Fachhochschule Luzern HSLU

Studiengang Digital Ideation, Bachelor
6. Semester

# TBD

Simon Hischier

March 9, 2019

# Inhalt

**1 Abstract** **2**

**2 Introduction** **2**

**3 Problem/Research Question** **2**

**4 Literature Review** **3**
4.1 Categories of Procedural Generation . . . . . 3
4.2 Rise of Scripting languages . . . . . 3

**5 Methodology and Results** **4**
5.1 Textures . . . . . 4
5.2 Tools for Procedural Content Generation . . . . . 5

**6 Conclusion and Future Work** **5**

**7 References and acronyms** **5**

# 1 Abstract

This work discusses the relation between the past lack of procedural content generation in the game industry with the recent uptake of procedural content generation in emerging tools for game developers and artists and the use of procedural content generation in the game industry. How is procedural content generation as part of games and as a developing tool changing so that it is gaining relevance in the games industry and related sectors? New tools and a more data-driven approach to procedural content generation in recent years has lead to a less noticeable but steady increase in usage across a number of disciplines and workfields and a shift in toolsets in the game industry. The paper observes and analyzes the trends and tries to explain the newfound interest through analogy, literature review, market and workflow analysis. We want to highlight what these new tools and approaches provide and what previous tools were missing and the conclusion why procedural content generation gains popularity again based on these changes.

# 2 Introduction

The cost to make games seems to be on an exponential curve as Raph Koster states[1]. He suggests among other things that game developers should focus a lot more on algorithmic and procedural approaches to keep the cost of games under control. On Quora, an online discussion forum, Steve Theodore gives an example on the time needed for assets in games. A model for the game Half-Life 1 took about 2 working weeks and a model 10 years later took him seven working weeks. Theodore states, that most extra time is spent on texturing, shading and animation[2]. Those are fields where Procedural Content Generation (PCG) has seen huge improvements.

Studios developing AAA games spend a lot more money on creating resources. Samuel Rantaeskola[2] states that in the past systems were less powerful and therefore the limiting factor was optimization. Nowadays the content and increasing complexity of content is the big limitation. More people working on a project increases overhead and the more work on a project the less impact there is when a single person is added to a team.

Ever-increasing cost of development does come with side effects. A lot of big studios closed down or had massive layoffs over the past year [3, 4, 5, 6] and this seems to be a long term trend backed by older numbers[7]. The race to higher costs leave midsize studios in an awkward position. They are too big to survive by creating small games but not big enough to compete with AAA games. Steve Theodore recalls seeing a lot of midsize studios closing down in the seattle area because they grew into big studios, went indie or died[2].

# 3 Problem/Research Question

todo

# 4 Literature Review

## 4.1 Categories of Procedural Generation

We are using PCG categories based on the book "Procedural Generation in Game Design"[8, p. 3] where the following four categories are described.

Integral The use of PCG is part of the game design from the start. This games rely heavily on a working PCG and even core gameplay can be affected. Games such as *Rogue* (A.I. Design, 1980) or a more modern game like *Dwarf Fortress* (Dwarf Fortress, 2006) are using PCG extensively and would not work without it. They need it to be the type of game they are. Changes to the project planning have vast implications on the codebase. These games are build around central algorithms and project changes will result in redrafting the algorithms.

Drafting Content From a game design perspective this games do not rely on PCG from the start. Game designers rely on PCG to generate initial drafts of game content such as the map or items. These drafts can be looked through by humans and are then handpicked. Some games use this method to generate a world which is then polished by humans. An example of generating and polishing is *Skyrim* (Bethesda Game Studios, 2011) and a more recent and sophisticated example is *Far Cry 5* (Ubisoft, 2018) where Carrier Étienne explains how the world was modified by humans but regenerated daily by his team[9].

Modal Some games are build with little or even without the use for PCG and it gets added later on during development. Even after release PCG can be added in the form of an "infinity mode" or as procedural maps in *Rust* (Facepunch Studios Ltd, 2013). While this type can add a lot or replay value to a game it is mostly used just for that and does not add innovative content to the game.

Segmented A game build with segmented content including closed off areas where PCG is used. The development can continue with or without these areas if the desired standard is not met by the algorithm. The term "areas" doesn't need to be restricted to levels. It can include parts of the game such as procedural music, graphical effects or randomized elements. The game developers have at all times the possibility to revert to hand-generated content.

