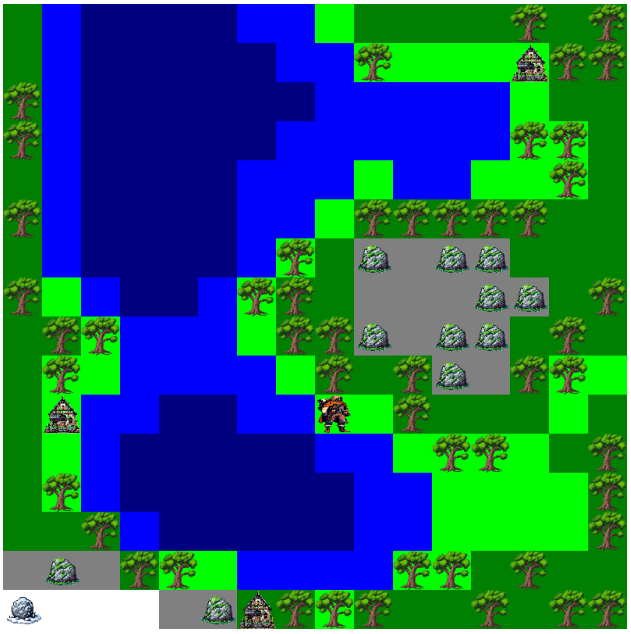
CHALMERS
UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG



Neuro-Symbolic Creation of Non-Playable Characters

An ablation study of the synergy of a Knowledge Graph in a Reinforcement Learning model.

Master's thesis in Game Design and Technology

Antonio Mangoni di S. Stefano

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Neuro-Symbolic Creation of Non-Playable Characters

An ablation study of the synergy of a Knowledge Graph in a Reinforcement Learning model.

Antonio Mangoni di S. Stefano

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Neuro-Symbolic Creation of Non-Playable Characters
An ablation study of the synergy of a Knowledge Graph in a Reinforcement Learning model.
Antonio Mangoni di S. Stefano

Supervisor: Kivanc Tatar, Data Science and AI, Computer Science and Engineering
Examiner: Morten Fjeld, Human Computer Interaction (HCI)

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Neuro-Symbolic Creation of Non-Playable Characters
An ablation study of the synergy of a Knowledge Graph in a Reinforcement Learning model.

Abstract

This thesis explores the integration of Knowledge Graphs (KG) with Reinforcement Learning (RL) to enhance the adaptability and efficiency of Non-Playable Characters (NPCs) in dynamic video game environments. By combining KG and RL in a neuro-symbolic approach, the study aims to address the lengthy and resource-intensive training processes typically required for RL agents. Using an ablation methodology, the research tests the effectiveness of various KG complexities, including partial and fully dynamic KGs, in a custom-built simulation environment employing Python, PyTorch, and Pygame. While results are pending, the study expects to demonstrate that KG-RL integration can significantly reduce training time and improve NPC adaptability. This research may offer broader implications for developing AI agents in various dynamic systems, laying foundational insights into the scalability and applicability of the KG-RL model.

Keywords: Knowledge Graphs, Reinforcement Learning, Neuro-Symbolic AI, Non-Playable Characters (NPCs), Dynamic Systems, Ablation Study, Video Game AI, Adaptability and Efficiency, Simulation Environments, AI Training Efficiency

Acknowledgements

First of all, I would like to thank my supervisor, Kıvanç Tatar, from the Data Science and AI division in the Computer Science and Engineering department at Chalmers. He was of great help in letting my ideas flourish while reining in the ever expanding scope of the project.

I would also like to thank the people at AI Sweden who offered me a place in the Master Thesis Talent Program. Being able to work at their office and participate in their culture was an invaluable experience, from being in a cohort of interesting people doing incredible projects to hearing the organisation's initiatives and work culture.

Finally, I would like to thank my family and friends, in particular my mother and sister, with whom through many long discussions I developed a clearer picture of my process and project roadmap.

Antonio Mangoni di S. Stefano, Gothenburg, November 2024

Contents

Nomenclature

CNN	Convolutional Neural Network
DRL	Deep Reinforcement Learning
GAT	Graph Attention Network
GDL	Geometric Deep Learning
GNN	Graph Neural Network
KG	Knowledge Graph
LLM	Large Language Model
NPC	Non-Playable Character
PPO	Proximal Policy Optimization
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
TSP	Travelling Salesman Problem

List of Figures

List of Tables

1

Introduction

This thesis investigates the use of Knowledge Graphs (KG) to enhance the training efficiency and reliability of Reinforcement Learning (RL) agents in video game development.

1.1 Background and Explanation of Technologies

In the rapidly evolving field of video game development, creating intelligent and adaptive Non-Player Characters (NPCs) remains a significant challenge. This thesis investigates the use of KG to enhance the training efficiency and reliability of RL agents in video game development.

RL is a type of machine learning where an agent learns to make decisions by taking actions in an environment to achieve some goals. The agent learns from the outcomes of its actions, without explicit instructions, using feedback from its own experiences to improve its performance. RL has shown promise in developing intelligent NPCs. However, RL agents often require extensive training time and resources to achieve satisfactory performance, particularly in complex game environments. Moreover, ensuring consistent and realistic behaviour of these agents remains a pressing concern.

KGs, on the other hand, are a way of storing information that enables the representation of real-world entities and their interrelations in a graph format, where nodes represent entities (e.g., objects, characters) and edges define the relationships between these entities. KGs provide a structured and interpretable framework of knowledge that can be used for a variety of applications, including enhancing the intelligence of AI agents. This graph-based structure provides a rich, interpretable framework of knowledge that can be leveraged to enhance AI systems.

Video games inherently possess rich structural information about their virtual worlds, including object relationships, spatial arrangements, and attribute hierarchies. This information is often already organized in scene graphs - data structures that manage the logical and spatial relationships of graphical elements. The natural alignment between these existing game data structures and KGs presents an opportunity to leverage this inherent organisation for NPC development. Rather than creating entirely new knowledge representations, this thesis proposes utilizing and extending these established structural frameworks to enhance NPC training and behaviour.

1.2 Research Objectives and Hypothesis

The primary question this research seeks to answer is: How can dynamically created KGs be effectively integrated with Reinforcement Learning to develop NPCs in complex game environments, and what implications does this integration have for NPC adaptability and training efficiency? Objectives include developing a framework for dynamic KG integration, evaluating its impact on NPC behaviour and learning efficiency, and assessing the approach's scalability and applicability to other fields.

The aim of this research is to explore how integrating KGs can provide a grounding mechanism for RL agents. This grounding is expected to reduce training time and improve the reliability of NPC behaviours by offering a structured framework of knowledge about the game environment and its dynamics.

I hypothesize that KGs can serve as a foundational layer of knowledge, enabling RL agents to make more informed decisions based on the relationships and attributes encoded within the graph. This integration is expected to reduce training time, enhance the consistency of NPC behaviours, and improve their adaptability to changing game environments.

The dynamic generation and updating of KGs as NPCs interact within the game pose computational and accuracy challenges. This includes determining the scope of the KG, such as whether it should encompass agent states, entities, and possibly terrain types, and ensuring it reflects the game's complexities in real-time.

Integrating KGs with RL models presents challenges in data compatibility, ensuring effective communication between the KG and the RL agent, and maintaining model performance throughout the integration process.

1.3 Motivation and Rationale

The motivation for this work is rooted in the challenges associated with the lengthy and resource-intensive training processes often required for RL agents to achieve satisfactory performance in complex game environments. Additionally, ensuring the reliability of these agents, in terms of consistent and realistic behaviour, remains a significant concern.

Integrating a KG into the RL model allows for the symbolic representation of the game world, which can be leveraged to guide the agent's learning process. This approach is anticipated to facilitate more efficient learning by providing the agent with access to a structured set of knowledge about the environment and action space, potentially reducing the amount of trial-and-error learning needed to develop effective behaviours.

This research explores the integration of KGs with RL within video game environments, serving as a preliminary exploration of neuro-symbolic models' potential applications in more complex real-world scenarios, such as digital twins for warehouses or urban planning. The goal is to develop agents capable of generalising and dynamically adapting to new or evolving environments. By focusing on creating NPCs that can learn and make decisions autonomously, this project aims to improve gaming experiences by providing more realistic and adaptable NPC behaviours.

1.4 Methodology

To address the research question—*What are the effects of integrating dynamically created Knowledge Graphs with model-based Reinforcement Learning on the adaptability and training efficiency of NPCs in complex game environments?*—a systematic approach was designed, involving the following steps:

A custom game environment was created, featuring diverse terrains (e.g., water, plains, and mountains) and various entities (e.g., trees and rocks). Outposts were established as key navigation points for the agent. A dynamic Knowledge Graph (KG) was integrated into the environment to represent discovered terrain, entities, and their relationships as the agent explored.

The RL agent was implemented using the Proximal Policy Optimization (PPO) algorithm, chosen for its robustness in complex environments. A curriculum learning strategy was employed, where the agent was initially presented with simple tasks, and the complexity of tasks increased as the agent's performance improved. This ensured gradual and effective learning.

During the agent's exploration, the environment's state, including terrain and entity relationships, was continuously updated in the Knowledge Graph. The graph neural network processed this KG to provide structured information that influenced the agent's decision-making process.

- **Training Phase:** The RL agent was trained across multiple scenarios, with variations in terrain and entity configurations, to simulate diverse game states.
- **Evaluation Metrics:** The agent's performance was measured by its ability to navigate between outposts, manage resources, and adapt to changes in the environment. The specific metric measured was the energy required to complete the travelling salesman problem.
- **Adaptability:** To test adaptability, the game environment was created randomly and sorted by route energy requirement, so it became progressively harder but also changed frequently.
- **Ablation Study:** An ablation study was conducted by varying the size of the Knowledge Graph. Different levels of graph detail and complexity were tested to observe the impact on the RL agent's performance. The size of the KG was the only variable, ensuring that all other parameters, including model size, remained consistent.

Data were collected from the training sessions, including performance metrics, decision logs, and the evolution of the Knowledge Graph over time. Comparative analysis was performed across different KG sizes to determine their impact on training efficiency and NPC behaviour reliability.

The results from the various tests and scenarios were synthesized to evaluate the impact of different Knowledge Graph sizes on the RL agent's performance. Statistical methods were employed to determine the significance of the observed effects, ensuring that the conclusions drawn were robust and reliable.

1.5 Contribution

Successful integration could lead to NPCs that autonomously learn and adapt, reflecting a deep understanding of the game world. This development could improve gameplay immersion and interaction, making NPCs more responsive to player actions and environmental changes.

By incorporating KGs, there is potential to reduce the time and resources required for training RL agents, thanks to a structured knowledge base that accelerates the learning process.

The findings could have implications beyond gaming, including in simulations and educational tools, where adaptive AI agents could significantly enhance engagement and learning outcomes.

2

Theory

2.1 Foundations of Reinforcement Learning in Game Environments

RL has emerged as a powerful paradigm in artificial intelligence, particularly in the domain of video game development. At its core, RL is a computational approach to learning from interaction, where an agent learns to make decisions by taking actions in an environment to maximize a cumulative reward signal [1]. This framework aligns naturally with the interactive nature of video games, making RL an ideal candidate for developing intelligent NPCs.

In the context of video games, RL agents (NPCs) interact with the game environment, receiving observations about the game state and selecting actions to perform. The game provides feedback in the form of rewards, which the agent uses to improve its decision-making policy over time. This process allows NPCs to adapt their behaviour based on the specific challenges and dynamics of the game world, potentially leading to more engaging and realistic interactions for players.

2.1.1 Deep Reinforcement Learning (DRL) and Neural Networks

Deep Reinforcement Learning (DRL) integrates the decision-making capabilities of RL with the representational power of deep neural networks. [2] This synergy enables agents to learn complex behaviours and strategies directly from high-dimensional inputs, such as raw pixel data in video games. The foundational principles of DRL revolve around the interaction between an agent and its environment, where the agent learns to make sequences of decisions by maximizing cumulative rewards.

Deep neural networks serve as function approximations in DRL, enabling the estimation of value functions, policies, or both. Notable architectures include: [2]

1. **Deep Q-Networks (DQN):** Combines Q-learning with deep neural networks to approximate the Q-value function, allowing agents to handle high-dimensional state spaces like those in Atari games.
2. **Policy Gradient Methods:** Directly parameterize the policy and optimize it using gradient ascent, facilitating continuous action spaces and stochastic policies.
3. **Actor-Critic Models:** Merge value-based and policy-based approaches by having separate structures (actor and critic) that respectively handle policy optimization and value estimation.

2.1.2 Exploration vs. Exploitation

A pivotal challenge in RL is balancing exploration (discovering new actions to improve knowledge) and exploitation (leveraging known actions to maximize rewards). [2] Effective exploration strategies are crucial, especially in environments with sparse or delayed rewards, as seen in complex video games like Montezuma's Revenge. [3] Techniques such as E-greedy policies, entropy regularization, and intrinsic motivation are employed to encourage adequate exploration.

2.1.3 Sample Efficiency

DRL algorithms often require vast amounts of data to learn effective policies, which can be computationally expensive and time-consuming. Enhancing sample efficiency is thus a critical area of research. Approaches to address this include:

1. **Experience Replay:** Combines Q-learning with deep neural networks to approximate the Q-value function, allowing agents to handle high-dimensional state spaces like those in Atari games.
2. **Hierarchical Reinforcement Learning (HRL):** Decomposing tasks into sub-tasks to simplify the learning process and enable the reuse of learned policies.
3. **Transfer Learning:** Leveraging knowledge from previously learned tasks to accelerate learning in new, related tasks.

2.1.4 Model-Based Reinforcement Learning

One particularly promising approach within RL for NPC development is model-based reinforcement learning. Unlike model-free methods that learn directly from experience, model-based RL involves learning a model of the environment's dynamics. This model allows the agent to predict the outcomes of its actions, enabling more efficient learning and strategic planning [1].

For NPC development, this approach offers several advantages:

1. **Sample Efficiency:** By learning a model of the environment, NPCs can learn from fewer interactions, reducing the computational resources required for training.
2. **Strategic Planning:** With a learned model, NPCs can simulate different courses of action and choose the most promising one, leading to more sophisticated behaviour.
3. **Adaptability:** As the game environment changes (e.g., through updates or player actions), the learned model can be updated, allowing NPCs to adapt their behaviour accordingly.
4. **Interpretability:** The learned model provides insight into how the NPC understands the game world, potentially making it easier for developers to debug and refine NPC behaviour.

2.2 Knowledge Graphs and Their Role in AI

2.2.1 Structure and Components of Knowledge Graphs

A Knowledge Graph consists of three primary components:

1. **Nodes (Entities):** These represent discrete objects, concepts, or ideas within the domain of interest.
2. **Edges (Relationships):** These connect nodes and describe how entities are related to each other.
3. **Properties:** These are attributes that provide additional information about nodes or edges.

This structure allows KGs to capture complex, multi-relational data in a flexible and extensible manner. For instance, in a game world, nodes might represent characters, items, and locations, while edges could denote actions like "uses", "contains", or "is located in".

2.2.2 Benefits of KGs in Representing Complex Game Worlds

Knowledge Graphs offer several advantages for representing the intricate dynamics of game environments:

- **Semantic Relationships:** KGs can capture nuanced relationships between game elements, allowing for rich, context-aware representations of the game world.
- **Scalability:** As game worlds expand, KGs can easily accommodate new entities and relationships without major restructuring.
- **Inference Capabilities:** The graph structure allows for logical inference, enabling NPCs to deduce new information based on existing knowledge.
- **Multimodal Integration:** KGs can integrate diverse types of information (textual, visual, numerical) into a unified representation.
- **Dynamic Updating:** KGs can be updated in real-time as the game state changes, providing a continuously accurate representation of the world.

These features make KGs particularly suited for representing the complex, evolving nature of modern game environments, providing a rich knowledge base for AI agents to reason over and interact with.

2.2.3 Knowledge Graph Embedding Techniques

To leverage KGs in machine learning models, including reinforcement learning agents, it's necessary to transform the discrete graph structure into continuous vector representations. This process is known as Knowledge Graph embedding. [4] Several techniques have been developed for KG embedding:

- **TransE:** This method represents relationships as translations in the embedding space.
- **ComplEx:** Uses complex-valued embeddings to model asymmetric relations.
- **RotatE:** Represents relations as rotations in complex vector space.

- **Graph Neural Networks (GNNs):** These models learn to aggregate information from a node's neighborhood, capturing both local and global graph structure.

Recent work has explored the integration of KG embeddings with reinforcement learning to provide agents with structured prior knowledge [5]. This approach shows promise in accelerating learning and improving generalization, particularly in complex environments like those found in modern video games. The choice of embedding technique can significantly impact the performance of downstream tasks. For instance, in the context of NPC development, an effective embedding should capture the hierarchical nature of game world elements (e.g., items within locations) and the diverse types of relationships between entities (e.g., causal, spatial, temporal).

As I progress through this thesis, I will explore how these KG representations can be integrated with reinforcement learning models to create more adaptive and intelligent NPCs, capable of reasoning over complex game world knowledge and adapting to dynamic environments.

2.3 Integration of Reinforcement Learning and Knowledge Graphs

The integration of RL and KGs represents a promising approach to enhance the capabilities of intelligent agents, particularly in complex and dynamic environments such as video games. This combination aims to leverage the strengths of both paradigms: the adaptive learning capabilities of RL and the structured knowledge representation of KGs.

The theoretical basis for this integration lies in the potential for KGs to provide a rich, structured prior knowledge to RL agents. In traditional RL, agents often start with minimal prior knowledge about their environment, leading to long learning periods and potential difficulties in generalization. By incorporating KGs, I can provide agents with a foundational understanding of the world they operate in, potentially accelerating learning and improving decision-making [5]. This integration offers several potential advantages:

- **Improved Sample Efficiency:** By leveraging the structured knowledge in KGs, RL agents can potentially learn more efficiently, requiring fewer interactions with the environment to achieve competent behaviour.
- **Enhanced Generalization:** The relational structure of KGs can help agents generalize their learning to new situations more effectively, by understanding the underlying relationships between entities and actions.
- **Interpretable Decision-Making:** The use of KGs can provide more transparency in the agent's decision-making process, as actions can be traced back to specific relationships or properties in the knowledge graph.
- **Dynamic Knowledge Incorporation:** As the game world evolves, new knowledge can be added to the KG, allowing the RL agent to adapt to changes without requiring complete retraining.

The work on graph embedding priors by Parikh et al. [5] provides a foundation for understanding how KG embeddings can be effectively integrated into RL frame-

works. Their approach demonstrates how graph-structured knowledge can be used to initialize RL agents with relevant prior information, leading to faster learning and better performance in multi-task scenarios.

Recent work by Zhang et al. [6] on robust knowledge graph embedding via multi-task reinforcement learning further illustrates the potential of this integration. Their approach not only improves the quality of KG embeddings but also demonstrates how RL can be used to enhance the knowledge representation itself, creating a symbiotic relationship between RL and KGs.

In the context of NPC development, this integration could lead to more sophisticated and adaptable characters. For example, an NPC could use the KG to understand complex relationships between game elements (e.g., item crafting recipes, character relationships, quest dependencies) while using RL to learn optimal strategies for navigating and interacting with the game world. This approach aligns with the concept of Feudal Networks for hierarchical reinforcement learning [7], where higher-level knowledge (potentially stored in a KG) can guide lower-level decision-making processes.

However, challenges remain in effectively combining these two approaches. These include determining the most effective ways to incorporate KG information into the RL process, handling potential inconsistencies between the KG and the actual game state, and scaling the integrated approach to very large and complex game worlds. The Navigation Turing Test proposed by Devlin et al. [8] could provide a framework for evaluating the effectiveness of this integration in producing human-like navigation behaviours in NPCs.

As I delve deeper into this thesis, I will explore specific techniques for integrating KGs with RL models, examining their impact on NPC behaviour and adaptability in dynamic game environments. This integration represents a step towards creating more intelligent, context-aware AI agents capable of nuanced decision-making in complex, evolving worlds, potentially bridging the gap between symbolic AI and sub-symbolic learning methods in game AI development.

2.4 Fast and Slow Thinking in AI Agents

2.4.1 Dual-Process Theory in Cognitive Science

The concept of dual-process theory in cognitive science, popularized by Daniel Kahneman's work "Thinking, Fast and Slow" [9], proposes that human cognition operates through two distinct systems:

- **System 1:** Fast, automatic, and intuitive thinking
- **System 2:** Slow, deliberate, and analytical thinking

This theory has its roots in earlier work by cognitive psychologists like Jonathan Evans [10], who identified heuristic and analytic processes in reasoning. The dual-process framework provides a powerful model for understanding human decision-making and has inspired new approaches in artificial intelligence.

2.4.2 Application to AI: Deep Learning and Tree Search

The application of dual-process theory to AI is exemplified in the work "Thinking Fast and Slow with Deep Learning and Tree Search" by Anthony et al. [11]. This research draws parallels between the dual-process theory and AI methodologies:

- **Fast Thinking (System 1):** Represented by deep learning models, which can quickly process inputs and generate responses based on learned patterns.
- **Slow Thinking (System 2):** Implemented through tree search algorithms, which perform deliberate, step-by-step planning and evaluation of possible action sequences.

The integration of these two approaches allows AI systems to balance rapid, intuitive responses with more considered, analytical decision-making. This combination has shown remarkable success in complex domains, as demonstrated by systems such as AlphaGo [12].

2.4.3 KG Integration and Dual-Process Thinking

The integration of KGs with RL can be viewed through the lens of dual-process theory, offering a novel approach to implementing "fast" and "slow" thinking in AI agents:

- **KGs as System 2:** KGs can serve as a basis for "slow" thinking, providing a structured representation of knowledge that allows for logical inference and deliberate reasoning. This aligns with the System 2 characteristics of analytical and rule-based thinking.
- **RL as System 1:** The learned policies in RL can be seen as a form of "fast" thinking, allowing agents to make rapid decisions based on learned patterns and experiences, similar to System 1's intuitive and automatic processes.

The combination of KGs and RL in this context offers several advantages:

1. **Grounded Intuition:** The fast, intuitive decisions made by the RL component can be grounded in the structured knowledge provided by the KG, potentially leading to more reliable and contextually appropriate actions.
2. **Guided Exploration:** The KG can guide the RL agent's exploration, focusing on relevant areas of the state space and potentially accelerating learning.
3. **Explainable Decision-Making:** The KG component can provide a basis for explaining the agent's decisions, addressing one of the key challenges in modern AI systems.

