Contents

1 Preface

This project can be divided into two parts, one half of which is partly based on the work done previous semester, with the other half produced this spring. It can be labeled as a split master thesis, in which some of theories, ideas and the developed application were carried over, and built upon.

Last semester the topic of investigation became procedural content generation, specifically with a focus on the creation of world interiors. Of particular interest back then were Lindemayer Systems, which were originally devised to visualize and model plants. However, as a study of the topic revealed, their usage went far beyond this scope. As a result a world creation grammar based loosely on these systems was devised. Based on this grammar, a game engine capable of building game worlds was developed.

Many of the theories and the developed application will be presented again in this report. The theories have however been expanded, both to work in conjunction with the other half of the project, but also in order to achieve a more solid theoretical base.

The other half of the project concerns itself entirely with narratives in video games. In particular, it was investigated how to tell narratives in a procedurally generated world, as effectively as possible, with the intention of investigating exactly to what extent it is possible.

This report is divided into ten major chapters, which are structured similarly to most traditional Medialogy reports. Chapters 2, 3 and 4 cover motivation, problem definition and research. Chapter 5 and 6 covers design and implementation, 7 and 8 the testing, and lastly, 9 and 10 are the discussion and conclusion, respectively.

Thanks to Stefania Serafin for her supervision, which I definitely should have made more use of, and TIGsource for the Assmblee competition, which provided me with the spritesheets and textures for the project.

2 Introduction

To argue that procedural content generation is the newest, latest, and greatest thing to pop up in the video game industry seems, from one perspective, silly. It was games such as *Elite* and *Rogue*, which were developed back in the early 80s that pioneered the techniques that allowed for games to be built algorithmically rather than by a designer. But at the same time, with a number of the biggest names in the video game industry suddenly starting to praise the methods and ideas pioneered, one may suggest that time is right to reinvestigate some of the ideas that David Braben, Glenn Wichmann and Michael Toy popularized almost 30 years ago.

It is not like some of the concepts and techniques related to procedural content generation have been completely ignored either. Blizzard and countless other companies put out games throughout the past two decades that have made extensive use of procedurally generated content.

Perhaps the most immediate benefit to procedurally generated content, is for small niche developers who cannot afford to hire designers and other graphics artists. Suddenly, even the smallest companies become able to build games with large and expansive worlds. Where such a feat would previously require a talented team of up to 100 artists, the majority of the world in a game like the indie massively multiplayer online game *Love* was built just by one person.

There does in fact seem to be an extraordinary amount of potential in games built this way; greater replay value[1] through randomly generated worlds, tons of customizability through randomized items, and even the ability to adjust the game experience to the player who is currently playing it.

The world of procedural content generation is however not without its challenges. Authored narratives, for one, seem like a highly problematic subject to approach in games with procedurally generated content. Arguably similar to the narrative paradox[2],

[1] Alas, replay value tends to become a highly subject term commonly used by video games players and media. It will within the context of this report be defined as the ability to replay a game and experience new content.

[2] Which deals with the problematic merging of narratives with interactivity

it appears that games which attempt to maintain a high level of procedurally generated content[3] must in some way or another forego or skimp on the narrative.

Some attempts to write games both narrative and procedural content have been made, perhaps most popularly in Blizzard's Diablo-series. In these games, the player must take to defeat the great demon Diablo, traversing a large number of dungeons. Most of the game is randomly generated, but whenever the game reverts into what can best be described as a storytelling mode, it does so in hand-made areas, and sections.

It is however not hard to imagine benefits of game which is entirely procedurally generated, yet maintains a narrative, even to the same extent as *Diablo*. The most immediate would be the enormous replay value such an experience might have, but other benefits such as variable game lengths[4] suddenly seem possible, enabling the developers to further adjust the game experience to the player. If the player is constantly thrown out into new situations, he may no longer be able to rely on the same strategies over and over, and will have to focus on constantly learning new approaches.

2.1 Preliminary Problem

The central theme in this report will centre on the procedural generation of game worlds, and the objective of telling narratives in these. Therefore the following initial problem formulation will be explored in the forthcoming pre-analysis;

How can authored narratives be told in procedurally generated worlds?

[3] Typically games which auto-generated worlds.

[4] http://www.steampowered.com/status/ep2/ep2_stats.php (retrieved 25/05/10) . A survey done by Valve suggest than less than 40% of the people who bought one of their games actually completed it. Perhaps procedurally generated worlds with adjustable lengths could be built, in order to satisfy those who hardly have the patience to sit through an entire game?

3 Preliminary Analysis

In order to explore and receive a sufficient understanding of the initial problem, it is necessary to first investigate the two main variables from the initial problem statement, procedural content generation and authored narratives. Specifically, it will investigate mechanisms, state of the art, and existing applications that have made use of procedural content generation in one sense or another. In this context, it also becomes necessary to produce and define a terminology, in which the rest of the project can be explored. Secondly, based on these definitions and observations, it will also be possible to delimit the scope of the project; as it will certainly be shown, algorithms for procedural content generation may permeate every level of a game's code and structure, and it therefore becomes a necessity to select a tight focus in order to answer the initial problem formulation.

The second half of this pre-analysis will deal largely with the definition of narrative in interactive media, and its implications in regards to structure, and perception of stories in video games. This is however a topic hotly debated. A large part of the debate spurs from the fact that, as some people believe, video games simply cannot be described as being narratives. While the purpose of this report is not to debate the validity of such claims, the discussions have generated a large number of different useful models, definitions and terminologies which may provide useful in order to determine exactly how to tell stories in procedurally generated worlds, and answer a number of questions that are bound to arise in the wake of initial problem.

3.1 Procedural Content Generation Part I

The purpose of this section is to provide a working definition and an extensive terminology of a number of concepts pertaining to procedural content generation, while at the same time providing a brief analysis of a number of existing applications all using procedural content generation to a greater or smaller extent. While the concept of procedural content generation has been around at least 30 years, it has only been vaguely explored in academic studies.

It seems to make sense to preface the discussion of Procedural Content Generation, and its techniques in a similar fashion to how the past semester's report was written. Namely, with a look at two of the video games, both of which originally played part in pioneering procedurally generated content in video games[5], *Elite* (1984) and *Rogue* (1980).

Rogue, originally developed by Glenn Wichmann and Michael Toy is built around exploring a number of pseudo-randomly generated dungeons that would be different every time the player died, and had to restart the game (Wichmann, 1997). The central idea behind the game was to create an environment which could be explored infinitely, with monsters, rooms, keys and weapons being put in different locations every time. This way, it would seem fresh every time it was loaded up. Elite employs a similar technique to generate its universe, although here the challenge was of a different nature. David Braben wanted a vast and incredibly rich universe which would take the player many months to fully explore. However at the time of development, most software companies were typically tiny in size, just as the technology available to them was, by modern standards, limited. As such, he had to come up with a different way to achieve his goals. His solution, rather than designing the entire universe by hand, it was instead generated through a set of simple mathematical principles. Using this method, he was able to generate a universe with 8 galaxies, each containing 256 planets complete with their own unique (and pseudo-randomly generated) names and inhabitants (Spufford, 2003).

Interestingly, the challenges faced by the original development teams still persist today. Smaller independent developers have to work with very limited budgets, and may therefore not be able to employ a limitless number of artists[6]. Instead, it makes sense for them to create much of the content automatically. Similarly, replayability, as outlined in the introduction, is a different feat which many game developers strive to achieve, and perhaps the best way to do this, is to have the player experience something new every time he plays the game.

Finally, some types of content and effects simply make more sense to do procedurally. *Far Cry 2* (2008, Ubisoft) makes use of a fire propagation algorithm to make the fire spread realistically throughout the game's locations. Similarly, *Pixel Junk Shooter* (2009,

[5] The demo scene was also instrumental to its development and is credited as being one of the big inspirations to one of the modern pioneers, Will Wright. In this report however, the aforementioned video games will remain the focus.

Sony) uses fluid dynamics to model different types of smoke and liquids, all of which can change and shape how the game world looks and plays.

The very ideas expressed and concepts developed by Braben, Wichmann and Toy seem to encapsulate the very essence of what procedural content generation is. Based on their work, it is hereby possible to provide a basic definition[7] of procedural content generation;

> Procedural Content Generation is generation of content in video games or interactive media through the use of algorithms, by a machine rather than a human designer.

However, this definition is by no means exhaustive to the concept of procedural content generation; it says little how the algorithms has been implemented, to what extent it is being used, and if it is even stochastic or completely pre-determinate.

Julian Togelius et al. (through perhaps the first serious academic attempt at creating taxonomy for the topic) suggest a number of continuums to distinguish between different types of procedurally generated content (Togelius, 2010). While some of the distinctions are perhaps not exhaustive, the list may prove a useful starting tool for any developer, or other academic discussions.

Here, distinctions are made between *online and offline created content* (p. 2); that is, whether the content is created only when the player plays the game (For instance, *Rogue*) or whether the developers get to decide how the universe ultimately looks (coincidentally *Elite*). However, this distinction may prove to be rather vaguely defined. Whereas the developers of Elite tried thousands of different random seeds before finally settling on a universe they liked, the universe is not generated until the player loads up the game. In other cases, such as *Far Cry 2*, procedural algorithms were used to generate large amount of different content, but the final product was however saved to a number of files containing specific (amongst others) vertex and texture information. It seems rather clear where *Far Cry 2* would be placed on this continuum (towards Online), but in the case of *Elite* it arguably becomes unclear. It seems to have a foot in each camp.

Secondly, a distinction between *necessary and optional content* (p. 3) is made. As a developer, it is necessary to consider whether the content you are procedurally generating is optional or essential to the experience. If the location which the player is travelling through is integral to the story line, it is important to make sure that the

[7] This definition, albeit in a slightly more crude form, was also provided in the previous semester's report, which in some aspects, as mentioned in the reading guide, served as a primer for this thesis.

algorithm which the game employs is fool proof. If the content is optional, such as a randomly generated weapon (of which the player is bound to pick up loads of different versions of), it matters little if it is borderline useless.

Random Seeds vs. Parameter Vectors dictates the extent to which the content is instantiated (p. 3) . The ends are here contrasted by, on one side, a single value. A parametric input may however specify detailed properties of the created content.

It is also possible to distinguish between content that is *stochastic or deterministic* (p. 3). Although all content generated on a computer is deterministic in nature, some algorithms may be set up so that they generate a different result with the same input. A completely deterministic algorithm, on the other hand, will always generate the same result. Again, *Rogue* and *Elite* seem like suitable examples of these two approaches. Similarly, creature animation in the game *Spore* is only generated once the creature has been created by the player, but there is nothing stochastic about the process.

Lastly, a distinction between *constructive and generate-and-test* algorithms is made. Constructive algorithms are written to generate the content only once, with the assumption that they are strong enough. Generate-and-Test algorithms are, on the other hand, are not necessarily strong enough, but due to an inherit testing-mechanism, they may be run more than once if the first results fails to meet the specified criteria (p. 3).

Outside of Togelius' distinctions, Mike West, co-founder at Neversoft, distinguishes between content which is generated through ontogenetic algorithms, and ones he calls teleological algorithms (West, 2008). Teleological algorithms are described as algorithms which directly models the properties of real systems. Ontogenetic algorithms, on the other hand, are merely approximations, which may skip over a whole number of steps in order to achieve the desired result. The fire propagation in *Far Cry 2* and the fluid dynamics from *Pixel Junk Shooter* are examples of teleological algorithms. Typical ontogenetic algorithms could be dungeon and universe generation algorithms, or l-systems which are used to simulate growth of plants.

Perhaps, as a contribution to these distinctions, it could be possible to also regard the *scope*, *breadth* and *depth* of the implemented algorithms. Briefly touched upon (and quite certainly related to) is the distinction between necessary and optional content. Some games feature only a single type of procedurally generated content, whereas others are comprised entirely of it. Some games use a multitude of different algorithms on many different levels, and in some cases, this directly affects what type of game is produced.

3.1.1 A note on Authorship and the Smallest Authored Unit

As outlined in the introduction, authorship is a central theme throughout the report. While this is something typically associated with narratives and stories, the term is in this report used also to contrast to the machine-generated content of the content-generating algorithms.

An example of authorship in this project could thus be the narrative which the player gets to experience throughout his playing sessions. But smaller and more concrete examples of this authorship could also be made. The audio the player listens to, when performing an action, the individual graphical tiles which make up the world, and the graphics representing the player and the non-player characters. All of these have been hand-crafted by a designer, and are all part of the authorship which makes up the game. In most video games which make use of procedurally generated content, algorithms are forced to make use of the authored content to generate worlds, weapons, vehicles and so forth[8]. It therefore makes sense to talk of the smallest authored unit, that is, the smallest piece of authored content that the algorithm can use to put together its goal.

To exemplify what is meant with this, consider a game such as Runic's *Torchlight* (2009), a game coincidentally inspired by *Rogue*, in which the player must traverse a number of deep and dark dungeons, completing various quests for the people who live in the town above the dungeon. The dungeons in the game are put together from a number of smaller set pieces (a hallway, a room etc), which are connected together through an algorithm. In this case, the smallest authored unit would then be these set pieces. Contrast this to a game such as *Spelunky* (Yu, 2008) where the player assumes the role of a little cave explorer who travels a large number of random caves in order to find as much treasure as possible, and occasionally, rescue a damsel in distress. The game is divided into a set of small tiles, which each tile representing a different material (a destructible rock, an impenetrable ditto and so on), and in this game, the smallest authored unit could be said to be one of these tiles.

The point being made here is that size and possible variation of the smallest authored unit directly influences how the game turns out. While neither of the smallest authored units in either game feature much variation, the levels in *Spelunky* can potentially be

[8] There are obvious extreme exceptions such as *Kkrieger*, the 96kb-sized first person shooter, where it is difficult to find any authored content, but these do not appear to be the norm.

much more varied as the unit simply is smaller, and as a result, can be combined in a greater amount of ways.

3.1.2 From Definitions to Narratives

The previous couple of pages attempted to provide a working definition of one of the most popular concepts within the video games industry, procedurally generated content. Secondly, the distinctions and terminology created by Togelius et al. was reviewed and evaluated. It is on the basis of these definitions that the rest of the report will be regarded.

The next logical step is to investigate the other major part of the project, authored narratives in video games.

3.2 Can Video Games Be Narratives?

It seems prudent to preface any foray into the discussion of narratives in video games by providing a general idea of the debate which have dominated most of the academic world interested in the topic of video game narratives. If anything, it may be useful simply because the debate has spawned a number of useful models and theories which may help answering the initial problem formulation.

Distinctively, it seems that two (arguably paradigmatically) opposing views exist, specifically the viewpoint by those who support narratology and those who support ludology as the primary tool of video game analysis. One group (Marie L Ryan (Ryan, 2006) (Ryan, 2001) (Ryan, 2004), Henry Jenkins (Jenkins, 2003) and Janet Murray (Murray, 1998) amongst) sees it necessary to include narrative tools to a greater or smaller extent as part of the video game analysis methodology. The opposing viewpoint, as theorized by the ludologists (including some people such as Espen Aarseth (Aarseth, 1997) (Aarseth, 2004), Jesper Juul (Juul, 2001), Gonzola (Gonzalo, 2003)), is contrarily centred on the concept of *ludus* (play). They at the same time argue that narrative analysis may not be a suitable tool for investigating video games. In fact, some prominent ludologists (Juul (Juul, 2001), Eskelinen (Eskelinen, 2001)) even argue that narratives in video games may be nothing more than mere ornamental details, or even work against the elements that make it a video game.

