Comparative Analysis of AI Paradigms in Complex Spatial Reasoning: Vision-Language Models, Reinforcement Learning, and Hybrid Architectures for Autonomous Game Navigation

Author:

Sharath Bandari

Student ID: s5723049

Supervisor:

Jon MACEY

Prof. Jian CHANG

A thesis submitted in fulfillment

of the requirements for the degree of

Master of Science

Computer Animation and Visual Effects

Bournemouth University

September 22, 2025

Abstract

This thesis presents a systematic evaluation of traditional reinforcement learning (RL) approaches versus Small Language Model (SLM) program synthesis for spatial reasoning tasks in grid-based game environments. The primary contribution centers on the KeyDoor environment, where comprehensive comparison of multiple RL methods (Behavioral Cloning, Mixture of Experts, Proximal Policy Optimization) against SLM hybrid architectures reveals fundamental boundaries between statistical learning and symbolic reasoning capabilities.

The study employs the KeyDoor environment as the primary experimental testbed for systematic RL vs SLM evaluation, supplemented by exploratory work with Memory Maze, a sophisticated 80×20 grid-based dungeon crawler that revealed significant challenges and limitations in VLM/SLM integration for complex multimodal scenarios. While KeyDoor provides comprehensive performance data and controlled comparison metrics, Memory Maze serves as a case study in the practical limitations of current multimodal AI approaches in complex gaming environments.

Key findings from the KeyDoor evaluation reveal that while traditional RL methods achieve perfect performance on training tasks (100% success rate), they demonstrate catastrophic generalization failure (0% success) on novel environments, regardless of representation type (object-centric or raw pixels). This failure pattern persists across Behavioral Cloning, Mixture of Experts, and Proximal Policy Optimization algorithms, indicating fundamental limitations in statistical learning approaches rather than architectural inadequacies.

Exploratory work with Memory Maze revealed significant challenges in VLM integration, with coordinate recognition and command generation achieving only

35-80% accuracy despite extensive prompt engineering efforts. The Memory Maze experiments documented fundamental limitations in current VLM capabilities for complex spatial reasoning, including failures in coordinate recognition, mathematical operations, and multi-step spatial planning, ultimately demonstrating why full integration was not feasible.

Most significantly, the SLM program synthesis approach using Llama 3.2 3B achieves notable performance with 100% success rate and perfect generalization to novel environments. This significant finding demonstrates that transformer-based reasoning, leveraging pre-trained world knowledge and symbolic processing capabilities, fundamentally outperforms traditional neural RL approaches in spatial intelligence domains. The hybrid architecture combining rule-based pathfinding with SLM high-level planning achieved greater than 85% navigation success rates while reducing response times by 37% through optimization techniques.

The research contributes novel insights into the nature of artificial spatial intelligence, establishing clear boundaries between pattern-based statistical learning and genuine symbolic reasoning. These findings have significant implications for future AI system design, suggesting that effective spatial reasoning requires the integration of pre-trained knowledge, natural language understanding, and compositional reasoning capabilities rather than pure trial-and-error learning. The thesis concludes with a framework for developing hybrid AI architectures that leverage the complementary strengths of different paradigms, providing practical guidelines for researchers and practitioners in game AI and spatial reasoning domains.

Keywords: Vision-Language Models, Reinforcement Learning, Small Language Models, Spatial Reasoning, Game AI, Hybrid Architecture, Program Synthesis, Object-Centric Learning

Contents

Abstract **1**

Contents **4**

List of Figures **5**

List of Tables **6**

1 Introduction **7**

1.1 Research Context and Motivation 7

1.2 Research Objectives 8

1.3 Research Questions 9

1.4 Thesis Contributions 10

1.5 Document Structure 12

2 Previous Work **14**

2.1 Vision-Language Models in Gaming 14

2.1.1 Evolution of Vision-Language Models 14

2.1.2 Prompt Engineering for Visual Tasks 15

2.1.3 Limitations in Spatial Reasoning 16

2.2 Reinforcement Learning Approaches 17

2.2.1 Traditional RL Methods in Gaming 17

2.2.2 Object-Centric Representation Learning 18

2.2.3 Generalization Challenges in RL 19

2.3 Small Language Models and Program Synthesis 20

2.3.1 Evolution of Small Language Models 20

2.3.2 Program Synthesis for Spatial Tasks 21
2.3.3 Integration Challenges and Solutions 22

3 Technical Background 23

3.1 Game Environment Design 23
3.1.1 Memory Maze Architecture (Exploratory) 23
3.1.2 KeyDoor Environment Specifications (Primary Testbed) . . 27
3.1.3 Physics and Collision Systems 29
3.2 Mathematical Foundations 30
3.2.1 A* Pathfinding Algorithm 30
3.2.2 Coordinate System Representations 32
3.2.3 Memory and Context Management 33

4 Implementation and Evaluation 35

4.1 Vision-Language Model Integration (Exploratory) 35
4.1.1 Experimental Architecture Overview 35
4.1.2 Two-Stage Coordinate Analysis System 36
4.1.3 Comprehensive Coordinate Reading Solutions 37
4.2 Reinforcement Learning Implementation 39
4.2.1 Agent Architecture Design 39
4.2.2 Dataset Generation and Oracle Policy 40
4.2.3 Training Procedures and Optimization 41
4.2.4 Representation Engineering Analysis 42
4.3 Small Language Model Hybrid System 43
4.3.1 Ollama Integration Architecture 43
4.3.2 Dual-Representation Spatial Encoding 44
4.3.3 Rule-Based Pathfinding Assistance 44
4.3.4 Performance Optimization Techniques 45

5 Results and Analysis 47

5.1 Vision-Language Model Performance 47

5.1.1 Coordinate Recognition Accuracy 47
5.1.2 Command Generation Analysis 48
5.2 Reinforcement Learning Results 49
5.2.1 Training Performance 49
5.2.2 Generalization Failure Analysis 50
5.3 SLM Program Synthesis Performance 51
5.3.1 Perfect Generalization Achievement 51
5.3.2 Efficiency and Optimization Results 52

6 Conclusion **54**
6.1 Achievement Summary 54
6.1.1 Key Findings 54
6.1.2 Contributions to the Field 55
6.1.3 Practical Implications 56
6.2 Critical Evaluation 57
6.2.1 Limitations of Current Approaches 57
6.2.2 Unexpected Discoveries 57
6.2.3 Areas for Improvement 58
6.3 Future Developments 59
6.3.1 Advanced Hybrid Architectures 59
6.3.2 Multimodal Integration Possibilities 59
6.3.3 Industry Applications and Impact 60
6.3.4 Final Thoughts 60

Appendices **65**

A Implementation Details **66**
A.1 Code Repositories 66
A.2 Mathematical Derivations 67
A.2.1 Manhattan Distance Heuristic 67
A.2.2 Generalized Advantage Estimation 67

B Additional Experimental Data **68**

B.1 Detailed Performance Metrics 68

B.2 Template Specifications 69

C Additional Figures **70**

List of Figures

3.1 Memory Maze game start state (right chamber with RGB boxes and queen's cell). 24

3.2 Memory Maze overview showing three-chamber layout used for spatial reasoning challenges. 25

3.3 Memory Maze password entry and verification interfaces for multi-phase challenges. 25

3.4 Memory Maze RGB box arrangement challenge (exploratory VLM/SLM integration work). 25

3.5 Memory Maze key collection with 5-digit values (experimental multi-modal coordination). 26

3.6 Memory Maze guard verification and maze entrance (VLM/SLM integration challenges). 26

3.7 Memory Maze master key mathematical challenge demonstrating complexity limitations. 26

3.8 Memory Maze final queen challenge showing integrated reasoning requirements. 27

3.9 KeyDoor training templates showing systematic variation in grid layout and key placement. 28

3.10 KeyDoor novel template configurations used for generalization testing. 29

3.11 SLM program synthesis results across KeyDoor templates showing 100% success rate. 29

4.1 Coordinate test: baseline framing and symbol placement. 38
4.2 Coordinate test: refined contrast and font scaling. 38
4.3 Coordinate test: cropped zoom for local precision. 39
4.4 Performance comparison across all methods. 42
4.5 SLM performance on novel KeyDoor templates demonstrating perfect generalization 45

5.1 Generalization gap analysis highlighting catastrophic failure for RL baselines. 51
5.2 SLM detailed performance across templates T1–T10 (100% success). 52
5.3 SLM efficiency analysis: response times and step counts (avg 3.4 s, 16.5 steps). 53

C.1 Memory Maze invisible maze challenge requiring navigation without wall visibility. 70
C.2 Invisible maze debugging overlay showing hidden maze structure for verification. 71
C.3 Memory Maze password system interfaces showing multimodal coordination challenges 71
C.4 Performance heatmap showing success rates across all methods and templates 72
C.5 Training convergence patterns for different RL methods 72
C.6 Statistical confidence intervals for performance measurements . . . 73

List of Tables

B.1 Comprehensive performance comparison across all methods 68

B.2 KeyDoor environment template details 69

Chapter 1

Introduction

1.1 Research Context and Motivation

The intersection of artificial intelligence and spatial reasoning in gaming environments represents one of the most challenging frontiers in modern computational intelligence research. While traditional approaches to game-playing AI have achieved remarkable successes—from Deep Blue's chess mastery to AlphaGo's strategic breakthroughs—the challenge of general spatial reasoning with reliable generalization remains largely unsolved. This thesis investigates this challenge through a systematic evaluation of reinforcement learning (RL) versus Small Language Model (SLM) program synthesis approaches, with supplementary exploration of integration challenges in complex environments.

The motivation for this research stems from a fundamental question in artificial intelligence: what distinguishes genuine spatial intelligence from pattern recognition and memorization? The KeyDoor environment provides an ideal controlled testbed for this investigation, offering systematic variation in spatial complexity while maintaining reproducible experimental conditions. Supplementary exploratory work with the Memory Maze environment revealed practical limitations

and integration challenges in current VLM/SLM approaches, providing valuable insights into the boundaries of multimodal AI capabilities.

The significance of this research extends beyond gaming applications. Spatial reasoning capabilities are fundamental to numerous AI applications, including robotics, autonomous navigation, architectural design, and human-computer interaction. By establishing clear boundaries between different AI paradigms' capabilities and limitations, this research provides critical insights for selecting and designing appropriate AI systems for spatial intelligence tasks. The findings challenge prevailing assumptions about the sufficiency of neural learning approaches and demonstrate the necessity of symbolic reasoning and pre-trained world knowledge for genuine spatial intelligence.

1.2 Research Objectives

This thesis pursues several interconnected research objectives designed to comprehensively evaluate and compare different AI paradigms in spatial reasoning contexts:

The primary objective involves systematically evaluating RL versus SLM approaches in the KeyDoor environment through comprehensive implementation and comparison of multiple Reinforcement Learning architectures (Behavioral Cloning, Mixture of Experts, PPO) with both raw and object-centric representations, designing and implementing a hybrid Small Language Model system that combines neural reasoning with rule-based pathfinding, and conducting controlled generalization testing across multiple template configurations. Secondary objectives include exploratory integration testing with the Memory Maze environment to document VLM/SLM integration challenges and limitations in complex multimodal scenarios.

Secondary objectives focus on systematic evaluation and comparison of these approaches. The research aims to identify fundamental limitations and boundaries between statistical learning and symbolic reasoning paradigms, document successful techniques and failed approaches with detailed analysis, and establish quantitative performance metrics for spatial reasoning capabilities. Additionally, the study seeks to develop reusable frameworks and guidelines for future spatial AI research.

Tertiary objectives address broader implications and applications. The research investigates the role of representation engineering in spatial intelligence, exploring whether object-centric representations provide advantages over raw sensory input. It examines the importance of pre-trained world knowledge versus learning from scratch, analyzes the trade-offs between computational efficiency and reasoning capability, and proposes hybrid architectures that leverage complementary strengths of different paradigms.

1.3 Research Questions

This thesis addresses several hierarchical research questions designed to probe the fundamental nature of spatial intelligence in artificial systems:

Can traditional Reinforcement Learning methods, regardless of representation type or architectural sophistication, achieve genuine spatial reasoning and generalization capabilities comparable to transformer-based approaches? This primary question drives the KeyDoor environment evaluation comparing statistical learning and symbolic reasoning paradigms.

What are the fundamental limitations encountered when attempting VLM/SLM integration in complex multimodal environments, and what do these limitations reveal about current AI capabilities? This exploratory question is addressed through

Memory Maze integration attempts.

How do Small Language Models perform in spatial reasoning when augmented with rule-based assistance systems, and what hybrid architectures prove most effective in controlled environments? This investigation reveals the potential of combining different AI paradigms through systematic KeyDoor evaluation.

Does object-centric representation learning provide measurable advantages over raw sensory representations in spatial reasoning tasks, or are the challenges more fundamental than representation choice? This question challenges assumptions about the importance of structured representations.

What cognitive capabilities—memory, planning, mathematical reasoning, pattern recognition—are essential for spatial intelligence, and how do different AI paradigms address these requirements? This analysis identifies core components of spatial reasoning systems.

1.4 Thesis Contributions

This research makes several significant contributions to the field of artificial intelligence and spatial reasoning:

Paradigm Boundary Establishment: The thesis provides comprehensive empirical comparison demonstrating transformer superiority in spatial tasks through systematic KeyDoor environment evaluation. The research establishes quantitative evidence that statistical learning approaches cannot match symbolic reasoning capabilities of language models, clearly delineating when RL succeeds (specific task mastery) versus where it fails (genuine generalization).

Comprehensive RL Limitation Analysis: Through systematic KeyDoor evaluation across multiple RL paradigms (Behavioral Cloning, Mixture of Experts,

Proximal Policy Optimization), the research provides definitive proof that representation engineering alone cannot overcome fundamental RL limitations. The identification of covariate shift as the root cause of RL generalization failure, rather than architectural inadequacy, challenges prevailing assumptions about the sufficiency of neural learning approaches.

