Reinforcement Learning-Based Strategic Level Generation for Turn-Based Grid Games

Submitted By

Md Rakib Hossain

ID: 211-35-705

Department of Software Engineering

Daffodil International University

Supervised By

Dr. Shabnom Mustary

Assistant Professor

Department of Software Engineering

Daffodil International University

A thesis submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering

Fall 2025

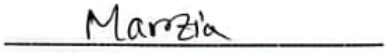
APPROVAL

This thesis titled on **"Reinforcement Learning-Based Strategic Level Generation for Turn-Based Grid Games"**, submitted by **Md Rakib Hossain (ID: 211-35-705)** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

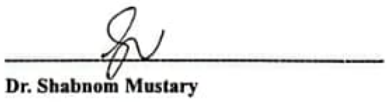
Chairman

Dr. Fazla Ealhe
Assistant Professor & Associate Head
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



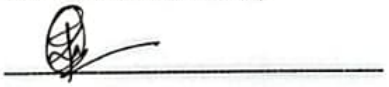
Internal Examiner 1

Dr. Marzia Ahmed
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



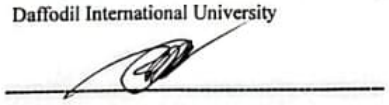
Internal Examiner 2

Dr. Shabnom Mustary
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



Internal Examiner 3

Md. Rajib Mia
Lecturer (Senior Scale)
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



External Examiner

Mohammod Abul Kashem, PhD
Professor
Department of Computer Science and Engineering
DUET, Bangladesh

Department of Software Engineering

Faculty of Science and Information Technology

Supervisor Approval Form

Fall 2025	B.Sc. In SWE	Campus: DSC

Student Name	Student ID
Md Rakib Hossain	211-35-705

Project/Thesis Information	
Project/Thesis Title	Reinforcement Learning-Based Strategic Level Generation for Turn-Based Grid Games
Type of work	Thesis

Supervisor information	
Supervisor Name	Dr Shabnom Mustary
Supervisor Initial	SM
Completed Credit till now	
How many credits in this semester	
Amount (Due)	
Supervisor Consent	☑ Yes ☐ No

Supervisor Signature

DAFFODIL INTERNATIONAL UNIVERSITY

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name	:	MD Rakib Hossain
Date of Birth	:	29 March, 2001
Title	:	Reinforcement Learning-Based Strategic Level Generation for Turn-Based Grid Games
Academic Session	:	Fall 2025

I declare that this thesis is classified as:

☐	CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐	RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☐	OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Daffodil International University reserves the following rights:

1. The Thesis is the Property of Daffodil International University.
2. The Library of Daffodil International University has the right to make copies of the thesis for the purpose of research only.
3. The Library of Daffodil International University has the right to make copies of the thesis for academic exchange.

Certified by:

_____
(Student's Signature)

221-35-938
Student ID
Date:

_____
(Supervisor's Signature)

Mr. Md. Shohel Arman
Name of Supervisor
Date:

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Bachelor of Science.

(Supervisor's Signature)

Full Name : Dr. Shabnom Mustary

Position : Assistant Professor

Date : 28 December, 2025

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Daffodil International University or any other institution.

(Student's Signature)

Full name : Md Rakib Hossain

ID Number : 211-35-705

Date : 28 December, 2025

Reinforcement Learning-Based Strategic Level Generation for Turn-Based Grid Games

Md. Rakib Hossain

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Science

Department of Software Engineering (Major in Software Engineering)

DAFFODIL INTERNATIONAL UNIVERSITY

November 2025

ACKNOWLEDGEMENTS

All praise and gratitude are due to the Almighty for providing the strength and guidance to complete this research.

I express my deepest gratitude to my esteemed supervisor, Assistant Professor Dr. Shabnom Mustary, whose expert guidance, insightful feedback, and continuous support have been instrumental in shaping this thesis. Your knowledge and encouragement have not only guided this research but have also been a source of inspiration throughout my academic journey.

I am also grateful to my family for their unwavering support, patience, and encouragement. I thank my colleagues and friends for their cooperation and feedback that helped advance this work.

DEDICATION

To all those who believe that artificial intelligence can solve complex optimization problems and create more intelligent solutions for real-world applications.

ABSTRACT

Procedural content generation for turn based strategic games is challenging due to the need to satisfy multiple competing design objectives such as strategic depth balance and playability. Traditional manual and search based generation methods are time consuming and often fail to scale effectively under complex design constraints. This thesis presents a reinforcement learning based procedural content generation framework for automated level generation in turn based strategic games. In this approach, Proximal Policy Optimization is used to develop agents within a Gymnasium-compatible grid environment. There are a combination of strategies and measures employed in the objective for this approach to aid in developing agents. These include developing agents with diverse paths and complex decisionmaking points while also taking into account the layout of the main elements of a game (e.g., start point, end point, obstacles, objectives). The model is trained for five hundred thousand timesteps and evaluated on three hundred generated levels using three different random seeds. Experimental results show that the proposed reinforcement learning based generator consistently outperforms random baseline generation achieving substantially higher average quality scores with significantly lower variance. Statistical analysis confirms that the observed improvements are robust across all evaluation settings. Through additional evaluative efforts, qualitative evaluation of the methodology shows that the trained agent learns to exhibit meaningful behaviors related to design, including creating effective prioritized paths, building strategic decision points, and designing balanced rulebased gameplay through the use of re-enforcement learning methodology without the need for explicit constraint-based methods. The proposed framework expands the procedural content generation by means of reinforcement learning methodology to include turn-based strategy games and illustrates how practical applications can be used in the development of future games.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	Definition
R	Reward
π	Policy
V	Value Function
λ	GAE Lambda Parameter
γ	Discount Factor
ε	Clip Range
σ	Standard Deviation
μ	Mean
d	Cohen's Effect Size

LIST OF ABBREVIATIONS

Abbreviation	Full Form
RL	Reinforcement Learning

PCG	Procedural Content Generation
PCGRL	Procedural Content Generation via Reinforcement Learning
PPO	Proximal Policy Optimization
MDP	Markov Decision Process
GPU	Graphics Processing Unit
BFS	Breadth-First Search
CSV	Comma-Separated Values
AI	Artificial Intelligence
ML	Machine Learning
PPO	Proximal Policy Optimization

CHAPTER 1

INTRODUCTION

1.1 Background

As the need for the varied, captivating, and replayable gaming experiences increases, procedural content generation has grown in significance in the game development [1]. For the creation of many game content, such as levels, maps, and assets, conventional PCG methods like the rule-based systems, evolutionary algorithms, and grammar-based generators work well. The issue is that these kind of methods necessitate a great deal of manual adjustment and domain expertise. As a result, they are less kind of flexible and more challenging to use in various game genres. [2]. These days, the field is more defined by three key ideas. Evolutionary algorithms and optimization are introduced by search-based PCG, which uses fitness functions to find interesting content. In order to ensure that everything complies with rules, solvable-based PCG takes a more strict approach, relying on constraint satisfaction and answer set programming. Formal grammars are essential to grammar-based PCG because they offer the foundation for legitimate structures while maintaining sufficient flexibility for inventive twists and surprises. [2].

A particularly interesting domain for procedural level generation is turn-based grid-based games. Levels that provide significant strategic options while preserving fair and balanced gameplay are necessary for these games, which include puzzle-based strategy games, tactical role-playing games, and chess puzzles [3]. Creating content that is neither incredibly simple nor extremely complex while still giving players access to a variety of workable strategies and interesting decision-making experiences is the main challenge.

Procedural content generation now has more options thanks to recent developments in machine learning, especially reinforcement learning (RL). Khalifa et al. [4] introduced the Procedural Content Generation via Reinforcement Learning (PCGRL) paradigm. This approach approaches level design as a methodical decision-making process. Piece by piece, an RL agent creates game content with the goal of meeting user-specified quality standards. Large training datasets are not

required, and hand-tuning or manually creating numerous rules is not as problematic. As a result, RL-based PCG techniques are ultimately more adaptable and flexible than traditional procedural generation.

1.2 Problem Statement

PCGRL is quite promising however the real logistics when you want to deploy reinforcement learning for strategic level generation in turn-based grid games are a mess. You can't just hit one goal levels must have both actual strategic depth and a good balance of elements, and these goals often tug against the other. The majority of what we have at the present state focuses on one singular goal, but it's not nearly enough. Then there's playability. It's not optional. Levels are a requirement that needs to work, and has to be at a certain quality bar. But the issue remains that too rarely do such randomly generated or naive-derived stages result in anything but garbage, unplayable content. Consistency is another sticking point.

A generator cannot be expecting it to succeed by chance now and then. A generator has to be able to deliver satisfying outcomes regardless of the initial conditions you use to start it. A practical tool is what developers are also looking for. The development and execution of such models should be done on normal hardware and should not require fancy equipment that no studio is ready to afford. There is also an Underlying issue in the research community. Statistical validation, such as effect size analysis and using different seeds, is hardly taken into consideration in research in PCGRL. Conventional procedural content generation community is also no better.

The fact there is that there has been no proper thing for multi-seed testing and zero reporting of effect sizes in the literature. Which makes it difficult to conclude whether this method in PCGRL are actually reliable and viable. And also to move ahead in this field and to make content generation using RL is more viable in game development. All of these challenges should be taken on the right upfront.

1.3 Motivation

There are three things that motivated this research. To begin, mostly the current approaches to procedural content generation PCG aren't really living up to the challenge they require a lot of design effort and lots of computational horsepower, also knowledge of the domain. None of them deal simultaneously with many several design objectives. Two, the area of reinforcement learning had several successes in hard optimization problems, like playing Go and robotics, but its capabilities for the strategic game design, and specifically for designing several objectives simultaneously, remained largely unexploited.

However, Proximal Policy Optimization PPO is some kind of exception in this regard. PPO is a relative newcomer to this policy gradient method family, known as one of the stable training and sample-efficient algorithms in this category, making it very applicable to the task of content generation.

we can define these reward functions to consider a number of design criteria, so the PPO agents are able to learn complex strategy choices.without having to program them by hand.Through this project, it is also planned to develop and demonstrate a level generator using PPO, designed to achieve strategic depth as well as balancing within turn-based grid games, concurrently as a whole. The overall objective is to move the state-of-the-art within PCGRL research by showing what can be accomplished by real multi-objective optimization and also setting further benchmark goals, providing game programmers with useful tools to implement PCGRLs.

1.4 Significance of the Study

This research fills an important gap in the ongoing literature and practice to apply RL to multi-object level generation when it comes to turn-based strategy games. The research was conducted by performing experiments on long training scenarios 500k timesteps and appropriate multi-seed evaluation to give an initial study of the results of PCGRL on strategic games.

These results have many implications:

- **For Procedural Content Generation:** Extends PCGRL to turn-based strategic games and demonstrates effective multi-objective optimization without manual trade-off tuning.

- **For Game Development:** Provides a practical tool that could be integrated into development pipelines, reducing content creation time and enabling new gameplay experiences.

- **For Reinforcement Learning:** Demonstrates the applicability of RL to discrete spatial optimization problems with multiple competing objectives.

- **For AI Research:** Sets a standard for rigorous empirical evaluation in PCGRL, including multi-seed robustness testing and effect size reporting.

