Towards An Ancestry and Codification of Procedural Content Generation Techniques

Todd Furmanski
USC School of Cinematic Arts, Media Arts + Practice
3470 McClintock Avenue, SCI 101
Los Angeles, California 90089
tfurmanski@gmail.com

ABSTRACT

Procedural Content Generation (PCG) used to be a method to circumvent technical limitations; at the present time PCG serves to automate and leverage potential surpluses of computing power. In this paper, I describe and recreate relevant antecedents to methods of PCG in game design, in order to better understand their use in the past and present. My research methods straddle both critical reading and coding useable artifacts. I acknowledge non-digital forms of PCG going back centuries, while asserting that procedural techniques require a closer understanding and intimate connection between theory and practice, criticism and practicality.

Categories and Subject Descriptors

B.6.1 [**Logic Design**]: Design Styles—*Cellular arrays and automata*; I.6.3 [**Simulation and Modeling**]: Applications; K.2.0 [**Computing Milieux**]: History of Computing—*Systems*; K.8.0 [**Personal Computing**]: General—*Games*

General Terms

Games

Keywords

Game Design; Simulation; Procedurally Content Generation.

1. INTRODUCTION

The term "Procedural Content Generation" can be applied to any number of aspects of game development, including animation, texturing, prose generation, and level design. PCG has come gradually into mainstream attention, with examples such as Will Wright's *Spore* [8] and the current trend of "Roguelike" dungeon games [13], but has roots in digital games going back decades, and instead of being classified as either a dead-end or solve-all, should be classified as its own creative set of disciplines [4, 5]. The process merges coding and art, even when its final products might be several steps removed from explicit authorial decisions. PCG aides small teams of developers by automation, and large teams by managing and extending complicated dynamics that cannot be feasibly created any other way. The potential of procedural content can move from the useful to the sublime: A game player can walk through landscapes the artist has never seen, hear music the composer never heard, and interact with dynamics the programmer never imagined. Fascination with automating a writer or artist's output predates modern computing theory, and modern forms of game production benefit from knowing how this intersection of art and logic developed.

Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG 2015), June 22-25, 2015, Pacific Grove, CA, USA.
ISBN 978-0-9913982-4-9.

2. RESEARCH GOALS

The multidisciplinary nature of PCG means that the conventional divide between critical theory and design practice particularly frustrates the goal of academic understanding. Confining PCG to the domain of programming ignores aesthetic and design values that define its purpose. However, trying to understand PCG while ignoring elements such as source code and computing logic can only result in an equally incomplete picture of how and why it shapes game development. Developers should not have to keep re-inventing the wheel, nor theorists conjecture on processes based on simple output. I mean to develop a more informed curriculum incorporating history, and a taxonomy organized by process. The programmatic nature of PCG invites an approach of playful experimentation – I hope not only to learn more about PCG but to directly participate in its future evolution.

3. THEORIES AND PRECEDENTS

Academic interest in PCG in recent years has produced a growing examination of its use in games. Mark Hendrikx, Gillian Smith, and Julian Togelius write about distinctions between online and offline processes [19], the ability to validate generated content, as well as how PCG can fit into many different places within the heirarchy of a game structure [7]. The role of human interaction and authorship can be placed upon a spectrum [17], and this variable amount of human agency provides a crucial dimension to understanding PCG along with the algorithms and grammars that usually come into discussions on automated content.

The terms "random," "chaos," "procedure," "simulation," and "emergence" come up in studying PCG, but often exist in opposition to each other. Procedural content can be static, defying iterative simulation. While randomness can serve as a "seed" for a procedural technique, the Oulipo group eschewed randomness while embracing procedure obsessively [11]. Artificial Life and Cellular Automata have a specific spectrum of system definitions [9], from "chaotic" to "complex" – terminology that can be very useful in communicating qualities of process in game design.

Noah Wardrip-Fruin summarizes the fundamental idea of the rhetorical capabilities of procedure and simulation when he describes what he calls the "SimCity Effect," [20] where iteraction with a dynamic system, such as a game, shapes a user's internal model of how the depicted system works. Games teach systems, and games relying extensively on PCG give an oppurtunity to compare such systems explictly with how a player might interact or perceive them.

3.1 Pre-Digital Procedural Content

My research looks at historical precedents going back to pre-digital times, with particular emphasis on the Oulipo Literature group [11], Dada [6], the works of John Cage [3, 16] and even earlier ruminations from the satirical computer in Swift's *Gulliver's Travels* to Ramon Llull's theological machine, and how Borges considered such an approach more suited to poetry than philosophy [2]. Authors such as Claire Bishop [1] and Mary Flanagan [6] provide further context on how history and cultures have shaped participatory art and play, including procedural art. Traversing through each of these examples reveals a fascinating evolution of the perceived relationship among the author, reader, and creative process as it relates to logic, reason, art, society, and agency.