This integration of KGs with RL for implementing dual-process thinking in AI agents represents a step towards more human-like reasoning in artificial systems. It offers a promising approach for developing NPCs that can exhibit both rapid, intuitive behaviours and thoughtful, context-aware decision-making in complex game environments.

As I progress through this thesis, I will explore how this dual-process approach, facilitated by the integration of KGs and RL, can be implemented and evaluated in the context of NPC development for dynamic game worlds.

2.5 Geometric Deep Learning and Graph Neural Networks

Geometric Deep Learning (GDL) is an emerging field that extends deep learning techniques to non-Euclidean domains such as graphs and manifolds. As outlined in the seminal proto-book by Bronstein et al. [13], GDL provides a unified framework for designing neural networks that can operate on various data structures, including graphs, point clouds, and meshes. This approach is particularly relevant for domains where data naturally exists in non-grid formats, such as social networks, molecular structures, and, pertinent to this study, game environments.

2.5.1 Importance of Graph Neural Networks

GNNs are a crucial component of Geometric Deep Learning, specifically designed to process graph-structured data. Their importance in the context of this research on integrating KGs with RL for NPC development cannot be overstated. GNNs offer several key advantages:

- **Relational Inductive Bias:** GNNs inherently capture the relational structure of data, making them ideal for processing KGs where relationships between entities are fundamental.
- **Scalability:** Unlike traditional graph algorithms, GNNs can efficiently process large-scale graphs, which is crucial for representing complex game worlds.
- **Learnable Representations:** GNNs can learn to embed nodes, edges, and sub-graphs into continuous vector spaces, facilitating downstream machine learning tasks.
- **Flexibility:** They can handle heterogeneous graphs with different types of nodes and edges, aligning well with the diverse entities and relationships in game environments.

2.5.2 Application to NPC Development

In the context of this research, GNNs serve as a bridge between the structured knowledge represented in KGs and the learning processes of RL agents. They allow us to:

1. **Process Dynamic KGs:** As the game world evolves and the KG is updated, GNNs can efficiently recompute embeddings, ensuring that the RL agent always has access to up-to-date information.
2. **Extract Relevant Features:** GNNs can learn to extract features from the KG that are most relevant for the RL agent's current task or state.
3. **Enable Multi-relational Reasoning:** By processing the complex relationships in game environments, GNNs can enable NPCs to reason about indirect connections and consequences of actions.

The Geometric Deep Learning framework, as presented by Bronstein et al. [13], provides a theoretical foundation for understanding how these graph-based models can be designed and optimized. It offers insights into the types of invariances and

equivariances that are desirable in different graph learning scenarios, which is crucial for developing robust and generalizable NPC behavithiss.

As I progress in this thesis, I will explore specific GNN architectures and how they can be integrated into this RL framework for NPC development. I will consider how the principles of Geometric Deep Learning can guide the design of neural network components that effectively leverage the structured knowledge in this game-world KGs, potentially leading to more intelligent and adaptive NPCs.

2.6 Dynamic Environments and NPC Adaptability

2.6.1 Challenges in Creating NPCs for Dynamic Game Worlds

The landscape of video game development has significantly evolved, with a growing trend towards games that feature constantly updating, dynamic worlds. This shift presents unique challenges in the creation of NPCs that can adapt to these ever-changing environments. Traditional methods of NPC design, which often rely on pre-scripted behaviours and static decision trees, are increasingly inadequate in these complex, evolving game worlds.

Key challenges include:

- **Continuous Learning:** NPCs must be capable of learning and adapting their behaviours as the game world changes, rather than relying solely on pre-programmed responses.
- **Contextual Understanding:** As new elements are introduced to the game, NPCs need to understand and interact with them in contextually appropriate ways.
- **Emergent behaviour Handling:** In dynamic environments, the interaction of multiple game systems can lead to emergent scenarios that NPCs must be able to navigate.
- **Performance Optimization:** Adaptive NPCs must be computationally efficient to maintain game performance, especially in large-scale, multiplayer environments.

2.6.2 The Role of Systemic Games

The concept of systemic games, as discussed in the article "Systemic Games" [14], plays a crucial role in driving the need for more adaptive NPCs. Systemic games are characterized by their complex, interacting systems that create emergent gameplay experiences. These games move away from scripted events towards more dynamic, player-driven narratives and interactions.

In systemic games:

- Game elements interact with each other in often unpredictable ways.
- Players have greater freedom to approach objectives through creative problem-solving.
- The game world responds dynamically to player actions and choices.

This systemic approach creates a more immersive and reactive game world, but it also significantly increases the complexity of NPC design.
NPCs in these environments must be able to:

- Understand and react to the current state of multiple, interacting game systems.
- Adapt their behaviours based on player actions and changes in the game world.
- Exhibit believable and contextually appropriate behaviours in a wide range of scenarios, including those not explicitly anticipated by game designers.

2.6.3 The Need for Adaptive NPCs

The rise of systemic games and constantly updating game worlds necessitates a new approach to NPC development. Adaptive NPCs, capable of learning and evolving their behaviours, are essential for maintaining the integrity and immersion of these complex game environments. This need aligns with the goals of this research in integrating KGs with RL.
By combining KGs and RL, I aim to create NPCs that can:

- Maintain a structured understanding of the game world (via KGs) that can be updated as the environment changes.
- Learn new behaviours and strategies (via RL) in response to changes in the game world or player actions.
- Exhibit more human-like decision-making processes, balancing "fast" intuitive responses with "slow" deliberative reasoning, as discussed in this exploration of dual-process theory.

This approach holds the potential to create NPCs that can truly adapt to the dynamic nature of modern video games, enhancing player experience and pushing the boundaries of what's possible in game AI. As I proceed, I will explore specific techniques and implementations that address these challenges, aiming to develop NPCs that can thrive in the complex, evolving worlds of systemic games.

2.7 Theoretical Implications and Potential Contributions

2.7.1 Beyond Gaming: Wider Applications of KG-RL Integration

While this research focuses on NPC development for video games, the integration of KGs with RL has implications that extend far beyond the gaming industry. This approach can potentially be applied to various domains where intelligent agents need to operate in complex, dynamic environments:

- **Robotics:** As demonstrated in military training simulations [15], graph-based approaches can enhance multi-agent RL systems. this KG-RL integration could improve robots' ability to understand and interact with their environment, particularly in collaborative or human-robot interaction scenarios.

- **Smart Cities:** The combination of structured knowledge and adaptive learning could be applied to optimize traffic management, energy distribution, or emergency response systems in urban environments.
- **Financial Systems:** In the complex world of finance, this approach could help develop more sophisticated trading algorithms that can adapt to changing market conditions while leveraging structured knowledge about economic principles and market dynamics.

2.7.2 Improved Explainability in AI Decision-Making

One of the significant potential contributions of this research is in the realm of explainable AI. The integration of KGs with RL offers a pathway to more transparent and interpretable AI systems:

- **Traceable Decision Paths:** By grounding RL actions in the structured knowledge of KGs, I can potentially trace the reasoning behind an agent's decisions, similar to the way human experts can explain their thought processes.
- **Symbolic-Subsymbolic Bridge:** this approach bridges the gap between symbolic AI (represented by KGs) and subsymbolic AI (represented by neural network-based RL), potentially offering a more complete and human-understandable model of decision-making.
- **Contextual Explanations:** The rich, relational information in KGs can provide context for an agent's actions, allowing for more nuanced and situation-aware explanations of behaviour.

This improved explainability aligns with growing demands for transparency in AI systems, particularly in sensitive applications like autonomous vehicles or medical diagnosis systems.

2.7.3 Broader Implications for Adaptive AI in Complex Systems

The development of adaptive AI agents capable of operating in complex, dynamic systems has broad implications across various fields:

- **Cognitive Science:** this work on integrating "fast" (RL) and "slow" (KG-based) thinking in AI agents contributes to ongoing research in cognitive architectures and dual-process theories of cognition [9], [10].
- **Complex Systems Management:** The ability to combine structured knowledge with adaptive learning could be valuable in managing complex systems like ecosystems, climate models, or large-scale industrial processes.
- **Lifelong Learning AI:** this approach of continuously updating KGs and adapting RL policies contributes to the development of AI systems capable of lifelong learning, a key challenge in artificial general intelligence research.

2.7.4 Towards More Human-Like AI Navigation

The integration of KGs with RL also has implications for developing more human-like navigation behaviours in AI agents. This aligns with research on the Navigation

Turing Test (NTT) [8], which aims to evaluate human-like navigation in AI systems. this approach could contribute to:

- Developing AI agents that can navigate complex environments in ways that appear more natural and intuitive to human observers.
- Creating NPCs that can explain their navigation choices in human-understandable terms, leveraging the semantic information in the KG.
- Advancing the field of cognitive mapping in AI, by providing a framework for agents to build and utilize mental models of their environment.

In conclusion, while my research is grounded in the specific context of NPC development for video games, its theoretical implications and potential contributions span a wide range of fields. By advancing the integration of structured knowledge with adaptive learning techniques, I aim to contribute to the development of more intelligent, explainable, and human-like AI systems capable of operating in complex, dynamic environments across various domains.

3

Related Work

3.1 LLM-KG Integration

The integration of KGs with various AI models has been a focus of research in recent years, particularly in enhancing decision-making processes in complex environments. Traditionally, much of this work has centred around combining KGs with RL models [16], leveraging the structured knowledge within KGs to improve the adaptability and contextual understanding of agents in dynamic systems. However, with the advent and rapid advancement of Large Language Models (LLMs), new opportunities have emerged for integrating these models with KGs to address the limitations inherent in each technology.

Pan et al. [17] provide a comprehensive roadmap for integrating LLMs with KGs, proposing three distinct frameworks that illustrate the potential synergies between these two technologies. These frameworks are:

This approach incorporates KGs during the pre-training and inference phases of LLMs, aiming to enhance the understanding of knowledge learned by LLMs. By integrating structured knowledge into LLMs, this framework seeks to improve the models' factual accuracy and interpretability, which are often areas of concern in traditional LLMs.

In this framework, LLMs are leveraged for various KG-related tasks, such as embedding, completion, construction, graph-to-text generation, and question answering. This approach uses the generalization abilities of LLMs to augment the structure and content of KGs, thereby improving their scalability and adaptability to new information.

In this approach, LLMs and KGs have equal roles and work together in a bidirectional manner, where both data and knowledge drive reasoning processes. This synergy allows for a more robust integration where the LLM can utilize structured knowledge from the KG in real-time, while the KG benefits from the LLM's ability to process and generate natural language content.

In the context of this thesis, the focus aligns with the third framework—Synergized LLMs + KGs. Here, the LLM takes on the role traditionally occupied by an RL model, processing natural language inputs and generating context-aware responses. Simultaneously, the KG serves as an additional input stream, providing structured, factual knowledge that enhances the LLM's decision-making capabilities. This integration allows the agent to engage in bidirectional reasoning, where the LLM and KG mutually reinforce each other, leading to more adaptable and intelligent behaviour in dynamic game environments.

This approach contrasts with KG-enhanced LLMs, which primarily focus on improving LLMs by integrating external knowledge during specific stages, and LLM-augmented KGs, which use LLMs to enrich the KG itself. While these frameworks provide valuable insights and have shown effectiveness in various applications, the synergized approach adopted in this thesis aims for a deeper integration, where the interaction between the LLM and KG is maximized to enhance the overall system's reasoning and adaptability.

The adoption of the Synergized LLMs + KGs framework in this thesis is particularly significant for the development of NPCs in video games. By leveraging both the natural language processing capabilities of LLMs and the structured knowledge provided by KGs, NPCs can achieve a higher level of contextual awareness and adaptability. This dual-input system enables NPCs to respond to player actions and environmental changes with greater sophistication, improving the overall gaming experience.

3.2 Voyager: An LLM-Powered Embodied Lifelong Learning Agent

The integration of LLMs into AI systems has catalysed significant advancements in developing agents that can autonomously interact with and learn from dynamic environments. A prime example of this is Voyager, an LLM-powered embodied lifelong learning agent. Voyager operates within the open-ended environment of Minecraft, where it independently explores, acquires diverse skills, and makes novel discoveries, all without human intervention. [18]

Voyager introduces three key components that bear conceptual similarities to the approaches utilized in this thesis for developing adaptive NPCs within dynamically sorted game worlds:

3.2.1 Automatic Curriculum

Voyager employs an automatic curriculum designed to maximize exploration by setting goals that dynamically adjust based on the agent's current skill set and environmental context. This concept is analogous to the dynamic sorting of game worlds by difficulty levels in this thesis. Just as Voyager adapts its exploration strategy based on its proficiency and environment, the NPCs in this research navigate game worlds that are dynamically sorted by difficulty. This sorting mechanism acts as a curriculum, guiding the NPCs to progressively engage with more challenging scenarios as they develop their skills, ensuring their learning remains structured and adaptive.

3.2.2 Skill Library

Voyager maintains a continually expanding repository of executable code that encapsulates complex behaviours, which it can retrieve and recombine to address new tasks. This concept aligns with the goal of this thesis to develop a skill library

embedded within a GNN trained on a KG. In this framework, the skills acquired by NPCs are stored not just as isolated actions but as embedded representations within the GNN. These embeddings enable NPCs to retrieve and apply relevant skills when facing new challenges, much like how Voyager composes simpler programs to execute complex tasks. Thus, the skill library becomes essential for ensuring that NPCs can generalize their learned behaviours across various game scenarios.

3.2.3 Iterative Prompting Mechanism

Voyager employs an iterative prompting mechanism that interacts with GPT-4 to generate and refine executable programs based on environmental feedback. While Voyager's specific prompting and in-context learning mechanism is tailored for code generation, the underlying principle resonates with the broader objective of this thesis: leveraging LLMs for real-time decision-making in NPCs. By integrating feedback from the environment, NPCs can refine their strategies and adapt their actions, akin to how Voyager refines its programs based on execution errors and task verification.

3.2.4 Application and Implications in This Thesis

Voyager's approach to lifelong learning through an automatic curriculum and a dynamic skill library directly informs the methodologies adopted in this thesis. The dynamic sorting of game worlds by difficulty serves as a curriculum that systematically escalates the challenges faced by NPCs, ensuring their learning path is structured and progressive. This parallels Voyager's curriculum, which is designed to maximize exploration and skill acquisition.

Moreover, embedding skills within a GNN trained on a KG provides a robust framework for NPC adaptability. By leveraging the structured knowledge within the KG, NPCs should access and apply skills that are contextually relevant, enabling them to perform complex actions with greater efficiency and accuracy. This concept aligns with Voyager's skill library, which facilitates the composition of new behaviours by retrieving and recombining previously learned skills.

3.3 Crafter: A Benchmark for Evaluating Agent Capabilities in Open World Survival Games

The Crafter benchmark, introduced by Hafner et al. (2022), serves as a significant milestone in evaluating the generalization, exploration, and adaptability of RL agents within complex, procedurally generated environments. Crafter is an open-world survival game that challenges agents to accomplish a diverse set of tasks, such as gathering resources, crafting tools, and defending against enemies. [19] This benchmark is particularly relevant to the development of intelligent NPCs in video games, as it encapsulates many of the challenges that these characters face in dynamic and unpredictable environments.

3.3.1 Relevance to NPC Development

The design of Crafter aligns closely with the goals of this thesis, which focuses on enhancing the training efficiency and reliability of NPCs through the integration of KGs with RL. Crafter's emphasis on procedural generation, where each game episode presents a unique and unpredictable world, mirrors the dynamic environments that NPCs must navigate in modern video games. The benchmark's structure—requiring agents to generalize across different scenarios and perform complex, multi-step tasks—highlights the limitations of current RL methods and underscores the potential benefits of incorporating structured knowledge through KGs.

3.3.2 Integrating Knowledge Graphs for Enhanced NPC Behaviour

While Crafter evaluates RL agents' abilities to perform in varied and challenging environments, integrating KGs into the RL framework can further enhance these capabilities. In the context of this thesis, KGs provide a structured representation of the game world, allowing NPCs to reason about the relationships between different entities (e.g., resources, tools, enemies) and make more informed decisions. By leveraging the semantic richness of KGs, NPCs can achieve better generalization and adaptability, as they can infer new knowledge from existing data and apply it to novel situations—a critical aspect of succeeding in the Crafter environment.

For example, an NPC equipped with a KG could use its knowledge of resource dependencies to optimize crafting strategies or anticipate potential threats based on the proximity of enemies and available shelters. This capability would not only reduce the trial-and-error learning typically required in complex environments but also enable more sophisticated behaviours that align with human-like reasoning and planning.

3.3.3 Experimental Insights and Potential Applications

The baseline experiments conducted in the Crafter environment reveal that even advanced RL methods like DreamerV2 and PPO struggle to match human performance, achieving only a fraction of the success rate observed in expert human players. This gap between human and agent performance suggests that additional mechanisms, such as KGs, could play a pivotal role in bridging this divide. By integrating KGs into the learning process, NPCs could access a richer set of contextual information, improving both their decision-making process and their ability to adapt to the dynamic challenges presented in environments like Crafter.

Moreover, the Crafter benchmark provides a valuable testing ground for the methodologies proposed in this thesis. By applying KGs within the Crafter framework, future research could explore how these structured knowledge representations impact the efficiency and effectiveness of RL agents, particularly in environments that demand a high degree of generalization and long-term reasoning.

4
Methods

4.1 Introduction to Methodology

This research explores the integration of KGs with Reinforcement Learning (RL) to enhance the decision-making capabilities of Non-Player Characters (NPCs) in video games. The methodology is designed to investigate how structured knowledge representation can be combined with learning-based approaches in complex, dynamic game environments.

4.1.1 Research Approach Overview

A custom game environment was developed as a test-bed for the NPC agents, where the primary task is a Travelling Salesman Problem (TSP) . The agent must navigate through various points (outposts) in the game world while minimizing energy consumption, which is influenced by terrain and other dynamic environmental factors.

Within this environment, an RL framework that incorporates a dynamically updated KG was implemented. Additionally, the methodology includes an ablation study, varying the size of the KG observation used as input to the agent's graph processor unit to evaluate how different levels of knowledge impact decision-making efficiency and adaptability. [20]

To support reproducibility and future research in this direction, all code and implementation details have been made publicly available [21].

The key components of the methodology include:

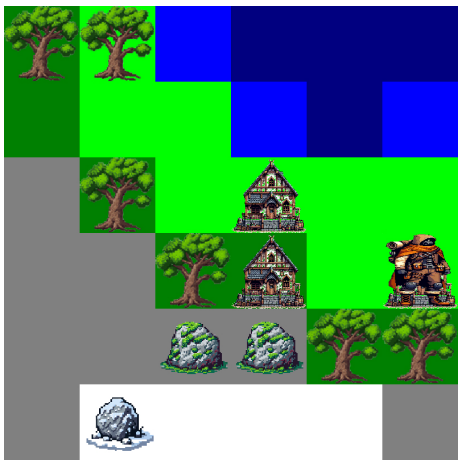


Figure 4.1: Game world example

- A procedurally generated game world
- A custom RL agent model combining visual processing with graph-based reasoning
- A dynamic KG that expands with agent exploration
- A curriculum learning approach for gradually increasing task difficulty
- An ablation study to evaluate the effects of varying KG observation sizes on agent performance
- An evaluation framework including benchmarking and performance metrics

4.1.2 Rationale for Methodological Choices

The integration of KGs with RL was chosen to address specific challenges in NPC development:

- **Sample Efficiency:** The incorporation of structured knowledge aims to reduce the required amount of trial-and-error learning.
- **Generalization:** KGs are used to enable reasoning about entity relationships, potentially improving behaviour generalization.
- **Interpretability:** The KG component is intended to provide insight into agent decision-making processes.
- **Adaptability:** The dynamic KG approach allows for continuous knowledge base updates.
- **Scalability:** GNNs are employed for efficient processing of larger, more complex game worlds.

4.2 Game World Environment

The system architecture shown in Figure 4.2 illustrates how the Game World Environment integrates with other components. At its core, the Environment Components (including Heightmap Generator, Environment, Entities, and Terrains) are created by the GameManager and form the foundation of the Game World. This Game World maintains both the physical environment and its Knowledge Graph representation, which are then processed by the Model Components to enable agent learning and decision-making.

4.2.1 Introduction to the Game World Environment

The custom game environment, as seen in 4.3, was developed for this research and acts as a controlled test bed to evaluate the integration of KG and reinforcement learning (RL) in the development of a non-player character (NPC). This environment was specifically designed to simulate complex and dynamic scenarios that challenge the decision-making abilities of NPCs, allowing for the systematic exploration of their behaviour and adaptability.

The environment offers several key benefits in the context of this study:

- It provides reproducible scenarios for controlled testing of NPC performance.
- The inclusion of diverse, procedurally generated challenges encourages NPC adaptability.

