University of
Sheffield

Spiking Neural Network Implementation of a Robotic Model of Hippocampal Reverse Replay for Reinforcement Learning

Dhruti Davey

Supervisor: Prof. Eleni Vasilaki

A report submitted in partial fulfilment of the requirement for the degree of MSc in Cybersecurity and Artificial Intelligence

in the

School of Computer Science

September 20, 2025

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Dhruti Davey

Date: September 20, 2025

Abstract

The hippocampus plays a central role in memory and navigation, with place cells forming spatial maps and reverse replay events propagating reward information backwards along recent trajectories. Computational studies have shown that replay accelerates reinforcement learning, but most rely on rate-based neural models that abstract away the temporal precision of spikes.

This dissertation develops a spiking neural network (SNN) implementation of hippocampal reverse replay, replacing rate-based neurons with Poisson place cells and leaky integrate-and-fire action cells. The model was tested in a grid-world navigation task and evaluated against a rate-based baseline.

The results show that while the spiking model exhibited noisier convergence and slower stabilization, it achieved comparable learning performance and preserved the reinforcement benefits of reverse replay. These findings support the role of replay in biologically realistic networks and highlight potential applications in neuromorphic reinforcement learning systems.

Abbreviations

SNN	**S**piking **n**eural **n**etwork
RL	**R**einforcement **l**earning
IF	**I**ntegrate-**a**nd-**f**ire
LIF	**L**eaky **I**ntegrate-**a**nd-**f**ire
ANN	**A**rtificial **n**eural **n**etwork
STDP	**S**pike-**t**iming-**d**ependent **p**lasticity
R-STDP	**R**eward-**m**odulated
	– **s**pike-**t**iming-**d**ependent **p**lasticity

Acknowledgement

I express my deepest gratitude to my supervisor, Prof. Eleni Vasiliaki, for their invaluable guidance, unwavering support, and insightful feedback throughout this research project. My thanks also goes to the School of Computer Science Teaching team for being helpful and understanding throughout this semester. I am also grateful to my family and friends for their constant encouragement and patience.

Contents

List of Figures

Chapter 1

Introduction

In mammals, the hippocampus contains place cells – neurons that fire selectively when the animal occupies specific locations which effectively forms a "cognitive map" of the environment [1]. When animals rest or sleep, sequences of place-cell activity corresponding to previously traveled paths are replayed in the hippocampus, often at accelerated speeds [2, 3]. These replay events can occur in both forward and reverse temporal order, but reverse replay (reactivating cells in the opposite order of traversal) has attracted particular interest [4]. Reverse replay is observed to occur especially after the animal reaches a reward site, and it has been hypothesized to strengthen the associations of recently taken steps with that reward [3]. Johnson and Redish (2005) proposed that replay serves as offline "practice" sequences for RL algorithms, and showed in a computational model that adding replay greatly accelerated the learning of a spatial path [2]. Whelan, Et al.'s work suggests that hippocampal replay provides internal training data that can improve learning efficiency in goal-directed navigation tasks [5].

This project builds on these ideas by implementing a spiking neural network (SNN) model of hippocampal reverse replay in a spatial navigation task. Spiking neural networks are the most biologically realistic form of artificial neural network: instead of continuous activation values, neurons communicate with discrete voltage spikes that occur at specific times [6]. In SNNs not just their average rate, but even the timing of the spike carries information. This temporal coding closely mimics real brain computation and enables rich dynamics and potential hardware efficiency. The canonical neuron in an SNN is the leaky integrate-and-fire (LIF) model, where input currents cause a membrane potential to rise until a threshold is reached, triggering a spike and a reset [7]. In our model, LIF neurons are used to represent action-selection cells, and use simple Poisson-spiking neurons for the place cells. A Poisson neuron generates spikes randomly with a fixed average rate, the simplest stochastic model of neural firing [7]. Modeling place cells as Poisson spike generators captures the known variability of hippocampal firing under steady conditions.This SNN framework allows us to extend existing rate-based model [5] into a more biologically grounded regime, investigating

how discrete spikes and their timing affect the learning process.

1.1 Aims and Objectives

The primary aim of this project is to develop and test a spiking implementation of a hippocampal replay model for reinforcement learning, and to comapare and evaluate its learning performance in a spatial task. Building on Whelan et al. (2022) [5], this dissertation aims to convert their rate-based neural model into a spiking framework. The specific objectives are:

- **Implement Spiking Network:** Develop a spiking neural network that replicates the network architecture of the original model, using Poisson spiking neurons as place cells and LIF neurons as action cells. The synaptic connections between place and action cells is learned through a reward-modulated spike-based rule (derived from the policy-gradient learning rule) that can incorporate replay activity as a training signal.
- **Construct Simulation Environment:** Create a 2D grid-world navigation environment (implemented in Python with Pygame) in which an agent must learn a goal-directed task. The grid-world represents the space as cells and has actions (moves) for turing left, turing right, going forwards and going backwards allowing movement in all four cardinal directions. This provides a simple spatial task in which place fields and navigation can be tested. The environment encodes locations via active place-cell patterns, and the agent's actions are be chosen by the spiking action cells.
- **Evaluate Learning Performance:** Run experiments to measure how the spiking agent learns the navigation task over time. The metrics used are the time to complete trials and the convergence of synaptic weights. These results are compared with the original rate-based model to assess whether the spiking implementation achieves similar or improved learning efficiency.

1.2 Overview of the Report

This dissertation is organized into six chapters:

- This chapter introduced the problem, the motivation for the study, and the aims and objectives of the study.
- Chapter 2 provides the necessary background, and a review of relevant literature.
- Chpater 3 is split into three sections:
  - In Section 3.1 the requirements of the project are outlined and analysed in more detail.
  - Section 3.2 details the design of the proposed system, including the navigation environment, agent's network architecture, selection of neuron models.

  - Section 3.3 describes the implementation of the designed model and the testing metrics used to validate its functionality and performance.

- Chapter 4 presents the results from the simulations, comparing the performance of the spiking neural network against baseline models and discussing the implications of the findings.

- Chapter 5 concludes the report by summarizing the key contributions, acknowledging the study's limitations, and proposing directions for future research.

1.3 Relationship to programme

This project falls in the domain of artificial intelligence and computational neuroscience and hence contributes to AI compnent of the programme, MSc Cybersecurity and Artificial Intelligence. The project required a good knowledge of reinforcement learning theory, neural network models, and a coding proficiency in Python. The machine learning module in the course covered bio-inspired methods which helped withe understanding and developing the agents network architecture. On the other hand, reinforcement learning was not covered in much detail in the module and required a study of the concepts along with guidance from my supervior.

Chapter 2

Background and Literature Review

This chapter starts by providing the foundational knowledge required to contextualize and justify the research presented in this dissertation. The structure is designed to logically guide the reader from the core biological phenomena to the specific computational model that this work extends. It begins by introducing the key biological concepts of hippocampal place cells, followed by reverse replay, establishing their hypothesized role in a navigation task. This is followed by reinforcement learning (RL) fundamentals, creating a bridge between the biological process and its computational function. Then the focus is brought to bio-inspired RL, followed by a review on existing hippocampal models.

The chapter then introduces Spiking Neural Networks (SNNs) as a biologically plausible computational framework. This is followed by an examination of how SNNs have been used in both RL and neuroscience models, highlighting their advantages and the current state of the art. This is followed by a detailed analysis of the specific rate-based model by Whelan and Vasilaki [5], which serves as the direct foundation for this dissertation.

2.1 Hippocampal Place Cells and Reverse Replay

2.1.1 The Hioppocampus and place cells

The hippocampus (Figure 2.1) is a banana-shaped structure in the temporal lobe that plays a key role in forming new memories and helps in navigating through space [1]. It works closely with other areas of the brain to temporarily store new experiences before gradually transferring them to long-term storage in other parts of the brain. The hippocampus operates in different states. One for when active exploring and learning, and second for consolidating memories at rest.

