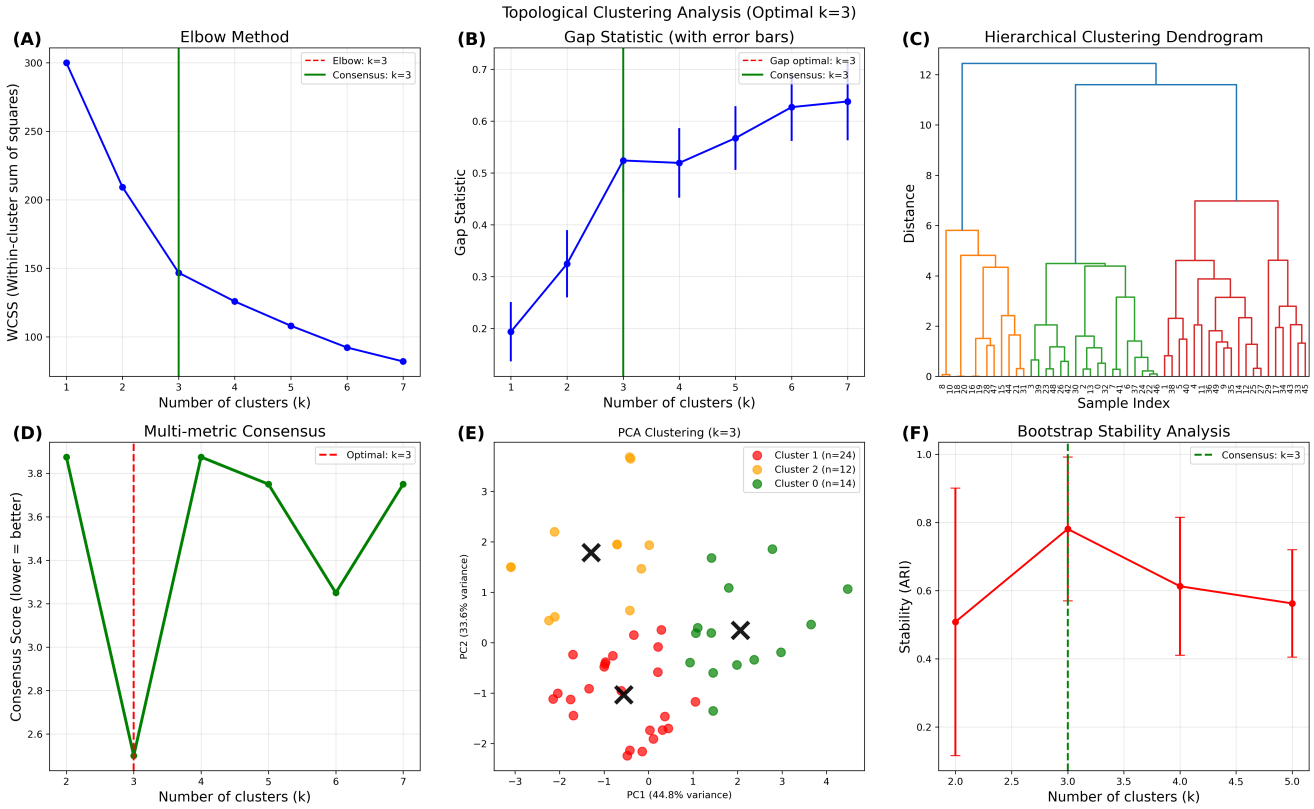


Figure 6.1: **Unsupervised structural clustering reveals three distinct topological groups (Optimal $k = 3$).** Six independent analyses were used to determine the number of structural clusters among evolved networks. (A) *Elbow Method*: The WCSS is plotted across k, with a red dashed line marking the elbow point at $k = 3$, and a solid green line marking the consensus value. (B) *Gap Statistic with Error Bars*: The gap statistic (mean log-reference dispersion minus log-observed dispersion) is plotted with 100 bootstrap replicates - the highest value at $k = 3$ shows optimal separation, with vertical green line similar to panel (A). (C) *Hierarchical Clustering Dendrogram*: Ward linkage method applied to the structural features shows a natural cut at $k = 3$. (D) *Multi-metric Consensus*: Combined scores from internal validation metrics (silhouette, Calinski–Harabasz, Davies–Bouldin, and cluster balance) are averaged for each k, dropping lowest at $k = 3$ (red dashed line), reinforcing the prior results. (E) *PCA*: The two principal components (PC1 = 44.8%, PC2 = 33.8% variance) are plotted with points coloured by assigned cluster (same colouring as in Hierarchical Clustering). Black crosses indicate centroids for each cluster in 2D space. (F) *Bootstrap Stability Analysis*: Adjusted Rand Index (ARI) is computed over 50 bootstrap resamples for each k, showing that $k = 3$ yields the most stable clustering (mean ARI = 0.793 ± 0.203). Vertical lines consistently highlight $k = 3$ as the optimal solution across all methods.

- Topology: Balanced excitation and inhibition with minimal complexity.

Cluster 2 ($n = 12$): Excitation-Dominant Networks

- E/I ratio: 3.86 (strongly excited, roughly 4:1)
- Mean excitatory weight: 8.51 ± 2.78 (high excitation)
- Mean inhibitory weight: 2.20 ± 1.45 (moderate inhibition)
- Mean of number of Excitatory loops: 2.25 ± 1.96 (highly recurrent)
- Mean of number of Hidden edges: 5.50 ± 2.31 (moderately connected interneurons)
- Topology: Dominated by self-excitatory loops and heavy excitation weights.

The inhibition dominant (C_0), balanced-sparse (C_1), and excitation-dominant (C_2) clusters are structurally united and distinguished by several attributes.

6.2 Performance Profiles of Topological Clusters

6.2.1 Experimental Setup for Testing

Unless stated otherwise, testing matched the baseline setup: open arena, one target and one distractor (once an object is collected it does not reappear), AB presented at 50% with the remaining 50% split uniformly across AA/BB/BA.

Every 1 ms, we evaluated each network on 1000 random worlds for: no noise (0 mV, noise-free condition that we refer to as 'baseline') and severe jitter (1 mV). Our analysis focused on three main metrics:

- **Target/Distractor (T/D) Performance (for definition see 3.4)**
- **Path Smoothness (for definition see 3.4)** Better scores greater than or equal to 12.5 (the mean across all 50 networks in noise-free condition) indicate more directional paths.
- **Robustness Ratios** is the retention of metric value such as $(\text{value}_{\text{noise}}/\text{value}_{\text{baseline (noise-free)}})\times 100\%$, providing a comparative index for all networks to evaluate how the metric (whether T/D or 1/ROCC) of each network is preserved under higher levels of noise.

In this chapter, "internal noise" refers to an additive random voltage drawn from a Gaussian distribution with mean 0 and variance ranging from 0.1 to 1.7 mV, applied to each neuron at every integration step (we report noise levels in mV, e.g. 1 mV).

6.2.2 Performance Under 1mV Noise

Statistical analysis with a one-way ANOVA shows that the clusters behave differently under 1 mV noise, $F(2, 47) = 3.43,\ p = 0.041$ (Fig. 6.2).

Performance (T/D at 1 mV):

- Cluster 0: 2.7 with an SD of 1.2 (8.2% return from T/D without noise 32.6)
- Cluster 1: 2.7 with an SD of 1.4 (12.8% return from T/D without noise 21.1)
- Cluster 2: 5.1 with an SD of 2.1 (25.1% return from T/D without noise 20.3)

Path Smoothness (1/ROCC at 1 mV):

- Cluster 0: 1.536 (10.1% return from 1/ROCC without noise 15.168)
- Cluster 1: 1.088 (7.9% return from 1/ROCC without noise 13.693)
- Cluster 2: 0.991 (8.7% return from 1/ROCC without noise 11.367)

Key Finding: Cluster 2, characterised by a dominant excitation topology, was the only group to exceed the good performance threshold of T/D$\geq$5 under high noise conditions (1 mV), outperforming the other clusters by a significant margin. In contrast, the inhibition-dominant Cluster 0 and the balanced-sparse Cluster 1 achieved lower ratios (T/D $\approx$ 2.7) despite having different topologies.

6.2.3 Relationships Between Topology and Performance

We performed a correlation analysis across the 50 evolved network topologies to investigate how specific structural features relate to performance under noise. This included the E/I (excitation/inhibition) ratio, the excitatory loop count, and the smoothness of the trajectory as possible predictors of task performance and robustness (Figure 6.3).

Outliers in Fig. 6.3: a few points stand out from the overall trends, particularly in panels A and B. In the case of excitation-dominant networks, those with a particularly large number of excitatory self-loops cluster as high T/D outliers and show relatively erratic trajectories. In contrast, one of the inhibition dominant networks shows a very smooth path but has a lower T/D in the presence of noise. The outliers are consistent with the trade-off captured in Fig. 6.2 and do not impact the general correlations.

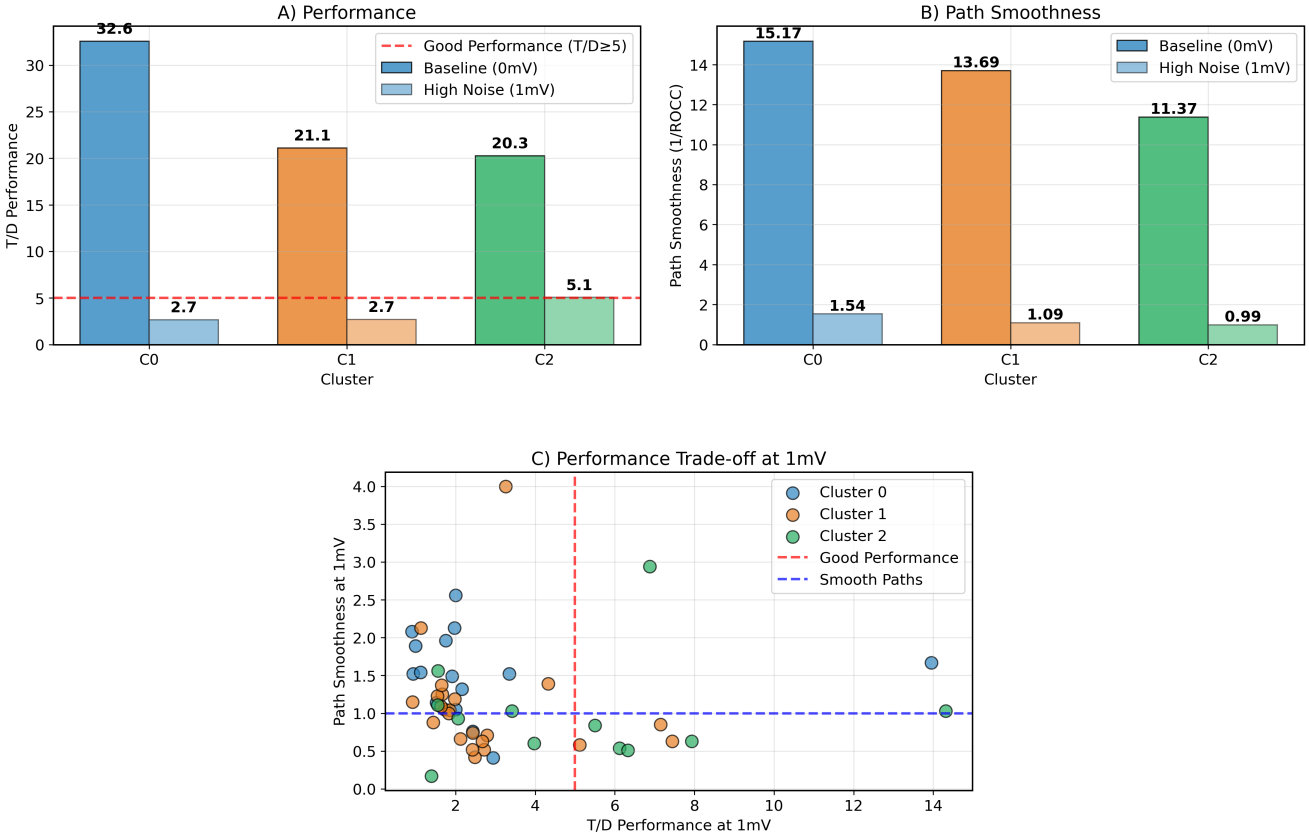


Figure 6.2: **Performance trade-off between pattern recognition and path smoothness across network clusters under baseline (0 mV) and high noise (1 mV) conditions.** (A) Performance (T/D) for clusters (C_0, C_1, and C_2) under noise-free (solid colours) and high-noise (faded colours) conditions. C_0 networks achieve the highest baseline performance but undergo the largest drop in noisy conditions; C_1 maintains moderate recognition, while C_2 - despite being lowest at baseline - retains enough accuracy to exceed the T/D ≥ 5 threshold under noise. (B) Path smoothness (1/ROCC) under baseline and high-noise conditions. All clusters exhibit large drops in smoothness under noise, with C_2 networks showing the highest absolute loss. (C) Scatterplot of T/D versus smoothness at 1 mV. Each point represents a network, coloured by cluster (C_0: blue; C_1: orange; C_2: green). Dashed lines indicate thresholds for "good" recognition (T/D ≥ 5) and "smooth" navigation. Clusters occupy distinct regions: C_0 trades high performance for smoother paths; C_2 keeps robust T/D at the cost of rougher navigation; C_1 shows a moderate trade-off between both metrics. Error bars show ± 1 SD across networks within each cluster.