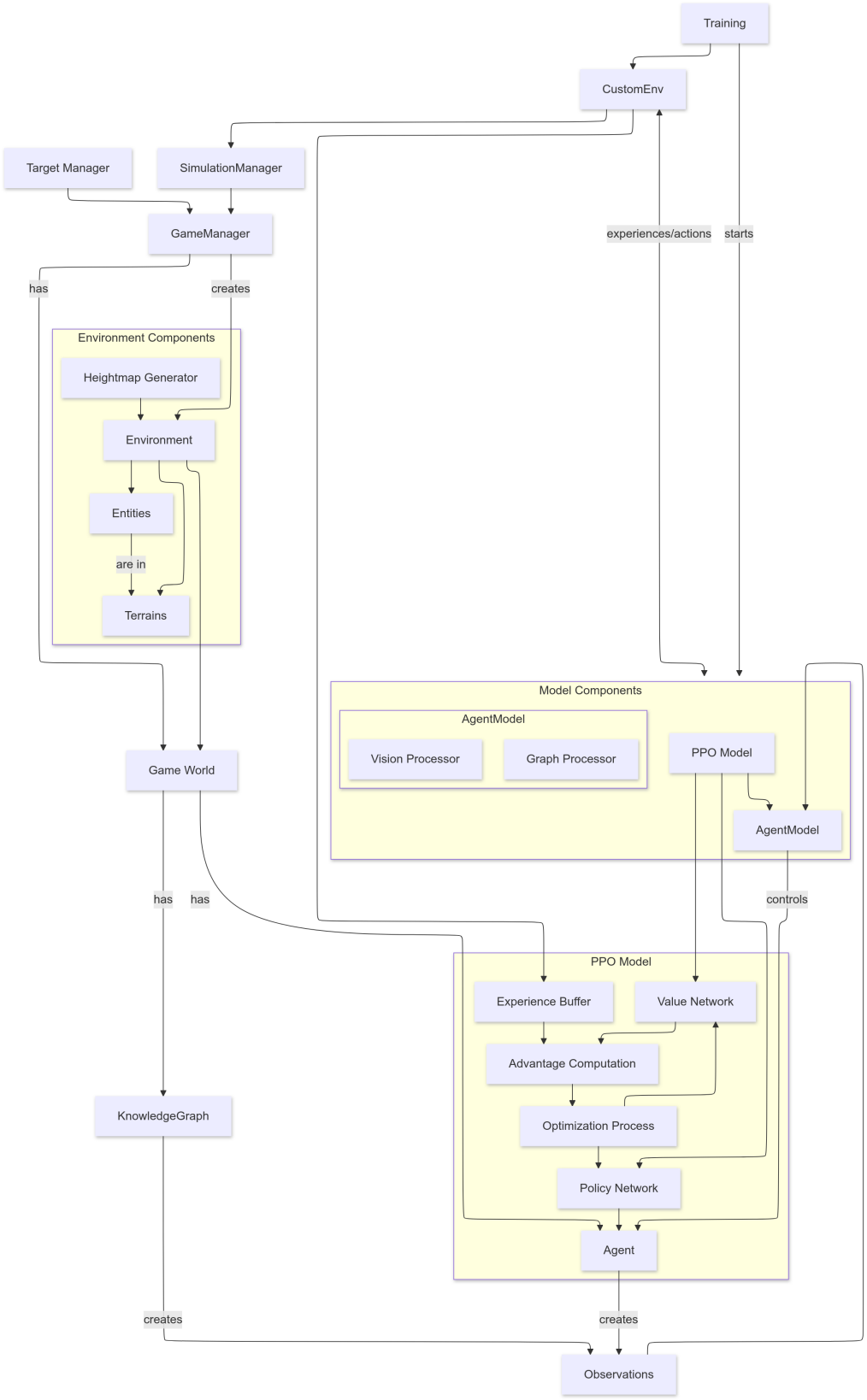


Figure 4.2: Main System Flow

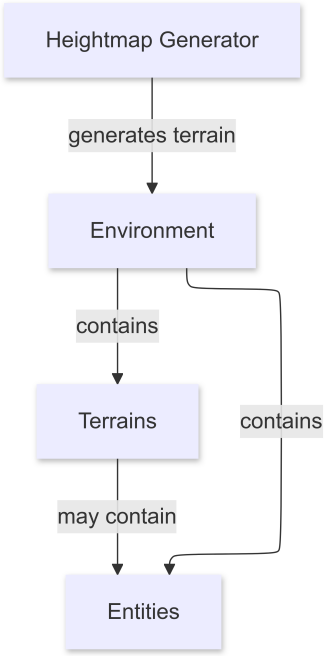


Figure 4.3: Flow diagram of the environment creation

- The systematic integration of KG-RL techniques can be thoroughly evaluated under various conditions.
- Performance metrics can be collected and analysed to assess the NPCs' decision-making processes and generalization capabilities.

4.2.2 Agent Design and Capabilities

The agent in this environment serves as the primary actor, tasked with navigating the procedurally generated world, discovering outposts, and optimizing its route between them. Its design is crucial to the study of how KG integration affects reinforcement learning in complex, dynamic environments.

4.2.2.1 Agent Attributes

The agent is characterized by several key attributes:

Vision Range: This parameter determines the size of the agent's observable area, directly influencing the input dimensions for both the Convolutional Neural Network (CNN) and GNN components of the model. A larger vision range allows the agent to perceive more of the environment in a single observation, potentially reducing the number of movements required to fully explore the world. Conversely, a smaller vision range necessitates more movement for comprehensive exploration.

Resource Capacity: The agent can carry up to five units each of wood and stone. This limitation introduces strategic considerations in resource management and utilization.

Energy Management: Unlike traditional energy systems in games, the agent's energy is not a renewable resource. Instead, the total energy expended over the

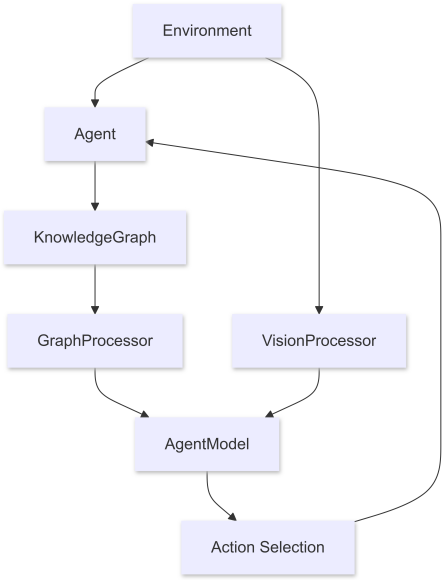


Figure 4.4: Agent Data Flow

course of the agent's journey serves as the primary optimization metric. The agent's goal is to minimize this total route energy.

4.2.2.2 Action Space

The agent's action space consists of 11 distinct actions:

Movement Actions: The agent can move Left, Right, Up, or Down. Each movement incurs an energy cost based on the terrain type of the destination tile.

Scout: This action allows the agent to discover new areas within its vision range, updating the Knowledge Graph with newly observed terrain and entities.

Build Path: The agent can use one unit of wood to create a wooden path on the current tile, reducing the energy cost for future traversals of that tile.

Place Rock: Using one unit of stone, the agent can modify water terrain. This action can transform Deep Water into Water, or Water into Plains, effectively altering the landscape to create more efficient routes.

Collect Resource: Four separate actions allow the agent to collect resources (wood or stone) from adjacent tiles in each cardinal direction.

4.2.2.3 Integration with Knowledge Graph and Environment

The agent's decision-making process is intimately tied to both its current observations and the cumulative knowledge represented in the Knowledge Graph. As shown in Figure 4.4, the Environment feeds information to both the Agent and VisionProcessor directly, while the Agent maintains and updates the KnowledgeGraph, which is then processed by the GraphProcessor. Both processed streams of information are combined in the AgentModel to determine Action Selection.

As the agent moves and acts:

- New terrain and entities discovered within the vision range are added to the Knowledge Graph.

- The agent's position within the Knowledge Graph is updated, ensuring that subsequent decisions are made with accurate spatial context.
- Actions that modify the environment (such as building paths or placing rocks) are reflected in both the game world and the Knowledge Graph, maintaining consistency between the two representations.

This integration between the agent's actions, the environment state, and the Knowledge Graph representation forms the core of the study's exploration into KG-augmented reinforcement learning. The agent must learn to effectively utilize both its immediate observations and the growing knowledge base to make decisions that minimize total energy expenditure while successfully navigating between outposts.

4.2.3 Development Tools

4.2.3.1 Python

Python was selected as the primary programming language due to its versatility and extensive ecosystem of machine learning and scientific computing libraries. The following advantages were identified:

- Python's readability and syntactic simplicity facilitate rapid prototyping and experimentation.
- An active community provides ongoing support and extensive documentation, enabling efficient troubleshooting and code enhancement.
- Libraries such as NumPy, Matplotlib, and Pandas were utilized for numerical computation, data visualization, and data manipulation, respectively.

4.2.3.2 PyTorch

The PyTorch deep learning framework was chosen for its capability to dynamically create computational graphs, allowing flexibility in model design. The following reasons motivated its selection:

- PyTorch supports efficient debugging, which is essential for developing complex models involving neural networks and graph-based structures.
- The framework offers efficient GPU acceleration, significantly reducing the training time for models.
- Its native support for both tensor computations and graph-based operations makes it well-suited for integrating KG and RL in this research.

4.2.3.3 Gymnasium

Gymnasium, a widely used API for reinforcement learning, provides a standardized interface for creating and interacting with RL environments. Its role in this research is pivotal for:

- Offering a consistent interface for implementing RL algorithms.
- Supporting the creation of custom environments that adhere to RL standards, facilitating seamless integration of the research-specific game world.

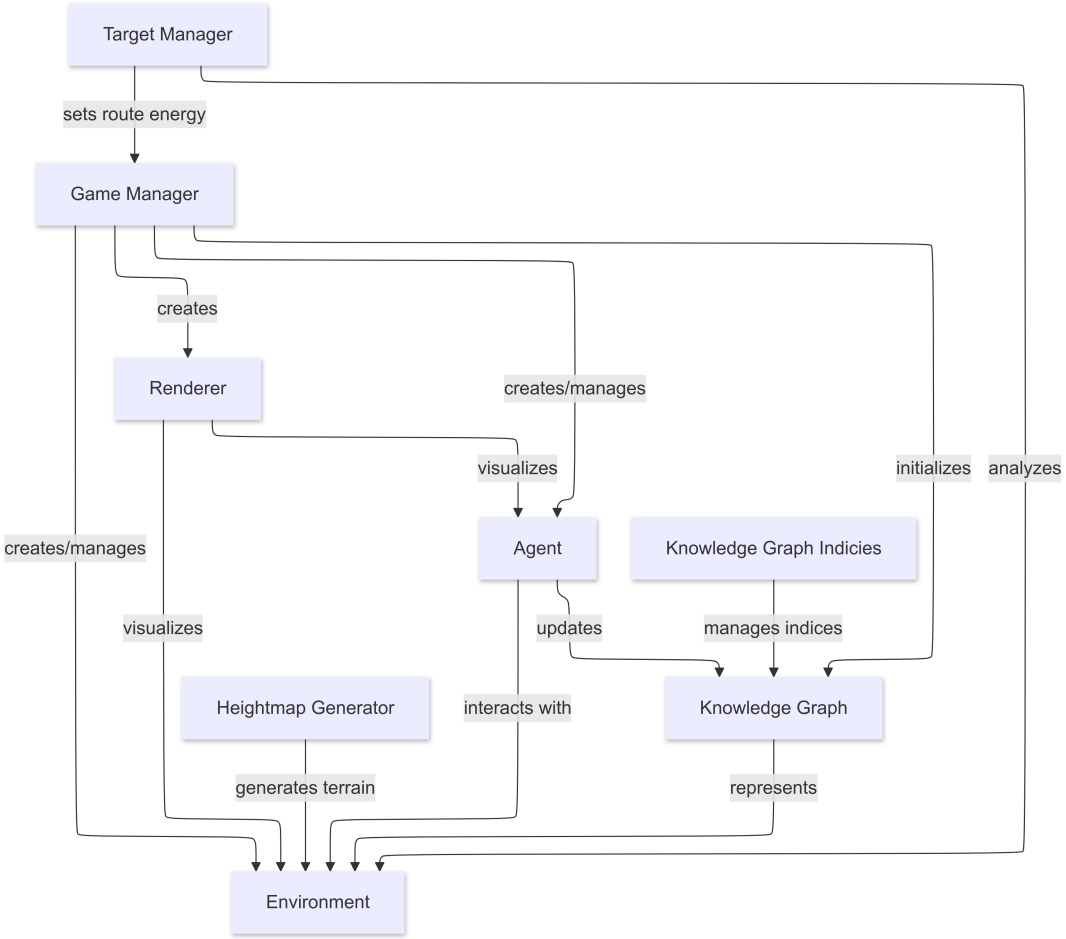


Figure 4.5: Game Manager to Environment

4.2.3.4 Pygame

Pygame was selected as the simulation environment framework due to its simplicity and cross-platform compatibility. The following aspects were emphasized:

- Gymnasium documentation uses Pygame to create custom environments
- Pygame allows rapid prototyping of 2D environments, keeping development focused on the KG-RL integration rather than game-specific complexities.
- The library provides low-level access to essential game components such as rendering

4.2.4 Environment Features

4.2.4.1 Procedural Generation

The game environment employs procedural generation techniques to create diverse and unpredictable game worlds. As shown in Figure 4.5, the Game Manager coordinates this process by managing the Target Manager which sets route energy targets, the Heightmap Generator which creates the terrain, and the Agent and Knowledge Graph components which interact with the generated Environment. As shown in 4.6

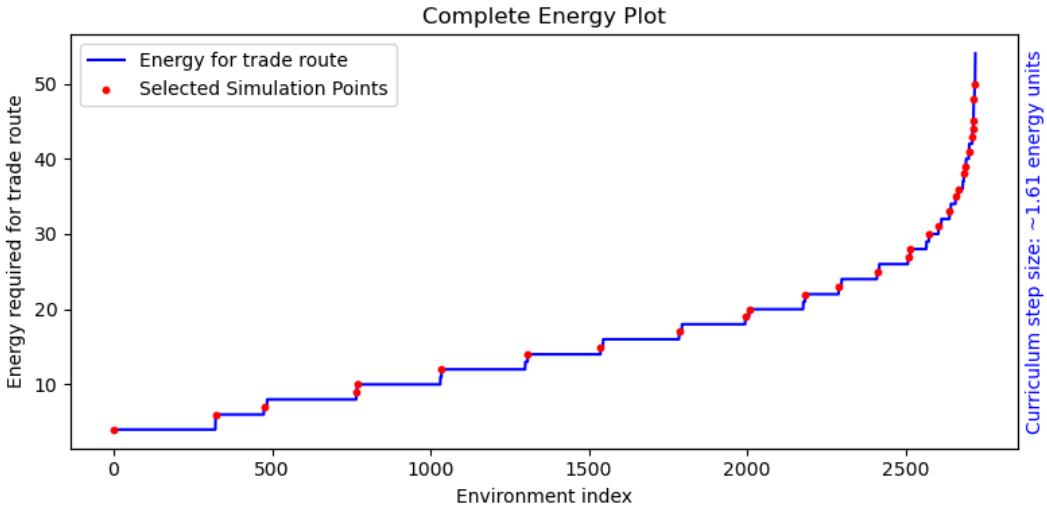


Figure 4.6: Complete Energy Plot of the curriculum steps

the distribution in which the curriculumns were created was not ideal for a smooth transition between curriculla, thus a mean curriculum step size was calculated and used to decide which ones would be used as the lesson steps. This can be seen in 4.7. This technique ensures:

- Each training episode presents a unique environment, which helps prevent NPCs from overfitting to specific scenarios and enhances their ability to generalize across varying conditions.
- The game world can scale in size and complexity without requiring manual design for each scenario, promoting the creation of diverse testing conditions.

4.2.4.2 Terrain Generation using Perlin Noise

The terrain in the game world is generated using Perlin noise, a gradient noise algorithm that simulates natural-looking terrain. The generation process involves:

- Creating a 2D noise map, where each point is assigned a value based on its position in the noise function.
- Mapping these noise values to different terrain types such as plains, hills, and water bodies, based on predefined thresholds.
- Applying post-processing techniques to ensure that the generated terrain is coherent and aligned with the desired game design objectives.

The key parameters in the Perlin noise generation process include:

- **Frequency:** Determines the size of the terrain features, with lower frequencies producing larger features like mountains and higher frequencies creating smaller, more detailed features.
- **Octaves:** Defines the number of layers in the noise map, each contributing additional detail to the terrain.
- **Persistence and Lacunarity:** Control how features of different sizes blend and how much detail is added with each octave.

These parameters can be adjusted to generate a wide variety of terrain types, ensuring that NPCs are exposed to environments that require adaptive behaviours.

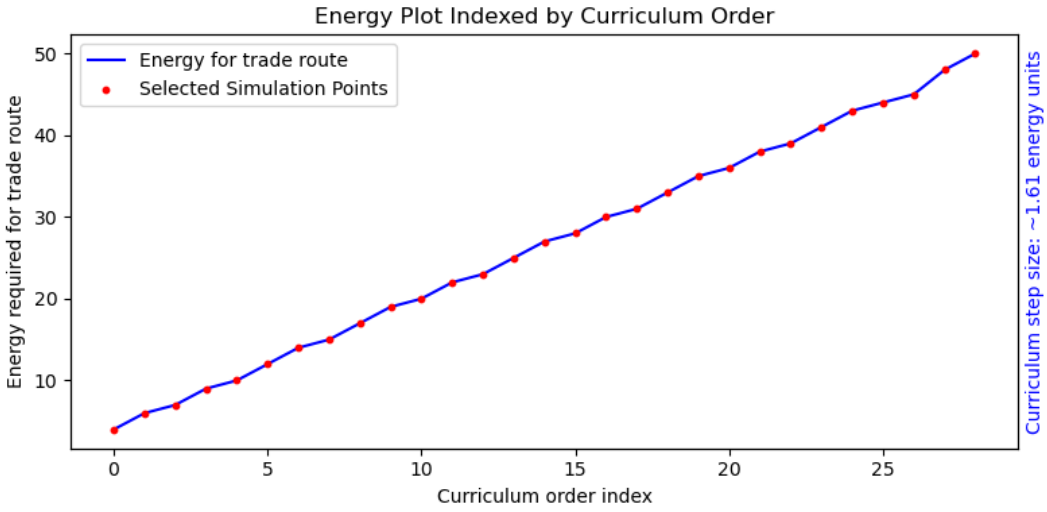


Figure 4.7: Energy Plot Indexed by Curriculum Order

4.2.4.3 Types of Terrain

Several distinct terrain types are present in the game world, each affecting NPC movement and decision-making:

- **Plains:** Flat terrain that is easy to traverse, requiring minimal energy expenditure from the NPCs, can find trees.
- **Hills:** Elevated terrain with moderate energy costs, better probability of finding trees.
- **Mountains:** Steep and challenging terrain, demanding high energy costs, can find rocks.
- **Snow:** Even higher energy costs, can also find rocks.
- **Water:** Steep energy cost, nothing to find.
- **Deep Water:** Steepest energy cost, nothing to find.

These terrain features influence the NPCs' movement costs and resource gathering, creating an environment that challenges the decision-making capabilities of NPCs trained with the KG-RL integration.

The terrains can be seen in 4.1 in light green, dark green, grey, White, light blue and dark blue respectively for Plains, Hills, Mountains, Snow, Water and Deep Water.

4.2.4.4 Entities in the Environment

In addition to diverse terrain, various entities populate the game world, further influencing NPC behaviour. These include:

- **Trees:** To gather wood, which is used to build a wooden path on a terrain tile, thus reducing its energy requirement.
- **Mossy and Snowy Rocks:** Used to gather stone, which is used to landfill water tiles. Deep Water into Water and Water into Plains.
- **Outposts:** The objectives of the agent in the travelling salesman problem.

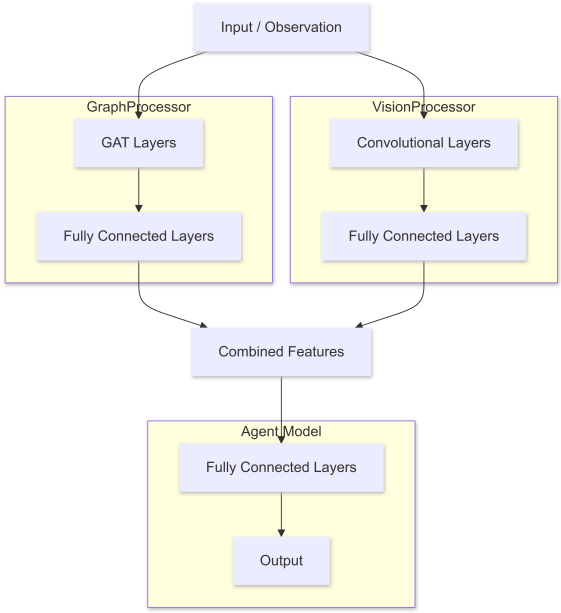


Figure 4.8: Model Architecture

4.3 Model Architecture

4.3.1 Introduction to the Model Architecture

The model architecture (4.8) developed for this research represents a sophisticated hybrid approach that combines visual processing capabilities with graph-based reasoning. This integration is designed to enhance Non-Player Character (NPC) decision-making by leveraging both the rich visual information of the game environment and the structured knowledge represented in a KG.

The architecture consists of three primary components:

- A Vision Processor based on CNNs
- A Graph Processor utilizing GATs
- An Agent Model that integrates the outputs of both processors

This hybrid approach is crucial in the context of Knowledge Graph-Reinforcement Learning (KG-RL) integration for NPCs. By processing both visual and graph-based information, the model can make decisions that are informed not only by what the NPC "sees" in the immediate environment, but also by the broader context and relationships represented in the Knowledge Graph. This combination allows for more sophisticated, context-aware decision-making that can adapt to complex and dynamic game environments.

The architecture's modular design, implemented using PyTorch and PyTorch Geometric, allows for flexibility in adjusting the complexity and capacity of each component. This adaptability is particularly valuable in the context of NPC development, where the complexity of the game world and the desired level of NPC intelligence can vary significantly.

4.3.2 Overall Structure

The model's structure is designed to process game state information through parallel pathways before integrating this information for final decision-making. Here's a high-level description of the three main components:

Vision Processor: This component takes the visual input from the game environment, typically in the form of a 3D tensor representing the visible game state (channels, height, width). It uses a series of convolutional layers, followed by batch normalization, ReLU activations, and fully connected layers to extract relevant features from this visual input. The number of convolutional layers and their parameters are configurable, allowing for adaptation to different visual complexity levels in game environments.

Graph Processor: This component processes the Knowledge Graph representation of the game state. It uses GAT layers to analyse the relationships and attributes encoded in the graph, allowing the model to reason about complex game dynamics and entity interactions. The Graph Processor can handle batched graph data, making it suitable for processing multiple game states simultaneously during training or inference.

Agent Model: This final component combines the outputs from the Vision Processor and Graph Processor. It concatenates the feature vectors from both processors, applies dropout for regularization, and then passes the combined features through a series of fully connected layers. The output of the Agent Model is a feature vector that can be used by a reinforcement learning algorithm (e.g., PPO) to determine the NPC's actions.

These components work together to process game state information in the following manner:

- The visual game state is fed into the Vision Processor, while the graph representation of the game state (node features, edge indices, and edge attributes) is passed to the Graph Processor.
- The Vision Processor extracts relevant visual features from the game state.
- Concurrently, the Graph Processor analyses the Knowledge Graph representation, extracting features that capture the relationships and broader context of the game state.
- The Agent Model takes the outputs from both processors, concatenates them, and processes the combined features through its fully connected layers.
- The final output of the Agent Model is a feature vector that encapsulates both visual and graph-based information, which can be used by a reinforcement learning algorithm to determine the NPC's action in the current game state.

This architecture allows the model to make decisions that are informed by both immediate visual cues and broader contextual knowledge, enabling more intelligent and adaptable NPC behaviour. The combination of CNNs and GATs in this manner represents an approach to addressing the challenges of NPC decision-making in complex, dynamic game environments.