Place cells are special neurons in the hippocampus, primarily in CA1 and CA3 regions, that fire when an animal is in a specific location in its environment [1]. Each place cell has its own "place field" which becomes most active in a particular spot. These cells work together like a GPS system, helping create a "cognitive map" of the environment [8]. When an animal moves through space, different place cells fire in sequence, and the timing of their

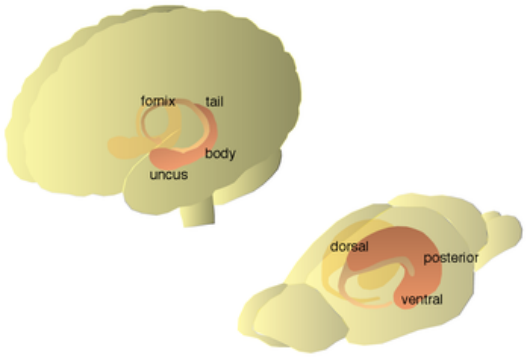

Figure 2.1: *Regions of the hippocampus in the human (left) and rat (right) brains [1].*

firing encodes information about both the animal's current location and where it is headed next. This same system that maps physical space also helps organize memories of events in time.

2.1.2 Hippocampal replay

Hippocampal replay refers to the reactivation of cell activity during rest or sleep, replaying recent experiences at accelerated speeds [9]. Forward replays replay sequences in the experienced order. They are often during planning at decision points, as in "vicarious trial-and-error" behaviors [10]. Reverse replays activate in backward order, typically post-reward, propagating value from goals to preceding states [4].

Girardeau, Et al. (2009) says replay strengthens synapses for consolidation via spike-timing-dependent plasticity (STDP) [11]. Caz'e, Et al. (2018) links it to RL, with reverse replay facilitating credit assignment in sparse reward scenarios [12]. Trends show that replay bias toward rewards, with larger rewards increasing reverse events [9].

2.2 Reinforcement Learning

2.2.1 RL Framework and the Credit Assignment Problem

RL provides a mathematical framework for understanding how agents learn optimal behavior through interaction with their environment [13]. An agent learns through trial-and-error and attempts to maximize cumulative reward by taking actions, observing consequences and updating its policy at the end of an episode.

One challenge in reinforcement learning is the temporal credit assignment problem. It is determining which past actions are responsible for current rewards [14]. This problem is particularly acute in sequential decision-making where actions have delayed consequences, making it difficult to establish causal relationships between actions and outcomes.

2.2.2 Policy Gradient Methods

Policy gradient methods are a type of RL algorithms that directly optimize the policy rather than learning value functions [15]. The REINFORCE algorithm provides a foundational approach to policy gradient estimation, enabling direct optimization of action selection policies through gradient descent methods.

2.2.3 Eligibility Traces and Biological Plausibility

Eligibility traces provide a mechanism for bridging temporal difference and Monte Carlo methods by maintaining a record of recently visited states and their recency of visitation [13]. The trace for each state decays over time and is refreshed whenever that state is revisited.

This mechanism allows recent experiences to have stronger influence on learning updates while maintaining fading memory of older experiences, enabling efficient temporal difference learning [16].

The biological plausibility of eligibility traces has made them particularly relevant for neuroscience-inspired RL implementations, with neural mechanisms like synaptic tags providing realistic substrates for solving temporal credit assignment [17].

2.3 Existing Computational Models of Hippocampal Replay

2.3.1 Early Models: Synaptic Plasticity and Sequential Activity

A number of models have been developed to explain the phenomena of hippocampal replay, each one emphasizing different aspects of the underlying mechanisms. Early models focused on the role of synaptic plasticity in creating and maintaining sequential activity patterns [18]. These models demonstrated how Hebbian learning rules could establish connectivity patterns that support replay of experienced sequences.

2.3.2 Directional Replay Mechanisms

Diba and Buzs'aki's model [19] provided important insights into the mechanisms underlying forward and reverse replay. Their work suggested that the direction of replay depends on the balance between feedforward and feedback connections, with reverse replay emerging from strong feedback connectivity that develops through experience-dependent plasticity.

2.3.3 Replay-RL Integration Models

Foster and Wilson [4] developed a computational framework linking replay to reinforcement learning, proposing that reverse replay serves to propagate reward information backward through state sequences. Their model demonstrated how biologically plausible learning rules could implement temporal difference learning through replay mechanisms, creating a bridge between machine learning concepts and neuroscience.

2.3.4 Neuromodulatory Control of Replay

Penny et al. [20] explored the role of neuromodulation in controlling replay dynamics. Their model showed how dopaminergic and cholinergic signals could regulate the timing and content of replay events, providing a mechanism for prioritizing important experiences based on reward prediction errors. This neuromodulatory control adds biological realism and functional sophistication to replay models.

2.3.5 The Whelan and Vasilaki Model: Rate-Based Robotic Implementation

Whelan and Vasilaki [5] developed a robotic model that successfully implemented hippocampal reverse replay for spatial learning tasks. Their model combined place cell representations with temporal difference learning, which showed improved learning performance compared to standard non-replay reinforcement learning model. However, this model relies on rate-based neural representations that abstract away the precise timing of individual spikes. The precise timing of neural spikes has been shown to carry important information in hippocampal circuits [21], suggesting that spike-based models could provide additional insights. Since this dissertation is based on the work of this paper, more about this paper is mentioned in Section 3.2.

2.4 Spiking Neural Networks

2.4.1 SNNs vs Traditional Artificial Neural Networks (ANNs)

Spiking neural networks offer distinct advantages over traditional artificial neural networks through their biologically-inspired discrete spike events rather than continuous signal propagation [6]. This event-driven computation leads to sparse temporal activity patterns that is more energy-efficient than the dense matrix operations required by traditional Deep Neural Networks, particularly when implemented on neuromorphic hardware platforms such as Intel's Loihi [22] and SpiNNaker [23]. A key distinguishing feature of SNNs is their ability to process temporal information through spike timing relationships, whereas traditional ANNs typically require explicit architectural modifications to handle temporal data effectively.

However, SNNs face significant training challenges that limit their widespread adoption. The complex dynamics of spiking neurons and the non-differentiable nature of spike operations make optimization difficult, requiring specialized approaches such as surrogate gradient methods [24] or conversion from pre-trained ANNs [6]. While the performance gap between SNNs and traditional ANNs is quite diminishing on simpler tasks like MNIST, disparities are more pronounced on complex, large-scale datasets. Despite these challenges, SNNs' temporal processing capabilities make them particularly suitable for applications involving time-series data and real-time processing with strict energy budgets, leading to recent hybrid approaches that combine traditional ANNs for feature extraction with SNNs for energy-efficient inference

[25].

2.4.2 Neuron Models: IF and LIF

The integrate-and-fire (IF) model provides the foundation for many SNN implementations. It captures essential dynamics of neural membrane potential integration and spike generation. The leaky integrate-and-fire (LIF) model is an extension of this framework which also includes membrane potential decay, thus providing a more realistic representation of neural dynamics while maintaining computational tractability [26]. These models enable event-driven computation that are more energy-efficient than traditional ANNs, as spikes are sparse in both space and time [27].

2.4.3 Poisson Spike Models

Poisson spike models represent neural activity as stochastic point processes where spikes are generated according to a time-varying firing rate, with the variance of spike counts proportional to the mean (this propety is observed in many cortical neurons) [28, 29]. Poisson encoding converts continuous input values into spike trains by normalizing the signal and using the normalized value as a firing probability, with higher input values generating more frequent spikes [6]. This encoding scheme provides a biologically plausible method for representing sensory information in spiking neural networks, as the stochastic nature of spike generation captures the inherent variability observed in real neural responses. Some recent applications have demonstrated that Poisson spike encoding can improve generalization performance in robotic navigation tasks when compared to deterministic encoding schemes [6], making it particularly suitable for modeling place cell activity where spatial information is encoded through probabilistic firing patterns.