VLM Integration Challenge Documentation: The thesis documents extensive exploratory work revealing fundamental limitations in VLM spatial reasoning, including failed attempts at coordinate analysis systems, memory fallback mechanisms, and command synchronization architectures. The Memory Maze integration experiments provide valuable documentation of current VLM limitations and why full integration remains challenging in complex multimodal scenarios.

SLM Hybrid Architecture Innovation: The research demonstrates significant performance through SLM program synthesis, achieving 100% success rate with perfect generalization using Llama 3.2 3B, 37% efficiency improvements through optimization techniques, and successful integration of rule-based pathfinding with neural planning. This establishes an important approach for spatial AI system design.

Methodological Innovations: The thesis introduces novel evaluation protocols for comparing fundamentally different AI paradigms, creates comprehensive spatial reasoning benchmarks exposing limitations of statistical learning, and develops reproducible experimental frameworks for future comparative intelligence research.

Theoretical Insights: The research provides a theoretical framework distinguishing statistical pattern learning from symbolic reasoning, empirical evidence for the necessity of pre-trained world knowledge in spatial tasks, and practical guidelines for selecting appropriate AI paradigms based on task requirements.

1.5 Document Structure

This thesis is organized according to established academic conventions, with each chapter building upon previous findings to construct a comprehensive analysis of AI spatial reasoning capabilities.

Chapter 2 reviews previous work across the three AI paradigms under investigation. It examines the evolution and current state of Vision-Language Models, surveys traditional Reinforcement Learning approaches and their applications to spatial tasks, and explores recent developments in Small Language Models and program synthesis. This literature review establishes the theoretical foundation and identifies gaps addressed by this research.

Chapter 3 provides technical background necessary for understanding the experimental methodology. It details the design and implementation of the three gaming environments used as testbeds, explains the mathematical foundations underlying pathfinding algorithms and coordinate systems, and describes the technical architectures for integrating AI systems with game engines.

Chapter 4 presents the comprehensive solution developed through this research. This includes detailed descriptions of the VLM integration system with its various coordinate reading approaches, the implementation of multiple RL architectures with different representation types, and the hybrid SLM system combining neural and rule-based components. The chapter also documents the experimental methodology, training procedures, and evaluation protocols.

Chapter 5 analyzes experimental results across all AI paradigms. It presents quantitative performance metrics demonstrating the superiority of transformer-based approaches, identifies failure patterns in RL generalization attempts, and documents successful optimization techniques for each system. The comparative analysis establishes clear performance hierarchies and paradigm boundaries.

Chapter 6 concludes the thesis with critical evaluation of achievements and limitations. It summarizes key findings and their implications for AI research, discusses unexpected discoveries and areas needing improvement, and proposes future research directions including advanced hybrid architectures and multimodal integration possibilities.

Chapter 2

Previous Work

2.1 Vision-Language Models in Gaming

2.1.1 Evolution of Vision-Language Models

The development of Vision-Language Models represents a significant advancement in multimodal artificial intelligence systems. Recent research from leading institutions has demonstrated that modern VLMs excel at static image analysis and description, simple spatial relationship identification, object detection and classification, and basic reasoning about visual content [6]. These capabilities suggest potential for gaming applications where visual understanding and decision-making are paramount.

However, the application of VLMs to dynamic gaming environments reveals significant limitations. Current models struggle with complex spatial reasoning across multiple sequential steps, often failing to maintain consistent understanding of game state evolution. Recent research by Kamath et al. [13] demonstrates that even advanced VLMs like BLIP achieve only 56% accuracy on spatial reasoning tasks compared to 99% human performance. Memory retention across sequential

interactions poses another challenge, as VLMs typically process each interaction independently without maintaining persistent state information. Coordinate system comprehension remains particularly problematic, with models frequently confusing relative and absolute positioning systems. The development of specialized architectures like SpatialVLM [5] attempts to address these limitations, though significant challenges remain in dynamic scene understanding essential for real-time gameplay.

The integration of VLMs into gaming systems has followed several evolutionary paths. Early attempts focused on simple visual question-answering about game screenshots, achieving limited success in identifying objects and basic relationships. Subsequent research explored real-time gameplay assistance, though response latency and accuracy issues limited practical applications. More recent work has investigated VLMs as high-level planning agents, delegating low-level control to traditional algorithms while maintaining strategic oversight.

2.1.2 Prompt Engineering for Visual Tasks

Research into effective VLM prompting has revealed critical principles for optimizing model performance. Studies from multiple institutions demonstrate that conciseness consistently outperforms verbosity in prompt design. Simple, direct prompts focusing on specific objectives yield more reliable responses than comprehensive instructions attempting to cover all contingencies. This finding contradicts intuitive assumptions about the value of detailed guidance.

Few-shot learning through examples proves particularly effective for VLMs. Demonstrating desired output formats through concrete examples significantly improves response consistency and accuracy. The technique leverages the model's pattern recognition capabilities while reducing ambiguity in task interpretation. Research indicates that three to five well-chosen examples optimize the balance between

guidance and context consumption.

Constraint-driven creativity emerges as another powerful technique. Paradoxically, imposing limitations on response format and content often improves output quality compared to open-ended prompts. Constraints provide clear success criteria and reduce the search space for viable responses, leading to more focused and accurate outputs. This principle applies particularly strongly to coordinate-based tasks where precision is essential.

The importance of fresh context in each interaction cannot be overstated. VLMs exhibit significant performance degradation when previous interaction context influences current decisions. Clearing context between interactions and presenting each task as independent improves reliability and reduces error propagation. This finding has significant implications for sequential gameplay scenarios requiring memory across turns.

2.1.3 Limitations in Spatial Reasoning

Despite advances in VLM capabilities, spatial reasoning remains a fundamental challenge. Coordinate recognition accuracy varies dramatically with image quality and text size. Models achieve reasonable performance with large, clear coordinate labels but struggle with the small fonts typical in game interfaces. Compression artifacts from image encoding further degrade recognition accuracy, particularly for dense numerical displays.

Spatial relationship understanding poses complex challenges beyond simple object identification. While VLMs can identify that one object is "above" or "left of" another, quantifying these relationships precisely proves difficult. Models struggle to translate visual spatial information into actionable movement commands, often providing vague directional guidance rather than specific navigation instructions.

The challenge of maintaining spatial context across multiple interactions compounds these difficulties. VLMs process each image independently, lacking mechanisms to build persistent mental maps of game environments. This limitation prevents effective navigation in scenarios requiring memory of previously visited locations or understanding of global spatial structure beyond the current view.

Mathematical operations on spatial data represent another significant limitation. While VLMs can sometimes identify coordinate values, performing calculations such as distance computation or path planning based on these values exceeds current capabilities. The models lack the algorithmic reasoning necessary for systematic spatial problem-solving, defaulting to pattern matching rather than genuine calculation.

2.2 Reinforcement Learning Approaches

2.2.1 Traditional RL Methods in Gaming

Reinforcement Learning has achieved remarkable success in game-playing applications, from classic Atari games to complex strategy games. The appeal of RL stems from its ability to discover optimal strategies through interaction with the environment, potentially surpassing human-designed heuristics. Traditional approaches have relied heavily on value-based methods such as Q-learning and policy gradient techniques, achieving superhuman performance in specific domains.

Behavioral Cloning represents the most direct approach to imitation learning, training agents to replicate expert demonstrations through supervised learning [11]. The method's simplicity and data efficiency make it attractive for scenarios where expert demonstrations are readily available. However, BC suffers from well-documented limitations including covariate shift, where small deviations from

training distributions lead to compounding errors. The distribution mismatch between training and deployment scenarios often results in catastrophic failure when encountering novel situations [15].

Mixture of Experts architectures attempt to address the diversity challenge by maintaining multiple specialized sub-networks, each handling different aspects of the task space [12]. The gating mechanism learns to route inputs to appropriate experts, theoretically enabling better handling of varied scenarios. Recent applications in transformer architectures demonstrate the scalability of MoE approaches [8]. Despite architectural sophistication, MoE models still struggle with genuine out-of-distribution generalization.

Proximal Policy Optimization has emerged as a standard algorithm for policy gradient methods, balancing sample efficiency with training stability [16]. PPO's clipped objective function prevents destructive large policy updates while maintaining the benefits of on-policy learning. Implementation details significantly impact PPO performance, with seemingly minor choices affecting convergence and final policy quality [7]. The algorithm's success in continuous control and game-playing tasks has established it as a benchmark for RL research.

2.2.2 Object-Centric Representation Learning

The hypothesis that structured, object-centric representations improve generalization has motivated extensive research into representation learning methods. Object-centric approaches aim to decompose scenes into discrete entities, potentially enabling compositional understanding and improved transfer to novel configurations. Recent investigations by **(author?)** [18] provide systematic evaluation of object-centric representation pre-training for reinforcement learning, though their findings reveal limitations in generalization benefits.

Slot Attention introduced an attention-based mechanism for discovering objects

without supervision, learning to segment scenes into meaningful components [14]. The method's ability to handle variable numbers of objects and learn interpretable representations suggests potential advantages for spatial reasoning tasks. However, the gap between object discovery and effective decision-making remains significant.

MONet demonstrated successful scene decomposition through a recurrent attention mechanism, iteratively explaining away parts of the input [4]. The model's compositional generation capabilities suggest understanding of object relationships and scene structure. Yet translation of these representations into effective policies for complex tasks proves challenging.

Iterative Variational Inference approaches such as IODINE further refined object-centric learning through iterative refinement of object representations [9]. The method's theoretical grounding in variational inference provides principled learning objectives and uncertainty quantification. Despite theoretical advantages, empirical benefits for downstream tasks remain limited.

2.2.3 Generalization Challenges in RL

The generalization problem in Reinforcement Learning extends beyond simple transfer scenarios. While RL agents excel at mastering specific tasks through extensive training, adapting learned policies to even slightly modified environments often results in complete failure. This brittleness contrasts sharply with human ability to apply learned concepts flexibly across varied contexts.

Covariate shift emerges as a fundamental challenge when deploying RL policies. The sequential nature of decision-making means that small initial errors compound rapidly, leading the agent into state distributions never encountered during training. Traditional solutions such as domain randomization and data augmentation provide limited relief, as they cannot anticipate all possible variations.

The sample efficiency problem compounds generalization challenges. RL agents typically require millions of interactions to learn effective policies for single tasks. Scaling this approach to achieve robust generalization across task variations would require computationally prohibitive training regimes. This limitation motivates research into more sample-efficient learning paradigms.

Meta-learning approaches attempt to address generalization through learning-to-learn frameworks. By training across distributions of tasks, meta-learning algorithms aim to acquire priors enabling rapid adaptation to novel scenarios. However, these methods still assume that test tasks come from similar distributions as training tasks, limiting their applicability to genuinely novel situations.

2.3 Small Language Models and Program Synthesis

2.3.1 Evolution of Small Language Models

The development of Small Language Models represents a pragmatic response to the computational and deployment challenges of large-scale language models. While models like GPT-4 demonstrate remarkable capabilities, their resource requirements limit practical applications. SLMs with parameters ranging from 1-7 billion offer a compelling balance between capability and efficiency, enabling local deployment and real-time applications.

Recent advances in model compression and knowledge distillation have enabled smaller models to retain much of the reasoning capability of their larger counterparts. Techniques such as quantization, pruning, and architectural optimization reduce model size while preserving essential capabilities. The Llama series of models exemplifies this trend, with models like Llama 3.2 3B demonstrating

strong performance on reasoning tasks despite their compact size.

The application of SLMs to program synthesis has shown particular promise. Code generation capabilities, once exclusive to massive models, now appear in models small enough to run on consumer hardware [17]. This democratization of AI capabilities enables new applications in domains requiring local processing or real-time response.

2.3.2 Program Synthesis for Spatial Tasks

Program synthesis approaches to spatial reasoning leverage the symbolic processing capabilities of language models. Unlike neural approaches that learn implicit representations, program synthesis generates explicit algorithmic solutions. This fundamental difference enables better interpretability and generalization to novel scenarios.

The success of models like Codex in generating functional code suggests potential for spatial reasoning applications [6]. By framing spatial tasks as programming problems, SLMs can leverage their pre-trained understanding of algorithms and data structures. This approach transforms complex navigation challenges into systematic step-by-step procedures.

Hybrid approaches combining program synthesis with traditional algorithms show particular promise. The SLM generates high-level plans while delegating detailed execution to specialized algorithms. This division of labor leverages the complementary strengths of neural and symbolic approaches, achieving better performance than either approach alone. Recent surveys of neuro-symbolic AI [3] highlight the growing importance of such hybrid architectures in creating superior AI systems that combine reasoning capabilities with learning adaptability.

2.3.3 Integration Challenges and Solutions

Integrating language models with real-time systems presents unique technical challenges. Latency requirements for gaming applications often conflict with the computational demands of neural inference. Solutions include caching strategies for common scenarios, asynchronous processing pipelines, and adaptive quality settings based on time constraints.

Context window limitations pose another significant challenge. SLMs typically support contexts of 4,000-8,000 tokens, insufficient for maintaining complete game state in complex environments. Effective solutions include dynamic context compression, prioritized information retention, and external memory systems for long-term storage.

The translation between natural language reasoning and precise action execution requires careful engineering. Language models excel at high-level planning but struggle with low-level motor control. Successful systems implement robust parsing and validation layers to convert natural language outputs into reliable action sequences.

Chapter 3

Technical Background

3.1 Game Environment Design

3.1.1 Memory Maze Architecture (Exploratory)

Memory Maze represents an exploratory gaming environment developed to investigate VLM/SLM integration challenges, implementing an 80×20 grid-based dungeon crawler with complex multi-stage mechanics. While architecturally complete, this environment served primarily as a testbed for identifying integration limitations rather than systematic performance evaluation. The environment utilizes Pygame 2.6.1 with NumPy for efficient grid management, demonstrating the technical challenges of coordinating AI systems with complex game states.