4.3.3 Vision Processor (CNN)

The Vision Processor is designed to extract meaningful features from the visual input of the game environment. It takes as input a tensor of shape (batch_size, channels, height, width), representing a batch of images or game state visual representations.

4.3.3.1 Structure

The Vision Processor utilizes a CNN architecture with the following key components:

Configurable number of convolutional layers: The architecture allows for a variable number of convolutional layers, defined by the "num_conv_layers" parameter. This flexibility enables the model to be adapted to different levels of visual complexity in game environments.

Batch normalization and ReLU activation: After each convolutional layer, batch normalization is applied to standardize the activations, followed by a ReLU (Rectified Linear Unit) activation function. This combination helps to stabilize training and introduce non-linearity into the model.

Flattening operation: After the convolutional layers, the output is flattened to transform the 3D feature maps into a 1D vector, preparing it for the fully connected layers.

Fully connected layers: A series of fully connected layers process the flattened features, gradually reducing the dimensionality to the desired output size. These layers allow the model to learn high-level representations from the convolutional features.

The Vision Processor outputs a tensor of shape (batch_size, features_dim), where features_dim is a configurable parameter (default is 96). This output represents a fixed-size feature vector for each input image in the batch.

4.3.3.2 Rationale for Using CNNs for Visual Processing

CNNs are particularly well-suited for processing visual data due to their ability to capture spatial hierarchies and local patterns in images. In the context of NPC decision-making:

Spatial hierarchy: CNNs can learn to recognize low-level features (e.g., edges, textures) in early layers and more complex, high-level features (e.g., objects, game elements) in later layers.

Translation invariance: The use of convolutional filters allows the model to detect features regardless of their position in the input image, which is crucial for understanding game environments where important elements can appear in various locations.

Parameter efficiency: Through weight sharing, CNNs can efficiently process large input images with relatively few parameters compared to fully connected networks.

Proven effectiveness: CNNs have demonstrated outstanding performance in various computer vision tasks, making them a reliable choice for processing visual game state information.

4.3.4 Graph Processor (GAT)

The Graph Processor is designed to process and extract meaningful features from the Knowledge Graph representation of the game state. It takes as input:

Node features: A tensor of shape (num_nodes, num_features) representing the features of each node in the graph.

Edge indices: A tensor of shape (2, num_edges) representing the connections between nodes.

Batch information: A tensor indicating which graph each node belongs to in a batched setting.

4.3.4.1 Structure

The Graph Processor uses a Graph Attention Network (GAT) architecture with the following components:

Configurable number of GAT layers: The architecture allows for a variable number of GAT layers, defined by the `num_gat_layers` parameter. This flexibility enables the model to capture different levels of complexity in graph relationships.

Multi-head attention mechanism: Each GAT layer employs multiple attention heads, allowing the model to jointly attend to information from different representation subspaces. The number of heads for each layer is configurable through the `gat_heads` parameter.

Global mean pooling: After the GAT layers, global mean pooling is applied to aggregate node features across each graph in the batch, producing a fixed-size representation for each graph.

Fully connected layers: Following the global mean pooling, a series of fully connected layers process the aggregated features, reducing the dimensionality to the desired output size.

4.3.4.2 Output Format and Dimensionality

The Graph Processor outputs a tensor of shape (batch_size, output_dim), where output_dim is a configurable parameter (default is 96). This output represents a fixed-size feature vector for each graph in the batch.

4.3.4.3 Rationale for Choosing GAT over Other Graph Neural Network Architectures

Graph Attention Networks were chosen for processing the Knowledge Graph representation due to several advantages:

Attention mechanism: GATs can learn to assign different importance to different nodes in a neighbourhood, allowing the model to focus on the most relevant information for each task.

Anisotropic: Unlike Graph Convolutional Networks (GCNs), which apply the same transformation to all neighbouring nodes, GATs can apply different transformations to different neighbours, potentially capturing more nuanced relationships.

Inductive learning: GATs can generalize to unseen nodes and graphs, which is crucial for NPCs operating in dynamic game environments where new entities or relationships may be introduced.

Parallelizable: The attention mechanism in GATs can be parallelized across all edges, making it computationally efficient.

Interpretability: The attention weights can provide insights into which relationships in the Knowledge Graph are most important for decision-making, potentially aiding in the development and debugging of NPC behaviours.

4.3.5 Agent Model

The Agent Model serves as the core component of the architecture, integrating the processed visual information from the Vision Processor and the graph-based features from the Graph Processor. Its primary purpose is to combine these diverse sources of information into a unified representation that can be used for NPC decision-making in complex game environments.

4.3.5.1 Structure

Integration of Vision Processor and Graph Processor: The Agent Model initializes and maintains instances of both the Vision Processor and Graph Processor. This modular design allows for independent processing of visual and graph-based data before integration.

Concatenation of Features: The outputs from the Vision Processor and Graph Processor are concatenated along the feature dimension. This operation combines the visual features (e.g., spatial information about the game environment) with the graph features (e.g., relational information from the Knowledge Graph) into a single, rich feature vector.

Dropout for Regularization: A dropout layer is applied to the concatenated features. With a default dropout probability of 0.25, this helps prevent overfitting by randomly setting a portion of the input units to 0 during training, which encourages the model to learn more robust features.

Fully Connected Layers for Final Processing: The combined features are then passed through a series of fully connected layers. These layers learn to process the integrated information, gradually reducing the dimensionality to produce the final output. The number and size of these layers are configurable, allowing for adjustment based on the complexity of the task and the available computational resources.

4.3.5.2 Output Format and Interpretation

The Agent Model outputs a tensor of shape (batch_size, features_dim), where features_dim is set to 192 by default. This output represents a comprehensive feature vector for each input in the batch, encapsulating both visual and graph-based information about the game state. In the context of reinforcement learning, this feature vector can be used by policy and value networks to determine actions and estimate state values, respectively.

4.3.6 Integration of CNNs and GNNs

The integration of CNNs and GNNs in this architecture allows for the simultaneous processing of spatial (visual) and relational (graph) information:

Parallel Processing: Visual data is processed by the CNN-based Vision Processor, while graph data is processed by the GAT based Graph Processor independently and in parallel.

Feature Concatenation: The outputs of these processors are concatenated, creating a joint representation that preserves both visual and relational information.

Joint Learning: The fully connected layers in the Agent Model learn to process this combined representation, effectively fusing the two types of information for decision-making.

4.3.6.1 Handling of Batched Graph Data

The architecture is designed to handle batched inputs efficiently:

Visual Data: The Vision Processor naturally handles batched image data, processing multiple game state visual representations simultaneously.

Graph Data: The Graph Processor uses PyTorch Geometric's utilities to process batched graph data. It handles multiple graphs in a batch by:

- Adjusting edge indices to account for batched node features.
- Using a batch vector to keep track of which nodes belong to which graph in the batch.
- Applying global mean pooling to aggregate node features for each graph in the batch.

Synchronized Batching: The Agent Model ensures that the batch sizes for visual and graph data are aligned, allowing for seamless integration of features from both sources.

4.3.6.2 Benefits of this Hybrid Approach for NPC Decision-making

Comprehensive State Representation: By combining visual and graph-based information, NPCs can make decisions based on both immediate visual cues and broader relational context, leading to more intelligent and context-aware behaviour.

Flexibility in Information Processing: The architecture can adapt to different types of game environments, whether they are more visually oriented or more dependent on complex relationships between game entities.

Improved Generalization: The dual-nature of the input allows NPCs to generalize better across different scenarios, as they can rely on either visual or relational information when one or the other is more relevant or reliable.

Enhanced Learning Efficiency: By providing structured relational information through the Knowledge Graph alongside visual data, the model can potentially learn complex behaviours more efficiently than from visual data alone.

Interpretability: The separate processing of visual and graph data allows for easier interpretation of what information the NPC is using for decision-making, which can be valuable for debugging and improving NPC behaviour.

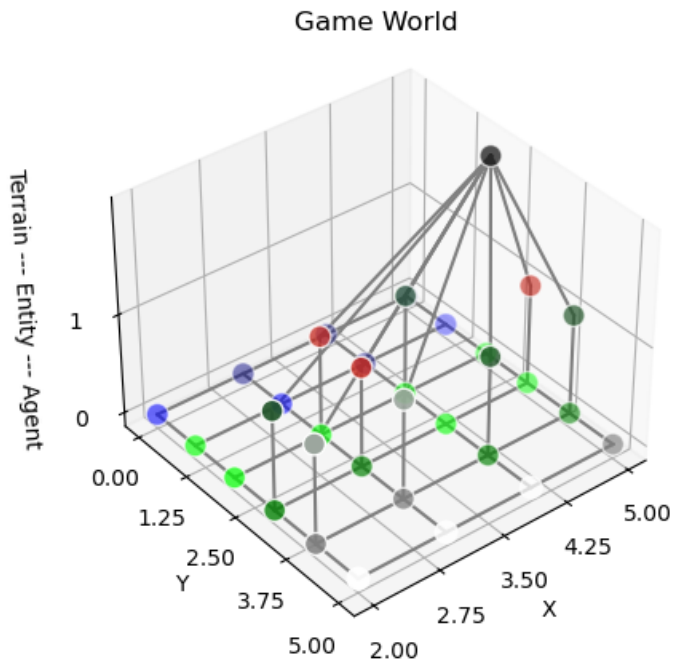


Figure 4.9: Knowledge Graph of the Game World

Scalability: The modular nature of the architecture allows for easy scaling of either the visual or graph processing components based on the specific requirements of different game environments or NPC roles.

4.4 Dynamic Knowledge Graph Integration

4.4.1 KG Structure and Components

The KG in this system is implemented as a dynamic, multi-layered representation of the game world. As shown in Figure 4.9, the colours of the nodes in the representation match those of the tiles in the game world, with outposts shown in red and the player in black. Its structure consists of:

Nodes: Represent entities in the game world, including terrain features, game objects, and the player.

Edges: Represent relationships between entities, such as spatial connections or interactions.

Attributes: Store information about nodes and edges, including their types and properties. For entities this includes the coordinates, z-level, entity type and mask, and the edges have the length and the mask.

The KG uses a three-layer structure to represent different aspects of the game world:

- Terrain Layer (z_level = 0): Represents the base terrain types.
- Entity Layer (z_level = 1): Represents objects and entities in the game world.

- Player Layer (z_level = 2): Represents the player's position.

Entities in the game world are mapped to KG components as follows:

- Terrain types (e.g., plains, water) are represented as nodes in the terrain layer.
- Game objects (e.g., trees, rocks, outposts) are represented as nodes in the entity layer.
- The player is represented as a single node in the player layer.
- Spatial relationships between adjacent terrain tiles are represented as edges.
- Interactions between entities and terrain are represented as edges between the entity and terrain layers.

4.4.2 Subgraph Input

In each stage of the ablation study, the input for the graph processor is determined based on the current completion percentage of the KG. The effective distance for subgraph extraction is calculated by multiplying the environment size by this percentage, with a minimum threshold set to the agent's vision range. This ensures that even with low KG completion, the agent has access to its immediate surroundings.

Using this calculated distance, a K-hop subgraph is extracted from the overall graph. This method retrieves the local neighbourhood around the agent, ensuring that the input size remains consistent regardless of the completion level. The resulting subgraph provides a fixed number of nodes and edges, maintaining uniformity in the graph processor's input during the learning process.

4.5 Reward System Design

The reward function is a crucial component of the reinforcement learning system, designed to guide the agent's behaviour towards efficient route planning and exploration. As shown in Figure 4.10, the reward R at each step t is calculated as follows:

R(t) = base_reward(t) + outpost_reward(t) + completion_reward(t) + improvement_reward(t) + proximity_reward(t) + penalty(t) Where:

- base_reward(t) = penalty_per_step *current_terrain_energy_requirement + time_penalty_factor* episode_step
- outpost_reward(t) = new_outpost_reward *(1 + outpost_reward_increase_factor* (outposts_visited - 1)) if new outpost reached, else 0
- completion_reward(t) = completion_reward * (1 + completion_time_bonus_factor / episode_step) if all outposts visited, else 0
- improvement_reward(t) = route_improvement_reward * improvement if new best route found, else 0
- proximity_reward(t) = closer_to_outpost_reward *(previous_distance - current_distance) / previous_distance if closer to unvisited outpost, else farther_from_outpost_penalty* (current_distance - previous_distance) / previous_distance
- penalty(t) = circular_behaviour_penalty if agent position in recent path, else 0

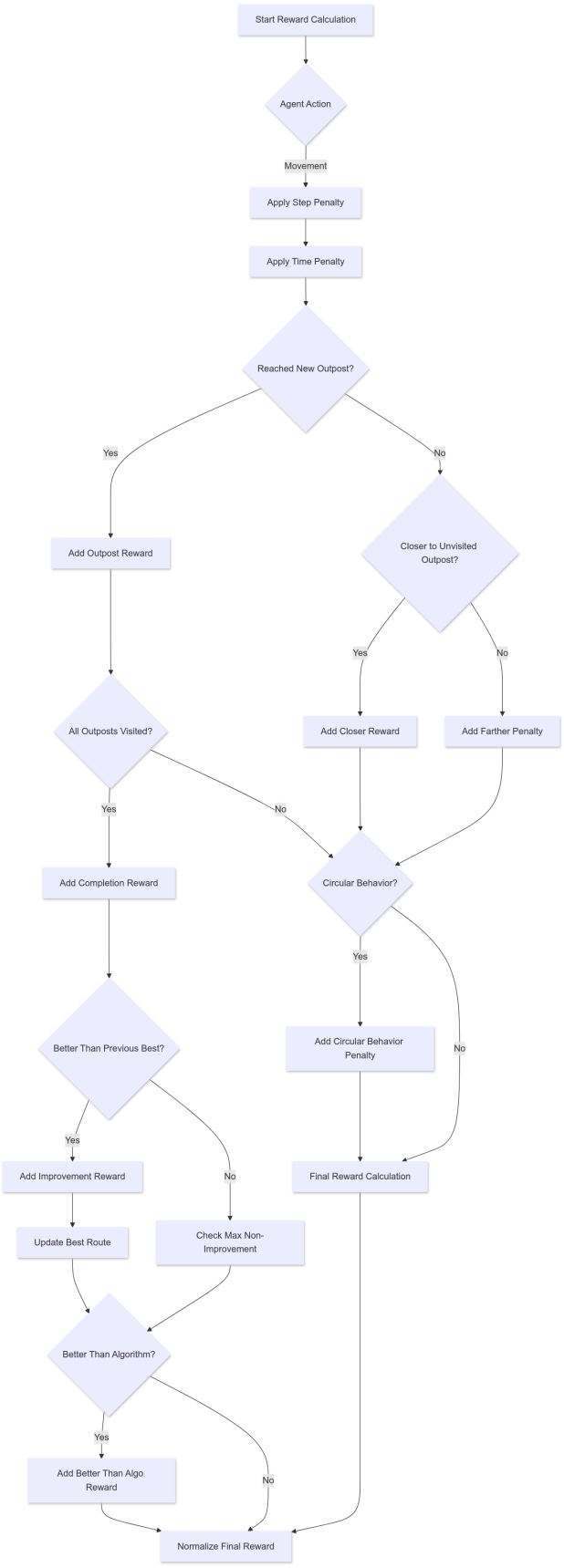


Figure 4.10: Reward Flow

The final reward is normalized to a range of [0, 100] to ensure consistency across different scenarios.

4.5.1 Rationale for chosen rewards and penalties

Base Reward: The penalty_per_step (-0.5) encourages the agent to complete the task quickly, while the time_penalty_factor (-0.01) adds increasing pressure over time.

Outpost Reward: The new_outpost_reward (30) provides a significant positive reinforcement for reaching key objectives. The increasing reward for subsequent outposts (outpost_reward_increase_factor = 0.5) encourages the agent to visit all outposts rather than repeating easy ones.

Completion Reward: The completion_reward (100) offers a large incentive for finishing the entire route. The time bonus (completion_time_bonus_factor = 1.0) rewards faster completions.

Improvement Reward: The route_improvement_reward (200) and better_route_than_algo_reward (200) strongly encourage the agent to find increasingly efficient routes, even surpassing the algorithmic best.

Proximity Reward: The closer_to_outpost_reward (0.55) and farther_from_outpost_penalty (-1.0) guide the agent towards unvisited outposts while exploring.

Penalties: The circular_behaviour_penalty (-2.0) discourages repetitive movements, promoting exploration and efficiency.

4.5.2 Balance between immediate goals and long-term capabilities

The reward system is designed to balance short-term task completion with long-term learning and improvement:

Short-term goals are encouraged through immediate rewards for reaching outposts and penalties for inefficient movements.

Long-term learning is promoted by:

- Increasing rewards for subsequent outposts, encouraging full route completion.
- Large rewards for route improvement, motivating the agent to optimize beyond just task completion.
- The efficiency calculation, which compares the agent's performance to an algorithmic best, encouraging continuous improvement.

To prevent reward hacking and unintended behaviours:

- The circular_behaviour_penalty discourages repetitive movements that might exploit local reward patterns.
- The normalization of rewards to a fixed range (0-100) prevents the agent from exploiting any single component of the reward function.
- The max_not_improvement_routes parameter (5) prevents the agent from stagnating on suboptimal strategies.
- The dynamic nature of the reward function, which considers factors like current efficiency and improvement, makes it harder for the agent to exploit fixed

patterns.

This reward system aims to create a learning environment where the agent is continuously motivated to improve its route-finding capabilities while efficiently completing the immediate task of visiting all outposts.

4.5.3 KG Expansion Process

The KG expands dynamically as the agent explores the game world. This process is managed by the agent, which calls specific methods to update the KG:

Discovering New Coordinates:

- The agent calls `discover_this_coordinate(x, y)` when exploring a new area.
- This method activates the corresponding terrain node and any entity nodes at that location.
- New edges are created to connect the discovered nodes to previously known nodes.

Adding New Entities:

- When a new entity is encountered, a new node is created in the entity layer.
- The `create_node()` method is used to add the new entity to the graph.

Updating Existing Nodes:

- The agent can call methods like `build_path_node(x, y)` or `elevate_terrain_node(x, y)` to modify existing nodes.
- These methods update the attributes of the corresponding nodes in the KG.

Removing Entities:

- When an entity is removed from the game world, the agent calls `remove_entity_node(x, y)`.
- This method deactivates the corresponding node in the KG and updates related edges.

The KG does not employ explicit pruning or consolidation mechanisms. Instead, it uses a mask system to activate or deactivate nodes and edges, allowing the graph to maintain a complete representation of the known world while focusing computations on relevant parts.

4.5.4 KG Utilization in Decision Making

The agent utilizes the KG for decision making through the following processes:

Querying the KG:

- The agent can access information about specific locations or entities by querying their corresponding nodes in the KG.
- This is done through methods like `get_node_idx(pos, z_level)` to retrieve node information.

Extracting Relevant sub-graphs:

- The `get_sub-graph()` method is used to extract a localized portion of the KG centred around the player's current position.
- This sub-graph includes all nodes and edges within a certain distance (defined by the `vision_range` parameter) from the player.

Reasoning and Inference:

- The GNN in the agent's model architecture performs reasoning over the extracted sub-graph.
- This allows the agent to infer relationships and make decisions based on both local and global information represented in the KG.

4.5.5 Integration with the Model Architecture

The KG integrates with the Graph Processor component of the agent's model architecture:

Interface with Graph Processor:

- The sub-graph extracted by `get_sub-graph()` is converted into a PyTorch Geometric `Data` object.
- This object contains node features, edge indices, and edge attributes, which are directly fed into the GAT layers of the Graph Processor.

Preprocessing of KG Data:

- Node features are normalized and encoded to match the input requirements of the GAT layers.
- Edge attributes are processed to represent the types and strengths of relationships between nodes.

KG Updates and Learning Process:

- As the agent explores and modifies the game world, the KG is continuously updated.
- These updates affect the sub-graphs extracted for decision-making, allowing the agent to adapt its behaviour based on the most current information.
- The dynamic nature of the KG challenges the agent to generalize its learning across changing graph structures, promoting adaptability in its decision-making process.

This dynamic KG integration allows the agent to maintain an up-to-date representation of the game world, make informed decisions based on both local and global information, and adapt its behaviour as it discovers and interacts with the environment.

4.6 Training

4.6.1 Parameters and Hyperparameters

The training process utilizes the Proximal Policy Optimization (PPO) algorithm from the Stable Baselines3 library. The following hyperparameters are used:

- n_steps: 4096 (2048 * 2) - The number of steps to run for each environment per update.
- batch_size: 512 - The number of samples processed before updating the model.
- learning_rate: 6e-4 - The step size for updating the model parameters.
- gamma: 0.995 - The discount factor for future rewards.

Additional parameters include:

- num_actions: 11 - The number of possible actions the agent can take.

- number_of_environments: 3000 - The total number of game environments created.
- number_of_curricula: 30 - The number of curriculum stages.
- min_episodes_per_curriculum: 4 - The minimum number of episodes to complete before considering advancement to the next curriculum stage.
- performance_threshold: 0.85 - The performance threshold for advancing to the next curriculum stage.
- total_timesteps: 100000 - The total number of timesteps for training.

4.6.2 Game environment parameters:

- num_tiles: 5 - The size of the game world grid.
- screen_size: 50 - The pixel size of the game screen.
- vision_range: 1 - The range of vision for the agent.

Knowledge Graph completeness values: [0.25, 0.5, 0.75, 1.0] - Used for ablation studies to test different levels of Knowledge Graph completeness, the size of the KG in relative size compared to the environment.

No formal hyperparameter tuning process was conducted. The hyperparameters were primarily adjusted to fit within the constraints of the available GPU memory.

4.6.3 Hardware and Software Setup

Hardware:

- GPU: NVIDIA GeForce RTX 3060
- GPU Memory: 6144 MiB

Software:

- NVIDIA Driver Version: 560.94
- CUDA Version: 12.6

The training process utilizes PyTorch for deep learning computations and Stable Baselines3 for the implementation of the PPO algorithm. The custom environment is implemented using the Gymnasium library, which provides a standardized interface for reinforcement learning environments. Pygame is used for rendering the game world.

4.7 Quantitative Research Methodology

4.7.1 Metrics for Evaluation

Several key performance metrics were collected during training to evaluate the agent's performance comprehensively:

Core Performance Metrics:

- Efficiency: Ratio between optimal path length and agent's path length
- Gap: Absolute difference between agent's path length and optimal path length
- Improvement: Relative change in performance over time
- Performance: Overall score considering multiple factors

Environmental Metrics:

- Game Manager Index: Tracking which environment configuration is being used
- Best Route Energy: Optimal energy cost for the current route
- Target Route Energy: Expected energy cost for the route
- Curriculum Step: Current stage in the curriculum learning process

Behavioural Metrics:

- Action Distribution: Frequency of each action type (Actions 0-10) chosen by the agent

This set of metrics was designed to capture both quantitative performance and qualitative behavioural patterns.