It should come as no surprise to the reader (especially given the problem formulation of this project) that the viewpoint chosen in this report is one in favour of narratology, but that is not to say the ideas behind Ludology and the theory which have been produced from it has not been considered and evaluated. This report assumes one specific point of view that video games can potentially, but not necessarily, contain narrative elements. Secondly, as a consequence hereof, it is possible to re-communicate the experiences the narrative in a video game. As the forthcoming section will illustrate, theories and models suggested by each party can often be used together, and as such, this thesis will make an attempt to make use of the theories of both.

3.2.1 Narrative Criticism

In order to explore narratives in procedurally generated worlds, it would not be completely non-sensical to explore various video game genres and structures in an attempt to determine which type of genres offer a higher level of “retellability”.

A point oft repeated by ludologists is how the quality of stories in video games remain lacklustre. Espen Aarseth writes;

> *"While many adventure games are clearly attempts at telling stories, cleverly disguised as games, the limited results they achieve (poor to nonexistent characterization, extremely derivatives action plots, and wisely, no attempts at metaphysical themes) should tell us that the stories are hostage to the game environment, even if they are perceived as the dominant factor"* (Aarseth, 2004 p. 367)

His logic is however fallacious. The suggestion that the quality of story-written games[9] makes them *"hostage to the game environment"* seems almost unconnected, and is hardly verifiable from the assertions he makes. Instead, the low quality of the writing in modern computer games may be due to a number of different factors not considered by Aarseth, for instance the newness of the medium (people are still learning how to use the medium), and the number of capable authors within the video games industry .

Juul similarly points out that if one were to retell the story of, for instance, Half-Life in detail, it would become excruciatingly dull (Juul, 2001). However, save the most didascalic[10] examples, stories are never retold the way they occur. Even on a basic level of storytelling, it is possible to differentiate between fabula, plot and the story which is being delivered. Therefore, it can be argued that Juul bases his argument on a faulty assumption.

If one were to consider the ludologist's point of view, that narrative is irrelevant to the structure of the game, quite a few things could be changed in any game without changing how it would play. However, these changes might also make the game semantically incongruent. If for instance, in the game of *Tomb Raider*, if Lara Croft was a male characters, some of the possible actions in the game would no longer make semantic sense; many of the possible actions are in large part due to her background story, how the authors envisioned her upbringing, her training, and so on.

If no narrative persists in a video game, the game may simply end up becoming a string of semantically incongruent events. As Michael Nitsche similarly points out in his book, *Video Game Spaces*,

[9] Coincidentally, it is the personal belief of the author of this thesis that many of games contain both decent characterization and some metaphysical themes. Perhaps a great example of this is the famous "*Would you kindly*" scene from Bioshock, which can be regarded a comment on free will, both in real life, but also within video games.

[10] In the context of this thesis, the didascalic can be considered the polar opposite of total abstraction.

> *"Without narrative elements, a 3D video game would be in danger of disintegrating into singular unconnected splinters of momentary interaction."* (Nitsche, 2008 p. 64)

As such, it is impossible to disregard narratives in video games, if anything because they seems to create the context in which the game is played. As will be seen, they may not be entirely instrumental to the experience, but if no narrative is present, the experience may gyrate towards abstractness. However both Aarseth (Aarseth, 2004) and Gonzola (Gonzalo, 2003) suggest that traditional definitions of narratives may prove unsatisfactory, when trying to analyse game experiences.

3.2.2 Can Video Games Be Simulations?

Gonzalo Gonzola argues in his article, *Simulation versus Narrative: Introduction to Ludology* (which coincidentally popularized the term itself) that video games should be viewed not as traditional media, but rather as simulations. (Gonzalo, 2003)

His definition, although he states himself that it must be treated as being provisory, states that,

> *"To simulate is to model a (source) system through a different system which maintains to somebody some of the behaviours of the original system."* (Gonzalo, 2003 p. 3)

He furthermore presents the argument (and this is a key argument presented by many ludologists) that simulations, unlike traditional media, are not representational. What this essentially means is that where both traditional media and video games can retain audiovisual characteristics, video games are manipulative, and can react to the inputs of the user.

Where his (admittedly provisory) model breaks down completely, is when studying context-less abstract games. While the most *die-hard* narratologists claims that Tetris can be interpreted as a narrative (Murray (Murray, 1998)), the game does not model anything tangible and real; it therefore does not make much sense to treat it as being a simulation, anymore than, as pointed out by Ryan, a game of Chess or Football (Ryan, 2006). As such, the game actually falls outside the definition provided by Gonzola.

Where his definition incidentally proves the most useful, in regards to the initial problem formulation, is when dealing with games that contain strong narrative elements, such as *Half-Life*. Gonzola suggests that while games contain narrative elements, these are typically in place to construct the rules, goals and constraints of the simulated system. In the case of *Half-Life,* the narrative is indeed used, both to construct the rules of the game, but also the objectives of the player.

In one particular harrowing sequence, the player, while trying to escape self-destructing Black Mesa Research Facility, comes across a number of soldiers from the military. The player gets to watch as a group of soldiers mow down a number of scientists running towards what they think is the rescue crew. Previously thought saviours, it turns out that they are actually in there to kill everything that moves, and make sure that no word gets out as to what happened there.

Here, the narrative is used to do two things; establish a new set of constraints in the game, through the introduction of a new enemy, and also to change the active goal. Where the goal had previously been to escape the facilities, before the roof comes falling down, the goal subsequently becomes to escape the grasp of the army, and avoid capture or even being killed by them.

Similarly, halfway through the game, the player is also introduced to a change in the rules of the game world. Where the player throughout the game is capable of moving, jumping, and manoeuvring relatively within the laws of physics, he is at one point introduced to a device, called the *long jump module*, which effectively changes the laws of physics, and thus the rules and constraints of the game. Equipped with this device, the player becomes capable of jumping much longer distances and access areas otherwise inaccessible, which becomes a useful tool later in the game.

It is not unheard of for narratologists to share some of the views suggested by Gonzola. Marie Laurie Ryan argues that;

> *"The narrativity of games is not an end in itself but a means toward a goal"* (Ryan, 2004 p. 349)

However, the role and extend of narratives in video games may vary depending on the type of game.

It therefore seems prudent to first investigate the role of narratives in different types of genres. Here, the dichotomy of *paida* and *ludus may prove useful*.

3.2.3 Paidia, Ludus and Player Roles

One distinction commonly made by both narratologists (Ryan, 2001) and ludologists (Gonzalo, 2003) (Juul, 2001) is the dichotomy of games intro the structures of *ludus* (game) and *paidia* (play), two terms originally coined by Roger Caillois Caillois used paidia to describe the type of games children play[11], whereas rule-based games such as chess and football can be described by the ludus structure (Ryan, 2001).

Both Jesper Juul and G. Gonzola argue that most games can be placed into either structure. Ludus games are typically goal-oriented, and can be exemplified with with first person shooter games, and real time strategy games. Typically, as Gonzola argues (Gonzalo, 2003), these games typically tells the player to do X in order to achieve or Y. *Paidia*, being tied with the concept of play, tend to include more open-ended goal-less games, such as *The Sims*, where the player himself gets to decide what he wants from the experience.

Marie L Ryan similarly distinguishes between the concepts of ludus or paidia, but does so in terms of the roles which players may assume whilst playing games. Ryan, basing her theory on the work of MUD[12] designer Richard Bartle uses the two terms to distinguish between the different types of player roles, primarily in multiplayer worlds (Ryan, 2001) (Bartle, 1996) However, as will be argued, these roles can, with some modification, also be applied in single player games. In particular, she sees *ludus* players as *Achievers* and *Killers*, whereas *paidia* covers *Socialisers* and *Explorers*.

What these characterizations (despite their "stereotypic-ness") may provide is an idea of exactly the type of narrative to design. What is necessary to consider, is that MUDs generally are persistent worlds. They typically contain no winning conditions, and the worlds continue existing, even after the player logs off. As such, some crucial distinctions must be made.

Bartle describes the achiever with the following;

> *"Achievers regard points-gathering and rising in levels as their main goal, and all is ultimately subservient to this."* (Ryan, 2004 p. 199) (Bartle, 1996)

This counterpart is easy to find in single player games; both in role play games (*Morrowind, Oblivion,* and especially the action role playing counterpart*, Diablo*), but the achievement system like the one present on Microsoft's Xbox 360 is also a perfectly valid example of a system where *achievers* are rewarded.

Killers can be described as;

[11] Eg. Games of Make-Believe

[12] A MUD is a multi-user Dungeon. These are, by many, considered one of the ancestors to modern massively online games.

> "*Killers get their kicks from imposing themselves on others [...] [They] attack other players with a view to killing off their personae*" (Bartle, 1996) (Ryan, 2004 p. 199)

Some games such as *Just Cause 1* and *2*, *Grand Theft Auto* and *Red Faction Guerilla*, employ gameplay mechanisms that reward destructive behaviour similar to how it may be rewarded in MUDs.

Marie Laurie Ryan points out that both *killers* and *achievers* tend to be so goal-oriented that narratives sometimes end up playing a very little role in their experience.

Socialisers are interesting from a narrativistic point of view. Where they depend largely on the interaction from other players in MUDs, Marie Laurie Ryan argues that they, in the context of single player games;

> *"...will exchange stories about the game-world, perform small narrative scripts of their own invention, and generally enjoy the enactment through role-playing of the narrative design written into the game."* (Bartle, 1996) (Ryan, 2004 p. 199)

Lastly, the explorers seek to push the boundaries within the game. They are described by Bartle as;

> *"Explorers delight in having the game expose its internal machinations to them [...] The real fun comes only from discovery, and making the most complete set of maps in existence."* (Bartle, 1996) (Ryan, 2004 p. 199)

Ryan lastly also argues that perhaps, it is necessary to add a 5th type of player role, namely those who may play the game for the story, or narrative (Ryan, 2004). These can be closely related with *explorers,* who seek to explore the spatial narrative of the world, but may fall within the *ludus* since experiencing the narrative, at least as they exist in many video games today, is a linear goal-oriented process. Table A illustrates the different player roles within the frameworks of *paidia* and *ludus*.

TABLE A | THE PROPOSED PLAYER ROLES

Paidia	Ludus
Socialisers	Killers
Explorers	Achievers
	Story Actor

Aarseth argues the fact that most modern games (outside of the classic sales pitch) rarely can be described using just one genre (Aarseth, 2004). And indeed, it seems as if several of the aforementioned player roles can similarly be combined and mixed. If a game such as *Crackdown* is considered, for instance, the game rewards a number of different play styles, allowing people to choose how to play, or even become some sort of a hybrid between the different roles. Scattered and hidden throughout the game world are a number of orbs that, when collected, unlocks a set of new abilities. This is a mechanic which rewards Bartle's *explorer*-type and to some extent also the *achiever*-type. However, this is by no means the main objective of the game, which is instead focused on an armed conflict between armed thugs and the city's police force. Thus, the main mechanics of the game is primarily meant for people who assume the *killer* role.

Where *Crackdown* serves as an example of a game in which these player roles may be interrelated, the *RPG*[13] genre, for instance the games produced by Bethesda, may serve as example of a genre where people get to practice different roles, perhaps even without any inter-relation at all. In the case of Bethesda's Oblivion, there are those who play it to play through the epic authored storyline. However, given the completely open world of the game, enjoyment from the game can be had in so many ways. The *achievers* can strive to either become as powerful as possible, or perhaps complete every single smaller quest available to them. *Killers* may get a kick out of wrecking havoc in the world, by slaying non-player characters (NPCs) both friendly and antagonistic. As the world of Oblivion is so vast, *Explorers* are free to explore the world as they see fit. Some

[13] Role playing game

people have even dedicated themselves to role-playing in the world, purposely avoiding all adventure, while writing stories about it[14].

While *paidia/ludus* dichotomy and the various roles, at first, may seem like they are useful for describing and categorizing genres and players only, they may also have a secondary purpose; specifically they may potentially be used as tools to determine the type of game and genre necessary to solve problem of this report. For instance, it makes little sense to pick a genre or develop a game which typically does not contain any authored narrative. As such, most *Paidia* games probably would make a bad fit for consideration.

If we are to look at the games of Will Wright, one of the industry's most outspoken supporters of procedurally generated content, and judge in which categories they would fall, most of them, save the first half of *Spore,* would fall into the category of *paidia*. The goals, both in *Sim City* and *The Sims*, are largely up to the players themselves to decide; whether it revolves building the biggest house, befriending your neighbours (*The Sims*), building the biggest city, or earning the most money (*Sim City*). *Spore* is arguably a bit different, as the first half of the game is entirely goal driven (become the superior species of your planet), and while there is a certain amount of freedom in regards to exactly how you can customize your species, most actions in the game are completely binary (befriend the other species or go to war). However, even this half of the game contains very little explicit narrative, leaving it entirely up to the player to construct his own story. Therefore, while Will Wright's games may be considered revolutionary in regards to procedural content generation, they may not be the best subject for study in regards to narrativism in video games. Instead, it seems it is necessary to study games with more coherent narrative structures.

The next section will deal with exactly this, namely examples and discussion of various video game structures, as suggested both by narrativists and ludologists.

3.3 Game Structure

Although Espen Aarseth also fundamentally disagrees with the notion of video games as narratives, his description of the structure in a number of video games, particularly also

[14] http://livinginoblivion.wordpress.com – *Livin' in Oblivion* is an example of a humorous blog about Nondrick the non-adventurer, and an instance of a computer game turned into a story-telling medium.

Half-Life (Valve, 1998) and *EverQuest* (SOE, 1997) is worth relating to (Aarseth, 2004) *Half-Life* is described as a string of pearls; where each pearl can be considered a little world, with its own set of possible choices. However, as the overarching structure of the game is entirely linear, the experience is ultimately roughly the same for each player. Figure 1 illustrates this type of structure.

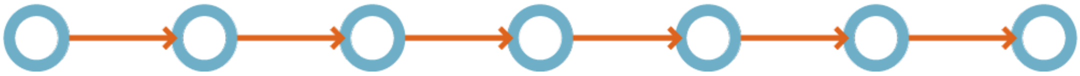


FIGURE 1 | THE PERFECTLY LINEAR STRUCTURE OF HALF-LIFE

Aarseth also examines the structure in *Everquest*, which he argues is actually somewhat similar to the one in Half-Life, however with one crucial modification. Where the structure of Half-Life can be represented a singular one dimensional string of events, Everquest can represented with a two-dimensional structure. Though progression through the game is still relatively linear, it is possible for players to take certain detours at various points in the game, and experience a number of side quests that may be less or more important to the progression of the game (Aarseth, 2004). Both of Aarseth's structures are entirely focused on the events which the player must experience in order to progress in the game, and focuses very little location and spatiality.

Marie Laurie Ryan also suggests in her book, Narrative as Virtual Reality, a number of different structures that are typically common in various types of different interactive media (Ryan, 2001 p. 246).

Her first model, the complete graph, is used to describe narratives which can be experienced in any order, although as she writes that constructing a narrative using such a model is perhaps nearly impossible, at least without losing coherence (p. 247). Figure 2 illustrates this type of structure.