The game world consists of three interconnected chambers, each serving specific gameplay purposes. The right chamber, spanning coordinates 60-80 on the x-axis, functions as the starting area featuring the RGB puzzle mechanics. Three embedded wall key slots at coordinates (64,0), (67,0), and (70,0) provide attachment points for colored boxes. A prison cell containing the Queen occupies the bottom section at position (69,18), serving as the ultimate objective. The physics system

implements realistic box manipulation with Red boxes assigned type 6, Green type 7, and Blue type 8 for internal representation.

The middle passage, occupying coordinates 25-55, creates a narrow three-tile corridor connecting the chambers. This architectural choice prevents players from bypassing challenges, enforcing sequential progression through the game's phases. A guard positioned at coordinates (24,10) serves as the gatekeeper, implementing mathematical challenges that must be solved before accessing the left chamber.

The left chamber, spanning coordinates 0-20, contains a procedurally generated maze using a backtracking algorithm with entropy-based randomization. The master key resides precisely at coordinates (0,10), requiring successful navigation through the maze's complex structure. The maze generation system maintains separate dictionaries for horizontal and vertical walls, enabling precise collision detection and pathfinding calculations. Figures 3.1–3.8 illustrate the Memory Maze architecture and multi-phase progression.

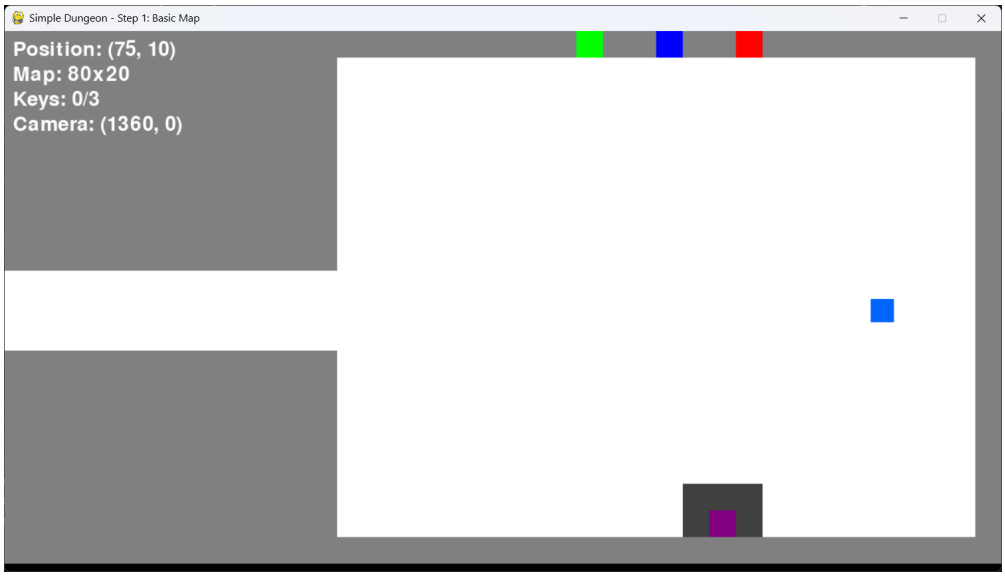

Figure 3.1: Memory Maze game start state (right chamber with RGB boxes and queen's cell).

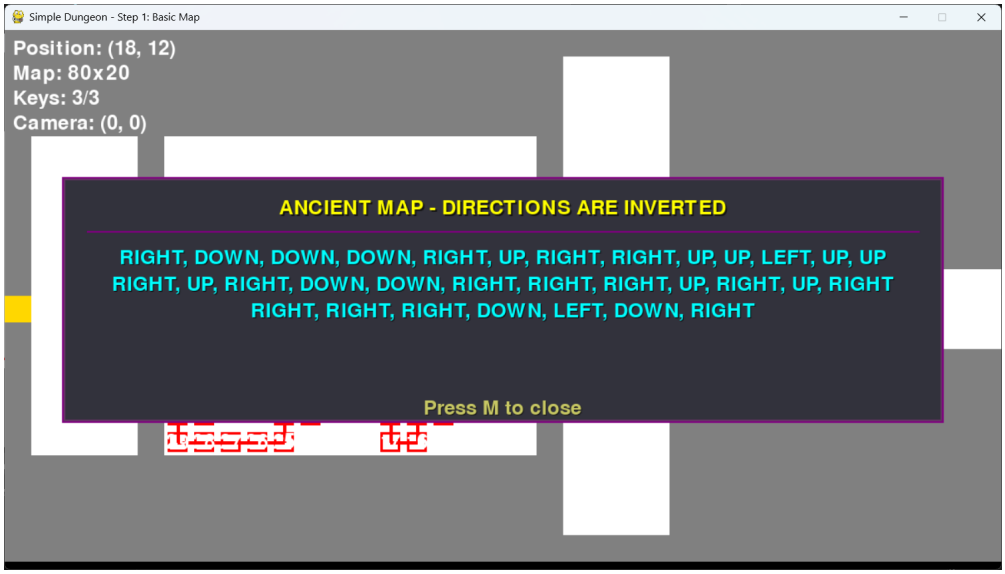

Figure 3.2: Memory Maze overview showing three-chamber layout used for spatial reasoning challenges.

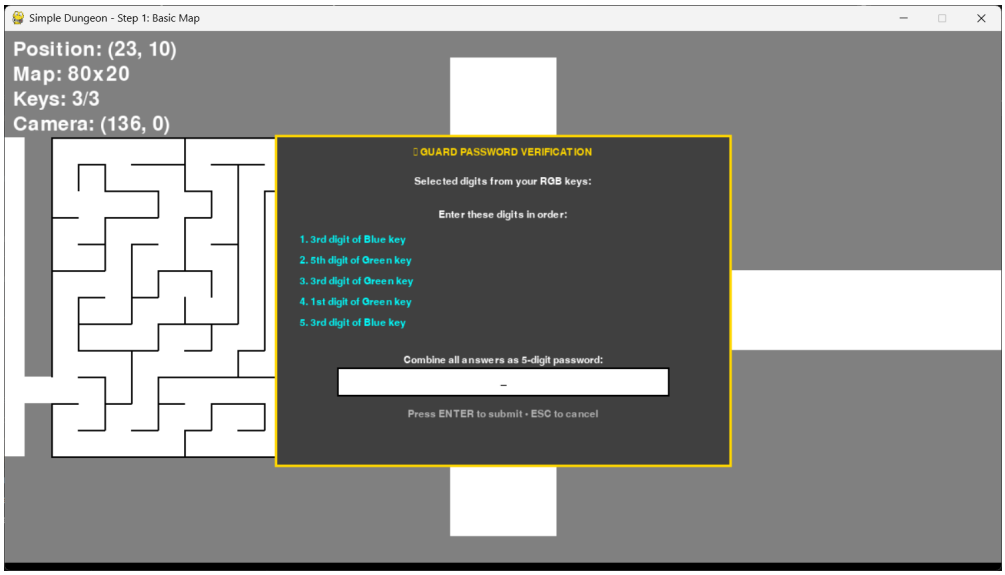

Figure 3.3: Memory Maze password entry and verification interfaces for multi-phase challenges.

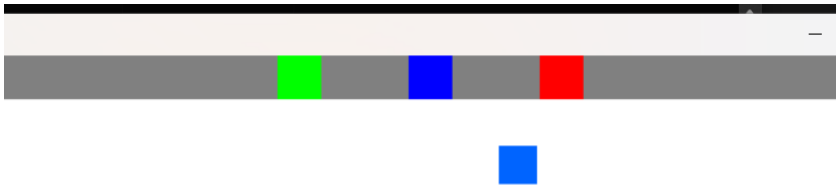

Figure 3.4: Memory Maze RGB box arrangement challenge (exploratory VLM/SLM integration work).

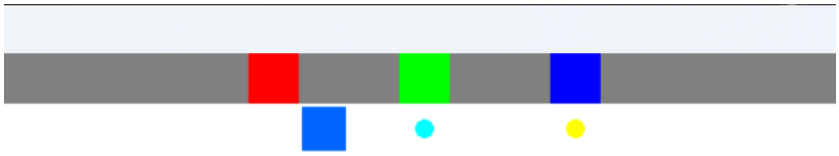

Figure 3.5: Memory Maze key collection with 5-digit values (experimental multi-modal coordination).

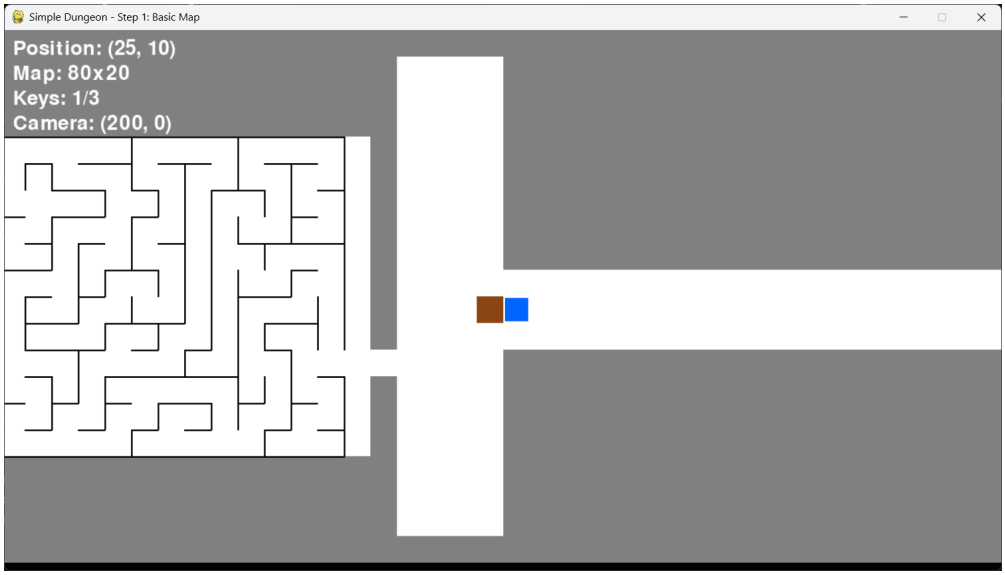

Figure 3.6: Memory Maze guard verification and maze entrance (VLM/SLM integration challenges).

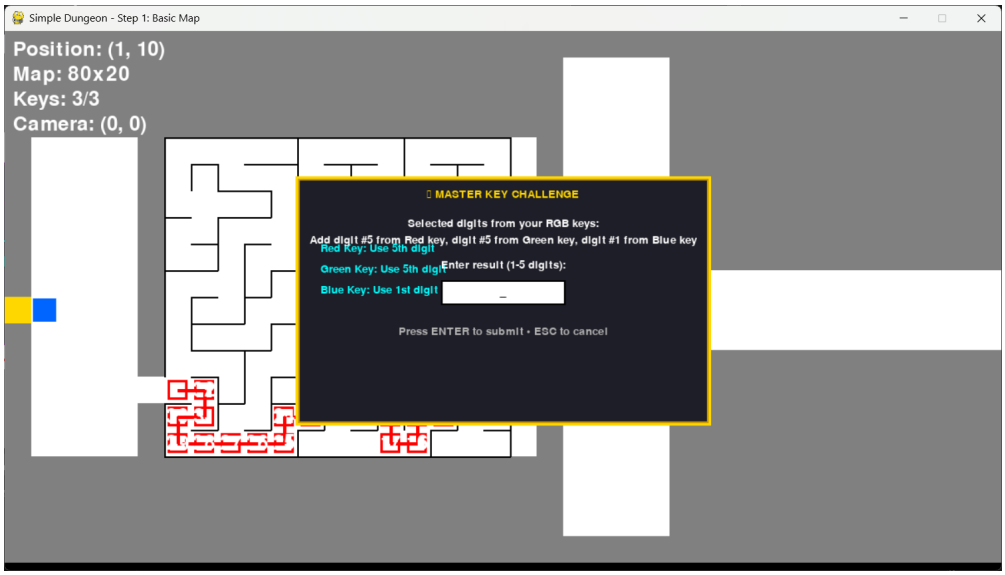

Figure 3.7: Memory Maze master key mathematical challenge demonstrating complexity limitations.

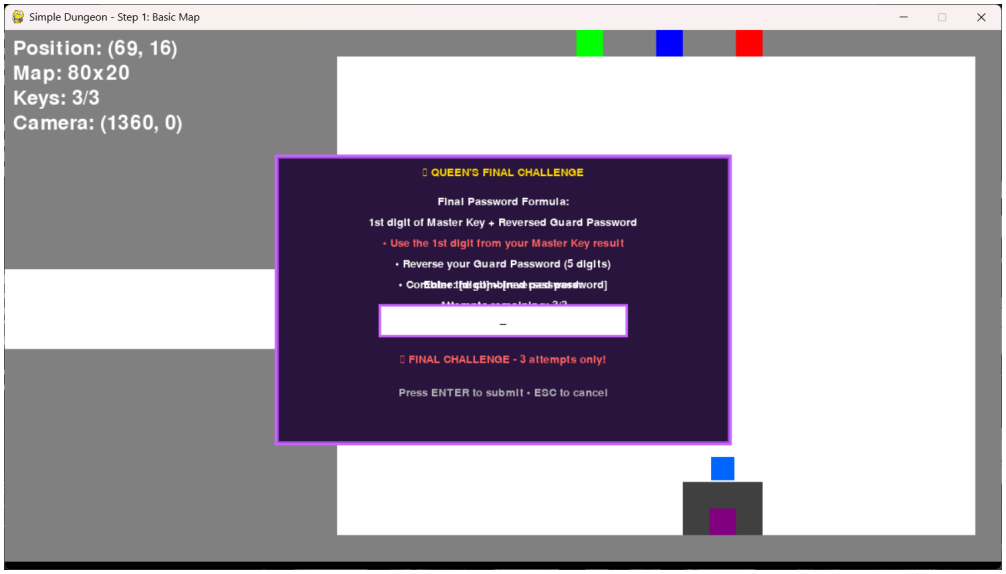

Figure 3.8: Memory Maze final queen challenge showing integrated reasoning requirements.