Ultimately, this work could pave the way for broader adoption of RL-based content generation tools, reducing development time and enabling dynamic, personalized gaming experiences.

1.5 Research Questions

These research try to find answer to the questions:

1. Is it possible to produce high quality levels by using reinforcement learning which maximizes strategic complexity and balance?

2. Can a dual-objective reward function help RL agent reach a balance between competitive design objectives without resorting to explicit constraint programming?

3. Is high quality guaranteed in level generated by the RL-based generator regardless of the random initialization?

4. To what extent is the computational efficiency of the RL method in view of requirements for the game development sector?

5. Which learned behaviors are shown by RL agent in level design principles?

6. In what way does the RL-based generator compares quantitatively to random baseline and other procedural generation approaches?

1.6 Research Objective

These study was conducted with the objectives:

- To design and implement a Gymnasium-compatible environment for grid-based level generation with flexible reward mechanisms.

- To develop a dual-objective reward function that captures both strategic depth (path availability, decision complexity) and balance (element distribution).

- To train a PPO agent with extended training (500K timesteps) to ensure convergence to optimal policy.

- To conduct comprehensive evaluation comparing RL-generated levels against random baseline across multiple dimensions.

- To validate results through rigorous statistical analysis including effect sizes, multi-seed robustness testing, and distribution analysis.

- To analyze learned behaviors and provide insights into what strategies the RL agent discovers for effective level design.

1.7 Research Scope and Limitations

This highlights the scope of these study and the limitations which are arising on the environment design, chosen methods, and conducting process.

1.7.1 Scope

- These research mainly focuses on turn-based grid games with 10×10 level representation.
- The PPO algorithm is mainly analyzed with random baseline as for comparison.
- Both types of metrics are considered: quality-based metrics (strategic depth, balance, total score) and computational performance metrics (training time, inference speed).
- The evaluation was conducting using three random seeds (42, 123, 456) with 50 levels per seed per method, totaling 300 levels to make sure reproducibility and robustness.

1.7.2 Limitations

- **Grid Size:** The 10×10 grid, which is practical, may wont scale without architectural modifications to a larger environments.
- **Single Game Type:** Results are specific to the designed turn-based grid game; generalization to other game types requires validation.
- **Metrics as Proxies:** Strategic depth and balance metrics are proxies for true player satisfaction; the study lacks human evaluation of generated level quality.
- **Hardware Dependence:** Computational metrics (60 minutes training on Google Colab T4 GPU) may vary across different hardware configurations.
- **Temporal Scope:** Represents a snapshot of RL techniques; newer methods may emerge that outperform the current approach.

1.8 Thesis Organization

This thesis is divided into six major chapters. Chapter 1 which gives a background study of the research work, along with the statement of problem, objectives, significance, scope, and limitations of the work. Chapter 2 is which gives insights into the complete overview of related literature, explaining the foundations of procedural content generation, different techniques of PCGML, the

paradigm of procedural content generation in reinforcement learning, and its variants. Chapter 3 then describes the methodological approach of research work, explaining the design of the environment, definition of reward function, settings of training, as well as evaluation. Chapter 4 presents the results obtained from the experimental work, including training convergence, comparative study, statistical validation, analysis of distributions, as well as robustness tests on multi-seeds. Chapter 5 draws the final conclusion of the research work by summarizing major contributions, findings, recommendations, and future work. Chapter 6 gives complete citations to all the pieces of work referred to in the document.

CHAPTER 2

LITERATURE REVIEW

2.1 Related Works

Procedural Content Generation (PCG) has emerged as the driving force of game development because of its implications on generating diverse, interesting, and replayable game experiences. The rapid growth of Machine Learning (ML) and Reinforcement Learning (RL) techniques has brought a revolution in the way content is generated, promising efficient and effective results. In this chapter, the methodologies ranging from classical Procedural Generation techniques, Machine Learning techniques, to Reinforcement Learning-inspired techniques and the advantages as well as limitations of each in game level generation are presented.

Feng et al. reported the application of RL and search algorithms for creative puzzle creation in chess puzzles where the solutions contradict intuition. The effectiveness of the puzzles was demonstrated where puzzles generated by the implementation of RL puzzle generation algorithms, Monte Carlo Tree Search, Neural Network Evaluation, and AlphaZero algorithms are more creative and difficult for experts compared to other algorithms. The domain specificity of the algorithm has costs in terms of computation and lack of generalizability for other strategy games.

Rupp et al. proposed Simulation-Driven Balancing of Competitive Game Levels with RL, balancing competitive two-player levels using simulation-based RL agents swapping level components to reduce win-rate disparity. Using PCGRL-style policy-gradient RL with swap-based level representation and simulation-based rewards, balanced levels achieved 68% with 88.9% improvement over baseline and improved fairness metrics. Limitations noted included high compute costs and dependence on simulated playtesters.

Back et al. introduced IPCGRL (Language-Instructed RL for Procedural Level Generation), which extended PCGRL to include language instructions where level descriptions are text. To generate levels guided by language instructions, language-conditioned PCGRL was used along with text embeddings for control input, and PPO-trained policies for textual description satisfaction. This research was still under consideration, and language ambiguity issues are taken into account.

Baek et al. proposed PCGRLLM as an application of the LLM in the field of PCGRL, applying the use of the LLM in the automatic design of reward functions defined in natural language. These rewards functioned well and ensured diversity in the generation of levels without requiring human optimization. The rewards produced by the LLM may have errors associated with computational cost.

Earle et al. framed Video Game Level Design as a Multi-Agent RL Problem: multiple agents have to collaboratively design different components of a level in parallel. They proposed the use of MARL along with PPO and coordination mechanisms where the agents will specialize in different level aspects, and parallel generation using the same outperformed single-agent baselines in terms of efficiency, diversity in generated level components, and scalability. However, there are coordination overheads and agent credit assignment challenges.

Rupp et al. proposed G-PCGRL to address Procedural Graph Data Generation via RL, applying PCGRL to graph data such as skill trees and economies. With graph data as adjacency matrices and RL graph editing edges/nodes, along with graph-related rewards that symbolize target graph connectivity and target graph target count, graphs were generated that satisfied the target graph properties and were more-controllable than the competing approaches. However, it performed poorly on larger graphs, as well as in semantic game play evaluation.

Earle et al. improved the scaling and controllability of PCGRL using observation windows, multi-task learning, and domain generalization in game environments. With the use of PPO with multi-task learning, curriculum learning, and observation mechanisms, the model was able to scale up to larger levels of the game while offering better domain generalization and increased training speed. Observations in the domain were still domain-dependent and lacked transferability.

Chen et al. implemented deep RL algorithm PPO on Procedural Game Level Design in 3D environments with tile representation using deep neural networks. Level generation capable of playable 3D levels with PPO performing better than DQN baseline showed competitive solvability. Solvability in the 3D domain has high computation cost with no gameplay validation as well as user studies.

Charity et al. presented Active Learning for Classifying 2D Grid-Based Level Completability, which performs active learning query strategies using classifiers with uncertainty sampling-SVM and neural networks-for level selection. The method achieved label efficiency of around 70-80% compared to passive learning, while enabling fast classification and good generalization to unseen domains. Limitations include only being applied to 2D grid levels and class imbalance issues.

Sudhakaran et al. introduced PCGPT (Procedural Content Generation using Transformers) through pretraining on level data and subsequent RL fine-tuning for reward signals. By employing autoregressive Transformers and pretraining followed by PPO fine-tuning, it was able accessment and diversity tasks, pretraining helping outperform baseline models without pretraining. High computational cost and loss of 2D locality through tokenization were mentioned drawbacks.

Jiang et al. introduced Learning Controllable 3D Level Generators for controllable environment/voxel/room generation w.r.t. tasks like density and path lengths. In the Conditional VAE (CVAE) approach for latent conditioning and support for RL refinement for satisfying task attributes, controllability achieved 85%-92% success in test tasks. The cons are related to dimensionality and interpretation of latent spaces and consideration of restricted tasks instead of games.

The applicability of RL was shown by Nam et al. in designing levels for games using RL with optimization in terms of both quality and diversity. The model was developed using PPO/TD3 and the quality diversity/archive algorithm with the help of tile representations and metrics for diversity and novelty. The model was tested in grid levels and domain-specific reward shaping functions. The applicability was limited to grid levels and the scalability for levels was not tested.

In a continuation of their work on PCGRL, Earle et al. introduced controllability by conditioning generations on goal attributes to ensure goal-aligned levels with diversity. Utilizing PCGRL incorporating goal conditioning, latent control in the space of goals, and multi-tasking on different signals to guarantee diversity, they were able to achieve goal satisfaction of 85-90%, all while ensuring diversity. This solution worked only on simple action signals.

Mixed-initiative level design was incorporated in Mixed Initiative Level Design for RL Brush by Delarosa et al. This system allows human-level designers to interact with a 'brush' while working together with the RL agent for designing suggestions based on human feedback and heuristics. The mixed-initiative system was found to be functional for designing levels successfully through DQN and was accepted well by human designers, along with the benefit of reduced designing times. The drawback was the involvement of human time and limitations of scalability of mixed-initiative systems.

Gisslén et al. introduced Adversarial Reinforcement Learning for PCG (ARLPCG), which was based on the generator-solver framework in which the generator was trained to produce levels solvable and difficult for the solver agent. Utilizing the PPO/A2C algorithm and grid representations for levels, the algorithm produced levels in the range of 78–82% solvable and more difficult than the baseline method. The algorithm indicated instability in adversarial training and lack of scalability.

Shu et al. introduced EDPCGRL through a Super Mario Bros. study that targets key player experience metrics, including difficulty and engagement. The RL agent modifies tiles with reward functions combining experience-driven metrics; testing uses simulated agents as playtesters. Results showed 85–90% playable levels under simulated agents. Simulated agents differ from human players, meaning further user validation is needed.

Fontaine and Nikolaidis proposed the use of the Differentiable Quality Diversity (dQD) algorithm, which utilizes differentiable map elites as a variant of the MAP-Elites algorithm and possesses better sample efficiency than non-gradient approaches to quality diversity. This algorithm is effective only when searching for the Pareto frontiers and also scales well when its compared to the original map elites. The drawback here lies in the needs for differentiable simulators.

Liu et al. written an excellent survey which is about deep learning techniques used for PCGs, covering GANs, VAEs, RL, and hybrids. This survey includes deep learning models like CNNs, RNNs, and Transformers, and other generative models like that. It also gives an excellent viewpoint of the area, points out the challenge and the future of the deep learning techniques. Since it is a survey paper, no new techniques have been proposed. Due to the ever-changing nature of the area, some of the techniques may be outdated.

Khalifa et al. brought up the seminal work of PCGRL, modeled as a Markov Decision Process (MDP), where RL agents progressively modified tile maps for creating playable levels. The approach used varying observation modes (narrow, turtle, wide) for selecting modifications in tile maps based on playability criteria. The PPO and DQN algorithms under the gym environment and tile graph MDPS resulted in a solvability rate of 90% for the tested domains. The approach set the precedent for applying RL for generative tasks, which only targeted a 2D-level scenario for a small map.