2.4.4 Learning Algorithms in SNNs

Recent advances in SNN learning algorithms have addressed some historical training challenges. Spike-timing-dependent plasticity (STDP) provides a biologically plausible learning mechanism based on relative timing of pre- and post-synaptic spikes [30]. Surrogate gradient methods enable backpropagation-like training in SNNs while maintaining biological plausibility [31].

2.4.5 SNNs in Reinforcement Learning

The integration of SNNs with RL has been achieved through reward-modulated spike-timing-dependent plasticity (R-STDP), where the modulation of STDP by a global reward signal enables RL [32]. R-STDP addresses the distal reward problem in RL by maintaining synaptic eligibility traces that store temporary memory of STDP outcomes until delayed reward signals are received [33, 34]. This approach allows spiking agents to solve RL tasks without requiring computation of global error gradients, making it more biologically plausible than traditional backpropagation-based methods [35]. Some implementations have successfully applied R-

STDP to autonomous mobile robot navigation and robotic arm control, demonstrating that reward-modulated plasticity can enable adaptive behavior in changing environments [36, 37]. However, challenges remain in continuous state and action spaces, where traditional policy gradient methods may fail and specialized reward-modulated learning rules are required [38].

2.5 Research Justification

Despite significant advances in computational models of hippocampal replay, several opportunities remain to extend the current literature through SNN implementations. SNN modeling may help explain the mechanism of hippocampal replays enabling animals to correctly choose a direction in a forked path upon multiple visit occurrences [39], building upon existing models that have primarily utilized rate-based representations.

The work of Whelan and Vasilaki [5], have provided valuable insights into hippocampal replay mechanisms, there is potential to enhance these models by incorporating the precise timing relationships that characterize real hippocampal replay events. The reduction of spike-based models to rate-based models though has become the standard, it could have some drawbacks [37]. Hippocampal replay consists of short activity waves across place cells during sleep or rest which represent previous animal trajectories and are thought to be critical for memory consolidation [28]. The temporal precision of these waves could potentially be better captured through spike-based approaches.

While SNNs offer the advantage of being more biologically plausible, closely mimicking the way real neurons function in the brain [40], there remains an opportunity to explore these advantages in hippocampal-RL integration. Most existing SNN-RL implementations focus on simple control tasks rather than the complex spatial-temporal dynamics required for hippocampal replay modeling. This dissertation addresses this by building on the proven Whelan and Vasilaki framework using biologically realistic spiking neurons, and exploring the potential benefits of temporal precision for modeling hippocampal replay mechanisms.

2.6 Summary

This chapter has established the foundational knowledge necessary to understand and justify the research presented in this dissertation. This chapter explored how hippocampal place cells create cognitive maps of spatial environments and how replay (reverse replay) mechanisms serve crucial functions in memory consolidation and RL. The chapter then bridged biological and computational perspectives by examining RL fundamentals, with particular more focus on temporal credit assignment problems and eligibility traces that work parallel with hippocampal replay functions.

The review of existing computational models showed a progression from early plasticity-based approaches to sophisticated RL-integrated frameworks, culminating in the Whelan and Vasilaki model which successfully demonstrated robotic implementation of hippocampal reverse replay. However, the relied on rate-based neural representations that cannot capture

the precise temporal dynamics characteristic of real hippocampal circuits.

The examination of SNNs established their advantages over traditional ANNs, including biological plausibility, temporal precision, and energy efficiency. Poisson spike model, LIF model and reward-modulated STDP provided the technical foundation for implementing place cells and action cells in a biologically realistic framework. The chapter concluded by identifying key gaps in current literature, particularly the absence of spike-based implementations of successful hippocampal-RL models, thereby providing clear justification for the research contribution presented in this dissertation.

Chapter 3

Methodology

This chapter contains the methodology employed to develop a spiking neural network (SNN) implementation of the robotic hippocampal reverse replay model originally proposed by Whelan and Vasilaki [5]. The research builds upon their established computational framework that demonstrated how hippocampal reverse replay could be used to improve stability and performance on a homing task through a three-factor reinforcement learning rule [41]. While their original study utilized a simulation of the biomimetic robot MiRo-e, this study's approach begins with a foundational grid-world implementation using `pygame` to replicate and validate their findings before advancing to the SNN implementation.

Transitioning from a rate-based model to a biologically plausible spiking model requires careful consideration of timing and neural coding. This chapter details the systematic approach taken to preserve the core functionality of hippocampal reverse replay while implementing neuromorphic principles through Poisson spiking place cells and Leaky Integrate-and-Fire (LIF) action cells.

The methodology is structured around three primary sections that collectively address the research objectives. The Requirements and Analysis section (Section 3.1) establishes the functional and non-functional requirements necessary for successful SNN implementation, decomposing the project into manageable components while defining evaluation criteria for validation [42]. The Design section (Section 3.2) presents the architectural decisions and ordinary differential equations used in the implementation of the model. Finally, the Implementation and Testing section (Section 3.3) outlines the practical development approach, highlighting contributions and establishing validation strategies.

3.1 Requirements and Analysis

3.1.1 Functional Requirements

The functional requirements [42] for this project encompass the successful conversion of the original hippocampal reverse replay model into a spiking neural network implementation while preserving its core reinforcement learning capabilities. These requirements are organized

hierarchically to address both the grid-world replication and the advanced spiking neural network implementation.

FR1 Grid-World Model Replication: The system must accurately replicate the spatial navigation and learning dynamics of the original Whelan and Vasilaki model within a discrete grid-world environment. This includes implementing place cell representations that encode spatial locations, action cell networks that determine movement decisions, and the reverse replay mechanism that reactivates recent spatiotemporal trajectories in reverse order. The pygame implementation must have all the elemets of the original MiRo simulation (although discrete), establishing a validated baseline for subsequent spiking neural network development.

FR2 Poisson Spiking Place Cell Implementation: The place cells must be converted from rate-based representations to Poisson spiking neurons that maintain spatial encoding while introducing biologically plausible temporal dynamics. Each place cell should exhibit spatially-tuned firing rates that reflect the agent's proximity to the cell's preferred location, with spike trains generated according to Poisson statistics (section 2.4.3). The spatial receptive fields must preserve the original model's spatial resolution and coverage, ensuring that the environment representation remains consistent across the conversion.

FR3 Leaky Integrate-and-Fire Action Cell Networks: Action cells responsible for movement decisions must be implemented as LIF neurons capable of integrating synaptic inputs over time and generating output spikes that drive behavioral responses. These neurons must maintain the connectivity patterns established in the original model while adapting the temporal integration properties to accommodate spike-based communication. The action selection mechanism must preserve the reinforcement learning capabilities that enable goal-directed navigation and path optimization.

FR4 Hippocampal Reverse Replay Preservation: The reverse replay mechanism is core to this study and must be successfully adapted to operate with spiking neural dynamics. This requires maintaining the temporal compression ratios observed in biological systems while ensuring that replayed spike sequences effectively propagate through the network to influence synaptic plasticity. The replay triggering conditions and sequence generation algorithms must be preserved from the original implementation.

3.1.2 Non-Functional Requirements

The non-functional requirements [42] establish the quality attributes and constraints that govern the system's implementation while ensuring biological plausibility, computational efficiency, and experimental validity.

NFR1 Biological Plausibility: The spiking neural network implementation must adhere to established principles of biological neural computation. Neuron models should exhibit physiologically realistic membrane dynamics, with time constants, threshold values, and reset mechanisms consistent with experimental observations from hippocampal neurons. Plasticity mechanisms and network connectivity patterns should reflect current understanding of hippocampal circuitry.

