Re-trainable Procedural Level Generation via Machine Learning (RT-PLGML) as Game Mechanic

Seth Cooper
se.cooper@northeastern.edu
Northeastern University
Boston, USA

Emily Halina
ehalina@ualberta.ca
University of Alberta
Alberta, Canada

Jichen Zhu
jicz@itu.dk
IT University of Copenhagen
Copenhagen, Denmark

Matthew Guzdial
guzdial@ualberta.ca
University of Alberta
Alberta, Canada

ABSTRACT

We present re-trainable procedural level generation via machine learning (RT-PLGML), a game mechanic of providing in-game training examples for a PLGML system. We discuss opportunities and challenges, along with concept RT-PLGML games.

CCS CONCEPTS

• **Human-centered computing**;

KEYWORDS

procedural content generation, machine learning, video games

ACM Reference Format:
Seth Cooper, Emily Halina, Jichen Zhu, and Matthew Guzdial. 2023. Re-trainable Procedural Level Generation via Machine Learning (RT-PLGML) as Game Mechanic. In *Foundations of Digital Games 2023 (FDG 2023), April 12–14, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3582437.3587210

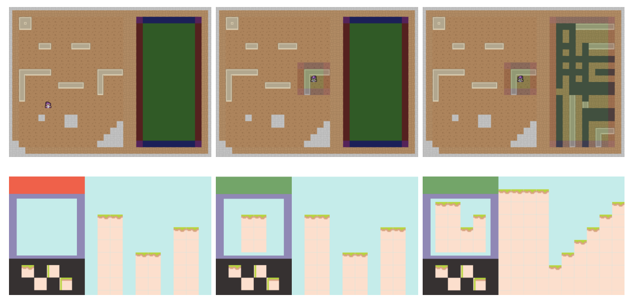

Figure 1: DungeonCollapse (top) and PatternCollapse (bottom) images.

1 INTRODUCTION AND BACKGROUND

Procedural Content Generation via Machine Learning (PCGML) [13] is an approach to procedural content generation based on learning to generate new content. In Summerville et al.'s survey of PCGML [13], they proposed that PCGML could be incorporated into game mechanics via roles such as "trainee", where players train PGCML to generate desired content, or "co-creator", where players and PCGML work together toward a goal and the PCGML learns from the player. These roles were derived from the AI design patterns work of Treanor et al. [14]. Later, Rieder [11] proposed several different game mechanics incorporating PCGML, including a "Train to Progress" mechanic, where the player must train PCGML to generate some missing piece of gameplay.

In this work we further develop re-trainable procedural *level* generation via machine learning (RT-PLGML) as a game mechanic; that is, re-training a level generator on new examples as a core mechanic. We propose that *allowing players to explicitly re-train level generators as a core game mechanic provides opportunities for research and player experience.* We discuss these opportunities, and provide example RT-PLGML game concepts.

Although there exist games that learn to adapt to players, or where players train in-game agents, we are specifically considering re-training PCGML level generators. In existing games where the player has some control over generating or re-generating parts of levels, typically the generators were trained (or had rules encoded) *before* game play, and are not *explicitly* re-trained by players in-game. For example, Dreams of Collapse [10] allows players to re-generate parts of the dungeon they are playing. Maureen's Chaotic Dungeon [16] uses WaveFunctionCollapse (WFC) [3] to generate platformer levels, and allows players to destroy and regenerate portions. Cavern Collapse [9], also using WFC, comes with a level editor that allows players to design some new training levels for gameplay.

2 OPPORTUNITIES AND CHALLENGES

To help guide future researchers and game designers in the investigation and creation of RT-PLGML games, we identify a number of (potentially overlapping) opportunities and open challenges. Each of these can provide novel and valuable game design problems and/or research projects.

• **ML Models:** RT-PLGML constrains what PCGML models may be appropriate. These models should be controllable and able to

FDG 2023, April 12–14, 2023, Lisbon, Portugal

ACM ISBN 978-1-4503-9855-8/23/04…$15.00
https://doi.org/10.1145/3582437.3587210

quickly incorporate new data, or the re-trainable condition is impossible. They must also allow for small datasets (since players can only provide a few examples) addressing the so-called “fundamental tension of PCGML” [6]. Thus approaches such as WaveFunctionCollapse [3] and n-grams [2] may be well suited. However, even these approaches may struggle without extension, opening up novel research problems. For example, basic parameters like the input and output size, or how much the input and output are already defined may all change in the course of play and models would have to adapt to this. RT-PLGML games would also benefit from natively interpretable models to aid in player understanding.

