Reinforcement learning-based dual-policy adversarial model for dynamic difficulty adjustment in a turn-based roguelike game

by

Roy Khoo

A Creative Component submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Academic Plan Committee:
Simanta Mitra, Major Professor

The student author, whose presentation of the scholarship herein was approved by the academic plan committee, is solely responsible for the content of this creative component. The Graduate College will ensure this creative component is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2026

This work is licensed under CC BY 3.0 US, by Roy Khoo, 2026.

TABLE OF CONTENTS

Page

LIST OF TABLES iv

LIST OF FIGURES v

ACKNOWLEDGMENTS vi

ABSTRACT vii

CHAPTER 1. Introduction 1

CHAPTER 2. LITERATURE REVIEW 3
2.1 Flow Theory and Player Psychology 3
2.2 Reinforcement Learning Foundations 4
2.3 Proximal Policy Optimization (PPO) 4
2.4 Multi-Armed Bandits 5
2.5 Memory Architecture in Reinforcement Learning 5
2.6 Tactical Game Mechanics 6
2.7 Player Modeling 7

CHAPTER 3. RELATED WORKS 8
3.1 Dynamic Difficulty Adjustments and Adaptive Game AI 8
3.2 Engagement Modeling and Experience-Driven Design 9
3.3 Position Relative to Prior Work 10

CHAPTER 4. DESIGN AND APPROACHES 11
4.1 System Architecture Overview 11
4.2 Game Simulation Environment 12
4.3 Observation and Action Space 13
4.4 Director Component 13
4.4.1 Engagement Reward Design 13
4.4.2 Challenge Reward Design 15
4.4.3 Engagement Metrics 15
4.4.4 Contextual Bandit Meta-Controller 16
4.5 Tactical Component 16
4.6 Proxy Player 18
4.7 Training Pipeline 18
4.8 Deployment 19

CHAPTER 5. RESULTS AND DISCUSSION 25
5.1 Architecture Comparison 25
5.2 Objective Comparison 26
5.3 Training Dynamics 27
5.4 Reward Decomposition 28
5.5 Metric Validation 29
5.5.1 Saturation Curve Analysis 29
5.5.2 Sensitivity Analysis 30
5.5.3 Correlation Analysis 31
5.6 Proxy Player Impact 32

CHAPTER 6. CONCLUSION AND FUTURE WORK 33
6.1 Summary of Contributions 33
6.2 Key Findings 33
6.3 Limitations 34
6.4 Future Work 34
6.5 References 37

LIST OF TABLES

Page

Table 4.1 Engagement reward components with trigger conditions and values 14

Table 4.2 Challenge reward components 15

Table 4.3 Episode-level engagement metrics 16

Table 4.4 Optuna-tuned hyperparameters 19

Table 5.1 Full comparison of twelve training configurations 26

Table 5.2 Architecture comparison aggregated across objectives 26

Table 5.3 Objective comparison using MLP architecture 27

Table 5.4 Reward component decomposition across 1,190 decision points 30

Table 5.5 Saturation curve outputs at $K = 10$ 31

Table 5.6 Pairwise Pearson correlations between episode-level metrics 32

LIST OF FIGURES

Page

Figure 2.1 Flow Theory Graph 3

Figure 4.1 System Architecture Overview 21

Figure 4.2 Into-the-Breach Tilesets 21

Figure 4.3 Input Tensors Overview 22

Figure 4.4 Example grid channels 23

Figure 4.5 Training Pipeline Overview 24

Figure 5.1 MLP training dynamics across 2M timesteps 28

Figure 5.2 LSTM training dynamics across 2M timesteps 29

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to several individuals who made this project possible. First and foremost, Dr. Simantha Mitra for his guidance and support throughout this creative component. I am grateful to Junhyung Shim, the teaching assistant for COMS 5990, whose feedback helped improved the quality and depth of this work.

Special thanks to my fellow MS student, Ishaq Mustafa Khan, with whom I collaborated closely with for this project. I would also like to extend my appreciation to Corbin Graham from Lockheed Martin who provided invaluable project ideas and direction while formulating the problem context and also for providing the opportunity for collaboration to work on this project.

ABSTRACT

Dynamic Difficulty Adjustment (DDA) systems traditionally optimize for challenge, but challenge alone does not guarantee engagement. A player may win at an ideal 50% rate yet still feel bored if games lack tension, close calls, or dramatic reversals. This work presents a reinforcement learning-based adversary for an Into the Breach inspired turn-based game that optimizes for engagement besides difficulty. The system employs a dual-objective architecture, which combines an engagement-based policy seeking to maximize player retention, paired with a challenge policy that maximizes raw difficulty. A contextual bandit meta-controller learns to blend these policies based on player state, easing pressure when players struggle and increasing it when they dominate. Engagement is operationalized through validated metrics including flow zone rate, close call frequency, and grit score. The proposed framework involves a hybrid architecture with two components. Namely, a director level controller manages macro-level parameters such as spawn configurations, unit aggressions and target priorities, while a tactical controller manages micro-level decisions on a per unit basis spanning movements and attacks. The pipeline explores algorithm comparison across PPO, A2C, DQN, QRDQN and TRPO alongside memory architecture such as MLP, LSTM and GTrXL, with the final system deployed as a FastAPI server for real-time game integration. The architecture is designed for future human player studies to confirm that the engagement proxies correlate with subjective player experience.

CHAPTER 1. Introduction

Video games have evolved from static, one-size-fits-all experiences into adaptive systems that respond to individual player behavior. At the heart of this evolution lies Dynamic Difficulty Adjustment (DDA), the practice of modifying game parameters in real time so that each player encounters a level of challenge matched to their skill and preferences. The concept has existed since the early days of commercial gaming, with titles such as Resident Evil 4 and Left 4 Dead employing hand-crafted heuristic systems that track player performance and adjust enemy counts, item drops, or spawn timing accordingly. Despite decades of incremental refinement, the fundamental approach has remained largely the same, monitor a proxy for player success (usually win rate or completion time), compare it against a target, and nudge difficulty up or down.

This traditional paradigm has two significant shortcomings. First, it treats difficulty as a single scalar quantity, typically the probability that the player will fail a given encounter. Second, it assumes that an optimal difficulty level exists at or near a 50% win rate, where the player wins about half the time. Both assumptions are problematic. A player who wins every encounter at 40% remaining health is having a very different experience from one who wins at 95% health, even though both achieve a 100% win rate. Similarly, a player who narrowly survives a tense battle and then recovers is likely more engaged than one who faces a mathematically equivalent challenge distributed uniformly across turns. What players remember and enjoy are moments, the close call, the desperate last stand, the comeback from near-defeat. Traditional DDA metrics are blind to these experiential qualities.

This work addresses that gap by reformulating DDA as an engagement optimization problem rather than a difficulty calibration problem. We introduce a dual-objective reinforcement learning architecture comprising a strategic director that controls enemy spawning and a tactical controller that governs per-unit movement and attack decisions. The director trains two PPO policies, one

optimizing for player engagement and one for raw challenge, blended at runtime by a contextual bandit meta-controller that adapts its mixing strategy to the player's current state. This architecture is designed to address three specific research questions:

RQ1. Which RL architecture works best for this task? We compare feedforward (MLP), recurrent (LSTM), and attention-based (GTrXL) memory architectures to determine whether temporal context improves the Director's ability to create engaging gameplay.

RQ2. Does separating challenge from engagement help? We evaluate whether training two distinct policies (engagement and challenge) and blending them via a contextual bandit produces better engagement outcomes than training a single policy for either objective alone.

RQ3. How can player engagement be quantified without access to player surveys? We design and validate a suite of telemetry-based engagement metrics grounded in flow theory, using mathematical analysis to confirm their internal consistency and independence from traditional performance measures like win rate.

The remainder of this report covers the theoretical background underlying this approach, a survey of related work, the full system design and training methodology, experimental results across twelve configurations, and conclusions with directions for future research.

CHAPTER 2. LITERATURE REVIEW

The development of an adaptive adversarial director requires a multidisciplinary foundation, combining principles of positive psychology with modern advancements in deep reinforcement learning and meta-learning. This section outlines the theoretical frameworks and algorithmic building blocks that comprise the proposed system.

2.1 Flow Theory and Player Psychology

The primary objective of this project is to move beyond win-rate optimization toward the maximization of player engagement. We ground our definition of engagement in Flow Theory by Csikszentmihalyi (1991). Flow represents a state of optimal experience where a participant is in a state of complete, energized, and enjoyable immersion in an activity [2.1]. This state occurs at the intersection of task difficulty and participant skill.

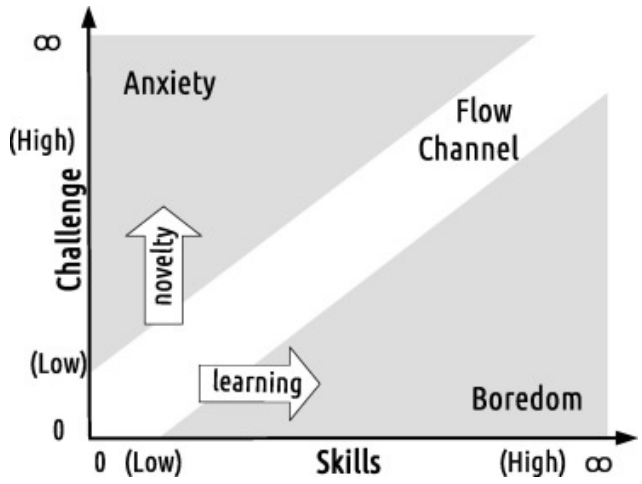


Figure 2.1: Flow Theory Graph

In the context of a tactical roguelike, we operationalize flow as a metric maintaining the player health (HP) between the ranges of 30% and 70%. If the player's HP remains consistently at 100%,

the challenge is insufficient to induce flow (boredom). Conversely, if the HP frequently drops to near-zero without recovery, the challenge exceeds player skill inducing anxiety. Our reward functions are mathematically centered around maintaining this equilibrium.

