Level Design on Rogue-like Games: An Analysis of *Crypt of the Necrodancer* and *Shattered Planet*

Brenner P. de Castro[1]* Rosilane R. da Mota[1]† Eduardo P. C. Fantini[2]‡

[1]Universidade Federal de Minas Gerais, Department of Photography, Theater and Cinema, Brazil
[2]Pontifícia Universidade Católica de Minas Gerais, Department of Computer Science, Brazil

ABSTRACT

Rogue-like games have acquired a special spot in the independent game industry. As a genre that includes procedural content generation, its games provide players with an incredible amount of content and *replayability*, which means each time they play, the game's environment is changed in a considerable way. This paper selected two modern Rogue-like games, Crypt of the Necrodancer and Shattered Planet, and addressed their game design choices, pointing out their advantages and problems. Each aspect was analyzed taking into matter its effect on players. Although most of game design theories were exposed as in any other game, some core perceptions about level design were shown to be different in Rogue-like games. Finally, a thought process for level design in Rogue-likes was proposed, based on the theories discussed and how procedural generation shapes the environment.

Keywords: Procedural Content Generation, Rogue-like, Replayability.

1 INTRODUCTION

Procedural content generation (PCG) concerns the process of generating content from algorithms instead of manually. In other words, rather than hand crafting assets or levels for a game, procedural generation allows designers to input sets of parameters to a computer, e.g. content type, size and variation, and receive a pseudo-random result based on an algorithm. Since the creation of Rogue [22], a role-playing game (RPG) that has procedurally generated dungeons, many developers have designed extremely successful games whose content is mostly created by PCG techniques.

Although such methods appear to have unpredictable results, programmers have shown that, with the right amount of rules, procedural worlds can be as well structured as handcrafted ones. Diablo 2 [7], Minecraft [14] and Crashlands [17] are some examples of the innumerable possibilities procedural content can provide both to creators and players. However, since many concerns about art and level design – previously trusted on professionals – are now in hands of a computer, designing an interesting and engaging procedurally generated game has shown to be a difficult task.

Regarding these matters and based on the main structure of Rogue, a whole genre of games was created. Numerous game developers have studied game design and procedural generation to produce extremely engaging games that are similar to the original Rogue. However, there is few written work discussing level design interests specifically for Rogue-like games.

This paper presents a discussion on procedural generation techniques and level design decisions involving two Rogue-like titles: Crypt of the Necrodancer (CotN) [16] and Shattered Planet (ShaP) [9]. The purpose of this research was to understand which lessons they teach about how game developers should think when designing procedurally generated levels for Rogue-like games.

*bpcastro@ufmg.br
†rosilane@ufmg.br
‡fantini@pucminas.br

2 RELATED WORK

The industry of games has been growing throughout the years and statistics show that the global market (including TV, computer, mobile and handheld games) will be worth around $113 billions in 2018 [23, 15]. As the game industry rises, independent (i.e. indie) and big company games get larger and more complex, requiring the work of a considerable number of professionals to create the desired product [10]. Aware of high production costs, developers have aimed for conserving time and resources while still preserving high quality of games.

With this in mind, researchers have recurred to procedural content generation as a method to generate game worlds more efficiently. Although some artists have shown noticeable resistance against PCG [6] – since it takes control from the creators –, the method has certain advantages, such as: quick generation of content; requiring less time, resources and professionals; and providing variety of results and *replayability* [24]. However, it is important to notice the efficiency of such techniques depends on the amount of control and optimization provided by its algorithm.

Linden et al. [24] listed various methods employed for procedurally generating dungeons. The research was divided by PCG technique types, e.g. cellular automata and generative grammars, and focused on the capability of algorithms to provide control over the final result. Linden also pointed out the main shortcomings for each method, e.g. low control and chaotic levels with cellular automata and complexity in relating result to parameters with genetic algorithms.

Hendrikx et al. [10], on the other hand, classified PCG methods depending on what type of material was produced. The main classifications were: game bits (trees, textures, sound); game spaces (level geometry); game systems (environment and its agents); game scenarios (puzzles, stories, quests); game design (level design); and derived content (content that does not affect gameplay directly).