4.7.2 Alternative Validation Approaches

While this research utilised a custom game environment for validation through ablation studies, utilising the Crafter environment [19] could have enhanced the validation by via benchmarking against other approaches.

Other options would include:

Comparative Studies:

- Traditional RL baselines without KG integration
- Alternative knowledge representation methods (e.g., semantic networks, ontologies)
- Human player performance benchmarking

Cross-Domain Validation:

- Different game genres (RPGs, strategy games)
- Non-game applications (robotic navigation, autonomous systems)

The custom environment was chosen primarily for:

- Precise control over experimental conditions, particularly KG completeness levels
- Direct alignment with research objectives
- Resource and time efficiency
- Ability to isolate key variables affecting KG-RL integration

Future work could benefit from implementing these alternative validation approaches to better establish the generalizability of the findings and compare performance across different contexts.

4.7.3 Ablation Study Design

An ablation study was conducted to isolate and examine the impact of KG completeness on the agent's performance. The primary component being ablated was:

- Knowledge Graph Completeness: Varied at levels of 0.25, 0.5, 0.75, and 1.0.

The experimental setup for the ablation study was structured as follows:

For each level of KG completeness:

- A new training instance was initialized.
- The environment was configured with the specified KG completeness.
- The model was trained and subsequently evaluated.
- Results, including mean reward, standard deviation, and detailed training metrics, were recorded.

A consistent base configuration was maintained across experiments, with KG completeness being the variable parameter.

This methodology was designed to enable a systematic evaluation of the relationship between KG completeness and agent performance. By isolating this variable, insights could be gained into the significance of knowledge representation in the learning process. The consistent base configuration across experiments ensures that observed differences can be attributed primarily to changes in KG completeness, allowing for a focused analysis of this component's impact on overall system performance.

5
Results

5.1 Introduction

5.1.1 Study Objectives

This study aimed to investigate the integration of KGs with RL for enhancing Non-Player Character (NPC) decision-making in video game environments. The primary objectives were:

- To evaluate the impact of varying KG completeness levels on agent performance
- To assess the effectiveness of a hybrid CNN-GAT model architecture in processing both visual and graph-based inputs
- To analyse the relationship between KG completeness and key performance metrics

5.1.2 Experimental Setup Overview

The experiment was conducted using a custom-built game environment, simulating a TSP with additional complexities. The environment featured procedurally generated terrain, multiple resource types, and strategically placed outposts. A hybrid model architecture combining CNNs and GATs was implemented to process both visual game state and KG representations.

The study employed an ablation methodology, varying the KG completeness levels at 25%, 50%, 75%, and 100%. For each completeness level, 25 independent runs were conducted. The Proximal Policy Optimization (PPO) algorithm was utilised for training the agent, with a complex, multi-component reward function guiding the learning process.

5.2 Overall Performance Metrics

5.2.1 Key Metrics Presentation

Three primary metrics were employed to evaluate agent performance:

- **Efficiency:** Defined as the ratio of optimal path length to the agent's path length, expressed as a percentage.
- **Gap:** The absolute difference between the agent's path length and the optimal path length.

- **Improvement:** The relative change in the agent's performance over time, calculated as (previous_efficiency - current_efficiency) / current_efficiency.

5.2.2 Summary Statistics

To better understand the impact of KG completeness on agent performance, an analysis was conducted for each completeness level: 25%, 50%, 75%, and 100%. These represent the data from the last 20% of the metrics collected, so it only the stable data.

The results are summarized in Tables 5.1 and 5.2, and are supplemented with plots that visualize the trends in the metrics over time, smoothed using aggregation windows to provide clearer insights.

Metric	25% KG	50% KG	75% KG	100% KG
Efficiency (Mean %)	8.96	5.97	3.17	6.84
Efficiency, 50th Percentile	2.68	5.48	2.88	4.08
Gap (Mean)	114.55	53.50	101.25	35.40
Gap, 50th Percentile	36.25	17.25	33.75	23.50
Improvement (Mean)	270.51	49.20	176.58	84.38
Improvement, 50th Percentile	-32.21	-16.60	-44.44	92.86

Table 5.1: Summary Statistics for 100,000 training steps.

Metric	25% KG	50% KG	75% KG	100% KG
Efficiency (Mean %)	25.99	3.16	1.59	3.62
Efficiency, 50th Percentile	4.30	0.09	0.09	1.51
Gap (Mean)	279.01	1360.78	1287.06	259.75
Gap, 50th Percentile	22.25	1133.50	1090.25	65.25
Improvement (Mean)	701.53	181.00	364.10	668.25
Improvement, 50th Percentile	0.00	0.00	0.00	0.00

Table 5.2: Summary Statistics for 1,000,000 training steps

5.2.3 Initial Observations on General Trends

Analysis of the extended experiment data reveals several striking trends.

Efficiency: There is a dramatic difference across KG completeness levels. The 25% KG completeness significantly outperforms all other levels with a mean efficiency of 25.99%, compared to the next highest of 3.62% for 100% KG completeness. This suggests that minimal KG information may be more beneficial than more complete information.

Gap: The gap values exhibit extreme variability. The 50% and 75% KG completeness levels show dramatically higher mean gaps (1360.78 and 1287.06 respectively) compared to 25% and 100% levels (279.01 and 259.75). This indicates that intermediate levels of KG completeness may lead to significantly suboptimal path planning.

Improvement: While all KG completeness levels show positive mean improvement values, there is high variability. The 25% and 100% KG completeness levels show the highest mean improvements (701.53 and 668.25 respectively). However, the median improvement for all levels is 0, suggesting that improvements are driven by outliers rather than consistent performance gains.

Non-linear Relationship: There is a highly non-linear relationship between KG completeness and performance metrics. The lowest KG completeness (25%) significantly outperforms higher completeness levels in terms of efficiency and maintains competitive performance in other metrics.

Performance Degradation: Higher KG completeness levels (50%, 75%, 100%) show substantially worse performance compared to the 25% level, particularly in efficiency and gap metrics. This suggests that more complete knowledge graphs may introduce complexities that the agent struggles to utilize effectively over extended training periods.

These observations reveal a counterintuitive relationship between KG completeness and agent performance, with minimal KG information leading to the best outcomes in terms of efficiency. This suggests that the integration of more complete knowledge graphs may introduce complexities that the agent struggles to leverage effectively, even with extended training. The high variability in performance metrics across all KG completeness levels also indicates that the agent's learning process remains unstable, regardless of the amount of knowledge provided.

5.3 Performance Across KG Completeness Levels

5.3.1 25% KG Completeness

At the lowest KG completeness level, the agent demonstrated the following performance:

Shorter Run (100,000 steps):

- **Mean Efficiency:** 8.96% (median 2.68%)
- **Mean Gap:** 114.55 (median 36.25)
- **Mean Improvement:** 270.51 (median −32.21)

Longer Run (1,000,000 steps):

- **Mean Efficiency:** 25.99% (median 4.30%)
- **Mean Gap:** 279.01 (median 22.25)
- **Mean Improvement:** 701.53 (median 0.00)

Figure 5.1 illustrates the efficiency over steps for the shorter run across all KG completeness levels without aggregation. The agent with 25% KG completeness maintains relatively high efficiency compared to higher completeness levels.

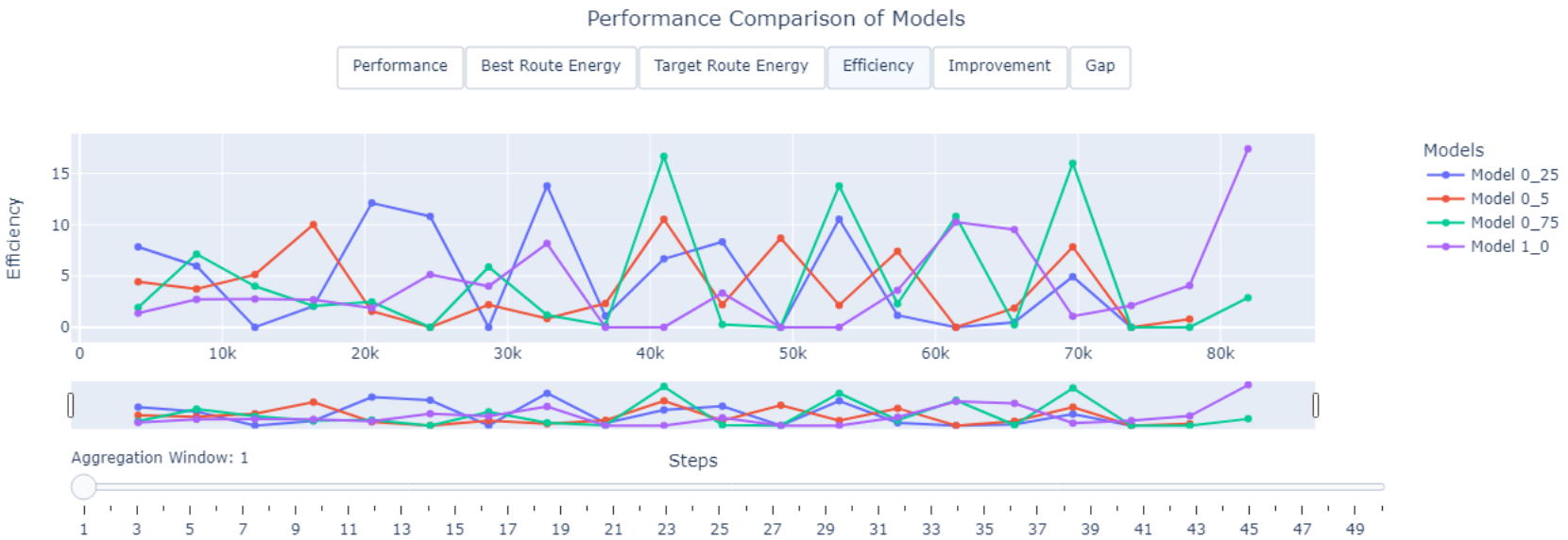


Figure 5.1: Efficiency over steps for the shorter run across different KG completeness levels (no aggregation).

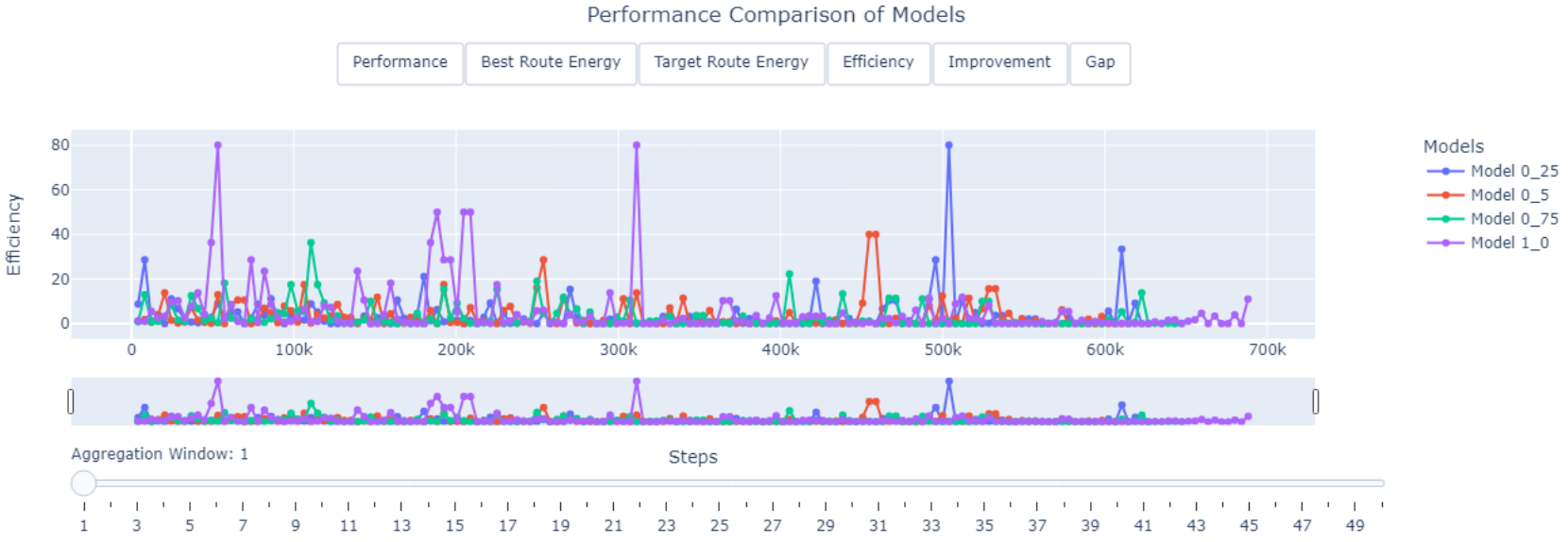


Figure 5.2: Efficiency over steps for the longer run across different KG completeness levels (no aggregation).

Similarly, Figure 5.2 shows the efficiency over steps for the longer run without aggregation. The agent at 25% KG completeness demonstrates a significant increase in mean efficiency over the longer training period.

Notably, despite having minimal KG information, the agent performs comparably or better than higher completeness levels in both the shorter and longer runs. This suggests that limited but relevant information may help the agent focus on immediate, actionable data, leading to better decision-making.

5.3.2 50% KG Completeness

At 50% KG completeness, the agent's performance metrics were:

Shorter Run (100,000 steps):

- **Mean Efficiency:** 5.97% (median 5.48%)
- **Mean Gap:** 53.50 (median 17.25)
- **Mean Improvement:** 49.20 (median −16.60)

Longer Run (1,000,000 steps):

- **Mean Efficiency:** 3.16% (median 0.09%)
- **Mean Gap:** 1,360.78 (median 1,133.50)
- **Mean Improvement:** 181.00 (median 0.00)

As depicted in Figure 5.1, the efficiency over time at 50% KG completeness in the shorter run shows moderate performance. However, in the longer run (Figure 5.2), the efficiency decreases significantly, indicating that the agent struggles to improve performance over time.

The gap metric in Figure 5.3 shows increased variability and higher values at 50% completeness in the longer run, indicating less optimal path selection compared to the shorter run.

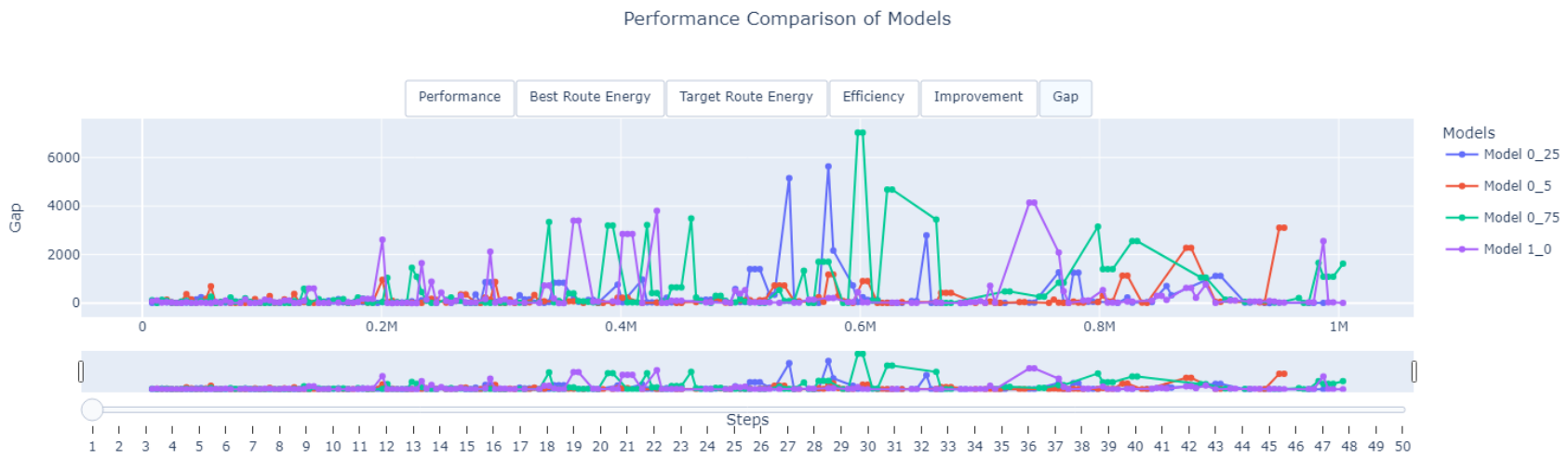


Figure 5.3: Gap over steps for the longer run across different KG completeness levels.

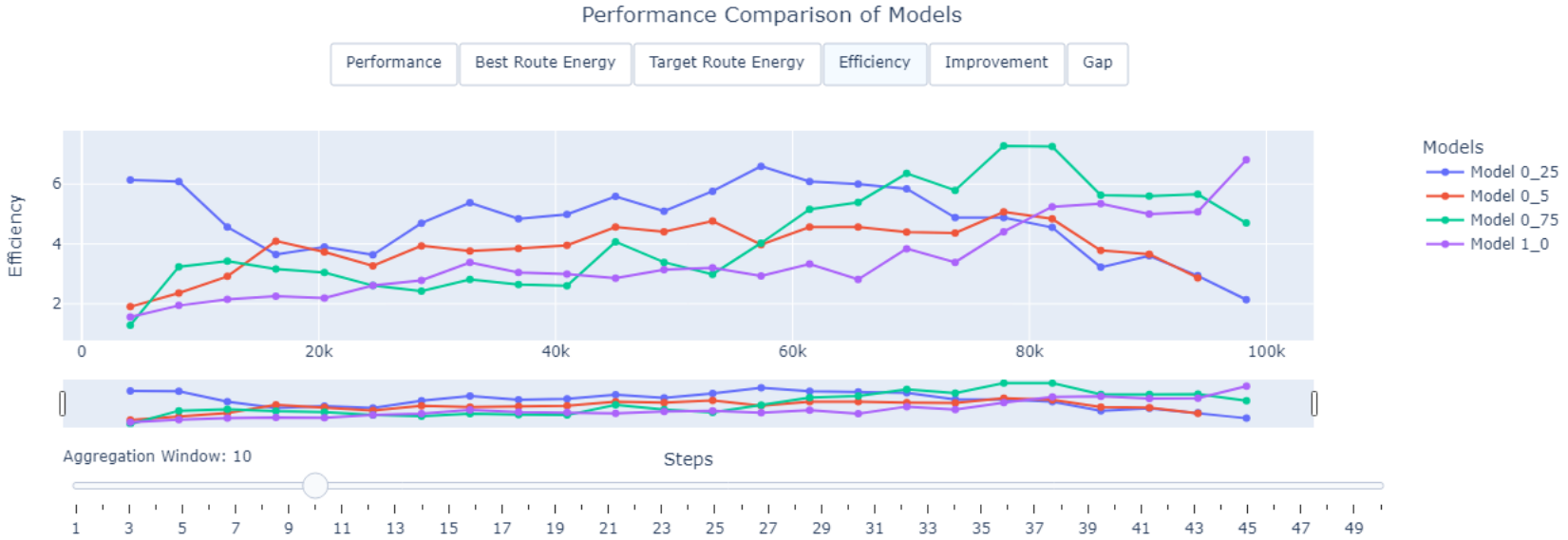


Figure 5.4: Efficiency over steps for the shorter run across different KG completeness levels (10-step aggregation).

This suggests that partial KG information at this level may introduce complexity that hinders the agent's ability to make efficient decisions, especially over extended training periods.

5.3.3 75% KG Completeness

At 75% KG completeness, the agent's performance was:

Shorter Run (100,000 steps):

- **Mean Efficiency:** 3.17% (median 2.88%)
- **Mean Gap:** 101.25 (median 33.75)
- **Mean Improvement:** 176.58 (median −44.44)

Longer Run (1,000,000 steps):

- **Mean Efficiency:** 1.59% (median 0.09%)
- **Mean Gap:** 1,287.06 (median 1,090.25)
- **Mean Improvement:** 364.10 (median 0.00)

Figures 5.1 and 5.2 indicate that efficiency remains low and exhibits significant fluctuations over time at 75% KG completeness in both runs. The agent's performance is highly variable, as evidenced by the discrepancies between mean and median values.

The high mean gap values, especially in the longer run, suggest that the agent struggles to utilize the increased amount of KG information effectively, leading to less optimal path selection and inefficient learning.

5.3.4 100% KG Completeness

With full KG information available, the agent's performance was:

Shorter Run (100,000 steps):

- **Mean Efficiency:** 6.84% (median 4.08%)
- **Mean Gap:** 35.40 (median 23.50)
- **Mean Improvement:** 84.38 (median 92.86)

Longer Run (1,000,000 steps):

- **Mean Efficiency:** 3.62% (median 1.51%)
- **Mean Gap:** 259.75 (median 65.25)
- **Mean Improvement:** 668.25 (median 0.00)

In the shorter run, as shown in Figure 5.4, the agent with 100% KG completeness demonstrates moderate efficiency, with less variability compared to 75% completeness. However, in the longer run (Figure 5.5), the efficiency decreases, and variability increases.

The gap metric in Figure 5.3 shows that although the mean gap is lower compared to 75% completeness, variability remains high, indicating inconsistent performance over time.

5.3.5 Comparative Analysis of Non-Linear Relationships

The relationship between KG completeness and agent performance exhibits notable non-linearity, as visualized in Figures 5.5 and 5.6, which compare the efficiency and

performance over time across different KG completeness levels, smoothed using a 20-step aggregation window.

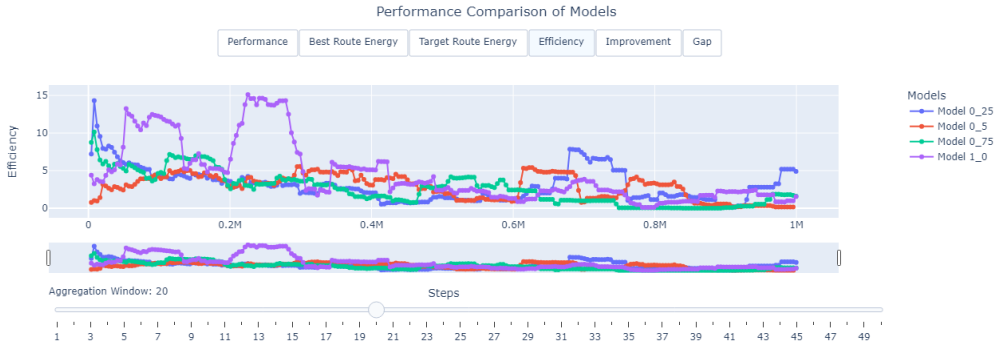


Figure 5.5: Comparison of efficiency over time across different KG completeness levels (longer run, 20-step aggregation).