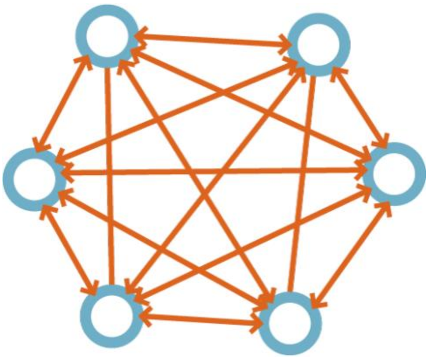


FIGURE 2 | THE COMPLETE GRAPH

Perhaps of greater interest is the two models entitled *vector with side branches* and *the maze,* and one called *Action Space, Epic Wander and Story-World* (henceforth, to ensure brevity this will be referred to as just the story-world). The first is identical to Aarseth's Everquest-structure, with a number of fixed narrative elements which the player must experience, and a number of smaller events which the play may experience at his own leisure. (p. 250)

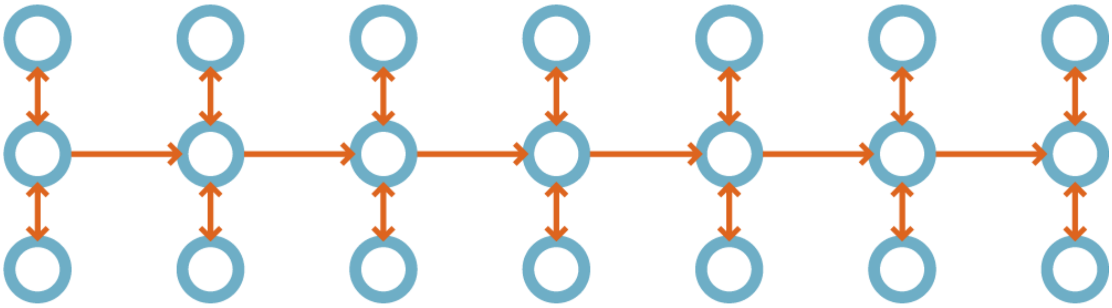


FIGURE 3 | THE VECTOR WITH SIDE BRANCHES

The maze is, according to Ryan, most commonly used in adventure games[15] (p. 251). Structured like a sparse graph[16], with more or less connections, the player is allowed a certain amount of freedom to experience the narrative as he/she sees fit. Some edges on the graph may be unidirectional others bidirectional, and as such, this type of structure can be structured in a multitude of ways. Figure 4 illustrates this type of structure.

[15] Marie Laurie Ryan's video game terminology may indeed be a little confusing to some. Adventure games, she describes, encompasses first person shooters like *Half-Life* and *Halo*, but also the role playing genre. Simulation games tend to encompass games such as *Sim City*, and other open ended strategy games, whereas mystery games tend to involve uncovering past mysteries, such as *Myst*. However other titles, such as Lucasarts' classic scumm games (Incidentally and perhaps even more confusingly, these are typically are typically described as Adventure Games by popular media) may also fit into this category.

[16] A sparse graph is a graph where every vertex is not connected to every other vertex. This can be contrasted with the previously presented complex (or dense) graph.

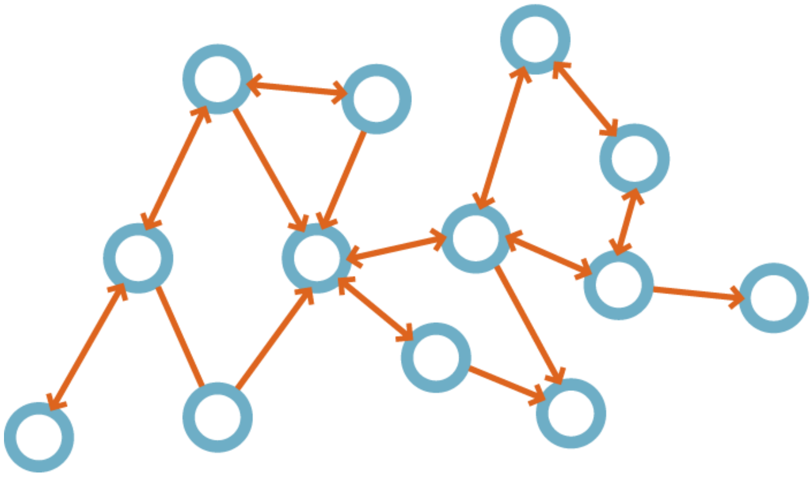


FIGURE 4 | THE MAZE

Lastly, the *story world*, whilst looking similar to the maze, is different in that it represents a spatial distribution of events and narratives. Whereas the previous structures are typically focused on one overarching narrative, the *story world* foregoes this in favour of smaller, localized narratives (p. 256). Marie Laurie Ryan uses the example of theme parks, which may contain several smaller thematically different sub-worlds within the theme park itself. However, certain open-ended RPGs (such as *Fallout 3*) may also to some extent share this type of structure, in which the player may visit a large of different cities, experience the daily life of multiple non-player characters and solve individual tasks proposed by the inhabitants.

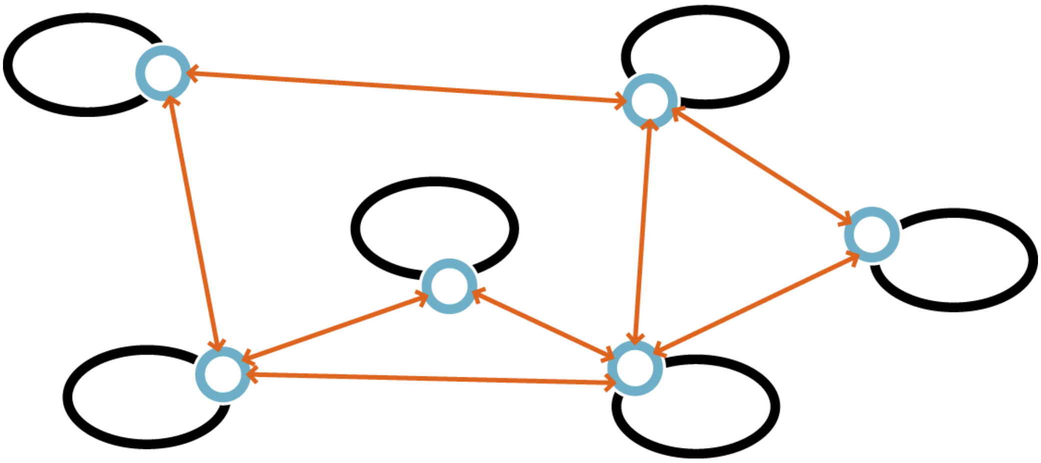


FIGURE 5 | THE STORYWORLD, IN WHICH EACH BIG ORB REPRESENTS ITS OWN SET OF EVENTS

These structures are by no means exhaustive. Classic *Lucasarts* games such as *Sam 'n' Max: Freelance Police, Day Of The Tentacle* and *Grim Fandango* sometimes feature pieces of narrative that can be experienced non-linearly, and others which must be experienced in order. As such, if one were to create a diagram based on their structure, it would look similar to either the *vector with side-branches* or *the maze* except that in order to traverse to the next major node on the graph, the player would have to experience all side-branches, in whatever order the player sees fit.

3.3.1 The Quest Structure

Perhaps, as Aarseth (Aarseth, 2004) and Nitsche (Nitsche, 2008 p. 56) suggest, in order to build a coherent video game structure, it may make sense to view games as quests, rather than narratives. According to Aarseth, every game which contains some notion of narrative must also per definition contain a set of quests. This is perhaps most obvious in open ended games such as *Grand Theft Auto* and *Fallout 3*, where the player seldom experiences one coherent narrative. Instead, the player may undertake a number of different assignments (quests) at different times, and complete them at their own leisure.

Joseph Campbell's monomyth (or The Hero's Journey) structure may, as suggested by Michael Nitsche, may prove a suitable model to use in conjunction with one of the previously outlined structures. Devised originally as a cyclic quest model, it can be used to describe a large number of folk tales, quests, and myths, and even large amount of modern literature. While, as Nitsche points out (p. 61) the model has been criticised because of its reductionist nature, it may still prove useful for a number of reasons, if anything, because it can be treated as a prototypical story structure.

The model can be divided into two halves. The upper half symbolizes the hero's ordinary world, or the "day" world, and the lower half which represents the world of adventure, or the "night" world. A story following the structure of the hero's journey can be divided into a number of smaller events. Typically, the ordinary world is missing something, or is in one way or another in need of healing. The hero is called to arms, but typically refuses his call, but only until he is introduced to a mentor. With his assistance, he ventures into the night world to answer the call. With the help of allies, he must here face his own innermost demons and a supreme ordeal. Having defeated this, the hero is granted the ability to heal the world, and must go back to the day world to do so. As previously mentioned, this quest model is cyclical in nature, and as such, upon returning, he may at this point be thrown into another adventure.

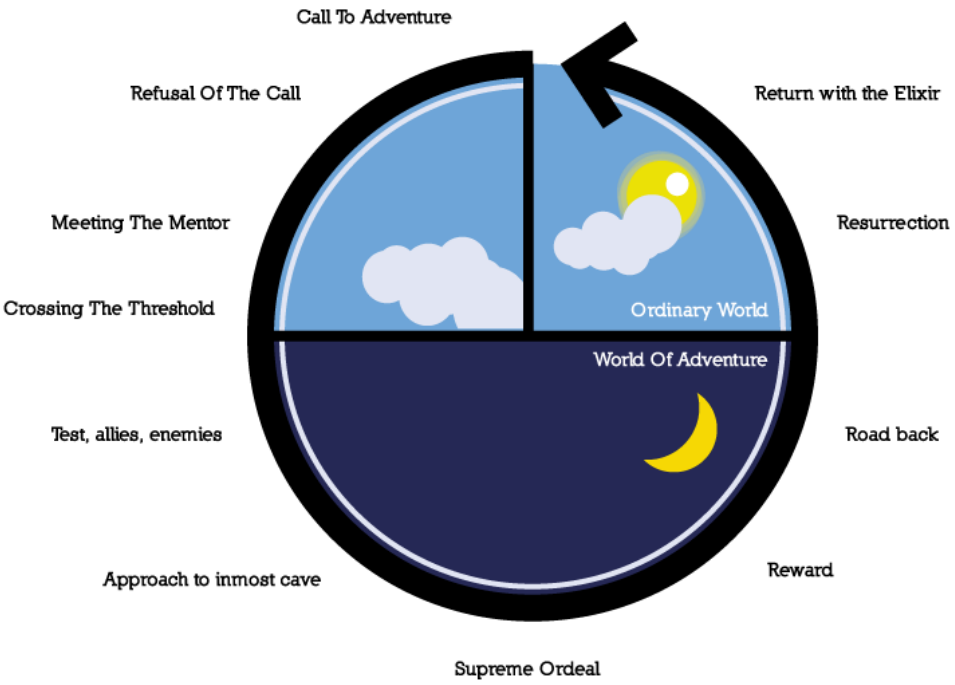


FIGURE 6 | THE MONOMYTH, AS PRESENTED BY (NITSCHE, 2007 S. 60)

While containing some preconceived notion of spatiality or time (although the hero's journey may be both physical and mental) the model contains no inherent spatial or temporal restrictions. As this project deals with the creation of a game world in which spatiality remains an unknown factor, this model may be an ideal fit for use, in combination with a generic game structure.

3.4 From Authorship to Procedural Content

The past few sections have covered primarily two topics, procedural content generation and narratives in video games. Although either section barely scratched the surface of both topics, it seems as if this point in the report is suitable for further delimiting the project, whilst redefining and stating the final problem statement. The brief foray into the world of procedural content generation revealed a multitude of possible ways to regard the topic. As the majority of the presented dichotomies could be treated as continuums, each point on the list can be regarded when developing video games. In fact, in the context of this project, a few of the points on the list already seems answerable, whereas others may have to wait until the project has been properly delimited, and new theory has been added.

The study of narratives in video games provided a number of useful models potentially useful in combination with the theories of procedural content generation. The proposed player roles, for example, are worth putting up for consideration when determining the type of game to develop. As the initial aim of this project was to work with narratives in a replayable randomly generated world, it makes sense to build the game around the story actor role. Moreover, using Gonzola's categorizations, the game would definitely have to fall within the category of *ludus*, as a stricter linear model of progression can be applied.

3.4.1 Introducing "Metroidvania"

Metroidvania, a truly terrible neologism invented by fans and media of the old *Castlevania* (1986, Konami) and *Metroid* (1986, Nintendo) games represents a genre that may be interesting to study in the context of this project.

Typically experienced from a 2D perspective, akin to platform games such as *Super Mario Bros* (1985, Nintendo) and *Sonic the Hedgehog* (1991, Sega), these games feature large *explorable* non-linear layouts. At the same time however, they are typically presented with a completely linear narrative structure.

Metroidvania games use a goal oriented game mechanic in which new sections are continuously opened up for the players. New abilities may lead to new areas (such as the ability to control the wind, which may allow players to cross an otherwise stormy bridge or the ability to jump higher, which might lead to previously inaccessible platforms). This structure is not dissimilar from older games such as *Doom* (1992, iD Software) and *Wolfenstein 3D* (1993, iD Software) where the players had to find keys in order to explore each micro-world. Where *Metroidvania*-style games are different, however, is that they are built around one continuous world, in which everything is experienced.

Using the models presented in 2.3, it is possible to present a new model that outlines a generic structure in the Metroidvania games. The layouts of the game worlds are typically non-linearly structured, whereas the narratives are typically presented in the same order. This allows for a unique structure, where the plot is driven forward in a fashion similar to first person shooters, such as *Half-Life,* but where the game may be laid out, structurally and spatially in a fashion similar to the maze. Figure 7 illustrates how the types of structures are merged in this type of game.

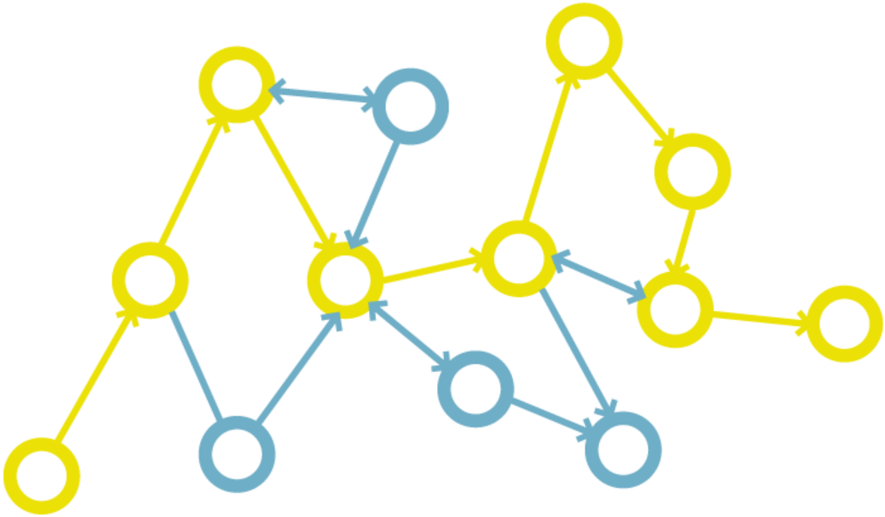


FIGURE 7 | THE "METROIDVANIA" STRUCTURE IN WHICH THE LAYOUT OF THE WORLD MAY BE NON-LINEAR, BUT THE WRITTEN NARRATIVE (THE YELLOW PATH) IS PRESENTED ENTIRELY LINEARLY.

If one were to consider the categorization of people playing Metroidvania games, they would typically fall within the category of explorers (albeit the most literal interpretation of the role) and story actors. As such, a stochastically generated world suddenly seems like an ideal for this type of genre. This will allow for the game to be different every time it is replayed, greatly increasing the *replayability* of the game.

It is worth considering the implications in terms of narrative to such a proposed game. Although a stochastically generated narrative may provide an even more varied experience, it seems necessary to consider Gonzola's and Ryan's points about the role of narratives in video games. Whether they are prevalent to the story or not, they are excellent mechanisms for setting up rules, goals and constraints for the game.