3.2 PCG in the Digital Era

With the digital era, starting in the 1970's, I look at the emergence of home computing, the first widespread computer games, and other entities such as the Demoscene subculture [15], whose procedural animations paralelled the development of this era in intriguing ways. "Platform Theory," which describes the relationship between a hardware structure and how it influences the software designed for it, aides in understanding the forces at work in this transitional era. Nick Montfort and Ian Bogost use Platform Theory to provide a structure of New Media and games in particular [10], proposing a hierarchy that describes the literal medium between the user and the device. Starting with the user or viewer, they describe Reception/Operation leading to Interface, then to Form and Function, deeper into Code and finally to Platform, the metaphorical "water" that the "fishes" of code might swim in but not necessarily notice. Analyzing PCG dynamics in this manner proves useful in comprehending the evolution of games in changing technical paradigms.

4. METHODS OF RESEARCH

The Media Arts and Practice program uses a hybrid approach to understanding digital media, encouraging extensive research into theory and criticism while also appreciating the benefits of producing media in order to better understand the relevant forces and dynamics at work.

4.1 Code Analysis

While I examine specific games known for their extensive use of PCG, wherever possible I also highlight crucial portions of source code that form the core of how the games generate their experience. Many of the sources will be from the earlier days of personal computing in the 1970's and early 1980's, due to their relative simplicity and accessibility, but also due to the foundational works found in this timeframe. These formative examples generally fall under the term "constructive," [18] relying on a bounded scope and predictable output. The example in listing 1 comes from the early role-playing game *Telengard*. Numerical constants shift values defined by the player's location in X,Y, and Z space to produce a pseudorandom enviorment that is nevertheless persistent. Using a coordinate as a parameter has some resemblance to Ken Perlin's process of creating graphical "noise." [14]

```
XO = 1.6915
YO = 1.4278
ZO = 1.2462
W0 = 4694

10010 Q = X * XO + Y * YO + Z * ZO
        + (X + XO) * (Y + YO) * (Z + ZO)
10020 H% = (Q - INT (Q)) * W0:
        IF H% / 256 > 5 THEN
        H% = H% - INT (H% / 256) * 256
10025 IF INT (H% / 256) > 0 THEN
        H% = (INT ((Q * 10 - INT (Q * 10))
        * 15 + 1) * 256) + H%
        - INT (H% / 256) * 256
```

Listing 1: Telengard's Coordinate-Based Dungeon Generation

Small pieces of such code provide examples useful for critical understanding as well as informing practice and production of games. Separated from the original game, this code can drive graphical visualizations, data heuristics, or be further modified and expanded upon.

4.2 Practical Demonstrations

My degree combines theory with a practice element, and I have created a series of works demonstrating the topics and ideas explored. I not only examine but replicate the processes studied, running or emulating their original hardware platforms where possible. After reproducing and verfying that a given procedure functions properly, I make adjustments, experimenting with the code, with the purpose of visualizing the consequential parameters and variables involved. Where would a system break down? How does a given hardware platform change otherwise identical instructions? I have built several interactive works, from simple but demonstratice "sketches" to fully playable games, and outline two examples in section 5 below.

5. CURRENT WORK AND FUTURE PLANS

I intend my research into the historical aspects of PCG to inform my current practical work, and have my authored works demonstrate the dynamics and potential of the techniques studied.

5.1 Historical Case Studies

I will organize my compiled research into the development of PCG cronologically, divided into specific examples of games and the techniques they use, delving into excerpts of relevant code. I will include interviews with the coders, artists, or communities involved where possible. Specific cases include foundational games such as *Elite* and *Rogue*, as well as an extenstive examination of *Noctis*, a space travel game started in 1996 by Alessandro Ghignola [12]. In *Noctis* a player can travel to and land on billions of unique planets - anticipating works such as *Spore* and *No Man's Sky*, all in an era where one had to program their own software renderer and store their work in less than a megabyte of memory. A devoted community has spent a great deal of time documenting (and re-coding) the game, and their experiences prove every bit as fascinating as the work itself.

5.2 Authored Works

I have several works, both completed and in progress, showing my own experiments with simulation and PCG, and the practice component of this dissertation will involve duplicating and expanding on core procedural techniques to better understand the line. One project is *Wu Xing*, a virtual space simulating the interaction of five elements, based on a classical Chinese cosmic model and reaction-diffusion cellular automata. The resulting landscapes produce floods, tectonic upheaval, droughts, and other analogous natural processes. Another project, *Forska*, shown in figure 1, procedurally generates a landscape in a 3D volume that a user navigates frame by frame, each one rendered in an impressionistic style. Using random seeds, an explorer can theoretically travel to billions of worlds, with interacting forces defining the landscape, architecture, weather, and agent behavior. I mean to use these works as further methods of communicating the qualities and potential of PCG.