NFR2 Programming Efficiency: The model must be implemented in a computationally efficient manner. The code should minimize computational overhead while maintaining numerical accuracy. Memory usage must remain within reasonable bounds to enable simulation of networks with hundreds to thousands of neurons. The implementation should leverage vectorized operations and parallel processing capabilities where appropriate to maximize computational throughput.

NFR3 Parameter Sensitivity and Robustness: The spiking implementation must demonstrate robustness to parameter variations while maintaining sensitivity to biologically relevant parameter ranges. Critical parameters such as membrane time constants, synaptic strengths, and plasticity rates should be tunable within physiologically plausible ranges without compromising system stability. The model should exhibit graceful degradation rather than catastrophic failure when operating outside optimal parameter regimes.

NFR4 Scalability and Modularity: The system architecture must support scalability to accommodate varying environment sizes and network complexities. The modular design should enable independent modification of neuron models, connectivity patterns, and learning rules without requiring extensive system-wide changes. This modularity facilitates systematic parameter studies and comparative analysis between different implementation approaches.

NFR5 Validation and Reproducibility: The implementation must support comprehensive validation through standardized testing protocols and reproducible experimental conditions. Random number generation must be controllable through seed values to ensure reproducible results across multiple simulation runs. The system should provide detailed logging and monitoring capabilities to facilitate performance analysis and debugging. Compatibility with standard data analysis tools and visualization packages (matplotlib) is essential for result interpretation and publication.

NFR6 Development Environment and Dependencies: The spiking neural network implementation must operate within a minimal computational environment to ensure broad accessibility and reproducibility. The core system should depend only on essential Python libraries including NumPy for numerical computations, Matplotlib for visualization and result analysis, and CSV for data input/output operations.

This minimal dependency approach contrasts with the original Whelan-Vasilaki implementation which required specialized robotics infrastructure including ROS (Robot Operating System), Ubuntu 20.04, Gazebo physics simulator, and MiRo robot-specific libraries. The spiking implementation must be platform-independent and executable on standard Python environments without requiring robotics middleware, specialized hardware, or complex simulation frameworks. This simplified environment ensures that the model can be easily deployed, tested, and validated across different computational platforms while maintaining full functionality.

3.2 Design

The model builds directly on the hippocampal–striatal architecture of Whelan and Vasilaki's model [5], with two key modifications:

1. **Spiking neural network formulation.** Place cells are modeled as Poisson spiking neurons instead of continuous-valued rate units, and action cells are modeled as leaky integrate-and-fire (LIF) neurons.

2. **Discrete environment.** The continuous robot navigation task is replaced by a **grid-world environment**, where the agent moves between discrete cells and learns to reach a fixed goal location.

These changes preserve the original logic of hippocampal reverse replay while recasting the model into a biologically plausible spiking domain.

3.2.1 Place Cells (Poisson Neurons)

To bridge the gap between the discrete grid-world environment and the continuous spatial encoding assumed in hippocampal models, Gaussian noise was added to the agent's discrete position at each timestep. Specifically, if the agent's grid position is denoted as (x_d, y_d), its effective position used for place-cell activation is sampled from a Gaussian distribution:

$$x_c \sim \mathcal{N}(x_d, \sigma_n^2), \quad y_c \sim \mathcal{N}(y_d, \sigma_n^2), \tag{3.1}$$

where σ_n is the noise standard deviation. The perturbed coordinates (x_c, y_c) are then used in the standard place-cell tuning function,

$$r_j(x_c, y_c) = r_{\max} \exp\left(-\frac{(x_c - x_j^c)^2 + (y_c - y_j^c)^2}{2\sigma_p^2} \right), \tag{3.2}$$

so that place-cell firing fields overlap smoothly as in continuous navigation tasks. This modification ensures that, although the agent moves on a discrete lattice, the neural representation retains graded spatial variability resembling a continuous environment.

Spike trains are generated as an **inhomogeneous Poisson processes**:

$$s_j(t) \sim \text{Poisson}(r_j(x, y)\Delta t)\,. \tag{3.3}$$

This replaces the rectified linear rate units (Eq. 3 to 5 in the original paper [5]) with a spike-based implementation.

During exploration, place cell firing reflects the agent's current position. At the reward location, replay is triggered by reactivating the terminal place field, propagating activity backwards along recent trajectories.

Synaptic Inputs and Short-Term Plasticity

Each place cell receives recurrent input from its eight neighbors, modulated by short-term plasticity:

$$I_j^{\text{syn}}(t) = \lambda \sum_{k=1}^{8} w_{jk}^{\text{place}}\, s_k(t) D_k(t) F_k(t), \tag{3.4}$$

where λ gates transmission ($\lambda = 0$ during exploration, $\lambda = 1$ during replay).

Short-term depression (D_k) and facilitation (F_k) evolve as:

$$\frac{d}{dt}D_k = \frac{1 - D_k}{\tau_{\text{STD}}} - s_k(t) D_k F_k, \tag{3.5}$$

$$\frac{d}{dt}F_k = \frac{U - F_k}{\tau_{\text{STF}}} + U(1 - F_k) s_k(t). \tag{3.6}$$

These dynamics ensure replayed sequences dissipate rather than amplify uncontrollably.

Intrinsic Plasticity

Intrinsic plasticity regulates excitability of place cells. Instead of firing rates, a filtered spike-rate estimate $\hat{r}_j$ is used:

$$\frac{d}{dt}\psi_j = \frac{\psi_{ss} - \psi_j}{\tau_\psi} + \frac{\psi_{\max} - 1}{1 + \exp[-\beta(\hat{r}_j - x_\psi)]}. \tag{3.7}$$

3.2.2 Action Cells (LIF Neurons)

Action cells represent possible directions in the grid world (e.g., up, down, left, right, diagonals). Each is modeled as a leaky integrate-and-fire neuron:

$$\tau_m \frac{dV_i}{dt} = -(V_i - V_{\text{rest}}) + R_m \sum_j w_{ij} s_j(t), \tag{3.8}$$

with threshold dynamics:

$$\text{if } V_i \geq V_{\text{th}} \quad \Rightarrow \quad V_i \to V_{\text{reset}},\ \text{spike emitted.} \tag{3.9}$$

At each decision step, the action corresponding to the **highest spike count** over a short window is selected. If spiking activity is weak, a semi-random exploration policy is used, ensuring coverage of the environment (analogous to Whelan's semi-random walk).

3.2.3 Eligibility trace and weight updates

In our model, we retain the same policy-gradient reinforcement learning rule as introduced [5]. Synaptic plasticity between place cells and action cells is governed by a three-factor learning rule with an eligibility trace. The update rule is:

$$\frac{d}{dt}w_{ij} = \frac{\eta}{\sigma^2}R(t)\, e_{ij}(t), \tag{3.10}$$

where η is the learning rate, σ^2 the variance of the Gaussian policy, and $R(t)$ the reward signal. The term e_{ij} denotes the synapse-specific eligibility trace, defined as:

$$\frac{d}{dt}e_{ij} = -\frac{e_{ij}}{\tau_e} + (y_i - \tilde{y}_i)(1 - \tilde{y}_i)\tilde{y}_i x_j, \tag{3.11}$$

with x_j the activity of the presynaptic place cell, y_i the sampled action cell output, and $\tilde{y}_i$ its mean activation. During reverse replay, we use the replay-modified form of the rule, again following the original derivation:

$$\frac{d}{dt}e_{ij} = -\frac{e_{ij}}{\tau_e} + (y_i^{\text{replay}} - \tilde{y}_i)(1 - \tilde{y}_i)\tilde{y}_i x_j, \tag{3.12}$$

$$\frac{d}{dt}w_{ij} = \eta\, e_{ij}. \tag{3.13}$$

This is kept to ensure a direct comparison of performance with and without the spiking implementation.