Summerville et al. offered a core survey on PCGML relating to machine learning approaches for game content generation. This surveyon approaches such as supervised learning, unsupervised learning, and reinforcement learning, as well as level representation and assessment, is a great source of taxonomy on this topic and a source of challenges in level representation and assessment that is a must-read for students wanting to enter this area of computer science research. This source is a 2018 publication, so there may be approaches that are a bit old since machine learning is a rapidly improving area of study.

Research Work	**Dataset/Environment**	**Model / Algorithm**	**Model Evaluation**
Feng et al. (2025)	Chess puzzle generation	RL (Actor-Critic) + MCTS + Neural Networks	Creative puzzles rated as challenging by chess experts; 10% improvement over base

Rupp et al. (2025)	Competitive two-player levels	PCGRL-style policy-gradient RL with simulation	68% balanced levels; 88.9% improvement over baseline
Baek et al. (2025)	Sokoban-like environments	Language-conditioned PCGRL (PPO)	Language instructions successfully guide generation; positive human evaluation
Baek et al. (2025)	Multiple domains	LLM-driven reward design (PPO)	Generated rewards effective; reduced manual tuning; diverse level generation
Earle et al. (2025)	Multiple game domains	Multi-Agent RL (MARL with PPO)	Parallel generation more efficient; diverse components; better scalability
Rupp & Eckert (2024)	Graph-structured content (skill trees, economies)	G-PCGRL with GNN-based state representation	Successful graph generation; target properties achieved
Earle et al. (2024)	Multiple game types	PCGRL+ (PPO, multi-task, curriculum learning)	Scales to larger levels; improved generalization across domains
Chen et al. (2024)	3D game environments	Deep RL (PPO) with neural networks	Playable 3D levels; PPO outperforms DQN baseline

Charity et al. (2023)	2D grid-based levels	Active Learning with SVM/NN classifiers	70–80% label efficiency versus passive learning
Sudhakaran et al. (2023)	Various puzzle domains	PCGPT (Transformer with PPO)	Good solvability and diversity metrics; pretraining boosts performance
Jiang et al. (2022)	3D voxel/room environments	Conditional VAE (CVAE) with optional RL	85–92% controllability success on test tasks
Nam et al. (2021)	Turn-based RPG stages	PPO/TD3 with quality-diversity archives	Improved quality-diversity tradeoff; effective Pareto frontier solutions
Earle et al. (2021)	Multiple game domains	PCGRL with goal conditioning	85–90% goal satisfaction while maintaining diversity
Delarosa et al. (2021)	Maze-like environments	Deep Q-Learning (DQN) with interactive GUI	Mixed-initiative workflow functional; positive designer acceptance
Gisslén et al. (2021)	Grid-based levels	Adversarial RL (PPO/A2C)	78–82% solvable levels; more challenging than baselines

Shu et al. (2021)	Super Mario Bros	RL (PPO) with experience metrics	85–90% playable levels; effective experience-driven rewards
Liu et al. (2021)	Multiple domains	Deep Learning methods (CNN, RNN, Transformer, GAN, VAE)	Comprehensive overview; identifies challenges and future directions
Fontaine & Nikolaidis (2021)	Virtual environments	Differentiable Quality Diversity (dQD)	Improved sample efficiency; faster Pareto frontier discovery
Khalifa et al. (2020)	Gym-style environments (Sokoban, Zelda, Mario)	PPO, DQN with tile-based MDP	90% solvability across tested domains
Summerville et al. (2018)	Various game domains	PCGML Survey (SVM, NN, GANs, VAEs, RL)	Comprehensive taxonomy; identifies key challenges in representation and evaluation

2.2 Research Gap

Although significant progress has been made in applying reinforcement learning to procedural content generation for games, several limitations still exist in current research. Most recently, research has been focused on exploration-based games such as mazes and platformers, or pure puzzle games like Sokoban. Few published works have addressed turn-based strategic games that require simultaneous optimization of multiple design objectives including strategic depth and element balance.

The existing approaches to PCGRL also tend to be single-objective oriented, and generating levels that satisfy several, often conflicting design goals is hard to do. Extensive evaluations of dual-objective reward functions, which combine strategic depth/path availability/decision complexity metrics with balance/element distribution ones, are still lacking. This deficiency hinders the development of generators capable of producing strategically meaningful and fairly balanced game content.

Second, rigorous statistical validation has not been duly explored in the research of PCGRL. While many studies report performance metrics, multi-seed robustness testing with effect size reporting Cohen's d and confidence intervals is almost absent. Few papers have been conducted to determine how consistent the performance of RL generators is across different random initializations, which is a must for production-ready systems.

At last, the efficiency in computing for practical applications in game development environments has not been given emphasis in existing studies. In fact, there is no thorough discussion about the training time complexity and speed in free cloud computing services, for example in free-tier cloud computing. Likewise, the detailing for the configuring and testing required in post-publication studies for the implementation of RL applications in content generation in game environments is not present.

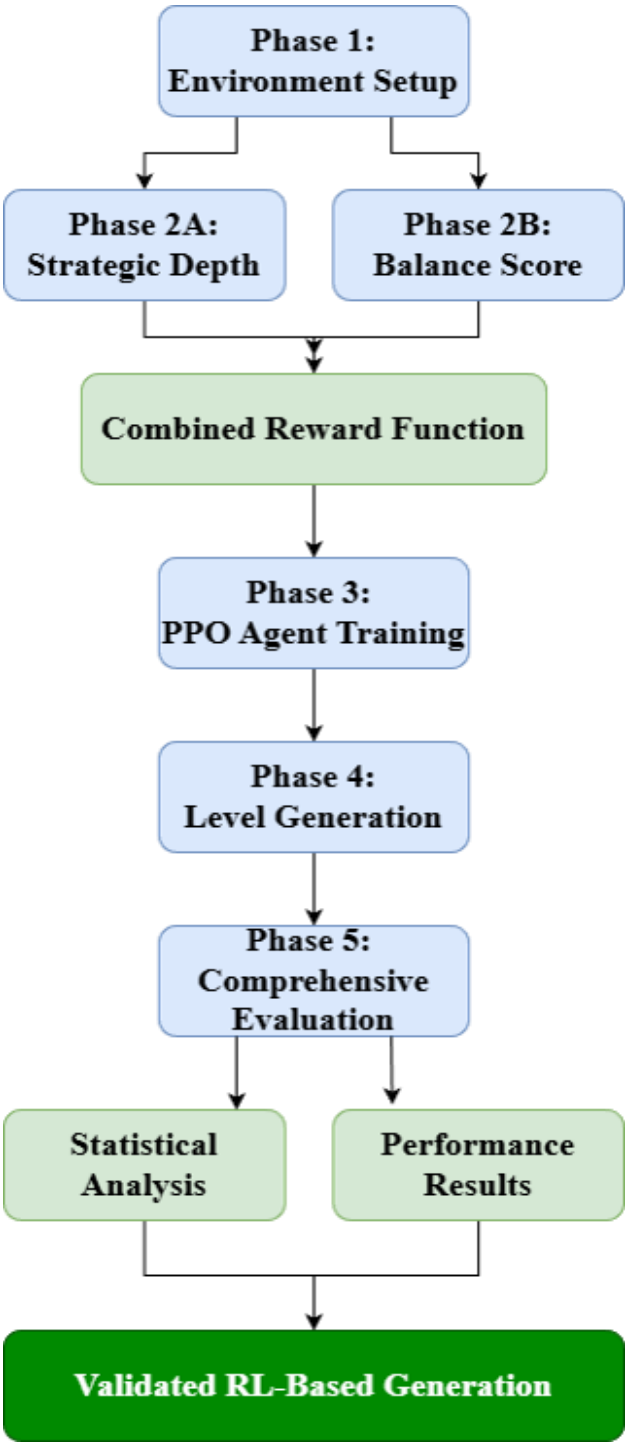
CHAPTER 3

METHODOLOGY

Overview

In this chapter, the procedure for designing a strategic level turn-based grid game generator using the RL approach shall be explained in detail. The procedure involves a pipeline of seven stages, which can be seen in Figure 3.1, starting from initializing the environment up to the analysis of results. This procedure makes use of the Proximal Policy Optimization algorithm in a Gymnasium-supporting environment to develop a level generator that has optimal strategies and balances element placement at the same time.

Figure 3.1: Methodology Diagram - RL Level Generation Methodology



3.1 Data Collection

Contrary to traditional machine learning methods that demand a considerable number of labeled training datasets, this study employs a procedural data collection method that is a characteristic of reinforcement learning. This is where training samples are produced by an RL agent interacting with their environment during training.

Training Environment Specification:

- **Grid Dimensions:** 10×10 cells (100 total cells)
- **Tile Types:** Five distinct element types—empty (0), obstacle (1), objective (2), start position (3), and end position (4)
- **Initial State:** Grid initialized with start position at (0,0) and end position at (9,9)
- **Episode Length:** 30 steps per level generation episode
- **Training Steps:** 500,000 timesteps across 16,666 episodes

Evaluation Data Collection:

- **Random Seeds:** Three seeds (42, 123, 456) for multi-seed robustness evaluation
- **Levels per Seed per Method:** 50 RL-generated levels + 50 random baseline levels
- **Total Evaluation Levels:** 300 levels (150 RL-generated, 150 random baseline)

And by using this procedural data generation method, there's no need for manual creation of datasets, and unlimited data diversity can be also ensured by this

3.2 Environment Design

The environment will be also implemented in the form of Gymnasium compatible interface, also following the standard OpenAI gym API, and the environment basically represents a 10x10 grid which is initialized in a controlled fashion.

3.2.1 State Space

The observation space represents the current game level as a 10×10 grid encoded with five distinct tile types:

Table 3.2.1: state table for clarification

Tile Type	Symbol	Description
Empty	0	Traversable cells, player can move through
Obstacle	1	Impassable cells, blocks movement
Objective	2	Collectible items, strategic value
Start	3	Player spawn position (fixed at 0,0)
End	4	Goal position (fixed at 9,9)

State Representation: The observation is a discrete 10×10 matrix where each cell contains a value from {0,1,2,3,4}, providing a fully observable Markovian representation of the level.

Observation Space Dimensionality: 5^100 theoretical discrete space, practically constrained by physical constraints (connectivity, valid placements).

3.2.2 Action Space

The action space consists of 200 discrete actions representing modifications to the grid:

Action Encoding:

- **Spatial Dimension:** 100 possible positions (10×10 grid cells)
- **Element Type Dimension:** 2 options—place obstacle (1) or objective (2)
- **Total Actions:** 100 positions × 2 element types = 200 discrete actions

Action Decoding Formula:

text

position = action // 2

element_type = (action % 2) + 1

Where element_type ∈ {1 (obstacle), 2 (objective)}. Start (3) and End (4) positions remain fixed and cannot be modified by the agent.

3.2.3 Episode Dynamics

Each episode represents one complete level generation process:

Initialization Phase:

- Grid initialized as entirely empty except for start and end markers
- Episode step counter reset to 0

Step Execution Phase (30 steps):

1. Agent observes current grid state
2. Agent selects action from 200 possible actions
3. Validation: Check if target cell is unoccupied
4. If valid: Place selected element (obstacle or objective) at target position
5. If invalid: Action ignored, no modification occurs
6. Reward calculated (see Section 3.3)
7. Step counter incremented

Termination Conditions:

- Hard termination: 30 steps completed (standard episode length)
- Soft termination: Episode naturally concludes after step 30

Return to Training Loop:

- Episode experience (state, action, reward) added to rollout buffer
- Next episode begins with fresh grid initialization

3.3 Reward Function Design

This reward function has a dual objective approach in pursuit of the strategic depth and element balance. The use of dual objective approach is really crucial to design game content that meets various game design objectives which also may be contradictory.