There are other challenges and limitations to consider, when building a *Metroidvania* type game with a procedurally generated world. If the world is generated stochastically, it becomes difficult, if not impossible to map out plot events spatially at the very beginning. Instead, it becomes necessary to apply an algorithm which spaces out the various plot events evenly. However, even if the spatial locations are unknown, it remains and important point that the story could still potentially be told in the same order as before.

Narratives can be told through different means, both auditory, visually, and perhaps even through tactile and haptic feedback. However one delimitation that will be imposed in this project is the sole focus on visuals as a story telling mean.

Secondly, this project will focus solely on narrative as an individual experience, rather than as a social one. Although no material on the subject of the latter has been studied, it is easy to imagine that the influence of other people experiencing the same narrative as one self may influence the perception of the experienced narrative. Therefore only single player games will be taken into consideration throughout the rest of this project.

3.5 Problem Formulation

The work done in the pre-analysis does indeed suggest a number of methods for telling an authored narrative in a procedurally generated world. It suddenly seems prudent to start investigating not the possibility, but the extent to which it is possible to do so, thus leaving us with the following final problem formulation;

To what extent can a linearly structured authored narrative be interpreted, if presented in a procedurally generated non-linear random video game world?

4 Analysis

As the focus has now shifted from the plausibility of telling a narrative in a procedurally generated space, to the possible effectiveness of such an endeavour, it becomes necessary to explore a new range of topics. The pre-analysis outlined a number of structural models, both useful to plot the events of the story, but also to outline the spatial organization of a potential game world. However, while it is now possible to structurally outline a game world, very little insight was given into how to actually tell an authored narrative. As such, the first topic explored in this analysis addresses specifically this. Briefly, it also touches upon various types of narratives presentable in these game worlds as these two topics are related.

Secondly, the topic of procedural content generation is once again explored in the analysis. Where the pre-analysis offered an almost taxonomic insight into the topic, the analysis will instead look at different tools, implementations, and methods available, in order to make an informed decision about as to how to actually implement an application to test the problem.

Lastly, the analysis investigates the topic of narrative analysis. Related to the previous discussions on narrative, this section regards another side of the topic, specifically the testability of the problem formulation. The overall aim of the section is to propose a method for testing exactly to what extend the users managed to perceive and interpret the story.

4.1 Narrative Architecture

Going back to the original motivation, of the project, it is worth pointing that players do not necessarily play *Diablo* for the story. As mentioned previously, the various mob bosses that players encounter over and over again, end up not as narrative elements[17], but rather as goals. In this scenario, the function of the narrative, as described by Gonzola, becomes merely to introduce and regulate the rules, and goals of the system. This does however not mean that creating a structured narrative becomes meaningless;

[17] It would be silly to tell a story in which the same group of people defeats the same villain over and over - but hey, is that not how must Saturday morning cartoons from the 80s worked?

merely that it is important to evaluate what functionality the authored narrative provides the game world with.

Henry Jenkins distinguishes between traditionally told narratives, and spatial stories, pointing out that video games may typically be the latter. He argues that most designers seem to focus on allowing players to move around in what he calls;

> *"narratively-compelling spaces"*. (Jenkins, 2003 p. 4)

This is not completely dissimilar, he argues, to traditional Japanese scroll paintings, but also classic myths and stories such as *the Illiad* and the stories of J.R.R. Tolkien, which are about spatiality as much as they are about the heroes' inner struggles.

Jenkins also introduces a numbers of different types of concepts, all of which relate to the concept of narratives in video games in one way or another.

His first concept, *Evocative spaces* suggests the creation of spaces which,

> *"[…] draw upon our previously exisiting narrative competencies."* (Jenkins, 2003 p. 6)

Described as a technique used traditionally in amusement park attractions and rides (Haunted Mansions, for instance), they have been crafted to emotions of previously encountered experiences.

This is a phenomenon also present in video games, where many games draw upon existing literature and movies to produce their universes. Traditional gangster and mafia movies (represented by titles such as *Scarface* and *Godfather*) have openly inspired video games, such as *Mafia* (2002, GOD) but also the games in the *Grand Theft Auto* franchise. As such, it may not be entirely possible to appreciate the stories and worlds of each game, without pre-existing knowledge of the genre as a whole.

Perhaps more generally *evocative spaces* also allow designers to create spaces connected to pre-existing universes and franchises. *Disney World* (or even *Lego Land*) exists on the basis and knowledge of exactly what *Disney* and *Lego* are. Similarly, games have been made which coexists with the universes of movies and even other games, despite telling a completely different story. An example of such a game is Westwood's *Blade Runner*, which is placed in the universe of the movie of the same name. The story told is however different from that of the movie, but as the game revolves around a number of the same locations, the player gets to re-experience some of the emotions he might have experienced upon first viewing the movie.

Secondly, he points out how narratives in video games can be enacted, rather than just viewed, read or watched like a traditional narrative. This idea is not completely different to some of the previously mentioned simulation theory by Gonzola (Gonzalo, 2003). although with one crucial difference. Gonzola claims that with video games being simulational, the player cannot possibly be experiencing a narrative. This is contrary to Jenkins' view in which the player becomes capable of enacting the story (Jenkins, 2003 p. 6).

Jenkins furthermore proposes the term micro narratives to describe narratives which take place on a local level within game worlds (p. 7). This is comparable to the *story-world* structure introduced in chapter 2.3, where the player may possibly traverse several different locations, experiencing several local events which are not necessarily related.

Some form of narrative may also be embedded into the game world. Jenkins' perhaps most obvious example relies on the notion of a classic detective story, in which two narratives exist semi-independent of each other. On one level, the player gets to explore the game world (being the enacted story), perhaps even freely and in any order he likes, and on a deeper level he gets to experience an authored embedded narrative which remains the same, for the player to discover. Perhaps the most famous example of this (as pointed out by both Ryan (Ryan, 2001 p. 253) and Jenkins (Jenkins, 2003 p. 9)) is Myst, where the player must uncover an ancient story through riddle solving.

Perhaps the most important point to take away from Jenkins' dissection of classic video game architecture and narrative is the importance of video games space itself. As it is suggested, environments can tell part of the presented narrative, by relying on the user's previous and existing knowledge regarding the game and knowledge of genre conventions.

A choice quote by Marie Laurie Ryan reads,

> *"When readers are really interested in "what happens" they do not want to find unnecessary obstacles thrown in their way."* (Ryan, 2004 p. 350)

It seems that the balance act of providing narrative in a video game is not an easy one. Described briefly in the introduction of the report, the narrative paradox deals with, to quote Ara Shirinan, developer of NightSky and The Red Star;

> *"The uneasy merging between narratives and gameplay."* (Shirinian, 2010)

This is a topic widely explored, and not a debate which will occur here. One academic point, as presented by Rune Klevjer may prove interesting though. He argues that cut-scenes, the way they are included in most modern games, can serve as representational events, not of things that happened, but of things to come. Here, the cut scene becomes a medium of *pre-telling* rather than re-telling. Moreover brief cut scenes can create breaks from the game play, and create a rhythm. Thus, they may become integral to the computer game experience, creating an almost oscillating effect between game play and cut scenes. (Klevjer, 2002)

A classic example of this can be experienced in a game like *Uncharted 2*, in which cut scenes are frequent, and typically no longer than a minute in length. Aside from giving the player a break,

4.2 Procedural Content Generation Part II

The aim of this section is to provide an extensive overview of a number of different techniques and algorithms used to procedurally generate content in video games, alongside with the academia supporting this work. The bulk of this section is based on work done previous semester on the same topic.

4.2.1 Existing Academia and Implementation

Although related work is typically presented at the beginning of a report or paper, this section is dedicated to exactly this. This is due to a number of reasons; as mentioned in the beginning of the report, the amount of academic work in regards to procedural content generation, especially related to video games, remains rather limited (but as this section will show, not non-existent). This can be sharply contrasted with the actual video games industry where this technology has been around for more than 30 years.

As such, in order to build up a strong theoretical section, it seems necessary to also include and consider the work done outside of the academic sphere.

Spelunky, and *Infinite Mario*[18] are two examples of side-scrolling games, which both use some form of procedural content generation in order to create the worlds which the players must traverse.

[18] http://www.mojang.com/notch/mario/ (retrieved 12/3/2010) – this free version of the game can be played here.

Infinite Mario is an open-source version of Nintendo's Super Mario Bros, in which the player gets to explore a randomly generated new level every time the game is run. Utilizing a *generate-and-test* algorithm (as described in chapter 1), the game generates which always remain possible to complete. What seems more interesting however, in this context, is the work done based upon this version of the game.

Spelunky, while perhaps less interesting academically, is worth briefly investigating simply as it seems more closely related to the M*etroidvania* genre than the other aforementioned work. As outlined in the pre-analysis, *Spelunky* tasks the player to travel through a number of randomly generated caverns in order to find treasure, and perhaps occasionally, rescue a damsel in distress. Where *Infinite Mario* I game oriented around dexterity, and dexterous challenges, Spelunky is however non-linear and is a game which rewards exploration.

On the more academic side of things, some efforts have also been made in regards to procedural generation of level design, also in 2 dimensions.

Kate Compton and Michael Mateas (Compton, 2006) have suggested a hierarchy for generation of levels in platform games. Distinguishing between individual components, patterns, cell structures, they argue that it is possible to build a set of interesting levels using the aforementioned elements, using a rhythmic structure. In their paper, the individual components constitute elements such as platforms or spikes which the player must avoid. Patterns are the collection of individual components, in any number of ways, either basic or more complex. A cell can subsequently be considered a set of patterns, which may be connected to other cells, either linearly or non-linearly. As such, patterns are typically linear, with the overall cell structure dictating specifically how linear the overall system is.

Whereas their implementation remains relatively limited, the consideration of dividing level structure into numerous hierarchical levels may be a sound one. If anything, it makes it possible to create separate focuses on the individual tasks and challenges the player must face, and the overall structure of the scenario which the player is participating in.

Smith, Mateas et al also suggests in their paper, *Rhythm-Based Level Generation for 2D platformers* (Smith, 2009), a grammar based approach to automatically generate levels. By generating first a rhythm, and then subsequently utilizing a "Geometry Grammar" to generate world geometry, they claim to be able to be capable of building a large number of interesting and varied levels. The rhythm is generated by measuring the player's movements in the game, respectively his jumping and moving.

Lastly, it seems of interest to cover to the Charbitat project, which has been conceived by Nitsche et al (Nitsche, 2006) (Ashmore, 2007). The project's web-page describes the mechanics of Charbitat with the following;

> *"As the player controls the main character of Charbitat, the game traces her behavior and translates those qualities into seed values for future space generation. Based on these seed values, more game space is generated as the player proceeds on her quest. Players generate a coherent game world as they explore and interact with it."* (Nitsche, 2007)

Perhaps more interesting however, is the fact the some work has actually been attempted at creating the first steps toward a methodology for creating a simple quest structure in their procedurally generated world. Their approach is however based on a simple key-and-lock structure, which procedurally generates new objectives and opens up new areas as the player proceeds.

This part of their project is not completely dissimilar to the aim of what is being attempted with this project, but one key difference exists. The Charbitat project uses a simple quest structure as a method of progressing, but the authors have done little to consider any narrative means, techniques or even side-effects which may occur when presenting a narrative in a procedurally generated world (Ashmore, 2007).

4.2.2 Potential algorithms

While in the process of determining a useful approach and method for procedurally generating worlds, it seems apt to briefly consider some of the ideas presented originally in the pre-analysis, perhaps specifically surrounding the ontogenetic and teleological approaches.

As previously outlined, the idea behind ontogenetic algorithms is to generate content through, what could possibly be labeled approximated algorithms. Instead of simulating real system, ontogenetic algorithms tend to skip a large number of steps in order to achieve the desired results. Teleological algorithms, on the other hand are meant to directly simulate the desired effect, and as such, provide a much more realistic result. The downside to these algorithms is that they may also end up producing completely unpredictable and thus unreliable results. Table B attempts to categorize a number of the previously mentioned approaches within these two categories.

TABLE B | VARIOUS ALGORITHMS, CATEGORIZED AS EITHER TELEOLOGICAL OR ONTOGENETIC

Teleological Algorithms	Ontogenetic Algorithms
Fire Propagation	(Grammar/rhythm) based platform game (Smith, 2009); also L-Systems (Lindemayer, 1996)
Fluid Dynamics	Dungeon Generation (*Rogue*; *Spelunky*)
Weather Simulation[19]	Procedural Platform Level Generation (Compton, 2006) ;also Infinite Mario

From witnessing table B, it is evident how ontogenetic algorithms simply are more suitable for generation of worlds and spaces in various capacities. Perhaps, in this context, it is worth considering the work done by A. Lindemayer, on the topic of Lindemayer systems (or L-Systems) (Lindemayer, 1996). Originally devised as a method of representing and modeling plant growth through algorithms, these systems, like the work done by (Smith, 2009) utilize grammars to represent the plants, or flowers.

However, as one will discover when exploring the topic, its usage is far from limited to modeling plant growth. Road networks have been grown (Delay, 2007), architecture (Müller, et al., 2006), (Parish, et al., 2001) and (Hidalgo, et al., 2008), just as some games (Furmanski, 2005) (Kremers, et al., 2009) have made use of the algorithms for different purposes. Therefore, it definitely seems that L-Systems, given their wide applicability, may be a suitable fit as a tool for generating architecture and video game spaces.

[19] Although depending how advanced this process is, this process could also be ontogenetic. Some video games contain more or less dynamic weather cycles (case in point: *Morrowind* and *Left 4 Dead 2*), but these merely change the light settings of the game, whilst playing additional sound effects while at the same displaying the game's particle system.

4.2.3 Grammars and L-Systems

The remainder of this section investigates the theory and idea behind L-Systems, and a number of similar grammars used to generate models, in all kinds and types of software.

Several different types of notation for and interpretations of L-Systems have been suggested, but throughout this report, the original definitions as provided by (Lindemayer, 1996) will be used.

The most basic L-System, the (0)L-System (also called the Context-Free L-System) can be defined through the following tuple;

$$G = (V, \omega, P)$$

In this context, G represents the system; V represents the alphabet from which the system is generated. ω represents the initial condition of the system (typically represented by a number of letters from the alphabet), and lastly, P represents the production rules.

To exemplify what this essentially means, the following example can be considered.

A system consists of the alphabet $V = \{A, B, C, D\}$, the following initial condition, $\omega = \{ABC\}$ and the following set of rules;

$$\boldsymbol{A} \rightarrow B$$
$$\boldsymbol{B} \rightarrow ABC$$

If i denotes the iterations in the system, the following set of strings can be constructed, if run through three iterations;

$$i = 0: ABC$$
$$i = 1: BABCC$$
$$i = 2: ABCBABCCC$$
$$i = 3: BABCCABCBABCCCC$$

It is also possible to have Context-Sensitive L-Systems (also (1)L-System) in which it is possible to create conditional rules. A rule in a (1)L-System can be written as $(\eta\iota, \alpha, \eta\tau, \chi)$ which, upon being iterated will result in $\eta\iota < \alpha > \eta\tau \rightarrow \chi$.

To briefly illustrate this, consider the previous example, except replaced with the following rules;

$$\boldsymbol{A} \rightarrow B$$

$$A < \boldsymbol{B} > C \rightarrow ABC$$

Again, i denotes the iterations, resulting in the following string;

$$i = 0: ABC$$
$$i = 1: BABCC$$
$$i = 3: BBABCCC$$
$$i = 4: BBBABCCCC$$

Where the first iteration is identical in each example, the latter quickly deviates given its conditional nature. ***B*** will only be rewritten into ***ABC*** if it is preceded by an ***A***, and followed by a ***B***.

The pre-analysis outlined the difference between deterministic and stochastic algorithms, and unsurprisingly, L-Systems can also posses either capability. This is useful when modeling a large number of plants at once, whilst wanting them look similar but not identical.

