Final Project

Design Document

Procreate Design Document

Author: Jared Schroeder CIT025989

Due: 12 Nov. 2014

Abstract

Glossary of Terms

Acronym	Meaning
PG	Procedural Generation
PLG	Procedural Level Generation

Table of Contents

1. Introduction

This design document is for Procreate - a ‘procedural generation-based level creator’ application being developed by Jared Schroeder for the Final Project course in the Bachelor of Games and Virtual Worlds. The document first describes the differences between traditional and procedural level creation, and notes the history, strengths and weaknesses of each method. It then introduces Procreate, and proposes its great worth for 2D game developers creating modern game levels.

Following this, the functionality and features of the tool are detailed, which are then further embellished with application conceptual layout images and written walkthroughs.

The aim of this document is to introduce Procreate to the reader, help them understand the benefits that make it worth developing, and then give them a basic knowledge of what they can do with it, what it looks like, and how they can use it.

1.1. Traditional and Procedural Level Creation

A game level is traditionally created by one or more digital artists, with the following steps:

1. Review the gameplay that the level will present to the player (the level's requirements)
2. Research and decide what the level will be like (what it looks like, what it contains, etc.)
3. Plan and design the actual level
4. Create the level with level editing software

This process achieves a level that is fully man-made – every part of it has been crafted by the artists. Levels created in this way contain a quality that is based on the amount of creativity, effort and volume of content that has gone into them. However, these three factors can take time to achieve, and some game developers may not have this time to produce high quality levels.

The time can be reduced by hiring more artists, however doing so might increase the cost of development, which might be unaffordable to developers with less money.

A game may have no digital artists working on it. In this case, non-artists have the difficult job of making game levels with a similar quality to games created with artists. Artists have more experience in their line of work, meaning that the levels they create will be of higher quality than non-artists.

Procedural level generation is one solution to these problems. Instead of using human artists to create game levels, it uses processes completed by a computer. To do it, one can follow these steps:

1. Research and decide what the level will be like
2. Research, brainstorm and decide which procedural generation algorithms (processes) to use that are suitable for the level
3. Code and test the algorithms, to make sure they're working correctly
4. Tweak the algorithms to produce the desired level
5. Repeat step 4 (with or without tweaking) to produce as many levels as desired

Procedural level generation solves many of the traditional game level creation problems. Firstly, once it is implemented, it can produce a great variety and volume of levels in a short time period, that if done with artists, would require more time and money to produce. It also needs to be noted that a group of artists can only create a finite set of concrete levels (levels that don't change in the game) whereas procedural level generation is capable of making countless levels that could provide the player with a new level and therefore a new experience every time they play.

Secondly, procedural level generation can be achieved by developers with minimal experience with art or art creation programs, because they can provide the computer with a small amount of digital art and it generates the level for them.

Thirdly, while game level data created by artists need to be stored with the game to be reproduced, procedurally-generated levels can be created when the game starts, and deleted when the game quits. This can cause the game's software files to be smaller in size, and thus be quicker to download, transfer, install and uninstall, because the level files do not need to persist.

So to recap, procedural generation has the capacity to reduce the cost and time required to create levels, it can increase game replayability by producing different levels each time the game is played, increases the throughput of game developers with little artistic experience, and can reduce the size of a video game's software files. These benefits have pushed procedural game level generation into the spotlight throughout the previous decade.

1.2. Procedural Level Generation in Games

A variety of modern games have used procedural level generation, proving that it is currently popular. A few examples are provided below to describe the method used, the results that it produced, and why it was used.

FIGURE 1: MINECRAFT (CLIFFE 2011) USES PG TO CREATE LARGE AND DIVERSE WORLDS TO BE EXPLORED (2011)

DON'T STARVE (DIRTYTABS 2014) USES NARRATIVE-BASED PG TO GENERATE VARIED PLAYER TASK-BASED LEVELS (2013)

THE BINDING OF ISAAC GENERATES A LARGE NUMBER OF VARIED LEVELS COMPRISED OF CONNECTED ROOMS (2011)

'BROGUE' CREATES CAVES TO BE EXPLORED AND GENERATES ITEMS TO BE COLLECTED BY THE PLAYER, WITH PG TECHNIQUES (2009)

1.3. Procedural Level Generation Problems

While procedural level generation provides the aforementioned benefits, it has some faults. These are detailed throughout this section.

The main problem with PLG is that it the levels it produces can eventually become meaningless to a player playing in them. This is because each level is very different, and doesn't have any meaningful elements that would usually be introduced by human artists creating the level. As such, a player can become used to the diversity in each level, and have no special experience with any of the levels.

Secondly, programmers implementing PLG in their game have to take the time to research the algorithms to use, code them, and then test them to make sure they're working . This takes time - a simple algorithm can be coded and tested in a day, however algorithms with greater complexity could take a few weeks to implement. Developers with less coding experience, for example artists, will require even more time to add a PLG technique to their game.

Thirdly, a game's PLG may require other development stages to have been completed. For example a game’s rendering system may have to be set up before a level generated with PG can be reviewed and evaluated. This means that PG levels cannot always be designed upfront.

Finally, if a game's procedurally-generated levels and level generation are embedded in the game code, they could be difficult to extract and reuse for other games. This is even more difficult if the game reusing old PLG code is written in a different computer language to the original game, because the code has to be perfectly translated from one language to another.

The Procreate application is the suggested solution to each of these problems.

2. Procreate

5. Bibliography

Cliffe 2011, *Minecraft 2.jpg*, viewed 6 June 2014, digital image, http://minecraft.gamepedia.com/File:Minecraft_2.jpg?version=810335ef249b76b83963c03f824a4d38

Dirtytabs 2014, *Don't Starve Gallery Treeguard.png*, viewed 6 June 2014, digital image, http://dont-starve-game.wikia.com/wiki/File:Don%27t_Starve_Gallery_Treeguard.png