## 4.2 Rise of Scripting languages

John K. Ousterhout wrote a paper[10] in 1998 where he analyzed the difference in use of system languages (C, C++, Java) and scripting languages (Python, Perl, Javascript). His analysis is important because in this paper we look at visual node systems (a part of the dataflow programming paradigm[11]) as comparable to scripting languages in the sense that they are more easily understandable. He pointed out, that scripting languages are much better suited for casual programmers based on several reasons.One reason is the size of programs. Scripting languages need less lines of code to get things done. Higher-level languages need several months of learning to master where as scripting languages lead to results in hours. This is of great importance, it allows a team to hire technical artists who are quickly ready to work on projects. However both languages are required, they are complementary.

> "Scripting languages assume that a collection of useful components already exists in other languages."[10, p. 2]

The limiting factor in larger adaptations are the power of computers. The paper finds that, the faster computers get the larger the applications built with scripting languages will be. Current developments on various applications indicate that computers are powerful enough. Workflows are shifting towards programs with scripting languages or visual node systems.

# 5 Methodology and Results

## 5.1 Textures

Textures is traditionally a big topic in game development. A lot of research went into textures because it's a big part of modern rendering pipelines and reaches back to the early days of modern computer graphics. Games in the 80's and 90's had hard limits on disk space[12] and designing games involved balancing storage space between program code, music and images. As an example on the limitation: Besides a single digital sample channel sounds on home consoles like the Nintendo Entertainment System (NES) were limited to hardware generated tones[13]. Consoles like the NES and Super Nintendo Entertainment System (SNES) had no chips to calculate complex 3D objects. The Picture Processing unit was used to display only sprites. To fill a TV screen the NES and SNES used texture tiling, a technique for creating small texture samples which can be tiled to create a seamless pattern. This technique is still in use for modern games with increased more complex textures and increased texture size. Games using PCG as an integral part rely heavily on tileable textures or alternatively use generated procedural textures for example Perlin[14] or Worley[15] noise. Those approaches come with disadvantages: Tiles are very repetitive and Perlin noise need very sophisticated algorithms to generate interesting levels. Another texture generating solution is using by-example noise algorithms. These algorithms take a stochastic example texture and generate larger versions out of it. Until now these were too slow for real time generation and therefore were only suitable for drafting content. A recent paper by Heitz and Neyret created an algorithm able to create stochastic textures on-the-fly[16]. This enables artists to build levels and getting direct feedback with final textures already applied during building. This technique even allows for algorithmic level generation while not using tile based textures. This is especially important because textures not based on tiles lead to hard edges and repeating patterns shown in Figure 1.

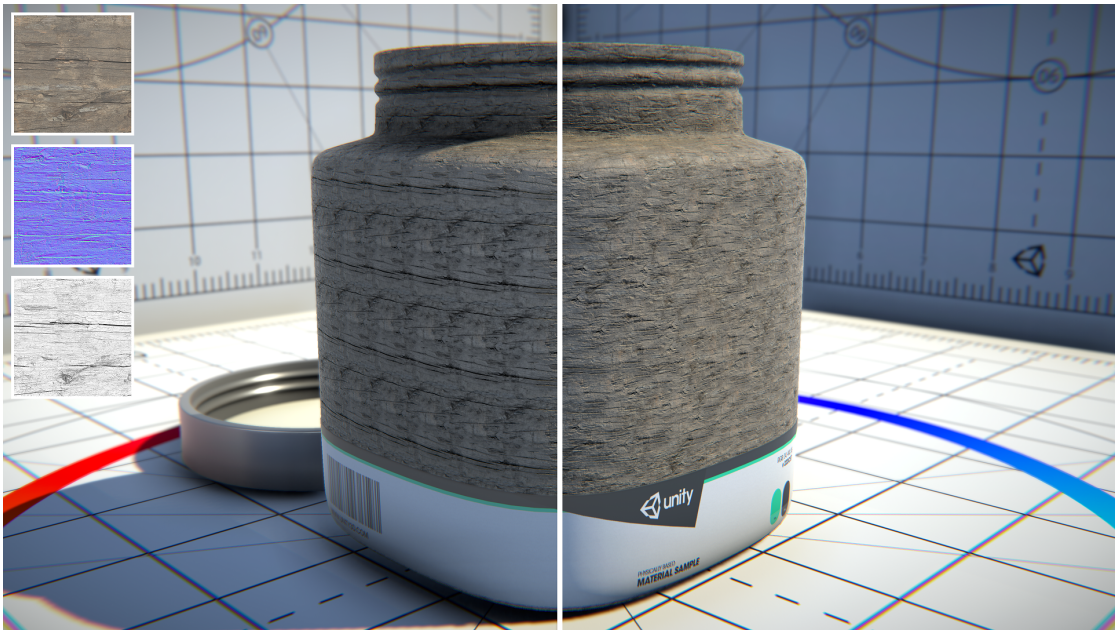


Figure 1: Wood detail maps with albedo, normal and occlusion textures. Retrieved 11:50, March 9, 2019 from https://blogs.unity3d.com/2019/02/14/procedural-stochastic-texturing-in-unity/