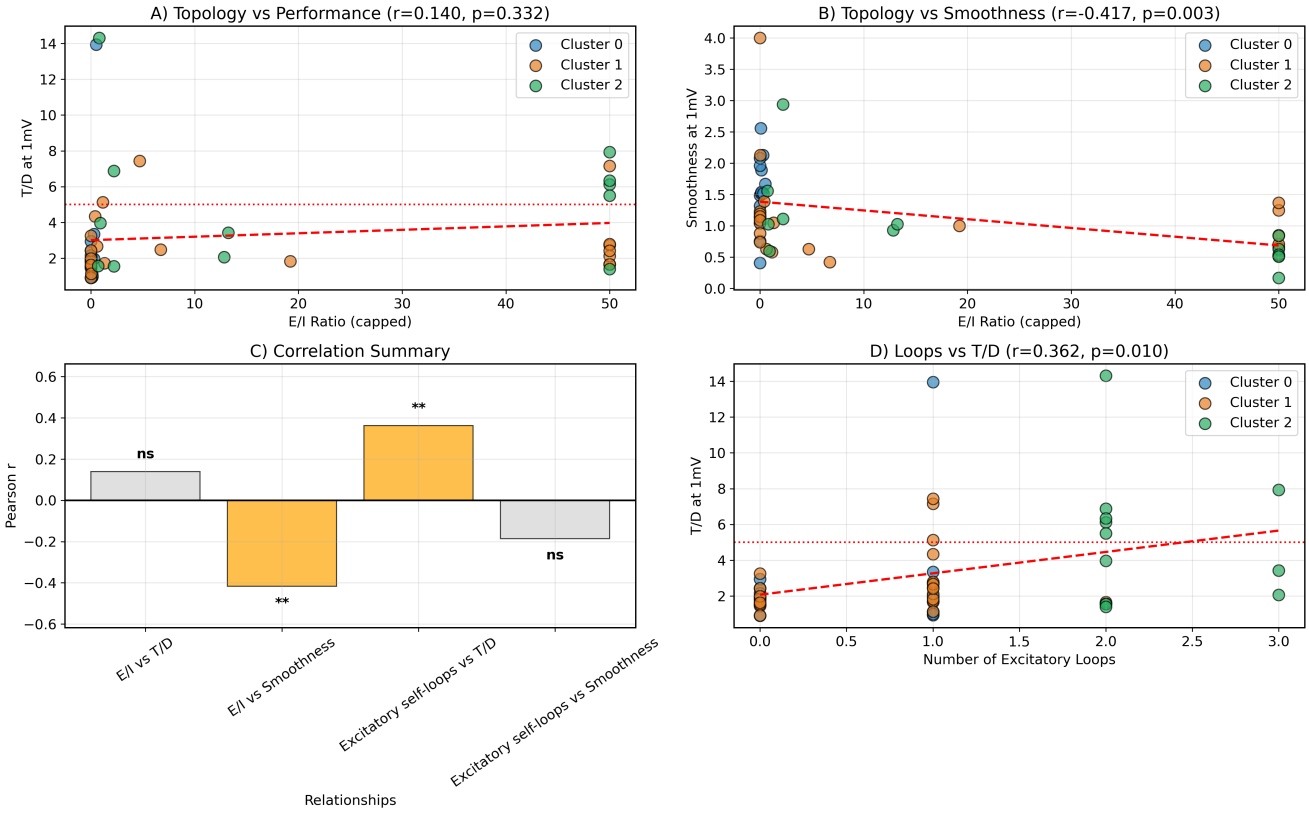


Figure 6.3: **Topological features predict differences in noise robustness across evolved networks.** Each data point represents a distinct network (n = 50), colour-coded by cluster (C_0: blue, C_1: orange, C_2: green). (A) T/D performance at 1 mV noise plotted against excitation/inhibition ratio (E/I) (capped at 50). Red line is Pearson linear fit; correlation was weak and non-significant (r = 0.140, p = 0.332). (B) Path smoothness at 1 mV noise versus E/I ratio. A significant negative correlation was observed (r = –0.417, p = 0.003), suggesting more imbalanced networks (Clusters 0 and 2) produce rougher trajectories. (C) Summary of Pearson correlation coefficients between the investigated topological features and the two performance measures (T/D performance and trajectory smoothness at 1 mV noise). The four bars correspond to: E/I ratio vs T/D, E/I ratio vs Smoothness, Excitatory self-loops vs T/D, and Excitatory self-loops vs Smoothness. *Orange bars denote statistically significant relationships (*p $<$ 0.05, *p $<$ 0.01), while grey bars indicate non-significant effects (ns, p $>$ 0.05). Excitatory self-loop count exhibited the strongest positive association with T/D robustness. (D) Number of excitatory self-loops versus T/D performance at 1 mV noise. The bold red dashed line is the Pearson linear fit, showing a strong positive relationship; networks with more self-excitatory loops maintain higher robustness under noise. Excitatory loop count showed the strongest positive correlation with T/D performance under noise.

6.3 The Performance-Smoothness Trade-off

6.3.1 Biological Perspective

Divergent performance profiles illustrate a basic trade-off between different strategies for SNNs controllers and their robustness to noise, a principle that is well established in computational neuroscience (Faisal et al. 2008).

6.3.2 Mechanistic Analysis

Excitation-dominant networks may exhibit better noise robustness because of several computational mechanisms:

- **Attractor Dynamics:** Greater excitation yields more stable states that absorb and correct noisy inputs via recurrent amplification (Hopfield 1982).
- **Signal Amplification:** Weak target signals buried within the noise are rescued by excitation loops, which overcome the noise threshold while retaining recognition through competition (Vogels and Abbott 2005).
- **Temporal Integration:** As emphasised in (Wang 2002), recurrent excitation allows a network to integrate weak or noisy inputs over time. This improves refinement in decision making and pattern recognition by combining information in several time steps.

Inhibition-dominant networks, while initially effective in suppressing low noise ($\sigma = 0.1mV$), tend to over-suppress input signals altogether in higher noise ($\sigma = 1mV$). This excessive inhibition can lead to signal attenuation, resulting in a degraded discrimination of the correct pattern and an increase in false recognition under complex, high-noise conditions.

Optimal Performance Topology: For noise robustness, excitation-dominant Cluster 2 emerged as the most successful architecture in terms of recognition accuracy, achieving the only threshold T/D = 5.1 under high noise ($\sigma = 1mV$). However, this improvement comes at the cost of motor trajectory: Cluster 2 exhibits the lowest path smoothness under noise (1/ROCC = 0.991), compared to Cluster 1 (1.088) and Cluster 0 (1.536). Despite its erratic navigation, the robustness of Cluster 2 appears to be due to its moderate excitation (E/I ratio = 3.86) and high recurrent connectivity (2.25 excitatory loops on average).

6.4 Second Evolution Setup: Minimalist Champion

6.4.1 Topological Evolution and Validation of Clustering Principles

The successful run demonstrated an interesting evolutionary plot marked by an increase in topological efficiency. Starting from networks with five interneurons at generation 100 (before that, at generation 50, the population best-of had only one interneuron, and fitness error function=0.95),

evolution repeatedly refined (pruned) network structure in a methodical way while performing better under noisy conditions.

By generation 300, the best network structure had converged to a minimal architecture with just two interneurons, both of which had strong excitatory self-loops (weight >5 units). This topology remained fixed (only the weights changed) for the rest of the evolution (generations 300–700), which implies that such minimal networks with specific recurrent connections are optimal for pattern recognition under noise.

This remarkably successful evolutionary run (T/D = 33, for $\sigma = 1mV$) reinforces the interpretation derived from the unsupervised clustering of the 50 networks. The networks that formed during the noise adaptation phase (generations 300-700) merged accurately into C_2 in our previous analysis.

- **Minimal interneuron count**: Stabilisation at 2 interneurons (matching Cluster 2 mean)
- **Strong excitatory recurrence**: Maintenance of 2 excitatory self-loops with weights > 5 units
- **Excitation dominance**: Evolution of high excitation-to-inhibition ratios (6 excitatory connections versus 2 inhibitory connections).
- **Superior noise tolerance**: noise robustness levels far exceed those observed in the original clustering analysis (T/D = 33, for $\sigma = 1mV$).

After topology stabilised (around gen 300), weights continued to adapt: self-excitatory loops and excitatory edges from the interneurons layer to the actuator nodes tended to strengthen, while inhibitory edges remained sparse and stable. This qualitative trend persisted through gen. 700.

The convergence of evolution that occurs under different experimental conditions: fixed noise training (clustering analysis) and incremental noise evolution (current experiment), reinforces the understanding that topologies driven predominantly by excitation with strong self-loop connections tend to be robust to noise. Self-loop convergence confirms the results from the initial clustering analysis where noise-resilient Cluster 2 (excitation-dominant networks) showed average characteristics of two to three excitatory self-loops and sparse inhibitory connections.

6.4.2 Change in Performance with Increasing Noise

The performance curve depicts two major evolution phases for the successful run, as shown in Figure 6.4.2.

Phase 1: generalising (generations 50–400)

Initially, the networks improved their general performance under noise conditions, as shown in Figure 6.4.2 (A). The performance ratio reached by generalists (generalising networks) in generation 400 was significant: the champion network performed under ideal conditions (T/D = 152.7) and had considerable noise tolerance (T / D = 33.0 under 1mV noise).

In fact, this performance was much better than what we obtained from the 50 previously evolved networks. For these networks, the best performing networks in Cluster 2 only obtained T/D = 5.1 under 1 mV noise. The improved noise tolerance (33.0 compared to 5.1) illustrates the extent to which performance was limited by static training conditions that let topological solutions rapidly fall into a local optimum (on average around generation 200) instead of focused evolutionary pressure.

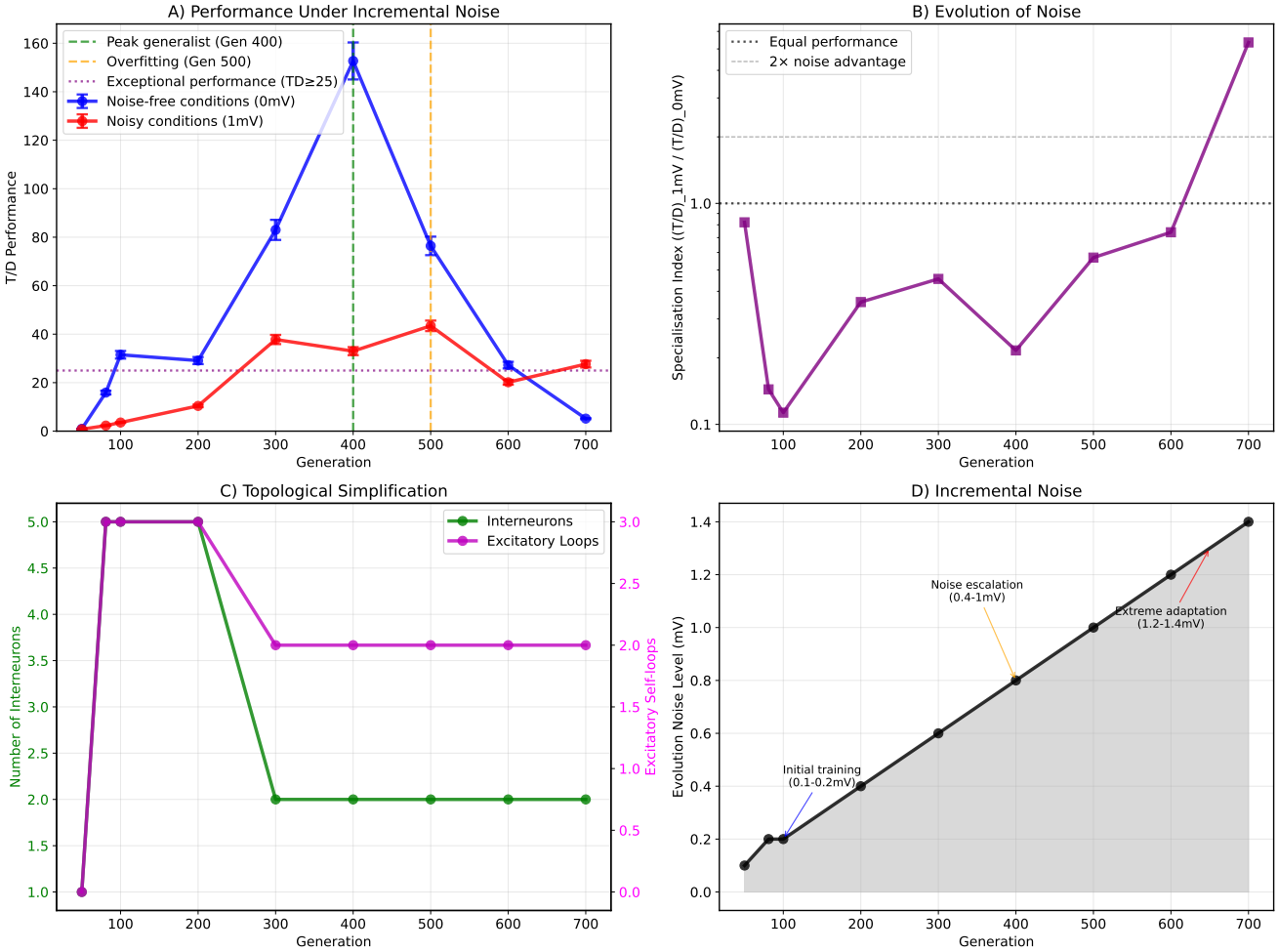


Figure 6.4: **The evolutionary curves highlight a significant adaptive change resulting from the accumulation of noise stress.** (A) There is an initial phase of generalist emergence, followed by specialist over-fitting (Gen 50-400, Gen 400-700, resp.). There is a substantial reduction after the clear peak at Gen 400. The networks respond too strongly to noise, resulting in a drop in performance in noise-free conditions after the peak. (B) The ratio of extreme specialisation ($T/D_{1\ mV}$ / $T/D_{0\ mV}$) where the final networks gained five times greater performance under 1 mV noise compared to noise-free conditions. (C) The convergence depicts topological evolution toward minimal 2-interneuron networks with self-excitatory loops. (D) The genetic algorithm consisted of starting noise parameters at 0.1 mv, increasing by 0.05 mV increments to reach 1.4 mV at generation 700.