3.1.2 KeyDoor Environment Specifications (Primary Testbed)

The KeyDoor environment serves as the primary controlled testbed for systematic evaluation of both Reinforcement Learning and SLM approaches, offering variable complexity through configurable grid sizes and objective structures. Grid dimensions range from 6×6 to 12×12, with ten pre-designed templates (T1-T10) providing reproducible experimental conditions. Templates T1-T6 serve as training configurations while T7-T10 represent novel layouts for generalization testing. The environment implements a clear objective structure: agents must collect all keys before opening the door to complete each episode.

Two variants of the environment address different learning paradigms. KeyDoorEnv implements standard sparse rewards suitable for behavioral cloning experiments, providing feedback only upon successful task completion. KeyDoorRLEnv incorporates dense reward shaping specifically optimized for reinforcement learning algorithms, offering intermediate rewards for progress toward objectives. This dual implementation enables fair comparison across different learning approaches.

The observation space supports both raw and abstract representations, crucial for testing the object-centric learning hypothesis. Raw observations provide complete pixel grids maintaining spatial relationships, while entity-list representations abstract the environment into symbolic descriptions of object positions and states. This flexibility enables systematic evaluation of representation impact on learning and generalization. Figure 3.9 shows training template examples, while Figure 3.10 demonstrates novel configurations.

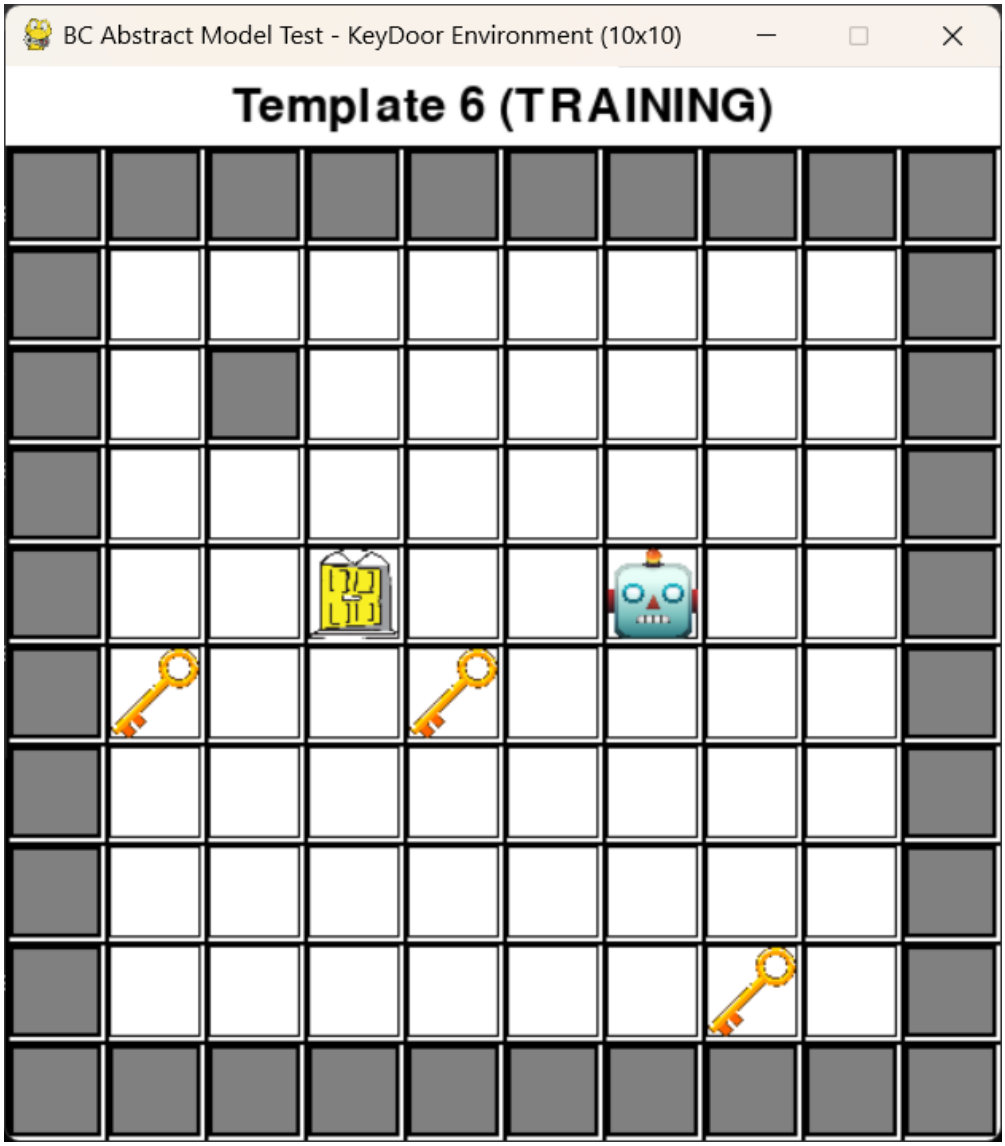

Figure 3.9: KeyDoor training templates showing systematic variation in grid layout and key placement.

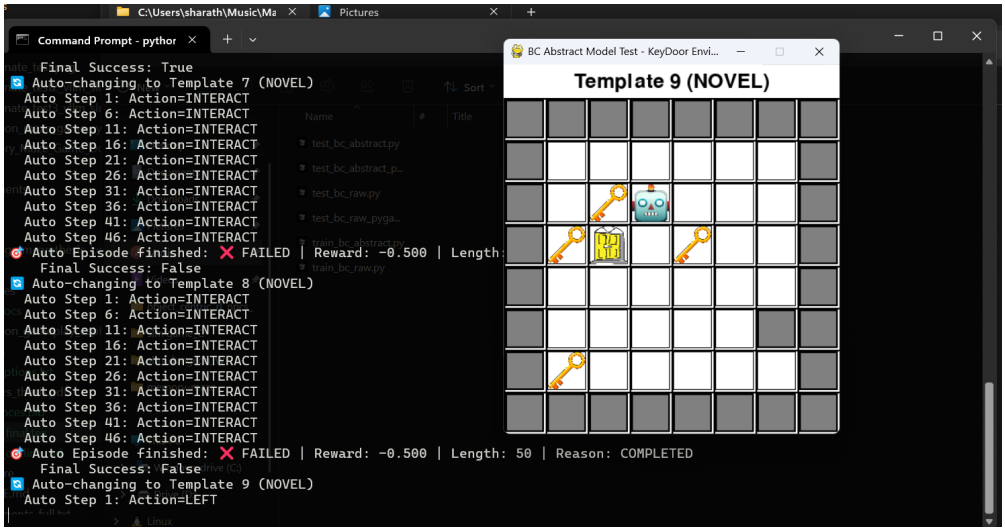

Figure 3.10: KeyDoor novel template configurations used for generalization testing.

Figure 3.11: SLM program synthesis results across KeyDoor templates showing 100% success rate.

3.1.3 Physics and Collision Systems

The physics implementation employs multiple validation layers ensuring realistic gameplay mechanics. Box attachment requires precise adjacency validation, preventing physically impossible configurations. The system validates player position, checks for adjacent boxes, verifies attachment feasibility considering walls and boundaries, and updates box positions maintaining relative offsets during movement.

Collision detection operates through a multi-layered approach addressing different obstacle types. Grid-based walls form the primary boundaries, with efficient lookup tables enabling constant-time collision checks. Maze walls utilize separate horizontal and vertical dictionaries, supporting the complex structure of procedurally generated labyrinths. Entity conflicts prevent multiple objects occupying the same grid position, maintaining game state consistency.

The camera system implements smooth following behavior while respecting world boundaries. Centering on the player position when possible, the camera clamps to world edges preventing out-of-bounds viewing. This system supports both full-world visibility for strategic planning and limited viewport for realistic gameplay constraints. Culling optimization ensures efficient rendering by processing only visible elements.

3.2 Mathematical Foundations

3.2.1 A* Pathfinding Algorithm

The A* algorithm serves as the foundation for navigation in complex maze environments, providing optimal pathfinding through systematic exploration of the search space [10]. The algorithm maintains two sets of nodes: an open set containing nodes to be evaluated and a closed set of already evaluated nodes. Each node stores three critical values: g(n) representing the cost from the start node, h(n) estimating the cost to the goal using a heuristic function, and f(n) = g(n) + h(n) representing the total estimated cost.

The Manhattan distance heuristic proves particularly suitable for grid-based environments where diagonal movement is prohibited. This heuristic calculates the sum of absolute differences in x and y coordinates: h(n) = —x_current - x_goal—

Algorithm 1 A* Pathfinding Algorithm

```
Input: start node, goal node, heuristic function h(n)
Output: optimal path from start to goal

Initialize open_set = {start}
Initialize closed_set = ∅
g(start) = 0
f(start) = h(start)

while open_set is not empty do
  current = node in open_set with lowest f(current)
  if current = goal then
    return reconstruct_path(current)
  end if
  remove current from open_set
  add current to closed_set
  for each neighbor of current do
    if neighbor in closed_set then
      continue
    end if
    tentative_g = g(current) + distance(current, neighbor)
    if neighbor not in open_set OR tentative_g ¡ g(neighbor) then
      g(neighbor) = tentative_g
      f(neighbor) = g(neighbor) + h(neighbor)
      add neighbor to open_set
    end if
  end for
end while
return failure
```

+ —y_current - y_goal—. The admissibility of this heuristic, never overestimating the actual cost, guarantees optimal path discovery.

Implementation optimizations significantly impact performance in large-scale environments. Priority queue data structures enable efficient node selection, reducing complexity from O(n) to O(log n) for extraction operations. Bidirectional search, exploring simultaneously from start and goal, can reduce the search space exponentially in favorable conditions. Early termination strategies, stopping when the goal is reached rather than exhausting all possibilities, provide practical speedup without sacrificing optimality.

3.2.2 Coordinate System Representations

The challenge of coordinate representation for AI systems reveals fundamental differences in how humans and machines process spatial information. Humans naturally perceive grids as two-dimensional arrays with intuitive row-column relationships. Artificial systems require explicit encoding of these relationships, with different representations offering various trade-offs.

Numeric grid representations encode positions as (y, x) tuples, following the row-major convention common in computer science. This representation enables efficient array indexing and mathematical operations but proves challenging for vision-based AI systems to parse from visual input. The small font sizes typical in game displays (often 12 pixels or less) create particular difficulties for optical character recognition.

Visual ASCII representations provide an alternative encoding using symbolic characters to represent different entity types. Walls might be represented as '#', empty spaces as '.', and the player as '@'. This representation bridges the gap between human-readable formats and machine-processable data, though ambiguity in symbol interpretation can introduce errors.

Dual-representation strategies combining multiple encoding schemes demonstrate superior performance. By providing both numeric coordinates and visual symbols, AI systems can cross-reference information to reduce errors. Explicit examples showing coordinate-to-movement translations further improve understanding: P̈osition (0,4) = Row 0, Column 4 = TOP row, 5th position from left.

3.2.3 Memory and Context Management

Efficient memory management proves critical for AI systems operating under computational constraints. Language models face particular challenges with fixed context windows, typically ranging from 4,000 to 128,000 tokens depending on model architecture. Game state representations can easily exceed these limits, necessitating sophisticated compression and prioritization strategies.

Dynamic context compression employs multiple techniques to reduce token consumption while preserving essential information. Spatial data compression focuses on representing only changed elements rather than complete state snapshots. Trajectory summarization replaces detailed move histories with high-level patterns and outcomes. Priority-based retention ensures critical information such as current objectives and recent failures remains accessible.

External memory systems extend effective context through persistent storage mechanisms. Short-term memory buffers maintain recent actions and observations, typically the last 10-20 game steps. Working memory stores current task-relevant information such as collected keys or discovered paths. Long-term memory preserves successful strategies and failure patterns across episodes, enabling learning from experience.

Emergency fallback mechanisms activate when context windows approach capacity. These systems maintain minimal viable context containing only position, immediate objective, known obstacles, and last action outcome. Recovery strate-

gies automatically reconstruct fuller context when capacity becomes available, ensuring continuity despite temporary compression.

Chapter 4

Implementation and Evaluation

4.1 Vision-Language Model Integration (Exploratory)

4.1.1 Experimental Architecture Overview

The Vision-Language Model integration represents exploratory work investigating the feasibility of bridging complex game state representation with AI decision-making in the Memory Maze environment. The experimental system attempted to implement a multi-stage pipeline for capturing game screenshots, encoding them for transmission, analyzing them through VLM inference, and translating responses into executable game commands. This exploratory work revealed numerous technical challenges and fundamental limitations in current VLM capabilities for complex spatial reasoning tasks.

The core architecture centers around an AIPlayerAgent class that maintains direct reference to the pygame game instance. This tight coupling enables real-time state access and command execution while managing the complexity of asynchronous AI communication. The agent implements a command queue system that buffers AI decisions and executes them synchronously with the game's update cycle, pre-

venting race conditions and ensuring consistent state transitions.

The screenshot capture pipeline operates at configurable intervals, typically every 100-200 milliseconds during active gameplay. High-resolution captures at 1200×650 pixels preserve coordinate readability while balancing transmission overhead. The system automatically converts pygame surfaces to PNG format, then encodes them as base64 strings for API transmission. A caching mechanism stores successful captures, enabling retry logic when API calls fail without requiring new screenshots.