3.2.4 Reverse Replay and Learning

Upon reaching the goal, reverse replay is initiated:

- Place cells are sequentially reactivated in reverse order of the trajectory.
- This drives replayed spiking in the action cells.
- Weights are updated with the same three-factor rule, but using replayed spikes:

This spiking replay replaces the original supervised replay update (Eqns. 20 and 21) while maintaining its role of reinforcing successful trajectories.

3.2.5 Environment and Rewards

The agent operates in a discrete $N \times N$ grid-world:

- **States:** grid cells.

- **Actions:** moves selected from action cell spiking populations.
- **Rewards:** $R = +1$ at the goal, $R = -1$ at obstacles, $R = 0$ otherwise.
- **Replay:** triggered 1s after reaching the goal, reinforcing the preceding trajectory.

3.3 Implementation and testing

3.3.1 Implementation Framework and Tools

Custom Implementation Approach

The implementation employed a custom-built simulation framework developed in Python, providing complete control over the underlying computational mechanisms while avoiding the complexity and compatibility issues associated with existing neural simulation packages (Brian2). This approach enabled direct implementation of the mathematical formulations presented in the previous section.

The custom implementation used NumPy for efficient vectorized operations on neural state variables and synaptic computations, significantly reducing computational overhead compared to approaches with loops. Matplotlib provides comprehensive visualization capabilities for both offline analysis and real-time monitoring during simulation execution.

Development Environment and Reproducibility

Version control through Git ensures systematic tracking of implementation changes and facilitates sharing development code though GitHub and enures reproducibility of results. The codebase implements comprehensive parameter logging and random seed control mechanisms, enabling exact reproduction of experimental results across multiple simulation runs. Seed-based random number generation controls all stochastic processes, including Poisson spike generation, parameter initialization, and environmental randomization, ensuring statistical reproducibility while maintaining experimental validity.

The implementation incorporates configurable parameter management through structured configuration JSON files that specify all model parameters, experimental hyperparameters, and output requirements. This approach enables systematic parameter exploration studies and ensures consistent experimental protocols across different simulation conditions.

3.3.2 Grid-World Environment and Baseline Implementation

Environment Design and MiRo Task Adaptation

The grid-world environment implements a discrete spatial representation that captures the essential navigation dynamics of the original MiRo robot study while providing computational efficiency and experimental control. The environment supports any grid size (but tested on 10×10 and 20×20 grid configurations), enabling analysis of scalability effects and parameter sensitivity across different spatial complexities.

Baseline Model Validation

The baseline implementation replicates the original Whelan and Vasilaki rate-based model within the grid-world environment to establish performance benchmarks for comparison with the spiking neural network implementation. The baseline model demonstrates successful spatial navigation learning, achieving convergence to optimal paths within 10–15 training episodes for the 10×10 grid configuration and 15–20 episodes for the 20×20 grid, consistent with the performance characteristics reported in the original study.

Parameter sensitivity analysis confirms the robustness of the baseline implementation across the parameter ranges used in the original paper. Critical parameters including learning rate (η) and time constant for eligibility trace (τ_e), both exhibit similar sensitivity profiles to those reported by Whelan's model, validating the accuracy of the grid-world adaptation.

3.3.3 Spiking Neural Network Implementation

Hyperparameter Optimization Strategy

For both the implementations, systematic hyperparameter optimization focusing on the eligibility trace time constant (τ_e) and learning rate (η), which critically influence learning dynamics and convergence stability. Random sampling exploration covers the parameter space with $\tau_e \in [0.04, 5]$ seconds and $\eta \in [0.001, 10]$, generating 5 parameter combinations for statistical analysis across both grid configurations.

The hyperparameter search evaluates each parameter combination across 10 independent simulation runs with different random seeds, providing robust statistical characterization of parameter sensitivity.

Real-Time Visualization and Monitoring

The implementation provides comprehensive real-time visualization capabilities that enable monitoring of network dynamics during learning. Live plotting displays include: (1) network-wide firing rates, (2) eligibility trace evolution for representative synapses, (3) place cell to action cell weight matrices visualized as vector fields indicating preferred movement directions, and (4) raster plots showing individual place cell spike times.

The weight visualization employs vector field representations where each grid location displays an arrow indicating the preferred action direction and strength based on the synaptic weights from the corresponding place cell to each action cell population. Convergence is assessed by monitoring the alignment of weight vectors toward the goal location and the magnitude of these vectors, providing intuitive visualization of learning progress and final policy quality.

3.3.4 Testing Strategy and Validation

Comparative Performance Analysis

The validation strategy centers on direct comparison between the rate-based baseline model and the spiking neural network implementation across identical experimental conditions. Primary performance metrics include trial completion times measured from episode initiation to goal reaching, with statistical analysis employing Mann-Whitney U tests to assess significance of performance differences given the non-normal distribution typical of reinforcement learning trial times.

The comparison includes analysis of learning curves averaged across multiple random seeds, convergence reliability measured through success rate statistics, and final policy quality assessed through path optimality metrics.

Weight Convergence Analysis

Weight convergence evaluation employs vector field analysis of the learned place cell to action cell connectivity patterns. Convergence criteria require that weight vectors from spatial locations closer to the goal point toward the goal with a vector magnitude exceeding 50% of the maximum observed weight strength. The convergence assessment tracks both the directional alignment and magnitude development across training episodes.

Statistical convergence analysis employs moving window variance calculations on weight vector directions, with convergence declared when the angular variance falls below threshold values for a sustained period of 10 consecutive episodes. This approach provides robust detection of learning completion while accounting for the stochastic nature of both the learning process and the spiking neural dynamics.

Chapter 4

Results and discussion

This chapter presents the outcomes of implementing and evaluating the spiking neural network (SNN) model of hippocampal reverse replay for reinforcement learning, alongside the baseline rate-based model. The results show the findings, the extent to which the project goals were achieved, and potential directions for further work.

4.1 Findings

Experiments were conducted on two grid-world environments: a 10×10 discrete grid and a 20×20 grid, both with a fixed goal locations.

- In the 10×10 configuration, each of the 100 place cells mapped one-to-one to grid locations (one-hot encoding). This provided minimal overlap between receptive fields, resulting in rapid place cell activation and clear spatial discrimination. A smaller grid size also was a factor in shorter trial times.
- In the 20×20 configuration, the same 100 place cells encoded space in a distributed manner, with Gaussian receptive fields covering approximately four grid squares each. This overlap introduced spatial ambiguity that required multiple cells to activate jointly to uniquely specify position, increasing the complexity of spatial-to-action mapping.

The following types of analyses were performed to characterize both the baseline rate-based model and the spiking model:

1. **Network-wide firing rates** after replay (Figure 4.1).
2. **Intrinsic plasticity** before running a reverse replay event (Figure 4.2)
3. **Eligibility trace evolution** for representative synapses (Figure 4.3).
4. **Weight matrices from place to action cells**, visualized as vector fields showing preferred movement directions (Figure 4.4 and 4.5).

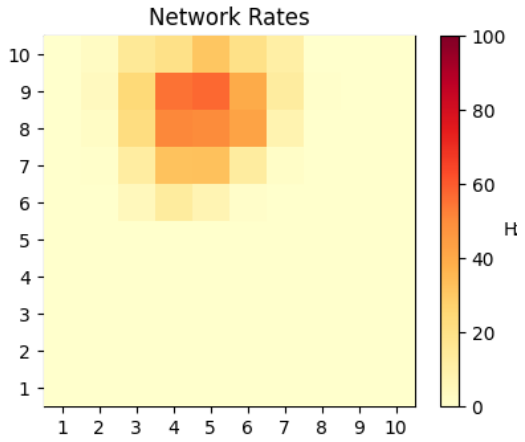

Figure 4.1: *Network firing rates after replay.*

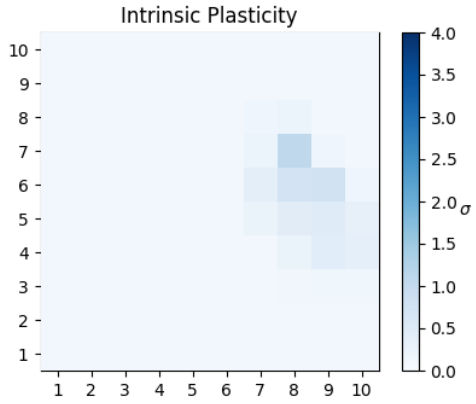

Figure 4.2: *Intrinsic plasticity before running a reverse replay event. (Different run from Figure 4.1)*

5. **Raster plots of place cell activity**, showing spiking during exploration and reverse replay (Figure 4.6).