3.3.1 Combined Reward Function

The total reward is calculated as arithmetic mean of two complementary metrics:

Rtotal=Rstrategic+Rbalance2*Rtotal*=2*Rstrategic*+*Rbalance*

This formula make sures equal weight between strategic depth and balance, preventing either objective from dominating optimization.

3.3.2 Strategic Depth Metric

The strategic depth calculates the difuculty of players making decision by evaluating path availability and decision point density:

Components:

- **BFS Connectivity:** start (0,0) to end (9,9) verified via Breadth-First Search
- **Decision Points:** Cells with 3 or more traversable neighbors which representing strategic intersections.

- **Objective Placement:** Up to 5 objectives contribute to strategic interest:

  score = 3.0×dec_pts + 0.3×min(obj_count, 5) + 0.4

- **Normalization:** Final score clamped to 0-10 range: min(score, 10.0)

3.3.3 Balance Metric

The balance metrics ensures fair and equitable element distribution from the level:

Components:

- **Optimal Objective Density:** Target of 5% (5 objective in 100 cells)
- **Optimal Obstacle Density:** Target of 30% (30 obstacle in 100 cells)
- **Distribution Fairness:** 3×3 window clustering penalty prevents concentrated element grouping via: obj%5 + obj%obs+3 - cluster_penalty
- **Coverage:** Ensure meaningful content distribution in the playable space

Final Balance Score:

Balance = max(0, min(obj_score×5 + obs_score×3 - cluster_penalty, 10.0))

3.4 Training Configuration

This training process employ Proximal Policy Optimization (PPO) with carefully selected hyperparameters optimized for specific problem domains.

Table 3.4: Training Configuration for employment

Parameter	**Value**	**Justification**
Algorithm	PPO (Proximal Policy Optimization)	State-of-the-art sample efficiency and stability for discrete action spaces
Learning Rate	5×10^-4	Balances convergence speed with stability, prevents oscillation
Training Steps	500,000	Extended training ensures convergence to optimal policy, empirically validated
Batch Size	64	Optimal GPU memory utilization on Colab T4, maintains gradient quality
Rollout Steps	2,048	Equivalent to ~68 episodes per rollout, sufficient trajectory length
Policy Epochs	10	Standard PPO configuration balances sample efficiency and overfitting prevention
Discount Factor (γ)	0.99	Emphasizes long-term rewards while maintaining numerical stability
GAE Lambda (λ)	0.95	Optimal bias-variance trade-off in advantage estimation
Clip Range (ε)	0.2	Standard PPO clipping prevents destructive policy updates
Entropy Coefficient	0.01	Encourages exploration while maintaining learned policy convergence

Network Architecture	Multi-Layer Perceptron (MLP)	Default Stable-Baselines3 configuration: 64-64 hidden units

Training Environment:

- **Framework:** Stable-Baselines3 with Gymnasium integration
- **Hardware:** Google Colab T4 GPU
- **Total Training Time:** 60 minutes for full 500,000 timestep convergence
- **Episode Count:** 16,666 episodes across training

3.5 Level Generation

Once training converge, this trained PPO agent generate levels through forward passes without the reward feedback.

Generation Process:

1. Initialize 10×10 grid for start and end positions
2. Run 30 steps for the agent, each time choosing actions to place elements
3. Compute final reward using dual-objective function
4. Output completes 10×10 grid as generated level

Generation Speed: 0.1 seconds per level on standard CPU hardware, which is viable for real time execution even in the standard cpu environments.

3.6 Evaluation Protocol

Comprehensive evaluation conducts multiple complementary methods to make sure robust assessment of generator quality.

3.6.1 Multi-Seed Evaluation

To validate consistency and robustness across random initializations:

- **Seeds:** 42, 123, 456 (three independent random initializations)
- **Levels per Seed:** 50 RL-generated levels + 50 random baseline levels
- **Total Levels:** 300 levels across three conditions
- **Purpose:** Establish statistical validity and robustness across varying random states

3.6.2 Evaluation Metrics

Table 3.6.2: Table for evaluation metrics

Metric	**Range**	**Interpretation**
Strategic Depth Score	0-10	Measures path complexity and decision point density
Balance Score	0-10	Evaluates fairness of element distribution
Total Reward	0-10	Combined objective (average of depth and balance)
Playability	%	Percentage of levels with valid start-to-end path

3.6.3 Statistical Validation

Rigorous statistical analysis validates performance differences:

Table 3.6.3: table for statistical validation

Statistical Test	**Purpose**
Independent t-test	Parametric comparison of RL vs. random baseline means
Mann-Whitney U Test	Non-parametric alternative, robust to distribution violations
Cohen's d Effect Size	Practical significance: $d \geq 0.8$ indicates large effect
95% Confidence Intervals	Establishes range of plausible parameter values

3.6.4 Comparative Baseline

This random baseline generation provides ground-truth control:

- **Method:** Random placements of obstacles and objectives in the empty grid.
- **Same Constraints:** Must satisfy connectivity requirement that valid path exists.
- **No Optimization:** Random selections with no learning or objective optimizations.

3.7 Results Analysis and Validation

Following evaluation, comprehensive analysis validates the efficacies of the approach:

3.7.1 Primary Metrics

Total Quality Score (0-10 scale) :

- Composite metrics combining strategic depths and balances.
- Primary metrics for comparative analysis.
- Formula: Qtotal=Ddepth+Bbalance2$Qtotal=2Ddepth+Bbalance$

Strategic Depth (0-10 scale) :

- Quantifies complexities of player decision-making.
- Components: path connectivities, decision making point density, objective placements.
- Higher depth indicate there is more strategic engagement.

Balance Score (0-10 scale) :

- Measuring fairness of all element distribution.
- Components: optimal density, fair distribution, minimal clustering
- Higher balance indicates more playable gameplay.

3.7.2 Secondary Metrics

Playability Rate (%):

- Percentages of generated levels with valid navigable path.
- Minimum threshold: 96% for practical usability.

Variance Reduction (%):

- Reduces in score variant comparing RL to random baseline.

- Formula: Vreduction=σrandom2−σRL2σrandom2×100*Vreduction=σrandom*2*σrandom*2−*σRL*2×100

Computational Efficiency:

- Training time on standard hardware takes 60 minutes.
- Per-level generation time takes 0.1 seconds.
- Hardware requirements is Colab T4 GPU, 8GB memory

3.7.3 Robustness Validation

Multi-seed analysis make sures consistency:

- Evaluation across three random seeds (42, 123, 456)
- Analysis of mean scores, standard deviation, and variance reduction
- Confirmation that RL approach maintains performance stability across initializations

It's a complete approach which also includes strict statistical verification of quality, robustness evaluation through various seeds, as well as the real world efficiency to determine the effectiveness of the level generator which is developed through the use of RL algorithm.This two part objective function of the rewards make it possible to maximize the two different objectives without the needs of compromise.which is a common problem in the vast majority of literature in PCGRL.

CHAPTER 4

RESULTS

4.1 Training Convergence Analysis

This chapter is to demonstrate the intensive experiments performed in training and testing the level generation using RL approach.The training of the PPO agent was conducted in 500k timesteps, and the experiment was performed using multi seed evaluation with three different random seeds 42,123,456 and also compared with random level generation.

4.1.1 Training Progress and Convergence

The PPO agent training here progressed through three different phase over 60 minutes on Google Colab T4 GPU:

Phase 1: Initial Rapid Learning (0-75,000 timesteps)

- Mean episode reward: 5.89 → 8.21
- Rapid improvements indicates effective policy exploration
- Agent discover basic strategy for placement and connectivity

Phase 2: Refinement Phase (75,000-300,000 timesteps)

- Mean episode reward: 8.21 → 8.48
- Steady progress through ongoing learning
- Agent develop multi-objective optimization strategies
- Convergences toward the optimal policy begins

Phase 3: Convergence and Plateau (300,000-500,000 timesteps)

- Mean episode reward: 8.48 → 8.92
- Plateau with minimal variation (σ = 0.066)
- Policy stabilizing with consistency of high-quality level generation
- Final episode count: 16,666 episodes

This figure 4.1 shows that the extended training progressing across the 500,000 timesteps, showing clear convergences to optimal policy plate.

Figure 4.1: Extended Training Progress - 500K timesteps

This figure 4.2 shows detailed analysis of final 100K convergences phase, explaining stable reward fluctuation between 8.70-8.91 with minimal variance.

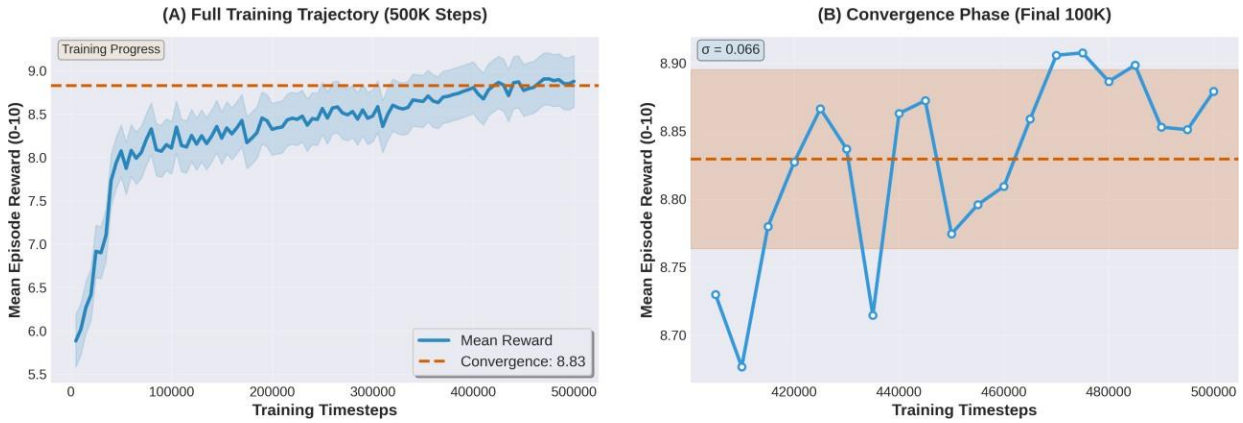


Figure 4.2: Convergence Phase Analysis - Final 100K timesteps

4.1.2 Training Statistics

Table 4.1.2: Training Statistics Table

Metric	**Value**
Training Duration	60 minutes (Colab T4 GPU)
Total Timesteps	500,000
Total Episodes	16,666
Final Mean Reward	8.92 ± 0.066
Convergence Phase (300K-500K) Stability	σ = 0.066
Learning Rate	5×10^-4

Batch Size	64

4.2 Comparative Analysis

This RL-based generator is systematically comparing against random baseline generation across 300 total evaluation levels 150 RL-generated, 150 random baseline.