Adams [1] presented a PCG technique utilizing graph grammars and analyzed the psychology and game design aspects of procedurally generated levels. The article defines two important factors of a level: the topology and the geometry. Topology refers to the sequence of main events happening through the level, while geometry refers to the molded space of the level. Furthermore, Adams suggested the use of main level design notions to evaluate PCG games: interesting level architecture; gameplay pacing; realism and feedback; non-linearity; careful asset revelation; and resource balance.

In terms of level design, Rouse III [11] addressed the most important aspects of design to consider when creating a level, such as pacing, level flow (i.e. critical path) and navigation. While Rouse III [11] covered these aspects in general, Short [18] considered procedural generation as her main factor in level design.

Concerning the definition of the Rogue-like genre, while most designers and players categorize the genre as having some features similar to Rogue, some developers prefer to use the Berlin interpretation [8], which implies the presence of all Rogue features in the game, without exception. In this context, the following factors are

considered to define what a Rogue-like game is, based on most definitions listed by Slash [19]: procedural environment generation; permanent failure, which means players usually lose most or all of their resources after dying; grid-based levels; high variety of items and enemies; limited inventory; and single character instances.

Finally, considering the analysis of Rogue-like games as procedurally generated products, Cruz [4] developed a tool to simulate a player and evaluate the quality of procedurally generated maps. Furthermore, Cruz presented a simple summary of some Rogue-like games and terms used in researches.

Based on all researches listed above, a methodology was chosen in order to effectively analyze Shattered Planet and Crypt of the Necrodancer concerning their level design.

3 Methodology

This paper focused on studying and discussing level design and procedural generation in Rogue-like titles – Shap and CotN, specifically. Considering the presence of PCG in both games, it was important to research and understand which techniques were used to construct their levels. Linden et al. [24], Leite and Lima [12] and Short [18] presented valid techniques for planning and building dungeon-like 2D levels through PCG.

First, the choice of ShaP and CotN was made contemplating the definition of Rogue-like listed in section 2 and examining the differences and similarities between both games. ShaP is a turn-based game where players control a clone made to explore a planet that has exploded. The objective of ShaP is to explore as much unreached territory as possible, while retrieving the DNA of creatures encountered. On the other hand, CotN is a rhythm-based game where players guide an adventurer inside a huge cavern. The objective of CotN is to traverse through dungeons after her lost father and defeat the Necrodancer.

Second, some level design concepts listed by Adams [1] and Rouse III [11] were used to analyze both games in section 4. Level architecture, exploration, lighting and reward systems were the main aspects considered. Each aspect was described in both games and then analyzed, pointing out any level design problems or advantages.

Finally, an overall thought process for projecting level design in rogue-like games was proposed based on all studies done throughout the research.

4 Level Design Aspects of Crypt of the Necrodancer and Shattered Planet

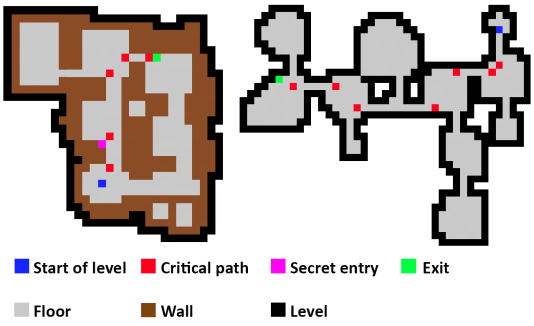

Figure 1: 2D pixel maps of a CotN Zone 2 level (left), adapted from the game, and one ShaP's development sample (right), adapted from [18].

Levels in rogue-like games are usually divided in certain types of structures, both programming and design-wise. In both CotN and ShaP, levels are divided by rooms connected by corridors, as seen in Figure 1. By breaking down levels into smaller parts, designers assume better control over random assignments of procedural generation.

In ShaP, for example, developers control the width, height, number and type of rooms in each level. Given these parameters, levels are composed by balanced numbers of small and large rooms (Figure 1) with regard to "causing different emotional reactions in the player" [18].