Phase 2: Specialist Adaptation (Generations 400–700)

A strategy change was particularly observed around noise = 1 mV (see Figure 6.4.2 (D)), where the gradual increase in noise began to intensify the evolutionary specialisation. Instead of focussing on performance under noise-free conditions, networks shifted towards maximising

performance under noise to the detriment of performance under noise-free conditions. Champion generation 700 showed high specialisation noise robustness with T/D = 27.6 at a noise level of 1.4 mV, which was 14 times the training noise for the 50 networks subjected to the classification analysis. However, this specialisation was coupled by a significant trade-off.

The noise-free performance ratio was reduced to 5.17, reflecting a decrease of 96.6% due to extensive overfitting of the noise levels.

As shown in Figure 6.4.2 (B) , in generation 400 the specialisation index (that is, the ratio of performance under noise to performance in noise-free (T/D_{1mV} / T/D_{0mV}) was 0.22 and in generation 700 it reached 5.35.

This suggests that the final champion was five times worse in noise-free conditions than in noisy conditions. Such performance shifts were observed alongside topological convergences from five to two interneurons (Figure 6.4.2 (C)) that achieved a minimal configuration by generation 300.

6.4.3 Generation 500 Champion: Optimal Balance

As part of the evolutionary process, the generation 500 champion is an interesting example that demonstrates an achievement of an optimum equilibrium between generalist capabilities and specialist adaptations. The evolution of this network in 1 mV noise conditions while maintaining reasonably good noise-free performance (T / D = 76.4 at 0 mV, T/D = 43.4 at 1 mV) makes it ideal for a zoom-in analysis.

The generation 500 topology includes two interneurons with two excitatory self-loops and only two inhibitory edges. Its strongly excitation-dominated profile (Exc/Inh ratio > 10) represents a system where internal activity is sustained through recurrent excitation.

Network Topology

The champion network has a 2-interneuron topology that emerged in all successful generations with noise adaptation (Figure **??**). Each interneuron has strong self-excitatory loops with weights greater than 5 and sparse inhibitory connections. Neuron N1 is activated directly by sensory input A and feeds to both motor outputs AL and itself. Neuron N2 receives input from sensory channel B, but only inhibits AL, therefore only AR is activated when the correct pattern AB comes from the left side.

Pattern Neural Dynamics

Analysis of spike trains over time demonstrates the discrimination ability of the network for the AB pattern as opposed to all other wrong combinations. (Figure 6.6).

AB Pattern Recognition (Target Response):

As the animat's left sensory field receives the AB pattern, neuron N1 shows rhythmic firing throughout the duration of the stimulus, and neuron N2 shows a burst followed by a period of sustained activity. Specifically, AR generates unique periodic spikes which provide power to the

right wheel, causing the animat to turn left towards the AB target. The left actuator (AL) remains quiescent. When AB is coming to the right sensory field, both actuators remain quiescent, and therefore the animat stays put.

Each non-AB pattern (AA, BA, BB) triggers a different combination of neural activation. In AA both interneurons are active and stimulate AL to fire with high rate, causing the animat to rotate right, and thus away from the distractor object. In the case of BA, N2 is temporarily inhibited by B. When A is presented, both N1 and N2 are highly activated due to their heavy excitatory self-loops. No further inhibition acting on N2 allows it to suppress output AL and activate AR, resulting in the animat turning right, away from the distractor that is on the left. BB pattern results in total silence in all nodes of the network, demonstrating a correct lack of response to inappropriate stimuli.

Contextual Navigation

Systematic testing in different environments reveals that the champion has developed a context-dependent strategy based on environmental diversity to infer the spatial position.

AB Occurrence Dependency:

Testing performance at various levels of AB shows a dependency on distractor percentages that emit incorrect patterns in the environment:

- 10% AB: T/D = 243/5 = 48.6 *(moderate performance, limited target information from input sequence)*
- 25% AB: T/D = 769/11 = 69.9 *(improved target detection)*
- 50% AB: T/D = 904/10 = 90.4 *(optimal performance, balanced input)*
- 75% AB: T/D = 876/10 = 87.6 *(maintained high performance)*
- 90% AB: T/D = 606/8 = 75.75 *(declining performance, reduced diversity in input)*
- 100% AB: T = 18 *(extremely low target collection, no distractor information received)*

The most significant performance drop occurs at 100% AB, where T equals 18 versus 904 at 50%.

The greatest performance drop occurs at 100% AB, where only 18 targets were detected compared to 904 at 50%. This indicates that the animat does not depend exclusively on target patterns, but is helped by distractors to be able to move. In the absence of distractors - as in the 100% AB condition — the environment becomes uniform and lacks contextual cues that allow the animat to make right turns. Although a comprehensive analysis of the trajectory data is still needed, these observations suggest that the variety of inputs, in the form of distractors, provides the essential spatial or contextual information necessary for accurate recognition.

Multiple objects in the arena:

When multiple objects are presented in the animat world, with higher density, the dual task of navigating and recognising AB then collecting targets becomes a more challenging task than what it has seen during evolution.

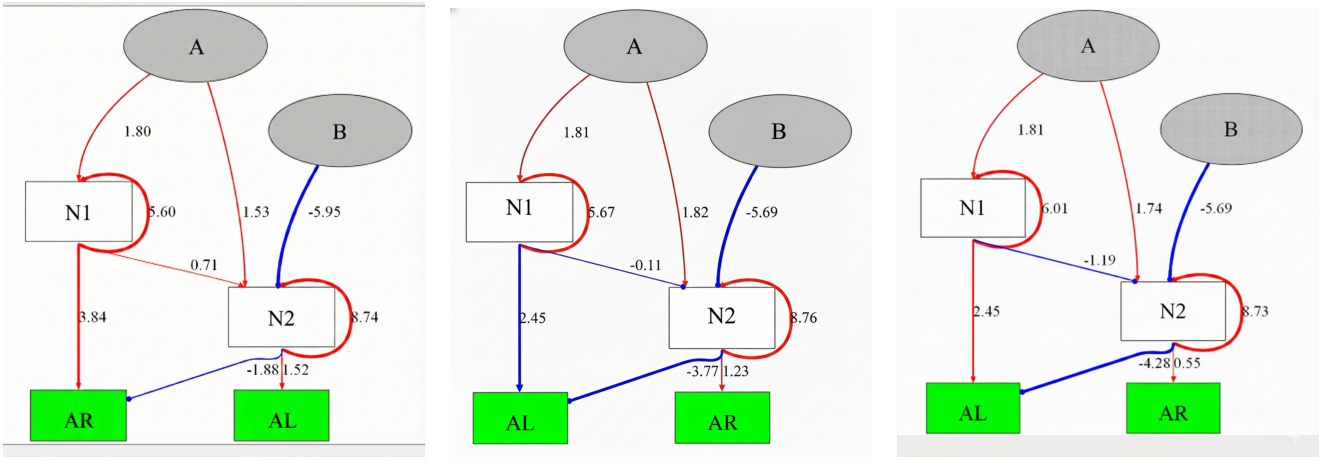
- 1 Target vs 3 Distractors: T/D = 791/8 = 98.87 (1000 targets and 3000 distractors in 1000 random worlds).
- 3 Targets vs 1 Distractor : T/D = 2004/4 = 501 (3000 targets and 1000 distractors in 1000 random worlds).
- 5 Targets vs 5 Distractors : T/D = 1854/12 = 154.5 (5000 targets and 5000 distractors in 1000 random worlds).

The remarkably high T/D ratio of 501 obtained with three active targets (we always calculate the gradient distance for the sensory pre-processing), when evaluated over 1000 random worlds (3000 targets and 1000 distractors in total), indicates that the animat maintains its performance when faced with a more challenging task.

Approach Mechanism: Activation of AR allows the animat to turn left toward the stimulus source that emits AB. No other pattern coming at the left sensory field could activate AR.

Escape Mechanism: The animat's right turn is initiated through activation of AL, which occurs only in the presence of the AA, BA or BB patterns on the left side. This helps guide the animat toward the target and prevents straying away or turning in a circle. Since the correct pattern AB never activates AL, no matter how strong the input signal is, this technique keeps the animat movement directional.

Environmental Integration: Cross-environmental or spatial coherence depends on the ability of the network to encode both targets and distractors, which requires some level of topological complexity. On the other hand, homogeneous environments, such as in the case of 100% AB trials, tend to suppress escape strategies, which often results in severe navigational stagnation (staying put in most trials), underscoring the system's evolutionary dependence on its diverse ecological environment.



(a) Minimalist champion at gen 200

(b) Minimalist champion at gen 500

(c) Minimalist champion at gen 650

Figure 6.5: Topological evolution of the minimalist champion network across generations. (a) Initial topology during sub-optimal performance phase. (b) Generalist topology achieving robust performance under both noise and noise-free conditions. (c) Specialist topology optimized for noisy conditions, with degraded performance when noise is absent.

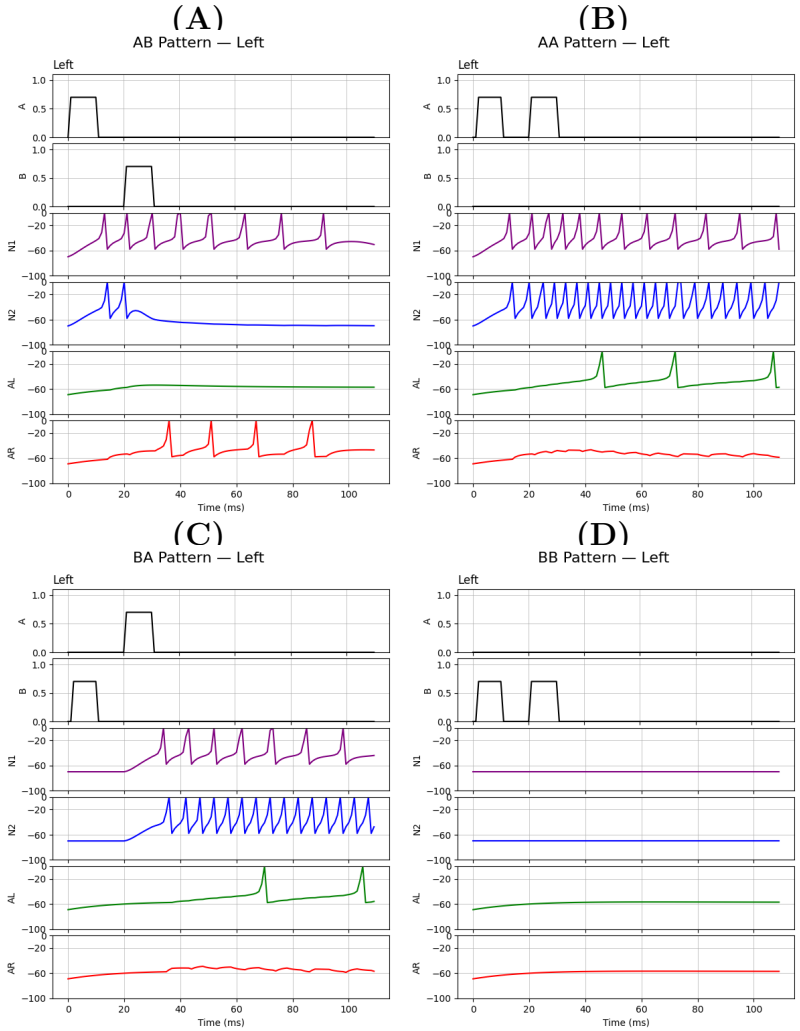


Figure 6.6: **Spike trains at level 0.7 (of the sigmoid gradient distance), for each pattern.** Spike trains obtained from the generation 500 champion's neural responses while different temporal patterns were emitted to the left sensory field. (A) AB pattern (target): N1 shows rhythmic firing, N2 is active, and AR is triggered, pushing the animat leftward. (B) AA pattern (distractor): Both interneurons exhibit high rate firing with high AL activation resulting in a rightward avoidance (there is no B to inhibit N2). (C) BA Pattern (distractor): when B is received first, it inhibits N2, but once A starts, N2 and N1 both fire aggressively due to their excitatory self-loops, and since no inhibition of N2 is coming, N2 inhibits AL and activates AR which allows the animat to turn right, away from the distractor placed on its left side. (D) BB pattern (distractor): Total neural silence suggests no movement during stimulus. Input traces show the sensory patterns ordering. N1, N2 indicate interneuron membrane potentials; AL, AR are the output neurons that control right and left navigation forces, respectively.

6.5 Discussion and Conclusion

6.5.1 Emergence of Autapses

Starting from basic spiking neural networks, Yaqoob et al. 2019 advanced to more sophisticated hierarchical systems only after achieving satisfactory results at lower levels with fixed sequence tasks such as the ABC pattern. These networks used autaptic self-excitation and feedback connections. These systems had the capability to cycle through states in a recurrent manner. Thus, the output neuron could increase when a target state was achieved even if state-dependent, activity-driven short-term memory of the neuron from past computations dampened the influence of previous computations. Building on this, later Yaqoob et al. 2023 reformulated minimal circuit behaviour as finite-state transducers that "transform" circuits into behaviours. they demonstrated how autapses help in remembering a trace of the last active state even after inputs have stopped. Due to this dual mechanism, even small networks are able to recognise and retain representations of temporal patterns for extended periods of time even after input stops.

