Fortress generation for Dwarf Fortress
Procedural Content Generation in Games Autumn 2014

Jakob Melnyk, jmel@itu.dk
Jacob Claudius Grooss, jcgr@itu.dk

December 12th, 2014

Introduction

This paper was written as part of a project for the "Procedural Content Generation in Games" course on the Games Technology track at the IT-University of Copenhagen.

For the project we could do one of two things: Develop a new PCG algorithm, or apply an existing algorithm (or combination of algorithms) to some game domain in a novel way.

Problem Statement

For our project, we decided to generate fortresses for the game Dwarf Fortress[1] by creating random map layouts and and then evolve the contents of the maps.

Our goal was to be able to generate fortresses that could be used in-game in practice, even though they may not be perfect fortresses[2]. We decided to not aim for perfect fortresses, as Dwarf Fortress is an amazing complex game and there are an immense number of variables to take into account in order to create a perfect fortress.

Background

Dwarf Fortress is a game about leading an expedition of dwarves in order to create a new home for them. The game takes place on a procedurally generated map with multiple layers, where the player can direct dwarves to perform various tasks (mining, gathering plants, building furniture, crafting weapons, and so forth). The goal of the game is to build a huge fortress for the dwarves and to keep them alive for as long as possible.

Dwarf Fortress is often described as having a very steep learning curve, as it tells the player nothing about how they play or what they are supposed to do. All of it is something the player has to figure out by themself. This often means that a player's first fortresses will be of very low quality, as they discover more and more things they have to add to it that they did not plan for.

In order to help the novice player out, we intend to create a fortress generator that can give them some basic layouts from which they can choose which one they like the most. These layouts should also give the player an idea of how many different things they need to play for in any future fortresses they may want to make.

Game Design

What's the design of the game you are going to use? Why do you need PCG in this game?

Design: Tile-based map where you are the leader of a dwarven expedition. You tell the dwarves what to do and attempt to build a huge fortress in which they can live and thrive.

PCG: Because coming up with a fortress layout can be very difficult, especially for new players. Having an idea of how to create a fortress would be an immense help.

---

[1] http://www.bay12games.com/dwarves/
[2] "Perfect" in this context meaning a fortress that encapsulates everything a fortress can have, with a layout that optimizes everything.

Figure 1: A screenshot of a very basic fortress.

Ikke sikker på dette kapitel er nødvendigt. Er ikke i eksempel rapporten vi har fået. Muligvis copy/pasta fra Modern AI, men står inde på PCG siden på Learnit

Methods

There are various different ways to generate dungeons[1, Chapter 3][2], but we did not feel that they did what we wanted. We wanted all of the rooms to be connected to the entrance as directly as possible, while still being able to connect to each other on the way.

For this purpose, we wrote our own algorithm, which works as follows:

1. Create a map of the user-specified dimensions and calculate the number of rooms we want, based on how many dwarves the user expects to have in their fortress.

2. Create an entrance to the fortress on the top layer along one side of the room.

3. For each layer in the map (starting at the top one), make a list that contains all positions that are not dug out, then do the following:

   (a) Choose a random position from the list of open positions and use it as one corner of the room.

   (b) Pick a random, diagonal direction (north/east, south/east, south/west or north/west), go 6 tiles that way and see if the positions in that direction are open.

   (c) If they are open, build a room in the middle 4x4 square, set the tiles around that square to room walls, and remove the positions now occupied by the room from the list of open positions.

   (d) Create a path from the room to the nearest entrance (stairs on the lower levels count as entrances to that level).

      (e) Once enough rooms have been created, or only 10% of the layer is open, create stairs to the next layer, and connect these stairs to the entrance of the current layer.

      (f) Repeat for each layer until enough rooms have been built.

Results

Did it work? How well? Provide some figures, and a table or two. How much time does it take? Remember to include significance values (remember the t-test?), variance bars... Reread some of the papers from class and compare how they report their results.

PCG Book 2.4: Evaluation functions

PCG Book 12: Evaluating Generators

Discussion

What are the strengths and shortcomings of your method? Why did you choose method X instead of Y? How well would it generalize to other game genres? How would you develop it further, if you had time?

References

[1] A lot. *Procedural Content Generation in Games*. Work in progress, 2014.

[2] Jeff Orkin. Three states and a plan: The a.i. of f.e.a.r. *Monolith Productions / M.I.T. Media Lab, Cognitive Machines Group*, 2006. URL `http://alumni.media.mit.edu/~jorkin/gdc2006_orkin_jeff_fear.pdf`. [Online; accessed December 7, 2014].