In a similar way, CotN also creates zones with distinct-sized rooms. In fact, players are able to change its surroundings by digging walls – brown area in Figure 1 –, either to maneuver around enemies or to create shortcuts, which opens more ways to overcome difficulties.

According to Short [18] and Totten [21], smaller rooms evoke a claustrophobic feeling on players while serving as tactical space due to its size. On the other hand, bigger rooms seem to provide players with more options, while enhancing their chances of being surrounded by enemies.

However, as seen in Figure 2 and 3, such feelings also depends on the player's current situation. In Figure 2, although the character was in a wider space, it was battling a boss while close to walls, which raises tension. On the other hand, in Figure 3, there are no threats around the character and only one enemy in vision. This means even if the character was in a corridor, with not many options of escape, there was no danger present at that moment, thus, no tension.

Another important strategy used to improve level generation is defining a critical path. According to Stout [20], critical path is the quickest path for a player to pass through in order to finish all tasks required to reach a level's exit. This path must be able to inform both computer (during generation) and player what is the right direction towards the exit. In Figure 1, red dots were used to indicate a presumption of each level's critical path, from the start of the level (blue dot) to the exit (green dot).

Figure 2: Gameplay of Crypt of the Necrodancer [25]

A simple way to inform players they are on the right path is to give hints, either using sound, visual or mechanical effects.

CotN does give an important hint to wherever the player must go inside levels, which is the presence of a boss (i.e. strong enemy) near the exit. Since the level's boss produces big sound effects and goes towards players way before they reach the boss room, they get feedback about where the exit is.

On the other hand, ShaP's levels do not hint the player about where the right path is. Although exploring is a key factor for the game, the presence of a critical path with no hints is pointless, because players will never be aware of it until they find the exit. Thus, players may get lost and this situation must be avoided [11].

Besides the size of rooms, both games use lighting as a mechanic to constrain players' vision. As seen in Figures 2 and 3, the light force lowers towards the border of the screen, and in CotN, players' vision can be enhanced with items. Similarly, they also

give hints of what hides in dark spaces by either showing a dark silhouette for items or shiny eyes for enemies. Controlling how light is in certain parts of the environment is important not just to control player's knowledge [13], but also to instigate different feelings, such as fear or peace [21].

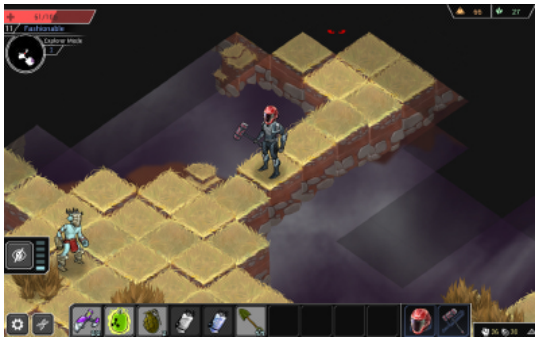

Figure 3: Gameplay of Shattered Planet

Last but not least, in both games it is common to see rewards. Reward is a term used to describe items given to players in exchange for some kind of effort, such as completing a quest, defeating a strong enemy or finding a key for a locked door [11].

In terms of rewards inside a level, one of the main problems of ShaP is not rewarding players properly. For example, in some levels of the game, there are locked doors that lead to secret places. According to Rouse III [11], if players need to do an extra effort to reach some area, it is important to make that place interesting and rewarding. However, in ShaP, sometimes such places are either empty or have only some enemies or common items, nothing that is different from other rooms. CotN's algorithm, though, usually puts either an extra-life potion or lots of enemies in these rooms, both rare to encounter. If ShaP and CotN's reward distributions were compared, it would be easy to see that placing surprising elements in hard-reaching places can make a big difference for players.

As discussed above, the same logic applies to *unlockable* features. This term refers to items, stats, characters, clothing or others that can be unlocked by players through the game. In both ShaP and CotN this system is present and, the same way as rewards, *unlockables* can give players a sense of evolution.