2.2 Reinforcement Learning Foundations

Reinforcement learning (RL) provides a natural framework for adaptive gameplay by modeling the interaction between an autonomous agent and a dynamic environment. Unlike traditional Dynamic Difficulty Adjustment (DDA) systems that rely on static, developer-authored rules, RL enables an agent to learn optimal strategies through trial and error. This process involves identifying actions that maximize long-term objectives rather than immediate tactical gains. This paradigm is especially suitable for game AI because gameplay unfolds sequentially and current choices affect future states. By shifting from manual tuning to a learned policy, a system can discover complex behaviors that maintain an equilibrium between player boredom and frustration.The standard formalization for these interactions is the Markov Decision Process (MDP), which is defined by the tuple (S, A, P, R, γ). The State (S) represents the environment's current configuration, while the Action (A) space consists of the available moves the agent can execute. The Transition Probability (P) defines the likelihood of moving to a new state given a specific action, and the Reward (R) function provides the scalar feedback used to guide the learning process. The Discount Factor (γ) determines the importance of future rewards. Through this mathematical structure, an RL agent learns to balance short-term tactical value with long-term strategic consequences to achieve a specific behavioral goal.

2.3 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm designed to improve training stability by preventing large, destructive policy updates [Schulman et al. (2017)]. While standard policy gradient methods often suffer from performance collapse, PPO employs a clipped surrogate objective:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)]$$

This mechanism constrains the probability ratio $r_t(\theta)$ between the new and old policy within a safe interval, typically $1 \pm \epsilon$. By limiting the size of policy changes at each step, PPO maintains a stable learning curve and reliable convergence, making it the industry benchmark for high-dimensional tactical environments Berner et al. (2019).

2.4 Multi-Armed Bandits

In scenarios where an agent must choose between multiple distinct strategies or "arms," the Multi-Armed Bandit (MAB) framework provides a mathematical solution for the exploration-exploitation trade-off. Unlike standard RL, which maps states to complex action sequences, a bandit meta-controller typically selects a high-level configuration or "mixing weight" to govern the behavior of underlying systems [Auer (2002)].

In the context of dynamic difficulty adjustment, a contextual bandit can be utilized to observe the current state of a player and select the most appropriate difficulty policy. This approach allows a system to remain flexible, switching between aggressive challenge behaviors and pacing-focused "Engagement" behaviors based on real-time feedback. This hierarchical structure is a recognized method for managing multi-objective optimization problems where different goals may conflict at different stages of an episode [Badia et al. (2020)]. In the system presented in this work, the bandit operates over eleven discrete arms corresponding to a mixing parameter alpha ranging from 0.0 to 1.0 in increments of 0.1, and conditions its selection on four player contexts derived from the aggregate health ratio: struggling, challenged, comfortable, and dominating.

2.5 Memory Architecture in Reinforcement Learning

Standard reinforcement learning architectures often rely on Multi-Layer Perceptrons (MLP), which are inherently reactive and process each game state as an isolated event. This lack of temporal context is a significant limitation for tasks like game directing, where an agent must

recognize long-term patterns such as a player's recovery arc after a series of near-failure events—to make informed pacing decisions. To address this, recurrent architectures like Long Short-Term Memory (LSTM) networks have traditionally been used to maintain a hidden state across time steps. While LSTMs allow for a form of short-term memory, they often struggle with vanishing gradients and representational bottlenecks when tasked with capturing dependencies over long gameplay sequences.

The introduction of the Transformer architecture has provided a more robust solution through the self-attention mechanism, which allows an agent to attend to specific, relevant past states regardless of their distance in time. In the domain of reinforcement learning, the Gated Transformer-XL (GTrXL) has emerged as a specialized variant that integrates Gated Recurrent Units (GRU) to stabilize the training process [Parisotto et al. (2019)]. This architecture enables the AI to maintain a high-capacity "narrative memory," ensuring that decisions are contextually relevant to the player's entire history within a session rather than just the immediate board state. By leveraging these attention-based memories, an adversarial director can orchestrate more sophisticated emotional curves by weighting past player performance against current tactical opportunities.

2.6 Tactical Game Mechanics

Tactical grid-based games represent a subset of strategy games where gameplay is defined by spatial reasoning, resource management, and turn-based decision-making. These environments, exemplified by titles such as Into the Breach, often function as discrete mathematical simulations where players must optimize unit positioning and ability usage on a constrained coordinate system. From a computational perspective, these games provide a rich state space for artificial intelligence research, as they involve perfect information, diverse unit types with unique move-sets, and environmental hazards. Because success in these games depends on anticipating adversary movement and mitigating damage over several turns, they serve as ideal benchmarks for

evaluating how an automated director can manipulate spatial and tactical variables to influence the overall difficulty and pacing of a session.

2.7 Player Modeling

Player modeling is the study of creating abstract representations of a human user's behaviors, preferences, and skill levels to personalize the gameplay experience. In the context of game AI, this typically involves analyzing interaction data to categorize players into specific profiles or clusters, such as those defined by their risk tolerance or mechanical proficiency [Yannakakis et al. (2013)]. Effective modeling allows a system to move beyond universal difficulty settings by predicting how a specific individual will respond to various challenges or pacing shifts. By integrating these models into a dynamic framework, an AI can adapt its behavior in real-time, ensuring that the challenge remains aligned with the user's evolving capabilities and psychological state. The original formulation of this project included a dedicated player modeling component intended to train alongside the adversarial director, with each system adapting to the other's evolving behavior. Due to time constraints and project prioritization, the current implementation substitutes a configurable proxy player with a fixed skill parameter in place of a learned player model. However, player modeling remains a promising direction for future work. Candidate approaches include inverse Q-learning to infer player reward functions from observed behavior and clustering over gameplay telemetry to identify player archetypes defined by strategic preferences and risk tolerance, which could then be used to condition the bandit's context selection on richer features than health ratio alone.

CHAPTER 3. RELATED WORKS

3.1 Dynamic Difficulty Adjustments and Adaptive Game AI

Commercial game AI has a long history of adaptive difficulty systems. The Director AI in Left 4 Dead uses a stress-based spawning model that monitors aggregate team performance metrics such as health, kill rate, and progression speed, then modulates zombie density and item placement along pre-authored intensity curves that alternate between peaks of action and valleys of calm Booth (2009). While effective at creating pacing variety, the system requires extensive manual tuning for each campaign and cannot generalize across player populations without rework. Alien: Isolation employs a dual-layer architecture in which a strategic Director controls macro-level pacing while a separate tactical system governs the alien's moment-to-moment behavior Thompson (2015). This separation of strategic and tactical concerns directly inspired the two-layer architecture in our system, though we replace both layers' rule-based logic with learned policies.

On the academic side, Hunicke and Chapman (2004) formalized DDA with their Hamlet system, which adjusted game parameters based on predicted player performance relative to a target difficulty trajectory. While foundational, Hamlet modeled difficulty as a single continuous variable and did not distinguish between qualitatively different experiences at the same difficulty level. Zohaib (2018) surveyed the broader DDA landscape and categorized approaches into heuristic, machine learning, and player modeling methods, highlighting a persistent gap: most systems optimize for performance metrics like completion rate or time-to-finish, but few attempt to directly optimize for engagement. Within RL-based DDA specifically, Ishii et al. (2018) trained agents to adjust parameters in a fighting game targeting a desired win rate, and Andrade et al. (2006) proposed a framework where an RL agent selected from discrete difficulty levels. Both demonstrated that RL can adapt difficulty online, but neither addressed the distinction between difficulty and engagement.

In competitive game AI more broadly, OpenAI Five demonstrated that PPO-based policies trained via self-play could achieve professional-level performance in Dota 2 [Berner et al. (2019)], and AlphaStar reached Grandmaster-level play in StarCraft II through a combination of supervised learning and multi-agent RL [Vinyals et al. (2019)]. These successes established PPO as a practical algorithm for high-dimensional sequential decision problems, but the agents optimize exclusively for winning. Our work applies PPO to a fundamentally different objective: managing the player's experience rather than defeating them. The hierarchical structure of our system, where a meta-controller selects among subordinate policies, draws on DeepMind's Agent57, which used a multi-armed bandit to select between a family of behavioral policies with varying exploration-exploitation trade-offs [Badia et al. (2020)]. We adapt this principle by treating engagement and challenge as two behavioral modes and using a contextual bandit conditioned on player state to blend between them.

3.2 Engagement Modeling and Experience-Driven Design

A separate line of research has focused on computationally modeling player experience rather than optimizing for performance metrics. Yannakakis and Togelius (2011) surveyed experience-driven procedural content generation and identified three categories of player experience models: subjective models based on self-report, objective models using physiological signals, and gameplay-based models derived from in-game telemetry. Our engagement metrics fall in the third category, using health trajectories, damage events, and outcome quality as proxies for engagement.

Cowley et al. (2008) investigated the relationship between physiological arousal and game difficulty, finding an inverted-U pattern consistent with flow theory: moderate challenge produced the highest engagement, while both extremes led to disengagement. This empirical finding supports our flow zone reward design, where health states in the moderate-challenge band are rewarded and both extremes are penalized. Lopes and Yannakakis (2019) explored deep RL for procedural content generation where an agent learned to generate game levels that optimized for predicted player experience, introducing the idea of using experience-driven metrics as RL

rewards. Their work focused on offline level generation rather than real-time difficulty modulation during play, but the principle of rewarding experience over performance directly informs our engagement policy's reward function.