A stochastic L-System is defined as $\mathrm{G}_{\pi} = (V, P, \omega, \pi)$

The newly added component π is denoted as the probability distribution, which goes from 0 to 100%. Any rule which uses the same predecessor (i.e. the same letter) must add up to the sum of 100% (or 1) in order to be valid. Consider example 1, but with following modifications to the rules;

$$A \overset{50}{\rightarrow} B$$
$$A \overset{50}{\rightarrow} AA$$

In this case, ***A*** has a 50% chance of either being rewritten into a ***B*** and equally a 50% chance of being rewritten into ***AA*** at each iteration step.

Lastly, we have parametric L-Systems, defined as;

$$r = (V, \sum, \omega, P)$$

By adding parameters to the system, and to the letters in the alphabet, it is possible to put control the growth of the L-System. The letter $\sum$ signifies the set of parameters.

It is also possible to combine these different L-Systems, so one L-System can be both parametric and also stochastic probabilities at the same time.

Most commonly, L-Systems use Turtle Interpretation to draw and model the plants (Prusinkiewicz, et al., 1985). In order to illustrate exactly how L-Systems can function visually, it is possible to consider the following example;

$$V = \{F, +, -\}$$

$$\omega = \{F\}$$

And the following rule:

$$F \rightarrow F[-F][+F]$$

F draws a line in the current direction, whereas + and – rotates the direction by 30 degrees. “[“ saves the current position and direction of the system, whereas “]” returns to the last saved position and direction.

Once again, with i denoting the current iteration, we generate the following strings;

$$\begin{aligned} &i = 0: F \\ &i = 1: F[-F][+F] \\ &i = 2: F[-F][+F]\big[-F[-F][+F]\big]\big[+F[-F][+F]\big] \end{aligned}$$

If this is however drawn as a picture, a simple tree structure will be displayed. Figure 8 display the model from iteration 0 to 3.

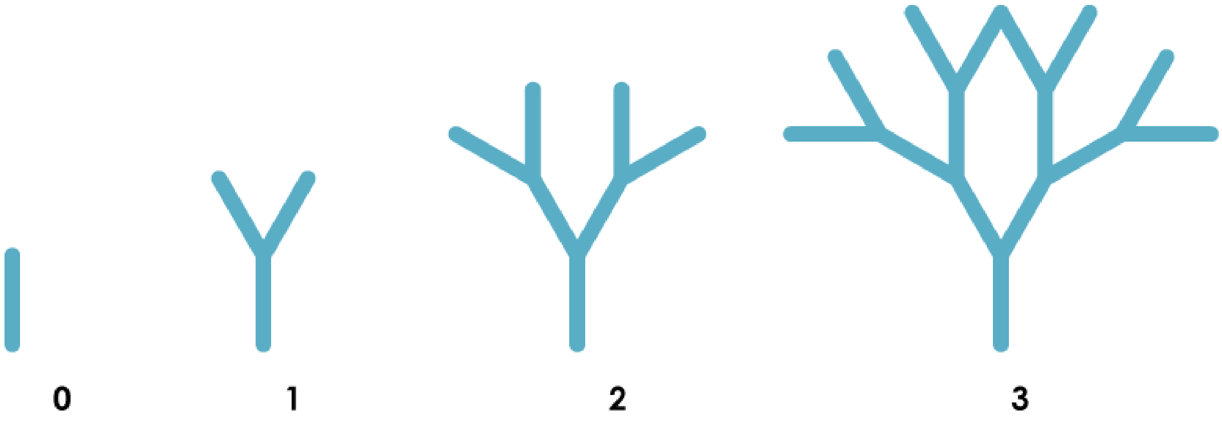


FIGURE 8 | THE L-SYSTEM, AS DESCRIBED IN THE TEXT

4.2.4 Alternative Applications Of L-Systems and Grammars

With a number of basic concepts of L-Systems covered, it seems prudent to explore in-depth a number of the previously mentioned alternative implementations.

Some of the criticism aimed towards L-Systems, particularly with the viewpoint that they may have alternative architectural applications, stems from the fact that they have been made to simulate plant growth, something which typically happens in open spaces. This is very much unlike building construction which is typically forced to have much tighter and more restrictive spatial constraints and requirements (Wonka, 2003).

However, some attempts at creating spatially aware (both externally and internally) L-Systems do exist, as both (Prusinkiewicz, 1994) and (Mech, 1996) illustrates. More interesting however, may be the solution proposed by Parish, Y.I.H. and Müller (Parish, et al., 2001), Their implementation creates a road network, which is not only spatially aware, but also contains no dead ends.

To achieve this, they implemented two new algorithms, under the names of 'Global Goals' and 'Local Constraints'. The 'Global Goals' algorithm is perhaps best described as a steering algorithm for the system. In order to create a road network using their implementation, it is necessary to provide not only a map containing topological features (for the road network to be "physically" placed on) but also a population density map which the Global goals algorithm is governed by. When the road network is then generated, the global goals function adjusts new branching roads in the right direction, toward the densest areas.

The local contraints function uses the topological map to dictate where the model can and can not go. If a suggested road extension crosses into the water or onto another uncrossable area (park, forrest etc), it also removed. The second function of the local constraints is to ensure that the road network does not contain any dead ends. This is done by connecting nodes if they are close enough to each other. If no node is present, a new one is created on the edge. Using a parametric L-System, they ensure that roads which have branched or connected to are grown further.

The L-System suggested by Parish et al. not only solves some of the problems otherwise inherint in L-Systems, but it may also have other usages. Chris Delay, a developer at Introversion Software have, based on the papers released by Parish et al. has not only replicated the structures, but also used the same algorithms, and different rule set, to generate building interiors (Delay, 2007).

Perhaps a short mention of a completely different type of L-System should also be given here. The FL-System. This implementation, suggested by (arvie, 2005) replaces the arguably archaic turtle interpretation with an object oriented approach. The FL-System is used to generate geometry in virtual spaces, and in addition to the classic symbols in the L-System, this system also adds rotation, scaling and translation as terminal functions.

4.1 Interpreting Video Game Stories

As the problem statement outlines, the aim of this project is to test exactly to what extent, it is possible to perceive and interpret a story in a procedurally generated world. The previous section covered various ideas and theories concerning stories in video games, but in order to achieve the goal of the project, it seems necessary to also be able to investigate exactly what and how the users' perceive stories in games.

Although not conceived as a method for analyzing video game narratives, the theories and models of narrative analysis may here be helpful to relate to.

Catherine Kohler Riessman attempts to define four different models of narrative analysis, all designed to extract certain knowledge from oral narratives (Riessman, 1993). She argues that by analyzing narratives through interview transcripts, it becomes possible to identify themes, create meaning and gain new knowledge. It also argued that, as different modes of analysis have different purposes, it also becomes possible to combine the different types of analysis.

Thematic Analysis, her first type, focuses on "what" is being said, striving to identify, as the name implies, the overall themes and meaning of the narratives. As she writes,

> *"The thematic approach is useful for theoriising across a number of cases, finding common thematic elements across research participants and the events they report."* (Riessman, 1993 p. 3)

On the other hand, this type of analysis does not study "how" things are being said, and as such the language itself is merely a "resource" (p. 3).

Structural Analysis tries to rectify this, by exploring the language and utterances of the interviewed, thus extending its referential nature. Riessman outlines different basic components in a narrative, which becomes possible to identify through this type of analysis, although all are not always present.

> *"The abstract (summary and/or point of the story); orientation (to time, place, characters, and situation); complicating action (the event sequence, or plot, usually with a crisis and turning point); evaluation (where the narrator steps back from the action to comment on meaning and communicate emotion – the soul of the narrative); resolution (the outcome of the plot); and a coda (ending of the story and bringing action back to the present)"* (Riessman, 1993 p. 3)

The shortcoming of structural analysis stems from the fact that it may only be used for a smaller numbers of interviews, as it is rather resource heavy.

Interactional Analysis can be seen as a (compared to the two previous types of analysis), as a collaborative form of analysis where the interviewee and the interviewed build up the narrative together. As Riesmann writes;

> *"Interest shifts to storytelling as a process of con-construction, where teller and listener create meaning collaboratively."* (Riessman, 1993 p. 3)

However, as also stated, both thematic and structural content may also be considered. This type of analysis is commonly seen in e.g. court cases where more than one part is actively participating in the conversation.

It is not unusual, both in structural and interactional analysis to indicate different types of utterance (or the lack thereof, such as pauses) in the transcripts

Performative analysis, the last type of analysis described by Riessman is built upon the ideas and theories surrounding the previous type of methodology, namely interactional analysis. Including a performative dimension into the analysis, a new number of new elements can suddenly become subject to investigation;

> *"actors allowed on stage in an oral narrative (e.g. characters and their positioning in a story, including narrator/protagonist); settings (the conditions of performance and setting of the story performed); the enactment of dialogue between characters (reported speech); and audience response."* (Riessman, 1993 p. 4)

Riessman argues that these studies are useful, for instance in cases of identity construction, through their involvement with the audience.

While all four modes of analysis are centered on oral performances, it does seem as if the two first may also be applicable in some form in a written context.

If one makes use of either the performative or especially the interactive analysis, it may become necessary to lead the user on. This is not necessarily a good thing in this project given that it may lead the user to unnecessary re-interpretation, something which may end up complicating the told narrative.

It seems however that in order to fully create meaning from retelling of video game narratives, it may also be necessary to briefly consider traditional video game discourse.

Perhaps Juul, as outlined previously in the pre-analysis, may unintentionally have had a point in regards to the retelling of video game narratives. If done completely in detail, retelling the narrative may not only become mind-numbingly dull, it may also produce a narrative with a incredible amount of superfluous detail.

In this context, it may become useful to turn to concepts of *playscripts,* as defined by Aldred et al. (Aldred, 2007). They suggest that it is possible to divide the experience of video games into 7 different types of scripts, all of which may be inter- or cross related to each other. These are dramatic scripts, interface scripts, interactive scripts, ludic scripts, implementation scripts, cognitive-narratological scripts and meta-gaming scripts.

Dramatic scripts are described as the scripts which provide context to the game. Using the example provided originally in section 2.2 of the report, this type of scripts is what differentiates Tomb Raider from any other third person action game, with a male protagonist.

The context expanded here is both in regard to characters, but also in regards to backdrop, and may be elaborated upon through the use of both supplementary material but may also be expanded through the actions which the player performs, while playing the game.

As Aldred et al. writes,

> *”The ‘world-building’ elements of the dramatic script include plot, character, dialogue, ideas, atmosphere, mise-en-scène, and various ‘synthespians’.”* (Aldred, 2007 p. 206)

This is coincidentally not completely dissimilar to the point emphasized by Jenkins previously, concerning the use of evocative narratives.

The *Interface Scripts* of a game consist of the scripts which the user must interact with in order to perform actions within the game. This is done through, for instance, the arrow keys, the mouse or perhaps a different type of game controller, and allows the player to move through the world, or manipulate other types of objects within the game world.

Ludic Scripts are defined upon the work done by Gonzola (Gonzalo, 2003), and consists of the action performed in the game in an attempt to win, or receive some amount of gain. This is perhaps easiest perceived in abstract games such a Football or *Tetris*, which consist of certain rules and objectives for the players. While it is not possible to win a game of a Tetris, the player can increase his score by creating rows of lines. At the same time, he has to avoid having the blocks reach the top of the screen, as this will make him lose the game. These would constitute ludic elements, making up the ludic scripts of the game.

Aldred et al. describe the interactive scripts as the following;

> *"The "interactive script" is the sum result of interface scripts (user actions) on dramatic and ludic scripts within the game world"* (Aldred, 2007 p. 206)

These scripts essentially create a feedback loop between the player and the computer game world. Using the same example of Tetris as before, the game will respond to the player's actions, and if the player manages to put together a line of blocks, the game will respond by rewarding him with points while removing the newly create line.

The line between the *interactive scripts* and perhaps the *ludic* is considerably unclear, although the inclusion of *dramatic scripts* into this category seems to suggest that some interplay may occur here, i.e. an action performed in the ludic script may have an effect on the dramatic script, and vice versa.

Implementation Scripts are different from the previously outlined scripts, in that they remain the only non-metaphoric script of the bunch. Simply put, the implementation scripts are the scripts from which the game is built, whether written in C++, C#, using DirectX, OpenGL or other APIs and frameworks (Aldred, 2007).

Cognitive-Narratological scripts have a number of functions. For one, they allow players familiar with one type of game to instantly sit down and play any game closely related to it, be it a sequel or the same genre. Again, a comparison to Jenkins' use of the term Evocative narratives can be made. However, these scripts are not limited to merely this function, as it is argued that these scripts allow players to;

> *"These formative scripts help us to structure narratives in memory, and are called upon whenever we listen to or tell stories"* (Aldred, 2007 p. 207)

Lastly, *metagaming scripts* detail which may be related to the gaming experience, but happen to have some relation to it. Perhaps the best example of this behavior comes when breaking out of behavior in a multiplayer game, either discussing the game or even other issues (Aldred, 2007).

All the scripts may also be connected, interrelated or may even be competing against each other.

The concepts of *Playscripts* may prove useful when looking to provide some analysis of the players' interpretation of a video game story. More specifically, it seems that it seems possible to distinguish between various elements within a game using these scripts, some of which may be relevant to look for when analyzing user transcripts whereas others may effectively be discarded. Many games contain generic non-player characters which the player may encounter throughout his game experience, for instance in the form of enemies. Whereas each enemy may not have a function in regards to the dramatic script, they may not be unrelated to it (for instance, in the case of *Half-Life*, the player's enemies are either sent by a secret government branch to silence the player, or are aliens which were sent there as a result of a scientific mishap).

As such, enemy encounters typically fall within the context of ludic scripts, but as the context is provided by the backdrop (and dramatic script) When analyzing transcripts, it would be expected that players may at least be able to identify their reason for being there.

Some scripts are less important, in the context of this project. Implementation scripts, or interface scripts are largely unrelated to any authored narrative, and consideration of the meta-gaming script may help to weed out useless or irrelevant information from any transcript or story summary.

Perhaps, it is also necessary to give some consideration between the plot and fabula, and how these coexist in video games.

In traditional literature, fabula is regarded as the actions which take place in a story. This includes everything, even events which may not necessarily be written down. Plot, on the other hand, is the sequencing and ordering, and even the decision of which events to tell. It does not necessarily make sense to talk about plot in the context of a video game, given the temporal nature of most video games, where the order of presented events is typically identical to the fabula[20] (Felluga, 2002).

Where the plot may however become important is when the players are tasked to retell a story. Here, they may re-arrange the temporal order of things in order to ensure that the presented story makes sense. This is not unlike the point presented above concerning enemy encounters in the game of Half-Life.

[20] Some exceptions do exist however. The first half of the game *Uncharted 2: Drake's Fortune* is actually played out in a flashback, just as the first episode *Sam and Max Season 3* plays around with the temporal order of things throughout most of the game.

Generally it seems that some considerations must be put towards the differences between the action which the player has to go through, and the retelling which is bound to be different. Going back and using the monomyth as base (presented in chapter 2.3.1), it seems that it may be possible to divide the nodes of the model into two different categories, namely major and minor events. In particular one node entitled, "Test, allies, enemies" seem to stand out. Video games, especially action-oriented ones tend to operate primarily within this sphere. More generally, it seems that this node, being a major event in the grand scheme, can be divided into a number of smaller events for the player to overcome. This is where generic enemy or other non-player character encounters would typically fit in. It is also where things such as bonus pick ups would figure into the game, as a aid for the player to complete his goal. These events are all re-occurring, and as such, it may not be necessary to identify this types of event more than once.