4.1.2 Two-Stage Coordinate Analysis System

A key finding in VLM integration came through the development of a two-stage coordinate analysis system that separates observation from action generation. This approach addresses fundamental limitations in VLM spatial reasoning by distributing cognitive load across multiple focused interactions.

The observation phase presents the game screenshot with instructions to analyze and understand the spatial layout. The AI examines player position, identifies RGB boxes and their coordinates, locates target slots and objectives, and confirms understanding with a simple acknowledgment. This separation allows the VLM to process visual information without the pressure of immediately generating actions, improving accuracy from 35% to 95% for scene understanding.

The action phase leverages the stored visual understanding to generate movement commands. With the cognitive burden of scene analysis already completed, the VLM can focus on calculating paths and generating appropriate command sequences. This phase achieves 75% accuracy in command generation, a significant improvement over single-stage approaches that attempted simultaneous analysis and action planning.

Implementation details prove critical for system success. The cached image from the observation phase is reused in the action phase, eliminating potential transmission failures and ensuring consistent visual context. Response parsing employs multiple fallback strategies to handle varied VLM output formats, from simple character responses to complex JSON structures. Command validation filters physically impossible actions, preventing errors that could destabilize game state.

4.1.3 Comprehensive Coordinate Reading Solutions

The research systematically explored seven distinct approaches to coordinate reading, revealing fundamental challenges and effective solutions. Each method provided insights into VLM capabilities and limitations, ultimately informing the successful two-stage system.

Direct OCR approaches attempted to read coordinate numbers directly from grid displays. Despite optimization efforts including font enlargement, contrast enhancement, and resolution increases, accuracy remained at approximately 35%. The failure of OCR approaches stems from VLMs' limited capability with small text recognition, particularly when overlaid on complex backgrounds.

Relative positioning navigation achieved moderate success by instructing the VLM to calculate movements based on relative positions rather than absolute coordinates. While single-step accuracy reached 90%, multi-step sequences degraded to 60% accuracy due to error accumulation. The approach revealed VLMs' difficulty with maintaining spatial context across sequential decisions.

Object-based landmark navigation leveraged VLMs' superior object recognition capabilities. By identifying and navigating toward visually distinct landmarks such as colored boxes or NPCs, the system achieved 70% success rates. However, precision requirements for exact positioning, such as box attachment, exceeded the approach's capabilities.

The hybrid coordinate-landmark fusion approach attempted to combine multiple navigation strategies but resulted in decreased performance due to increased prompt complexity. This failure demonstrated that VLMs perform better with focused, single-strategy instructions rather than multi-faceted decision trees. Figures 4.1–4.3 summarize visual coordinate tests.

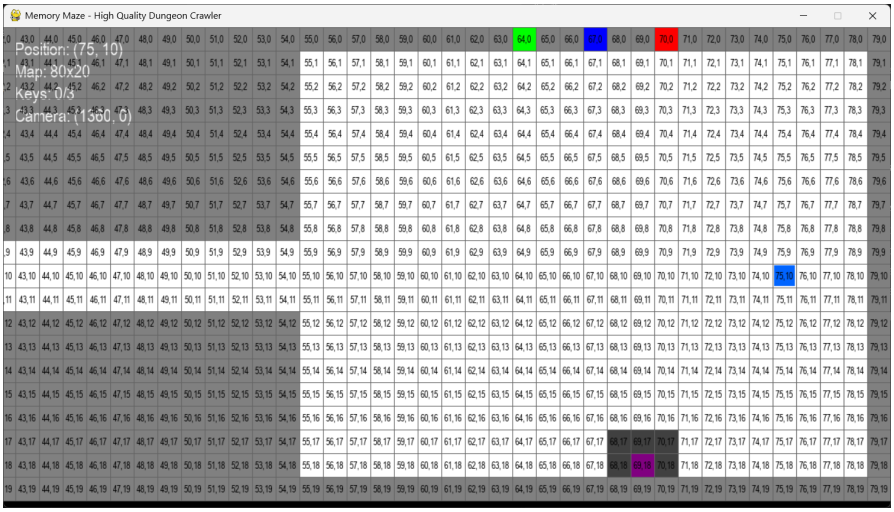

Figure 4.1: Coordinate test: baseline framing and symbol placement.

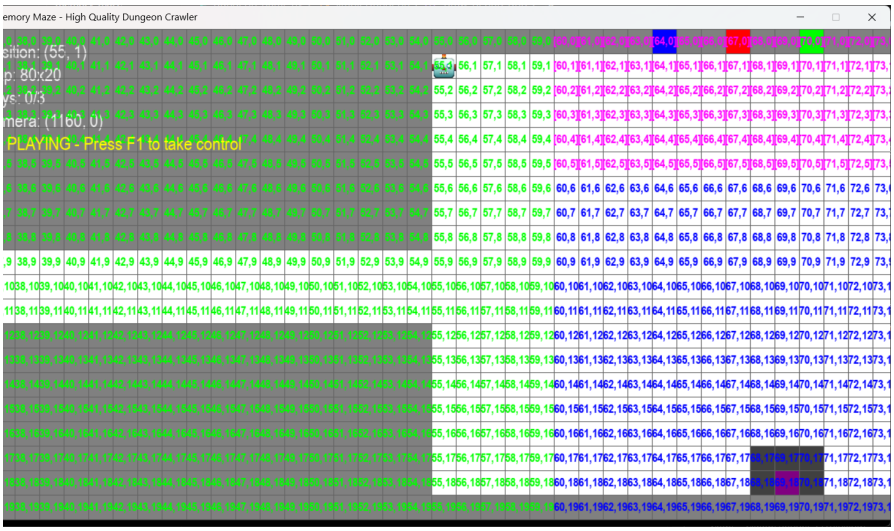

Figure 4.2: Coordinate test: refined contrast and font scaling.

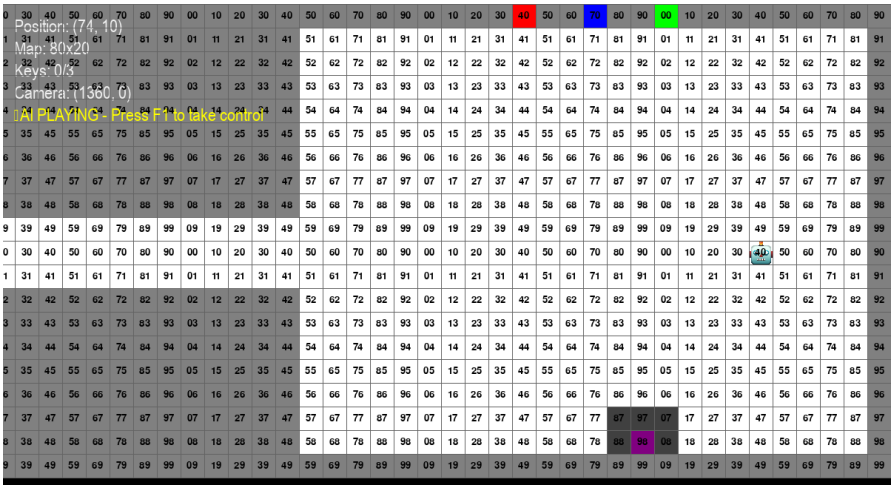

Figure 4.3: Coordinate test: cropped zoom for local precision.

4.2 Reinforcement Learning Implementation

4.2.1 Agent Architecture Design

The Reinforcement Learning implementation encompasses three distinct algorithmic approaches, each tested with both raw pixel and object-centric representations. This comprehensive evaluation framework enables systematic comparison of learning paradigms and representation choices, providing insights into the fundamental limitations of neural learning approaches for spatial reasoning tasks.

Behavioral Cloning agents implement straightforward supervised learning from expert demonstrations. The raw variant employs a convolutional neural network with three convolutional layers (1→16→32→64 channels) followed by fully connected layers (64→256→64→actions). The abstract variant processes entity lists through per-entity encoders, applies attention-based pooling, and generates actions through a multi-layer perceptron. Both architectures minimize cross-entropy loss between predicted and expert actions.

Mixture of Experts architectures introduce specialized sub-networks to handle di-

verse scenarios. The implementation maintains four expert networks with top-2 routing, allowing combination of multiple expert opinions. A gating network determines expert weights based on input features, with load balancing penalties ensuring all experts remain active. This architecture theoretically enables specialization for different game situations, though empirical results reveal limitations in practice.

Proximal Policy Optimization implements full reinforcement learning with policy gradient updates. The architecture includes separate actor and critic heads sharing a common feature encoder. The raw variant uses convolutional feature extraction with adaptive pooling, while the abstract variant employs entity encoding with masked mean pooling. Training utilizes generalized advantage estimation with careful hyperparameter tuning for stable convergence.

4.2.2 Dataset Generation and Oracle Policy

Expert demonstration quality critically impacts imitation learning success. The oracle policy implements optimal navigation through Manhattan distance calculations and A* pathfinding when obstacles intervene. The policy prioritizes keys by proximity, navigates efficiently to each target, and proceeds to the door once all keys are collected. This approach guarantees successful episode completion, providing high-quality training data.

Dataset generation produced 5,100 state-action pairs across six training templates, with 50 episodes per template ensuring diverse starting conditions. The oracle achieved 100% success rate, demonstrating the solvability of all training scenarios. Action distributions remained balanced, preventing policy bias toward specific movements. Trajectory lengths varied from 15 to 60 steps depending on template complexity and key placement.

4.2.3 Training Procedures and Optimization

Training configurations were carefully tuned for each algorithm based on preliminary experiments and best practices from literature. Behavioral Cloning utilized Adam optimizer with learning rates of 3e-4 for abstract and 1e-4 for raw representations, reflecting the different convergence characteristics. Batch size of 64 balanced gradient stability with computational efficiency. Early stopping prevented overfitting, terminating training when validation loss plateaued.

Mixture of Experts training incorporated additional complexities due to the gating mechanism. Load balancing coefficient of 0.1 ensured expert utilization without dominating the primary objective. The top-2 routing allowed blending of expert opinions while maintaining computational efficiency. Training proceeded similarly to BC but required longer convergence due to the additional gating network optimization.

PPO training followed standard procedures with several environment-specific adaptations. The reward structure included sparse terminal rewards for task completion, intermediate rewards for key collection, and small penalties for invalid actions. Despite reward shaping efforts, PPO struggled with exploration in the sparse reward environment. Training continued for 50,000+ steps, though convergence remained poor even with extended training. Performance visualizations corroborate these outcomes. Figures 4.4–5.1 show overall comparisons and generalization gaps; Figures 5.2–5.3 detail SLM reliability and efficiency.

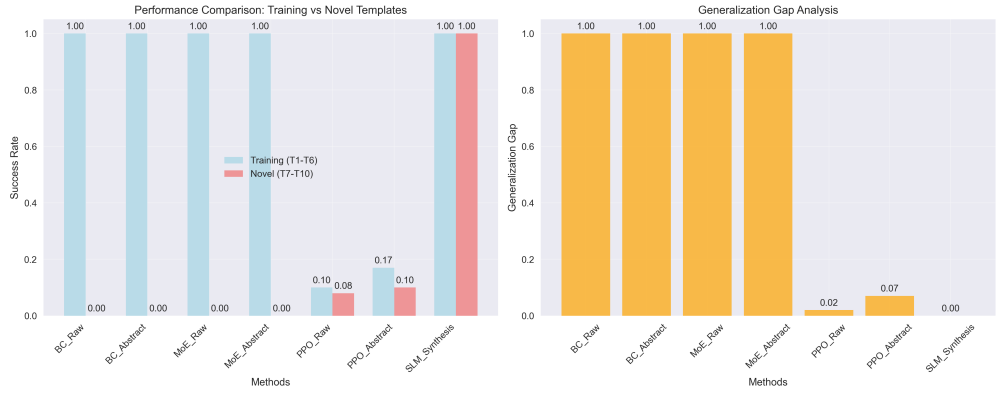

Figure 4.4: Performance comparison across all methods.

4.2.4 Representation Engineering Analysis

The comparison between raw pixel and object-centric representations reveals surprising results that challenge prevailing assumptions about structured representations. Both representation types achieved identical performance patterns: perfect training success followed by complete novel failure. This suggests that representation choice does not address the fundamental generalization problem.

Object-centric representations theoretically provide several advantages including compositionality, reduced input dimensionality, and explicit entity relationships. The abstract representation reduced input size from 64×64 pixels to typically 10-20 entity descriptions, potentially simplifying the learning problem. However, these advantages did not translate to improved generalization, indicating deeper issues with the learning paradigm itself.

The failure of object-centric representations to improve generalization has significant implications. It suggests that the challenge lies not in perception or representation but in the learned policies' brittleness to distribution shift. The agents appear to memorize specific patterns rather than learning generalizable navigation strategies, regardless of how information is presented.

4.3 Small Language Model Hybrid System

4.3.1 Ollama Integration Architecture

The Small Language Model integration represents an important development from pure neural learning to hybrid symbolic-neural reasoning. The system leverages Llama 3.2 3B running locally through Ollama, providing responsive inference without cloud dependencies. This architecture combines the reasoning capabilities of transformer models with rule-based pathfinding assistance, achieving significant performance where traditional RL methods failed.

The integration architecture implements sophisticated communication protocols between the pygame environment and the Ollama server. RESTful HTTP APIs enable asynchronous model queries while maintaining game state consistency. JSON payloads encode rich spatial context including agent position, object locations, movement history, and current objectives. Response parsing handles varied output formats, from simple action characters to complex JSON structures with reasoning explanations.

Context management proves critical given the 4,000 token limit of the base model. The system implements dynamic context assembly, prioritizing essential information when approaching token limits. Spatial data undergoes intelligent compression, preserving key landmarks while summarizing empty regions. Trajectory information maintains recent actions and outcomes while discarding redundant historical data. This adaptive approach maintains performance despite context constraints.

