

PCG-Based Game Design: Enabling New Play Experiences through Procedural Content Generation

Gillian Smith*, Elaine Gan[†], Alexei Othenin-Girard[†], Jim Whitehead*

Center for Games and Playable Media
University of California, Santa Cruz
Santa Cruz, CA, USA

*{gsmith, ejw}@soe.ucsc.edu [†]{egan, aotherin}@ucsc.edu

ABSTRACT

This paper discusses the concept of procedural content generation-based (PCG-based) game design as a way to create new kinds of playable experiences. We examine the different ways that PCG is currently used in games, and how that use impacts the meaning of the game and the player's experience. Finally, we discuss the design and implementation of an experimental PCG-based 2D platformer called *Rathenn*, which provides the player with control over the level they are playing while they explore both the physical and generative spaces of the game.

Categories and Subject Descriptors

K.8.0 [Personal Computing] General – Games.

General Terms

Design.

Keywords

Procedural level generation, game design, game design theory.

1. INTRODUCTION

Procedural content generation (PCG) has many purposes in game design. Perhaps its most common use is to promote replayability, since varied content can lead to drastically different play experiences. It has also been used to bypass technical limitations, as in the game *Elite* [2], in cases where there is not enough disk space or memory to store a game world. More recent efforts in both the industry and academia [1, 8, 16] have used PCG techniques to build games that adapt to a player's skill level or preferences, either at runtime or offline.

But while there are many games that use procedural content generation, most of them have the same or similar trappings of their genre as a game without PCG. For example, *Rogue* [17] would undoubtedly be a different game without the use of PCG, but the core difference between *Rogue* and a version *Rogue* without PCG would be that the game could not entirely surprise the player more than on the initial playthrough. *Rogue* uses similar mechanics and aesthetics as other dungeon crawling

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

PCGames 2011, June 28, Bordeaux, France.

Copyright 2011 ACM 978-1-4503-0804-5/11/06 ...\$10.00.

games: exploring new spaces, killing enemies, and picking up loot.

However, there are other games, which we call *PCG-based games*, that create a playable experience inherently structured by PCG and are therefore able to expand or subvert the conventions of their genre or even create an entirely new genre of game. We do not imagine that there is a clear dividing line between a PCG-based game and a non-PCG-based game; rather, this is a spectrum along which games can be measured according to three different measures: replayability and adaptability, its relationship to game mechanics and dynamics (as defined by Hunicke, LeBlanc, and Zubek [6]), and the player's control over content.

This paper presents our initial work towards creating a heavily PCG-based experimental game: *Rathenn*, a 2D platformer that provides the player control over the level they play and subverts some of the traditional tropes of a 2D platformer. The concept of space in the game has a dual meaning: it refers to both the physical space that the player is exploring through traditional platforming mechanics, and the generative space of the underlying system that is automatically generating new content in response to player decisions. Differently colored ladders lead to newly generated rhythm groups, and the player can decide to take a different path at any time and see untaken paths laid out before him. The game is currently open-ended.

2. PCG-BASED GAME DESIGN

In this section, we analyze the different ways that procedural content generation is used in games and how that impacts the player's experience. Our analysis often examines particular aspects or sub-systems of a game, and games can be considered PCG-based with regard to one of these measures but not the others.

2.1 Replayability and Adaptability

One of the earliest examples of PCG is in the game *Rogue*, made in 1980. *Rogue* uses PCG to improve the replayability of the game: on each playthrough, the player encounters a new level. A concept related to replayability is *adaptability*: the use of procedural content generation to adjust content in reaction to player actions or skill levels. Adaptability is a way to improve the replayability of a game, and also potentially a way to diversify the player base by encouraging players who have different play styles or skills. Replayability is the primary purpose of PCG in a number of games, such as *Civilization IV* [3] and *Canabalt* [14].