• **Data:** The choice of training data is another open challenge for RT-PLGML. For some games, it may make sense to pre-train a model, with the player’s goal being to understand or approximate the unseen training data. From there, a player could then finetune this pre-trained model after giving sufficient examples [5]. In this case, transfer and few shot learning methods have clear utility. In other cases it may make more sense to train from scratch, based on a small set of examples. Regardless of the starting point, the set of training data will be updated as the game progresses. This creates challenges: retaining all training data reduces the impact of each new addition, while removing older training data raises questions around which datapoints to remove. Automated approaches to analyzing training data like Shapely values may be relevant, but further research is needed [8].

• **Explainable ML and ML Education:** RT-PLGML relies on a player having some form of understanding of the underlying model in order to find solutions to in-game problems. This is impossible if the player cannot develop an accurate understanding of a model’s decision making process, which is typically the case with so-called black box models. This creates a clear opportunity for the application of explainable ML approaches, which seek to provide explanations for the behavior of black box ML models. There is also a need of XAI to assist in managing the vast amount of possible gameplay scenarios [17]. However, explainable ML approaches typically assume a post hoc analysis without time constraints, which would not be the case for a live, interactive experience. Design patterns have been used as an approach to explain PCGML, but only in a traditional, post hoc analysis [4]. RT-PLGML creates a challenge for the development of explainable in-game ML models, with the potential to extend to real world environments.

A player of an RT-PLGML game cannot be assumed to have any AI or ML background. Despite this, players will be expected to understand how a model learns, how to re-train the model, what biases may be present, and how to “debug” the model with additional or alternative training data. This presents a great opportunity for ML education, like games that attempt to teach players about aspects of ML such as bias in “Survival of the Best Fit” [1].

• **HCI and Player-AI Interaction:** RT-PLGML games could provide a fruitful area for human-computer interaction research. Despite people increasingly interacting with ML models in their day-to-day lives, how exactly users want to or would most benefit from interacting with ML models is still underexplored. Games provide an existing interaction paradigm that is low-stakes and lightweight. This should then simplify user testing and development issues. RT-PLGML can also support the investigation of specific HCI tasks, such as the study of player-AI interaction [18]. For example, related to explainable ML, we can investigate how different presentations (e.g. visualizations or animations) of models and algorithms impact player understanding and player mental models [15].

• **Novel Player Experiences:** One major reason to investigate or develop RT-PLGML games would be to provide new forms of play [18]. However, we anticipate these games to present significant game design challenges. A central design challenge around player experience is balancing designers’ authorial control and players’ much greater agency. Since different players can progress in the game differently, how can a designer give all of them a desirable player experience (e.g., rising difficulty level), and avoid unbalanced gameplay or uninteresting solutions dominating? As Rieder noted, “Train to Progress” games “offer an exciting approach for creating a unique player experience but require a very sophisticated game design and much balancing and polishing” [11, p. 46]. The challenge is clear, but solutions could provide entirely novel types of player interactions and experiences.

3 EXAMPLE GAME MECHANICS

We provide two example 2D, tile-based mechanics based on RT-PLGML. Images shown in Figure 1; with most visuals from Kenney [7]. In both games, the RT-PLGML mechanic *ML model* is based on WFC [3], as it works well with small training datasets; however other techniques, such as Markov Random Fields [12], may be appropriate. All training *data* is provided by the player in-game, and re-training happens from scratch.

• **DungeonCollapse:** The player controls a wizard in a top-down dungeon. The player places a staff on the ground to indicate the training example. WFC is then used to fill a blank region. Boundary conditions, indicated by color overlays, impact which tile can be placed in the blank region. Filling is animated, to improve the *explainability* of the generator. The player can only place their staff on dirt tiles, preventing use of the fill area itself for training. We envision different *player experiences* could be provided by games including: *progress puzzle* where the player provides examples that can fill in a level and allow them to make progress; *management*, where the player provides examples to generate levels that satisfy various personas of NPCs that wander around; *head-to-head*, where players generate levels difficult for other players, while completing levels generated for them; and *playground*, where the player makes levels they want to explore.