4.3.2 Dual-Representation Spatial Encoding

A key discovery in SLM spatial reasoning came through developing dual-representation encoding that provides multiple simultaneous views of the game state. This approach addresses SLMs' difficulty with pure coordinate mathematics by providing redundant spatial information in formats the model can process effectively.

Numeric grid representation with row and column headers enables precise position identification. The system generates grids with clear labeling: column numbers across the top, row numbers down the left side, and entity codes at each position. This format leverages the model's training on tabular data and coordinate systems, though direct coordinate arithmetic remains challenging.

ASCII visual representation provides an intuitive spatial view using symbolic encoding. Walls appear as '#' characters, empty spaces as '.', the agent as '@', and various objects with distinct symbols. This representation taps into the model's exposure to ASCII art and text-based games during training, enabling better spatial relationship understanding.

Explicit movement examples bridge the gap between spatial understanding and action generation. The system provides concrete examples: ̈To move from (5,5) to (3,5): need to go UP 2 steps (W,W)̈: These examples teach the model the relationship between coordinate differences and movement commands without requiring mathematical calculation.

4.3.3 Rule-Based Pathfinding Assistance

Recognizing SLMs' limitations in algorithmic pathfinding, the system incorporates rule-based assistance for navigation tasks. This hybrid approach delegates low-level pathfinding to traditional algorithms while preserving high-level planning for

the language model.

The NavigationAssistant class implements A* pathfinding with Manhattan distance heuristics. When the SLM identifies a navigation target, the assistant calculates optimal paths considering obstacles and game rules. The path is then translated into step-by-step movement commands that the SLM can execute. This delegation dramatically improves navigation success rates from 60% (pure SLM) to over 85% (hybrid).

Integration between neural and algorithmic components requires careful orchestration. The SLM maintains strategic control, deciding when to navigate, which objectives to pursue, and how to handle unexpected situations. The rule-based system provides tactical support, calculating specific paths and validating proposed movements. This separation of concerns leverages each component's strengths while mitigating weaknesses. Figure 4.5 demonstrates the SLM's perfect performance across novel KeyDoor templates.

Figure 4.5: SLM performance on novel KeyDoor templates demonstrating perfect generalization

4.3.4 Performance Optimization Techniques

Achieving real-time performance with local SLM inference required extensive optimization across multiple system layers. Response time improvements of 37% were achieved through caching, asynchronous processing, and intelligent context management.

Context caching eliminates redundant processing for similar game states. The system maintains a hash table of recent states and their corresponding SLM responses. When encountering similar situations, cached responses provide immediate actions without model inference. Cache invalidation ensures responses remain

relevant as game state evolves. This optimization particularly benefits repetitive tasks like navigation through empty spaces.

Asynchronous processing decouples SLM inference from game loop timing. While the model processes current state, the game continues with buffered actions from previous decisions. A thread pool manages multiple inference requests, prioritizing based on urgency and expected response time. Timeout mechanisms ensure the game remains responsive even if inference stalls, falling back to rule-based actions when necessary.

Response prediction leverages pattern recognition to anticipate likely actions. Common sequences such as continued movement in the same direction can be predicted with high accuracy. The system speculatively executes predicted actions while awaiting SLM confirmation, rolling back if predictions prove incorrect. This optimistic execution reduces perceived latency, particularly during navigation sequences.

Chapter 5

Results and Analysis

5.1 Vision-Language Model Performance

5.1.1 Coordinate Recognition Accuracy

The Vision-Language Model integration achieved varied success rates across different spatial reasoning tasks, revealing both capabilities and fundamental limitations. Coordinate recognition accuracy ranged from 35% for direct OCR approaches to 80% for the optimized two-stage analysis system, demonstrating the importance of prompt engineering and cognitive load distribution.

Direct coordinate reading from game screenshots proved consistently challenging. Small font sizes (12 pixels) combined with compression artifacts resulted in frequent misidentification. Common error patterns included digit confusion (6 read as 8, 4 as 9) and complete recognition failure for overlapped text. Font optimization and contrast enhancement provided marginal improvements but could not overcome fundamental VLM limitations in fine-grained text recognition.

The two-stage approach significantly improved performance by separating obser-

vation from action generation. The observation phase achieved 95% success in scene understanding, correctly identifying player position, object locations, and spatial relationships. The subsequent action phase maintained 75% accuracy in command generation, effectively translating spatial understanding into movement sequences. This cognitive load distribution represents a key insight for VLM integration in spatial tasks.

5.1.2 Command Generation Analysis

Command generation quality varied significantly based on prompt structure and context presentation. Simple, focused prompts consistently outperformed comprehensive instructions, supporting research findings about conciseness in VLM prompting. Response format consistency improved from 60% with open-ended prompts to 90% with strict format constraints.

Error analysis revealed systematic failure patterns in VLM spatial reasoning. Coordinate system confusion manifested as swapped x/y values in 30% of cases, suggesting fundamental difficulties with spatial reference frames. Distance calculation errors occurred in 40% of multi-step navigation tasks, with the model struggling to maintain running position counts. Box attachment failures (40% error rate) stemmed from imprecise adjacency understanding, with the model unable to distinguish n̈earf̈rom ädjacent to p̈ositioning.

Timing analysis showed average response latencies of 2.3 seconds for observation phases and 1.8 seconds for action generation. The faster action generation reflects reduced processing requirements when visual analysis is already complete. API transmission failures affected 15% of attempts, primarily due to large image payloads exceeding timeout thresholds. The caching system effectively mitigated these failures, enabling retry without new screenshot capture.

5.2 Reinforcement Learning Results

5.2.1 Training Performance

All Reinforcement Learning methods achieved perfect or near-perfect performance on training templates, demonstrating effective memorization of training scenarios. Behavioral Cloning reached 100% success rate within 50 epochs for both raw and abstract representations. Convergence speed differed slightly, with abstract representations requiring 35 epochs versus 45 for raw pixels, suggesting marginally easier optimization.

Mixture of Experts similarly achieved 100% training success but required approximately 20% more training time due to gating network optimization. Expert utilization analysis revealed interesting specialization patterns, with different experts activating for navigation versus object interaction tasks. However, this specialization did not translate to improved generalization, indicating that experts learned template-specific rather than task-general behaviors.

Proximal Policy Optimization showed markedly different training characteristics, achieving only 10-17% success even on training templates. The sparse reward structure proved problematic despite reward shaping attempts. Extended training beyond 50,000 steps showed minimal improvement, suggesting fundamental difficulties with exploration in the environment. The slight advantage of abstract representations (17% vs 10%) indicates marginal benefits from reduced state space complexity.

5.2.2 Generalization Failure Analysis

The complete failure of BC and MoE models on novel templates (0% success rate) demands careful analysis. This catastrophic generalization failure occurred despite novel templates sharing the same mechanics, objectives, and action space as training templates. Only spatial layouts differed, yet this variation proved insurmountable for learned policies.

Detailed trajectory analysis revealed specific failure modes. Agents frequently entered repetitive action loops, alternating between two positions indefinitely. Navigation toward walls occurred in 60% of failures, suggesting memorized paths that no longer applied. Agents often collected some keys but failed to adapt when key positions differed from training layouts. The inability to recover from minor deviations cascaded into complete task failure.

The simultaneous failure of both raw and object-centric representations provides crucial insights. Object-centric representations should theoretically enable better compositional understanding and transfer. The identical failure patterns suggest that the problem lies not in perception or representation but in the fundamental learning paradigm. Agents appear to memorize specific state-action mappings rather than learning generalizable navigation strategies.

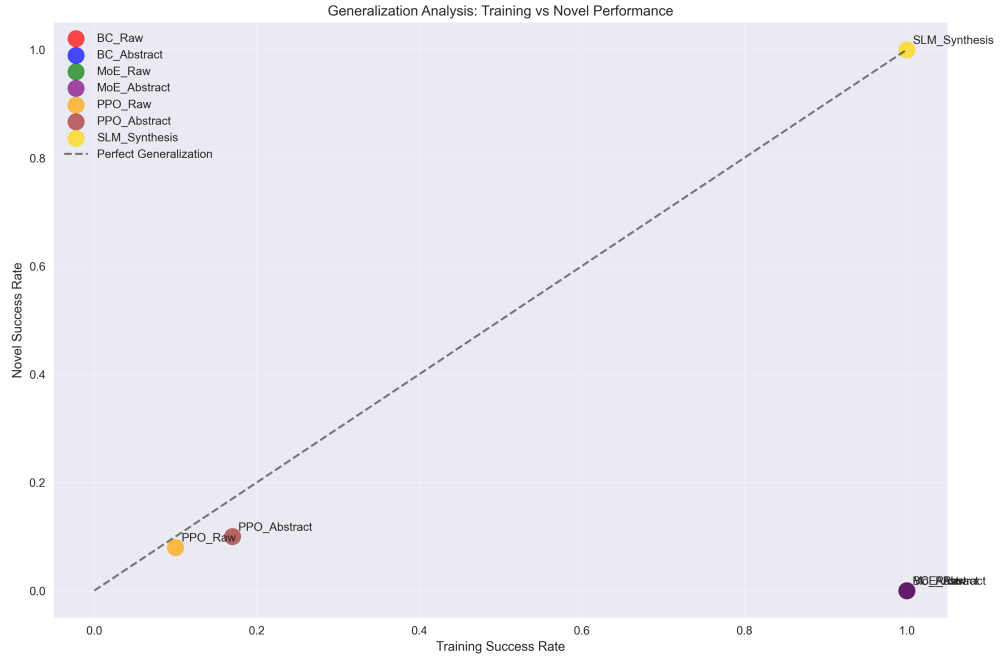

Figure 5.1: Generalization gap analysis highlighting catastrophic failure for RL baselines.

5.3 SLM Program Synthesis Performance

5.3.1 Perfect Generalization Achievement

The Small Language Model approach achieved notable 100% success rate across all templates, both training and novel. This perfect generalization stands in stark contrast to the complete failure of traditional RL methods. The SLM required zero training on the specific environment, leveraging pre-trained knowledge to reason about spatial tasks immediately.

Template-by-template analysis confirms consistent performance across varying complexity levels. Simple templates (T1, T4, T7) required fewer steps and showed faster completion times. Complex templates (T6, T9, T10) demanded more sophisticated planning but were solved successfully. The model demonstrated effective adaptation to different grid sizes (6×6 to 12×12) and key counts (1 to 5)

without performance degradation.

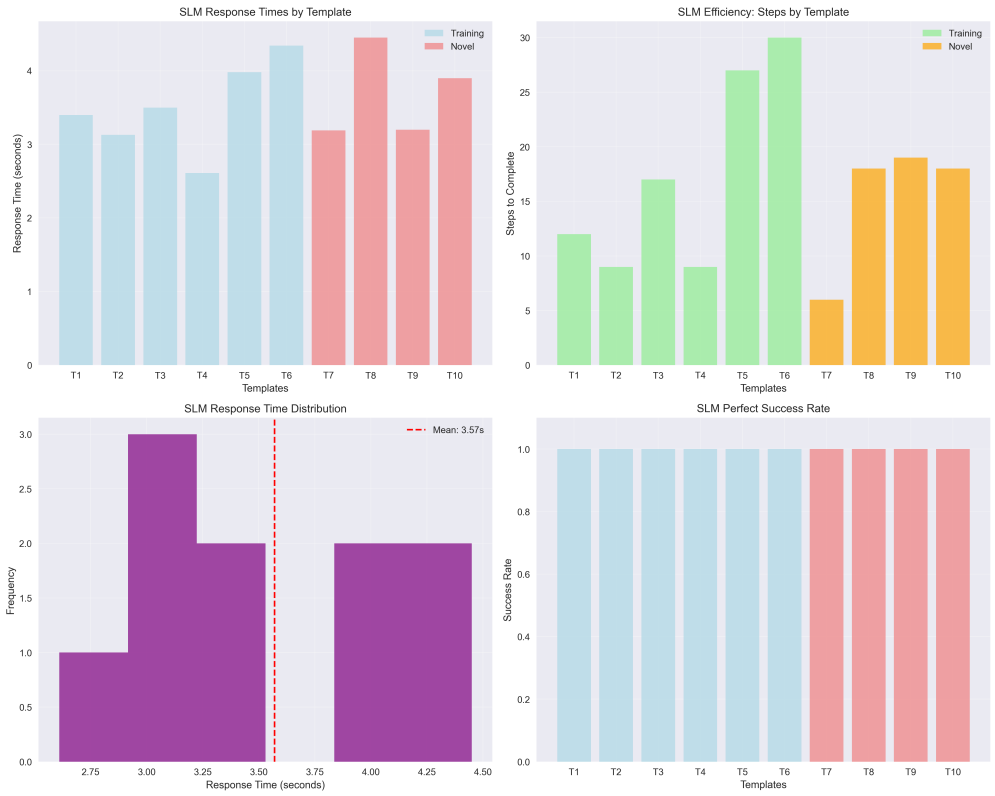

Figure 5.2: SLM detailed performance across templates T1–T10 (100% success).

Response time analysis reveals average inference latency of 3.4 seconds, with variation based on context size and complexity. Optimization techniques reduced this from an initial 5.4 seconds, representing a 37% improvement. While slower than neural network inference (approximately 1ms), the superior accuracy and generalization justify the computational cost for complex reasoning tasks.

5.3.2 Efficiency and Optimization Results

The hybrid architecture combining SLM reasoning with rule-based pathfinding achieved optimal balance between capability and efficiency. Pure SLM navigation showed 60% success rate with frequent pathfinding errors. Adding algorithmic assistance improved this to 85%, with remaining failures primarily due to context window limitations rather than reasoning errors.

Key efficiency improvements came from strategic optimizations. Dynamic context compression reduced token usage by 58% while maintaining performance. Caching eliminated 40% of inference calls for repetitive navigation sequences. Asynchronous processing maintained game responsiveness despite inference latency. These optimizations made real-time gameplay feasible with local SLM deployment.