Unlike other studies, we focus on autonomous embodied animats whose ecological parameters sculpt the evolution of spiking neural networks, rather than a designed and predetermined model. Such evolutionary approaches enable the emergence of sophisticated multidimensional pattern recognition systems essential for navigation. Compared to conventional robotics, where behaviours are often programmed in advance and designed to achieve specific tasks, our animats discover their own paths through dynamical interactions with the environment (Vasu and Izquierdo 2017).

Every embedded network must integrate preprocessed sensory inputs and make spatial decisions within a given time period, which is more challenging (both cognitively and computationally) than the ABC tasks outlined by Yaqoob et al. 2023. In this study, minimalism was not a main goal; however, we limited the number of interneurons to 8 during evolution to obtain a simple enough network to decode and read its spike trains.

Regardless of the considerable variations across these experiments, there was one common feature all the evolved neural architectures shared: independently, they developed multiple circuits containing autaptic loops. As described in Yaqoob et al. 2023, certain newly introduced neuron types with self-excitatory loops can sustain activity and information storage in the absence of external stimuli during prolonged silent intervals.

Based on the results of the cluster analysis, the networks belonging to Cluster 2, which had the highest mean number of excitatory self-loops (2.25 loops), displayed noise resilience with a performance ratio of $T/D = 5.1$ under testing. These results provide further empirical support for the hypothesis Yaqoob et al. 2023 that autapses function as integrative filter systems that rely on prolonged neural firing to silence short-term disturbances.

Some evolved networks included self-inhibitory loops that gated reactivation for a brief period after a spike. Although basic models Yaqoob et al. 2023 focused on spreading excitation through autapses, more advanced controllers included self-inhibitory loops that allowed firing suppression immediately after activation.

This level of control allowed the animat to navigate directionally (inhibition-rich networks

from cluster 0 had the highest noise-free path smoothness and performance compared to other clusters).

Finally, both the simpler sequence detectors described by Yaqoob et al. 2023 and the more complex evolved controllers analysed in this work employed autapses to solve temporal pattern recognition tasks in the presence of noise. The variety of self-autaptic loops that emerged suggests that these self-targeting connections may be fundamental for learning, adaptation, and precise temporal coordination within spiking neural networks.

Chapter 7

Cross-Platform Validation and Physical Robot Integration

7.1 Introduction

The biological realism of SNNs has earned them growing recognition for adaptive pattern recognition and decision making in time-sensitive settings. As presented in the previous chapters, the focus was towards building and simulating an SNN model which aimed to identify certain sensory patterns. However, its robustness can only be validated when it is shown to work in multiple computational systems and physical settings.

This chapter provides a two-step validation for the SNN champion (see 5.4): (i) the CPU implementation of the GReaNs simulator was manually ported to the GeNN framework on the GPU, and (ii) the GeNN implementation was embedded on a mobile robot. The GeNN framework allows for the powerful simulation of SNNs on NVIDIA GPUs by converting high-level neural models into optimised CUDA code (Yavuz et al. 2016, Stimberg et al. 2020). GeNN has been shown to outperform traditional CPU-based simulators, and even some uses of neuromorphic hardware, in some cases using real or near real-time with much less energy than is typically expected (Knight and Nowotny 2018). In this work, GeNN was used as a back-end execution system. After manual port, no GeNN was used for any further learning on the model.

During the physical validation phase, the trained model was deployed on a robot with an onboard camera facing forward (The robot belongs to the University of Sussex, during a demonstration at the ICANN 2018 conference). The visual input received by the robot was presented as full-screen coloured flashes—red for stimulus A, blue for stimulus B, and black for silence. The SNN successfully detected the AB sequence and ignored all other combinations (AA, BB, BA) to activate an appropriate motor response only with the correct pattern AB. Although this demonstration was performed under constant controlled conditions with respect to space and light, it showed that the learning of the spiking network was behaviourally passed from the simulation to the physical system.

The need for embodied validation of neuromorphic systems has grown because it serves to prove that bioinspired models are functional and resilient to noise in real world environments

(Indiveri, Bartolozzi et al. 2022). Our results parallel those of other attempts at neuromorphic robotics, such as cerebellar SNNs managing delayed feedback control (Abadía et al. 2021) or real-time stabilisation in Loihi-based drones (Stagsted et al. 2020). In addition, the use of GeNN in our validation efforts contributes to the increasing literature that GPU-accelerated SNN simulation is capable of closing the gap between practical neuroscience and embodied intelligence (Knight, Komissarov et al. 2021, Yavuz et al. 2016).

The demonstration of the physical robot together with the GeNN-based deployment constitutes compelling proof of the functional robustness and transferability of the proposed model. This chapter describes the validation process, the preparation of experiments, and the outcomes, considering them in terms of multi-platform integration as well as practical use of spiking neural controllers.

Manual Topology Transfer The evolved GReaN network (5 neurones, fixed weights) was first transcribed by hand: topology and synaptic weights were copied onto paper and subsequently encoded in 'model.cc' (See Appendix C) using GeNN. No algorithmic fitting or surrogate training was applied; therefore, the GeNN model is a literal structural copy.

7.2 Background

Although earlier chapters mention the design and function of spiking neural networks (SNNs), it is pertinent here to discuss the importance of validation across various computational and physical platforms.

As the size and complexity of spiking neural models grow, the achievement of efficient simulation becomes increasingly challenging. The resource-intensive large-scale or real-time SNNs are often out of reach for conventional simulators based on CPUs. This has led to the development of frameworks such as GeNN. GeNN translates high-level model definitions into CUDA-compiled code, producing a significant execution speed on NVIDIA GPUs (Yavuz et al. 2016). The advantages of GeNN make it particularly attractive for real-time applications, including closed-loop robotic control (Knight and Nowotny 2018).

Cross-platform validation ensures that a specific model performs functionally across multiple different simulation back-ends or hardware configurations. This justifies the existence of tools such as PyNN (Davison et al. 2009), which enables cross-engine execution of a single model to check its output correctness. As an example, in our case study, the initial model was created to function within a CPU context, but it was subsequently transitioned to GeNN and executed on GPU hardware during live demonstrations of the robotic system. This form of validation is important for not only reproducing results, but estimating accuracy in resource constrained environments as well.

Despite computation equivalence, physical validation strongly focusses on whether a neural model can be reliably executed when embedded into a real-world physical agent. This involves real-world sensors and motors which are burdened by the complications of noise and latency. This

has been noted in a number of recent works, e.g., from control of SNN-based unmanned aerial vehicles or commonly known as drones (Stagsted et al. 2020) to cerebellar feedback systems (Abadía et al. 2021). These instances capture the expanding appreciation that the capability to validate SNNs by deploying them physically is a considerable thrust toward progress.

This chapter focusses on both forms of validation, cross-platform and embodiment, and performs them using the temporal pattern detection mechanisms of the trained SNN controller from Chapter (5). The final subsections of this chapter provide more detail on how the GeNN version of the model was embedded into the robotics platform, how the experimental environment was prepared, and how behavioural data was captured.

7.2.1 Manual Porting Workflow (GReaNs → GeNN)

The AdEx network, originally evolved in GReaNs and verified through CPU simulations (5), was exported as a weight matrix alongside a list of neurons. Each weight, both excitatory and inhibitory, along with each type of neuron, was implemented in GeNN ('model.cc'). To maintain numerical stability under 1 ms time steps, continuous-time sigmoid units were chosen in place of AdEx spiking nodes, while they retained the required monotonic I-O curves for AB recognition. Thus, in both platforms, all connection weights, signs, fan-ins, and even neurons are the same; the only difference is the update equations (spike-based versus rate-based).

7.3 GeNN-Based Simulation

In order to evaluate the computation performance and the real-world value of the model, the spiking neural network was executed on an NVIDIA GPU using the GeNN framework and was presented during the live demonstration at the ICANN 2018 conference. All AdEx neurons along with the static synapses were reimplemented in GeNN as rate-based Sigmoid, TN2Linear, TBSigmoid, and CPU4Sigmoid cells with equivalent weight magnitudes.

The choice of GPU neural networks as the one to use was based on their ability to speed up CUDA simulations of neural networks by 1-2 orders of magnitude greater than through traditional CPU based methods (Knight and Nowotny 2018). Although the original model had been implemented and tested on GReaNs simulation, GeNN's CUDA parallelism provided near real-time execution capability. This performance significantly proved its ability to operate as part of a closed-loop sensory motor control system on an actual robotic platform.

For the purposes of deployment, a mobile robot equipped with a camera and a laptop for image processing was configured to process a stream of randomised sequences. Signal A was represented by a red screen, signal B with a blue screen, and the silence (absence of a signal) was represented by a black screen. The sensing layer of the network converted these visual cues into spike trains. The network was expected to accurately detect the AB pattern (red followed by blue) and respond accordingly while inhibiting responses to BB and BA sequences. Alternatively, an opposing motor reaction was required for AA patterns.

Turning left or right based on the corresponding "red-blue" signal was mapped from the output of the decision layer, which served as the final level of processing. For each trial, the screen was rotated to the left or right to best fit the view of the mobile robot's forward facing camera.

The mean red and blue channels over the field of view were downsampled to generate analogue currents which were then injected into the TN2 input neurons every millisecond. This process replicated the current-injection algorithm implemented in GReaNs.

Despite the constraints, the robot only turned towards the AB pattern when emitted from the corresponding side. This result confirmed that the GeNN-based model had a temporal recognition adaptability that functions within real-world constraints. The successful implementation of the model on GPU hardware also confirmed that simulated behaviours with GeNN were indeed reproducible and functionally realisable on a robotic platform.

The GeNN-compatible version of the model was implemented using custom C++ and CUDA code, generated through the GeNN framework. The code for this implementation is provided in Appendix C.

7.4 Experimental Setup and Results

The GeNN-accelerated model was implemented on a physical robot platform to perform an embodied real-world temporal sequence recognition validation which serves as the second layer of validation. This step verified whether the GeNN-accelerated simulation held true when exposed to noisy visual input and motor actuation.

A mobile robot with a single front camera and simple motor functions was used for the experiment. The experimental configuration included a laptop placed in the vicinity of the robot, used to showcase visual stimuli denoting discrete temporal events in full screen mode. These events were as follows:

- Red screen (A): represents the first signal of the target pattern.
- Blue screen (B): represents the second signal.
- Black screen: used as an interstimulus interval to represent silence (between patterns and between signals).

As black screens were interspersed between each stimulus displayed in a fixed sequence, every stimulus within a sequence had a designated fixed duration proportional to the time periods used during evolution. Only the AB sequence of red and then blue was meant to trigger a correct turn.

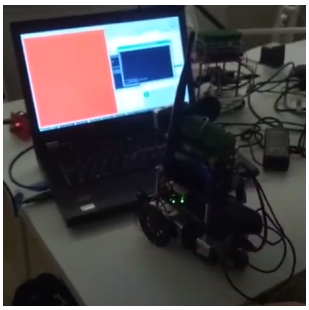
Because the robot had a limited visual field, it could not independently determine the spatial origin of the stimuli. To recreate the directional response (as the simulated directional response during simulation was left/right signal detection through lateral input), the robot was manually placed on the left or right side of the stimulus screen based on the test condition. The motor output was governed by a single neuron in the actuator layer of the network; the robot turned towards AB when a spike is detected.

To minimise variability, a constant environmental setting was sustained for all trials; lighting, the distance of the robot from the screen, and the absence of background images that could serve as visual distractions. Although spatial perception and positioning of the system were simplified to achieve manual control, the reliability and specificity of the pattern detection system were tested under controlled conditions.

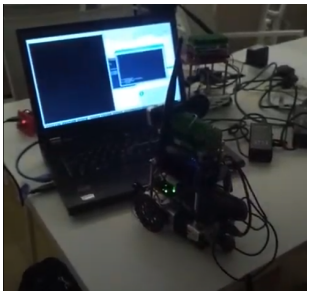
The robot showed consistent behaviour by ignoring the wrong patterns (BA, BB) and only turning right when 'AA' is presented from the left side (avoidance) and responding only to the AB pattern: turning right when AB is presented from the right side and turning left when AB is presented from the left side. This confirmed the integrity of the temporal decision making of the model when it relies on live sensory inputs and highlighted the adaptiveness driven by real-time response. This demonstrated that the model could generalise from idealised simulations to real sensorimotor interactions, achieving a significant landmark in the validation of SNNs in real life.

The Figure 7.1 illustrates that the robot responds to the correct AB pattern. The wrong ones induced silence and avoidance in the case of AA, which constitutes validation (watch the AB-pattern robot demo: `https://youtube.com/shorts/DZi8tBw-_0o?feature=share`). This captures further evidence for the hypothesis that the network was not only computably correct but functionally adaptable in physically closed-loop settings.

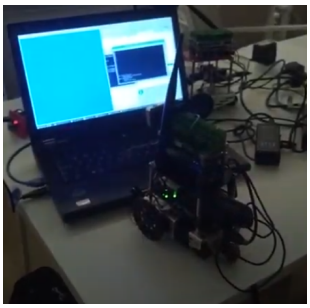
Despite the robot lacking the capability to autonomously locate lateral stimuli due to the restriction of one front-facing camera, the achievement of autonomously structuring stimulus-response cycles in designed learning platforms represented a further milestone in the transfer of behaviour. This indicates that the trained SNN was able to transfer effectively from simulation to real deployment, even with hardware and perceptual limitations. This corroborates that the hand-ported GeNN network accurately captures the decision-making rationale of its original GReaNs.