Bontchev (2016) reviewed adaptation mechanisms in serious games and educational software, distinguishing between challenge adaptation (adjusting what the player faces), assistance adaptation (adjusting what help is available), and content adaptation (adjusting what the player sees). Our dual-objective approach spans these categories: the challenge policy corresponds to challenge adaptation, while the engagement policy incorporates elements of all three through its multi-component reward that considers not just the magnitude of difficulty but the quality of the experience it produces.

3.3 Position Relative to Prior Work

Our work differs from prior DDA research in three key respects. First, we explicitly separate challenge and engagement into distinct optimization objectives rather than treating engagement as a byproduct of appropriate challenge. Second, we use a bandit-based meta-controller to adaptively mix these objectives based on player context rather than relying on a fixed blending rule or a single composite reward. Third, we employ a two-layer architecture separating strategic spawning decisions from tactical per-unit control, inspired by the Director and alien split in Alien: Isolation but implemented through learned policies rather than hand-crafted rules. While each of these individual ideas has precedent in the literature, their combination into a unified trainable architecture for real-time DDA is, to our knowledge, novel.

CHAPTER 4. DESIGN AND APPROACHES

4.1 System Architecture Overview

The proposed system is structured as a two-layer architecture that separates macro-level strategic decisions from micro-level tactical execution. The strategic layer, referred to as the Director, controls enemy spawning parameters: how many enemies appear each turn, their types, grid positions, aggression levels, and target priorities. The tactical layer operates downstream of the Director, receiving the Director's aggression and priority assignments and translating them into concrete per-unit decisions about where each enemy moves and what it attacks. This separation mirrors the dual-layer design found in Alien: Isolation's AI [Thompson (2015)], but replaces hand-authored rules with learned policies at the strategic level.

At each turn during gameplay, the flow proceeds as follows. The game simulation presents its current state, consisting of an 8-by-8 grid with terrain, unit positions, and health values alongside 22 scalar features summarizing game context. The Director's contextual bandit meta-controller observes the player's aggregate health ratio and selects a mixing parameter alpha from eleven discrete values. With probability alpha, the challenge policy's spawning action is used, while with probability one minus alpha, the engagement policy's action is used. The selected action determines enemy spawning decisions for that turn. The tactical layer then resolves each enemy unit's movement and attack based on the Director's aggression and priority assignments. Finally, the player (or proxy player during training) executes their turn, and the cycle repeats.

The Director component is the primary focus of this work, housing the dual-objective reinforcement learning architecture and the bandit-based policy mixing mechanism. Therewith the system architecture is presented over at Figure 4.1

4.2 Game Simulation Environment

The training environment is implemented as a faithful recreation of Into the Breach's core mechanics on an 8-by-8 grid [Figure 4.3]. Each episode consists of up to eight turns with three player controlled mechs, up to four destructible buildings that share a collective power grid, and dynamically spawned enemy units. The simulation supports five enemy types, each with distinct movement patterns, attack ranges, and behavioral characteristics: basic melee units, ranged units that attack from a distance, chargers that move in straight lines, artillery that strikes indirectly, and flying units that ignore terrain restrictions. Terrain features include water hazards that instantly kill grounded units, mountains that block movement, forest tiles, and open ground.

Gameplay follows a fixed turn structure: the Director spawns enemies at designated grid edges, enemies telegraph their intended attacks by highlighting target tiles on the grid, the player responds by moving mechs and using abilities to disrupt or mitigate enemy actions, and finally enemies execute their telegraphed attacks. The push mechanic, a defining feature of Into the Breach's tactical depth, is fully implemented. Attacks can displace units into adjacent tiles, creating emergent tactical situations. An enemy pushed into water is destroyed, a unit pushed into another unit causes collision damage to both, and a unit pushed into a building damages both the unit and the structure. This mechanical richness is important because it means the Director's spawning decisions interact with the board state in complex ways. Placing an enemy adjacent to a water tile creates a fundamentally different tactical situation than placing it in open ground, even if both positions are equidistant from the player's mechs.

The simulation is implemented in skeleton_game.py as a Gymnasium-compatible environment, allowing direct integration with standard RL training libraries. A known limitation is the sim-to-real gap: the simulation approximates but does not perfectly replicate the undergraduate game team's engine. An observation adapter bridges this gap at inference time by translating between the simulation's internal state representation and the live game's JSON format.

4.3 Observation and Action Space

The observation space combines spatial and scalar information to give the policy a comprehensive view of the current game state. The spatial component consists of N binary 8-by-8 grid channels encoding terrain types (ground, water, forest, burning, mountain, crumbled), unit positions by type (combat mech, cannon mech, artillery mech, enemy ranged, enemy web, enemy flying), building locations, telegraphed attack zones, player attack range, enemy attack range, and a spawn mask indicating valid spawn locations. These channels are flattened into a single vector of length N times 64. A scalar vector of M values captures aggregate game context: total mech HP ratio, turn number, power grid level, cumulative damage dealt and received, and binary flags for recent close calls and recovery events. The two components are concatenated into a single input vector of dimension (N times 64) + M.

A legal action mask is applied to the position logits before the policy selects an action, ensuring that the Director can only spawn enemies on valid tiles (unoccupied grid edges). The masked observation then passes through the memory encoder (MLP, LSTM, or GTrXL depending on the configuration) and into the policy network, which outputs a MultiDiscrete action specifying enemy type, grid coordinates (x, y), and target priority for up to three spawns per turn.

4.4 Director Component

The Director is the core contribution of this work. It houses two PPO policies trained with different reward functions, a contextual bandit that selects the mixing parameter between them, and configurable reward structures for both engagement and challenge objectives.

4.4.1 Engagement Reward Design

The engagement reward function combines six distinct components that together capture the experiential qualities associated with engaging gameplay. Each component is parameterized by configurable constants defined in the EngagementRewardConfig dataclass, allowing the reward to be tuned without modifying code. Table 4.1 summarizes the components.

Table 4.1: Engagement reward components with trigger conditions and values

Component	**Condition**	**Reward**	**Purpose**
Flow zone	HP in 30–70%	$+1.0$	Maintain optimal challenge
Flow penalty (below)	HP $< 30\%$	$-1.0 \times d \times 2.0$	Discourage overwhelming
Flow penalty (above)	HP $> 70\%$	$-1.0 \times d \times 1.0$	Discourage passive play
Adaptive ease-off	HP $< 30\%$, no spawns	$+0.3$	Back off when struggling
Adaptive pile-on	HP $< 30\%$, spawns ≥ 3	-0.5	Penalize overwhelming
Adaptive pressure	HP $> 80\%$, spawns ≥ 1	$+0.2$	Push when coasting
Adaptive passive	HP $> 80\%$, no spawns	-0.2	Penalize inaction
Close call	HP $< 25\%$, alive	$+0.5$	Create tension moments
Recovery	HP was $< 30\%$, now $+10\%$	$+0.3$	Reward comeback arcs
Mutual combat	Both sides deal damage	$+0.3$	Encourage active fights
Close victory	Win, HP $< 50\%$	$+3.0$	Reward exciting finishes
Easy victory	Win, HP $> 70\%$	-0.5	Penalize boring wins
Early defeat	Loss before turn 4	-2.0	Penalize crushing player
Late defeat	Loss after turn 4	-1.0	Acceptable late loss

The flow zone component is the largest contributor. When the player's aggregate health ratio falls between 30% and 70% of maximum, a reward of +1.0 is granted. When health falls below this range, a penalty proportional to the distance below the lower bound is applied, scaled by a below-flow multiplier of 2.0 to strongly discourage situations where the player is being overwhelmed. When health exceeds the upper bound, a milder penalty with a multiplier of 1.0 is applied, reflecting that being too safe is less harmful than being too threatened.

The adaptive spawning component rewards context-appropriate behavior. When the player is struggling (health below 30%), the Director receives +0.3 for not spawning new enemies and -0.5 for spawning three or more. When the player is coasting (health above 80%), the Director receives +0.2 for spawning at least one enemy and -0.2 for remaining passive with fewer than two enemies on the field. This creates an asymmetric pressure: the system is more strongly punished for overwhelming a struggling player than for being passive with a comfortable one.

Close calls (+0.5 when health drops below 25% without reaching zero), recovery arcs (+0.3 when health rebounds from below 30%), and mutual combat (+0.3 when both sides deal damage) reward the experiential patterns that create memorable gameplay moments. At episode end,

outcome quality is scored on a gradient: close victories (health below 50%) receive +3.0, easy victories (health above 70%) receive -0.5, early defeats receive -2.0, and late defeats receive -1.0.

4.4.2 Challenge Reward Design

The challenge policy's reward function is deliberately simple, optimizing for raw aggression. It grants +0.2 per point of damage dealt to player mechs, +0.5 per point of building damage, +1.0 per building destroyed, and +2.0 per mech disabled. Enemy losses are penalized at -0.3 per enemy killed. A large bonus of +10.0 is granted if the player is defeated entirely. This creates a policy that plays to win as aggressively as possible, serving as the counterweight to the engagement policy's more nuanced optimization. [Table 4.2]

Table 4.2: Challenge reward components

Component	**Trigger**	**Reward**
Damage to player	Per point of mech damage	+0.2
Building damage	Per point of building damage	+0.5
Building destroyed	Per building destroyed	+1.0
Mech disabled	Per mech at 0 HP	+2.0
Enemy killed	Per enemy killed by player	−0.3
Player defeated	All mechs/buildings lost	+10.0
Player won (easy)	Player wins, HP $>$ 30%	−3.0
Player won (close)	Player wins, HP $<$ 30%	−1.0

4.4.3 Engagement Metrics

Beyond the per-step reward function, several episode-level metrics are computed for evaluation. These metrics are not used during training but provide interpretable summaries of episode quality.