### 5.2 Tools for Procedural Content Generation

At the Blender Conference Andrew Price tried to predict changes to the 3D industry in the next 5 to 10 years. A big point besides machine learning assistance becoming a standard is the adaption of procedural workflows becoming mainstream[17]. Price states that procedural texturing made by Substance Designer in combination with Substance Painter is saving big studios money in the range of hundreds of thousands of US Dollars.

## 6 Conclusion and Future Work

todo

## 7 References and acronyms

## Acronyms

**NES** Nintendo Entertainment System. 4

**PCG** Procedural Content Generation. 2–4

**SNES** Super Nintendo Entertainment System. 4

# References


[1] Raph Koster. Gamasutra: Raph Koster's Blog - The cost of games. https://www.gamasutra.com/blogs/RaphKoster/20180117/313211/The_cost_of_-games.php, 2018.

[2] Mike Prinke, Eric Backeberg, Samuel Rantaeskola, Steve Theodore, Matti Porkka, Eric Chung, Bill Hitchens, and Rowley Gregory. Why have video game budgets skyrocketed in recent years? - Quora. https://www.quora.com/Why-have-video-game-budgets-skyrocketed-in-recent-years, 2017.

[3] Evan Lahti. Here are all the major game studios that have closed in the past year — PC Gamer. https://www.pcgamer.com/here-are-all-the-major-game-studios-that-have-closed-in-the-past-year/, 2018.

[4] Jason Schreier. Activision-Blizzard Employees Brace For Massive Layoffs. https://kotaku.com/activision-blizzard-employees-brace-for-massive-layoffs-1832488999, 2019.

[5] Alex Walker. EA's Australian Studio Hit By Massive Layoffs — Kotaku Australia. https://www.kotaku.com.au/2019/02/eas-australian-studio-hit-by-massive-layoffs/, 2019.

[6] Jason Schreier. Guild Wars 2 Developer ArenaNet Plans For Mass Layoffs — Kotaku Australia. https://www.kotaku.com.au/2019/02/guild-wars-2developer-arenanet-plans-for-mass-layoffs/, 2019.

[7] Luke Plunkett. Every Game Studio That's Closed Down Since 2006. https://kotaku.com/every-game-studio-thats-closed-down-since-2006-5876693, 2012.

[8] Tanya X Short and Tarn Adams. *Procedural Generation in Game Design*. A. K. Peters, Ltd., Natick, MA, USA, 1st edition, 2017.

[9] Étienne Carrier. Procedural World Generation of Ubisoft's Far Cry 5 — Etienne Carrier — Houdini HIVE Utrecht - YouTube. https://www.youtube.com/watch?v=NfizT369g60, 2018.

[10] J.K. Ousterhout. Scripting: higher level programming for the 21st Century. *Computer*, 31(3):23–30, mar 1998.

[11] Wikipedia contributors. "Dataflow programming" Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Dataflow_programming&oldid=884393969, 2019.

[12] Mark Ferrari. 8 Bit & '8 Bitish' Graphics-Outside the Box - YouTube. https://www.youtube.com/watch?v=aMcJ1Jvtef0, 2016.

[13] Wikipedia contributors. "Nintendo Entertainment System" Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Nintendo_Entertainment_System&oldid=885685790, 2019.

[14] Ken Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3):287–296, 1985.

[15] Steven Worley. A cellular texture basis function. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 291–294. ACM, 1996.

[16] Eric Heitz and Fabrice Neyret. High-Performance By-Example Noise using a Histogram-Preserving Blending Operator. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(2):1–25, aug 2018.

[17] Andrew Price. The Next Leap: How A.I. will change the 3D industry - Andrew Price - YouTube. https://www.youtube.com/watch?v=FlgLxSLsYWQ, 2018.

# List of Figures

1 Wood detail maps with albedo, normal and occlusion textures. Retrieved 11:50, March 9, 2019 from https://blogs.unity3d.com/2019/02/14/procedural-stochastic-texturing-in-unity/ . . . . . 5

# List of Tables