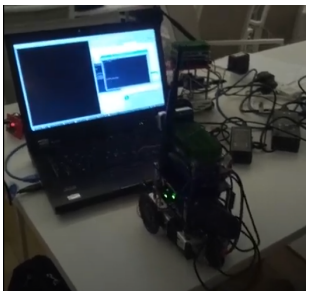
(a) Red (signal A)



(b) Silence between A and B



(c) Blue (signal B)



(d) Silence between patterns + robot response

(e) Side view of the GENN robot hardware

Figure 7.1: GENN robot response to a full 'A–B' temporal pattern. (a) Screen displays red (signal A). (b) Black screen: the 3 ms inter-signal gap. (c) Screen displays blue (signal B). (d) Final black (inter-pattern silence): the moment the network triggers a right-turn. (e) Hardware overview of the GENN robot (antenna, camera, wheels, onboard CPU).

Chapter 8

Conclusion

8.1 Research Motivation

This study aimed to investigate how *bio-inspired* spiking neural networks (SNNs), which were evolved via a genetic algorithm (GA), are able to form strong decision-making rules to solve non-trivial temporal pattern recognition tasks while foraging in a 2D space, despite environmental or internal noise.

While other researchers concentrated on basic static classification or simple navigational tasks, this work emphasises (i) explicit temporal recognition, (ii) embodied foraging, and (iii) biologically realistic neural noise.

The focus was not to "build a controller" but rather to allow evolution to find minimal architectures that a designer would not develop intuitively.

8.2 Major Findings

(1) Accidental static-noise evolution and its lesson. A significant batch consisting of 250 independent runs was carried out under the conditions of **static low-amplitude internal noise** (0.1 mV SD), which was not intended and was caused by a coding mistake.

From these runs, 50 evolved controllers were selected to perform cluster analysis. These illustrated the role of *excitatory autapses* and some specific tendencies of the network topology. However, none of the networks exhibited the simplicity and efficacy of generalisation of the later discovered minimalist champion.

This phase illustrates how static, low-noise conditions can enable evolution to converge on functional but brittle controllers, and how quickly evolutionary dynamics stagnate in the absence of ecological pressure.

(2) Incremental internal noise promotes robust solutions. The four-neurons 'minimalist champion' emerged in one of 40 independent runs conducted under **incremental internal noise**, where Gaussian noise was injected every 1 ms with σ starting at 0.1 mV and increasing by 0.05 mV every 50 generations and reaching 1 mV at generation 500, and 1.4 mV at generation 700.

This controller was the only one that solved the AB vs AA/BB/BA task and also generalised to higher noise ($\sigma = 2\,\text{mV}$) without retraining.

This rare discovery suggests a core insight: incremental noise can guide evolution toward robustness, allowing the emergence of minimal yet generalisable solutions that static evolution rarely produced.

(3) Autapses are essential for a robust controller Topology analysis uncovered the repetitive occurrence of self-excitation and self-inhibition across evolved networks. Self-excitation helped sustain activity shortly after input A, and then self-inhibition delayed the response and appears to dampen the response and stabilise the system by suppressing unnecessary firing. Together, these autaptic loops provide transient eligibility traces to some degree, allowing the network to differentiate AB from AA, BB, and BA while smoothing out membrane noise during foraging.

(4) The Transition to real-world Setting Each neuron of the evolved AdEx network model was manually copied into a GeNN rate model which could run on a GPU in real time. This enabled implementation on a physical robot.

Behavioural trials further validated that the decision rule is embedded in the topology, not the biophysics: the robot responded by turning towards AB, ignoring BA and BB, and performed an avoidance turn on AA.

Although spikes were critical for uncovering that topology, they were not necessary for its operational execution.

8.3 Limitations

- Evolution experiments incorporated a constrained mutation scheme in which gene-type changes, junk insertions, and crossover were turned off. It is likely that the discovery of new classes of topologies, including more robust or higher-performing networks, can be achieved by expanding the mutation set and allowing controlled crossover to take place. Under more complex selective pressures, an increase in population size as well as broader distributions of gene types may enable the discovery of novel motifs.

- The AB/AA/BB/BA paradigm is purposefully low-dimensional. Naturalistic foraging is multiparameter and partially observable; whether these patterns scale remains to be seen.

- The GeNN implementation used static sigmoidal units, which has verified task logic. However, this does not demonstrate that fully spiking AdEx dynamics can close the gap between reality and models without using simplifications.

- Although pattern frequency and clustering coefficients were calculated and targeted functional resilience probes were performed by node deletion and synaptic weight adjustments,

these analyses remained undiscussed and unreported. Incorporating these findings into future publications can enhance the mechanistic interpretation of evolved topologies while strengthening structure-behaviour relationships.

- The majority of successful animats developed under static, low-noise conditions, with one standout controller emerging from forty runs that applied biologically plausible noise (Gaussian, 1mV SD at 1ms step) incrementally. This individual later generalised to 2mV noise without retraining, a remarkable occurrence. This result raises the possibility of what we could name: 'minimalist champion bias', where one configuration finds a highly generalisable solution. Further research could expand these searches and extend the evolutionary time to investigate the reliability and conditions under which these adaptive controllers can be evolved.

8.4 Future Work

1. **GA hyperparameter sweep.**

   Focus on population size, mutation variance, and crossover rate and assess the emergence of noise-robust controllers to determine the emergence of consistent topological structures.

2. **Spike model deployment.**

   Evaluate unmodified porting of AdEx networks to neuromorphic hardware such as Loihi-2 and SpiNNaker-2 with respect to energy, latency and their robustness metric compared to the rate based model.

3. **Lesion and ablation studies.**

   Identify essential motif elements by removing feedback loops or individual synapses in silico to classify as necessary or redundant.

4. **Richer temporal patterns.**

   Progress from two-symbol sequences to hierarchical patterns like A(BB)A or extend to multisensory streams to evolve more complex timing hierarchies, as well as overlapping streams.

5. **Energetic cost as a fitness term.**

   Encourage plausible biological frameworks by setting limits on high firing rates and large synaptic fan-ins during evolutionary processes.

6. **Sensory pre-processing methods.** Explore alternative methods to manual preprocessing of sensory input (sigmoid + digitalisation pipeline), and instead use input encodings that mirror biological systems, for example: Time-to-First-Spike (used in auditory), or the exponential decay (close object = high current = high firing rate).

8.5 Final Remarks

This work illustrates how *evolutionary search* optimisation can identify exceptionally small circuits capable of solving complex temporal tasks, even in the presence of biologically plausible noise.

After emergence, those circuits are easily modifiable for application without behavioural degradation.

Finally, this work supports a two-part paradigm towards embodied intelligence:

(*i*) evolution should be guided by spikes and noise sculpting the controller;

(*ii*) execute the robot with the smallest and most accurate replica.

How much this paradigm can scale towards naturalistic, lifelong cognition is to be determined by future work that expands the genetic search space and advances the controllers into fully spiking neuromorphic hardware.

Appendix A

Reproducibility Tables

A.1 Experiment 1 (IEEE)

Table A.1: GA (IEEE experiment).

Parameter	**Value**
Independent runs	50
Population size	300 genomes
Max generations	2500 (early stop if fitness < 0.1 for 100 consecutive generations)
Evaluation per gen	13 random worlds per animat
Target / distractors	Targets (2) emit AB 30% of the time; distractors (2) emit AA, BB, or BA (one type at a time)
Signal structure	A and B are 8 ms each; 3 ms A–B gap; 100 ms silence between patterns
Fitness	Penalty for early distractor collection; circling penalty 0.2
Champion selection	Rank by T/D over 1000 random worlds (1 T, 1 D); top 10 analysed
Typical champion architecture	3 interneurons (top 10)

Table A.2: Evaluation and environment (IEEE experiment).

Parameter	**Value**
Test length	1000 random worlds (also 100 patterns $\approx$ 11 900 ms demo)
World	Toroidal arena (also open arena); default density 1 T + 1 D
Sensors	Distance-gradient $\rightarrow$ sigmoid; two levels (0.33, 0.67) used for analysis
Actuators	Baseline force = 2.0
Primary metrics	T/D ratio; also T/N and D/N where N are neutral objects

Table A.3: Robustness test matrix (IEEE experiment).

Test Parameter	**Observations**
Target frequency	0–100% AB; reliable once AB 20%
Distractor type	AA, BB easier; BA most confusable
Object number/size	Up to 6T+6D with proportional arena scaling
Actuator force	Optimal T/D near 2–3× baseline; >3 increases distractor hits
Temporal A–B gap	Robust 0–6 ms; degrades 7–9 ms; fails $>$ 10 ms
Inter-pattern gap	Robust up to 300 ms
Signal energy/noise	1 ms blanks within 8 ms signal collapse performance
Sensor gains	Moderate reductions tolerable; too low $\Rightarrow$ no movement

A.2 Experiment 2 (ICANN)

Table A.4: GA (ICANN experiment).

Parameter	**Value**
Independent runs	200
Success criterion	Fitness < -0.25; 20 runs met this
Controller model	AdEx SNN; same animat/body/environment and GA as IEEE
Within-pattern silence	Poisson gap, mean 30 ms
Winner architecture	4 interneurons (smooth non-circling trajectory)

Table A.5: Signal variability and winner performance (ICANN).

Test Parameter	**Observations**
AB frequency test	1% AB: 700/1000 (T/D); 10% AB: >960/1000 (T/D); ≥10% ⇒ near-perfect
A–B gap (test)	17–43 ms: near perfect; 45 ms: failure; 1 ms: 346/1000 targets
Between-pattern silence	Up to 1000 ms: T/D > 50; <40 ms: fewer targets, zero distractors
Signal duration	Train 10 ms; 5 ms fails/circles; 13 ms ⇒ T/D ≈ 7.79

Table A.6: Environment, actuator strength, and parameter perturbations (ICANN winner).

Test Parameter	**Observations**
Object density	6T+6D baseline; 10T+10D: T/D ∼12–18
Extreme imbalance	1T+20D: 999/1000 targets; 170/20 000 distractors (T/D ≈ 5.88)
Actuator strength	×2: T = 1000, D = 44; ×3: T = 889, D = 85
Inhibitory synapses	Halved: T = 1000, D = 77; ×6: T = 1000, D = 101
Excitatory synapses	Small changes ⇒ T/D → 1 (failure)
Membrane reset V_r	±3 mV: T/D ≈ 2.65

A.3 Experiment 3 (Chapter 6)

Table A.7: Topology features and clustering analysis.

Parameter	**Value**
Features (7)	Sum_exc_w, Sum_inh_w, n_exc_loops, n_inh_loops, total_loops, num_hidden_edges, sparseness
Preprocessing	z-score normalisation across 50 networks; compute E/I ratio where applicable
Cluster selection	Multiple criteria (elbow, gap, Ward dendrogram, consensus, PCA, bootstrap ARI)
Consensus k	k = 3 (stable); PCA: PC1 ∼44%, PC2 ∼34%; mean ARI ∼ 0.78

Table A.8: Cluster characterisation (n=50).

Cluster	**n**	**E/I ratio**	**Mean exc. w**	**Mean inh. w**
Cluster 0: Inhibition-dominant (C_0)	14–15	low	low	high
Cluster 1: Balanced-sparse (C_1)	23	moderate	moderate	moderate
Cluster 2: Excitation-dominant (C_2)	12	high	high	low

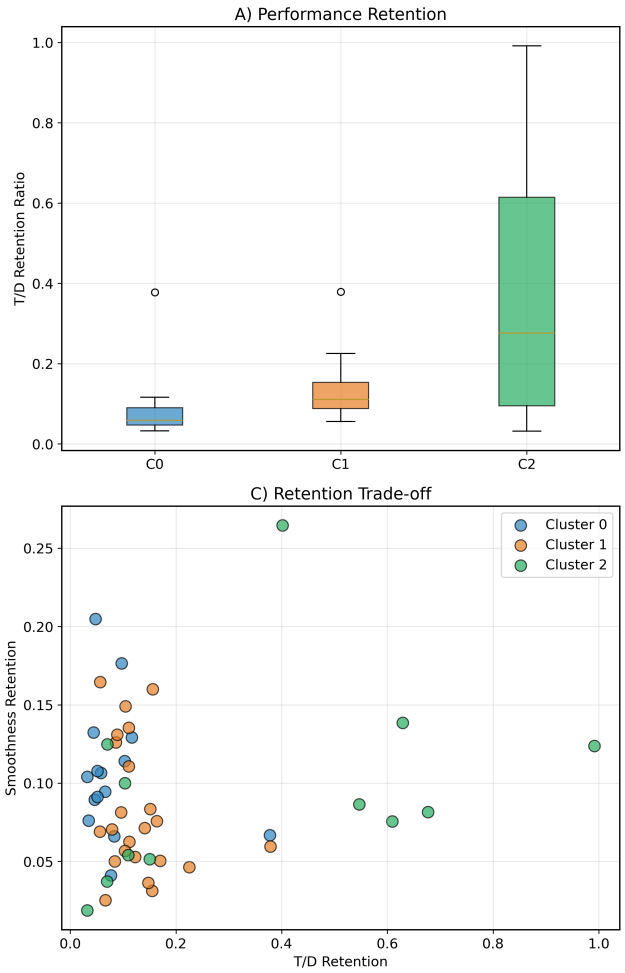
Table A.9: Noise robustness protocol and test metrics.