• **PatternCollapse:** Primarily envisioned as a *matching puzzle*, where the player makes an example in the tile editor, with the goal being to be able to generate a given level using WFC. Player may gain insight in the *explainability* of the model by exploring what desired patterns do and do not exist in their example.

4 CONCLUSION

In this work we have presented re-trainable procedural level generation via machine learning (RT-PLGML) as a game mechanic, and two game concepts centered around RT-PLGML game mechanics. We note these are not the only opportunities or ways to make such games, but show some potential of this direction. We hope to further develop the games to study RT-PLGML.

REFERENCES

[1] Gabor Csapo, Jihyun Kim, Miha Klasinc, and Alia ElKattan. 2019. *Survival of the Best Fit*. https://www.survivalofthebestfit.com

[2] Steve Dahlskog, Julian Togelius, and Mark J. Nelson. 2014. Linear levels through n-grams. In *Proceedings of the 18th International Academic MindTrek Conference: Media Business, Management, Content & Services*. 200–206.

[3] Maxim Gumin. 2016. WaveFunctionCollapse. https://github.com/mxgmn/WaveFunctionCollapse

[4] Matthew Guzdial, Joshua Reno, Jonathan Chen, Gillian Smith, and Mark Riedl. 2018. Explainable PCGML via game design patterns. *arXiv:1809.09419* (Sept. 2018).

[5] Emily Halina and Matthew Guzdial. 2022. Threshold Designer Adaptation: Improved Adaptation for Designers in Co-creative Systems. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. 4943–4949.

[6] Isaac Karth and Adam M. Smith. 2019. Addressing the fundamental tension of PCGML with discriminative learning. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*. 1–9.

[7] Kenney. 2022. Free game assets. https://www.kenney.nl/assets.

[8] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020. From local explanations to global understanding with explainable AI for trees. *Nature machine intelligence* 2, 1 (2020), 56–67.

[9] Darcy Matheson. 2022. Cavern Collapse. https://prometheusgamedev.itch.io/cavern-collapse

[10] Nathan Partlan. 2020. Dreams of Collapse. https://npcdev.itch.io/dreams-of-collapse

[11] Bernhard Rieder. 2018. *Using procedural content generation via machine learning as a game mechanic*. Master's thesis. University of Applied Sciences Technikum Wien, Vienna, Austria. https://github.com/bernhardrieder/PCGML-Game-Mechanics

[12] Sam Snodgrass and Santiago Ontañón. 2017. Learning to generate video game maps using Markov models. *IEEE Transactions on Computational Intelligence and AI in Games* 9, 4 (Dec. 2017), 410–422.

[13] Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K. Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. 2018. Procedural Content Generation via Machine Learning (PCGML). *IEEE Transactions on Games* 10, 3 (Sept. 2018), 257–270.

[14] Mike Treanor, Alexander Zook, Mirjam P Eladhari, Julian Togelius, Gillian Smith, Michael Cook, Tommy Thompson, Brian Magerko, John Levine, and Adam Smith. 2015. AI-based game design patterns. *Proceedings of the 10th International Conference on the Foundations of Digital Games 2015* (Dec. 2015).

[15] Jennifer Villareale and Jichen Zhu. 2021. Understanding mental models of AI through player-AI interaction. arXiv:2103.16168 [cs].

[16] Andy Wallace. 2019. Maureen's Chaotic Dungeon. https://globalgamejam.org/2019/games/maureens-chaotic-dungeon

[17] Jichen Zhu, Antonios Liapis, Sebastian Risi, Rafael Bidarra, and G Michael Youngblood. 2018. Explainable AI for designers: A human-centered perspective on mixed-initiative co-creation. In *2018 IEEE Conference on Computational Intelligence and Games*. 1–8.

[18] Jichen Zhu, Jennifer Villareale, Nithesh Javvaji, Sebastian Risi, Mathias Löwe, Rush Weigelt, and Casper Harteveld. 2021. Player-AI interaction: what neural network games reveal about AI as play. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–17.