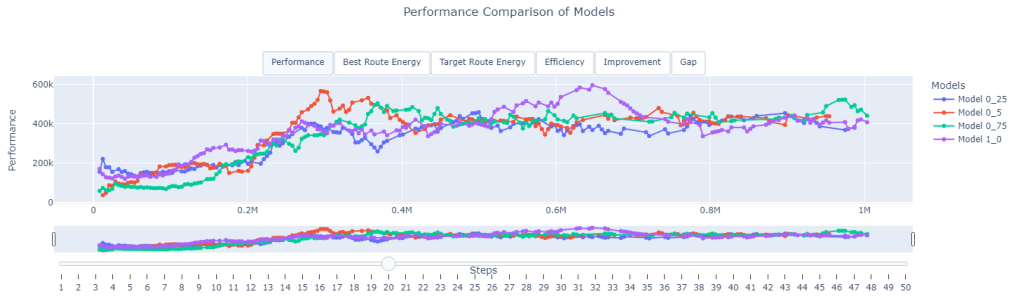


Figure 5.6: Comparison of improvement over time across different KG completeness levels (longer run, 20-step aggregation).

Efficiency: A U-shaped trend is observed, with the lowest efficiency at 50% and 75% completeness levels and higher efficiency at both minimal (25%) and full (100%) KG information. This suggests that partial KG information might be more detrimental than either minimal or complete information.

Gap: As shown in Figure 5.3, the gap metric exhibits a parabolic relationship, peaking at 50% and 75% completeness in the longer run. This indicates that more KG information does not necessarily lead to better path optimization, especially when the information is partial.

Improvement: The tables 5.1 and 5.2 show the improvement across different KG completeness levels. The agent at 25% and 100% completeness levels demonstrates higher improvement over the longer run, while the agent at 50% and 75% completeness levels shows less improvement.

5.3.6 Observations on General Trends

Analysis of the aggregated data reveals several key trends:

Performance Degradation with Partial KG: Intermediate KG completeness levels (50% and 75%) result in lower efficiency and higher gap metrics compared to

minimal (25%) and full (100%) KG completeness, especially over the longer training run. This suggests that partial KG information may introduce complexity without providing sufficient context for effective decision-making.

Minimal KG Information is Beneficial: The 25% KG completeness level consistently outperforms higher completeness levels in terms of efficiency and maintains competitive performance in other metrics. This indicates that limited but relevant information may help the agent focus on immediate, actionable data.

High Variability at Higher Completeness Levels: The standard deviations and discrepancies between mean and median values for efficiency and gap metrics are larger at 50% and 75% KG completeness levels, indicating greater inconsistency in performance, especially over extended training periods.

Non-linear Relationship Between KG Completeness and Performance: The trends suggest a complex relationship where neither minimal nor complete KG information guarantees optimal performance, and partial information may hinder learning. The agent seems to benefit most from either minimal or complete KG information.

Unstable Learning Process: The high variability and inconsistent improvement metrics across all KG completeness levels suggest that the agent's learning process remains unstable, regardless of the amount of knowledge provided. This instability is more pronounced in the longer training runs.

These observations reveal a counterintuitive relationship between KG completeness and agent performance. The integration of more complete knowledge graphs may introduce complexities that the agent struggles to leverage effectively, even with extended training. The plots provide visual evidence of these trends and highlight areas where the agent's performance could be further investigated.

5.4 Learning Challenges and Model Behaviour

The performance metrics across different KG completeness levels revealed significant challenges in the agent's learning process and overall behaviour. This section delves deeper into these issues, focusing on the improvement values, suboptimal performance indicated by efficiency and gap metrics, and potential reasons for the observed learning outcomes.

5.4.1 Analysis of Improvement Values

The improvement values across different KG completeness levels displayed variability, with mean improvements being positive but median improvements often being zero or negative.

Despite the positive mean improvement values, the median improvements were negative or zero in the shorter run and zero in the longer run. This indicates that while there were instances of significant improvement, these were not consistent across episodes. The positive means suggest that improvements were driven by outliers, with a few episodes showing large gains, while the majority of episodes did not exhibit substantial improvement.

5.4.2 Efficiency and Gap Metrics Indicating Suboptimal Performance

The efficiency and gap metrics further indicate suboptimal performance by the agent can also be seen in the tables 5.1 and 5.2.

Low Efficiency: Mean efficiency values were generally low across KG completeness levels. For example, in the longer run, mean efficiency values were:

The low percentages indicate that the agent's path was consistently much longer than the optimal path, especially at higher KG completeness levels.

High Gap Values: Mean gap values in the longer run were significantly higher for the 50% and 75% KG completeness levels.

These large differences between the agent's path length and the optimal path length emphasize the agent's inability to find efficient routes, particularly at intermediate KG completeness levels.

5.4.3 Exploration of Potential Reasons for Suboptimal Learning

Several factors could have contributed to the observed learning outcomes:

Environmental Complexity: The procedurally generated game world, with its varied terrain and resource distribution, may have presented a level of complexity that the agent struggled to navigate effectively. The diversity of scenarios may have hindered the agent's ability to generalize learned strategies.

Influence of KG Completeness: The non-linear relationship between KG completeness and performance suggests that partial knowledge graphs at 50% and 75% completeness may have introduced complexity without providing sufficient benefit. The agent may have been overwhelmed by the additional information without being able to leverage it effectively.

Learning Algorithm Limitations: The Proximal Policy Optimization (PPO) algorithm may not have been fully optimized for this problem domain. The lack of formal hyperparameter tuning might have affected the agent's ability to learn efficiently.

State Representation Challenges: Integrating visual and graph-based inputs may have presented challenges in forming a coherent internal representation of the environment. The agent may have struggled to reconcile information from the CNN and GAT components, leading to suboptimal decision-making.

Reward Function Complexity: The complex, multi-component reward function may have made it difficult for the agent to associate specific actions with positive outcomes. The balance between immediate rewards and long-term goals might not have been effectively achieved.

These potential factors highlight the multifaceted challenges in developing an agent capable of efficiently solving the task. The interaction between the environment, the amount of information provided, the model architecture, and the learning algorithm all contribute to the observed performance.

5.4.4 Summary

The analysis indicates that while there were instances of improvement, the agent's learning was inconsistent and often suboptimal. The positive mean improvements were counterbalanced by zero or negative median improvements, suggesting that significant gains were not widespread across episodes. The low efficiency and high gap metrics further emphasize the agent's difficulties in finding optimal paths.

The findings suggest that both minimal and full KG completeness levels performed better than intermediate levels, but overall performance remained suboptimal. Future investigations could explore adjustments to the model architecture, more extensive hyperparameter tuning, and modifications to the reward function to address these challenges.

5.5 Reward Function Analysis

The reward function plays a crucial role in guiding the agent's learning process. In this study, a complex, multi-component reward function was implemented to encourage desired behaviours and penalize suboptimal actions. This section breaks down the reward function, discusses its complexity, and analyses the potential impact of reward normalisation on learning clarity.

5.5.1 Reward Function Components

The reward function, shown in 4.10, comprises several components, each designed to address specific aspects of the agent's performance:

Base Reward:

$$R_{base} = step\_penalty \cdot current\_terrain\_energy\_requirement + time\_penalty \cdot episode\_step \tag{5.1}$$

New Outpost Reached Reward:

$$R_{outpost} = new\_outpost \cdot (1 + increase\_factor \cdot (outposts\_visited - 1)) \tag{5.2}$$

Route Completion Reward:

$$R_{completion} = completion\_reward \cdot (1 + completion\_time\_bonus\_factor / episode\_step) \tag{5.3}$$

Route Improvement Reward:

$$R_{improvement} = route\_improvement\_reward \cdot improvement \tag{5.4}$$

Proximity Reward:

$$R_{proximity} = closer\_to\_outpost\_reward \cdot \frac{previous\_distance - current\_distance}{previous\_distance} \tag{5.5}$$

if closer to an unvisited outpost, else

$$farther\_from\_outpost\_penalty \cdot \frac{current\_distance - previous\_distance}{previous\_distance} \tag{5.6}$$

Circular Behaviour Penalty:

$$R_{circular} = circular\_behaviour\_penalty \tag{5.7}$$

The total reward is the sum of these components:

$$R_{total} = R_{base} + R_{outpost} + R_{completion} + R_{improvement} + R_{proximity} + R_{circular} \tag{5.8}$$

5.5.2 Reward Function Complexity

The complexity of this reward function presents several challenges for the learning process:

Multiple Objectives: The reward function attempts to balance multiple objectives simultaneously, such as reaching outposts, optimizing route efficiency, and avoiding circular behaviour. This multi-objective optimization can be challenging for the agent to parse and prioritize.

Temporal Dynamics: Some reward components, like the completion reward and improvement reward, depend on long-term outcomes. This temporal aspect can make it difficult for the agent to associate specific actions with their eventual rewards.

Non-linear Scaling: The outpost reward increases non-linearly with the number of outposts visited, potentially overshadowing other reward components as the episode progresses.

Competing Incentives: The proximity reward encourages the agent to move towards unvisited outposts, while the circular behaviour penalty discourages revisiting recent positions. These competing incentives might create confusion in the agent's decision-making process.

Environment-Dependent Components: The base reward's dependence on terrain energy requirements introduces variability based on the specific environment configuration, potentially making it harder for the agent to generalise strategies.

5.5.3 Reward Normalisation and Its Potential Impact

The reward function includes a normalisation step, where the final reward is scaled to a range of [0, 100]. This normalisation process has several implications:

Bounded Reward Range: Normalisation ensures that rewards remain within a consistent range across different episodes and environments. This can help stabilise the learning process by preventing extreme reward values.

Relative Importance Preservation: The normalisation maintains the relative importance of different reward components within each step. However, it may obscure the absolute magnitude of improvements or deteriorations in performance across episodes.

Potential Loss of Granularity: Scaling the reward to a fixed range might lead to a loss of granularity, especially for small improvements or subtle behavioural changes. This could make it challenging for the agent to distinguish between slightly better or worse actions.

Difficulty in Reward Shaping: The normalisation process makes it more challenging to implement reward shaping techniques, as the impact of adjusting individual reward components becomes less predictable after normalisation.

Interpretation Challenges: Normalized rewards may be more difficult for human observers to interpret, as the raw values no longer directly correspond to specific outcomes in the environment.

The complexity of the reward function, combined with the normalisation process, may contribute to the learning challenges observed in the agent's performance. The agent must learn to balance multiple, sometimes competing objectives, while also dealing with the potential loss of information due to reward normalisation. This complexity could explain the inconsistent performance across different KG completeness levels and the overall difficulty in improving performance over time.

Future iterations of this study might benefit from simplifying the reward function, focusing on fewer, more direct objectives, and carefully considering the trade-offs involved in reward normalisation. Additionally, techniques such as reward shaping or curriculum learning could be explored to help the agent better navigate the complex reward landscape.

5.6 Knowledge Graph Integration Effectiveness

The integration of KGs into the reinforcement learning framework was a key aspect of this study. This section examines the effectiveness of this integration by analyzing performance variations across KG completeness levels, discussing unexpected results, and exploring potential issues in KG information utilization.

5.6.1 Examination of Performance Variations Across KG Completeness Levels

The performance metrics across different KG completeness levels revealed a complex and non-linear relationship. The key findings are summarized in tables 5.1 and 5.2.

The variations suggest that the relationship between KG completeness and agent performance is not straightforward, and that increasing KG information does not necessarily lead to better performance.

5.6.2 Discussion on Unexpected Results with Increased KG Completeness

Several unexpected results emerged from the analysis:

Performance Degradation at Intermediate Levels: The agents with 50% and 75% KG completeness levels exhibited worse performance compared to those with 25% KG completeness. In the longer run, the mean efficiency at 25% KG completeness was 25.99%, whereas at 50% and 75% completeness it dropped to 3.16% and 1.59%, respectively. Similarly, the mean gap values were significantly higher at these intermediate KG levels, indicating less optimal path selection.

Higher Variability and Inconsistency: The standard deviations and discrepancies between mean and median values for efficiency and improvement metrics were larger at 50% and 75% KG completeness levels. This indicates greater inconsistency in performance, especially over extended training periods. For instance, the median improvement values were zero in the longer run for all KG completeness levels, suggesting that significant improvements were not consistently achieved.

Unexpectedly Better Performance with Minimal KG Information: Agents with 25% KG completeness consistently outperformed those with higher KG completeness levels in terms of efficiency and gap metrics. This suggests that minimal KG information may be more beneficial than partial or even full completeness in certain contexts.

These results challenge the intuition that more complete knowledge should lead to better and more consistent performance.

5.6.3 Potential Issues in KG Information Utilization in Decision-Making

Several factors may have contributed to the observed issues in KG information utilization:

Information Overload: As KG completeness increases, the agent may struggle to process and prioritize the growing amount of information effectively. This could lead to decision paralysis or suboptimal choices, particularly evident in the performance degradation at intermediate KG completeness levels.

Cognitive Overload with Partial Information: Partial KG information might introduce confusion, as the agent cannot form a complete understanding yet has more data to process than with minimal information. This could hinder the agent's ability to learn effective strategies and may explain the poorer performance at 50% and 75% KG completeness levels.

Graph Processing Limitations: The Graph Attention Network (GAT) used to process the KG may have difficulties effectively capturing and utilizing the complex relationships represented in more complete graphs. The increased complexity at higher KG completeness levels could overwhelm the GAT's capacity, leading to suboptimal feature extraction and decision-making.

State Representation Challenges: Integrating visual and graph-based inputs may have presented challenges in forming a coherent internal representation of the environment. The agent may have struggled to reconcile information from the CNN and GAT components, leading to poor decision-making even when more information was available.

Mismatch Between KG Structure and Task Requirements: The three-layer structure of the KG (terrain, entity, player) may not optimally represent the information needed for the specific task of optimizing routes between outposts. Critical information might have been obscured or diluted in the graph structure, making it difficult for the agent to extract actionable insights.

The only inter entity edges are between the agent and the other entities, which does not fully capture the relationships in the game world.

These potential issues highlight the challenges in effectively integrating KG in-

formation into the reinforcement learning process. The complex interaction between the KG structure, the agent's ability to process graph information, and the specific requirements of the task created a challenging learning environment.

5.6.4 Implications for Future Work

Future work could focus on several areas to address these challenges:

Refining KG Structure: Modifying the KG to better align with the task requirements may improve the agent's ability to utilize the information. This could involve restructuring the graph to emphasize critical path-planning information or simplifying the representation to reduce complexity.

Enhancing Graph Processing Capabilities: Improving the capacity and architecture of the GAT or exploring alternative graph neural network models may help the agent process more complex graphs effectively. Techniques such as hierarchical GNNs or incorporating attention mechanisms specifically designed for large graphs could be beneficial.

Improving State Representation Integration: Developing methods to better integrate visual and graph-based inputs might enhance the agent's internal representation of the environment. This could involve creating fusion layers that effectively combine features from both modalities.

Optimizing Subgraph Extraction: Enhancing the subgraph extraction process to include more relevant information for global path planning may improve the agent's decision-making. Techniques such as incorporating heuristics to select important nodes and edges could be explored.

Hyperparameter Tuning and Algorithm Adjustments: Conducting extensive hyperparameter tuning or exploring alternative reinforcement learning algorithms may address learning inefficiencies observed in the study. Algorithms specifically designed for environments with sparse rewards or long-term planning requirements might offer better performance.

By addressing these areas, future research could enhance the effectiveness of KG integration in reinforcement learning, leading to agents capable of better utilizing available knowledge to make optimal decisions in complex environments.

5.7 Variability and Consistency Issues

The experimental results reveal significant variability and consistency issues across different KG completeness levels. This section analyses the high standard deviations observed in the performance metrics, discusses the sensitivity to initial conditions and environment configurations, and explores the implications for model generalization and robustness.

5.7.1 Analysis of High Standard Deviations Across Metrics

The summary statistics reveal substantial variability in all performance metrics across KG completeness levels:

Efficiency: In the longer run (1,000,000 steps), standard deviations ranged from 4.66 (50% KG completeness) to 18.94 (25% KG completeness), with the highest variability observed at minimal KG completeness. This suggests that while minimal KG information can lead to higher efficiency, it also introduces greater unpredictability in performance.

Gap: Standard deviations were particularly high, ranging from 126.79 (100% KG completeness) to 3,707.59 (50% KG completeness). The extreme variability at 50% KG completeness suggests that partial information creates significant inconsistencies in the agent's ability to find optimal routes.

Improvement: Standard deviations for improvement values were also notable, ranging from 0.00 (all KG completeness levels in the longer run) to 527.28 (25% KG completeness in the shorter run). This indicates considerable inconsistency in learning progress across different runs, with improvement largely driven by outliers rather than consistent trends.

These high standard deviations suggest that the agent's performance is highly variable and inconsistent, with both minimal and partial KG completeness levels introducing substantial unpredictability in learning and decision-making.

5.7.2 Discussion on Sensitivity to Initial Conditions or Environment Configurations

The wide range of performance metrics suggests high sensitivity to initial conditions and environment configurations:

Extreme Outliers: Instances of exceptionally high efficiency (e.g., 25.99% at 25% KG completeness) and extremely large gaps (e.g., 1,360.78 at 50% KG completeness) indicate that certain initial conditions or environment configurations can lead to dramatically different outcomes. These outliers point to the agent's struggle with consistency across different environments.

Inconsistent Trends: The best and worst performances do not align consistently with specific KG completeness levels. For example, 25% KG completeness showed both the highest mean efficiency and one of the highest variability levels, suggesting that factors beyond KG completeness significantly influence performance.

Variable Learning Progress: Improvement values ranged from significant positive gains (e.g., 701.53 at 25% KG completeness in the longer run) to zero median improvement across all KG completeness levels, indicating that the agent's ability to learn and improve varied greatly across different runs. This variability may be due to differences in initial conditions, environment configurations, or how the agent processed available KG information.

These observations highlight that the agent's performance is heavily influenced by the specific configuration of each run, rather than being consistently determined by the level of KG completeness. The unpredictability suggests that the agent may be overfitting to certain scenarios or struggling with exploration in more complex configurations.

5.7.3 Model Generalization and Robustness

The observed variability and consistency issues have several implications for the model's generalization capabilities and overall robustness:

Limited Generalization: The high variability in performance metrics suggests that the model struggles to generalize across different environment configurations. The agent's inconsistent behavior at intermediate KG completeness levels (50% and 75%) highlights its difficulty in applying learned strategies across different scenarios.

Lack of Robustness: The presence of extreme outliers, such as large gaps and variable efficiency, indicates a lack of robustness in the model. A more robust model would demonstrate more consistent performance across various initial conditions and environment configurations, regardless of the amount of available information.

Overfitting to Specific Configurations: The wide range of performance outcomes, particularly at 50% and 75% KG completeness, suggests that the model may be overfitting to specific environment configurations. The high gap values at these levels indicate that the agent is not consistently finding optimal paths, further underscoring its difficulty in generalizing across different environments.

Challenges in Reliable Deployment: The inconsistency in performance poses significant challenges for reliably deploying the model in real-world scenarios or more complex game environments. Without predictable behaviour, the agent may not be able to perform well in unseen situations, reducing its practical utility.

Need for Improved Stability: The high variability across all performance metrics highlights the need for improved stability in the learning process. Techniques such as experience replay, better exploration strategies, or hierarchical reinforcement learning may help to mitigate this issue and create a more stable learning environment for the agent.

5.7.4 Summary of Implications

The substantial variability and sensitivity to initial conditions observed in this study reveal several critical challenges in developing a robust and generalizable reinforcement learning agent that effectively leverages KG information. The findings suggest that both minimal and partial KG information introduce significant unpredictability, and the agent's performance is highly dependent on specific environment configurations.

Future work should focus on:

- **Improving Model Robustness:** Developing methods to ensure more consistent performance across different environments and initial conditions, potentially through enhanced exploration strategies or improved use of KG information.
- **Refining KG Utilization:** Exploring techniques to better integrate and prioritize KG information within the agent's decision-making process, particularly at intermediate completeness levels.
- **Enhancing Stability:** Implementing techniques that stabilize learning, such as meta-learning or more sophisticated reward structures, to reduce the agent's sensitivity to variations in the environment and initial conditions.

By addressing these areas, future research can improve the generalization and reliability of agents using KG-enhanced reinforcement learning, paving the way for more robust applications in complex and dynamic environments.

6

Ethical Considerations

6.1 General Principles

The development and deployment of AI, both in gaming and other fields, must adhere to a comprehensive ethical framework that prioritizes human welfare and environmental sustainability. This section lays out the foundational ethical principles that guide AI development, ensuring its use aligns with broader societal and ecological goals. The ethical foundation focuses on long-term societal betterment, environmental responsibility, and ethical accountability.

6.1.1 Human-Centric Development

The primary objective of AI development should be the betterment of humanity. AI systems must be designed with the well-being of individuals and communities in mind. This principle requires that AI development not only optimizes for performance and efficiency but also considers the ethical implications of how the technology affects human lives. This includes ensuring that AI systems do not cause harm, perpetuate discrimination, or diminish human agency.

In the context of gaming, AI technologies should enhance player experiences without manipulating or exploiting them. AI-driven gaming systems, whether controlling non-player characters (NPCs) or influencing game mechanics, should contribute positively to users' well-being and personal growth. For instance, AI systems must be designed to foster positive social interactions and psychological development, rather than reinforcing harmful behaviours or encouraging over-reliance on the technology.

6.1.2 Environmental Stewardship

AI development, especially in computationally intensive applications like gaming, has a notable environmental footprint. High-performance AI models often require significant energy consumption, both during training and in real-time gameplay. As part of the ethical foundation, it is imperative that AI systems be developed with sustainability in mind, ensuring they minimize negative impacts on the environment.

AI developers must seek to reduce energy consumption through optimization techniques such as model compression, pruning, and the use of more energy-efficient hardware. Additionally, cloud-based AI solutions, which centralize computational demands, should be designed to minimize carbon emissions while balancing issues like latency and data privacy.