The engagement score is a weighted combination: 0.4 times flow zone rate, plus 0.3 times the close call component (scaled by 10 and capped at 1.0), plus 0.2 times combat rate (scaled by 5 and capped at 1.0), plus 0.1 times outcome quality. The grit score uses a Michaelis-Menten saturation curve of the form damage/(damage + K) with K = 10, multiplied by a survival factor and

augmented by a close-call bonus capped at 0.3. The tension score captures the average magnitude of HP change per turn, quantifying how dynamic the experience is.

Table 4.3: Episode-level engagement metrics. Variables: f = flow zone rate, c = close call component (scaled by 10, capped at 1.0), m = combat rate (scaled by 5, capped at 1.0), o = outcome quality (categorical: close win = 1.0, moderate win = 0.7, easy win = 0.5, late loss = 0.3, early loss = 0.2), d = cumulative damage to player, s = survival multiplier (1.0 if won, 0.5 if lost), b = close call bonus (0.1 per close call, capped at 0.3)

Metric	**Formula**	**Target**
Engagement	$0.4f + 0.3c + 0.2m + 0.1o$	Higher = better
Flow zone rate (f)	turns in [0.3, 0.7] / total	Higher = better
Grit	$\min(\frac{d}{d+10} \cdot s + b,\ 1.0)$	Higher = harder
Tension	mean($\lvert\Delta\text{HP}\rvert$)	Higher = dynamic
Close call rate	near-death / total turns	Moderate
Win rate	victories / episodes	$\approx 50\%$

4.4.4 Contextual Bandit Meta-Controller

The meta-controller uses a contextual bandit with Upper Confidence Bound (UCB) arm selection to choose the policy mixing parameter alpha at each turn.

The context space consists of four discrete states defined by the player's aggregate health ratio: struggling (below 30%), challenged (30% to 50%), comfortable (50% to 70%), and dominating (above 70%). Each context maintains independent arm statistics, so the bandit effectively learns four separate policies for how to mix challenge and engagement under different player conditions. The action space consists of eleven arms corresponding to alpha values from 0.0 to 1.0 in increments of 0.1. UCB arm selection balances exploitation of high-reward arms with exploration of under-sampled arms, with an exploration constant that decays over episodes from 1.0 toward a minimum of 0.1.

4.5 Tactical Component

The tactical component serves as the execution layer of the hybrid adversary, converting the director's high-level pressure decisions into concrete per-enemy actions on the board. The

controller operates over the complete game state, encompassing terrain layout, occupied positions, living units on both sides, and the director's current directive (aggression level, target priority). Its responsibility is distinct from the director's: it does not decide whether the system should apply more challenge or engagement-focused pressure, but rather determines how that decision manifests through individual enemy movement and combat on the grid. The controller implements a search-and-score policy that evaluates the full space of legal actions for each living enemy on every turn. For a given enemy, it first computes all reachable positions by expanding outward from the unit's current location, accounting for movement allowance, web status (which immobilizes grounded units), terrain traversability (flying units ignore water and mountains while grounded units do not), and tiles reserved by other enemies earlier in the same resolution pass. From each candidate position, the controller evaluates legal attack targets based on type-specific rules: web and flying enemies attack adjacent tiles, while ranged enemies attack along cardinal lines subject to projectile blocking by mountains. A legal action mask filters out any invalid combination, ensuring the controller never produces an illegal move or attack.

Action selection is governed by a scoring function that combines local tactical opportunity with the director's strategic intent. Buildings receive higher scores when the director prioritizes infrastructure pressure, while weakened mechs score higher when the target priority is set to weakest unit. The director's aggression parameter enters as a continuous bias: at high aggression, the controller favors closing distance and dealing immediate damage even at the cost of exposure; at low aggression, it favors maintaining threatening positions while avoiding the player's projected attack zones. This means the same board state can produce substantially different enemy behavior depending on the strategic context, without requiring separate policy networks. The controller's output follows a staged resolution format: if an attack is available from the current position, it emits an attack or web action directly; if a move enables an attack, it emits a compound move-and-attack; otherwise, it generates a repositioning move toward the highest-priority objective. This staged format produces interpretable actions where each output explicitly encodes whether the enemy is attacking in place, advancing to strike, or repositioning for future pressure.

The architectural separation between strategic and tactical layers means the director does not need to reason about tile-level pathfinding or attack geometry, and the tactical controller does not need to optimize long-horizon engagement or manage episode-level pacing. The two layers form a complementary hierarchy: the director specifies the desired pressure pattern, and the tactical controller translates that pattern into board-level enemy behavior. This modularity also means either layer can be independently improved or replaced, as the tactical controller could be substituted with a learned policy, behavior trees, or MCTS-based planning without modifying the director's architecture or training pipeline.

4.6 Proxy Player

Training an RL-based Director requires a simulated opponent that approximates human play. The proxy player uses a skill-weighted heuristic system where a configurable skill parameter between zero and one controls decision quality. At skill 1.0, the proxy player always selects the highest-scoring action available; at skill 0.0, it selects uniformly at random. Intermediate values interpolate between these extremes using a softmax distribution over action scores, with the temperature inversely proportional to skill. Action scoring considers damage dealt to enemies, threat avoidance (moving away from telegraphed attacks), building protection (interposing mechs between enemies and buildings), and push exploitation (displacing enemies into water or into each other). The default training skill of 0.7 approximates an average-to-above-average human player. A known limitation is that the proxy player proved too effective relative to the Director's spawning capacity, achieving a 98% win rate across all configurations. This suppresses the close call reward component and is discussed further in Section 5.

4.7 Training Pipeline

The experimental comparison spans twelve configurations across two stages. Stage 1 selects the best-performing RL algorithm from PPO, A2C, DQN, QRDQN, and TRPO using an MLP policy network. Stage 2 takes the winning algorithm (PPO) and compares three memory

architectures: MLP (feedforward baseline), LSTM (sequential history via sb3-contrib RecurrentPPO), and GTrXL (attention over time via RLlib).

All configurations were trained for two million timesteps on the Iowa State Nova HPC cluster using GPU-accelerated nodes with eight parallel environments. Hyperparameters were tuned using Optuna with 50 trials per configuration [Table 4.4]. The best-performing hyperparameters for the MLP were: hidden layers [128, 256] with tanh activation, learning rate 3.0e-4, batch size 32, rollout length 256 steps, 7 PPO epochs per update, discount factor 0.9713, clipping parameter 0.1685, and entropy coefficient 0.008. The GTrXL configuration used 4 attention heads with dimension 32, a memory length of 50, and sequence length of 20. Training telemetry was logged to Weights and Biases for monitoring convergence. A custom callback recorded engagement metrics at the end of each episode for MLP and LSTM configurations.

Table 4.4: Optuna-tuned hyperparameters

MLP/LSTM	**Value**	**GTrXL**	**Value**
Hidden layers	[128, 256]	Attn heads	4
Activation	tanh	Attn dim	32
Learning rate	3.0e-4	Memory len	50
Batch size	32	Seq length	20
n_steps	256		
PPO epochs	7		
γ	0.9713		
Clip range	0.1685		
Entropy coeff	0.008		

4.8 Deployment

The trained model is deployed as a FastAPI server on a university Linux VM. The server exposes a POST endpoint at /decide that accepts game state JSON conforming to the undergraduate game team's API contract and returns enemy action decisions. Three inference modes are supported: mock (random decisions for testing), heuristic (rule-based decisions), and rl

(trained model with bandit alpha selection). A /stats endpoint returns current bandit statistics including per-context arm values and the current alpha.

The architecture supports two deployment modes. In the current Director-only mode, the server sends spawn requests with aggression and priority parameters, and the game engine's built-in AI handles unit movement using those parameters. In the planned combined mode, the Tactical Controller generates explicit unit_commands (move_to, attack, ability) for each enemy unit, giving the AI system full control over enemy behavior. This modular design allows incremental integration: the Director can operate independently while the tactical component is developed and tested separately.