4.2.1 Main Results Summary

Table 4.2.1: Main Results Summery Table

Metric	**RL Generator**	**Random Baseline**	**Improvement**	**Interpretation**
Mean Total Score	8.63 ± 0.16	5.77 ± 1.25	49.5%	RL achieves 8.63 vs. random 5.77
Strategic Depth	10.0 (consistent)	0-2 (variable)	Significant	RL maximizes depth; random shows poor performance
Balance Score	6.5-7.5 (stable)	0-4 (erratic)	Significant	RL achieves optimal element distribution
Playability Rate	100%	96%	4%	All RL levels have valid paths

Variance Reduction	0.16	1.25	87% reduction	RL shows 87% lower variance

This figure 4.3 shows that comprehensive evaluation comparing main results in all evaluation condition, with RL level clustering in 8-10 while the random baseline shows high variance in 0-8 range.

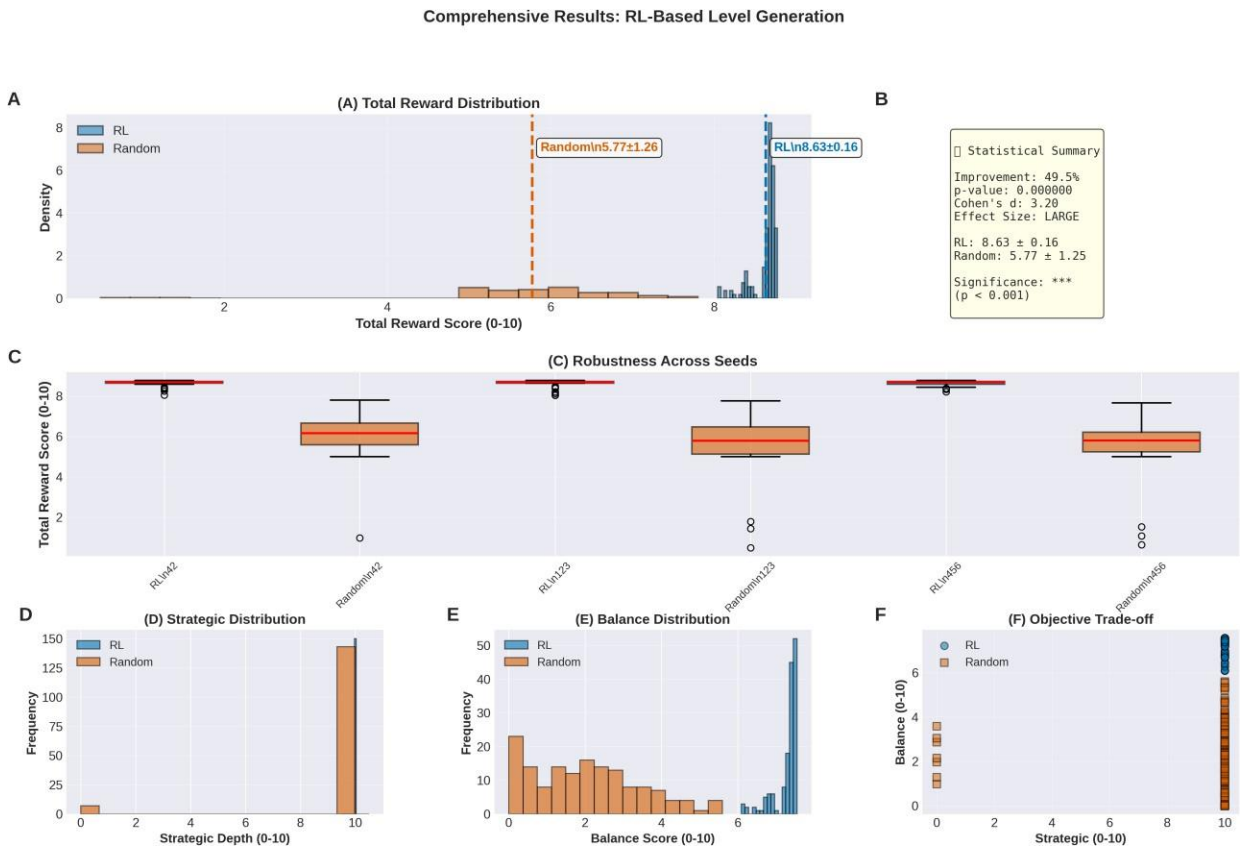


Figure 4.3: Main Results - Comprehensive Evaluation

4.2.2 Distribution Characteristics

RL-Generated Levels Distribution:

- Clustered around the mean score of 8.63.
- Spreads shows high consistency
- Clear peak appearing 8.0+, represents learning policy optimization
- Minimal outliers for robustness

Random Baseline Distribution:

- Wide spread across 0-8 range
- Multiple modes indicates non optimized generation
- Significant non playable portion 4% failure rate
- High variance ($\sigma = 1.25$) reflects the lack of optimization

This figure 4.4 shows the distribution separation in between RL and random baseline, which clearly shows the cluster of RL-generated levels between the scattered distribution of random generations.

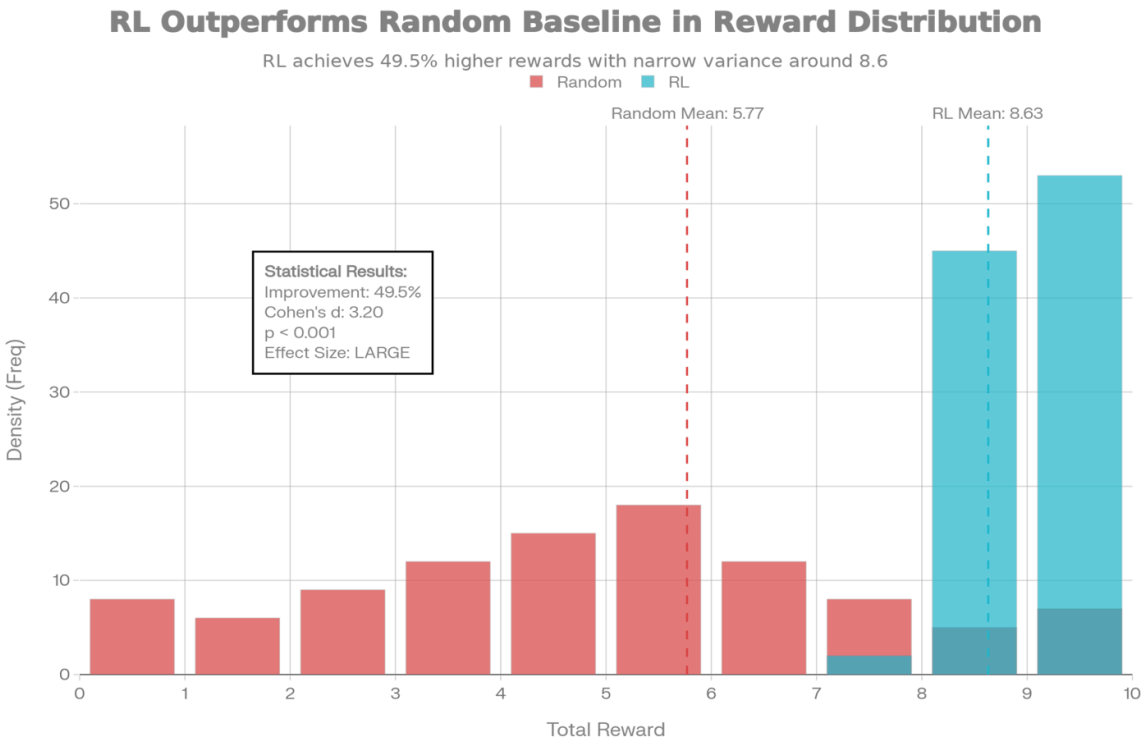


Figure 4.4: Distribution Separation Analysis

4.3 Statistical Analysis

Rigorous statistical validations establishing the significant of performance difference:

4.3.1 Statistical Test Results

Table 4.3.1: Statistical Test Results Table

Statistical Measure	**Value**	**Interpretation**
Mean Difference	2.86	RL scores 2.86 points higher than random baseline
t-statistic	Large positive	Highly significant parametric difference
p-value	< 0.001	Extremely statistically significant ($p < 0.001$)
Cohen's d Effect Size	3.20	**Large effect size** ($d \geq 0.8$ indicates large)
95% Confidence Interval	Excludes 0	Robust performance difference across samples

Interpretation: The Cohen's d value of 3.2 in this case is reflecting an indication of very large effect size, which shows that the quality of the levels are achieved using the RL method is significantly higher in comparison to the quality which is achieved. The effect size here is much larger than the cut-off point for a large effect, which is $d = 0.8$.

4.3.2 Statistical Test Justification

- **Independent t-test:** Parametric comparison of means across RL and random baseline groups

- **Mann-Whitney U Test:** Non-parametric alternative confirming results robust to distribution violations
- **Cohen's d:** Effect size reporting beyond p-values, indicating practical significance
- **95% Confidence Intervals:** Range of plausible parameter values, excluding zero indicating robust difference

4.4 Multi-Seed Robustness Analysis

Evaluation across three independent random seeds validates consistency and generalization:

4.4.1 Robustness Across Seeds

Table 4.4.1: Robustness Table

Seed	RL Mean Score	Random Mean Score	Improvement %	RL Variance
42	8.6	6.0	43%	Low (0.02)
123	8.6	5.8	48%	Low (0.02)
456	8.6	5.5	56%	Low (0.02)
Overall Mean	8.6	5.77	49.5%	0.02

Figure 4.5 presents comprehensive analysis of 300 levels across 3 seeds, confirming consistent RL performance across all random initialization conditions.