Incorporating environmental responsibility into AI development ensures that advancements in gaming technology do not come at the expense of ecological sustainability. Ethical AI practices should aim to strike a balance between technological innovation and environmental preservation, ultimately contributing to a more sustainable future.

6.1.3 Long-Term Ethical Accountability

AI systems, particularly those that evolve or learn over time, pose significant ethical challenges due to their potential long-term impacts on society. Developers must anticipate the far-reaching consequences of AI technologies and design these systems with long-term ethical accountability in mind.

In gaming, where AI-driven environments and NPCs can mirror real-world scenarios (e.g., digital twins), the responsibility for anticipating how these systems affect broader societal behaviours and norms becomes crucial. For example, if game AI systems are later adapted for use in real-world applications, developers must ensure that the ethical guidelines established in virtual settings translate effectively to physical environments.

As AI becomes more advanced and integrated into both gaming and broader societal applications, continuous ethical review processes must be in place. These processes should involve collaboration with ethicists, regulators, and other stakeholders to ensure AI systems remain aligned with societal values and do not lead to unintended harmful outcomes.

6.1.4 Inclusive Design and Equity

The ethical foundation for AI in gaming also mandates that the technology be inclusive, ensuring accessibility for a diverse range of players. AI systems must be designed to accommodate various player abilities, preferences, and cultural contexts, creating a gaming environment that is fair and engaging for all.

This principle of inclusivity also applies to the development side. AI research and game development should prioritize diversity in the teams that create these technologies. A diverse workforce is more likely to detect and mitigate biases in AI systems, leading to more equitable and representative gaming experiences.

Developers should proactively address socioeconomic disparities that may arise from AI integration in gaming, such as ensuring that AI-enhanced games are not limited to players with access to expensive hardware. By focusing on inclusivity and equity, AI can enhance the gaming industry while ensuring it remains accessible to players from all backgrounds.

6.1.5 Transparency and Informed Consent

AI systems in gaming must be transparent in their decision-making processes. Players should be informed about the capabilities and limitations of AI-driven elements within a game, such as NPCs or adaptive difficulty systems. Providing transparency builds trust between developers and users, allowing players to understand the role of AI in shaping their gaming experience.

Additionally, transparency is crucial for ethical accountability, enabling players and regulators to identify potential biases, errors, or unintended consequences in AI behaviour. For example, if an AI system is dynamically adjusting a game's difficulty, the criteria for those adjustments should be clear, and players should have the option to provide feedback or adjust the system if needed.

Informed consent becomes particularly important as AI systems collect and analyse player data to deliver personalized gaming experiences. Players must be made aware of how their data is used, what it contributes to the game's AI, and what privacy safeguards are in place. Ethical AI development in gaming, therefore, requires clear and accessible communication between developers and players regarding data usage and AI behaviours.

6.2 Ethical Safety and Transparency in AI Development

The integration of AI in gaming and its potential real-world applications presents numerous ethical challenges related to safety, transparency, and responsibility. As AI systems become more advanced and embedded in both virtual environments (such as gaming) and physical contexts (such as robotics or digital twins), ensuring robust ethical oversight becomes essential. This section outlines the ethical imperatives of safety, transparency, and accountability in AI development, with a focus on both gaming and real-world applications.

6.2.1 Safety in Virtual and Real-World AI Applications

Safety is a critical concern in AI development, particularly when systems transition from controlled virtual environments (e.g., video games) to more complex, unpredictable real-world applications. In gaming, AI systems often control non-player characters (NPCs), manage game environments, or adapt game difficulty based on player behaviour. In real-world applications, these systems can be used for robotics, autonomous vehicles, or simulations that directly impact physical environments.

6.2.1.1 Sim-to-Real Gap and Risk Management

The sim-to-real gap refers to the discrepancy between AI performance in a simulated environment (such as a video game) and its performance in the unpredictable, dynamic conditions of the real world. While gaming environments are controlled and bounded by the rules of the virtual world, real-world scenarios introduce levels of complexity that are not always captured in simulations. This can lead to unforeseen behaviours or safety risks when AI is applied outside its original domain.

To mitigate these risks, AI developers must incorporate robust safety mechanisms in both virtual and real-world systems. These mechanisms should include:

- Fail-safe protocols that allow AI to shut down or revert to safe states in case of errors or unpredictable behaviour.
- Adaptive safety features that enable AI to learn and respond to new or unforeseen situations safely.

- Continuous testing and validation of AI systems in real-world conditions to ensure that they operate reliably and ethically outside of controlled environments.

For example, if an AI system trained in a gaming environment is later used in robotics, it must be extensively tested to ensure it can handle real-world scenarios safely, particularly in edge cases where traditional simulations may fall short.

6.2.1.2 Accountability and Monitoring in AI Safety

Ensuring the safety of AI systems requires continuous oversight throughout their lifecycle. Developers must not only design AI systems to be safe, but also implement mechanisms for monitoring their performance after deployment. This is especially important in both gaming and real-world applications, where the system's impact may evolve as it interacts with users or complex environments.

AI systems in games, for example, might need to adjust NPC behaviours dynamically based on real-time player input. In these cases, ensuring that the AI operates safely and within ethical bounds requires constant monitoring and the ability to address any unexpected behaviours that arise during gameplay. Similarly, real-world AI applications must have accountability structures that allow for regular audits, transparency reports, and mechanisms for users or regulators to report malfunctions or ethical concerns.

6.2.2 Transparency in AI Decision-Making

Transparency is crucial for building trust between developers, users, and other stakeholders in both gaming and real-world applications of AI. AI systems, especially those that learn or adapt over time, often make decisions in ways that are not always easily understood by humans. This black-box problem can lead to confusion, mistrust, or ethical dilemmas when users are unaware of how an AI system arrives at its conclusions.

6.2.2.1 Explainable AI (XAI) in Games and Beyond

In gaming, transparency can significantly enhance player experience by providing insight into how AI-controlled NPCs or game systems operate. For example, players might benefit from understanding how an adaptive difficulty system adjusts the game based on their performance or how an NPC reacts to their choices in a branching narrative. This can also prevent frustration or confusion, as players are more likely to trust an AI-driven system when they understand the logic behind its actions.

Transparency can be achieved through Explainable AI (XAI) techniques, which aim to make the decision-making processes of AI systems more accessible and understandable to users. For example:

- Player dashboards or in-game tools could display metrics that show how NPCs or game systems are adjusting based on player input.
- Post-game reports might explain how an AI-driven system adjusted difficulty, providing players with feedback on their performance and how the AI responded.

In real-world applications, XAI becomes even more critical. Whether used in autonomous vehicles, healthcare, or robotics, AI systems must provide clear, interpretable explanations of their actions, particularly in high-stakes scenarios. This can help ensure accountability by making it easier to identify and address errors, biases, or unintended consequences.

6.2.2.2 Transparency in AI Ethics and Data Usage

Another key aspect of transparency involves informing users about how their data is being used by AI systems. In both gaming and real-world contexts, AI systems often rely on large datasets to train and improve their algorithms. In games, for example, data on player behaviour might be used to optimize game balance or improve NPC interactions. In real-world applications, this data might be used to fine-tune systems that affect public safety or personal privacy.

To ensure ethical data usage, developers must be transparent about:

- What data is being collected (e.g., player actions, in-game choices).
- How the data is used to improve AI models or personalize experiences.
- Who has access to the data and how it is protected from misuse or security breaches.

Players and users should be given informed consent, meaning they should clearly understand and agree to how their data will be used by AI systems. Providing users with options to opt-out or control certain aspects of data collection is also an important ethical consideration.

6.2.3 Responsibility in AI Development and Deployment

The ethical responsibility of AI developers extends beyond the immediate safety and transparency of the systems they create. It also involves ensuring that AI systems do not perpetuate harm, bias, or unfairness, whether in virtual or real-world environments.

6.2.3.1 Bias and Fairness in AI Systems

AI systems, whether used in gaming or real-world applications, can unintentionally reinforce biases present in the data they are trained on or the algorithms that power them. For example, an AI system trained on biased data may exhibit discriminatory behaviour in both game settings (e.g., biased NPC interactions) and real-world applications (e.g., biased hiring algorithms or law enforcement tools).

To address these concerns, developers must take proactive steps to identify and mitigate bias in AI systems. This includes:

Diverse and representative data: Ensuring the training data used for AI systems reflects a wide range of experiences, perspectives, and identities. Bias auditing tools: Implementing tools that can detect and correct biased outcomes in AI decision-making processes. Inclusive design practices: Ensuring that development teams are diverse and inclusive, as this helps identify potential biases and design systems that are fairer and more equitable.

6.2.3.2 Ethical AI in the Gaming Industry

In the context of gaming, developers have a unique responsibility to ensure that AI-driven systems do not exploit players. For instance, adaptive AI that adjusts game difficulty or recommends in-game purchases should be designed in a way that maintains fairness and does not manipulate players into excessive spending or engagement.

Ethical responsibility also extends to the potential social impacts of AI systems. In multiplayer games, AI-driven systems should encourage positive social interactions and avoid fostering toxic behaviours. Similarly, AI companions in solo play should be designed to enhance player enjoyment and engagement without diminishing the value of human social interactions in cooperative settings.

6.2.4 Conclusion

Ethical safety, transparency, and responsibility are fundamental pillars of AI development in both gaming and real-world applications. As AI systems become more complex and integrated into society, it is crucial that developers build transparent, accountable, and safe systems. This involves ensuring that AI systems can be trusted to operate ethically, both in virtual environments like gaming and in real-world scenarios where the consequences of failure or bias can be far-reaching. By prioritizing safety, transparency, and responsibility, the gaming industry—and AI development as a whole—can align technological progress with ethical imperatives.

6.3 Adaptive AI, Accessibility, and Inclusion

As AI systems in gaming evolve, one of their most significant contributions lies in the potential to adapt to players' abilities, preferences, and individual needs. Adaptive AI offers the ability to create personalized gaming experiences, making games more accessible and inclusive for a wide range of players. However, this adaptability must be carefully designed to strike a balance between assisting players and preserving the core challenges that make games engaging. This section explores the role of adaptive AI in promoting accessibility, fostering inclusion, and ensuring a balanced player experience.

6.3.1 Adaptive AI for Personalized Gameplay

Adaptive AI refers to the ability of AI systems to adjust dynamically to individual player performance, preferences, and behaviour. By learning from a player's actions, adaptive AI can tailor the gaming experience in real time, creating a personalized interaction that accommodates each player's unique skills and play style. This allows games to be more flexible and enjoyable for players with different abilities, whether they are casual gamers or highly skilled players.

6.3.1.1 Dynamic Difficulty Adjustment (DDA)

One of the most common uses of adaptive AI is DDA, where the game's AI dynamically alters the difficulty based on a player's performance. For instance, if a player is consistently struggling with a certain level or mechanic, the AI can reduce the difficulty by making enemies less aggressive or providing more health packs. Conversely, if a player is breezing through challenges, the AI can introduce more difficult obstacles to maintain a sense of challenge.

Advantages of DDA:

- **Inclusive Gameplay:** DDA allows players of varying skill levels to engage with the same game content, making games more accessible to a wider audience without the need for explicit difficulty settings.
- **Player Retention:** By adapting to players' abilities, games can prevent frustration or boredom, ensuring that players remain engaged and feel a sense of progression.

However, there are challenges associated with implementing DDA effectively. If the AI overcompensates by making the game too easy or too difficult, it can diminish the player's sense of accomplishment or engagement. Striking the right balance between support and challenge is key to maintaining a satisfying gaming experience.

6.3.1.2 Adaptive NPCs and Player Interaction

Adaptive AI can also influence how non-player characters (NPCs) behave, responding to individual player preferences and interactions. For example, NPCs in role-playing games can adjust their dialogue, reactions, or quest-giving behaviours based on a player's previous choices or gameplay style. This can create more immersive and engaging experiences, as players feel that the game world is reacting uniquely to their actions.

Benefits of Adaptive NPCs:

- **Enhanced Immersion:** Players feel more connected to the game world when NPCs respond in nuanced ways to their choices, creating a more personalized narrative.
- **Inclusive Storytelling:** Adaptive NPCs can ensure that players with different play styles—whether combat-focused or story-driven—experience a game tailored to their interests.

6.3.2 Enhancing Accessibility through AI

Accessibility in gaming has become an important focus, as developers strive to make games more inclusive for players with disabilities or those who may face barriers to traditional game mechanics. Adaptive AI plays a crucial role in enhancing accessibility, offering ways for games to adjust to the physical, sensory, or cognitive needs of players.

6.3.2.1 AI for Players with Physical Disabilities

For players with physical disabilities, adaptive AI can assist by reducing the physical demands of gameplay. AI systems can modify controls or gameplay mechanics to

accommodate different levels of dexterity or mobility, making games more accessible without compromising the overall experience.

For example:

- **Simplified control schemes** can be dynamically implemented based on the player's input, allowing those with limited mobility to perform complex actions with fewer inputs.
- **Input prediction and assistance** powered by AI can help players execute actions in-game more easily, such as by predicting intended moves or automating complex sequences.

6.3.2.2 AI for Cognitive and Sensory Needs

AI systems can also adapt gameplay for players with cognitive or sensory needs, offering modifications that make games more accessible for individuals with conditions like dyslexia, attention disorders, or visual or auditory impairments.

For example:

- **Text-to-speech systems**, driven by AI, can read in-game dialogue or menus aloud for players with visual impairments.
- **Simplified interface design** can be activated for players who need less visual clutter or fewer cognitive distractions during gameplay.
- **AI-driven visual and audio cues** can help players with auditory or visual impairments navigate game environments, such as offering alternative feedback for sound-based mechanics through vibrations or on-screen prompts.

The adaptability of AI ensures that players with various accessibility needs can enjoy the same core gameplay experience, while also having features tailored specifically to their abilities.

6.3.3 Fostering Inclusivity through Customization

Beyond accessibility, adaptive AI can foster inclusivity by allowing games to be customized based on player preferences and cultural contexts. This personalization goes beyond difficulty and accessibility, focusing on making the game experience relevant and engaging for players from diverse backgrounds.

6.3.3.1 Cultural Adaptation and Representation

Games can be more inclusive when they reflect the cultural diversity of their players. Adaptive AI systems can modify game content to better represent players' cultural backgrounds, providing personalized experiences that respect and celebrate diversity. For example, AI-driven systems could adjust dialogue, in-game references, or character customization options to reflect players' linguistic or cultural preferences.

This type of customization is particularly important in global gaming markets, where players from different regions may have different expectations or norms around game content. AI can help bridge cultural gaps by making games feel more relevant and tailored to diverse audiences. Player-Centric Storytelling

AI systems can also create more inclusive and immersive experiences by tailoring stories and game interactions to a player's identity and play style. This includes:

- **Personalized narratives** that adjust based on player choices, allowing players to explore storylines that reflect their individual values, preferences, and interests.
- **Inclusion of diverse characters** and representation of different gender identities, ethnicities, and social backgrounds in ways that are meaningful to the player.

Through this level of customization, AI can ensure that all players feel represented and included in the game world, enhancing the overall sense of immersion and belonging.

6.3.4 Balancing Adaptation with Challenge

While adaptive AI offers tremendous potential for making games more inclusive and accessible, it is also essential to maintain a balance between providing assistance and preserving the challenge that is central to gaming. Many players derive satisfaction from overcoming difficult tasks, learning new skills, and achieving mastery through repeated effort. If AI systems overcompensate by making games too easy, players may feel disengaged or lose their sense of achievement.

6.3.4.1 Preserving Engagement Through Adaptive AI

A key challenge in designing adaptive AI systems is ensuring that they help players without reducing the inherent difficulty or enjoyment of the game. This balance can be maintained by:

- **Gradual adaptation:** Rather than making immediate and drastic changes, AI systems can adjust difficulty in more subtle, gradual ways that ensure players still feel challenged.
- **Player feedback mechanisms:** Allowing players to provide feedback on how they perceive the game's difficulty or assistance level ensures that AI systems adjust according to the player's preferences, keeping the game challenging but not frustrating.
- **Adaptive skill progression:** AI can dynamically adjust the complexity of in-game challenges to align with the player's growth in skill, ensuring that players experience a sense of mastery and accomplishment as they progress.

Maintaining a balance between adaptation and challenge is crucial in delivering an engaging and fulfilling gaming experience that caters to diverse audiences without diminishing the essence of the game.

6.3.5 Conclusion

Adaptive AI has the potential to revolutionize gaming by enhancing accessibility, inclusivity, and personalization for players of all backgrounds and abilities. By dynamically adjusting to individual player needs, adaptive AI systems can make games more engaging and accessible, while also fostering a sense of inclusivity by reflecting diverse cultural contexts and personal preferences.

However, careful design is required to ensure that these adaptive systems maintain the balance between providing necessary assistance and preserving the core

challenges that define great gameplay. When executed thoughtfully, adaptive AI can offer a richer, more inclusive gaming experience for all players.

6.4 Technological and Socioeconomic Impact

6.4.1 Technological and Socioeconomic Impact

The rapid advancement of AI in gaming introduces significant technological and socioeconomic challenges, particularly in terms of computational demands, environmental sustainability, and economic disparities. AI-driven games, especially those employing machine learning models for real-time interaction, require powerful hardware and substantial computational resources. These requirements have broader implications for both the environment and the accessibility of such games across different socioeconomic groups. This section explores the computational and hardware requirements, environmental impact, and economic disparities associated with the integration of AI in gaming.

6.4.2 Computational and Hardware Requirements

AI in gaming, particularly in areas like adaptive NPC behaviour, procedural content generation, and real-time decision-making, relies heavily on computational power. Training and running AI models, especially those based on deep learning algorithms, necessitate high-performance hardware, such as powerful graphics processing units (GPUs) or tensor processing units (TPUs).

6.4.2.1 AI Training and Real-Time Inference

Training AI models is an inherently resource-intensive process, requiring large datasets and considerable computational power. In the context of gaming, AI models must often be trained to perform a wide range of tasks, such as learning player behaviours, creating adaptive narratives, or dynamically adjusting difficulty levels. This training phase often takes place in data centres equipped with specialized hardware, which requires significant energy consumption and computational resources.

Once these models are trained, real-time inference—where the AI makes decisions during gameplay—places additional demands on the hardware. Players, particularly those using high-performance AI-driven games, may need expensive, state-of-the-art gaming systems to fully experience the capabilities of advanced AI. This dependency on high-end hardware can create barriers to entry for many players, especially those in lower-income brackets, and can also affect independent developers who may lack the resources to implement such AI systems effectively.

6.4.2.2 Model Compression and Optimization

To address the high computational requirements of AI-driven games, several techniques can be employed to reduce the processing load. These include:

- **Model compression:** Reducing the size of AI models by removing redundant layers or parameters, which can significantly lower the computational power required.
- **Quantization:** Reducing the precision of the data used in AI models, which can reduce computational costs without significantly affecting performance.
- **Edge computing:** Offloading some of the computational tasks to local hardware (like the player's gaming console or PC) to reduce the load on centralized servers.

While these techniques can help optimize performance, they also introduce trade-offs in terms of accuracy and responsiveness, particularly for games that rely on real-time interaction. Developers must carefully balance computational efficiency with the quality of the AI-driven experiences they aim to deliver.

6.4.3 Environmental Impact

The growing demand for high-performance AI in gaming has significant environmental consequences. The energy consumption associated with training and running AI models, combined with the production and disposal of powerful hardware, contributes to a substantial carbon footprint. As gaming increasingly adopts AI-driven systems, addressing the environmental impact of these technologies becomes critical.

6.4.3.1 Energy Consumption in AI Development

Training large AI models, especially those used in cutting-edge games, requires substantial computational power. This often leads to the deployment of high-performance data centres, which consume vast amounts of energy to train and maintain AI systems. The computational resources needed for AI models in gaming contribute to rising carbon emissions, especially when powered by non-renewable energy sources.

Additionally, AI models need to be retrained periodically as new data becomes available or as the game evolves. This continuous training cycle further exacerbates the energy demands of AI development. While some data centres have adopted renewable energy sources to mitigate their environmental impact, the overall carbon footprint of AI remains a significant concern.

6.4.3.2 Hardware Manufacturing and E-Waste

The hardware required to run AI-driven games efficiently—such as GPUs, gaming consoles, and custom-built PCs—presents additional environmental challenges. The manufacturing of these devices requires mining rare materials and metals, which is energy-intensive and contributes to environmental degradation.

Moreover, as gaming hardware becomes obsolete more quickly due to the rapid pace of technological advancement, the issue of electronic waste (e-waste) becomes more pressing. Players and developers frequently upgrade their systems to accommodate new AI capabilities, contributing to the growing problem of e-waste. Efforts to create more energy-efficient hardware or improve recycling processes for outdated devices are essential to mitigate this environmental impact.

6.4.3.3 Potential Solutions for Sustainability

Several strategies can be employed to reduce the environmental footprint of AI in gaming:

- **Cloud computing:** By offloading computationally intensive tasks to the cloud, players can access AI-driven experiences without requiring high-end hardware. Centralized cloud servers can also be optimized for energy efficiency.
- **Energy-efficient hardware:** Developing more sustainable hardware, such as energy-efficient GPUs or consoles, can help reduce the overall power consumption of AI-driven games.
- **Green AI initiatives:** These initiatives focus on reducing the environmental impact of AI through the use of renewable energy sources, model optimization techniques, and carbon offset programs.

While these solutions present viable paths toward a more sustainable future for AI in gaming, their implementation requires collective action from developers, hardware manufacturers, and policymakers.

6.4.4 Economic Disparities

The integration of AI in gaming can widen the gap between players with access to high-end hardware and those who lack the financial resources to invest in such technology. As AI systems become more sophisticated and hardware requirements increase, players from lower-income backgrounds may find themselves unable to access the latest AI-driven games. This economic disparity can also extend to game developers, especially smaller, independent studios, who may struggle to implement advanced AI systems due to resource constraints.

6.4.4.1 Access to AI-Driven Games

As gaming increasingly incorporates advanced AI features, there is a risk of creating a two-tiered system where only wealthier players can afford the necessary hardware and internet infrastructure to access these games. AI-driven games often require high-performance graphics cards, fast processors, and ample storage—components that can be prohibitively expensive for many gamers. This digital divide may limit access to the most innovative gaming experiences for a significant portion of the player base.

Additionally, the cost of internet access—particularly the need for high-speed connections to support cloud-based AI gaming—can further exacerbate economic disparities. In regions with limited access to high-speed internet, players may be unable to engage with AI-enhanced gaming experiences, creating unequal access across different geographic and socioeconomic groups.