Parameter	**Value**
Baseline noise σ (training)	1.0 mV (Gaussian, 1 ms step)
Test noise σ	0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0 mV
Performance retention (T/D)	$\frac{\text{T/D at noise } \sigma}{\text{T/D at } \sigma=0}$
Smoothness retention (ROCC)	Rate-of-change-of-curvature; lower ⇒ smoother
Retention smoothness	$\frac{\text{ROCC at } \sigma}{\text{ROCC at } \sigma=0}$

Appendix B

50 Networks Analysis

B.1 Noise Robustness Summary Across Clusters



D) Cluster Summary Statistics

Cluster	N	TD Retention	Smooth Retention	Exc/Inh
C0	15	0.086	0.107	0.10
C1	23	0.129	0.083	828265.01
C2	12	0.366	0.096	4472502.74

Figure B.1: Noise robustness by topological cluster across the 50 evolved networks. (A) T/D retention. (B) Smoothness retention (1/ROCC). (C) Trade-off scatter by cluster. (D) Cluster summary (counts and cluster-median retention values).

B.2 Representative Networks by Cluster

The spike train responses as well as the network topology of a spiking neural network (SNN) from Cluster 2 are marked by prominent excitatory self-loops. This SNN controls an animat foraging within a two-dimensional environment where it should move toward targets that emit the AB input pattern while avoiding distractors that emit AA, BA, or BB patterns. Patterns are input from the left or right, but these do not overlap temporally.

- **(1) Input encoding:** Inputs A and B (black traces) activate sensory neurons N1–N4 depending on the side of emission. Excitatory self-loops connectivity trigger spiking in N1–N4, particularly in response to correct pattern AB: A followed by B.
- **(2) AB discrimination:** When AB is presented, the emission of B after A activates spiking in N3 and N4 due to temporal coincidence by prior A activity. In contrast, AA, BA remain quiescent, and only BB produces a forward move (both actuators generate one spike each), when input level is high.
- **(3) Motor activation:** For AB from the left, activation flows through N4 to AR, pushing for a right turn. For AB from the right, N3 activates AL, triggering a left turn.
- **(1) Sensory encoding:** A and B activate N1 and N2. Neuron N1 receives both direct excitation and recurrent feedback from itself and N2.
- **(2) AB response:** Only in the AB case does the timing of A followed by B result in strong spiking in N1, which in turn drives N2. This occurs because A triggers N1, and B subsequently excites both N1 and N2. In contrast, AA or BB patterns fail to produce this cascade.
- **(3) Directional activation:** When AB is presented from the left, the B-driven N2 activity propagates to AR, triggering a right turn. When AB is presented from the right, the activity drives AL, pushing for a left turn.
- **(4) Distractor suppression:** For AA and BB, repeated input causes either no response for lower input values or triggering a spike in AL and another in AR, pushing the animat forwards instead of towards BB source. For BA, the reversed stimulus order fails to establish the A $\rightarrow$ B chain through N1, and motor neurons remain quiescent.
- **AB pattern (from left):** Input A moderately activates N1, which primes the network. The following B input immediately activates N2. N2 excites N3, which projects to AR and pushes the animat to turn left. Self-inhibition in N1 and N2 ensures that AL is not triggered.
- **AB pattern (from right):** A input activates N3 and N2, followed by B input that triggers N2. The resulting activation propagates to N1 and activates AL. AR spikes once, but since AL spikes twice, hence animat turned right.

- **AA pattern:** Repeated A inputs inhibit N1 at high input values and no motor output is engaged. When the input level is lower (0.4), after AA signal, the inhibitory effect on N1 decreases and N1 gets triggered, which allowed AL to spike twice, so the animat turned right.
- **BA pattern:** B input prematurely activates N2 and N1 before the network is primed by A. The subsequent A input is not enough for N2 to remain in sustained activity hence only AL spikes once regardless of the input level (left/right).
- **BB pattern:** Sequential B inputs activate N2 which activates both N1 and AR, even with N3 being dormant, AL spikes once. Therefore the animat moves forward, and not towards the BB source.
- **Functional role:** Only AB sequences with correct temporal order trigger the N2 $\rightarrow$ N1 $\rightarrow$ AR cascade, resulting in directional motor activation.

B.2.1 Cluster 2: Excitatory Dominant

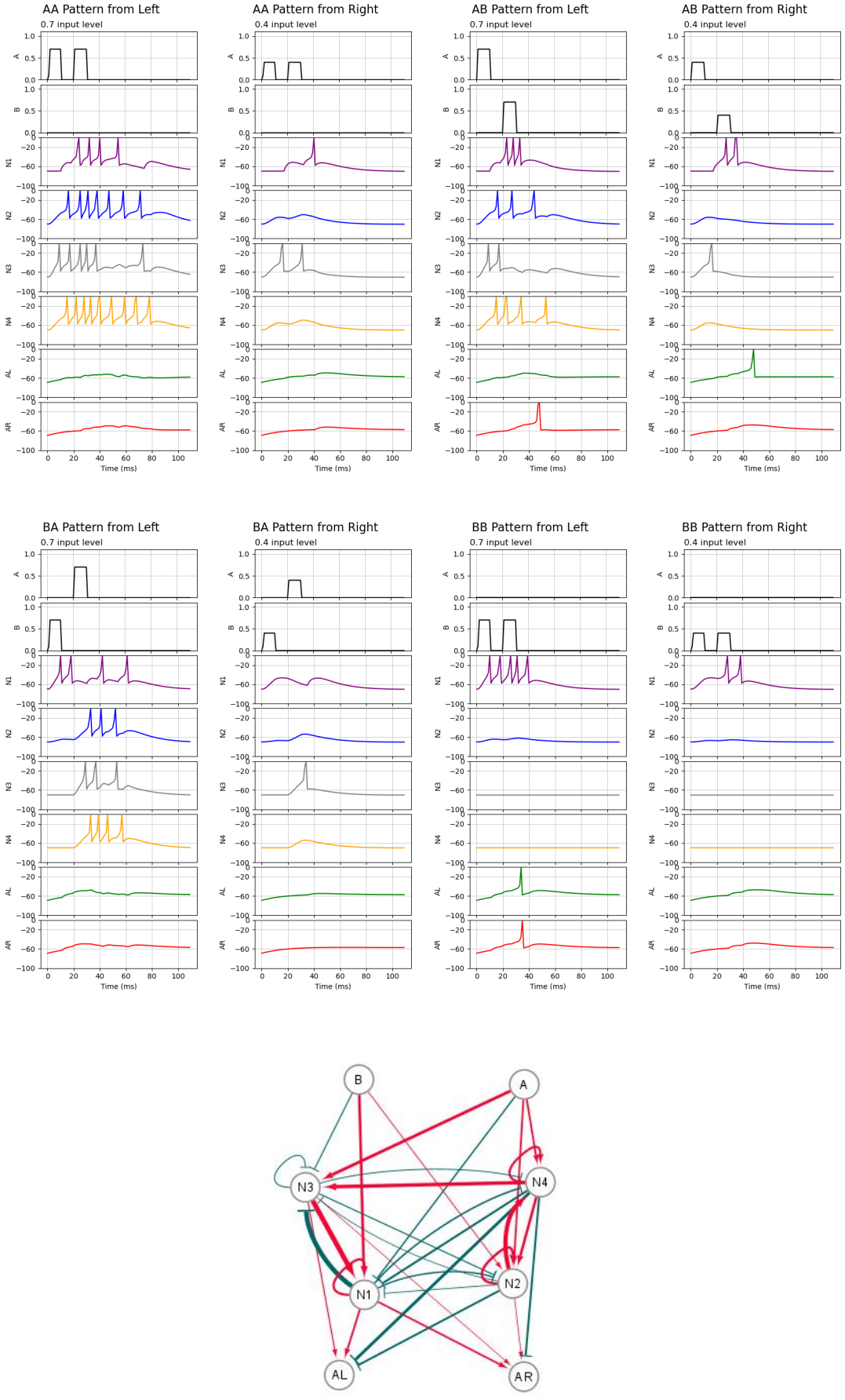


Figure B.2: Spike train responses to AA, AB, BA, and BB input patterns presented from the left and right sides, along with the corresponding network topology for a representative SNN from Cluster 2 (excitatory dominant). Each panel shows input spike trains, output neuron responses, and network structure.

B.2.2 Cluster 1: Balanced-Sparse

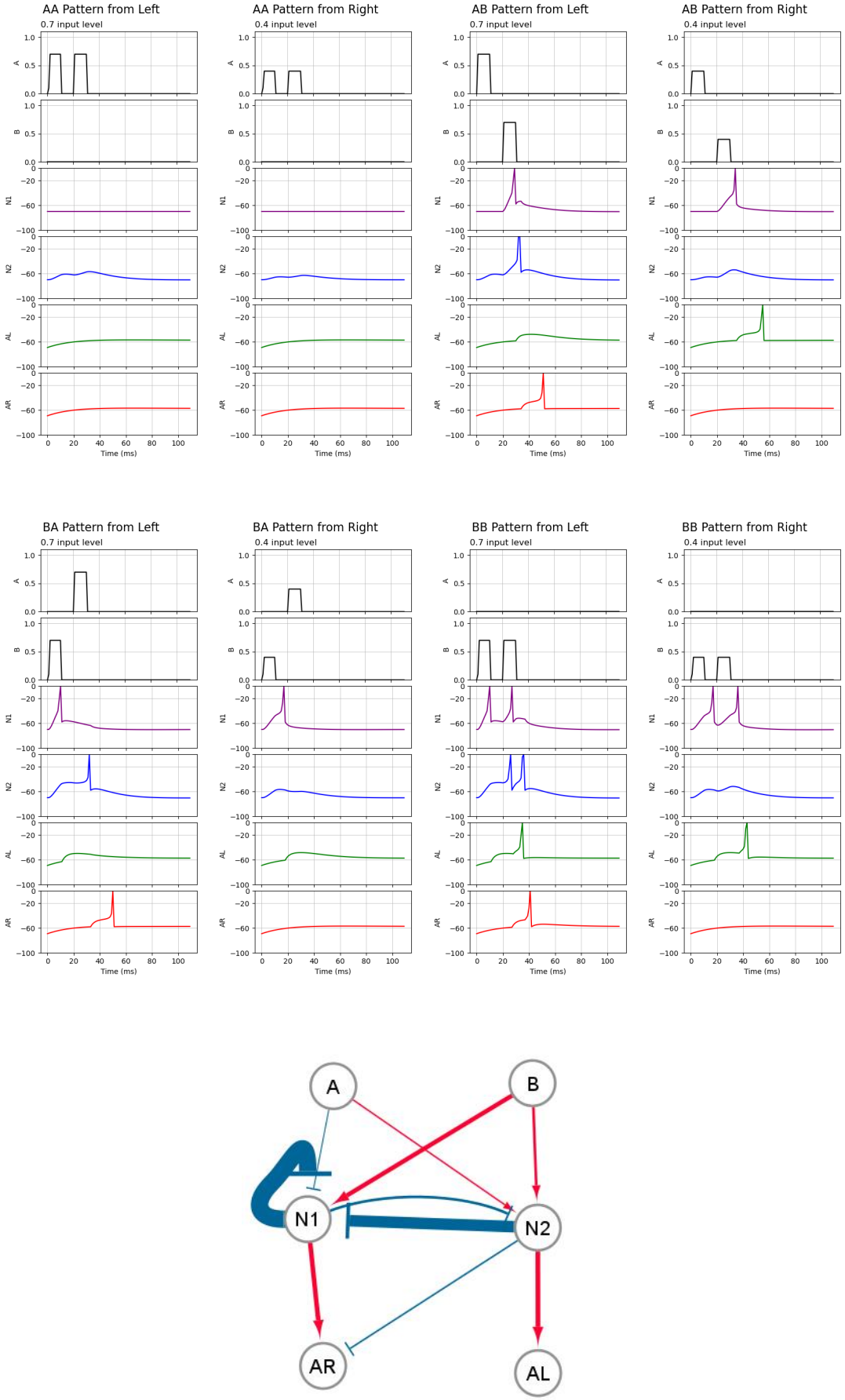


Figure B.3: Spike train responses to AA, AB, BA, and BB input patterns presented from the left and right sides, along with the corresponding network topology for a representative SNN from Cluster 1 (balanced-sparse). Each panel shows input spike trains, output neuron responses, and network structure.