On the other hand, certain events typically only occur once, be it the supreme ordeal (which in video games, is typically related to fighting the antagonist) or introductionary events. Based on this, it is possible to suggest a dichotomy, dividing various events into Major (or dramatic) and minor events.

TABLE C | THE VARIOUS MINOR AND MAJOR EVENTS

Major Events (Dramatic)	Minor Events
Unique Dialogue	Generic Enemy minion / NPC Encounters
Boss fights	Bonus Pick Ups

The ideas and concepts presented through narrative analysis may similarly be worth considering in the context of the project, given the possibility of identifying themes and creating meaning from narratives. By having players retell their experiences, and subsequently analyzing this input, potentially categorizing the events, it should be possible to decide to what extent users were capable of interpreting the narrative within the game.

4.2 System and Project Requirements

The past number of sections in the pre-analysis and analysis details a number of theories, ideas and concepts, all of which should potentially be considered when designing a procedurally generated world containing a coherent and linear narrative. Secondly, the analysis also covers some ideas and methods for interpreting narratives, and creating meaning from them. In order to answer the initial problem, it is necessary to consider all of these elements both in the design of the test, and as a result, also in the design of the application.

4.2.1 Narrative Requirements

The tools of narrative analysis seem like an ideal match for this project. They allow researchers to create sense and meaning out of people's personal stories and experiences. However, written or told narratives are not necessarily completely faithful representations of the past. Thus, it seems necessary to help the users along the way, making use of narrative theory and experiences familiar to the user in order to ensure that users are capable of retelling their experiences as precisely as possible.

That is not to say that the goal of any narrative in a procedurally generated world is to contain as little ambiguity as possible; rather that in order to answer the problem formulation of this project, it is necessary to produce a narrative which is as clear and obvious as possible, in order to test and see if players generate meaning from the actual story, or create their own.

- **Familiarity**
  The presented narrative must contain some notion of familiarity, in order to increase the player's ability to not recognize, but also retell the events of the game. These ideas tie in both with the theory presented by Jenkins, but also by Aldred et al.
- **Monomythical Structure**
  Locations, characters and various other mise-en scene are typically relatable elements, where users' can draw upon their experiences in order to build an idea of the narrative in their mind. However, the monomythical structure (or The Hero's Journey) is an example of an almost prototypical structure which should be relatable and recognizable to most people as well. As such, it makes sense that

this project's attempt to deliver a narrative in a procedurally generated random world makes use of this structure.

- **Transcripts**
  In order to adequately interpret the players' experience with the game, it is necessary for them to deliver and present some sort of retelling of the story, be it oral or in a written form. Only then will it be possible to investigate whether or not telling narratives in completely procedurally generated worlds is possible.
- **Events**
  Division of events into major and minor is a necessity when attempting to create meaning from the players' experience. This allows the grouping of events into categories, something which will make interpretation easier.
- **Spatial Layout**
  Related both to world building requirements, and to the narrative requirements, some thought must be given the potential spatial layout of the world. The typical *Metroidvania* structure is suggested as the layout to strive for. This type of layout has the advantage of being non-linear, allowing for the option of exploring, while at the same time having the capability of presenting a linear narrative. As such, it seems ideal for multiple type of players.

4.2.2 World Building requirements

While the l-system grammar may be ideal for generating models of plants and flowers, study of its usage reveals large number of different examples that all make use of L-System, or L-System inspired grammar for a wide variety of uses, especially in generation of virtual architecture.

The strength of such grammar for the generation of virtual spaces and architecture makes it a suitable fit for generation of the proposed non-linear game world structure. However, there are a set of requirements that this grammar must fulfill.

- **It must be spatially aware.**
  This entails a system, similar to the local constraints function described by (Parish, et al., 2001). Like real architecture, video games structures can be faced with equally restrictive spatial requirements. Therefore, in the final system, spatial awareness is a requirement in order to ensure
- **Classic Properties**
  Classic L-Systems, depending on the active definition, contain a large number of different properties. They can be stochastic, parametric, context-less or context-

sensitive and so on. Each of these properties should be considered when constructing the world building grammar. The stochastic properties of the grammar can ensure that the world appears random and different every time the game is loaded, the parametric properties can ensure further spatial restrictions, perhaps in lieu with what Parish et al. achieved with the road network.

- **Generate-And-Build vs. Constructive**
  Considering the various dichotomies presented by Togelius in the pre-analysis, one must decide whether to create a constructive algorithm, or one which builds the world, and subsequently tests the fitness of the world. A constructive world requires a foolproof algorithm in its place. Given the large amount of randomness necessarily to build an entire world, this may not necessarily be the best approach.
- **Online vs. Offline**
  A further distinction is made between online and offline created content. Discussing only the structure of the world, one of the early goals set in this report pertains to creating a new random world every time the game is loaded. As such, the structure should be generated when the player loads up the game, rather than pre-exist within the game files.

5 Design

The analysis outlined a number of requirements necessary to achieve and implement in order to answer the problem formulation. The aim of this section is to provide a design based on these requirements.

This chapter will deal with a number of different topics. First, the overall concept for the game is presented. This can be viewed as a sort of summary for the final design, putting each and every element together. Subsequent sections will detail various processes related to the narrative, the structure, and the game itself (that is, ludic considerations).

The final game has been entitled, "Castle of the Flying Panda" and is a non-linearly structured game with a linearly presented narrative. The game takes place inside an old castle, where the player, assuming the role a Wire-Fu[21] Hero is tasked to free the President's daughter, as she has been kidnapped by unknown mysterious forces. Throughout the game, he encounters both allies and enemies until he is finally faced with person who turns out to be behind the entire thing. As it turns out, it is his own brother.

While the game borrows from the structure of the *Metroidvania* games, most game-play mechanisms are borrowed from Super Mario Bros, though with the one crucial difference that the player is also capable of traversing tall vertical corridors.

5.1 Narrative Design

When designing the narrative, primarily two things were considered. Having already decided upon the decision of division into minor and major events, the next logical step was to decide how this would actually work, in combination with the structure of the world.

Second to that, it became prudent to decide and design the actual story itself, based on the lessons learned in the analysis chapter.

[21] This neologism will be explained in further detail in section 4.1.1

5.1.1 The Story

The design of the story is, as suggested in the requirement specification, heavily influenced by the monomythical proto-story structure. Eight dramatic events, each corresponding to a specific point on the monomyth were created. The following section aims to outline each point in the story, and will at the same time refer to its position on the monomyth.

- The first event simply introduces the backstory to the game. Some time in the near future, the president's daughter (of an unspecified country) is kidnapped, while attending a party at the playboy mansion. The player(in the shape of what is best described as half a panda, half a ninja), and his mentor is tasked to free her from the mysterious kidnappers. On the monomyth, it corresponds to the *call to adventure*.

- The second event is meant both as an introduction to the main character, the Panda Ninja, but also to serve as an introduction to the world of adventure, and as such corresponds to the point entitled *Crossing The Threshold*.

- After traversing the castle for a while, the player meets up with his mentor who has unfortunately also been captured by the mysterious kidnappers. This corresponds to the point entitled *Meeting the Mentor* on the monomyth. Typically, the protagonist meets his mentor before he enters the world of adventure, where the mentor must convince the protagonist to take the quest. In the case of this game, it is merely infered that some contact preceeded the game's actual events.

- The mentor sends Panda Ninja further into the castle, tasking him to find the spy they have planted inside the castle, the Undercover CIA Sous Chef, as he possibly posseses some knowledge of how to defeat the kidnappers. The monomyth presented by Nitsche (Nitsche, 2008) has one point entitled, *Allies, enemies, Test*. As argued previously, the majority of most video games are concerned with specifically this point, so for the sake of this project, it has been divided into three separate points, with the meeting with the Sous Chef entitled just *Allies.*

- As the player traverses the castle, he is going to encounter a number of hostile minions, which he can either choose to defeat or run past. These events are counted as minor events, and will therefore not count toward the total number of dramatic events.

- The Sous Chef tasks Panda Ninja with once and for all turning off these robots. The Panda Ninja must reach a place deep within the castle and turn off a console controlling all the robots. Only then will he be able to face off whomever is behind the entire thing. This corresponds to the test part of the monomyth structure.

- From here, the player must enter the evil kidnappers' *innermost cave*, and face off with the villain in a final showdown. It is revealed that the villain is in fact Panda Ninja's brother, making the last encounter extra difficult. This corresponds to the *supreme ordeal* on the monomyth.

- Afterwards, he has free access to go get the *reward*, and meet the President's Daughter, who, aside from initial nervousness, is thrilled to see Panda Ninja. The game ends with a closing comment, remarking that Panda Ninja and the President's Daughter return to civilization, and that the world is once again at peace.

Each event has also been plotted into Figure 9, displaying each event as they match the monomyth. For a full transcript of the manuscript, please refer to the appendix.

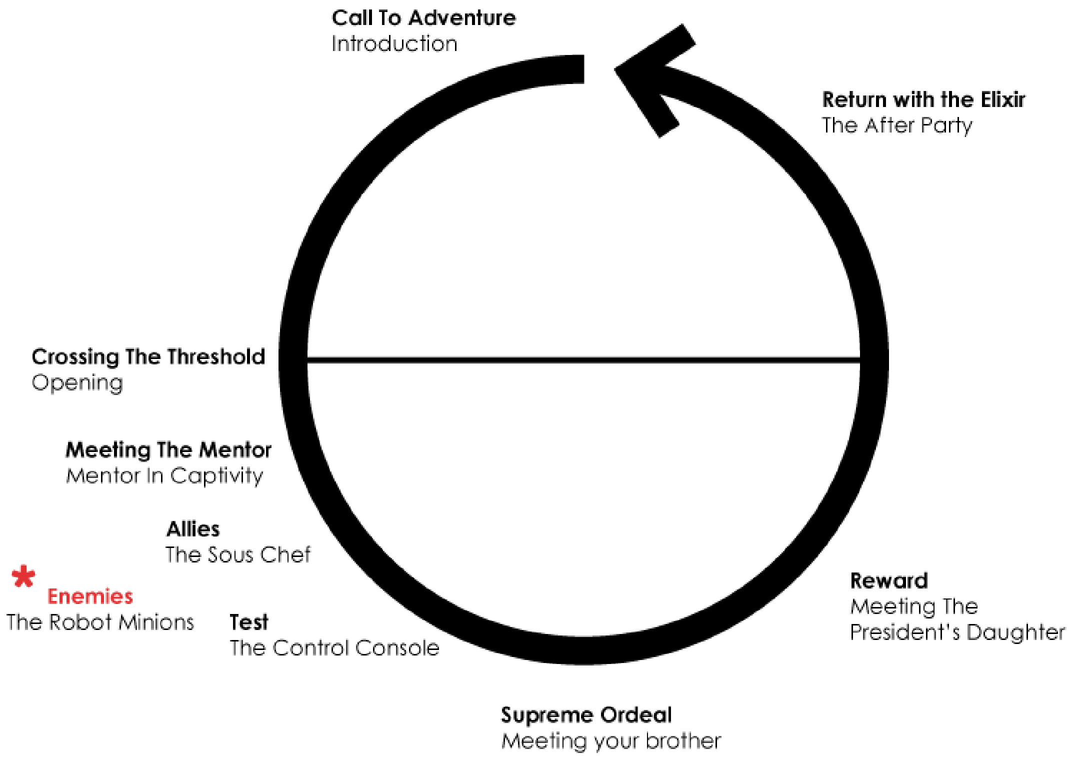


FIGURE 9 | THE MONOMYTH, ADAPTED TO THE GAME *CASTLE OF THE FLYING PANDA*. THE EVENT ENTITLED 'ENEMIES' IS A SPECIAL CASE, AS IT IS REGARDED AS AN EVENT WHICH WILL OCCUR MULTIPLE TIMES THROUGHOUT ANY PLAYSESSION, AND TYPICALLY DOES OCCUR MULTIPLE TIMES IN MOST VIDEO GAMES.

Several attempts at creating some familiarity for the player has been attempted. The Panda Ninja, and robot minions are inspired from traditional Chinese action/adventure movies, such as *The House of the Flying Daggers* and *Hidden Tiger, Crouching Dragon*[22], and also earlier Bruce Lee kung fu movies. In essence, this approach is inspired by Jenkins' evocative spaces.

The final encounter is also a parody on the movie Star Wars V, where it is revealed that Luke Skywalker, the protagonist, is indeed related to the man who is behind it all.

Having the President's daughter being introduced in the very last room of a very big castle, and only after defeating the arch villian is also a reference to the classic Mario games in which this was the pre-dominant structure[23].

One important point was to let the player feel as if he gets to enact the story, that is, he is not only going to be a random person partaking in the world, instead the actions he performs drive the narrative and the story forward. This is perhaps most evident at the

[22] These movies sometimes go under the moniker *Wire-Fu* movies seeing that the characters in the movies are often capable of completely defying gravity, and going everywhere that they please.
[23] Although unlike the Mario games, the princess is not in another castle.

'Test' where the player's actions effectively changes the power balance in the castle in favour of the good guys.

Many of the dramatic events within the game are similarly written to be narratives, not only of re-telling but also of pre-telling, as Klevjer emphasizes in his article. What this essentially means, is that they are set up to introduce the next goal, and the next event within the game, creating some anticipation for the player. The player's mentor tells Panda Ninja to seek out the undercover CIA sous chef, as he will know what to do. The idea is for the player to anticipate what happens next, perhaps invoking some curiousity as to what happens next.

Every cut scene is also kept relatively brief, with the game containing a total of 35 lines of dialogue, again to keep it in line with the ideas presented by Klevjer. The attempt is to create a rhythm between the intense action sequences, and some slower moments, where the player will be able to relax for a few seconds.

5.2 Adapting the Monomyth

With the authored narrative constructed, it seems only suitable that the next objective is to match both the dramatic and minor events with the *Metroidvania* structure.

Recalling the structure presented on page 21, and considering it as a graph, the dramatic events in the story can be directly mapped to the graph, with only a few considerations. The first couple of events must be located near the player's starting location. The encounter with the sous chef must be located around halfway into the graph, whereas the encounter with the villain and the reward must be mapped to the two vertices on the graph which are the furthest away from the player spawns in the beginning.

Distributing minor events is equally straight forward. As mentioned, these events occur multiple times, and as such, can occur at any vertex in the graph. Figure 10 proposes exactly how to transfer the monomyth structure to the Metroidvania structure.