Map generation serves two purposes in *Civilization*: to provide a novel area to explore at the beginning of the game, and to frame resource trading and combat later in the game. But these two purposes are also fulfilled by other aspects of the game design, most notably decisions made by other players in the game (either

human or AI-controlled). Indeed, the *Civilization* games provide a number of hand-authored maps and scenarios for players that are extremely popular and replayed to attempt different strategies. Thus, while an important aspect of the game, we judge that *Civilization* is not a PCG-based game. On the other end of the spectrum, *Canabalt* is a side-scrolling 2D platformer with procedurally generated level segments. The goal of the game is to play for as long as possible before crashing into a building or obstacle. PCG is also used for replayability in this game: it is impossible to memorize the timing of jumps to play the game, and players must instead hone their skills at reacting rapidly to a obstacles. *Canabalt*'s design is inextricably tied to its use of PCG, thus we label it a PCG-based game.

2.2 Game Mechanics and Dynamics

We can draw a distinction between PCG systems that augment traditional mechanics and those which enable new mechanics entirely. Consider the first-person shooter *Borderlands* [5], which uses PCG to automatically generate a staggering number of weapons (approximately 17,750,000 unique combinations [12]) by combining different weapon properties. In this game the procedural weapon generation takes place within the context of familiar first-person shooter mechanics: *Borderlands* is not a PCG-based game in this regard. However, if we more closely examine the impact of PCG on the play experience, we can see that the vast amount of procedural content available creates a new play dynamic where players are constantly adjusting their game-play to take advantage of new weapons in a way that wouldn't be possible without the PCG system. Thus we judge *Borderlands* to be non-PCG-based with regard to its mechanics, but PCG-based with regard to its dynamics.

On the other hand, Jason Rohrer's latest release, *Inside a Star-Filled Sky* [13] is a good example of a game whose mechanics are PCG-based. Players navigate a space in which they can zoom into or out of recursively nested levels, each one generated from a seed passed from an object in a higher or lower level. Play events that take place on one level of play affect levels above and below themselves, which in turn reinforce or reseed the procedurally generated levels that the game is building. It would be impossible to hand-author this game, as the mechanic of the game is integrally tied to the PCG system.

2.3 Player Control over Content

Many games that incorporate PCG tend to have the generator behind the scenes, out of reach of players. *Rogue* and *Civilization* are both examples of this: while the player interacts with the generated content itself, there is no mechanism within the game for interacting with the generator itself. This property is also true of some games with runtime PCG: *Minecraft* [11] players have control over how quickly they can explore the procedurally generated space, but no control over what they find there.

For those games that do provide the player with access to the generator, there are two forms of access offered: indirect and direct. Indirect control is often used in games that adapt in realtime to the player's skill level, as in the *Polymorph* project [8]. A skilled player could choose to take certain actions, such as falling down gaps or being killed by enemies, which would result in the level adapting to become easier, thus reducing the appearance of certain difficult combinations of level components. *Warning Forever* [10] provides another interesting example of

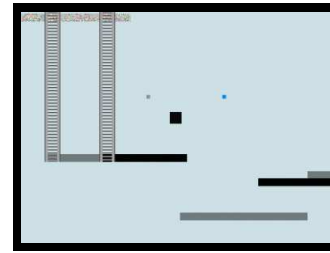


Figure 1. A screenshot of the current version of *Rathenn* during play.

indirect player control: each new boss is generated in a way that adapts to the previous means of destruction. While players cannot directly control the shape of the enemies they fights, they can adopt different strategies that result in drastically different enemy types.

3. RATHENN: AN EXPERIMENT IN PCG-BASED DESIGN

Rathenn is an experimental PCG-based game that expresses the idea of creation through exploration. The primary motivation throughout our design process has been to build a game that highlights procedural content generation and provides a new kind of playable experience. The purpose of the current iteration of the game is to answer questions about the kind of control that the player has over the generator and what further control the player may wish to have, and to determine methods for the player to explore the generative space of the system. We also aim to explore the idea that the player can see the paths he chose not to take and the play opportunities that he could have had by making different decisions. The current version of the game is available to play online.¹

3.1 Game Description

A key aesthetic in *Rathenn* is a sense of exploration, both in terms of physical exploration and exploration of the generative space of the game. The game uses a variant of the Launchpad level generator [15] to create new level segments during play. Parameters for Launchpad control probabilities for the appearance of geometry, the length of level segments and distribution of player actions, and physics properties for the player. As players move laterally through a level, they must overcome a series of challenges in a procedurally generated level segment. At the end of each segment, the player encounters a set of ladders that she can choose to climb. These ladders cause a new segment to be generated, with updated properties from the prior segment as determined by the color of the ladder. When the player reaches the end of a ladder, a new segment is available for further exploration. Figure 1 shows a screenshot of *Rathenn* during play.