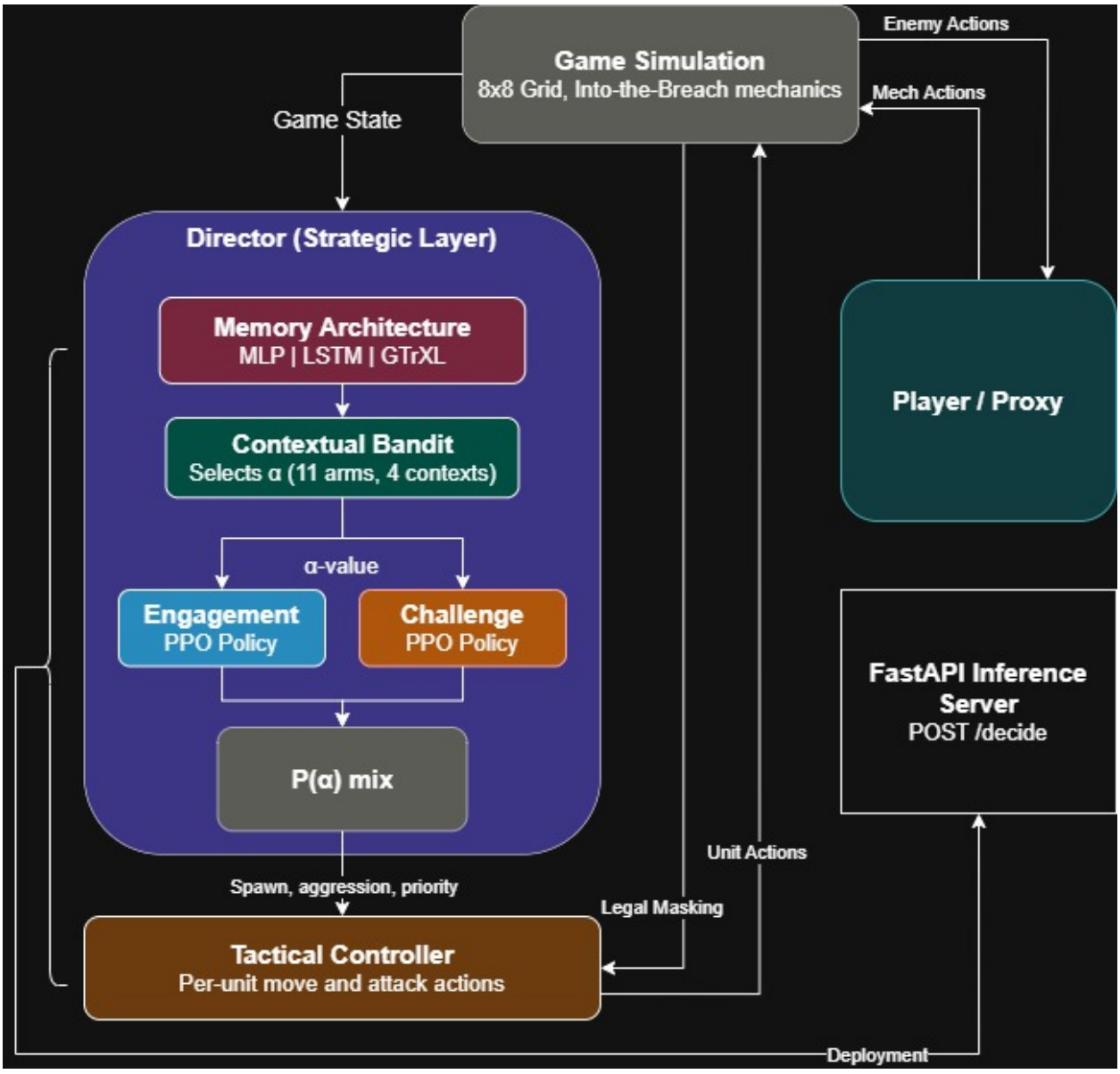


Figure 4.1: System Architecture Overview

alt=Figure showing Into-the-Breach tilesets including buildings, ally unit, enemy unit in a 8x8 board