5 Understanding Level Design in Rogue-like games

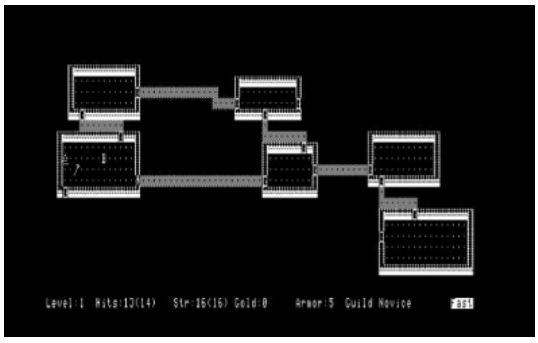

Figure 4: Gameplay of Rogue. It is noticed the similarities of level architecture between Rogue, ShaP and CotN. [5]

In this section, an overview of some techniques used in Rogue-like games is made, considering ShaP, CotN and the discussions above.

As seen in Figures 1, 3 and 4 and noted by Short [18] and Slash [19], rogue-like games tend to be organized by rooms and corridors in a grid. Considering the taxonomy defined by Cruz [4] and samples adapted from Short [18] – shown in Figure 5 –, it is noticeable that most PCG techniques for Rogue-like levels are online, which means their generation happens during gameplay, even if it uses seeds [4].

First, with algorithms defined, it is important to choose how rooms should be distributed and connected with each other. The algorithm used by Leite and Lima [12] presented a possible way to create paths between points.

Second, defining the critical path will help the algorithm to guide room connections towards the exit [18]. After connecting the main rooms, it is important to add secondary paths and secret rooms, as seen in Figure 1, in order to make the level exploration more interesting [20]. However, as noticed by Rouse III [11], a good designer must keep in mind that, throughout levels, rewarding players accordingly to their efforts is critical.

Additionally, "if your game relies on exploration for a large part of its gameplay value, it is probably a bad idea to make players backtrack through large sections of the level that they have already explored in order to continue in the game" [11]. In other words, players should not be dragged into similar or the same areas multiple times, otherwise they may feel confused or bored.

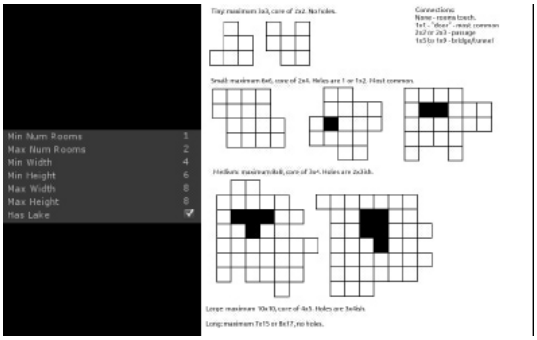

Figure 5: Left: example of level parameters changed by a designer in ShaP. Right: samples of rooms produced by ShaP's algorithm [18]

Finally, surrounding PCG and level design, it is noticed that all factors analyzed in section 4 and listed in this section contribute to the *replayability* of Rogue-like games. Technically, any game should be replayable, since the player can always start it again. However, *replayability* refers to the hability of a game to provide different gameplay experiences [2]. In terms of gameplay, PCG in Rogue-likes guarantees variety of set ups every time the game is played. As stated by Adams [3], other strategies to increase a game's replayability are: adding chance as part of gameplay, e.g secret rooms or treasure chests in CotN; non-deterministic enemies, i.e. enemies that do not always respond to similar situations in an equal way; and providing players with different roles and strategies, e.g. in CotN and ShaP players can choose from different characters which have distinct attributes and mechanics.

6 Conclusion

This papers presented a small overview of PCG techniques and a quick analysis of two Rogue-like games in order to understand how they are designed. Although there is a considerable number of academics researching about PCG and level design, knowledge from published games is usually kept secret.

After studying PCG techniques through all surveys and level design theory, it was possible to analyze ShaP and CotN in a proper, though still incomplete, way. ShaP and CotN were proved to be games that show accurately how grid-based levels can be generated

and which design problems one must be careful about when projecting these levels along with a computer.

In the future, one important advance with this research would be to combine and organize all surveys and taxonomies relating procedural generation, Rogue-like games and level design into one extended database. Likewise, it would be crucial to create a tutorial presenting basic information in order to start programming and designing Rogue-like games.

