Adventures in Hyrule: Generating Missions & Maps For Action Adventure Games

[Extended Abstract]

Becky Lavender
Department of Computing & Mathematics
University of Derby
Derby, UK
becky@beckylavender.co.uk

Tommy Thompson
Department of Computing & Mathematics
University of Derby
Derby, UK
tommy@t2thompson.com

ABSTRACT

This paper discusses ongoing research into the construction of action adventure maps akin to those found within classic Nintendo series *The Legend of Zelda*. Following existing work in the composition of missions and spaces for the action-adventure genre, this work adopts these methods into a open-source 2D tile-based game with similar mechanics and tropes to that of the Zelda series.

1. INTRODUCTION

Procedural content generation (PCG) algorithms are popular in video game development given the opportunities present in crafting artefacts for player consumption. However, it is crucial that assurances can be given for consistency and validity of content. This paper concerns itself with the generation of Action-Adventure (AA) 'dungeons', which require not only valid topology and structure, but the accurate construction of intricate key and lock puzzles puzzles for players to solve. The focus on these puzzles is what differentiates AA maps from those found in roguelikes, where experience and loot dictate progression and map layout is not integral.

Previous work by Dormans found in [3] denotes AA dungeons as comprised of two distinct structures: mission structure and the physical space of the dungeon, with their relationship largely responsible for what differentiates them from other genres. Dormans argues that these structures should be developed independently; with a unique grammar for each component. Despite this assertion, this technique has yet to be applied to a typical 2D tiled AA game. As such, some typical AA mechanics have not been applied with the system. Moreover, obstacles in shape grammar generation are not described to an implementation-ready level of detail in existing work.

In this paper, we highlight ongoing work aimed at adopting this two-tier approach to overcome problems inherent in AA dungeon generation. This is achieved by creating a mission and map generator, and applying the result to *Mystery of Solarus* [7]: an open-source game heavily inspired by *Legend of Zelda: A Link to the Past* [5].

Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG 2015), June 22-25, 2015, Pacific Grove, CA, USA.
ISBN 978-0-9913982-4-9.

Alphabet:

bl = boss (level)	G = Gate	l = lock
bm = boss (mini)	g = goal	lf = lock (final)
C = Chain	H = Hook	lm = lock (multi)
CF = Chain (Final)	ib = item (bonus)	n = nothing/exploration
CL = Chain (Linear)	iq = item (quest)	S = Start
CP = Chain (Parallel)	k = key	t = test
e = entrance	kf = key (final)	ti = test (item)
F = Fork	km = key (multi piece)	ts = test (secret)

Figure 1: The alphabet for mission grammar to represent an action-adventure dungeon akin to *Legend of Zelda*, detailed in [3].

2. BACKGROUND

Dormans' method involves first creating a mission grammar which consists of an alphabet shown in Figure 1. This grammar is comprised of non-terminal, high level symbols ('Parallel Chain', 'Fork') and terminal low level symbols ('key', 'lock', 'test'), with rules for rewriting non-terminal symbols with terminal ones. Some nodes are connected by 'tight connections', which dictate that nodes must be placed consecutively in a map. For instance, a treasure chest containing a reward should be placed directly after a mini boss fight. These rules are recursively applied to an initial graph during the generation process, until a completed mission graph such as the one in Figure 2 is achieved.

Mission graphs are then taken as input into the space generation system. Shape rules, which each correspond to a symbol in the alphabet, dictate which room should be placed for each node in the mission graph. For the purposes of this work, a grid-based map is produced in which each room contains a mission symbol (Figure 3). Dormans' sentiments on the benefits of quest and map generation were echoed in [8], which cites the coupling of generators as a step towards multilevel PCG. The addition of context to space generators has also been suggested as means to avoid generated content which is perceived as bland and random [2]. Similar examples of context being added to Action-Adventure spaces can be seen in projects such as *GameForge* [4] and *Charbitat* [2].

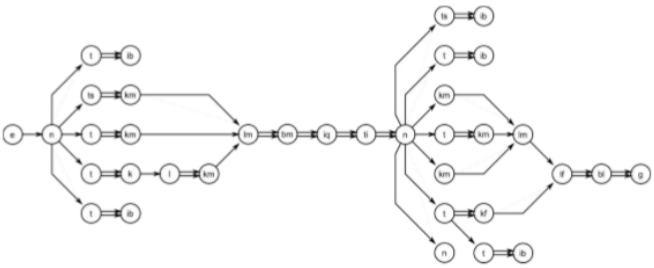

Figure 2: An example mission. Each node has a symbol which represents a room type from Figure 1. Taken from [3].

