Procedural Content Generation for Computer Games

A survey of techniques used for procedural content generation for computergames, 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. Improve variety, allow developers to populate large areas, allow customisation to player preferences, reduce assets, allow control/tweaking, reduce labor costs Automating content generation, or augmenting manual content production

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.

2. BACKGROUND

Building on the work of Hendrikx et. al [4] this paper provides a modern overview of the range of procedural content generation techniques used in games today.

2.1 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.

3. 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.

3.0.1 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 [6]

3.0.2 Models

visual variety, procedural construction fill space - speedtree[3], procedural cities Borderland's guns Grammars Speedtree

3.0.3 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.

3.0.4 Effects

Procedural generation of environmental effects such as fire, water, smoke and clouds. Provides believable variety (starter point for citations in [4]) Particles GAR? Procedural rendering effects - allows graphical styles that are radically distinct from traditional photorealistic techniques [5]

3.0.5 Music

Allows great variety - specify a style, and get infinite variations. [2] <Does this belong in designers? often reacts to in-games event, used to build mood>

3.1 Benefits

Download sizes - procedural variation - Borderland's enemies

3.1.1 Future work

4. DESIGNERS

4.1 Varieties

4.1.1 Content scale

Speedtree

4.1.2 Replayability

4.1.3 Challenge

4.2 Benefits

4.2.1 Future work

5. USERS

5.0.2 Experience

Valve's AI Director Bethesda's Radiant Storytelling

5.0.3 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 lab

5.1 Benefits

5.1.1 Future work

player control over procedural content generation

6. 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.

7. REFERENCES

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

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

[3] 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.

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

[5] 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.

[6] 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.

APPENDIX

A. PROJECT BRIEF