Procedural Content Generation for Computer Games

A survey of techniques used for procedural content generation for computer games, classified by beneficiary.

Thomas Smith
Electronics and Computer Science
University of Southampton
taes1g09@ecs.soton.ac.uk

ABSTRACT

Modern computer games make use of a wide variety of procedural content generation (PCG) techniques that serve a number of purposes during the development of a game. Though PCG techniques have been used since some of the earliest computer games[1], they are now becoming increasing relevant as AAA games become larger and more detailed and indie games teams become smaller. PCG techniques can easily amplify production efforts by automating asset generation or augmenting manual content production, and they can be a powerful tool to allow developers to create and populate varied and believable playspaces. They can also play a role in customising each player's experience to improve engagement and entertainment. This paper provides a high-level classification of a range of techniques used for different types of content, and attempts to discern common aspects and transferable approaches that can help to promote a more unified, standard approach to content generation for games.

1. INTRODUCTION

Intoduction to the topic, explanation of lack of structure. Reference to the age of things, Elite[1], nethack, increasing use in successful commercial games. The need for procedural content generation.

1.1 Background

Building on the work of Hendrikx et. al [6] this paper provides a modern overview of the range of procedural content generation techniques used in games today.

1.2 Methodology

Given the broad range of procedural content techniques, this review organises approaches first by the primary user of each method. Lists a number of the main purposes that PCG is used for within that category, along with examples and academic literature in that area.

2. ARTISTS

When producing the raw content that goes into a game, procedural generation techniques can provide a more efficient method or greater variety than building everything by hand. Many of these techniques are not unique to the field of computer games and are also used to produce computer-generated graphics for all kinds of media, from animated movies to photorealistic bcakgrounds in advertisements and print media.

2.0.1 Textures

Perlin Noise Grammars Long history of using perlin noise effects for material textures. Originally developed by Ken Perlin in <year>, it provides <definition>. Used for everything from clouds to marble. Pattern-based textures. Ideal for providing high-resolution texturing across large landscapes. Algorithmically generates textures with specified features, useful for either repetetive applications such as hich-rise city blocks or aperiodic such as natural landscapes [8]

2.0.2 Models

visual variety, procedural construction fill space - speedtree[4], procedural cities Borderland's guns Grammars Speedtree

2.0.3 Animation

Rather than create animations for all possibilites (counterexample - assassins creed 3 had <number> of distinct animations) Respond to conditions that weren't known at design time: user content generation - Spore[**?**], allows charaters to react to a vast range of physical conditions - Jedi Unleashed force push, Emotion engine.

2.0.4 Effects

Procedural generation of environmental effects such as fire, water, smoke and clouds. Provides believable variety (starter point for citations in [6]) Particles Generation of effects representing spells and weapons allows customisation based upon the properties of the ability [5] Procedural rendering effects - allows graphical styles that are radically distinct from traditional photorealistic techniques [7]

2.0.5 Music

Allows great variety - specify a style, and get infinite variations. [3] <Does this belong in designers? often reacts to in-games event, used to build mood>

2.1 Benefits

Two of the main benefits associated with artists' use of PCG techniques are the reduction in labor required to make a large variety of variations on a them, and the reduced storage requirement to represent this variets. This is becoming increasingly relevant as games become larger and the status quo shifts from physical to digital distribution - smaller games means faster delivery Download sizes - procedural variation - Borderland's enemies

2.1.1 Future work

3. DESIGNERS

3.1 Varieties

3.1.1 Content scale

Use of PCG allows designers to populate large game spaces with a high level of detail and variety. Middleware packages such as Speedtree[4] allow entire forests to be generated and customised, and similar approaches exist for generating believable cities and landscapes.

3.1.2 Replayability

The use of <semi-random> techniques means that playspaces genearted using PCG may be different for each player, and for each player's playthrough of a game. The variety ensures that content is fresh each time, and minimises the effects of repetition - players are not able to memorise the precise route through each dungeon and locations of treasures, and so there is a sense of diecovery and exploration each time.

3.1.3 Challenge

On of the main factors that affects players' enjoment of games is their ability to remain in a state of 'flow' - the sensation that their abilities are matched to the challenge provided by the game[2]. In order to cater to the wide range in ability of players that each game may attract, many games have previously offered the option of customising the challenge experienced to one of a number of pre-set 'difficulty levels'. <Read through 3yp for citations on difficulties with self-assessment and mutability>. Rather than <set> the degree of challenge at design time, Lopes et al. show that it is possible to adaptively generate or alter aspects of the game[10] in order to match the observed ability of the player[9].