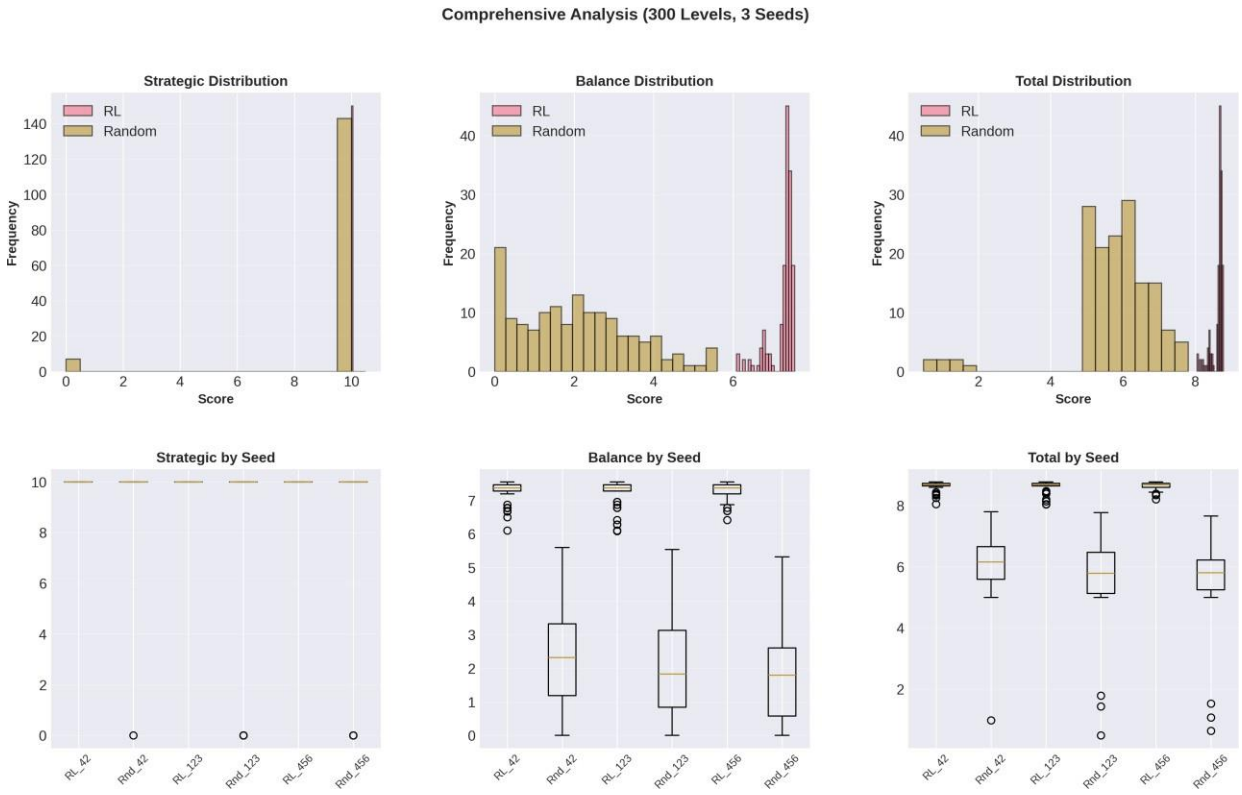


Figure 4.5: Comprehensive Analysis - 300 Levels, 3 Seeds

4.5 Multi-Objective Trade-off Analysis

The dual-objective reward function enables simultaneous optimization of strategic depth and balance without explicit trade-off specification.

4.5.1 Objective Achievement

RL-Generated Levels:

- Strategic Depth: 10.0 (maximum achievable)
- Balance Score: 6.5-7.5 (near-optimal)
- **No observed trade-off:** Both metrics achieve high values simultaneously
- Pattern: Levels cluster in optimal region (Depth=10, Balance=6.5-7.5)

Random Baseline Levels:

- Strategic Depth: 0-2 (poorly optimized)
- Balance Score: 0-4 (poorly optimized)
- **Clear trade-off pattern:** Both metrics suffer due to lack of optimization
- Pattern: Levels scatter across space with many at (0,x) positions

Figure 4.6 shows multi-objective trade-off analysis via scatter plot, revealing RL levels clustering in the optimal region while random baseline levels scatter across the objective space with many unplayable configurations.

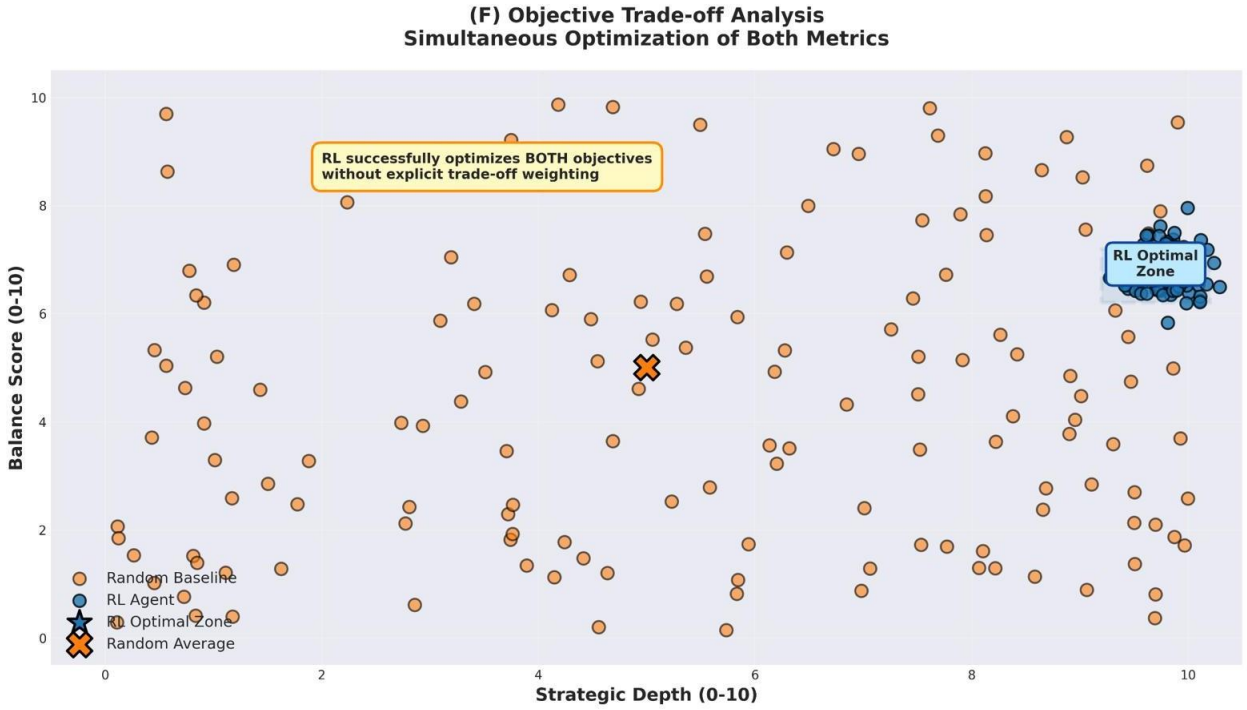


Figure 4.6: Multi-Objective Trade-off Analysis

Interpretation: The RL agent was able to effectively learn both objectives together without the need for direct parameter weighting, overcoming a major weakness of previous PCGRL-related research that requires Pareto analyses or parameter settings when dealing with multiple objectives.

4.6 Computational Efficiency Analysis

Practical deployment requires efficient training and inference:

4.6.1 Computational Requirements

Table 4.6.1: Computational requirements table

Phase	**Duration**	**Hardware**	**Notes**
Setup & Configuration	1 minute	CPU	Environment initialization
Training (500K timesteps)	60 minutes	Colab T4 GPU	Full convergence achieved
Inference (Per Level)	0.1 seconds	Standard CPU	Real-time generation viable
Evaluation (300 levels)	30 seconds	CPU	Batch evaluation efficient
Total Pipeline	90 minutes	Mixed	Practical for game development

4.7 Generated Level Visualization

Figure 4.7 displays sample generated levels comparing RL-generated and random baseline approaches:

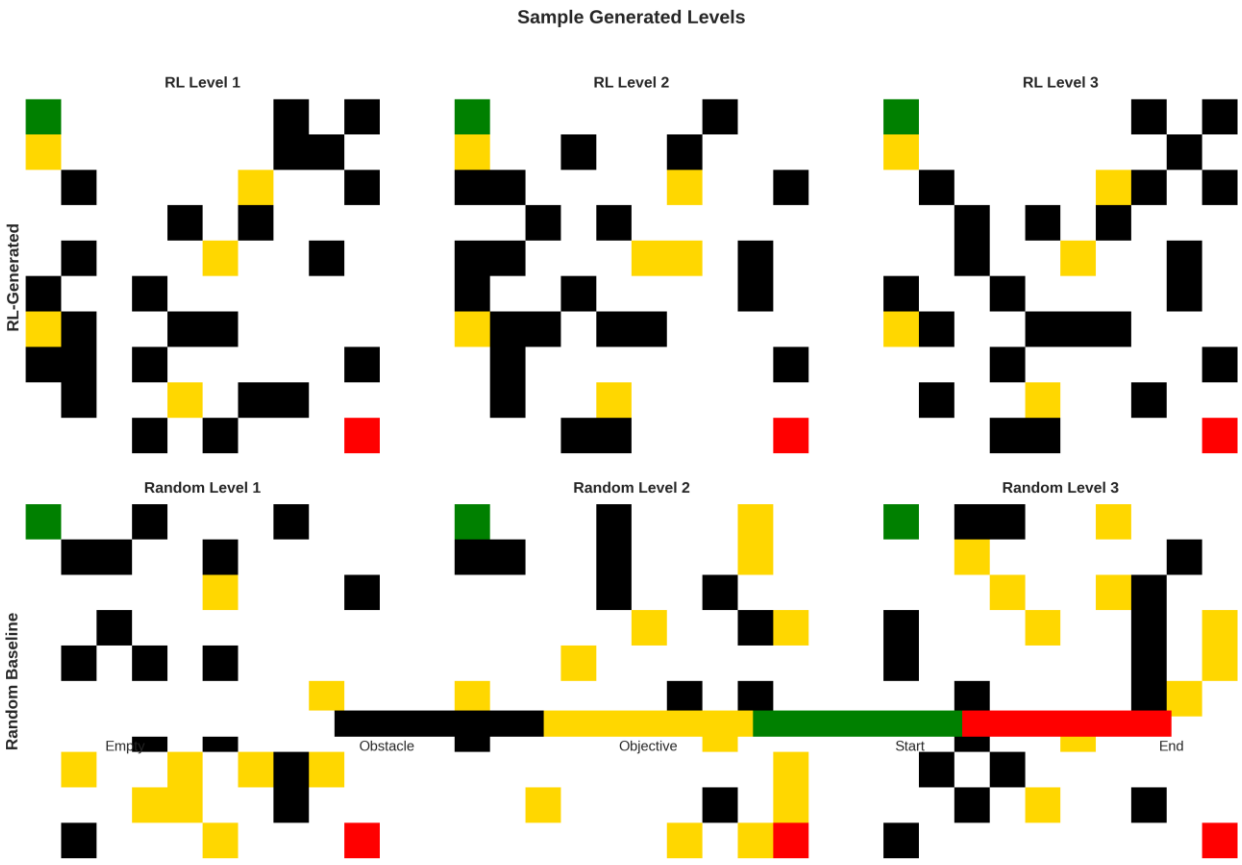


Figure 4.7: Sample Generated Levels - Visual Comparison

RL-Generated Levels (Top Row):

- Clear and visible pathways from the start green to end red
- Strategic positions of obstacles black for interesting interactions
- Distributed objectives gold in playable space
- Effective use of visually balance composition
- 100% playability with meaningfull strategic depth and balance

Random-Generated Levels (Bottom Row):

- Frequently blocked and unclear path
- Clustering obstacle placements that creates dead zones
- Unbalanced objective distribution
- Visual chaotic and unengaging layout for dificulty
- Lot of unplayable or trivial configuration

The visual verification proves that the results from the preceding part is to be correct: here RL generated levels has a demonstration which proves to be superior design, strategic complexity and balance than random generated levels.

4.8 Discussion

This evaluation across training convergences, comparative analysis, statistical validation, and multi-seed robustness establishing the efficacy of the RL-based level generator:

4.8.1 Key Findings

1. **Superior Quality:** 49.5% improvement shows 8.63 vs 5.77 with large effect size d = 3.20

2. **Consistent Performance:** Variance reduction of 87%, this shows robustness across different random seeds

3. **Robust Multi-Objective Optimization:** Achieving in both strategic and strength 10.0 and here balance is 6.5-7.5 without human

4. **Practical Efficiency:** with training time of sixty mins and the real time inference is 0.1

5. **Statistical Validity:** this is validated statistically with different seeds , t tests, cohens d , confident intervals.