There are six colors of ladder corresponding to three sets of parameters that can be altered. The two colors belonging to each set are complementary colors, denoting opposite changes. The color of the ladders at the end of each segment is determined by how many coins of each color the player has collected. For example, if the player has more red coins than green coins, a red ladder will appear instead of a green ladder. There is always one

¹ <http://users.soe.ucsc.edu/~gsmith/rathenn/prototypes/pcg11/>

ladder per decision type available to the player. Activating a ladder causes a shift in the color of the background towards the chosen color. Players are thus visually cued as to the impact of their decisions on system parameters.

- **Red** More enemies, fewer gaps
- **Green** More gaps, fewer enemies
- **Orange** More wait actions, fewer jump actions
- **Blue** More jump actions, fewer wait actions
- **Purple** Avatar can run faster and jump higher; springs are more likely to appear
- **Yellow** Avatar can run slower and not jump as high; springs are less likely to appear

To help give the player a sense of the generative space they are exploring, it is possible for them to backtrack during the game and choose a different ladder at each decision point. The paths that they could have taken appear as a faded image displayed in the background of the player's current field of play.

3.2 A PCG-Based Game

In Section 2 we discussed three ways in which a game can be PCG-based. *Rathenn* was designed as a PCG-based game according to all of these measures.

It is a PCG-based game with regard to replayability and adaptability for two reasons. Firstly, through its mechanics for creating content according to player decisions and providing support for the player to change that decision and take a different path. Secondly, the same decisions on different playthroughs will still lead to different levels, as the players' decisions alter probabilities for generated level segments rather than the level segments themselves.

Mechanics-wise, *Rathenn* is consciously positioned within the tradition of the platformer. The basic actions available to the player are those typical of existing platformers, including jumping over gaps, killing enemies, avoiding stompers, and climbing ladders. Ladder climbing serves a dual purpose in the game, however; it allows the player to explore both the physical level and the space of potential levels that can be created by the generator. This PCG-based mechanic leads to player strategies, or dynamics, that are shaped by the design of the PCG system. By choosing different ladders at particular times, the player can carefully shape the qualities of the level.

Like *Inside a Star Filled Sky*, *Rathenn* builds generated spaces in response to player actions, but in this case players have a measure of direct control over the spaces that the game builds. The game's generator can be actively manipulated by players as they explore the playable space of the game and the generative space of the system. *Rathenn* is an experiment in opening up the generator in its entirety to players, allowing direct control over it as a function of gameplay.

3.3 Future Directions

We consider the current prototype of the game to be fairly successful. There is a need for further playtesting in subsequent iterations to tune the amount by which parameters are adjusted for the different ladders, and also to determine an appropriate interface for viewing greyed out paths. However, *Rathenn* is still in its early stages, and there are a number of other, more major directions we would like to explore.

Better incorporating coin generation and collection into gameplay. Currently, the amount of coins of a certain color that have been collected determine the colors of the ladders at the end of each segment. We would like to investigate providing even more direct control over generated levels to the player, and coins seem a good mechanism for this. We intend to look at how coin collection could be used to electively change certain segments of a level (e.g. turning a jump over a gap into a stomper).

Separating physics from the generation process. Level segments are generated according to the known physics properties of the world at the time of generation: this ensures that all level segments are guaranteed to be playable. We plan to investigate ways to encourage the player to explore both the physical and generative spaces more fully, and one way to do this would be to separate the physics properties of the world from the level generator. If the player explores and discovers a segment of the level that cannot be traversed due to an inadequate jump height, they could then be forced to explore the space to find a level element that changes the world physics before being able to move further in that direction.