6.4.4.2 Impact on Game Developers

Smaller, independent game developers face economic challenges when trying to incorporate AI into their games. Developing, training, and running AI models requires

not only significant computational resources but also expertise in machine learning and AI, which may be beyond the reach of smaller studios. As a result, AI innovation in gaming may become concentrated within larger, well-funded studios, leaving independent developers at a disadvantage.

This disparity could result in a concentration of AI-driven innovation within major gaming companies, potentially stifling creativity and diversity within the industry. Smaller developers may struggle to compete with larger studios that have the financial means to invest in cutting-edge AI technology, leading to fewer opportunities for independent studios to contribute to AI advancements in gaming.

6.4.4.3 Potential Solutions to Address Economic Disparities

To mitigate the economic disparities created by AI in gaming, several strategies can be considered:

- **Cloud-based gaming platforms:** By offering AI-driven gaming experiences through cloud-based platforms, developers can reduce the hardware requirements for players, making these games more accessible to a wider audience. Players would no longer need to invest in expensive hardware to access the latest AI innovations, as the computational tasks would be handled by cloud servers.
- **Open-source AI tools:** Providing open-source AI frameworks and tools can help democratize access to AI technology, allowing smaller developers to implement AI systems without the need for extensive resources or expertise.
- **Partnerships and funding for smaller studios:** Larger companies or industry organizations could create programs to support independent developers in adopting AI technologies, providing them with resources, training, and funding to develop AI-driven games.

Addressing the economic disparities in AI-driven gaming is essential to ensuring that the benefits of AI advancements are distributed more equitably across both players and developers, regardless of their financial resources.

6.4.5 Conclusion

The technological and socioeconomic impact of AI in gaming presents both challenges and opportunities. The computational and hardware demands of AI systems, combined with the environmental consequences of energy consumption and e-waste, highlight the need for more sustainable development practices. At the same time, the growing disparity between players with access to high-end hardware and those without underscores the importance of finding solutions to bridge the economic divide in gaming. By optimizing AI models, developing more energy-efficient hardware, and fostering inclusive access to AI-driven games, the gaming industry can advance technological innovation while promoting environmental responsibility and social equity.

6.5 Future Directions and Challenges

The integration of AI into gaming is poised to significantly shape the future of the gaming industry and society at large. As AI technologies become more sophisticated, the long-term impacts and ethical challenges associated with their use must be carefully considered. These challenges extend beyond the gaming experience, influencing social dynamics, cognitive behaviours, and broader ethical frameworks. This section explores the future directions of AI in gaming and the associated ethical challenges that arise as the technology evolves.

6.5.1 Long-Term Societal Impacts

AI in gaming has the potential to influence societal norms and behaviours, not just within the gaming environment but in real-world contexts as well. The increasing realism and complexity of AI-driven systems can blur the boundaries between virtual and real-world experiences, leading to significant shifts in how people interact with both AI and each other.

6.5.1.1 Shifting Social Interactions

One of the most profound impacts AI may have in gaming is the way it reshapes social interactions. As AI-controlled non-player characters (NPCs) become more sophisticated and capable of mimicking human-like behaviours, players may develop deep emotional connections with these AI entities. While this can enhance immersion and player engagement, it also raises concerns about the potential for social isolation.

Players may become accustomed to engaging with AI companions in ways that replace or diminish real-world social interactions. As AI-driven characters in games become more empathetic and responsive, they could reduce the incentive for players to seek human companionship or socialization. This could have long-term effects on social behaviours, particularly among younger players, who may spend extended periods interacting with AI entities rather than real people.

6.5.1.2 Impact on Cognitive Skills and Problem-Solving

Another long-term consequence of AI in gaming is its influence on cognitive development and problem-solving skills. AI-driven games can offer players complex, dynamic challenges that require adaptive thinking and strategic planning. These experiences may enhance certain cognitive skills, such as pattern recognition, decision-making, and situational awareness.

However, the reliance on AI-driven systems to guide or assist players through these challenges could also diminish opportunities for players to independently develop these skills. For example, games that adjust difficulty in real-time or provide AI-assisted solutions may reduce the need for players to experiment, fail, and learn through trial and error. This could have broader implications for how people approach problem-solving and decision-making in real-world situations.

6.5.1.3 AI as a Tool for Education and Training

Looking ahead, the gaming industry's integration of AI could extend beyond entertainment into education and training. AI-driven games could be used as simulation tools to train individuals in various fields, such as medical procedures, urban planning, or military strategy. By using AI to create realistic and adaptive virtual environments, gaming platforms could serve as a powerful tool for skill development and professional training.

While this opens up new possibilities for gamified learning and training environments, it also raises questions about the transferability of skills learned in these settings to real-world applications. Developers and educators will need to ensure that the skills learned in AI-driven simulations are applicable in practical contexts, and not limited to the virtual scenarios in which they were developed.

6.5.2 Ethical Challenges

As AI continues to evolve, new ethical challenges emerge that must be addressed to ensure the responsible development and use of AI in gaming. These challenges encompass issues such as data privacy, player manipulation, bias, and the psychological impact of interacting with lifelike AI entities.

6.5.2.1 Data Privacy and Security

The personalization of AI-driven gaming experiences often relies on extensive data collection. AI systems collect and analyse player behaviour, preferences, and interactions to optimize gameplay and deliver personalized content. While this can enhance the gaming experience, it also raises significant privacy concerns.

Players may not always be fully aware of the extent to which their data is being collected or how it is being used. This lack of transparency poses ethical concerns, particularly if data is used for purposes beyond gameplay, such as targeted advertising or sold to third parties. Ensuring informed consent and implementing strict data protection measures are critical to addressing these challenges.

Moreover, the integration of AI systems with cloud gaming platforms means that large amounts of personal data are stored on external servers. This raises questions about data security, particularly in regions with less robust privacy protections. Developers and platform providers must prioritize the protection of sensitive player information, ensuring that data is encrypted, anonymized, and securely stored.

6.5.2.2 Player Manipulation and Addiction

The adaptive nature of AI in gaming has the potential to create more immersive and personalized experiences, but it also opens the door to player manipulation. AI systems can be designed to track and predict player behaviour in ways that may encourage excessive gaming or in-game spending. For example, AI could identify moments of player vulnerability—such as frustration after repeated failures—and strategically offer in-game purchases or rewards to keep the player engaged.

This type of manipulation raises ethical concerns, particularly when it comes to vulnerable populations, such as younger players or those prone to addictive be-

haviours. Developers must be cautious not to exploit players through AI-driven systems that encourage compulsive gaming or spending. Implementing ethical guidelines around the use of AI for in-game purchases and engagement strategies will be essential to mitigate these risks.

6.5.2.3 Bias and Fairness in AI Systems

Bias in AI systems is a well-documented issue across many fields, and gaming is no exception. AI systems used in games, especially those that personalize experiences for players, may unintentionally reinforce cultural, racial, or gender biases present in the data they are trained on. This can manifest in various ways, from biased NPC interactions to unequal representation of diverse characters in game narratives.

As AI-driven games continue to grow in complexity, developers must prioritize fairness and inclusivity in the design of these systems. This involves not only ensuring that the training data is representative of diverse populations but also implementing mechanisms to audit AI behaviour for bias. Additionally, the development of inclusive game narratives and characters that reflect a wide range of cultural and social experiences will be critical to addressing these ethical concerns.

6.5.2.4 Psychological Impact of Lifelike AI

As AI systems in gaming become more advanced, they are likely to create characters and environments that are increasingly indistinguishable from real life. This raises questions about the psychological impact of interacting with highly realistic AI entities, particularly in terms of empathy and emotional well-being.

On one hand, AI-driven characters that exhibit empathy and responsiveness could help players develop emotional intelligence and social skills. On the other hand, prolonged interaction with AI entities may desensitize players to human emotions or reduce their ability to engage meaningfully with real-world relationships. Additionally, the immersion provided by lifelike AI entities could lead to players becoming overly attached to virtual companions, potentially leading to emotional distress when those relationships are severed.

Understanding the psychological effects of interacting with lifelike AI entities will require ongoing research, particularly as AI becomes more integrated into daily life through gaming and other applications.

6.5.3 Regulatory and Ethical Oversight

The ethical challenges presented by AI in gaming necessitate a framework for regulatory and ethical oversight. As AI continues to evolve and its applications in gaming become more widespread, it will be essential to establish clear guidelines and standards to ensure responsible innovation.

6.5.3.1 Ethical Frameworks for AI in Gaming

Developers, researchers, and policymakers must collaborate to create ethical frameworks that address the unique challenges posed by AI in gaming. These frameworks

should include guidelines for:

- **Data privacy and consent:** Ensuring that players are fully informed about how their data is collected, used, and protected.
- **Fairness and bias mitigation:** Implementing tools and strategies to detect and correct biases in AI-driven systems.
- **Transparency:** Requiring AI developers to provide clear explanations of how AI systems operate and how they influence gameplay.
- **Psychological safeguards:** Establishing ethical guidelines to prevent player manipulation and mitigate the potential for addiction or emotional harm.

6.5.3.2 Collaborative Research and Development

The future of AI in gaming will also require greater collaboration between developers, ethicists, psychologists, and policymakers. This interdisciplinary approach will help ensure that the development of AI systems is guided by ethical considerations and that the technology is used in ways that promote social well-being rather than exacerbating existing problems.

6.5.3.3 Conclusion

As AI in gaming continues to advance, it will bring with it both unprecedented opportunities and significant ethical challenges. The long-term societal impacts of AI, including shifts in social interaction and problem-solving, must be carefully monitored. Ethical challenges related to data privacy, player manipulation, bias, and the psychological effects of interacting with lifelike AI entities will need to be addressed through regulatory frameworks and responsible development practices. By proactively tackling these issues, the gaming industry can harness the power of AI to create innovative, immersive experiences while upholding ethical standards that prioritize player well-being and societal progress.

7

Conclusion

The study aimed to assess the impact of varying KG completeness levels on the performance of NPCs. Additionally, the effectiveness of a CNN-GAT hybrid model in processing both visual and graph-based inputs was evaluated. Key performance metrics, including efficiency, gap, and improvement, were analysed to understand the model's effectiveness. Experimental Setup

A custom-built game environment was designed, featuring procedurally generated terrain to simulate a TSP. The experiment was conducted with different KG completeness levels (25%, 50%, 75%, and 100%), and the agents were trained using the Proximal Policy Optimization (PPO) algorithm.

7.1 Key Performance Metrics and Summary Statistics

Three primary metrics were used to evaluate the performance of the NPCs across different KG completeness levels:

- **Efficiency**: This metric represents the ratio of the optimal path length to the agent's actual path length, expressed as a percentage. Higher efficiency indicates that the agent's path was closer to the optimal one.
- **Gap**: The gap is the absolute difference between the length of the agent's path and the optimal path length. A smaller gap reflects better performance, as the agent's path deviates less from the optimal solution.
- **Improvement**: This metric measures the relative change in the agent's performance over time, comparing previous and current efficiency. It is calculated as shown in subsection 5.5.1, and reflects how the agent's decision-making evolves through training.

Two sets of summary statistics were collected in tables for **100,000 steps** for the short run (5.1) and **100,000,000 steps** for the long run in (5.2) across the different KG completeness levels: **25%**, **50%**, **75%**, and **100%**.

In the short run (Table 1), the **25% KG completeness level** showed the highest efficiency, with a mean of 8.96%, while intermediate levels of KG completeness (50% and 75%) exhibited poorer performance, with lower efficiency and larger gaps. **Improvement** metrics were similarly variable, with the 25% level showing notable improvements in some cases, but higher KG levels demonstrated greater instability.

In the long run (Table 2), a similar trend emerged. **25% KG completeness** again outperformed the higher completeness levels, achieving a mean efficiency of 25.99%. In contrast, the **50% and 75% KG completeness** levels showed signifi-

cantly lower efficiencies and larger gaps, indicating that partial knowledge from the KG may hinder the agent's ability to plan effectively over extended training periods.

7.2 Performance Across Different KG Completeness Levels

At **25% KG completeness**, the agent exhibited the highest performance across all key metrics. The mean efficiency in the long run reached **25.99%**, outperforming the higher KG completeness levels. **Improvement** metrics were also notably higher, particularly in the long run, where significant gains were observed over time.

Minimal knowledge from the KG allows the agent to focus on more immediate, actionable information, leading to better decision-making. The agent's performance benefited from a simplified knowledge structure, improving efficiency and adaptability.

Figures illustrate the agent's steady improvement in efficiency and its relatively low gap across both short and long runs, reinforcing the effectiveness of reduced KG completeness.

Performance at **50% KG completeness** experienced a **significant drop**, especially in the long run. The mean efficiency fell to **3.16%**, and gap metrics increased substantially, with agents deviating further from the optimal path. Improvement values were minimal, indicating the agent struggled to leverage the partial knowledge effectively.

Partial knowledge introduces unnecessary complexity without providing substantial benefit. The agent appears unable to synthesize fragmented information, resulting in poorer performance and decision-making.

Figures show the steep decline in efficiency over time, alongside increasing gaps, highlighting the challenges faced by the agent at this level of completeness.

Similar to the **50% KG completeness** level, performance at **75% KG completeness** was marked by **high variability** and low efficiency. The mean efficiency in the long run was **1.59%**, with the gap continuing to widen as the agent failed to effectively plan optimal paths. Improvement metrics were inconsistent, further underscoring the learning instability.

Increasing KG data beyond 50% results in further confusion for the agent. The additional knowledge overwhelms the agent's decision-making process, leading to erratic and inefficient behaviour.

Figures reveal high fluctuations in both efficiency and improvement metrics, indicating significant instability throughout the training process.

At **100% KG completeness**, the agent's performance was moderate in the short run, with a mean efficiency of **6.84%**. However, in the long run, efficiency dropped to **3.62%**, with the agent showing signs of difficulty in maintaining consistent performance over time. While gap metrics were smaller than those at 50% and 75% completeness, the overall efficiency remained lower than at **25% KG completeness**.

Full knowledge does not guarantee optimal performance. Although the agent had access to complete information, it struggled to translate this into effective decision-

making over extended training periods.

Figures show initial improvement in the short run, followed by a levelling off and decline in efficiency over the long run, highlighting the limitations of full knowledge.

7.2.1 Information Overload

At higher levels of KG completeness (50%, 75%, and 100%), the agent likely experienced information overload, where the abundance of data became too difficult to process effectively. This excessive data introduced unnecessary complexity, causing the agent to struggle with decision-making. As a result, decision paralysis may have occurred, where the agent was overwhelmed by the volume of information, leading to suboptimal path planning and performance, particularly in the long run. Integration of Visual and Graph-based Inputs

The hybrid CNN-GAT model, designed to process both visual and graph-based inputs, may not have fully reconciled these two streams of data. The disparity between processing visual inputs (from the game environment) and structured knowledge from the KG could have led to suboptimal decision-making, as the agent struggled to integrate the two sources of information efficiently. This misalignment likely contributed to poor performance at higher KG completeness levels.

7.2.2 Learning Algorithm Limitations

The Proximal Policy Optimization (PPO) algorithm, while effective in many reinforcement learning scenarios, may have shown limitations in this specific setup. The high complexity of integrating KGs with RL may require further hyperparameter tuning or more specialized algorithms to fully unlock the agent's potential. The lack of optimisation could have hindered the agent's learning process, contributing to learning instability and inefficiencies observed across different KG completeness levels.

7.3 Key Findings and Future Directions

7.3.1 Limitations of Current Approach

This research revealed significant limitations in the integration of Knowledge Graphs with model-free Reinforcement Learning:

- **Performance Issues:** While 25% KG completeness showed better results than higher completeness levels, overall performance remained suboptimal across all configurations, indicating fundamental limitations in the current approach.
- **Integration Challenges:** The model-free PPO algorithm struggled to effectively utilize the KG's symbolic representation, suggesting that the lack of an internal predictive model may be a crucial missing component.
- **Information Processing:** Partial knowledge (50% and 75%) introduced unnecessary complexity, leading to significant performance degradation. The

agent struggled to process incomplete information effectively, resulting in lower efficiency and higher variability in decision-making.
- **Architectural Mismatch:** The combination of PPO with KG demonstrated that simply providing structured knowledge to a model-free RL algorithm is insufficient for improving decision-making capabilities.

7.3.2 Critical Insights and Improvements

The results strongly suggest several areas for improvement:
- **KG Structure and Processing:**
  - Simplify graph structure to reduce information overload
  - Restructure to highlight critical path-planning relationships
  - Explore hierarchical graph networks or transformer-based models
  - Improve encoding strategies for more nuanced environmental understanding
- **State Representation:**
  - Implement fusion layers for better visual and graph-based information integration
  - Incorporate richer data fusion techniques
  - Use attention mechanisms to dynamically weigh input importance
  - Develop better methods for subgraph extraction based on agent goals
- **Model-Based Integration:**
  - Explore model-based RL algorithms that can leverage both predictive capabilities and symbolic representation
  - Develop architectures that bridge symbolic and subsymbolic representations
  - Focus on complementary representation approaches rather than treating KG as just an input source

7.3.3 Future Research Directions

Based on these findings, several critical areas for future research emerge:
- **Algorithm Development:**
  - Investigate model-based RL algorithms specifically designed for KG integration
  - Develop methods for maintaining consistency between different knowledge forms
- **Architectural Improvements:**
  - Design new architectures that effectively bridge symbolic and predictive models
  - Implement more sophisticated data fusion techniques
  - Develop better methods for dynamic knowledge updating
- **Scalability and Performance:**
  - Research efficient methods for maintaining and updating both KG and predictive models
  - Develop techniques for managing computational complexity

- Create approaches for handling increasing environmental complexity

7.3.4 Implications for the Field

While the current implementation did not achieve optimal performance, this research provides valuable insights:

- **Knowledge Integration:** The finding that minimal, well-structured knowledge (25% KG completeness) outperformed higher completeness levels emphasizes that how knowledge is represented and utilized is more important than the amount of knowledge available.
- **Learning Framework:** The results indicate that a more comprehensive learning framework is needed that can effectively combine symbolic and predictive modeling approaches.
- **Future Applications:** These findings contribute to the broader question of enhancing NPC adaptability through KG-RL integration, with implications for developing more sophisticated AI systems in dynamic, complex environments.

It must be noted that this framework needs to be tested using existing model-based algorithms to better understand the direction of future research and validate these proposed improvements.

To facilitate further exploration of these research directions and encourage collaborative advancement in the field, the complete implementation of this framework has been made publicly available. This includes the custom game environment, model architectures, and training pipelines.

By sharing these resources with the research community, I hope to enable other researchers to build upon this work, test alternative approaches, and contribute to the ongoing development of more effective KG-RL integration methods for NPC development. I am committed to continuing this research and developing the framework further, and welcome collaboration from others interested in advancing this field.

Bibliography

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," *ArXiv*, vol. abs/1912.10944, 2019. [Online]. Available: `https://arxiv.org/pdf/1912.10944.pdf`.

[3] C. Hesse, "Learning montezuma's revenge from a single demonstration," OpenAI, Tech. Rep., 2018. [Online]. Available: `https://openai.com/index/learning-montezumas-revenge-from-a-single-demonstration/`.

[4] Y. Li, Z. Li, P. Wang, *et al.*, "A survey of graph meets large language model: Progress and future directions," *arXiv preprint arXiv:2311.12399*, 2023. [Online]. Available: `https://arxiv.org/pdf/2311.12399v1.pdf`.

[5] N. Parikh, Z. Horvitz, N. Srinivasan, A. Shah, and G. D. Konidaris, *Graph embedding priors for multi-task deep reinforcement learning*, 2020. [Online]. Available: `http://irl.cs.brown.edu/pubs/graph_embed_deeprl_ws.pdf`.

[6] Z. Zhang, F. Zhuang, H. Zhu, *et al.*, "Towards robust knowledge graph embedding via multi-task reinforcement learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021. [Online]. Available: `https://arxiv.org/pdf/2111.06103.pdf`.

[7] A. S. Vezhnevets, S. Osindero, T. Schaul, *et al.*, "Feudal networks for hierarchical reinforcement learning," in *International Conference on Machine Learning*, PMLR, 2017, pp. 3540–3549. [Online]. Available: `https://arxiv.org/pdf/1703.01161.pdf`.

[8] S. Devlin, R. Georgescu, I. Momennejad, *et al.*, *Navigation turing test (ntt): Learning to evaluate human-like navigation*, 2021. arXiv: `2105.09637 [cs.AI]`. [Online]. Available: `https://arxiv.org/pdf/2105.09637.pdf`.

[9] D. Kahneman, "Maps of bounded rationality: Psychology for behavioral economics," *American economic review*, vol. 93, no. 5, pp. 1449–1475, 2003.

[10] J. S. B. Evans, "Heuristic and analytic processes in reasoning," *British Journal of Psychology*, vol. 75, no. 4, pp. 451–468, 1984.

[11] T. Anthony, Z. Tian, and D. Barber, "Thinking fast and slow with deep learning and tree search," *Advances in neural information processing systems*, vol. 30, 2017.

[12] D. Silver, A. Huang, C. J. Maddison, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[13] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, *Geometric deep learning: Grids, groups, graphs, geodesics, and gauges*, 2021. arXiv: 2104.13478 [cs.LG].

[14] "Systemic games," Accessed: 2023-12-02. [Online]. Available: https://the-artifice.com/systemic-games-philosophy.

[15] L. Liu, N. Gurney, K. McCullough, and V. Ustun, "Graph neural network based behavior prediction to support multi-agent reinforcement learning in military training simulations," *2021 Winter Simulation Conference (WSC)*, pp. 1–12, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:247059572.

[16] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *ArXiv*, vol. abs/1706.03741, 2017. [Online]. Available: https://arxiv.org/abs/1706.03741.

[17] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, "Unifying large language models and knowledge graphs: A roadmap," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–20, 2024. DOI: 10.1109/TKDE.2024.3352100.

[18] G. Wang, Y. Xie, Y. Jiang, *et al.*, *Voyager: An open-ended embodied agent with large language models*, 2023. arXiv: 2305.16291 [cs.AI].

[19] D. Hafner, "Benchmarking the spectrum of of agent capabilities," *ArXiv*, vol. abs/2109.06780, 2021. [Online]. Available: https://arxiv.org/pdf/2109.06780.pdf.

[20] A. Devlic, *Discussion on ablation studies in reinforcement learning*, Personal communication at Gothenburg Artificial Intelligence Alliance (GAIA) Conference, Sony AI, Mar. 2024.

[21] A. M. di S. Stefano, *TSP_RL_KG*, 2024. [Online]. Available: https://github.com/antoniomangoni/TSP_RL_KG.