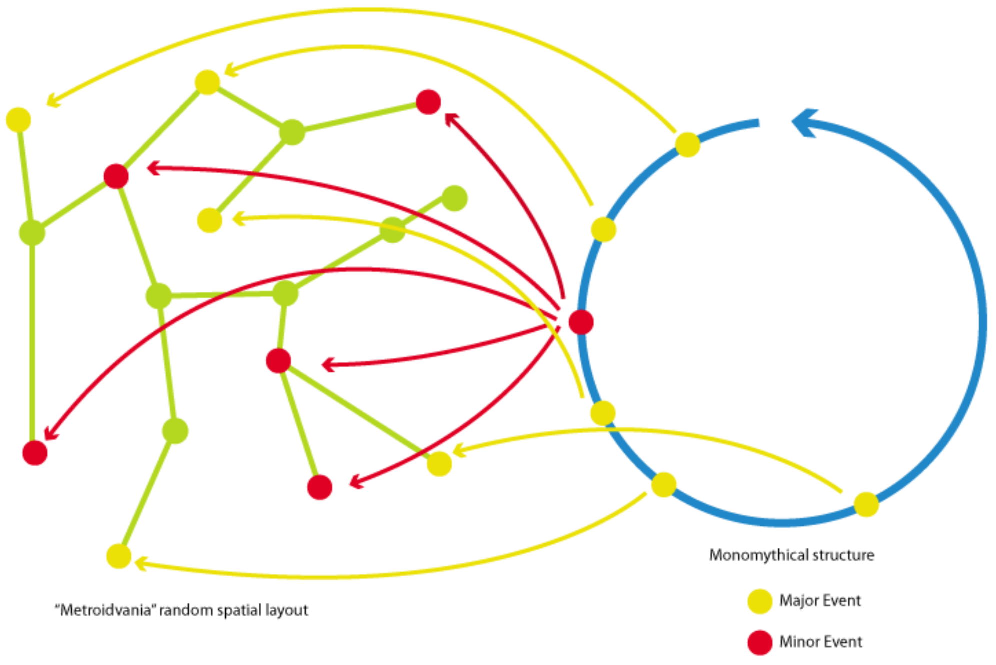


FIGURE 10 | THE TRANSFER FROM MONOMYTH TO METROIDVANIA

5.3 Ludic Considerations

The aim of this report is not to investigate new game-play techniques or create new game genres. Therefore, very little thought has been given in the research sections toward actual game design, but that does not mean it should be completely ignored.

One of the main ideas is to create and build a game which should feel immediately familiar to the player. As such, it therefore makes sense to build upon existing genre conventions. Metroidvania games are most commonly played out in only 2 dimensions[24], and as such, they do share a number of similarities with traditional platform games.

As the story draws some inspiration from *Super Mario Bros*, it makes sense that the ludic script does too. By allowing players to defeat enemies by jumping on them, the players are awarded with a sense of familiarity, and will not have to spend too much time thinking about how to play the game, allowing them to focus on the dramatic events.

[24] Although the specific type open world structure present in this type of game is starting to reach games with 3-dimensional exploration as well. Batman: Arkham Asylum is a game which can arguably be traced back to the early Metroid games, just as the newer Metroid games (in particular, the Prime series) features and rewards exploration as well.

Similarly the game draws inspiration from traditional wire-fu movies, and it makes sense to let the game be inspired by these. Instead of a traditional jumping mechanism, the players are capable of flying through the environments, if they wish, in an attempt to not create a link with the movies, but also to ease the navigation.

5.4 In Summary

The aim of this was section was to provide an idea of some of the thoughts going into the story. Some of the theories and models outlined previously (monomyth, enacting stories, *gameplay* of cutscenes) were applied in the design, in order to construct not only a coherent narrative, but also one which plays to the strengths of video games. With the final design decided, the next step is to move to the implementation.

6 Implementation

The aim of this chapter is to describe and present a number of things. The first section is dedicated to presenting the game, Castle Of The Flying Panda, as it exists in its current form, in as much detail as necessary. As the process of developing the game could be considered iterative, some considerations regarding this will consequently be presented in the same context.

Secondly some detail in regards to how the engine was developed will be featured. This part is divided into two parts, one presenting an overall system structure from which the game was developed, with the second half dedicated to detailing discussing two of the major components; namely the algorithm that generates the architecture and the algorithm which inserts and builds the various dramatic events within the game.

Some considerations regarding the implementation of the interactive and ludic elements will follow, and finally some thoughts regarding the implementation of the visual assets will be presented.

6.1 The Game

The game, *Castle Of The Flying Panda*, has been developed with the use of XNA 3.1, Microsoft's game development framework for hobbyists and other amateur developers, and was written in the language C#.

The game has the player traversing a big castle in search of a missing person. Throughout the game, the player is going to experience a number of encounters, both friendly and not-so-friendly. The lesser friendly ones can be disposed of, in the same way enemies are defeated in the classic *Super Mario Bros* and *Sonic the Hedgehog* games, by jumping on them. Any other collision with the player will on the hand make the player lose health. In vein with traditional Wire-Fu movies, the player is capable of not only jumping, but also flying at his own leisure. As the player travels through the maze of corridors and rooms, he must go through a number of dramatic events in order to progress the game and the story.

Figure 11 displays two screenshots from the game.

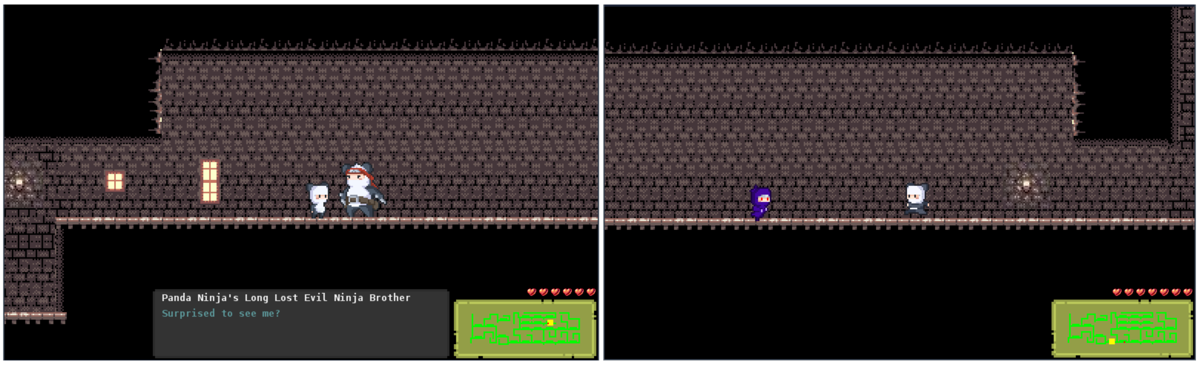

Panda Ninja's Long Lost Evil Ninja Brother
Surprised to see me?

FIGURE 11 | TWO SCREENSHOTS FROM THE FINAL GAME

The iterative approach to the development of this project meant a few things. As seen in Figure 11, the initial version of the game merely generated a set of rooms and corridors, with no game elements for the player to explore. The idea behind this version of the engine was simply to test the world creation algorithm, and make sure that the generated world was consistent.

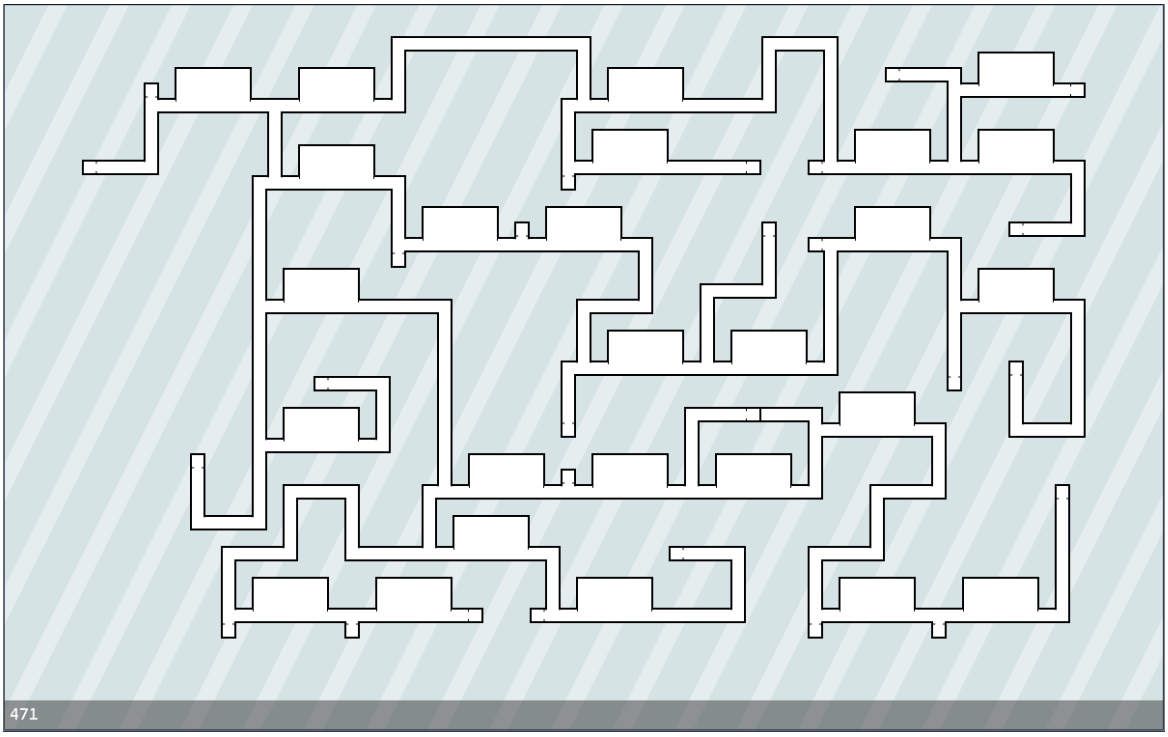


FIGURE 12 | THE OLD VERSION OF THE GAME, WHICH WAS ONLY CAPABLE OF GENERATING LAYOUTS

The second version of the game included basic navigational features of the game. The aim of this version was simply to ensure that navigation worked properly.

The third version of the game contained the event-engine (or *Eventgine* abbreviated), and was also the basis for a thorough pilot test. The aim of this test was to ensure that it is possible to play through the game and experience the story, without encountering unnecessary annoyances. Some shortcomings were reported, specifically in regards to minion encounters, where it was not particular clear when the player was hit.

The final version of the game included some adjustments to collision detection, corrections and modification of the existing dialogue, and additional visual feedback, to make the player feel more connected to the game world. Secondly, the ability to dispose of the enemy minions by jumping on them was also added.

The briefly iterative approach chosen for the development the game, came both as a consequence of this project being a split master thesis, but it was also done in the hope that the both the story and game experience could be built upon both narrative theory and with the help of participating users. The changes suggested to the game were however merely related to the ludic aspects of the game.

6.2 World Creation Algorithms

This section covers the implementation of the algorithms which are used to procedurally generate a random world.

6.2.1 System Structure

The system can be divided into a number of different major processes, all of which work together to build the world. Figure 13 displays the overarching structure of the system.

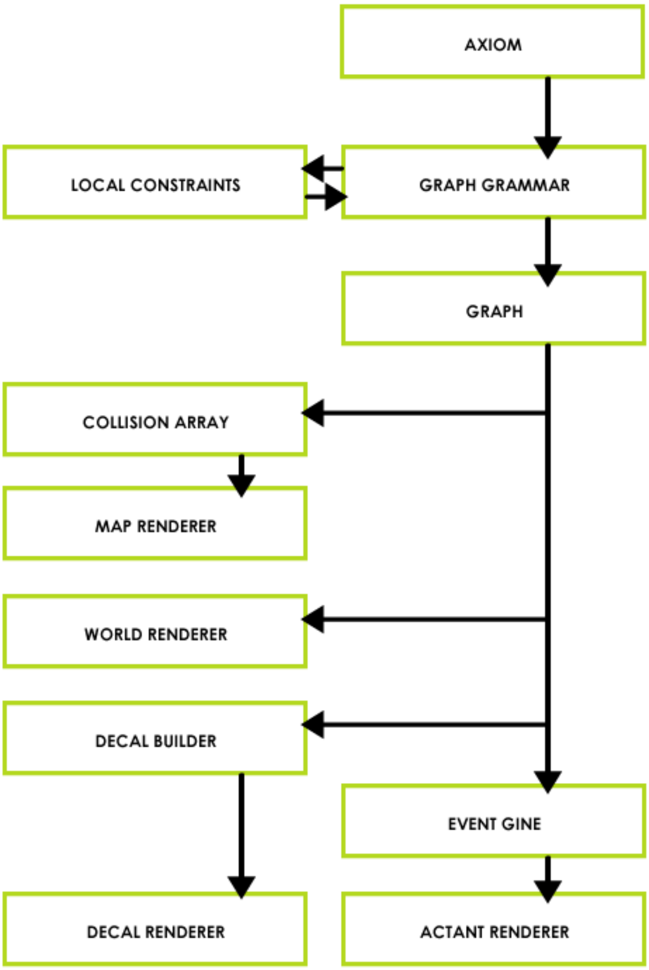


FIGURE 13 | THE STRUCTURE OF THE WORLD BUILDER

In order to build the system, an axiom, a beginning state of the system is given. More specifically, the axiom consists of a single vertex in the graph, representing the beginning of a corridor. This is given to the graph grammar, which, while interacting with a local constraints function (similar to the one presented by Parish et al.) generates a graph. From here, the information is sent four ways.

A collision array is generated, which is used to check for collision when the player navigates the game. Secondly, the graph contains information regarding room and corridor structure. This information is sent to the world renderer, which draws the first part of the world. In order to add detail to the world, this very same information is also sent to what has been entitled the 'decal builder'. The function of this class is to add ornamental details, such as windows and holes in the wall, randomly to the world. Last but not least, the graph structure is sent to the *EventGine* which uses this information to insert the dramatic and minor events into the game. Positional information regarding both unique characters and minions are subsequently sent to the actant renderer, which display these.

One note has to be added concerning the collision array. It is an array containing information as whether to each tile in the world being passable or not. It was however also discovered that this array could serve to illustrate the map of the game world, if represented 2 dimensionally.

6.2.2 The World Builder

The world builder is based loosely on the ideas and concepts described in section 4.2.3. It is not a classic L-System, considering the fact that it uses different rules for translation and positioning of the nodes. Secondly, whereas L-Systems can be parametric, these are typically very simple parameters. The parameters in this system are based, both on the local constraints function, but also on other variables, such as the amount of times the system has attempted to grow from them. Lastly, while the parallel nature of L-System allows for entire model to grow at once, only certain vertices can spawn new ones.

The notation style is however identical to the L-system notation presented in the analysis chapter.

The system's alphabet A consists of the following characters:

$$A = \{\,(\mathbf{Query\ Vertex});\ (\mathbf{Break\ Vertex});\ (\mathbf{Room\ Vertex}); \\ (\mathbf{Corridor\ Vertex}); (\mathbf{Base\ Vertex}); (\mathbf{Base\ Vertex} \cdot \mathbf{set})\,\}$$

The rule-set for the world was also present in the previous semester's report. It has however been modified slightly, both with a slightly modified more accurate terminology, and also with an updated rule-set, to emphasize corridors being generated slightly more often.

1. $(\mathbf{Query\ Vertex}) \overset{\mathbf{30}}{\rightarrow} (\mathbf{Break\ Vertex})(\mathbf{Room\ Vertex})$
2. $(\mathbf{Query\ Vertex}) \overset{\mathbf{70}}{\rightarrow} (\mathbf{Break\ Vertex})\big(\mathbf{Corridor\ Vertex}(!\,\mathbf{grown})\big)$
3. $\Big(\mathbf{Room\ Vertex}\big(!\,\mathbf{grown}, \mathbf{if}(\mathbf{LC})\big)\Big) \rightarrow (\mathbf{Room\ Vertex}(\mathbf{grown}))(\mathbf{Base\ Vertex} \cdot \mathbf{set})(\mathbf{Query\ Vertex})$
4. $\Big(\mathbf{Room\ Vertex}\big(!\,\mathbf{grown}, \mathbf{if}(!\,\mathbf{LC})\big)\Big) \rightarrow (\mathbf{Query\ Vertex})$
5. $\big(\mathbf{Break\ Vertex}(!\,\mathbf{grown})\big) > (\mathbf{Room\ Vertex}) \overset{\mathbf{16}}{\rightarrow} \big(\mathbf{Break\ Vertex}(\mathbf{grown})\big)(\mathbf{Query\ Vertex})$
6. $\big(\mathbf{Break\ Vertex}(!\,\mathbf{grown})\big) > \big(\mathbf{Room\ Vertex}(\mathbf{grown})\big) \overset{\mathbf{84}}{\rightarrow} \big(\mathbf{Break\ Vertex}(\mathbf{grown})\big)$
7. $\Big(\mathbf{Corridor\ Vertex}\big(!\,\mathbf{grown}, \mathbf{if}(\mathbf{LC})\big)\Big) \rightarrow \big(\mathbf{Corridor\ Vertex}(\mathbf{grown})\big)(\mathbf{Base\ Vertex})(\mathbf{Query\ Vertex})$
8. $\Big(\mathbf{Corridor\ Vertex}\big(!\,\mathbf{grown}, \mathbf{if}(!\,\mathbf{LC})\big)\Big) \rightarrow (\mathbf{Query\ Vertex})$
9. $(\mathbf{Query\ Vertex}(\mathbf{AttemptedQueries} == \mathbf{10}) \rightarrow (\mathbf{Break\ Vertex}(\mathbf{grown}))$

The query vertex is used to generate rooms and corridors. It has a 30% chance of generating a new room, and a 70% chance of generating a corridor. The query node is, in both rule **1** and **2**, substituted with a **(Break Vertex)**, which are vertices used for potentially branching corridors.