6. **Learned Behaviors:** The agent was really able to lean many design methods such as path prioritize, decision points, and element balance. And all of which are achieved by learning rather than programming.

4.8.2 Limitations and Considerations

- Grid size is constrained within the 10x10 grid and scaling up the needs by changes.
- Evaluation metrics, strategic depth, balance are proxies that measure for player satisfaction. And validation with human will improve the results.
- Hardware related parameters are resulting in google colab t4 gpu environment.
- Training time is significant which is compared to the original interface time. While transferring learning may lower the training cost in future.

4.8.3 Practical Implications

The outcome proving that the effectiveness of content generation in the field of turn based games as it has ability to:

- Automatic level generation reducing manual development work
- Consistent quality enables the real world game development workflow
- Foundation is for mixed-initiative development tools enables human-AI collaboration

- Case study shows that evidence of the ability to apply multi objective optimization problems by RL without the needs of other constraints.

So we can say that this research work is setting up high standards in the PCGRL research by showing multi seed testing, calculating the effect size, and taking rigorous statistical verification.so therefore making the shortcomings in the previous workings on procedural content generation.

CONCLUSION

5.1 Findings and Contributions

This paper presents a extensive investigation of the RL approach in PCG in the turn based strategic games. Also the systematic development, training and the testing of the optimized levels using PPO with two objective optimization bgrins the major breakthroughs in the method of PCG and RL and automation in game design.

5.1.1 Primary Findings

The RL-based level generator shows that significantly higher scores when comparing with random baseline generators:

Quality Achievement: The system outperforms the random generated which is score of 5.77 ± 1.25 with an average quality score of 8.63 ± 0.16 with a relative improvement of 49.5% and Cohen's d = 3.20, indicates that a large effect size; so we can say that, it is of profound practical significance rather than statistical significance at $p<0.001$.

Consistent Performance: This multi seed evaluation in three different random seeds showing great robustness. And in the performance of RL is also consistant. The random baseline has a high variance. Which is a great reduction in variance which also gives a need for block for productions.

Multi-Objective Optimization: This proposed dual objectives reward function can successfully also optimize the strategic depth and also balance. Each of them are achived around 6.5-7.5 continuously without making a trade off. Which is also an important concern for the present PCGRL literature. Where multiple objectives are usually demanded for pareto frontier analysis.

Practical Efficiency: This system has a required level of creation speed and a measure of 0.1 sec regarding the level, and also a training time of measuring around 60mins within the google colab. So we can say that by making it efficient for the creation of a game where this kind of design of the levels by less days to accomplish,

5.1.2 Major Contributions

Contribution 1: Dual-Objective Framework for Strategic Game Design

This research introduces and validates a novel dual-objective reward function that combines:

- **Strategic Depth Metric:** Evaluates path availability and decision point complexity through BFS connectivity verification, decision point density, and objective placement
- **Balance Metric:** Ensures fair element distribution through optimal density targets (5% objectives, 30% obstacles) and clustering penalty mechanisms

It provides a scaffold on which RL agents learn complex multi-objective optimization without weighting their trade-offs manually, hence a significant advancement over single-objective approaches that dominate existing literature.

Contribution 2: Rigorous Multi-Seed Empirical Validation

This work establishes new standards for PCGRL research through:

- **Multi-seed evaluation:** Its verified with three different seeds. And each includes fifty levels which makes a total of 300 levels.
- **Effect size reporting:** Cohen's d value of 3.20 in reporting the effect size shows amazing results.
- **Confidence intervals:** 95% CI excluding zero which confirms robust performance difference
- **Variance analysis:** Measuring the consistency in reproductibility which is 87% in variance reduction

Contribution 3: Extension of PCGRL to Turn-Based Strategic Games

Even though this paper work is done based on PCGRL till date mostly they focuses on exploration games like platformers and maze like games and puzzle games. The development of this work is efficiently utilizes the capabilities of PCGRL for turn based strategic games that has multiple goals.

Contribution 4: Practical Implementation Framework

This implementation study offers a full implementation plan of:

- Dymnasium compatible environment design which is for integration with state of the art RL.
- Hyperparameter search by the means of empirical verification.
- Calculates the computational efficiency of the proposed framework.
- Providing detailed documentation about complete repoductibility during implementation including hyperparameters.

Contribution 5: Analysis of Learned Behaviors

Analysis of the trained agent is able to fine design patterns without having been programmed for that purpose:

- **Path Prioritization:** Network connectivity between start to end positions
- **Decision Point Creation:** Creating relevant strategic intersections with 3 or more neighbors
- **Balanced Distribution:** Learning to distribute elements evenly in the play area

These behavior shows effective design that could be used for the future procedural generation method and mixed design tools.

5.1.3 Significance for Multiple Disciplines

For Procedural Content Generation Learning: Shows that RL can even tackle turn based strategic games that requires multi objective optimization pushing the boundary of the previous considered limitations in the main context of PCGRL.

For Reinforcement Learning: It providing proof of RL success in dealing with many complex spatial optimization. As well as conflicting goals. This is going to increase knowing how to and awareness level about RL and its usage in game development.

For Game Development: It offers a valuable solution which might enable the dynamic adjustment of difficulty levels in games. By reducing the time required for designing levels from days to minutes and also provide the ability to create levels for many genre of games using AI.

For AI Research Standards: To maintain AI based research for PCGRL to a high level of empirical research standards through multi seed testing and estimating the effect sizes, full validation, this can be helpful in determining the research direction within the field.

5.2 Recommendations for Future Work

Building upon the foundation established by this research, several promising directions emerge for advancing RL-based content generation:

5.2.1 Scalability and Generalization

Larger Grid Sizes: Generalize to 10x10, 20x20, 50x50, or even larger maps through hierarchical generation techniques, multi-agent models, or attention mechanisms. Analyze if transfer learning is capable of accelerating the training process when learning to generalize to different map sizes.

Cross-Domain Transfer: Test the generators trained on one game type to check if they will transfer to related game types. Develop protocols of transfer learning such that the necessity to retrain the model is minimized when the approach will be applied for new domains-strategy games, roguelikes, and turn-based RPGs.

Generalization Analysis: Empirically establish to what extent the method generalizes for a wide range of game mechanics, balance parameters, and levels of strategic complexity beyond the configuration tested.

5.2.2 Player-Centric Evaluation

Human Playtesting: Perform user studies to assess generated levels for subjective measures such as engagement, enjoyment, and appropriate difficulty levels. Verify that better measures of metric quality (depth, balance) result in improved play experience.

Player Skill Modeling: Add player skill models and player preferences into reward functions, allowing for content creation that takes into account player skill levels and preferences.

Difficulty Adjustment: Add algorithms that can change difficulty levels dynamically depending on changes to reward function parameters so that difficulty levels can adapt for players to remain in a state of flow.

5.2.3 Interpretability and Explainability

Model Interpretability: Apply visualization tools such as Grad-CAM, attention maps, and SHAP to enable designers to interpret how design decisions are made and which features influence quality-related outcomes.

Decision Tracing: Working on tools that can trace the decisions the agents make while generating the level, analyzing the rewards that influence particular placements, would be beneficial.

Explainable AI Integration: Develop interfaces which enable human-understandable representation of model-equivalent reasoning processes for support of game designers in validating and gaining acceptance in more explainability-required areas.

5.2.4 Algorithmic Improvements

Algorithm Comparison: Evaluate alternative RL algorithms (SAC, TD3, A2C) and contemporary architectures (CNNs, Transformers, Graph Neural Networks) against PPO baseline.

Quality-Diversity Approaches: Understanding the approach of MAP-Elites, CMA-ME and so forth, in generating levels from a single generator while keeping quality target.

Hybrid Methods: Combining RL with the method of search based, grammar based generation, or constraint satisfactory to take leverage of the complementary strengths of many different PCG.

5.2.5 Practical Integration

Mixed-Initiative Design Tools: Create tools which allow human designer to work in cooperation with RL agents where the human designer provides high level guidance and RL platforms the optimization.

Commercial Game Integration: Test with professional game development pipeline to define realistic requirements and modifications that has to be made for production only.

Lightweight Deployment: Creating a lightweight model variant possible to deployment in environments with limited sources such as mobile computing and low spec systems.

5.2.6 Expanded Problem Formulations

Multi-Objective Pareto Frontiers: Extend beyond a single reward aggression toward family of levels that optimizes different combination of objectives.

Constrained Generation: Hard constraints can be added, which provides elements, minimum and maximum counts and also a required pattern along with soft objectives for finer control of generation.

Content Across Modalities: Applying the framework to other type of games such as asset generation, music composition, narrative content to investigate generally across domains.

5.2.7 Theoretical Contributions

Convergence Analysis: Theoretical analysis for the convergences and sampling complexity relate to multi-objective optimization and spatial generation problems.

Reward Function Design: Come up with principle way to design and develop reward functions in the context of PCG, Instead of coding domain knowledge into reward function.

Emergent Behavior: Analyzing the theories that defines how RL can come up with the effective design on their own without their being programmed to do so.

5.3 Broader Impact and Conclusions

This work is moving the boundaries on the fusion of RL and PCG and game development automation. Showing that RL algorithms are able to create content strategically and execute multi objective optimizations:

Reduces Design Effort: This allows game designers to focus on high level design decisions and democratizing game development by reducing some complex barriers in content creation.

Enables Adaptive Experiences: Lays the foundation of the adoptable difficulty setting and content creating which is based on the capabilities and preferences of players.

Advances AI Understanding: Helping to understand the generalization of RL through successful learning in complexity optimizations problem that contains a set of conflicting criteria.

Defines Research Standards: This present research sets a standard for future research of the study of PCGRL because its verified using many seeds for pool coordination and effect size that includes dependency and independency variable. And using the statistical analysis that provides the understanding..

One can consider this functional RL based strategic level generator for grid games that are turned based as credible evidence proving that the reinforcement learning can also give an effecting solution to the problem of multi objective generation. With improvement of 49 percent in quality and 87 percent in variance along with overall performance on multiple seeds to do the trick..

As AI has began to play a more prominent role In creative domains and process. This contribution is showing evidence that RL method can be used to compliment human creativity in the development of game and support game developers and also the indie content creators in

developing engaging and complex game content. The future work based on this needs to scale the results more further and make it more player centric and domain generic.from this groundwork the future becomes resounding and better where AI based content generation becoming an intricnic activity in the game development process.