Performance across parameter sweeps of eligibility trace time constant $\tau_e \in [0.04, 5]$ s and learning rate $\eta \in [0.001, 10]$ was evaluated by measuring trial time to reach the goal.

The best-performing configurations for both the rate-based and spiking models are reported (Figure 4.7, 4.8).

Comparative Observations

- Both rate-based and spiking models achieved similar performance in terms of average trial time across learning.
- The spiking model showed slightly noisier weight convergence compared to the smooth trajectories of the rate-based model. This is consistent with the stochasticity of spiking activity and highlights increased biological plausibility.

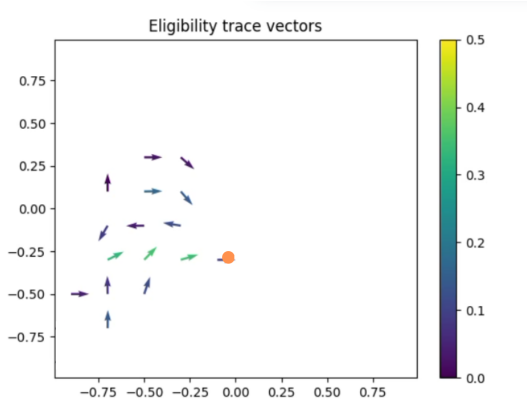

Figure 4.3: *Eligibility traces in spiking model after agent reaches goal.*

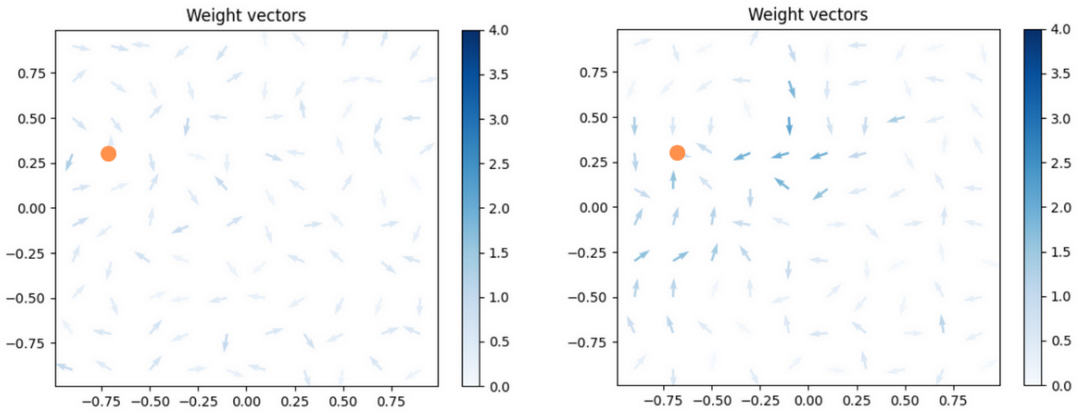

Figure 4.4: *Weight vector updates from episode 1 (left) to episode 20 (right)visualized of the rate-based model.*

- Reverse replay consistently improved learning stability, particularly for short eligibility time constants, mirroring the findings of Whelan & Vasilaki.
- In the larger 20×20 grid, the distributed representation slowed learning slightly in both models, as expected, since trajectory disambiguation required more place cell co-activations.

4.2 Goals Achieved

The primary goal of this project was to reproduce the hippocampal reverse replay reinforcement learning model of Whelan and Vasilaki, and to extend it into a spiking neural network framework. This goal was partially achieved:

- The **rate-based model was successfully reproduced** in the grid-world setting. Trial times and weight evolution were consistent with the original study, though conducted in discrete rather than continuous space.

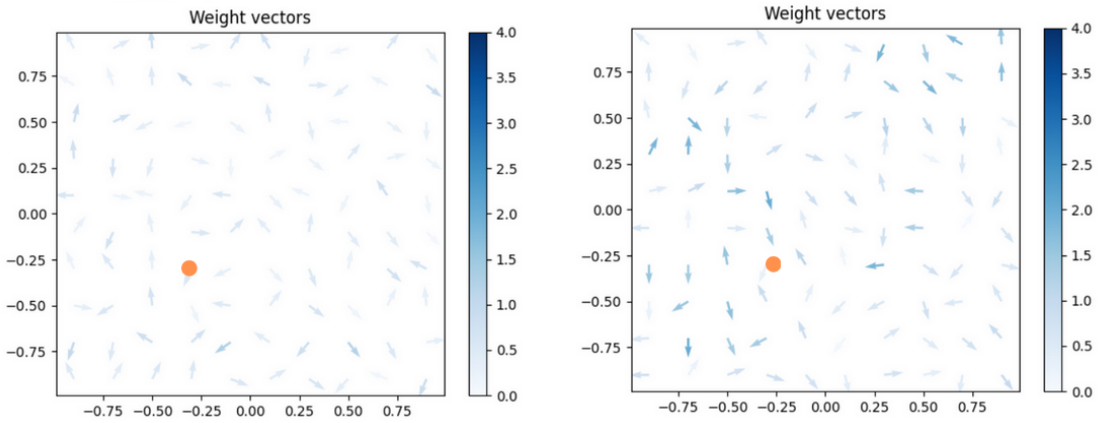

Figure 4.5: *Weight vector updates from episode 1 (left) to episode 20 (right)visualized of the spiking model.*

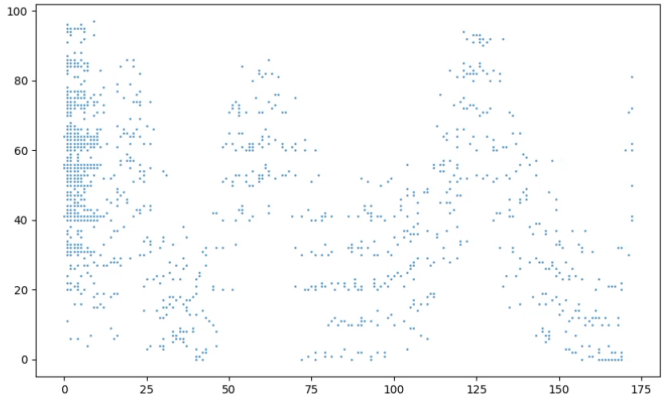

Figure 4.6: *Raster plots for Poisson place cells in one episode.*

- The **spiking neural network implementation was completed**, replacing continuous place and action cells with Poisson and LIF neurons, respectively. Raster plots confirm correct spiking dynamics during exploration and replay.
- The models were tested across parameter sweeps, and performance comparisons show that the spiking model maintains functionality, albeit with noisier convergence.

However, some objectives remain only partially fulfilled:

- A **continuous-world implementation** was planned as an additional task if time allowed, but not completed. The Gaussian noise added to the agent's discrete position provided a partial approximation of continuity.
- **Comprehensive testing** across a wider parameter space, particularly for intrinsic plasticity dynamics, remains outstanding.