Figure 4.2: Into-the-Breach Tilesets

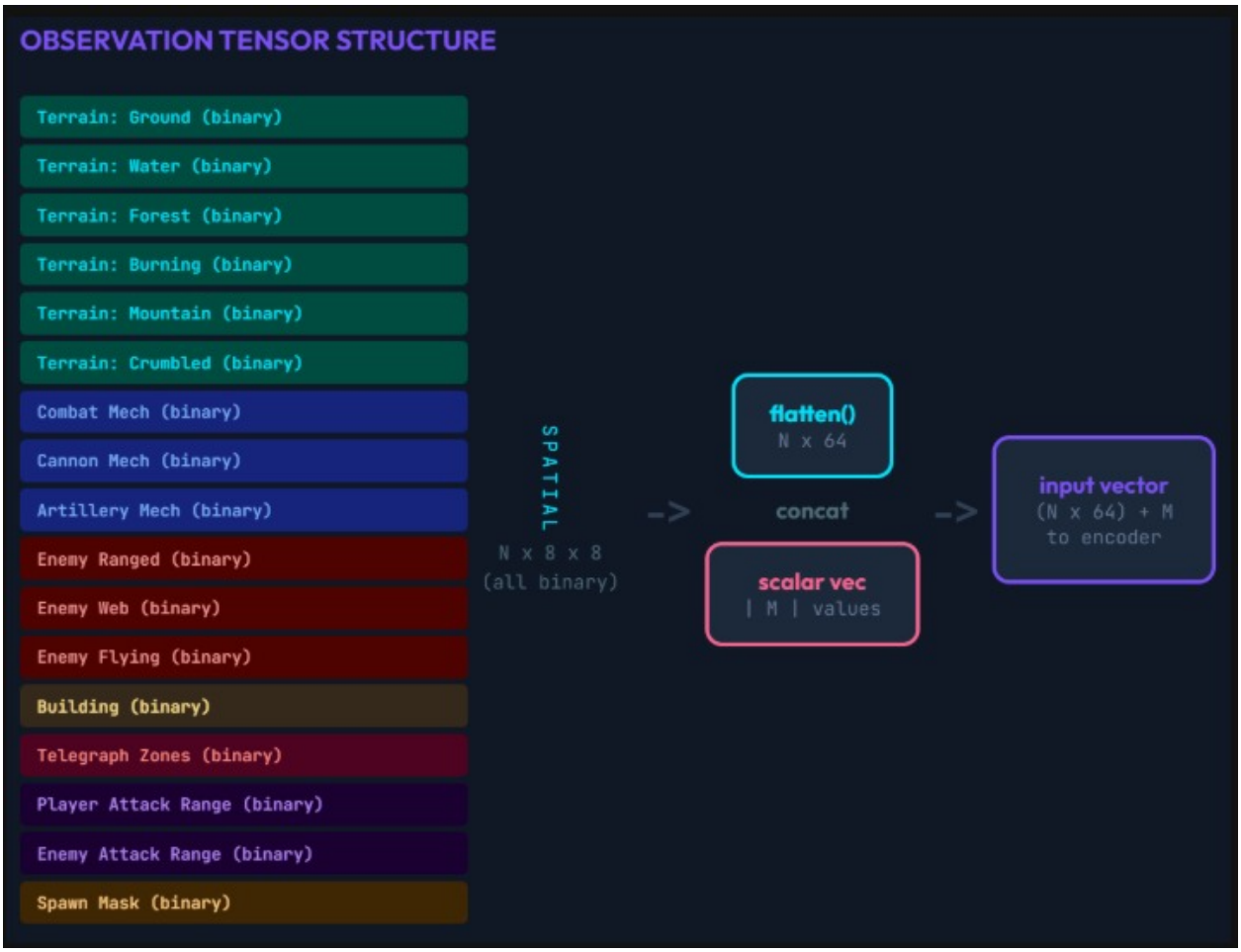


Figure 4.3: Input Tensors Overview

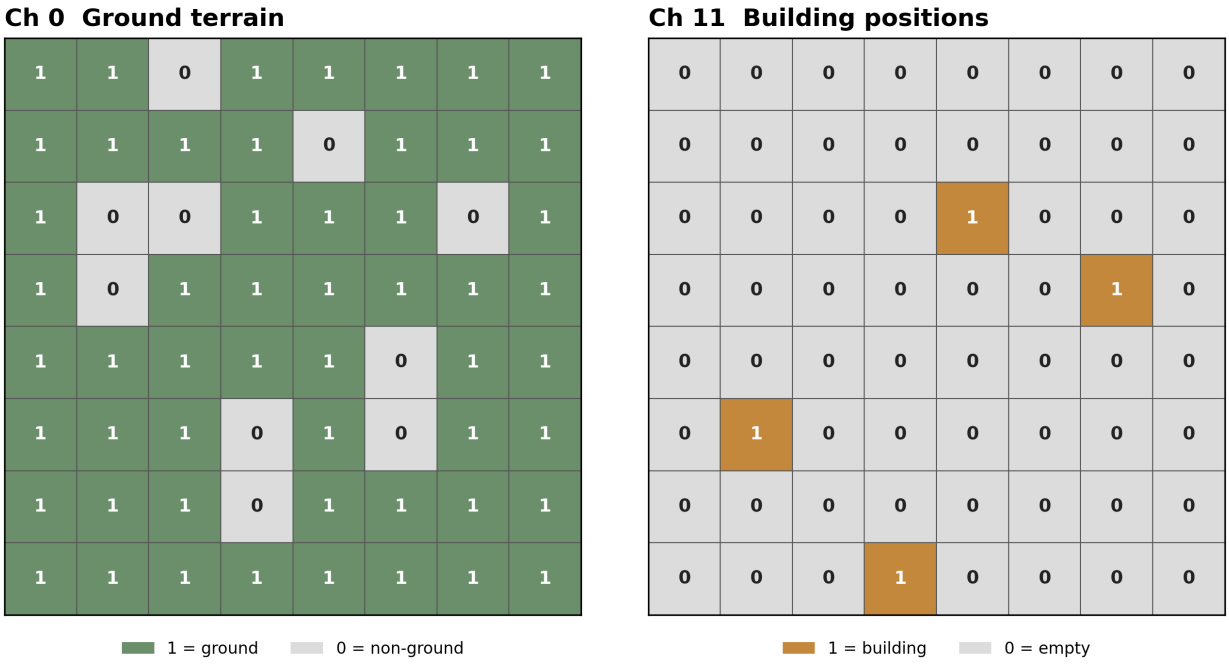


Figure 4.4: Example grid channels

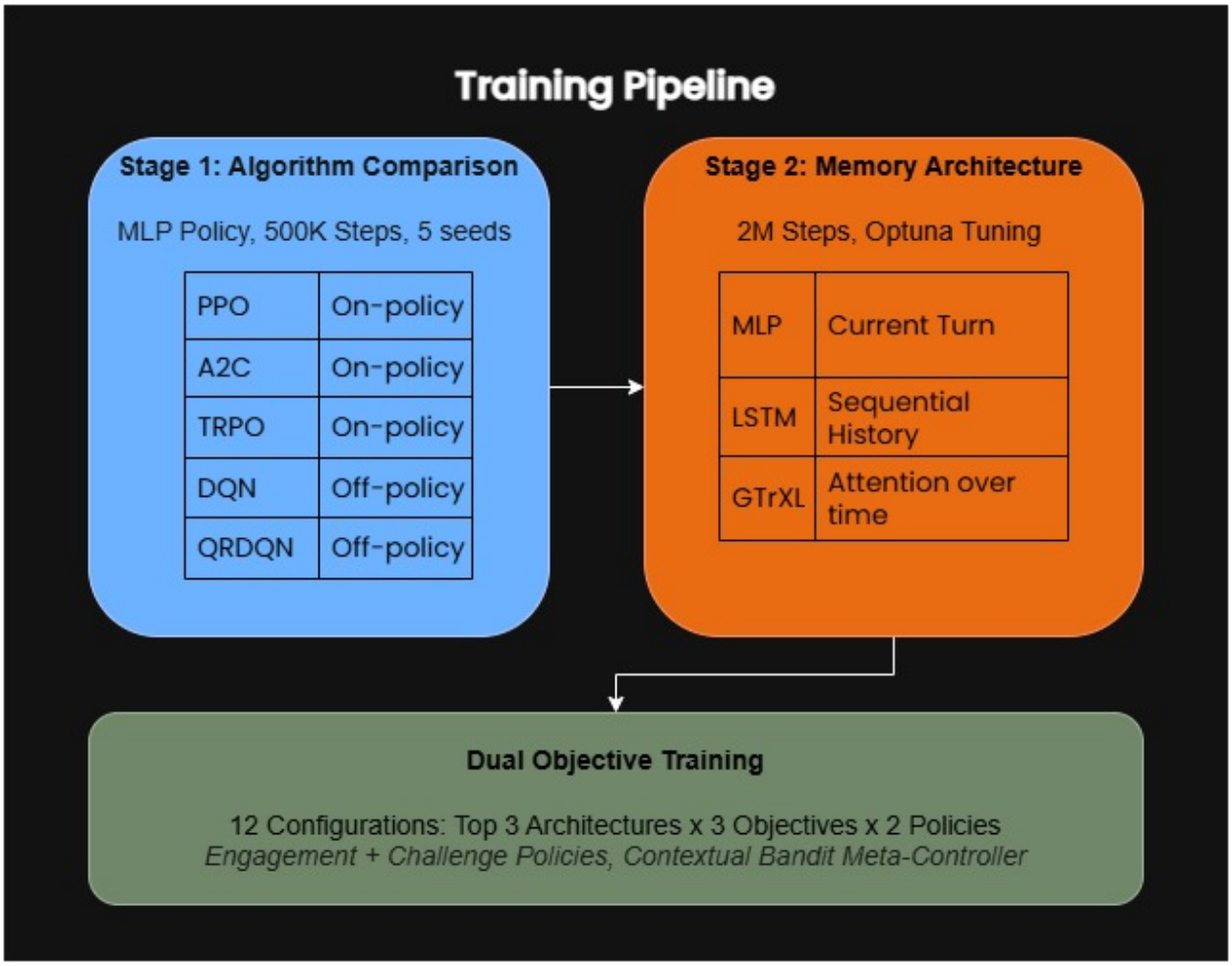


Figure 4.5: Training Pipeline Overview

CHAPTER 5. RESULTS AND DISCUSSION

This section presents experimental results across the twelve training configurations and validates the engagement reward formulas through mathematical analysis. All MLP and LSTM results reflect aggregation over 222,000 or more episodes per configuration. GTrXL evaluation is limited to cumulative reward due to framework incompatibilities between RLlib and Stable-Baselines3 callbacks. The results are organized to address the three research questions posed in Chapter 1: architecture selection (Section 5.1), the value of separating challenge from engagement (Section 5.2), and engagement quantification without player surveys (Sections 5.4 and 5.5).

5.1 Architecture Comparison

Table 5.1 presents the complete results across all twelve configurations. The most significant finding is the clear superiority of the feedforward MLP architecture across all measured dimensions.

Table 5.2 aggregates these results by architecture, averaging across objectives and policy types.

The MLP achieves an average reward of 5.24, outperforming LSTM (4.51) by 16.2% and GTrXL (4.58) by 14.4%. On the engagement metric, the MLP averages 0.389 compared to LSTM's 0.378, a 2.9% improvement. Training time further favors MLP at 1.6 hours per configuration, compared to 3.2 hours for LSTM and 10.3 hours for GTrXL.

This result can be attributed to the current game implementation's lack of meta-progression. The simulation does not include upgrades, ability unlocks, or evolving unit capabilities across turns or episodes. Each turn's decision space is fully determined by the current board state, meaning there is no persistent strategic context that would benefit from sequential memory. The 22 scalar features in the observation space already summarize the relevant cumulative information

Table 5.1: Full comparison of twelve configurations (3 architectures × 3 objectives × 2 policies). GTrXL metrics marked empty due to RLlib callback incompatibility and transformer attention differences

Configuration	Engage	Flow	Win %	Grit	Reward	Time
MLP Dual / Engagement	0.414	0.373	98.4	0.293	5.45	1.6h
MLP Dual / Challenge	0.408	0.358	98.8	0.288	5.19	1.5h
MLP Single / Engagement	0.392	0.381	97.7	0.291	5.39	1.5h
MLP Single / Challenge	0.342	0.296	97.8	0.295	4.94	1.6h
LSTM Dual / Engagement	0.379	0.295	97.6	0.264	4.53	3.2h
LSTM Dual / Challenge	0.372	0.278	98.8	0.268	4.36	3.2h
LSTM Single / Engagement	0.379	0.295	98.4	0.273	4.55	3.2h
LSTM Single / Challenge	0.383	0.306	97.9	0.271	4.60	3.2h
GTrXL Dual / Engagement	-	-	-	-	4.52	10.3h
GTrXL Dual / Challenge	-	-	-	-	4.99	10.2h
GTrXL Single / Engagement	-	-	-	-	4.63	10.3h
GTrXL Single / Challenge	-	-	-	-	4.17	10.3h

Table 5.2: Architecture comparison aggregated across objectives

Architecture	Avg Reward	Best Reward	Avg Time	Relative Speed
MLP	5.24	5.45	1.6h	1× (baseline)
LSTM	4.51	4.60	3.2h	2× slower
GTrXL	4.58	4.99	10.3h	6.4× slower

(total damage, turn number, health ratios, recent events), leaving no residual signal for recurrent or attention-based architectures to extract from raw observation history. In a game implementation with meta-progression, where the player's available actions change over time and past choices constrain future options, memory architectures would likely show a more meaningful advantage. This represents a promising direction for future evaluation.

5.2 Objective Comparison

Table 5.3 compares the three training objectives using MLP architecture results, since MLP performed best and provides full metric reporting.

Table 5.3 compares the three training objectives using MLP architecture results. The dual-objective configuration achieves the highest engagement score (0.414) and cumulative reward

Table 5.3: Objective comparison using MLP architecture

Objective	Engagement	Flow Zone	Win Rate	Grit	Reward
Dual	**0.414**	0.373	**98.4%**	0.293	**5.45**
Engagement	0.392	**0.381**	97.7%	0.291	5.39
Challenge	0.342	0.296	97.8%	**0.295**	4.94

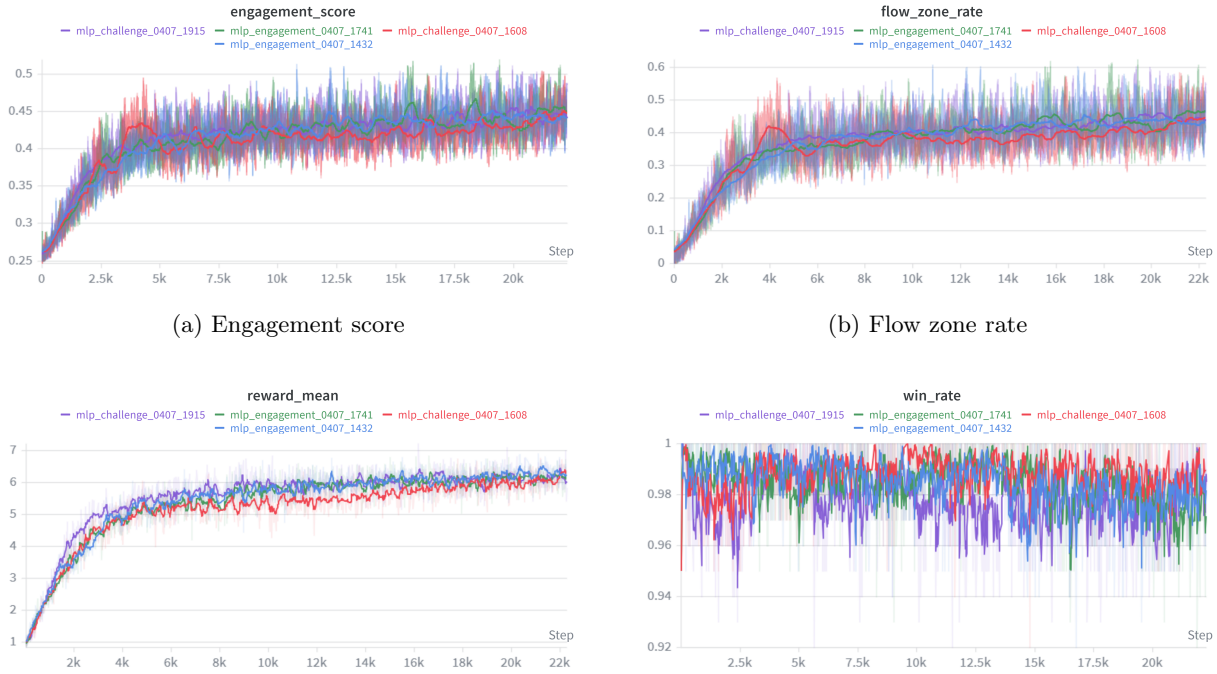
(5.45), outperforming the single-engagement objective by 5.9% on engagement and the challenge-only objective by 21.1%. Challenge-only training produces the lowest engagement (0.342) and flow zone rate (0.296), confirming that optimizing for raw difficulty does not produce engaging experiences. The single-engagement objective achieves the highest flow zone rate (0.381) but lower composite engagement than the dual configuration, suggesting that flow zone maintenance alone is insufficient without the counterbalancing pressure from the challenge policy.