Exposing the system's internal state. While *Rathenn* provides the player with direct control over the input to the underlying PCG system, there is no way for the player to understand the generator's internal state. Certain sets of parameters are harder for the generator to meet than others. For example, a request for more enemies to jump on combined with a request for more wait actions is somewhat contradictory: enemies require jump actions, and an increase in wait actions leads to a decrease in jump actions. We are considering exposing this state through either visual feedback in the background, or by explicitly representing the generator as a character in the game.

4. EXPLORING MEANING IN RATHENN

As the use of PCG now enables a player to drive her own experiences in a vast multiplicity of playable levels, we must begin to reconsider how meaning might emerge from these new design affordances. What makes each level significant – or at least significant enough to motivate continued engagement? *Rathenn*'s play is open-ended and player-driven, leaving open two crucial ways of structuring meaning in traditional platformers: an explicit narrative (in favor of play as creation/exploration), and an explicit goal (in favor of play as process/performance). In prying these open, we propose that a game shifts from being a space of contest and narrative enactment [7] to a site of exploration and becoming [4]. The player is positioned in not one, but now two interrelated dimensions: the levels where play unfolds, and the generative space that gives form to the levels. Expanding our thinking from a sculpting of space to include a sculpting of relations and negotiable platforms can potentially scaffold new ways of considering the question of meaning.

Media theorist Henry Jenkins describes game design as the sculpting of space [7]. This is helpful in considering traditional platformers, almost all of which plot game play against a series of designer-created spatial challenges. These challenges are designed to lead the player through a specific skill progression or narrative arc. However, in *Rathenn*, these challenges are constructed procedurally, and open-ended exploration reduces the ability for a game designer to direct a player through such progressions or arcs. We begin by reconsidering a level in *Rathenn* as a "site" that opens up many potential spaces, rather than a single,

prescribed space. Cultural theorist Miwon Kwon's important articulation of site-specificity [9] is a helpful framework for this analysis: "site" is defined as a contingent set of complex and largely asymmetrical relations. In *Rathenn*'s case, we can point out four interdependent relations: the player, the designer, the affordances of the underlying generative system, and ideologies within which player, designer, and system are embedded. Meaning in the game therefore is always shifting and negotiable. We are experimenting with exploring meaning in *Rathenn* in two ways.

***Rathenn* redefines the traditional meaning of success and failure.** As in traditional platformers, dexterity-based challenges motivate continued exploration and mark deepening engagement in the game. However, in *Rathenn*, getting caught under a stomper or not completing a jump does not entirely disrupt the play experience, but instead provides an opportunity for the player to constructively reshape the game world. While the player needs to be able to pass a challenge to keep exploring, she can also choose different paths for exploration. Challenges motivate further exploration of the site, rather than attributing value to particular actions or skill levels. This is a new affordance that procedurally generated game play opens up, and one that still requires much research. In allowing for such an immense field of play, it is important to consider how player choices actually build up to a larger, player-defined schema: a source of meaning. Without this larger sense, players are reduced to arbitrarily moving from Kwon's "one place to another", or passing one challenge for another.

***Rathenn* holds the possibility for player-directed content and meaning.** Procedurality and support for player choices provides an opportunity to disrupt the ideologies embedded in traditional platformers. As digital artist and game theorist Mary Flanagan suggests [4], critical play is a technology that allows players to imagine and inhabit particular kinds of worlds and apply their own interpretations. Foucault might easily unpack the rule systems of traditional platformers (quests, coins, enemies, friends) as a disciplinary apparatus for a capitalist labor force that functions on the accumulation of wealth. With the new rule systems of a PCG-based platformer such as *Rathenn* comes the potential for a designer to express new meanings and seed different interpretations. But whereas traditional games with hand-authored spaces and stories can be strictly designed to send a specific message, games such as *Rathenn*, where the player's choices create new spaces and stories, provide the player with a potentially powerful toolbox for creating their own instances of the game with their own intended meanings. A great deal of further study is needed to determine how best to provide these tools, how players might interact with them, and how expressive this toolbox can be. The task of designing PCG-based games that provide such possibilities to the player has only just begun.

