

A Comparison of Procedural Level Generation Techniques for 2D Platform Games

COMP110 - Computer Architecture Essay

1507290

November 26, 2015

bah

1 Introduction

Procedural content generation for 2D platform games is a relatively new field in comparison to games of other genres[1]. Many diverse approaches to this problem have been developed, each having advantages and disadvantages. [?]. There are chunk-based approaches which rely on hand-designed chunks of level; rhythm-based models that generate a level based on sequences of player actions; and evolutionary approaches that use genetic algorithms. It could be argued that the most desirable level generator would be the one that requires the least human input yet offers high amounts of control and is able to generate a wide variety of fun and playable levels consistently.

2 Comparison

2.1 Overview

Smith et al.[2] base their level generation tool, Launchpad, on the concept of rhythm. They define rhythm as being the ideal sequence and timing of button presses that the player must execute; for example, a series of jumps. They use a two-tiered approach to ensure that rhythm is always present regardless of the level geometry. Rhythm groups are generated first, before being translated into geometry using design grammars. This results in levels being both playable and fun.

Sorenson et al.[3] are also inspired by rhythm with their technique, but instead define rhythm more loosely as alternating periods of high and low difficulty. They use a genetic algorithm to generate levels, with their model of fun as a fitness function. Levels that have a high fitness value are able to pass their genetic information on to future generations, resulting in levels gradually being improved. Although other approaches using evolutionary algorithms have been developed[4], Sorenson et al.'s offers more direct control. They designed the algorithm to be general and able to apply to any genre of game.

Mawhorter and Mateas[5] take a different approach to level generation with Occupancy-Regulated Extension (ORE). Instead of levels being generated from individual components, ORE uses pre-authored chunks of level and stitches them together based on anchor points that represent places the player can occupy. The system was designed to prioritise variability, with no strict measures in place to ensure playability, unlike the other two algorithms.

2.2 Human Input and Control

Launchpad is designed to require minimal effort whilst offering high levels of control. It allows the rhythm, the path that the level should follow, and the desired number of each component to be specified. A number of levels are generated and assessed before presenting a pool of the best matches to the designer. In practice, this means that Launchpad may be better for aiding the design of levels, rather than generating them in-game. Even if the closest matching level could automatically be chosen, generating a large pool of levels will make the process take longer with no visible results.

Sorenson et al.'s approach also allows the number of each design element to be restricted. Additionally, the designer can adjust the difficulty of the generated levels by changing one parameter. If desired, part of the level can be hand-designed and the rest of the level generated around it. This means that level profiles for each part of the game could be specified, allowing control over progression whilst maintaining variation.

Of the three methods, ORE requires the most human effort. Level chunks must be hand-authored, meaning that the output is greatly influenced by the quality of the chunks. Additionally, in order to achieve the desired result, an understanding of how the algorithm works is required. One advantage of hand designing chunks, however, is that chunks can be of any nature, allowing more complex structures to be generated.

2.3 Flexibility and Variability

Smith et al. claim that Launchpad is able to produce a wide variety of levels due to the rhythm generation being separated from geometry generation. That is, levels with the same rhythm can have many different variations. Certain patterns and tendencies can be observed in the levels generated by

Launchpad, however. Unfortunately, the emphasis on rhythm means that the generated levels are only appropriate for dexterity based platform games.

This is also an issue with Sorenson et al.'s algorithm, as the model of fun used in the fitness function is based around challenge and difficulty with respect to skill. On the other hand, there is the scope to adjust the fitness function. If required, a fitness function based on a different model of fun could be developed, to allow levels suitable for other types of platform games to be generated.

ORE could potentially generate levels appropriate for other types, such as exploration based platform games. This may be possible as chunks can be designed with any desired result in mind, whilst the algorithm does not need adjusting since the playability of levels is not assessed.

2.4 Playability and Design

Consequently, this means that ORE may not always produce playable levels (that is, levels able to be completed). This greatly reduces the possible applications of ORE, as it would be risky to use this generator to randomly generate levels in-game in case the player is presented with an unbeatable level. One option is to use an additional algorithm to determine whether the level is playable before presenting it to the player, but that would use additional resources and extend the time taken.

The other two approaches have comprehensive algorithms in place to guarantee that all generated levels are both playable and enjoyable. Launchpad has a physics engine that knows how fast the player moves and how high they can jump, which is used when generating the level geometry to ensure that the resulting level is possible to complete. Sorenson et al. specify the requirements for playability in the form of constraints, meaning that unplayable

levels are repaired using constraint satisfaction before being returned to the population of acceptable levels.

In addition to having measures in place to ensure playability, Sorenson et al. and Smith et al. both ensure that the levels are fun. Both of their definitions of fun are based on the concept of 'flow' as described by Csikszentmihalyi [?]. The fitness function used in Sorenson et al.'s genetic algorithm is based on their comprehensive model of fun[6], ensuring that the population evolves towards levels with a higher fitness value. Smith et al.'s definition of fun is defined in the algorithm that generates the rhythm groups. Although 'fun' is subjective, these two algorithms ensure that at least some form of theoretical 'fun' will be present in the game.

ORE is not based on any model of fun. The enjoyability of the resulting level depends both on how well the chunks have been designed and the implementation of the chunk selection algorithm. Again, this puts a lot of emphasis on the designer. With chunks designed by the authors of the algorithm, ORE came fourth in the *2010 Mario AI Championship* [7], whilst Sorenson and Pasquier managed to place third, despite the approach being generalised.

3 Conclusion

Whilst ORE provides great control over the design of a generated level, Launchpad and Sorenson et al.'s approach provide greater flexibility and reliability with little effort.

References

- [1] K. Compton and M. Mateas, “Procedural level design for platform games,” pp. 109–111, June 2006.
- [2] G. Smith, J. Whitehead, M. Mateas, M. Treanor, J. March, and M. Cha, “Launchpad: A rhythm-based level generator for 2-d platformers,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, pp. 1–16, March 2011.
- [3] N. Sorenson, P. Pasquier, and S. DiPaola, “A generic approach to challenge modeling for the procedural creation of video game levels,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, pp. 229–244, Sept 2011.
- [4] F. Mourato, M. P. dos Santos, and F. Birra, “Automatic level generation for platform videogames using genetic algorithms,” in *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology*, pp. 8:1–8:8, 2011.
- [5] P. Mawhorter and M. Mateas, “Procedural level generation using occupancy-regulated extension,” in *IEEE Symposium on Computational Intelligence and Games (CIG)*, pp. 351–358, Aug 2010.
- [6] N. Sorenson and P. Pasquier, “The evolution of fun: Automatic level design through challenge modeling,” in *Proceedings of the First International Conference on Computational Creativity (ICCCX)*, pp. 258–267, 2010.
- [7] N. Shaker, J. Togelius, G. Yannakakis, B. Weber, T. Shimizu, T. Hashiyama, N. Sorenson, P. Pasquier, P. Mawhorter, G. Takahashi, G. Smith, and R. Baumgarten, “The 2010 mario ai championship: Level

generation track,” *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, pp. 332–347, Dec 2011.