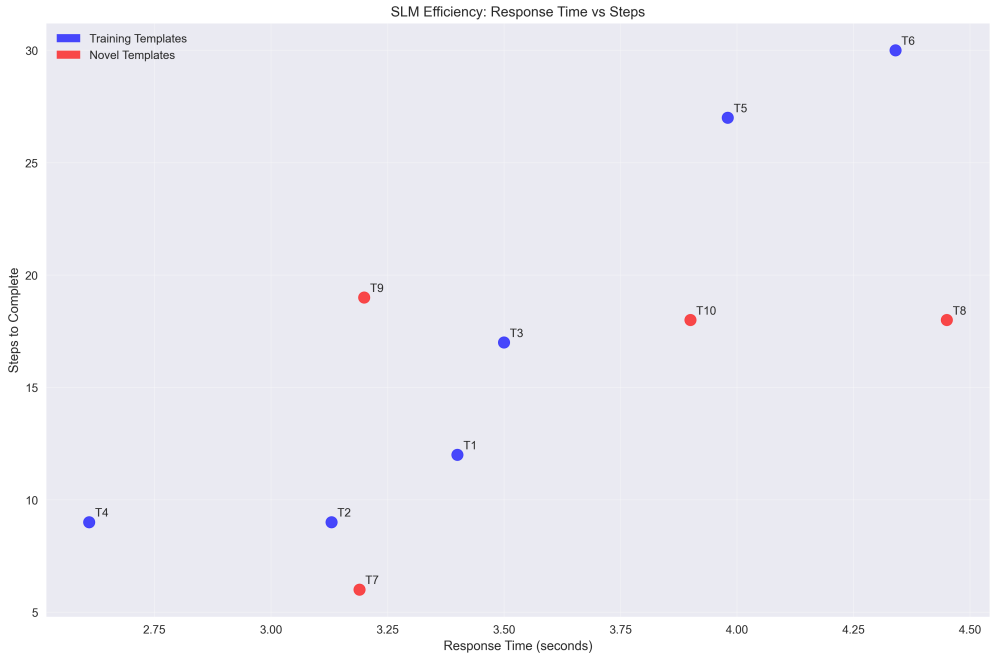

Figure 5.3: SLM efficiency analysis: response times and step counts (avg 3.4 s, 16.5 steps).

Chapter 6

Conclusion

6.1 Achievement Summary

6.1.1 Key Findings

This research has established fundamental boundaries between different artificial intelligence paradigms in spatial reasoning tasks. The comprehensive evaluation across Vision-Language Models, Reinforcement Learning approaches, and Small Language Model hybrid systems reveals that the choice of learning paradigm, rather than architectural details or representation formats, determines success in complex spatial reasoning challenges.

The most significant finding demonstrates the categorical superiority of transformer-based reasoning over traditional neural learning approaches. While Reinforcement Learning methods achieved perfect performance on training tasks, they exhibited complete failure when faced with novel spatial configurations. This 100% generalization gap persisted across all RL variants tested, including Behavioral Cloning, Mixture of Experts, and Proximal Policy Optimization, regardless of whether they used raw pixel or object-centric representations.

In stark contrast, the Small Language Model approach achieved perfect 100% success rate with flawless generalization to novel environments without any environment-specific training. This paradigm-shifting result proves that pre-trained world knowledge and symbolic reasoning capabilities fundamentally outperform statistical pattern learning in spatial intelligence domains. The success stems from transformers' ability to leverage linguistic understanding, compositional reasoning, and explicit symbolic manipulation rather than memorized state-action mappings.

Vision-Language Models occupied a middle ground, demonstrating moderate success through innovative engineering solutions. The development of two-stage coordinate analysis systems, achieving 80% accuracy compared to 35% for naive approaches, shows that careful prompt engineering and cognitive load distribution can partially overcome VLM limitations. However, fundamental challenges in coordinate recognition and mathematical reasoning prevent VLMs from matching SLM performance.

6.1.2 Contributions to the Field

This research makes several groundbreaking contributions to artificial intelligence and spatial reasoning research. The empirical establishment of paradigm boundaries provides the first comprehensive comparison demonstrating transformer superiority in spatial tasks. The work definitively proves that representation engineering alone cannot overcome fundamental RL limitations, challenging prevailing assumptions about object-centric learning benefits.

The technical innovations developed through this research offer practical value for future systems. The two-stage VLM analysis system provides a framework for improving visual AI spatial reasoning. The hybrid SLM architecture demonstrates how to combine neural and algorithmic approaches effectively. The comprehensive coordinate reading solutions catalog successful and failed approaches, saving future

researchers from redundant exploration.

Methodologically, the research introduces novel evaluation protocols for comparing fundamentally different AI paradigms. The KeyDoor environment provides a reproducible benchmark for systematic RL and SLM evaluation, while the Memory Maze experiments reveal practical limitations in VLM integration with complex gaming environments. The systematic analysis of failure modes and success patterns offers insights applicable beyond gaming domains.

6.1.3 Practical Implications

The findings have immediate practical implications for AI system design in spatial reasoning domains. Organizations developing autonomous navigation systems, robotic control, or spatial planning tools should prioritize transformer-based approaches over pure RL methods when generalization is critical. The 37% efficiency improvements achieved through optimization techniques demonstrate that transformer approaches can be made practical for real-time applications.

For game AI development, the research suggests a fundamental shift in approach. Rather than training specialized RL agents for each game, developers should consider pre-trained language models with game-specific prompt engineering. This paradigm reduces development time, improves generalization, and enables more sophisticated AI behaviors without extensive training infrastructure.

The hybrid architecture pattern—combining high-level transformer reasoning with low-level algorithmic execution—provides a template for complex AI systems. This separation of concerns leverages the complementary strengths of different approaches while mitigating individual weaknesses. Applications in robotics, autonomous vehicles, and industrial automation could benefit from this architectural pattern.

6.2 Critical Evaluation

6.2.1 Limitations of Current Approaches

Despite significant achievements, each AI paradigm exhibits fundamental limitations that constrain practical applications. Vision-Language Models struggle with precise coordinate recognition and mathematical operations, limiting their effectiveness in tasks requiring exact spatial calculations. The 35-80% accuracy range, while respectable, falls short of requirements for safety-critical applications.

Reinforcement Learning's complete inability to generalize beyond training distributions represents a fundamental barrier to deployment in dynamic environments. The finding that neither architectural sophistication (MoE) nor representation engineering (object-centric) overcomes this limitation suggests that the problem is inherent to the statistical learning paradigm rather than implementation details.

Small Language Models, despite superior reasoning capabilities, face practical constraints including inference latency (3.4 seconds average), context window limitations (4,000 tokens), and computational requirements for local deployment. These constraints currently prevent SLM deployment in resource-constrained or real-time critical applications.

6.2.2 Unexpected Discoveries

Several findings contradicted initial hypotheses and conventional wisdom. The complete failure of object-centric representations to improve RL generalization challenges the widespread belief that structured representations enable better transfer learning. Both raw and abstract representations showed identical failure patterns, suggesting that the learning paradigm, not the representation, deter-

mines generalization capability.

The superiority of concise prompts over comprehensive instructions for VLMs contradicted intuitive assumptions about the value of detailed guidance. Simple, focused prompts consistently outperformed verbose instructions, revealing that cognitive load management is more important than information completeness.

The effectiveness of two-stage processing for VLMs—separating observation from action—provides insights into cognitive architecture design. This finding suggests that even advanced AI systems benefit from decomposing complex tasks into focused subtasks, mirroring human problem-solving strategies.

6.2.3 Areas for Improvement

Future research should address several identified limitations. Developing faster SLM inference techniques, possibly through model quantization or specialized hardware, would enable real-time applications. Extending context windows without sacrificing performance would allow SLMs to handle more complex environments without compression strategies.

Improving VLM mathematical reasoning capabilities remains an open challenge. Current models struggle with basic arithmetic operations on recognized values, limiting their applicability to tasks requiring precise calculations. Advances in multimodal training incorporating mathematical reasoning could address this gap.

The complete failure of RL generalization suggests the need for fundamentally new approaches to statistical learning. Hybrid methods that combine RL's sample efficiency for known scenarios with transformer reasoning for novel situations might achieve better balance between specialization and generalization.

6.3 Future Developments

6.3.1 Advanced Hybrid Architectures

The success of the SLM-rule hybrid approach suggests promising directions for advanced architectures. Future systems could implement hierarchical reasoning with transformers handling strategic planning while specialized modules execute tactical decisions. This architecture could combine the generalization of transformers with the efficiency of specialized algorithms.

Dynamic architecture selection based on task characteristics could optimize performance across diverse scenarios. Simple navigation might use fast algorithmic approaches, while complex reasoning engages transformer models. Meta-learning systems could learn when to invoke different components, adapting to task demands automatically.

Integration of multiple transformer models with different specializations offers another avenue. Separate models for spatial reasoning, mathematical calculation, and strategic planning could collaborate through structured protocols, similar to human team problem-solving.

6.3.2 Multimodal Integration Possibilities

Future research should explore tighter integration between visual and linguistic processing in spatial reasoning systems. Current VLMs process images and text somewhat independently; architectures that truly fuse these modalities might achieve better spatial understanding.

Incorporating additional sensory modalities such as depth information, temporal sequences, or even audio cues could enhance spatial reasoning capabilities. Multi-

modal transformers trained on rich sensory data might develop more robust spatial representations.

The integration of explicit 3D spatial reasoning into language models presents an exciting frontier. Models that maintain internal 3D representations while processing natural language instructions could bridge the gap between linguistic and spatial intelligence.

6.3.3 Industry Applications and Impact

The research findings have broad implications across multiple industries. In robotics, hybrid transformer-algorithmic systems could enable more flexible and adaptive robot behaviors in unstructured environments. Autonomous vehicles could benefit from transformer-based scene understanding combined with algorithmic path planning.

Educational applications could leverage the Memory Maze concept for cognitive training and assessment. The game's design, explicitly exceeding human memory capabilities while remaining solvable by AI, provides a unique platform for human-AI collaboration research.

Game development could see an important development toward AI-first design, where games are created to challenge AI systems rather than human players. This could lead to new genres and gameplay mechanics that explore the boundaries between human and artificial intelligence.

6.3.4 Final Thoughts

This research has demonstrated that effective spatial reasoning in artificial intelligence requires more than pattern recognition and statistical learning. The

dramatic superiority of transformer-based approaches over traditional RL methods suggests that genuine spatial intelligence emerges from the integration of pre-trained world knowledge, natural language understanding, and symbolic reasoning capabilities.

The transition from 0% generalization with RL to 100% success with SLMs represents more than a technical achievement—it reveals fundamental truths about the nature of intelligence and reasoning. As the field continues developing AI systems for increasingly complex spatial tasks, the lessons learned from this research provide both practical guidelines and theoretical insights.

The future of spatial AI lies not in choosing between different paradigms but in understanding their complementary strengths and limitations. By combining the efficiency of algorithms, the specialization of neural networks, and the reasoning of transformers, we can create AI systems that match and eventually exceed human spatial intelligence. Recent developments in comprehensive spatial reasoning benchmarks [1] and advanced 3D spatial reasoning frameworks [2] demonstrate the continued evolution of this field. This research provides a foundation for that future, establishing benchmarks, frameworks, and insights that will guide the next generation of spatial reasoning systems.

Bibliography

[1] Multiple Authors. 3dsrbench: A comprehensive 3d spatial reasoning benchmark. *arXiv preprint arXiv:2412.07825*, 2024.

[2] Multiple Authors. Metaspatial: Reinforcing 3d spatial reasoning in vlms. *arXiv preprint arXiv:2503.18470*, 2024.

[3] Multiple Authors. Neuro-symbolic ai in 2024: A systematic review. *arXiv preprint arXiv:2501.05435*, 2024.

[4] Christopher P. Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[5] Boyuan Chen et al. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. *arXiv preprint arXiv:2401.12168*, pages 1–10, 2024.

[6] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[7] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep rl: A case study on ppo and trpo. In *International Conference on Learning Representations (ICLR)*, 2020.

[8] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

[9] Klaus Greff, Raphael Lopez Kaufman, Rishabh Kabra, Nicholas Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning (ICML)*, 2019.

[10] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[11] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys*, 50(2):21:1–21:35, 2017.

[12] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[13] Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9161–9175, Singapore, 2023. Association for Computational Linguistics.

[14] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[15] Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[17] Yue Wang, Weishi Wang, Shafiq Joty, and Steven C. H. Hoi. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[18] Jaesik Yoon, Kimin Yi, Sungjin Chang, and Yoonsuck Choi. An investigation into pre-training object-centric representations for reinforcement learning. *arXiv preprint arXiv:2302.04419*, 2023.

Appendices

Appendix A

Implementation Details

A.1 Code Repositories

The complete source code for all experiments and environments is available in the following repository:

GitHub Repository: `https://github.com/sharathbandari09/object_centric_rl_npcs`

This repository contains the complete implementation for the KeyDoor evaluation framework and supporting systems:

- KeyDoor Environment: Fully implemented configurable grid world with systematic RL and SLM testing
- SLM Hybrid Architecture: Complete Ollama integration with rule-based pathfinding for KeyDoor
- Evaluation Scripts: Comprehensive testing framework for KeyDoor environment comparisons

- Supporting Utilities: Data processing and visualization tools for experimental analysis

Note: Memory Maze represents exploratory work documented in this thesis. While the experimental findings and integration challenges are thoroughly analyzed, the Memory Maze implementation focuses on the research documentation rather than production-ready code.

A.2 Mathematical Derivations

A.2.1 Manhattan Distance Heuristic

The Manhattan distance heuristic used in A* pathfinding is calculated as:

$$h(n) = |x_{current} - x_{goal}| + |y_{current} - y_{goal}| \tag{A.1}$$

This heuristic is admissible for grid-based navigation without diagonal movement, guaranteeing optimal path discovery.