This result validates the dual-objective architecture. The tension between engagement and challenge policies produces a Director that neither overwhelms the player nor allows them to coast unchallenged. The bandit's ability to blend these competing pressures based on player context yields better composite engagement than either objective can achieve in isolation.

5.3 Training Dynamics

Analysis of training progression across two million timesteps reveals that engagement metrics were still improving at the end of training. All MLP configurations started at an engagement score of approximately 0.25 and rose to between 0.44 and 0.49 by the final checkpoint, with a 7 to 10 percent improvement from the midpoint to the endpoint. This trajectory suggests that additional training beyond two million steps would yield further performance gains.

The episode-level distribution over the final 2,000 episodes of the MLP Dual Engagement configuration shows a mode in the 0.4 to 0.5 range (42.7% of episodes), with a mean of 0.439 and standard deviation of 0.108. Approximately 1.6% of episodes achieved the maximum engagement score of 0.7, while 28.1% fell in the 0.3 to 0.4 range and 20.2% in the 0.5 to 0.7 range.



(a) Engagement score

(b) Flow zone rate

(c) Mean reward

(d) Win rate

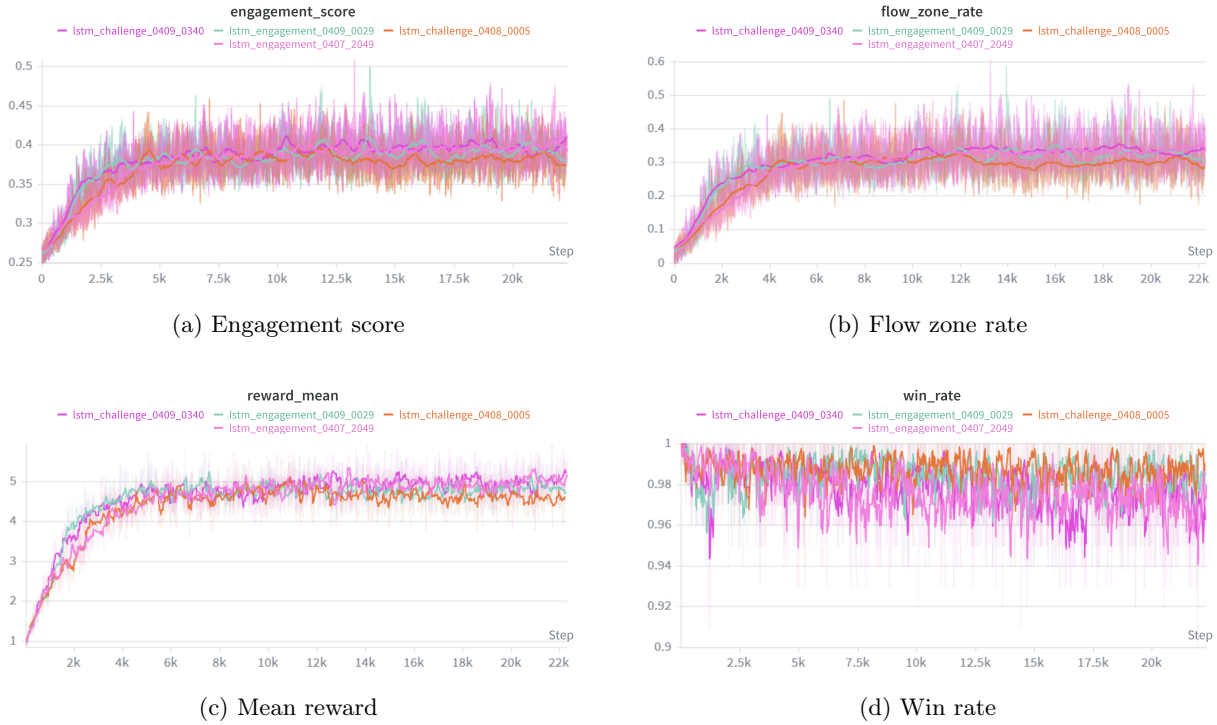
Figure 5.1: MLP training dynamics across 2M timesteps. Engagement and flow zone rate show continued upward trends at training end. Reward converges to 5–6 range. Win rate remains above 96% across all configurations

5.4 Reward Decomposition

To understand which components of the engagement reward drive learned behavior, a decomposition analysis was performed across 1,190 simulated decision points.

Mutual combat is the dominant component, contributing 70.3% of total reward and triggering on every decision point. This is expected as in any non-trivial game state with enemies present, both sides are typically dealing damage. Flow zone contributes positively in 19.2% of steps and negatively in 80.8%, reflecting the difficulty of maintaining health within the 30 to 70 percent band when the proxy player maintains high health most of the time. Adaptive pressure contributes 18.8%, activating when the player is coasting and the Director spawns enemies in response.

The close call component is nearly absent, triggering once in 1,190 steps (0.1%). This is a direct consequence of the proxy player's effectiveness, which with a 98% win rate and consistently



(a) Engagement score

(b) Flow zone rate

(c) Mean reward

(d) Win rate

Figure 5.2: LSTM training dynamics across 2M timesteps. Engagement plateaus lower (∼0.38) than MLP (∼0.44). Flow zone rate saturates around 0.30 compared to MLP's 0.40. Reward converges to 4.5–5.0 range, approximately 19% lower than MLP

high health, the player rarely drops below the 25% threshold. This component carries 30% weight in the engagement score formula but is effectively inert during training. A less capable proxy player or a more constrained Director spawning budget would allow this component to activate.

5.5 Metric Validation

The engagement reward formulas are validated through three complementary analyses.

5.5.1 Saturation Curve Analysis

The grit score uses a Michaelis-Menten saturation curve to map cumulative damage to a bounded score:

Table 5.4: Reward component decomposition across 1,190 decision points

Component	Mean	% of Total	Triggered	Notes
mutual_combat	0.300	70.3%	1190/1190	Always active
flow_zone	0.192	45.1%	229/1190	HP in 30–70%
flow_zone_penalty	−0.136	−32.0%	961/1190	Outside flow zone
adaptive_pressure	0.080	18.8%	477/1190	Spawn when coasting
adaptive_passive	−0.010	−2.4%	60/1190	Idle when coasting
close_call	0.0004	0.1%	1/1190	Near-death survival

$$\text{grit}_{\text{damage}} = \frac{d}{d+K} \tag{5.1}$$

where d is cumulative damage to the player and K is the half-saturation constant. This functional form has three useful properties, it equals zero when no damage is dealt, it approaches 1.0 asymptotically as damage increases, and it exhibits diminishing returns so that the difference between 5 and 10 damage matters more than the difference between 25 and 30. The half-saturation point occurs at $d = K$, where the output is exactly 0.5. To validate $K = 10$, we solve for the K that maps the typical damage range onto the desired output range. Given typical episode damage between 5 and 15 and a desired output range of 0.3 to 0.6:

$$K = \frac{d \cdot (1 - o)}{o} \tag{5.2}$$

where o is the desired output at damage level d. At the lower bound ($d = 5$, $o = 0.3$), this yields $K = 11.67$. At the upper bound ($d = 15$, $o = 0.6$), this yields $K = 10.0$. Averaging gives $K_{\text{recommended}} = 10.83$, confirming that the current $K = 10$ is appropriate. Table 5.5 verifies the output at key damage values.

5.5.2 Sensitivity Analysis

Sensitivity analysis was performed on three key parameters, `flow_zone_reward`, `close_call_reward`, and `ease_off_reward`. For each parameter, its value was swept across a range while all others were held constant, and the total reward was computed across

Table 5.5: Saturation curve outputs at $K = 10$

Damage (d)	**Output $\frac{d}{d+10}$**	**Interpretation**
0	0.000	No challenge
5	0.333	Low challenge
10	0.500	Half-saturation point
15	0.600	High challenge
25	0.714	Diminishing returns
50	0.833	Approaching ceiling

representative game state scenarios. Results confirm expected directional effects, increasing `flow_zone_reward` widens the reward gap between in-zone and out-of-zone states, increasing `close_call_reward` strengthens the incentive for near-death states, and increasing `ease_off_reward` strengthens the incentive to stop spawning when the player is struggling. No parameter exhibits discontinuities, sign reversals, or extreme sensitivity within reasonable ranges.

5.5.3 Correlation Analysis

Pairwise Pearson correlations between episode-level metrics confirm that the engagement formulas capture the intended relationships. The correlation coefficient is computed as:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{5.3}$$

Table 5.6 reports the results across five metric pairs.

The most important result is the near-zero correlation between win rate and flow zone rate ($r = -0.09$). This confirms that the engagement metric is not a proxy for win rate. A player can experience high engagement regardless of whether they ultimately win or lose.