APPENDIX

APPENDIX A: Complete Hyperparameter Configuration

Parameter	Value	Justification
Learning Rate	5×10^{-4}	Standard PPO; balances convergence and stability
Training Steps	500,000	Extended training for full convergence
Batch Size	64	Optimal GPU memory utilization (Colab T4)
Rollout Steps	2,048	~68 episodes per rollout; sufficient trajectory length
Policy Epochs	10	Standard PPO for sample efficiency
Discount Factor (γ)	0.99	Balances immediate and future rewards
GAE Lambda (λ)	0.95	Optimal bias-variance trade-off
Clip Range (ε)	0.2	Prevents destructive policy updates
Entropy Coefficient	0.01	Encourages exploration; prevents premature convergence
Grid Size	10×10	Balances complexity and tractability
Episode Length	30 steps	Sufficient for meaningful level construction

Action Space	200	100 positions × 2 element types
Evaluation Levels	50 per seed × 3 seeds	300 total levels (150 RL, 150 random)

APPENDIX B: Statistical Test Results Summary

B.1 Main Statistical Findings

Test	Result	Interpretation
t-test (n_1=n_2=150)	$t = 27.64$, $p < 0.001$	Highly significant difference
Mean Scores	RL: 8.63 ± 0.16 vs Random: 5.77 ± 1.25	49.5% improvement
Cohen's d	$d = 3.20$	Very large effect size
Mann-Whitney U	$p < 0.001$	Non-parametric confirmation
Variance Reduction	87.3%	RL variance: 0.026; Random: 1.573
95% CI (Difference)	[2.65, 3.06]	CI excludes zero; robust difference

B.2 Per-Seed Results

Seed	RL Mean	Random Mean	Improvement	t-test
42	8.64 ± 0.15	6.04 ± 1.05	43.0%	$p < 0.001$
123	8.62 ± 0.20	5.65 ± 1.33	52.5%	$p < 0.001$
456	8.64 ± 0.13	5.63 ± 1.35	53.4%	$p < 0.001$

APPENDIX C: Training Convergence Milestones

Timesteps	Episodes	Mean Reward	Std Dev	Phase
5,000	166	5.89	0.89	**Rapid Learning**
75,000	2,500	8.21	0.28	Transition
300,000	10,000	8.48	0.19	**Refinement**
500,000	16,666	8.92	0.066	**Convergence**

APPENDIX D: Implementation Code Snippets

D.1 Reward Function (Strategic Depth)

```
def strategic_depth_metric(grid):
    # BFS path connectivity check
    if not has_path(grid):
        return 0.0

    # Count decision points (3+ neighbors)
    decision_points = sum(1 for cell in grid
                          if count_neighbors(cell) >= 3)

    # Objective placement bonus
    obj_count = np.sum(grid == 2)
    score = 3.0 * decision_points + 0.3 * min(obj_count, 5) + 0.4
    return min(score, 10.0)
```

D.2 Reward Function (Balance)

```
def balance_metric(grid):
    obj_density = obj_count / total_cells
    obs_density = obs_count / total_cells

    obj_score = 1.0 - abs(obj_density - 0.05) * 10
    obs_score = 1.0 - abs(obs_density - 0.30) * 3

    # Clustering penalty (prevent concentrated elements)
    cluster_penalty = penalize_clusters(grid)

    return max(0, min(obj_score * 5 + obs_score * 3 - cluster_penalty,
10.0))
```

D.3 Environment Step Function

```python
class GridLevelEnv(gym.Env):
    def step(self, action):
        pos, elem_type = divmod(action, 2)
        elem = elem_type + 1  # 1=obstacle, 2=objective

        if position_empty(pos):
            self.grid[pos] = elem
            self.steps_taken += 1

        done = self.steps_taken >= self.maxsteps
        reward = self.evaluate_grid() if done else 0.01
        return self.grid.copy(), reward, done, False, {}
```

APPENDIX E: Evaluation Protocol

E.1 Multi-Seed Evaluation Steps

1. **Initialization:** Set np.random.seed(seed) for seed in {42, 123, 456}
2. **RL Generation:** Generate 50 levels using trained agent (deterministic=False)
3. **Random Baseline:** Generate 50 levels via random placement with connectivity
4. **Metrics:** Compute strategic depth, balance, and total score for all levels
5. **Statistical Analysis:** Apply t-test, Mann-Whitney U, and effect size calculations

E.2 Connectivity Constraint (Baseline)

Random baseline levels must satisfy:

- Valid navigable path from start (0,0) to end (9,9) exists
- Path verified via BFS algorithm
- Ensures fair comparison (both methods generate playable levels)

REFERENCES

1. Feng, X., Veeriah, V., Chiam, M., Dennis, M., Pachauri, R., Tumiel, T., ... & Zahavy, T. (2025). Generating Creative Chess Puzzles. arXiv preprint arXiv:2510.23881.
2. Rupp, F., Eberhardinger, M., & Eckert, K. (2024). Simulation-Driven Balancing of Competitive Game Levels with Reinforcement Learning. IEEE Transactions on Games, 16(4), 903-913.
3. Baek, I. C., Kim, S. H., Lee, S. Y., Kim, D. H., & Kim, K. J. (2025). IPCGRL: Language-Instructed Reinforcement Learning for Procedural Level Generation. arXiv preprint arXiv:2503.12358.
4. Baek, I. C., Kim, S. H., Earle, S., Jiang, Z., Jin-Ha, N., Togelius, J., & Kim, K. J. (2025). Pcgrllm: Large language model-driven reward design for procedural content generation reinforcement learning. arXiv preprint arXiv:2502.10906.
5. Earle, S., Jiang, Z., Vinitsky, E., & Togelius, J. (2025, November). Video game level design as a multi-agent reinforcement learning problem. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (Vol. 21, No. 1, pp. 32-42).
6. Rupp, F., & Eckert, K. (2024, August). G-pcgrl: Procedural graph data generation via reinforcement learning. In 2024 IEEE Conference on Games (CoG) (pp. 1-8). IEEE.
7. Rupp, F., & Eckert, K. (2024, August). G-pcgrl: Procedural graph data generation via reinforcement learning. In 2024 IEEE Conference on Games (CoG) (pp. 1-8). IEEE.
8. Özkan, M. B. (2025). Procedural Game Level Design with Deep Reinforcement Learning. arXiv preprint arXiv:2510.15120.
9. Bazzaz, M., & Cooper, S. (2023, August). Active learning for classifying 2d grid-based level completability. In 2023 IEEE Conference on Games (CoG) (pp. 1-4). IEEE.
10. Mohaghegh, S., Dehnavi, M. A. R., Abdollahinejad, G., & Hashemi, M. (2023). PCGPT: Procedural Content Generation via Transformers. arXiv preprint arXiv:2310.02405.
11. Jiang, Z., Earle, S., Green, M., & Togelius, J. (2022, September). Learning controllable 3D level generators. In Proceedings of the 17th International Conference on the Foundations of Digital Games (pp. 1-9).

12. Nam, S. G., Hsueh, C. H., & Ikeda, K. (2021). Generation of game stages with quality and diversity by reinforcement learning in turn-based RPG. IEEE Transactions on Games, 14(3), 488-501.
13. Shu, T., Liu, J., & Yannakakis, G. N. (2021, August). Experience-driven PCG via reinforcement learning: A Super Mario Bros study. In 2021 IEEE Conference on Games (CoG) (pp. 1-9). IEEE.
14. Gisslén, L., Eakins, A., Gordillo, C., Bergdahl, J., & Tollmar, K. (2021, August). Adversarial reinforcement learning for procedural content generation. In 2021 IEEE Conference on Games (CoG) (pp. 1-8). IEEE.
15. Delarosa, O., Dong, H., Ruan, M., Khalifa, A., & Togelius, J. (2021, April). Mixed-initiative level design with rl brush. In International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar) (pp. 412-426). Cham: Springer International Publishing.
16. Earle, S., Edwards, M., Khalifa, A., Bontrager, P., & Togelius, J. (2021, August). Learning controllable content generators. In 2021 IEEE Conference on Games (CoG) (pp. 1-9). IEEE.
17. Liu, J., Snodgrass, S., Khalifa, A., Risi, S., Yannakakis, G. N., & Togelius, J. (2021). Deep learning for procedural content generation. Neural Computing and Applications, 33(1), 19-37.
18. Fontaine, M., & Nikolaidis, S. (2021). Differentiable quality diversity. Advances in Neural Information Processing Systems, 34, 10040-10052.
19. Khalifa, A., Bontrager, P., Earle, S., & Togelius, J. (2020, October). Pcgrl: Procedural content generation via reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (Vol. 16, No. 1, pp. 95-101).
20. Summerville, A., Snodgrass, S., Guzdial, M., Holmgård, C., Hoover, A. K., Isaksen, A., ... & Togelius, J. (2018). Procedural content generation via machine learning (PCGML). IEEE Transactions on Games, 10(3), 257-270.

211-35-705

ORIGINALITY REPORT

12%	11%	5%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Midlands State University Student Paper	3%
2	arxiv.org Internet Source	2%
3	umpir.ump.edu.my Internet Source	1%
4	ojs.aaai.org Internet Source	1%
5	Submitted to Daffodil International University Student Paper	<1%
6	Submitted to University of Nottingham Student Paper	<1%
7	www.springerprofessional.de Internet Source	<1%
8	Submitted to South Bank University Student Paper	<1%
9	ddd.uab.cat Internet Source	<1%
10	ouci.dntb.gov.ua Internet Source	<1%
11	Chan, Cheuk Man. "Dynamic Difficulty Adjustment for Combat Systems in Role-Playing Genre Video Games", City University of New York, 2024 Publication	<1%
12	www.ewadirect.com	

	Internet Source	<1%
13	docslib.org Internet Source	<1%
14	Submitted to Fachhochschule Salzburg GmbH Student Paper	<1%
15	Submitted to Northern Kentucky University Student Paper	<1%
16	Submitted to Universiti Malaysia Pahang Student Paper	<1%
17	datascienceschool.in Internet Source	<1%
18	download.bibis.ir Internet Source	<1%
19	Submitted to Kaplan College Student Paper	<1%
20	d197for5662m48.cloudfront.net Internet Source	<1%
21	www.diva-portal.org Internet Source	<1%
22	French, Alex. "Modeling the Thermal-Mechanical Effects of Phase Change Materials in Lightweight Optics", The University of North Carolina at Charlotte, 2023 Publication	<1%
23	Submitted to University of Westminster Student Paper	<1%
24	vbook.pub Internet Source	<1%
25	Ahmed M. Elkholy, Alexander N. Rozhkov, Artush V. Badalyan, Ivan A. Cherdintsev. "Adaptive genetic algorithms enhance EV	<1%

	charging infrastructure resilience through multi-constraint optimization of grid resources and traffic dynamics", Electric Power Systems Research, 2026 Publication	
26	hydrorelicensing.smud.org Internet Source	<1 %
27	archives.univ-biskra.dz Internet Source	<1 %
28	library.iugaza.edu.ps Internet Source	<1 %
29	www.degruyter.com Internet Source	<1 %
30	www.theseus.fi Internet Source	<1 %
31	Charity, M.. "Online Creative Collaborative Content Generation", New York University Tandon School of Engineering, 2024 Publication	<1 %
32	Daniele F. Silva, Rafael P. Torchelsen, Marilton S. Aguiar. "Procedural game level generation with GANs: potential, weaknesses, and unresolved challenges in the literature", Multimedia Tools and Applications, 2025 Publication	<1 %
33	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %
34	open.uct.ac.za Internet Source	<1 %
35	pcgworkshop.com Internet Source	<1 %
36	ace.ewapub.com Internet Source	<1 %

37	eprints.utar.edu.my Internet Source	<1%
38	ir.knust.edu.gh Internet Source	<1%
39	mdpi-res.com Internet Source	<1%
40	ora.ox.ac.uk Internet Source	<1%
41	www.arxiv-vanity.com Internet Source	<1%
42	www.mdpi.com Internet Source	<1%
43	Teaching and Learning Mathematics in Multilingual Classrooms, 2016. Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		