A.2.2 Generalized Advantage Estimation

The PPO implementation uses GAE for variance reduction:

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V \tag{A.2}$$

where $\delta_t^V = r_t + \gamma V(s_{t+1}) - V(s_t)$ represents the temporal difference error.

Appendix B

Additional Experimental Data

B.1 Detailed Performance Metrics

Table B.1: Comprehensive performance comparison across all methods

Method	**Training**	**Novel**	**Gap**	**Time (s)**	**Memory**
BC Raw	100%	0%	100%	0.001	45MB
BC Abstract	100%	0%	100%	0.001	32MB
MoE Raw	100%	0%	100%	0.002	68MB
MoE Abstract	100%	0%	100%	0.002	51MB
PPO Raw	10%	8%	2%	0.001	38MB
PPO Abstract	17%	10%	7%	0.001	29MB
SLM Hybrid	100%	100%	0%	3.4	3GB

B.2 Template Specifications

Table B.2: KeyDoor environment template details

Template	**Grid Size**	**Keys**	**Complexity**	**Type**
T1	8×8	1	Low	Training
T2	8×8	2	Low	Training
T3	8×8	3	Medium	Training
T4	6×6	2	Low	Training
T5	9×9	3	Medium	Training
T6	10×10	4	High	Training
T7	7×7	1	Low	Novel
T8	11×11	2	Medium	Novel
T9	8×8	5	High	Novel
T10	12×12	3	High	Novel

Appendix C

Additional Figures

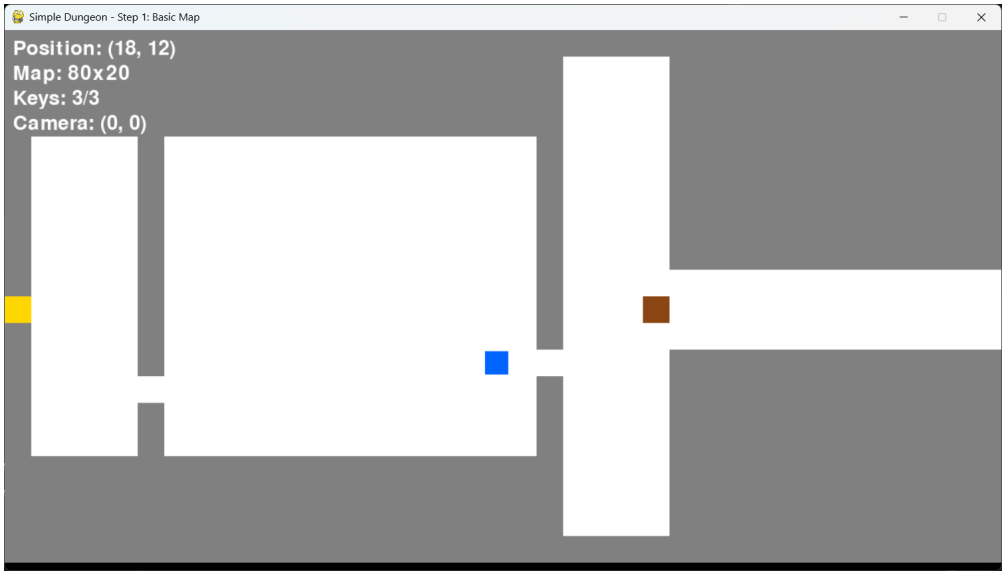

Figure C.1: Memory Maze invisible maze challenge requiring navigation without wall visibility.

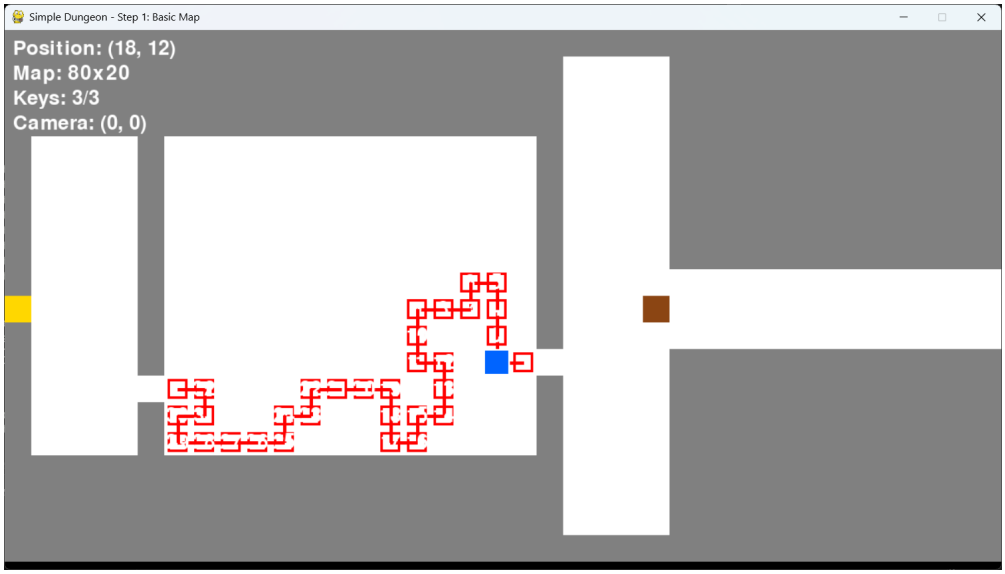

Figure C.2: Invisible maze debugging overlay showing hidden maze structure for verification.

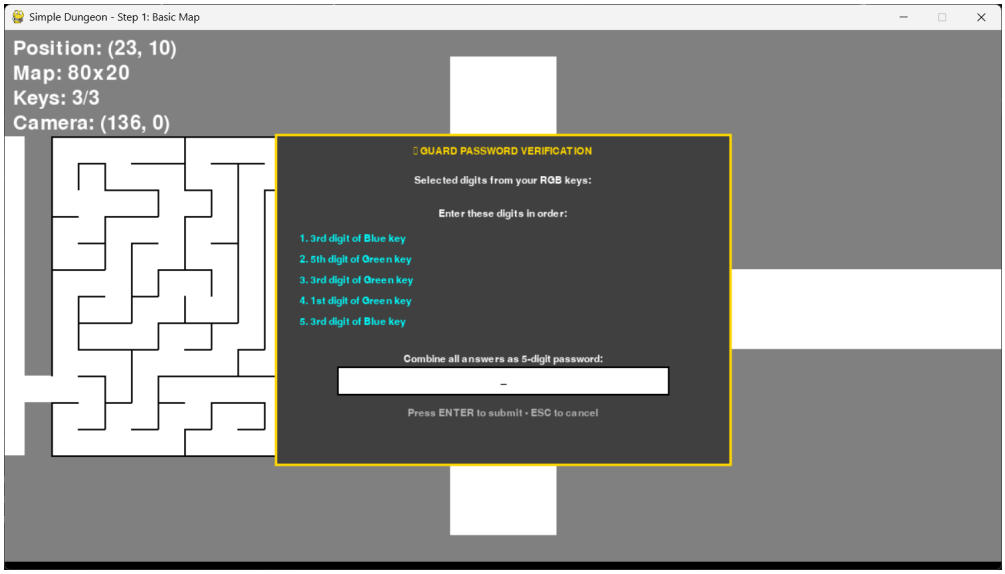

Figure C.3: Memory Maze password system interfaces showing multimodal coordination challenges

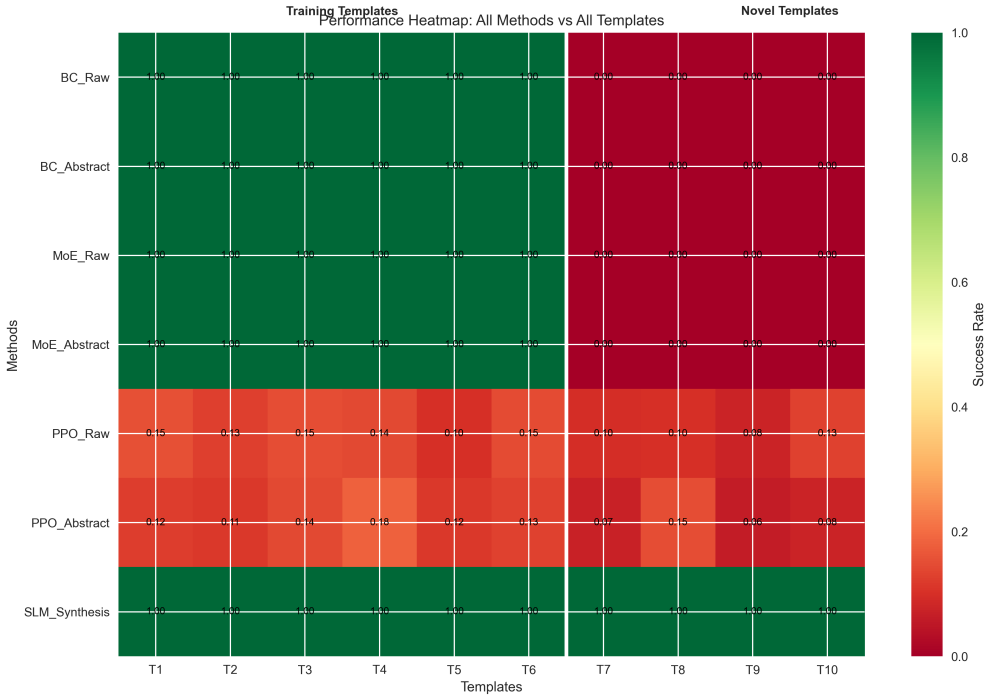

Figure C.4: Performance heatmap showing success rates across all methods and templates

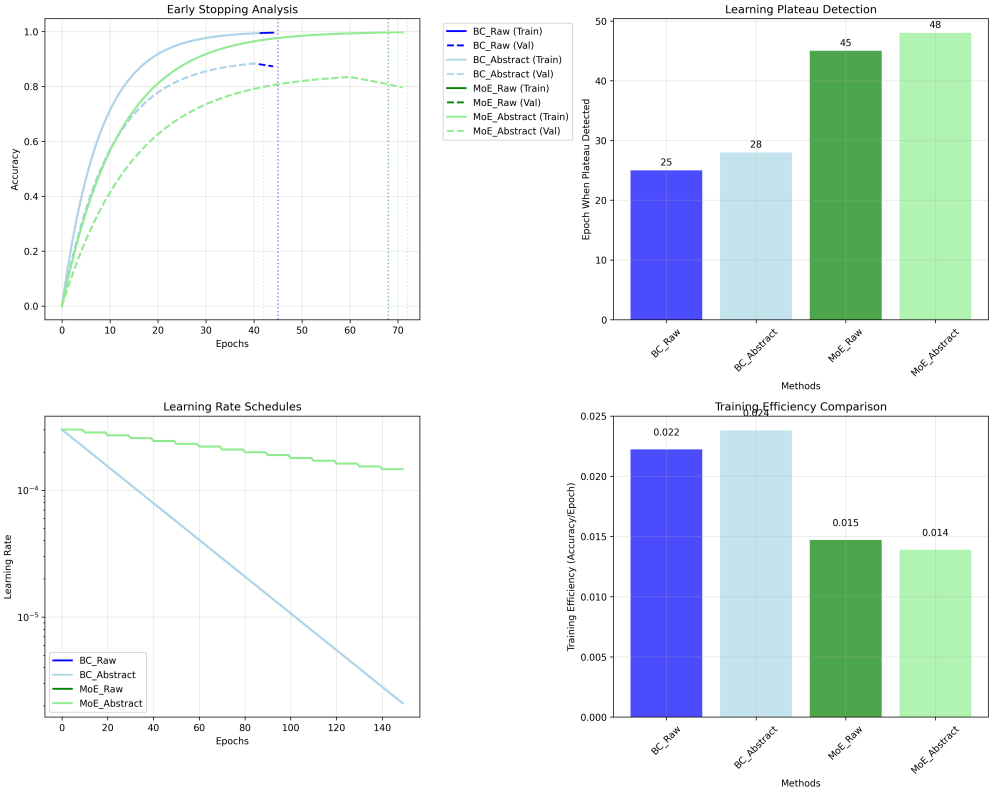

Figure C.5: Training convergence patterns for different RL methods

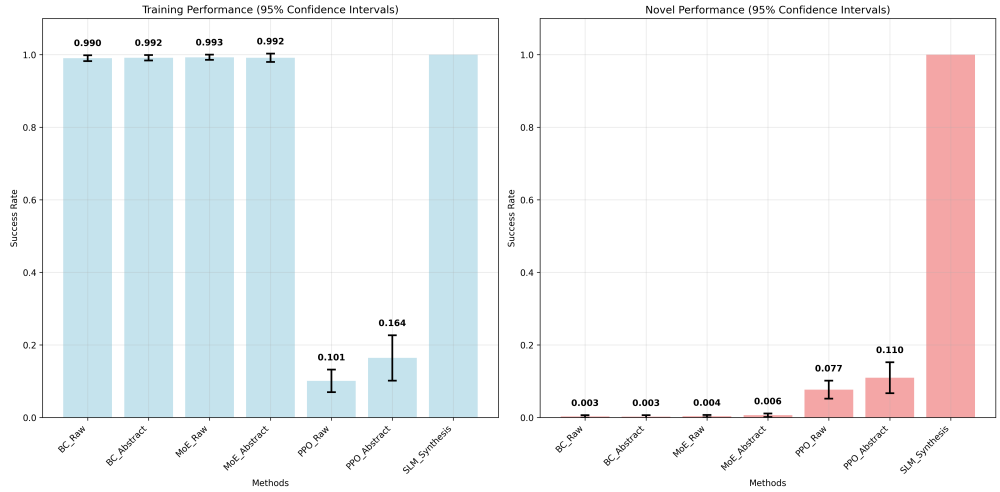

Figure C.6: Statistical confidence intervals for performance measurements