B.2.3 Cluster 0: Self-Inhibitory Dominant

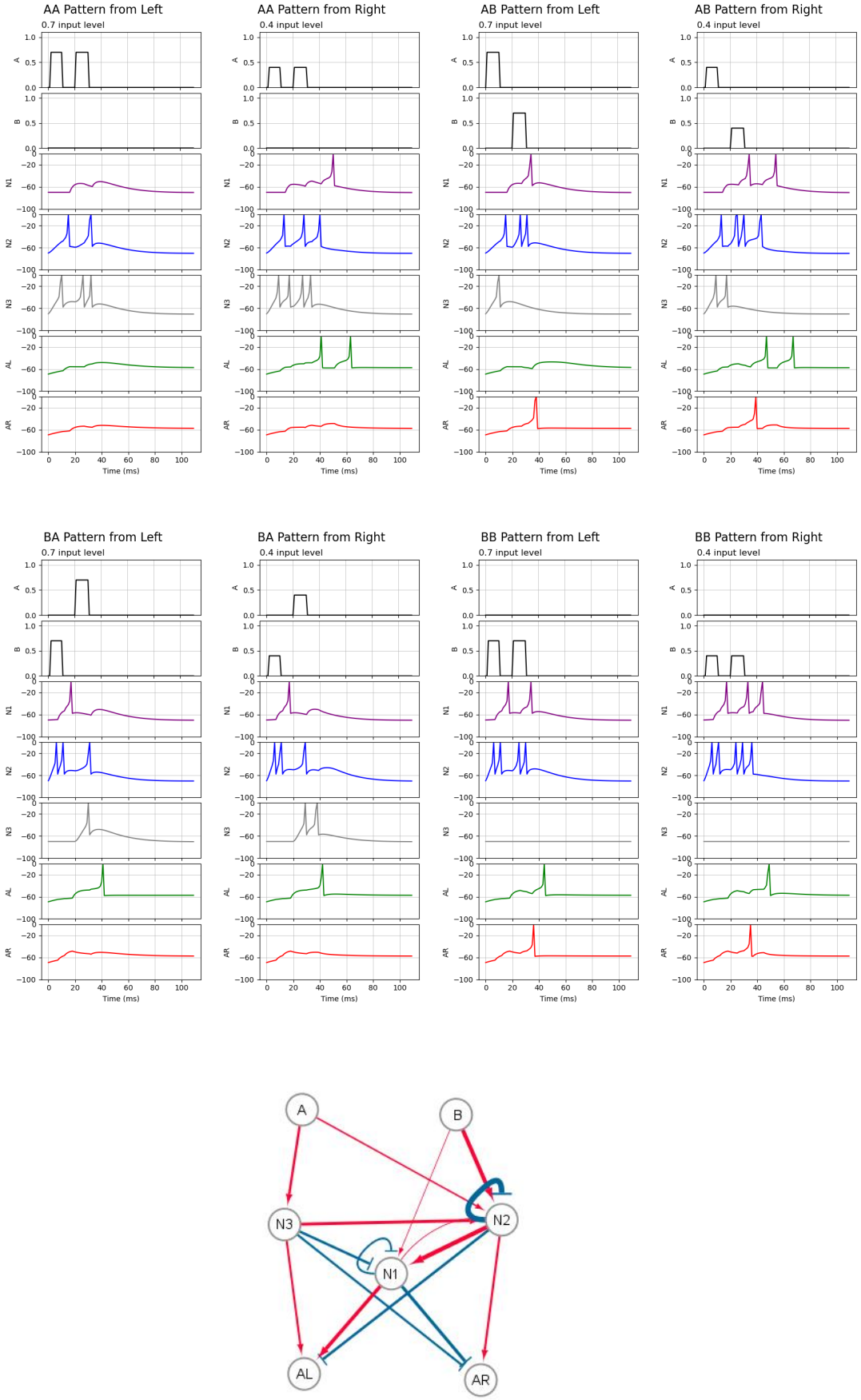


Figure B.4: Spike train responses to AA, AB, BA, and BB input patterns presented from the left and right sides, along with the corresponding network topology for a representative SNN from Cluster 0 (self-inhibitory dominant). Each panel shows input spike trains, output neuron responses, and network structure.

Appendix C

GeNN and Robot Implementation

C.1 From GReaNs to GeNN

Table C.1: Equivalence matrix between CPU/GReaNs and GPU/GeNN models.

Aspect	**CPU/GReaNs**	**GPU/GeNN**
Network topology	5 neurons; fixed connectivity/weights	Identical topology/weights (manual copy to model.cc)
Neuron model	AdEx spiking	Rate units (Sigmoid/TN2Linear/TBSigmoid)
Synapses	Static; E/I signs preserved	Static; signs/magnitudes preserved
Time step	1ms	1ms
Learning	None	None
Purpose	Cross-platform validation	GPU execution for live demo

C.2 Execution Environment

Table C.2: Build environment for GeNN run.

Component	**Specification**
Host OS	Ubuntu 18.04 LTS
GPU	NVIDIA GeForce GTX 1080; VRAM 8GB
CUDA toolkit / driver	10.0 / 410.xx
GeNN	3.2.x (code generated from model.cc)
Compiler	gcc 7.5.0
Build flags	-O3 -DNDEBUG -march=native -std=c++11
Artifacts	model.cc, generated CUDA kernels, executable

C.3 Validation

Table C.3: Stimuli during cross-platform validation.

Sequence	**Presentation coding**	**Expected decision**
AB (target)	Red (A), 3 ms black, Blue (B)	Turn toward stimulus side
AA	Red, black, Red	Avoidance
BA	Blue, black, Red	Suppress response
BB	Blue, black, Blue	Suppress response

C.4 Robot Platform, Mapping, and Behavioural Protocol

Table C.4: Robot platform and connections.

Component	**Specification**
Robot base	Two-wheel differential-drive ground platform (custom BoB-style chassis)
Sensors	Forward-facing USB camera (HD, fixed-focus)
Actuators	Left/right DC gear motors via motor driver
Onboard compute	Host laptop (Intel i7-class CPU, 16GB RAM) running GeNN
Display for stimuli	External 24" 1080p monitor facing robot
I/O mapping	Decision unit → turn command (left/right/none)
Notes	Robot positioned left or right of the screen per trial; start at fixed mark

Table C.5: Stimulus timing and camera→current mapping.

Parameter	**Specification**
Colour coding	Red=A, Blue=B, Black=silence
Inter-signal gap	3ms black between A and B (AB)
Inter-pattern gap	Black screen; fixed duration as in validation
Frame processing	Mean R/B channels → downsampled analogue currents
Injection	Currents to TN2 input neurons at 1 ms step
Parity	Same current-injection algorithm as in simulation

Replication: Randomise AB/AA/BA/BB blocks; place robot left and right of the screen; launch the GeNN executable; verify 1 kHz injection; log decision unit and motor commands; repeat N trials per condition; summarise using the table above.

Table C.6: Trial matrix and observed behaviours.

Stimulus sequence	**Side of presentation**	**Observed behaviour**
AB	Left/Right	Turn towards stimulus side (selective)
AA	Left/Right	Avoidance (turn opposite)
BA, BB	Left/Right	Suppression (no turn)

Appendix D

Versions and Bill of Materials

D.1 Software/Hardware Versions

Table D.1: Software and hardware versions.

Component	**Version/Specification**
OS	Ubuntu 18.04.6 LTS (dev/build); cluster run on Rocky Linux 9 nodes (PBS/Torque)
CUDA / Driver	10.0 / 410.xx
Compilers	gcc 7.5.0; g++ 7.5.0
MPI & scheduler	OpenMPI 4.0.5; PBS/Torque (core32 queue)
Packages	Python 3.7; numpy, pandas 1.19; matplotlib 3.1 (for plotting/analysis)
GeNN / GReaNs	GeNN 3.2.x; GReaNs (HPC quickstart) at repo below, branch main.

D.2 Robot Bill of Materials

The canonical list we used is maintained here: `https://github.com/BrainsOnBoard/GeNN_Robotics/blob/master/robot-shopping-list.md`.

Table D.2: Robot BOM.

Category	Item / Model / Notes	Qty
Chassis & mechanics	2-wheel base, wheels & tyres, caster, mounting plates, fasteners	1 set
Motors & drivers	12V DC gear motors (L/R); motor driver (e.g. TB6612FNG/L298N class)	2 + 1
Power	3S Li-Po (11.1 V) 2200–5000 mAh; XT60 leads; inline fuse; charger	1 set
Compute	Laptop with NVIDIA GPU (GTX-class); 16GB RAM minimum	1
Perception	USB camera (e.g. 1080p Logitech-class) + mount/bracket	1
I/O & comms	Powered USB hub; USB-Serial (FTDI); assorted USB cables	assorted
Display	24" 1080p LCD/LED monitor for stimuli	1
Misc.	DC-DC buck (12V→5 V); wiring loom; 3D-printed adapters; spares	assorted

D.3 Data Availability

- **Scope.** The shared artefacts are the animats themselves (controllers), not analysis logs or derived metrics. Specifically:
  - The set of 50 evolved animats used in Chapter 6.
  - The exceptional run animat from Experiment 2 (Chapter 6).
  - The champion animat discussed in Chapter 4 (IEEE).
- **Location.** Public GitHub repository: `https://github.com/bensmailchama-boop/greans-hpc`
  Folder: `Examples/TPR_animats`.
- **How to view/use.** These animats are to be opened in the GReaNs Windows GUI. Successful compilation has been verified only with Microsoft Visual Studio 2012 (VS12). After building locally on Windows (VS12), launch the GUI and load the animats from `Examples/TPR_animats`.
- **What is not included.** Full analysis outputs (logs, figures, aggregate tables) are not part of this bundle; only the animat controllers needed for inspection/replay in the GUI are provided.

D.4 Code Availability

GReaNs — HPC quickstart

Repository (branch main): `https://github.com/bensmailchama-boop/greans-hpc`
Commit: `8c92539f195f5e61ecfabaa4ddce7f074163570a`

Job script: `cluster/pbs_gaworker.sh`
Env: `scripts/env.sh`

Bibliography

Abadía, I. et al. (2021). 'A cerebellar-based solution to the nondeterministic time delay problem in robotic control'. *Science Robotics* 6. DOI: 10.1126/scirobotics.abf2756. URL: https://doi.org/10.1126/scirobotics.abf2756.

Abbott, L. F. and S. B. Nelson (2000). 'Synaptic plasticity: taming the beast'. *Nature Neuroscience* 3, pp. 1178–1183.

Abdelmotaleb, A. et al. (2014). 'Evolving spiking neural networks for temporal pattern recognition in the presence of noise'. In: *Artificial Life 2014: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems*. Ed. by H. Sayama. MIT Press, pp. 965–972. DOI: 10.7551/978-0-262-32621-6-ch157.

Anderson, J. R. (2004). *Cognitive psychology and its implications*. Macmillan.

Ayers, J. (2004). 'Chemoreception in the lobster, Homarus americanus: implications for behavior and neural coding'. *Journal of Comparative Physiology A* 190.8, pp. 663–677.

Bensmail, C., V. Steuber, N. Davey et al. (2017). 'Evolving spiking neural networks to control animats for temporal pattern recognition and foraging'. In: *Computational Intelligence (SSCI), 2017 IEEE Symposium Series on*. IEEE, pp. 1–8.

Bensmail, C., V. Steuber and B. Wróbel (2018). 'Spiking Neural Network Controllers Evolved for Animat Foraging Based on Temporal Pattern Recognition in the Presence of Noise on Input'. *Journal of Computational Neuroscience* 45.1, pp. 21–45. DOI: 10.1007/s10827-018-0694-4.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York: Springer.

Braitenberg, V. (1984). *Vehicles: Experiments in Synthetic Psychology*. Cambridge, MA: MIT Press.

Brette, R. and W. Gerstner (2005). 'Adaptive exponential integrate-and-fire model as an effective description of neuronal activity'. *Journal of neurophysiology* 94.5, pp. 3637–3642.

Caporale, N. and Y. Dan (2008). 'Spike timing–dependent plasticity: a Hebbian learning rule'. *Annual review of neuroscience* 31, pp. 25–46.

Carr, C. E. and M. Konishi (1990). 'Circuitry for coincidence detection in the owl's brainstem'. *Journal of Neuroscience* 10.10, pp. 3227–3246.

Chung, J. et al. (2014). 'Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling'. *arXiv preprint arXiv:1412.3555*. URL: https://arxiv.org/abs/1412.3555.

Davies, M. et al. (2018). 'Loihi: A Neuromorphic Manycore Processor with On-Chip Learning'. *IEEE Micro* 38.1, pp. 82–99. DOI: 10.1109/MM.2018.112130359.

Davison, A. P. et al. (2009). 'PyNN: a common interface for neuronal network simulators'. *Frontiers in Neuroinformatics* 2. DOI: 10.3389/neuro.11.011.2008. URL: https://doi.org/10.3389/neuro.11.011.2008.

Dayan, P. and L. F. Abbott (2001). *Theoretical neuroscience: computational and mathematical modeling of neural systems*. MIT Press.

Destexhe, A. and D. Contreras (2006). 'Neuronal computations with stochastic network states'. *Science* 314.5796, pp. 85–90. DOI: 10.1126/science.1127241.

Faisal, A. A., L. P. J. Selen and D. M. Wolpert (2008). 'Noise in the nervous system'. *Nature Reviews Neuroscience* 9.4, pp. 292–303. DOI: 10.1038/nrn2258.

Fischer, T. and C. Krauss (2018). 'Deep learning with long short-term memory networks for financial market predictions'. *European Journal of Operational Research* 270.2, pp. 654–669.

Floreano, D., P. Dürr and D. Mattiussi (2008). 'Neuroevolution: From architectures to learning'. *Evolutionary Intelligence* 1.1, pp. 47–62. DOI: 10.1007/s12065-008-0002-4.

Floreano, D. and C. Mattiussi (2008). *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. Cambridge, MA: MIT Press. ISBN: 9780262062718.

Floreano, D. and F. Mondada (1996). 'Evolution of Homing Navigation in a Real Mobile Robot'. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics* 26.3, pp. 396–407. DOI: 10.1109/3477.502755.

Furber, S. B. et al. (2014). 'Spinnaker: A multi-core system-on-chip for massively-parallel neural net simulation'. *Proceedings of the IEEE* 102.5, pp. 652–665.

Gerstner, W. and W. M. Kistler (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press.

Gollo, L. L. et al. (2012). 'Mechanisms of zero-lag synchronization in cortical motifs'. *PLoS Computational Biology* 8.9, e1002596. DOI: 10.1371/journal.pcbi.1002596.

Goodfellow, I., Y. Bengio and A. Courville (2016). 'Deep learning'.

Graves, A., A.-r. Mohamed and G. Hinton (2013). 'Speech recognition with deep recurrent neural networks'. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649.

Hebb, D. O. (1949). *The Organization of Behavior: A Neuropsychological Theory*. New York: Wiley.

Hinton, G. et al. (2012). 'Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups'. *IEEE Signal Processing Magazine* 29.6, pp. 82–97.

Hochreiter, S. and J. Schmidhuber (1997). 'Long Short-Term Memory'. *Neural Computation* 9.8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

Hodgkin, A. L. and A. F. Huxley (1952). 'A quantitative description of membrane current and its application to conduction and excitation in nerve'. *The Journal of physiology* 117.4, pp. 500–544.