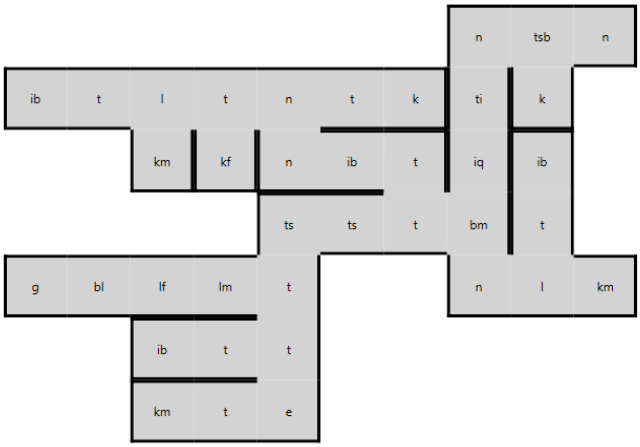

Figure 3: An example map in which each room represents a node from the mission graph.

3. OVERVIEW OF DEVELOPMENT

Our mission generator takes an alphabet, a set of rules and an initial graph as input, iterating through and rewriting sections of the graph until a completed mission graph is produced, which contains only terminal symbols. Our map generator subsequently iterates through nodes in the mission graph and places a corresponding room from an existing door. Once the mission has been completely accounted for in the space, all extraneous doors are removed. A number of challenges were faced in development, some of which had been alluded to in [3] but not described in detail. This includes the conservation of mission structure by defining a complex graph traversal order for room placement and the prevention of paths of rooms being placed in dead ends. The latter is an issue that would prevent tightly connected rooms being placed from their corresponding parents. In addition, nodes in long tightly connected chains require room positions to be reserved, albeit without placing them prematurely and thus corrupting mission structure (e.g. placing keys behind the doors they correspond to).

The maps produced were applied to open source game *Mystery of Solarus*, with rooms created by hand using the Solarus map editor [1]. The game produced shares the mechanics and aesthetic of *The Legend of Zelda: A Link To The Past* [5], released on the Super Nintendo in 1991. However, one constraint is that dungeon maps are made up solely of fixed-size rectangular rooms, whose placement dictate the dungeon layout. Traditional 'Zelda' mechanics such as monsters, traps, secrets, dungeon items, mini-bosses, multi-part keys and locks are all exhibited. Despite this, the implementation imposes some limitations: it was necessary to design some rooms symmetrically, given our inability to pre-empt which direction the player would come from or be headed to.

In keeping with existing research in procedural content generation systems [6], it is important that the expressive range and variability of our generators are evaluated. Both the mission and space generators can be evaluated against core metrics mission linearity, map linearity, leniency and path redundancy. This analysis is complete at the time of writing, but merits discussion that is beyond the remit and length of this paper. It is our intention to discuss this in a future publication.

4. SUMMARY

Our work demonstrates reproduction of the mission grammar generator in [3] for Zelda-style games. Existing issues with the space generation section of [3] are acknowledged, with this work providing solutions to the problems encountered. It is our intention to document these solutions fully in future. We also successfully add context to maps and consequently the dungeons they produce. By producing a method through which to solve the issues involved in generating AA content, we provide evidence that procedural action adventure games have potential and hopefully pave the way for more research in the area. Ultimately, adding context to PCG enables us to move away from bland, random content which is novel solely because it is generated, and towards purposeful content which is novel in and of itself.

5. REFERENCES

[1] Solarus: An ARPG game engine [sic]. `http://www.solarus-games.org/`. [Online; accessed 11-April-2015].

[2] C. Ashmore and M. Nitsche. The quest in a generated world. In *Proc. 2007 Digital Games Research Assoc.(DiGRA) Conference: Situated Play*, pages 503–509, 2007.

[3] J. Dormans. Adventures in level design: generating missions and spaces for action adventure games. In *Proceedings of the 2010 workshop on procedural content generation in games*, page 1. ACM, 2010.

[4] K. Hartsook, A. Zook, S. Das, and M. O. Riedl. Toward supporting stories with procedurally generated game worlds. In *Computational Intelligence and Games (CIG), 2011 IEEE Conference on*, pages 297–304. IEEE, 2011.

[5] Nintendo. The legend of zelda: A link to the past, 1991.

[6] G. Smith and J. Whitehead. Analyzing the expressive range of a level generator. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, page 4. ACM, 2010.

[7] Solarus. Zelda mystery of solarus dx, 2008.

[8] J. Togelius, A. J. Champandard, P. L. Lanzi, M. Mateas, A. Paiva, M. Preuss, K. O. Stanley, S. M. Lucas, M. Mateas, and M. Preuss. Procedural content generation: Goals, challenges and actionable steps. *Artificial and Computational Intelligence in Games*, 6:61–75, 2013.