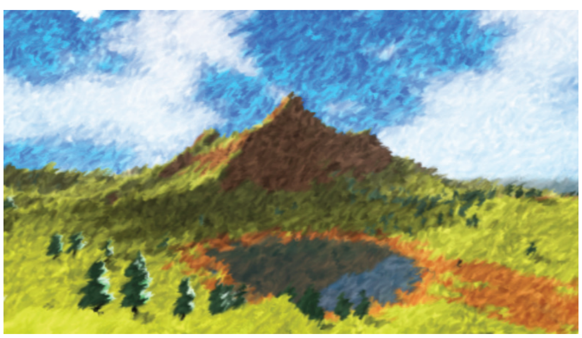

Figure 1: Forska, a procedurally generated navigable impressionist landscape.

6. CONCLUSION

A proper understanding of PCG requires a thorough understanding of programming, art, theory, and design. In combining an exploration of early non-digital forms of PCG with an analysis of early digital works, and of course current explorations, I seek to develop a more informed curriculum and theory of procedural process and algorithmic tools. In developing and experimenting with such methods myself, I aim to demonstrate a working knowledge with the goal of illustrating the scope and potential of this multidisciplinary space.

7. REFERENCES

[1] C. Bishop. *Artificial Hells: Participatory Art and the Politics of Spectatorship*. Verso Books, 2012.

[2] J. L. Borges and E. Weinberger. *Selected Non-Fictions*. Penguin Books, New York, 2000. 155–159.

[3] J. Cage. *Silence: Lectures and Writings*. Calder and Boyars, 1968.

[4] D. Cook. "Content Is Bad." Lost Garden. `http://www.lostgarden.com/2007/02/content-is-bad.html`.

[5] A. Doull. "Death of the level designer." Procedural Content Generation Wiki. `http://pcg.wikidot.com/the-death-of-the-level-designer`.

[6] M. Flanagan. *Critical Play: Radical Game Design*. MIT Press, 2009.

[7] M. Hendrikx, S. Meijer, J. V. der Velden, and A. Iosup. Procedural Content Generation for Games: A Survey. ACM Transactions on Multimedia Computing, Communications and Applications. *ACM Transactions on Multimedia Computing, Communications and Applications.*, 2011.

[8] D. Hopkins. The Future of Content – Will Wright's Spore Demo at GDC 3/11/2005. `http://www.donhopkins.com/drupal/node/35`.

[9] Langton, Christopher G. (ed.). *Artificial life II: Proceedings of the Workshop on Artificial Life: Held February 1990 in Santa Fe, New Mexico*. Number 10 in Santa Fe Institute studies in the sciences of complexity proceedings. Proceedings volume. Addison-Wesley, 1992.

[10] N. Montfort. *Racing the beam: the Atari Video computer system*. Platform studies. MIT Press, 2009.

[11] W. F. Motte, editor. *Oulipo: a Primer of Potential Literature*. French literature series. Dalkey Archive Press, 1st dalkey archive ed edition, 1998.

[12] This is the Webterritory of Fottifoh. `http://anynowhere.com/`.

[13] C. Nutt. "'Roguelikes': Getting to the heart of the it-genre" Gamasutra. `http://www.gamasutra.com/view/feature/218178/roguelikes_getting_to_the_heart_.php`.

[14] K. Perlin. "Making Noise." Noise Machine. `http://www.noisemachine.com/talk1/`.

[15] T. Polgár. *Freax: The Brief History of the Demoscene. Volume 1 Volume 1*. CSW Verlag, 2005.

[16] K. Silverman. *Begin Again: a Biography of John Cage*. Alfred A. Knopf, 1st ed edition, 2010.

[17] G. Smith. Understanding Procedural Content Generation: A Design-Centric Analysis of the Role of PCG in Games. *Proceedings of the 2014 ACM Conference on Computer-Human Interaction*, 2014.

[18] J. Togelius, T. Justinussen, and A. Hartzen. Compositional procedural content generation. *Proceedings of FDG Workshop on Procedrual Content Generation*, 2012.

[19] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne. Search-Based Procedural Content Generation: A Taxonomy and Survey. *Computational Intelligence and AI in Games, IEEE Transactions*, 3(3):172–186, 2011.

[20] N. Wardrip-Fruin. *Expressive Processing: Digital Fictions, Computer Games, and Software Studies*. MIT Press, 2012.