Hopfield, J. J. (1982). 'Neural networks and physical systems with emergent collective computational abilities'. *Proceedings of the National Academy of Sciences* 79.8, pp. 2554–2558. DOI: 10.1073/pnas.79.8.2554.

Ijspeert, A. J. et al. (2007). 'Swimming and walking with an amphibious salamander robot'. *IEEE Transactions on Robotics* 23.5, pp. 1007–1021.

Indiveri, G., C. Bartolozzi et al. (2022). 'Embodied neuromorphic intelligence'. *Nature Communications* 13. DOI: 10.1038/s41467-022-28487-2. URL: https://doi.org/10.1038/s41467-022-28487-2.

Indiveri, G. and T. K. Horiuchi (2011). 'Frontiers in neuromorphic engineering'. *Frontiers in neuroscience* 5, p. 118.

Indiveri, G. and S.-C. Liu (2015). 'Memory and learning in neuromorphic systems and spiking neural networks'. *Frontiers in neuroscience* 9, p. 118.

Izhikevich, E. M. (2003). 'Simple model of spiking neurons'. *IEEE Transactions on neural networks* 14.6, pp. 1569–1572.

— (2007). *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*. Cambridge, MA: MIT Press.

Izquierdo, E. J., R. D. Beer and K. S. Espenschied (2008). 'Evolving dynamical neural networks for adaptive behavior'. In: *From Animals to Animats 10 (SAB 2008)*. Vol. 5040. Lecture Notes in Computer Science. Springer, pp. 272–283.

Jin, Y., B. Sendhoff and Y. Tan (2010). 'Evolutionary multiobjective optimization for dynamic neural network structures'. *Neurocomputing* 71.13–15, pp. 2720–2733. DOI: 10.1016/j.neucom.2008.09.017.

Jones, M. and E. Teeling (2006). 'Balancing adaptability and stability in biological systems'. *Nature* 439, pp. 293–294.

Kashtan, N. and U. Alon (2005). 'Spontaneous Evolution of Modularity and Network Motifs'. *Proceedings of the National Academy of Sciences* 102.39, pp. 13773–13778.

Knight, J. C., A. V. Komissarov and T. Nowotny (2021). 'PyGeNN: A Python Library for GPU-Enhanced Neural Networks'. *Frontiers in Neuroinformatics* 15. DOI: 10.3389/fninf.2021.659005. URL: https://doi.org/10.3389/fninf.2021.659005.

Knight, J. C. and T. Nowotny (2018). 'GPUs Outperform Current HPC and Neuromorphic Solutions in Terms of Speed and Energy When Simulating a Highly-Connected Cortical Model'. *Frontiers in Neuroscience* 12. DOI: 10.3389/fnins.2018.00941. URL: https://doi.org/10.3389/fnins.2018.00941.

Lago-Fernández, L. F. et al. (2000). 'Fast response and temporal coherent oscillations in small-world networks'. *Physical Review Letters* 84.12, pp. 2758–2761. DOI: 10.1103/PhysRevLett.84.2758.

Lagorce, X. et al. (2017). 'HOTS: A hierarchy of event-based time-surfaces for pattern recognition'. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.7, pp. 1346–1359.

Laurent, G. (1999a). 'A systems perspective on early olfactory coding'. *Science* 286.5440, pp. 723–728.

— (1999b). 'A systems perspective on early olfactory coding'. *Science* 286.5440, pp. 723–728. DOI: 10.1126/science.286.5440.723.

Maass, W. (1997). 'Networks of spiking neurons: the third generation of neural network models'. *Neural networks* 10.9, pp. 1659–1671.

Markram, H. et al. (1997). 'Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs'. *Science* 275.5297, pp. 213–215.

McDonnell, M. D. and L. M. Ward (2011). 'The benefits of noise in neural systems: bridging theory and experiment'. *Nature Reviews Neuroscience* 12.7, pp. 415–425. DOI: 10.1038/nrn3061.

Meister, M. and M. J. Berry (1999). 'Neural codes: populations of neurons and how they represent information'. *Science* 282.5391, pp. 256–262.

Merolla, P. A. et al. (2014). 'A million spiking-neuron integrated circuit with a scalable communication network and interface'. *Science* 345.6197, pp. 668–673. DOI: 10.1126/science.1254642.

Michel, O. (2004). 'Webots: Professional Mobile Robot Simulation'. *International Journal of Advanced Robotic Systems* 1.1, pp. 39–42. DOI: 10.5772/5614.

Mnih, V. et al. (2015). 'Human-level control through deep reinforcement learning'. *Nature* 518.7540, pp. 529–533. DOI: 10.1038/nature14236.

Mos, J. (2004). 'Stochastic resonance: theory and applications'. *Nonlinear Phenomena in Complex Systems* 7.2, pp. 92–106.

Mozafari, M. et al. (2019). 'Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks'. *Pattern Recognition* 94, pp. 87–95.

Nanda, N. et al. (2023). 'Progress Measures for Grokking via Mechanistic Interpretability'. *arXiv preprint arXiv:2301.05217*.

Naud, R. et al. (2008). 'Firing patterns in the adaptive exponential integrate-and-fire model'. *Biological Cybernetics* 99.4-5, pp. 335–347. DOI: 10.1007/s00422-008-0264-7.

— (2011). 'Parameter extraction and classification of three cortical neuron types reveals two distinct adaptation mechanisms'. *Journal of Neurophysiology* 106.1, pp. 364–376.

Neftci, E. O., H. Mostafa and F. Zenke (2019). 'Surrogate gradient learning in spiking neural networks'. *IEEE Signal Processing Magazine* 36.6, pp. 61–63.

Nolfi, S. and D. Floreano (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA: MIT Press. ISBN: 9780262140706.

Olah, C. et al. (2020). 'Zoom In: An Introduction to Circuits'. *Distill*. DOI: 10.23915/distill.00024.

Pavlov, I. P. (1927). *Conditioned Reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex*. London, UK: Oxford University Press.

Pfeifer, R. and J. Bongard (2007). *How the Body Shapes the Way We Think: A New View of Intelligence*. Cambridge, MA: MIT Press.

Ponulak, F. and A. Kasiński (2011). 'Introduction to Spiking Neural Networks: Information Processing, Learning and Applications'. *Acta Neurobiologiae Experimentalis* 71.4, pp. 409–433.

Pouget, A., P. Dayan and R. S. Zemel (2000). 'Information processing with population codes'. *Nature Reviews Neuroscience* 1.2, pp. 125–132. DOI: 10.1038/35039062.

Reil, T. and P. Husbands (2002). 'Evolution of Central Pattern Generators for Bipedal Walking in a Real-Time Physics Environment'. *IEEE Transactions on Evolutionary Computation* 6.2, pp. 159–168. DOI: 10.1109/4235.996017.

Roy, K., A. Jaiswal and P. Panda (2019). 'Towards Spike-Based Machine Intelligence with Neuromorphic Computing'. *Nature* 575.7784, pp. 607–617. DOI: 10.1038/s41586-019-1677-2.

Rueckauer, B. et al. (2017). 'Conversion of continuous-valued deep networks to efficient event-driven networks for image classification'. *Frontiers in neuroscience* 11, p. 682.

Schuman, C. D. et al. (2017). 'A survey of neuromorphic computing and neural networks in hardware'. *arXiv preprint arXiv:1705.06963*.

Singer, W. (1999). 'Neuronal synchrony: a versatile code for the definition of relations?' *Neuron* 24.1, pp. 49–65.

Sporns, O., G. Tononi and R. Kötter (2004). 'Complex networks: Structure and dynamics'. *Nature Reviews Neuroscience* 5.10, pp. 799–807.

Spüler, M. et al. (2015). 'Co-adaptive Brain–Computer Interface for Closed-Loop Neuromodulation in Chronic Stroke'. *Frontiers in Neuroscience* 9, p. 354. DOI: 10.3389/fnins.2015.00354.

Stagsted, R. K. et al. (2020). 'Towards neuromorphic control: A spiking neural network based PID controller for UAV'. In: *Proc. Robotics: Science and Systems (RSS)*.

Stanley, K. O. and R. Miikkulainen (2002). 'Evolving neural networks through augmenting topologies'. In: *Evolutionary computation*. Vol. 10. 2. IEEE, pp. 99–127.

Stanley, K. O., B. D. Bryant and R. Miikkulainen (2005). 'Real-Time Neuroevolution in the NERO Video Game'. *IEEE Transactions on Evolutionary Computation* 9.6, pp. 653–668. DOI: 10.1109/TEVC.2005.856205.

Stein, R. B., E. R. Gossen and K. E. Jones (2005). 'Neuronal variability: noise or part of the signal?' *Nature Reviews Neuroscience* 6.5, pp. 389–397. DOI: 10.1038/nrn1668.

Stimberg, M., D. F. M. Goodman and T. Nowotny (2020). 'Brian2GeNN: accelerating spiking neural network simulations with graphics hardware'. *Scientific Reports* 10. DOI: 10.1038/s41598-019-54957-7. URL: https://doi.org/10.1038/s41598-019-54957-7.

Sutton, R. S. and A. G. Barto (2018). *Reinforcement Learning: An Introduction*. 2nd. MIT Press. ISBN: 9780262039246. URL: http://incompleteideas.net/book/the-book-2nd.html.

Taubner, L., S. Rungta and W. Maass (2019). 'Learning Probabilistic Spiking Neural Networks through Probabilistic Backpropagation'. In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=B1lTojCqtX.

Tavanaei, A. et al. (2019). 'Deep learning in spiking neural networks'. *Neural Networks* 111, pp. 47–63. DOI: 10.1016/j.neunet.2018.12.002.

Thorpe, S. and J. Gautrais (1998). 'Rank order coding', pp. 113–118.

Tononi, G., O. Sporns and G. M. Edelman (1999). 'Measures of degeneracy and redundancy in biological networks'. *Proceedings of the National Academy of Sciences* 96.6, pp. 3257–3262. DOI: 10.1073/pnas.96.6.3257.

Turrigiano, G. G. (2008). 'The self-tuning neuron: synaptic scaling of excitatory synapses'. *Cell* 135.3, pp. 422–435.

Turrigiano, G. G. and S. B. Nelson (2000). 'Homeostatic plasticity in the developing nervous system'. *Nature Reviews Neuroscience* 1.2, pp. 97–107.

Turrigiano, G. G. (2011). 'Too many cooks? Intrinsic and synaptic homeostatic mechanisms in cortical circuit refinement'. *Annual Review of Neuroscience* 34, pp. 89–103. DOI: 10.1146/annurev-neuro-060909-153238.

Vasu, M. C. and E. J. Izquierdo (2017). 'Evolution and Analysis of Embodied Spiking Neural Networks Reveals Task-Specific Clusters of Effective Networks'. *arXiv preprint arXiv:1704.04199*.

Vogels, T. P. and L. F. Abbott (2005). 'Signal propagation and logic gating in networks of integrate-and-fire neurons'. *Journal of Neuroscience* 25.46, pp. 10786–10795. DOI: 10.1523/JNEUROSCI.3508-05.2005.

Wang, X.-J. (2002). 'Probabilistic decision making by slow reverberation in cortical circuits'. *Neuron* 36.5, pp. 955–968. DOI: 10.1016/S0896-6273(02)01092-9.

— (2010). 'Neurophysiological and computational principles of cortical rhythms in cognition'. *Physiological Reviews* 90.3, pp. 1195–1268. DOI: 10.1152/physrev.00035.2008.

Watts, D. J. and S. H. Strogatz (1998). 'Collective dynamics of 'small-world' networks'. *Nature* 393.6684, pp. 440–442. DOI: 10.1038/30918.

Wilson, H. R. (1985). 'Hierarchical structure in visual cortex: evidence from functional maps and population responses'. *Biological Cybernetics* 51.5, pp. 301–311. DOI: 10.1007/BF00336999.

Wróbel, B. and M. Joachimczak (2011). 'Using the Genetic Regulatory Evolving Artificial Networks (GReaNs) Platform for Signal Processing, Animat Control and Artificial Multicellular Development'. In: *Proceedings of the DevLeaNN Workshop*, pp. 26–28.

Yao, X. (1999). 'Evolving artificial neural networks'. *Proceedings of the IEEE* 87.9, pp. 1423–1447.

Yaqoob, M., V. Steuber and B. Wróbel (2019). 'The Importance of Self-excitation in Spiking Neural Networks Evolved to Recognize Temporal Patterns'. In: *Artificial Neural Networks and Machine Learning – ICANN 2019: Theoretical Neural Computation*. Ed. by I. V. Tetko et al. Springer International Publishing, pp. 758–771. DOI: 10.1007/978-3-030-30487-4_59. URL: https://link.springer.com/chapter/10.1007/978-3-030-30487-4_59.

— (2023). 'Autapses enable temporal pattern recognition in spiking neural networks'. *bioRxiv*. Preprint. DOI: 10.1101/2023.11.16.567361. URL: https://www.biorxiv.org/content/10.1101/2023.11.16.567361v1.

Yavuz, E., J. Turner and T. Nowotny (2016). 'GeNN: a code generation framework for accelerated brain simulations'. *Scientific Reports* 6. DOI: 10.1038/srep18854. URL: https://doi.org/10.1038/srep18854.

Yin, P. et al. (2020). 'Visual analytics for spiking neural networks'. *Journal of Visual Analytics* 1.1, pp. 1–15.

Zenke, F., B. Poole and S. Ganguli (2017). 'Continual Learning Through Synaptic Intelligence'. In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Vol. 70. PMLR, pp. 3987–3995. URL: https://proceedings.mlr.press/v70/zenke17a.html.