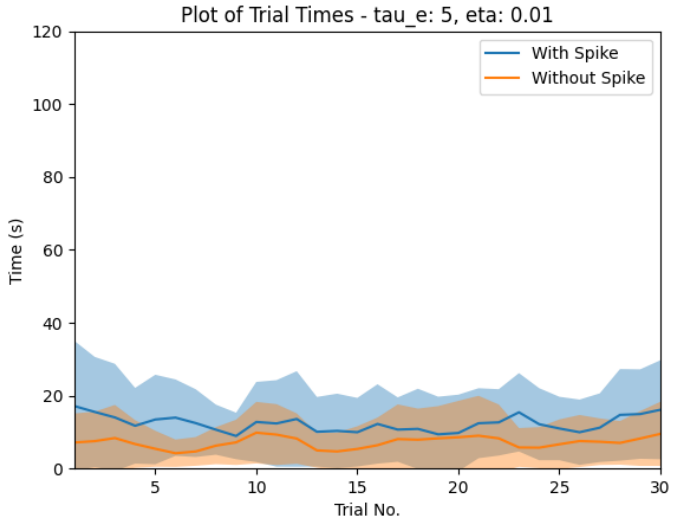

Figure 4.7: *Best trial times reported in the* 10×10 *grid. (Spiking model is more noisy)*

- **LIF neuron plots** could be included, and further analysis of their firing statistics (e.g., inter-spike interval distributions, membrane potential trajectories) could strengthen the biological interpretation.

4.3 Further Work

This work opens several directions for future research:

1. **Hyperparameter tuning.**
   - A more systematic exploration of intrinsic plasticity parameters $(\psi_{ss}, \psi_{\max}, \tau_\psi)$ could clarify their contribution to replay stability and trajectory reinforcement.
2. **Environmental complexity.**
   - Introducing walls or obstacles into the grid-world would allow testing how reverse replay aids in avoiding dead-ends or learning detours, providing a closer approximation to the original continuous navigation setting.
3. **Extended continuous implementation.**
   - Developing a full continuous-world simulation would allow direct comparison with the original MiRo robot experiments.
4. **Scaling experiments.**

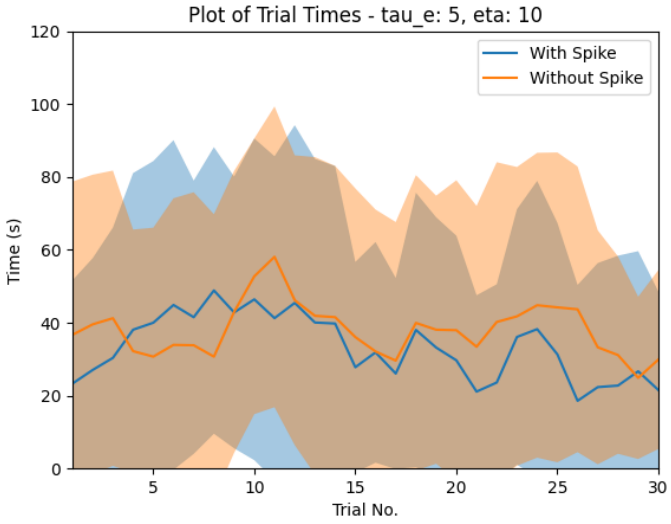

Figure 4.8: *Best trial times reported in the* 20×20 *grid.*

- Increasing the number of place cells beyond 100 could enable higher-resolution encoding in the 20×20 grid and beyond, or decreasing the number of action cells from 72 to 4, testing scalability and efficiency of the spiking replay mechanism.

4.4 Summary

In summary, this project demonstrates that a spiking implementation of hippocampal reverse replay for reinforcement learning can reproduce the performance of the original rate-based model in discrete environments. While noisier due to spiking variability, the SNN remains functionally effective, supporting the hypothesis that replay mechanisms can accelerate and stabilize reinforcement learning in biologically inspired networks.

Chapter 5

Conclusion

This dissertation set out to extend the hippocampal reverse replay model of reinforcement learning, originally developed by Whelan and Vasilaki, into a biologically plausible spiking neural network (SNN) framework. By replacing rate-based neurons with Poisson place cells and leaky integrate-and-fire (LIF) action cells, and by implementing the model in a grid-world environment, this project investigated how discrete spike events and their variability influence replay-driven learning.

5.1 Summary of Contributions

The main contributions of this work can be summarized as follows:

1. **Grid-world replication of the original model.** The rate-based hippocampal reverse replay model was successfully adapted from a continuous robotic setting into a discrete grid-world simulation. This provided a controlled testbed for validating key mechanisms such as replay-triggered credit assignment using intrinsic plasticity and reinforcement learning via eligibility traces.

2. **Spiking neural network implementation.** Place cells were modeled as Poisson spiking neurons with Gaussian receptive fields, and action cells as LIF neurons for navigation choices. This modification introduced temporal dynamics and biological noise while preserving the structural and functional features of the original model.

3. **Comparative evaluation.** Systematic experiments were conducted on both 10×10 and 20×20 grid environments. The smaller grid allowed one-hot spatial encoding, while the larger grid required distributed coding across place fields. Across both cases, the SNN implementation demonstrated similar learning performance to the rate-based baseline in terms of trial completion times, though it was noisier with weight convergence.

4. **Replay-driven reinforcement learning.** Both models showed the benefits of reverse

replay, reinforcing trajectories leading to reward and stabilizing learning. Visualizations confirmed that the spiking replay mechanism retained the functional role of the original supervised replay scheme [5].

These findings collectively support that hippocampal replay can be meaningfully modeled in a spiking domain without losing its reinforcement learning functionality.

5.2 Reflection on Goals

The goals set out at the start of this project were **partially achieved**. The spiking implementation was successfully validated against the original model. The introduction of Gaussian noise to discrete positions provided a degree of continuity, bridging the gap between grid-based and continuous environments. However, a full continuous-space implementation was not done.

The hyperparameter sweeps covered learning rate and eligibility trace time constants, but further exploration of intrinsic plasticity dynamics and extended network scaling was beyond the scope of the current work.

5.3 Limitations

Several limitations emerged during the project:

- **Biological complexity.** The neuron models used were relatively simple. While Poisson and LIF neurons capture core firing characteristics, they abstract away dendritic integration, adaptation, and realistic synaptic delays [7].
- **Environmental simplification.** The grid-world environment, though useful for reproducibility, does not fully capture the continuous nature of a navigation task.
- **Computational constraints.** Running spiking simulations with large networks proved resource-intensive, limiting the number of runs and scale of experiments that could be performed within the timeframe.

5.4 Future Directions

Building on the foundations laid in this dissertation, several promising directions emerge:

1. **Continuous-world navigation.** Implementing the SNN model in a physics-based, robotic simulator (e.g., MuJoCo, ROS-Gazebo) or even a pygame implemented continous world would enable closer comparison with biological experiments and the original MiRo platform.
2. **Hyperparameter and plasticity tuning.** Systematic exploration of intrinsic plasticity parameters could shed light on how excitability regulation contributes to replay stability. Incorporating more advanced forms of synaptic plasticity (e.g., voltage-dependent STDP [43]) could also increase biological realism.

3. **Environmental complexity.** Introducing obstacles, variable rewards, or dynamic goals would test the adaptability of replay mechanisms in more challenging navigation scenarios.

4. **Scalability.** Extending the model to larger environments, larger neural populations, smaller sction space, would help assess how replay scales with network size and whether noise accumulation affects long trajectories.

This dissertation demonstrates that replay-driven reinforcement learning can be effectively implemented in a spiking neural network. Despite the added noise and variability inherent to spikes, the SNN replicated the functional benefits of the original rate-based model. This finding support to the hypothesis that hippocampal replay is compatible with biologically realistic neural dynamics along with it's computationally usefulness.

By bridging the gap between rate-based abstractions and spiking implementations, this work contributes to efforts in computational neuroscience to understand how memory, replay, and reinforcement learning interact in the brain. It also talks about future applications of SNN-based replay mechanisms in robotics and neuromorphic systems, where the energy-efficient and event-driven nature of spikes may offer practical advantages.