3.2 Benefits

Speed up work - large, beliveable diverse playspaces
close the gap between designer and player - allow modification based on individual information about the player.

3.2.1 Future work

Improve automated critics [6]

4. USERS

4.0.2 Experience

Valve's AI Director Bethesda's Radiant Storytelling

4.0.3 Agency

Typically, no direct player control over adaptive generation (cite hamlet) however, in some games that make use of procedural generation it can be a benefit to give the player some degree of direct control over the generation process - for example a recent addition to the GAR was the ''Weapons Lab', a portion of the game where players may spend in-game resources to customise their procedurally-generated weapons. (cite GAR's weapons lab) Radiant Story? GAR's weapons

4.1 Benefits

Users get a more engaging experience that can be tailored specifically to their preferences and abilities

4.1.1 Future work

5. CONCLUSION

We can see that a variety of <stakeholders> benefit from the improvement of procedural content techniques, and that there are a wide range of existing techniques used for a plethora of different reasons.

6. REFERENCES

[1] D. Braben and I. Bell. *Elite*. (Armstrad CPC), Acornsoft, 1984.

[2] J. Chen. Flow in games (and everything else). *Communications of the ACM*, 50(4):31–34, April 2007.

[3] K. Collins. An introduction to procedural music in video games. *Contemporary Music Review*, 28(1):5–15, 2009.

[4] A. de la Re, F. Abad, E. Camahort, and M. Juan. Tools for procedural generation of plants in virtual scenes. In *Computational Science – ICCS 2009*, pages 801–810. Springer Berlin / Heidelberg, 2009.

[5] E. Hastings, R. Guha, and K. Stanley. Interactive evolution of particle systems for computer graphics and animation. *IEEE Transactions on Evolutionary Computation*, 13(2):418–432, 2009.

[6] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup. Procedural content generation for games: a survey. 2012.

[7] M. A. Kowalski, L. Markosian, J. Northrup, L. Bourdev, R. Barzel, L. S. Holden, and J. F. Hughes. Art-based rendering of fur, grass, and trees. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 433–438. ACM Press/Addison-Wesley Publishing Co., 1999.

[8] S. Lefebvre and F. Neyret. Pattern based procedural textures. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, I3D '03, pages 203–212, New York, NY, USA, 2003. ACM.

[9] C. Liu, P. Agrawal, N. Sarkar, and S. Chen. Dynamic difficulty adjustment in computer games through real-time anxiety-based affective feedback. *Intl. Journal of Human–Computer Interaction*, 25(6):506–529, 2009.

[10] R. Lopes and R. Bidarra. Adaptivity challenges in games and simulations: a survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(2):85–99, June 2011.

Procedural Content Generation for Computer Games

Thomas Smith (taes1g09@ecs.soton.ac.uk)
Supervisor: Rikki Prince (rfp@ecs.soton.ac.uk)

Problem: Procedural Content Generation (PCG) is the name applied to the growing field of techniques that attempt to provide content for modern games through the use of algorithmic generators rather than manual construction. As AAA games get larger and indie studios get smaller, all varieties of computer game developer have an incentive to automate the production of material of all kinds for their games. A large number of specialised techniques exist, each tailored to generating a particular kind of content, from rulesets to textures to enemy or object behaviour. Increasingly, concepts from the field of AI are being used to improve the quality and relevance of the generated content, whether that be through intelligently breeding possibilities from a population of solutions or applying machine learning to create a map between player behaviour and optimal content. Each genre of game and each kind of generated content has a range of potential techniques that may be useful for its production, and typically the PCG techniques used in games are individual bespoke applications of these various methods as each game requires different content and there are not yet standard approaches to integrating multiple kinds of content generation.

Goals: Many existing papers on PCG deal with a single aspect or particular technique. A smaller number consider a range of previous techniques for the generation of a particular range of game content, such as terrain or cities, and draw comparisons on their relative merits. The goal of this project will be to attempt a high-level classification of a large range of techniques used for many different types of content, and attempt to discern common aspects that can help to promote a more unified, standard approach to content generation for games. The final output will consist of a detailed report, an A1 poster and accompanying presentation, and hopefully a short paper for submission to the Fourth Workshop on Procedural Content Generation in Games at FDG2013.

Scope: In order to attempt to ensure that the project goals remain achievable, the scope should be restricted to only recognised and established PCG techniques with obvious application to games. It may be useful to consider related areas for comparison, such as player-driven content generation or procedural PCG generators, but the focus should remain limited to applicable and useful methods in order to facilitate classification and comparison.