Also, it is critical to encourage academic researches and game developers from the industry to collaborate towards teaching level design and procedural generation techniques to novices. Although it seems good to keep secret complex programming and design techniques, sharing knowledge with fellow developers should be instigated.

REFERENCES

[1] D. Adams. Automatic generation of dungeons for computer games. 2002.

[2] E. Adams. Replayability, part one: Narrative. http://www.gamasutra.com/view/feature/131483/replayability_part_one_narrative.php, May 2001.

[3] E. Adams. Replayability, part two: Game mechanics. http://www.gamasutra.com/view/feature/131468/replayability_part_2_game_.php, Jul 2001.

[4] G. Cruz. Estimativa de qualidade de mapas procedurais para jogos do gênero roguelike. 2014.

[5] S. Curd. Crypt of the necrodancer chegando ao ps4 e ao ps vita. http://www.hookedgamers.com/features/2011/03/16/eras_of_gaming_1980_-_1984.html, Mar 2011.

[6] G. Duncan. No man's sky: How i learned to love procedural art. In *Game Developers Conference 2015*, 2015.

[7] B. Entertainment. Diablo 2. Digital Game, Jun 2000.

[8] B. Y. Games. Crypt of the necrodancer. Digital Game, Apr 2015.

[9] K. Games. Shattered planet. Digital Game: http://www.kitfoxgames.com/shattered-planet/, Jul 2014.

[10] M. Hendrikx, S. Meijer, J. van der Velden, and A. Iosup. Procedural content generation for games: A survey. In *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, volume 9, Feb. 2013.

[11] R. R. III. *Game Design: Theory and Practice, Second Edition*. Wordware Publishing Inc., 2005.

[12] G. Leite and E. Lima. Geracao procedural de mapas para jogos 2d. In *Computing Track Short Papers (Proceedings of SBGames 2015)*, pages 244–247, Nov. 2015.

[13] M. C. Machado, E. P. C. Fantini, and L. Chaimowicz. Player modeling: What is it? how to do it? In *Proceedings of SBGames 2011. Tutorials Track - Computing*, Nov. 2011.

[14] Mojang. Minecraft. Digital Game: https://minecraft.net/en/, 2009.

[15] Newzoo. Global games market will grow 9.4 https://newzoo.com/insights/articles/global-games-market-will-grow-9-4-to-91-5bn-in-2015, Apr 2015.

[16] Roguebasin. Berlin interpretation. http://www.roguebasin.com/index.php?title=Berlin_Interpretation, Jun 2016.

[17] B. Shenanigans. Crashlands. Digital Game: https://www.crashlands.net, 2015.

[18] T. Short. Level design in procedural generation. http://www.gamasutra.com/blogs/TanyaShort/20140204/209176/Level_Design_in_Procedural_Generation.php, Apr 2014.

[19] Slash. What is a roguelike. http://www.roguetemple.com/roguelike-definition/, Jul 2014.

[20] M. Stout. Learning from the masters: Level design in the legend of zelda. http://www.gamasutra.com/view/feature/134949/learning_from_the_masters_level_.php, Jan 2012.

[21] C. W. Totten. Designing better levels through human survival instincts. http://www.gamasutra.com/view/feature/134779/designing_better_levels_through_.php, Jun 2011.

[22] M. Toy and G. Wichman. Rogue, 1980.

[23] UKIE. Uk video games fact sheet. http://ukie.org.uk/sites/default/files/UK%20Games%20Industry%20Fact%20Sheet%2029%20April%202016_0.pdf, 2015.

[24] R. van der Linden, R. Lopes, and R. Bidarra. Procedural generation of dungeons. In *IEEE Transactions on Computational Intelligence and AI in Games*, volume 6, pages 78–89, Nov. 2013.

[25] H. Wilson. Crypt of the necrodancer chegando ao ps4 e ao ps vita. http://blog.br.playstation.com/2015/07/10/crypt-of-the-necrodancer-chegando-ao-ps4-e-ao-ps-vita/, Jul 2015.