Bibliography

[1] G. Buzsaki, "Hippocampus," *Scholarpedia*, vol. 6, no. 1, p. 1468, 2008, accessed: 2025-09-17. [Online]. Available: http://www.scholarpedia.org/article/Hippocampus#Hippocampal_place_cells_and_episode_cells

[2] A. Johnson and A. D. Redish, "Hippocampal replay contributes to within session learning in a temporal difference reinforcement learning model," *Neural Networks*, vol. 18, no. 9, pp. 1163–1171, 2005.

[3] H. F. Ólafsdóttir, D. Bush, and C. Barry, "The role of hippocampal replay in memory and planning," *Current Biology*, vol. 28, no. 1, pp. R37–R50, 2018.

[4] D. J. Foster and M. A. Wilson, "Reverse replay of behavioural sequences in hippocampal place cells during the awake state," *Nature*, vol. 440, no. 7084, pp. 680–683, 2006.

[5] M. T. Whelan, A. Jimenez-Rodriguez, T. J. Prescott, and E. Vasilaki, "A robotic model of hippocampal reverse replay for reinforcement learning," *Bioinspiration & Biomimetics*, vol. 18, no. 1, p. 015007, 2022.

[6] K. Yamazaki, V.-K. Vo-Ho, D. Bulsara, and N. Le, "Spiking neural networks and their applications: A review," *Brain sciences*, vol. 12, no. 7, p. 863, 2022.

[7] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[8] J. O'Keefe and J. Dostrovsky, "The hippocampus as a spatial map: preliminary evidence from unit activity in the freely-moving rat." *Brain research*, 1971.

[9] R. E. Ambrose, B. E. Pfeiffer, and D. J. Foster, "Reverse replay of hippocampal place cells is uniquely modulated by changing reward," *Neuron*, vol. 91, no. 5, pp. 1124–1136, 2016.

[10] A. Johnson and A. D. Redish, "Neural ensembles in ca3 transiently encode paths forward of the animal at a decision point," *Journal of Neuroscience*, vol. 27, no. 45, pp. 12 176–12 189, 2007.

[11] G. Girardeau, K. Benchenane, S. I. Wiener, G. Buzsáki, and M. B. Zugaro, "Selective suppression of hippocampal ripples impairs spatial memory," *Nature neuroscience*, vol. 12, no. 10, pp. 1222–1223, 2009.

[12] R. Cazé, M. Khamassi, L. Aubin, and B. Girard, "Hippocampal replays under the scrutiny of reinforcement learning models," *Journal of neurophysiology*, vol. 120, no. 6, pp. 2877–2896, 2018.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html

[14] E. Pignatelli, J. Ferret, M. Geist, T. Mesnard, H. van Hasselt, O. Pietquin, and L. Toni, "A survey of temporal credit assignment in deep reinforcement learning," *arXiv preprint arXiv:2312.01072*, 2023.

[15] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.

[16] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine learning*, vol. 3, no. 1, pp. 9–44, 1988.

[17] E. M. Izhikevich, "Solving the distal reward problem through linkage of stdp and dopamine signaling," *Cerebral cortex*, vol. 17, no. 10, pp. 2443–2452, 2007.

[18] M. V. Tsodyks, W. E. Skaggs, T. J. Sejnowski, and B. L. McNaughton, "Population dynamics and theta rhythm phase precession of hippocampal place cell firing: a spiking neuron model," *Hippocampus*, vol. 6, no. 3, pp. 271–280, 1996.

[19] K. Diba and G. Buzsáki, "Forward and reverse hippocampal place-cell sequences during ripples," *Nature neuroscience*, vol. 10, no. 10, pp. 1241–1242, 2007.

[20] W. D. Penny, P. Zeidman, and N. Burgess, "Forward and backward inference in spatial cognition," *PLoS computational biology*, vol. 9, no. 12, p. e1003383, 2013.

[21] K. D. Harris, J. Csicsvari, H. Hirase, G. Dragoi, and G. Buzsáki, "Organization of cell assemblies in the hippocampus," *Nature*, vol. 424, no. 6948, pp. 552–556, 2003.

[22] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *Ieee Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[23] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[24] F. Zenke and S. Ganguli, "Superspike: Supervised learning in multi-layer spiking neural networks. arxiv," *arXiv preprint arXiv:1705.11146*, vol. 10, 2017.

[25] C. Lee, A. K. Kosta, A. Z. Zhu, K. Chaney, K. Daniilidis, and K. Roy, "Spike-flownet: event-based optical flow estimation with energy-efficient hybrid neural networks," in *European conference on computer vision*. Springer, 2020, pp. 366–382.

[26] A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. homogeneous synaptic input," *Biological cybernetics*, vol. 95, no. 1, pp. 1–19, 2006.

[27] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[28] R. Moreno-Bote, "Poisson-like spiking in circuits with probabilistic synapses," *PLoS computational biology*, vol. 10, no. 7, p. e1003522, 2014.

[29] A. Amarasingham, T.-L. Chen, S. Geman, M. T. Harrison, and D. L. Sheinberg, "Spike count reliability and the poisson hypothesis," *Journal of Neuroscience*, vol. 26, no. 3, pp. 801–809, 2006.

[30] G.-q. Bi and M.-m. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type," *Journal of neuroscience*, vol. 18, no. 24, pp. 10 464–10 472, 1998.

[31] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[32] R. V. Florian, "Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity," *Neural computation*, vol. 19, no. 6, pp. 1468–1502, 2007.

[33] S. Skorheim, P. Lonjers, and M. Bazhenov, "A spiking network model of decision making employing rewarded stdp," *PloS one*, vol. 9, no. 3, p. e90821, 2014.

[34] H. Fang, Y. Zeng, and F. Zhao, "Brain inspired sequences production by spiking neural networks with reward-modulated stdp," *Frontiers in Computational Neuroscience*, vol. 15, p. 612041, 2021.

[35] S. F. Chevtchenko and T. B. Ludermir, "Combining stdp and binary networks for reinforcement learning from images and sparse rewards," *Neural Networks*, vol. 144, pp. 496–506, 2021.

[36] H. Lu, J. Liu, Y. Luo, Y. Hua, S. Qiu, and Y. Huang, "An autonomous learning mobile robot using biological reward modulate stdp," *Neurocomputing*, vol. 458, pp. 308–318, 2021.

[37] A. Juarez-Lora, V. H. Ponce-Ponce, H. Sossa, and E. Rubio-Espino, "R-stdp spiking neural network architecture for motion control on a changing friction joint robotic arm," *Frontiers in Neurorobotics*, vol. 16, p. 904017, 2022.

[38] E. Vasilaki, N. Frémaux, R. Urbanczik, W. Senn, and W. Gerstner, "Spike-based reinforcement learning in continuous state and action space: when policy gradient methods fail," *PLoS Computational Biology*, vol. 5, no. 12, p. e1000586, 2009.

[39] D. Heeger *et al.*, "Poisson model of spike generation," *Handout, University of Standford*, vol. 5, no. 1-13, p. 76, 2000.

[40] I. Sibanda. (2025, February) Advances in artificial neural networks: Exploring spiking neural models. Accessed: 2025-07-18. [Online]. Available: https://www.computer.org/publications/tech-news/trends/spiking-neural-models

[41] N. Frémaux and W. Gerstner, "Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules," *Frontiers in neural circuits*, vol. 9, p. 85, 2016.

[42] I. Sommerville, *Software Engineering*, 9th ed. Addison-Wesley, 2011.

[43] C. Clopath, L. Büsing, E. Vasilaki, and W. Gerstner, "Connectivity reflects coding: a model of voltage-based stdp with homeostasis," *Nature neuroscience*, vol. 13, no. 3, pp. 344–352, 2010.