The strong negative correlation between flow zone rate and average HP ($r = -0.88$) confirms internal consistency, time in the flow zone (HP in $[0.3, 0.7]$) is mechanically opposed to high average HP. The strong positive correlation between flow zone rate and HP variance ($r = 0.86$) validates the tension hypothesis, while episodes with more time in the flow zone exhibit larger HP swings, indicating a more dynamic experience.

Table 5.6: Pairwise Pearson correlations between episode-level metrics

Metric Pair	r	**Interpretation**
flow_zone_rate vs avg_hp	-0.88	Strong negative (expected)
flow_zone_rate vs hp_variance	0.86	Strong positive (validates tension)
flow_zone_rate vs ep_length	0.44	Moderate positive
avg_hp vs hp_variance	-0.89	Strong negative
win_rate vs flow_zone_rate	-0.09	Near zero (confirms independence)

5.6 Proxy Player Impact

The proxy player's 98% win rate across all configurations represents both a validation and a limitation. As validation, it confirms that the Director does not learn to trivially defeat the player, which would produce high challenge scores but low engagement. As a limitation, it means the close call component (30% weight in engagement score) almost never activates, and the theoretical maximum engagement score is approximately 0.47 rather than the formula's nominal maximum of 1.0.

With real human players who exhibit more variable decision-making, the engagement landscape would be richer. The Director would have opportunities to learn from the close call signal, potentially producing qualitatively different spawning patterns. The dual-objective architecture would likely show more differentiation from the single-objective baseline when trained against a more varied player population, as the bandit would encounter a wider range of player contexts.

CHAPTER 6. CONCLUSION AND FUTURE WORK

6.1 Summary of Contributions

This work makes four primary contributions. First, we design and implement a dual-objective reinforcement learning architecture that explicitly separates challenge from engagement, training two independent PPO policies with distinct reward functions and combining them through a contextual bandit meta-controller. Second, we develop a multi-component engagement reward function grounded in flow theory, incorporating flow zone maintenance, close call detection, adaptive spawning pressure, recovery arc recognition, mutual combat, and graduated outcome scoring, with mathematical validation through saturation curve analysis, sensitivity testing, and correlation analysis. Third, we conduct a systematic comparison of three memory architectures (MLP, LSTM, GTrXL) across three training objectives, producing the finding that feedforward networks outperform recurrent and attention-based alternatives in this domain due to the absence of meta-progression mechanics. Fourth, we deliver a complete pipeline from HPC-based training with hyperparameter optimization to a FastAPI inference server integrated with a live game engine through a JSON API contract.

6.2 Key Findings

RQ1: Which RL architecture works best for this task? The feedforward MLP outperforms both LSTM (+2.9% engagement, +16.2% reward) and GTrXL (+14.4% reward) while training 2 to 6.4 times faster. This advantage stems from the current game implementation's lack of meta-progression: without upgrades, ability unlocks, or evolving unit capabilities across turns, each turn's decision space is fully determined by the current board state. The 22 scalar features in the observation space already summarize the relevant cumulative information, leaving no residual signal for recurrent or attention-based architectures to extract.

RQ2: Does separating challenge from engagement help? The dual-objective configuration outperforms both single-objective alternatives, achieving 5.6% higher engagement than engagement-only training and 21.1% higher than challenge-only. This validates the core architectural hypothesis: separating challenge from engagement into distinct policies and blending them through a contextual bandit produces better engagement outcomes than optimizing for either objective alone. Challenge-only training confirms that raw difficulty optimization actively harms engagement, producing the lowest flow zone rate (0.296) of any configuration.

RQ3: How can player engagement be quantified without player surveys? The correlation analysis confirms the foundational claim of this work: engagement is independent of win rate ($r = -0.09$). Flow zone rate correlates strongly with HP variance ($r = 0.86$), validating the tension hypothesis that time in the moderate-challenge band produces more dynamic experiences. The grit score's Michaelis-Menten curve is mathematically validated with $K = 10$ mapping the typical damage range onto the desired output range. These results demonstrate that engagement can be operationalized through observable game-state telemetry without requiring subjective player reports, though human studies remain necessary to confirm that these proxies correlate with actual player experience.

6.3 Limitations

Several limitations constrain the interpretation of these results. The proxy player achieves a 98% win rate, suppressing the close call component (30% weight in engagement score) and capping achievable engagement at approximately 0.47. No real player data was collected; engagement is validated mathematically but not against subjective experience. Training was conducted for two million steps, but metrics were still improving, suggesting longer training would yield higher scores.

6.4 Future Work

Several directions emerge from these findings, spanning player modeling, reward design, and architectural extensions.

Player Modeling. The most impactful extension would replace the fixed proxy player with a learned player model. Inverse Q-learning could infer player reward functions from observed behavior, revealing what individual players actually optimize for rather than assuming a uniform skill parameter. Clustering over gameplay telemetry (action sequences, risk tolerance, objective prioritization) could identify player archetypes such as aggressive, cautious, or objective-focused playstyles. These richer player representations could then condition the bandit's context selection on features beyond health ratio alone, enabling the system to adapt not just to how well a player is doing but to how they prefer to play.

Synergy Quantification. Tactical roguelikes exhibit nonlinear payoffs where certain action combinations are worth more than the sum of their parts. Pushing an enemy into water destroys it outright, a result far more valuable than the individual effects of a push or a water tile. A formal framework for quantifying these synergies would allow the Director to reason about the emergent tactical opportunities it creates through spawning decisions, either setting up synergy opportunities for the player (to maintain engagement) or denying them (to increase challenge).

Difficulty as Execution Tolerance. An alternative framing of difficulty focuses not on enemy strength but on the margin of error available to the player. A board state where only one sequence of moves protects all buildings is harder than one where three different approaches work, even if the enemies are identical in both cases. Formalizing this notion of execution tolerance, perhaps through the branching factor of viable action sequences at each decision point, would provide a complementary difficulty metric that captures the tactical tightness of a situation rather than its raw threat level.

Meta-Progression Mechanics. Introducing upgrades, ability unlocks, and evolving mech loadouts into the simulation would create genuine long-range dependencies that memory architectures could exploit, potentially reversing the MLP advantage observed in the current experiments. This would also make the game more representative of commercial roguelikes where between-run progression is a core engagement driver.

Extended Training and Reward Redesign. Training beyond two million steps would allow convergence and likely improve engagement scores. Redesigning the engagement and challenge rewards to produce more behaviorally distinct policies would give the bandit greater leverage, potentially demonstrating the dual-objective architecture's theoretical advantage.

Architectural Extensions. Monte Carlo Tree Search integration for the tactical layer would add planning-based lookahead to complement the RL-based strategic layer. A Graph Neural Network encoder for the spatial observation could better exploit the grid structure than the current flattened representation, particularly for reasoning about spatial relationships between units, buildings, and terrain hazards.

Human Player Study. A player study following the methodology used to evaluate Left 4 Dead's Director [Booth (2009)] would validate whether the engagement metrics correlate with subjective player experience. Participants would play sessions under different Director configurations (engagement-focused, challenge-focused, static baseline) and provide ratings on perceived difficulty, enjoyment, tension, and desire to continue playing. This would close the gap between mathematical validation and experiential validation, confirming or refuting the assumption that flow zone rate, close calls, and grit score are meaningful proxies for what players actually find engaging.

6.5 References

Andrade, G., Ramalho, G., Gomes, A. S., and Corruble, V. (2006). Dynamic game balancing: An evaluation of user satisfaction. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, pages 3–8.

Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs.

Badia, A. P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, D., and Blundell, C. (2020). Agent57: Outperforming the atari human benchmark.

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. (2019). Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Bontchev, B. (2016). Adaptation in affective video games: A literature review. *Cybernetics and Information Technologies*, 16(3):3–34.

Booth, M. (2009). The AI systems of Left 4 Dead. In *Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*.

Cowley, B., Charles, D., Black, M., and Hickey, R. (2008). Toward an understanding of flow in video games. *Computers in Entertainment*, 6(2):1–27.

Csikszentmihalyi, M. (1991). Flow: The psychology of optimal experience. *Design Issues*, 8(1):80.

Hunicke, R. and Chapman, V. (2004). AI for dynamic difficulty adjustment in games. In *Challenges in Game AI Workshop, Nineteenth National Conference on Artificial Intelligence*, pages 91–96.

Ishii, A., Ito, S., and Ishimura, N. (2018). Dynamic difficulty adjustment with reinforcement learning in fighting games. In *Proceedings of the IEEE Conference on Games*.

Lopes, P. and Yannakakis, G. N. (2019). Deep reinforcement learning for procedural content generation. In *Proceedings of the IEEE Conference on Games*.

Parisotto, E., Song, H. F., Rae, J. W., Pascanu, R., Gulcehre, C., Jayakumar, S. M., Jaderberg, M., Kaufman, R. L., Clark, A., Noury, S., and et al. (2019). Stabilizing transformers for reinforcement learning.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms.

Thompson, T. (2015). The perfect organism: The AI of Alien: Isolation. Gamasutra/AI and Games.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., et al. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.

Yannakakis, G. N., Spronck, P., Loiacono, D., Andre, E., Yannakakis, G. N., Spronck, P., Loiacono, D., and Andre, E. (2013). *Player modeling*. Schloss Dagstuhl - Leibniz-Zentrum fur Informatik.

Yannakakis, G. N. and Togelius, J. (2011). Experience-driven procedural content generation. *IEEE Transactions on Affective Computing*, 2(3):147–161.

Zohaib, M. (2018). Dynamic difficulty adjustment (dda) in computer games: A review. *Advances in Human-Computer Interaction*, 2018:1–12.