5. CONCLUSION

Rathenn is a work-in-progress 2D platformer 2D platformer that aims to subvert the traditional meaning of platformers through its use of procedural content generation. In this way, *Rathenn* is what we call a PCG-based game: it is heavily dependent on its use of PCG for providing the desired player experience.

By creating this game, we hope to learn more about the generative space of the *Launchpad* procedural level generator and how we can improve its design for use in a game. We also plan to further

explore how the design of the level generator affords different game mechanics and aesthetics. Our experience in creating *Rathenn* has shown that PCG-based game design, much like game design in general, is a highly iterative process with the game pushing constraints on the generative system and vice versa.

It is our hope that the work presented here, and our discussion of how PCG is currently used in games, will promote the further study of PCG as a means to create new playable experiences.

6. ACKNOWLEDGMENTS

This work is supported by the National Science Foundation, grant no. 1002852. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. Our thanks also to Professor Noah Wardrip-Fruin and his Winter 2011 Playable Media class at UC Santa Cruz for their valuable feedback on an earlier prototype of *Rathenn*.

7. REFERENCES

- [1] Booth, M. 2009. The AI Systems of Left 4 Dead. *Keynote, Fifth Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE09)*, Palo Alto, CA, 2009.
- [2] Braben, D. and Bell, I. 1984. *Elite* (Amstrad CPC). Acornsoft.
- [3] Firaxis Games. 2005. Sid Meier's Civilization IV (PC Game).
- [4] Flanagan, M. 2009. *Critical Play: Radical Game Design*. MIT Press: Cambridge, MA.
- [5] Gearbox Software. 2009 *Borderlands* (PC Game), 2K Games.
- [6] Hunnicke, Robin, LeBlanc, Marc, and Zubek, Robert. 2004. MDA: A Formal Approach to Game Design and Research. In *Proceedings of the AAAI Challenges in Game AI Workshop*. San Jose, CA. July 25-26, 2004.
- [7] Jenkins, H. 2004. Game Design as Narrative Architecture. In *First Person: New Media as Story, Performance, and Game*, Harrington, P. and Wardrip-Fruin, N. eds. MIT Press, Cambridge, MA: 2004, pp. 118-130.
- [8] Jennings-Teats, M., Smith, G., and Wardrip-Fruin, N. 2010. Polymorph: A Model for Dynamic Level Generation. In *Proceedings of the Sixth Artificial Intelligence in Interactive Digital Entertainment Conference (AIIDE10)*, Palo Alto, CA, 2010.
- [9] Kwon, M. 2002. *One Place After Another: Site-Specific Art and Locational Identity*. MIT Press: Cambridge, MA.
- [10] Ohkubo, H. T. 2005. *Warning Forever* (PC Game). Hikware.
- [11] Persson, M. 2010. *Minecraft* (PC Game). Mojang.
- [12] Robinson, A. 2009. Gearbox Interview: Randy Pitchford on *Borderlands*' 17 Million Guns. *Computer and Video Games*, July 28, 2009. Available: <http://www.computerandvideogames.com/220328/interviews/gearbox-interview/>
- [13] Rohrer, J. 2011. *Inside a Star-Filled Sky* (PC Game).
- [14] Saltsman, A. 2009. *Canabalt* (Online Game). Available: <http://www.adamatomic.com/canabalt/>
- [15] Smith, G., Whitehead, J., Mateas, M., Treanor, M., March, J., and Cha, M. 2011. Launchpad: A Rhythm-Based Level Generator for 2-D Platformers". *IEEE Transactions on Computational Intelligence and AI in Games* 3, 1 (Mar. 2011), pp1-16.
- [16] Togelius, J., De Nardi, R., and Lucas, S. M. 2007. Towards automatic personalised content creation for racing games. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games 2007 (CIG07)*, Honolulu, HI, 2007, pp. 252-259.
- [17] Toy, M., Wichman, G., Arnold, K., and Lane, J. 1980. *Rogue* (PC Game)