Rule **3** takes the **(Room Vertex)** and checks to see, using the local constraints object, to see if it fits into the space of the world. If it does not, rule **4** ensures it is replaced with a query vertex.

Rule **5** and **6** dictate corridor branching. The *grown* parameter specifies whether or not the grammar has attempted to branch away at that specific point. If it has not, it has a 16% chance of branching away. No matter the outcome, the grown parameter is changed to *true*, ensuring it will not be attempted again.

Rule **7** and **8** are used entirely for corridor generation. They work similarly to rule 3 and 4, by first invoking the local constraints object to check and see if they actually fit into the game world. If the object returns false, the **(Corridor Vertex)** is reverted into a **(Query Vertex)**. One notable difference from the room placement rules is that the length of the corridor is variable.

The very last rule is in place to ensure the world creation algorithm does not continue indefinitely. Occasionally it happens that it simply is impossible for the engine to place a corridor or a room, despite numerous tries. When this occurs, the **(Query Vertex)** is turned into a **(Break Vertex)** with the *grown* parameter set to true.

In this context, it seems appropriate to specify how new vertices are inserted into game world. Corridors are inserted in a random direction, relative to the current position of the query node. Given the current design of the standard room in the game, rooms are always inserted in a direction, positive with the x-axis of the system. When the algorithm is surrounded by rooms and corridors, and unsuccessfully tries to insert new rooms or corridors, it does so at random. It does not consider the immediate environment around it, in an attempt to utilize as much space as possible. This is entirely by design. Had the graph grammar algorithm attempted to do this, more intricate space may have been generated. However the structure would also have become more maze-like, and perhaps less like a real space.

6.2.3 Reinterpreting Local Constraints

A brief comment regarding this project's interpretation of Parish et al.'s local constraint function seems prudent. The current implementation is noticeably simpler than is present in the *CityGen* engine. Specifically, it checks for two things only, namely checks whether any suggested piece is out of bounds, and also whether or not the suggested piece overlaps with any existing game architecture. It is called every time a new room or corridor is attempted generated.

Figure 14 displays a layout generated by the final graph grammar.

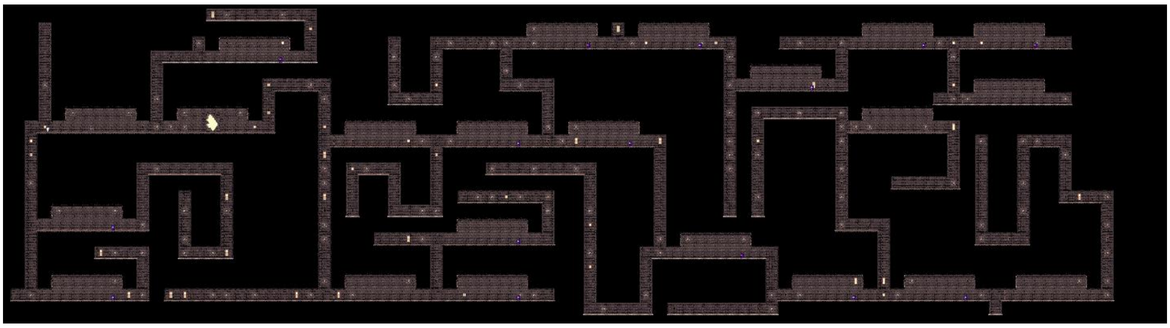


FIGURE 14 | A RANDOM WORLD GENERATED BY THE CURRENT VERSION OF THE WORLD BUILDER

6.2.4 Event Engine

The purpose of the event engine is twofold. First, it places all the major events into the game (containing unique lines of dialogue), and subsequently, the world is filled with minions.

This is done by generating a list of all the room vertices in the graph, and storing their position. A number of approaches were considered in regards to placing major and minor events into the game world. The first, to simply randomly place events into the world was discarded, seeing that it would break with the linear sequencing of the narrative.

This is where a small observation came into play; Numbering each room in the list of rooms, the location of any major event must be at a higher room number than the preceding event, and a lower room number than the forthcoming. This way, it is always potentially possible to experience the narrative sequentially and in the proper order.

At first, some experimentation was done in an attempt to give some variation to the location of each major event. This would mean that in one game world the first major event would occur in the very first room on the rooms list, in another, it would be the 2nd or 3rd. However given the variable size of world, this algorithm quickly proved problematic.

Instead, each major event receives a fixed number for the room list, with the two last major events being allocated to respectively the second last and last rooms on the list.

In order to ensure that players do not experience the events out of order, every room containing a major event has been locked (with an accompanying graphic for the door), making it impossible for the player to enter the room, if the previous event has not yet been experienced.

6.3 Interaction Implementation

The collision detection in *Castle of the Flying Panda* works through a few different methods. As shown in the system overview, once the graph of the world has been built, it is used to generate an array containing information as to which tiles can be maneuvered in, and which can not.

The collision detection algorithm works by taking the location of the player, refactoring the size of the world to the array, and checking if the avatar's current position is allowed. If it is not, the avatar is moved back into a maneuverable tile.

Collision detection with enemies works in a somewhat similar fashion. At every frame update, the game checks to see if the avatar's bounding rectangle intersects with a minion, or if he has been created, the boss. Secondly, the location of the collision is investigated. If the bottom of the avatar's bounding rectangle collides with the top of an enemy, it is safe to assume that the player is attempting to jump and land on the enemy, in an attempt to dispose of it.

Where the player can dispose of each minion by jumping on them once, both the boss and the player were given health counters to ensure they could take more than one hit. During the initial pilot test, the player was given no more than 4 lives (or hearts), whereas the boss had to be hit 3 times in ordered to be defeated. Whereas the latter was perceived as being reasonable, the first had to be adjusted, and in the final testable version, the player was given 8 lives.

In order to ensure that the player does not skip ahead in the events within the game the major events, which the player are not yet meant to see, are all closed off with locked doors. Collision detection with these, work similarly to how it works with enemies.

6.4 Visuals

The final chapter of the implementation will concern itself with the work that went into creating a coherent visual representation of the game. As outlined previously, the story takes place inside a large dark and mysterious castle, whereas the cast of the game are inspired directly by Kung Fu and Wire-Fu movies.

Visually, the art style of the game is kept in a visual style similar to old 8- and 16 bit games, with limited color palettes and up-scaled graphics. Most of the graphics within the game are based on the work done in The Independent Gaming Source *Assemblee*[25] competition, which is open for everyone to use. Some assets have however been created from scratch, in the same visual style as the rest of the game.

6.4.1 Graphics

The level tile set contains of 15 different tiles, which can be combined together to form various layouts. Each tile is 64x64 pixels wide, save the room, which is 448x128.

[25] Which can be accessed from here: http://www.tigsource.com/2009/10/24/tigsource-presents-assemblee-competition/

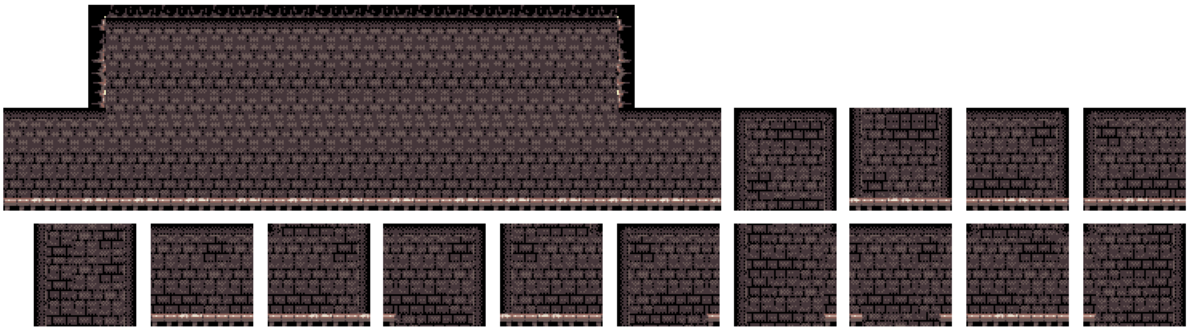


FIGURE 15 | ALL OF THE TILES USED TO VISUALIZE THE GAME.

Each major event in the game has been assigned either a character, or a different ornamental details to create some variety within the environment. Figure 16 displays both characters and ornamental details.

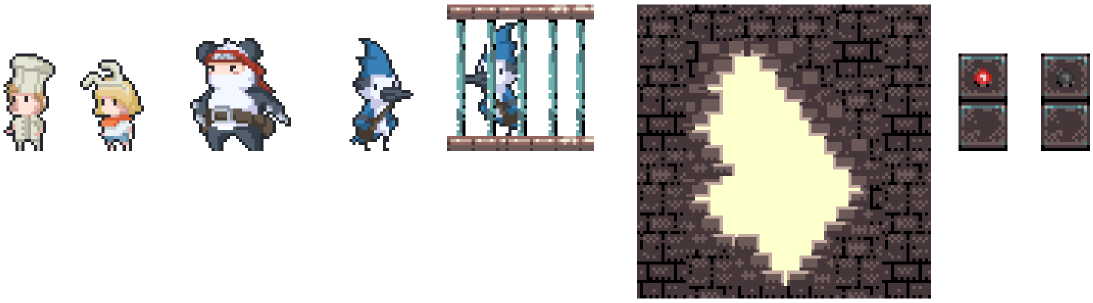


FIGURE 16 | THE VARIOUS CHARACTERS AND MAJOR EVENT PROPS FROM *THE CASTLE OF THE FLYING PANDA*

In order to animate the characters, spritesheets were used. These are simply image files containing all the intermittent steps of one animation within one file. One character typically needs more than one spritesheet, each corresponding to the each animation available to the character. In the case of the hero, Panda Ninja, 4 different sprite sheets are available. One containing a running animation, one with a falling animation, one with a jumping animation, and finally an animation for when the character is just standing around. Figure 17 displays one spritesheet for the character.

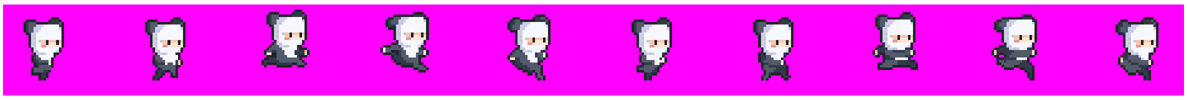
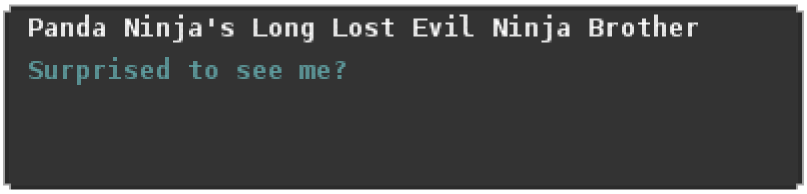


FIGURE 17 | THIS SPRITESHEET FOR PANDA CONTAINS THE RUNNING ANIMATION

6.4.2 Interface Implementation

This subsection is dedicated to explaining how the individual pieces of the interface are put together.

The dialogue box (which pops up every time a major event occurs) and the background for the map are both part of the *assemblee* package. The font used to display the is *Bitstream Vera Sans Mono,* another freeware font.



Panda Ninja's Long Lost Evil Ninja Brother
Surprised to see me?

The map is generated dynamically every time the game is started. As mentioned previously, in the opening section of this chapter, world collision information is stored in a 2-dimensional array when the world is generated. Each position in the array is given either a value of 1 or 0, which signifies whether or not the tile it references to is passable. Drawing only the tiles that are a passable, it is possible to produce a small map of the world.

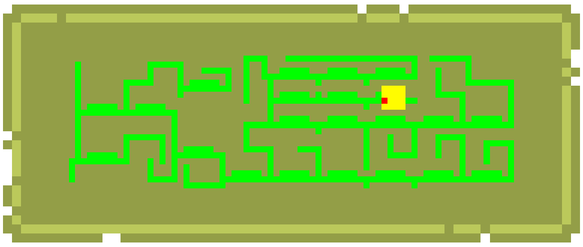


FIGURE 18 | THE MAP FROM THE GAME. THE RED DOT INDICATES THE POSITION OF THE PLAYER, WHEREAS THE YELLOW SQUARE INDICATES THE NEXT OBJECTIVE.

Last but not least, a brief mention of the health icons is warranted. Once again, the graphics come from the *assemblee* package. They are a rendered right above the map in order to collect all the interface elements in one area of the screen.

FIGURE 19 | THE HEALTH BAR AS THE PLAYER SEES IT IN THE GAME

6.4.3 Ornamental Details

In order to create subtle varieties in the level design, a number of ornamental details (or decals) were added to the game.

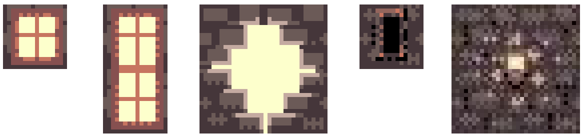


FIGURE 20 | THE VARIOUS "DECALS" USED IN THE GAME

Each decal is rendered on top of existing architecture. At first, it was considered to simply create a number of different hallways containing the different decals. The world renderer would then simply pick a new hallway at random, and render this. This was however deemed inefficient for a number of reasons. For one, the amount of hallways to create would become substantially larger. Without considering the room, the world consists of 14 different hallways. With 5 different decals, this would account to a total of 70 pieces to make. Secondly, it would be largely memory inefficient. If we consider that each hallway is 64x64 pixels, the game would have to contain 286720 pixels worth of memory. Putting decals on top of each highway, the number is almost 5 times smaller.

The decals are added both to the different hallways, but also to the rooms, although only in the corners of each room in order to ensure that they not collide with any "major event" graphics.

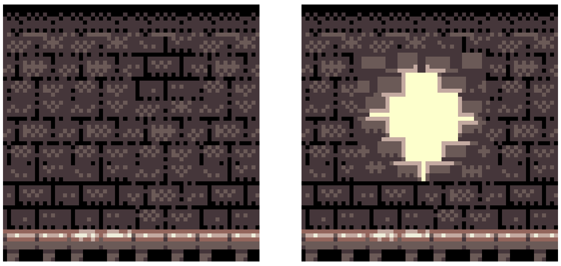


FIGURE 21 | AN EXAMPLE OF WORLD ARCHITECTURE WITH DECAL RENDERING LAYER INACTIVE (TO THE LEFT) AND ACTIVE (TO THE RIGHT)

6.4.4 Effects

As a direct result of the early pilot testing, extra special effects had to be implemented into the game. It was apparent that the people testing the game did not clearly perceive when they were hit by enemy minions, just as they felt that some feedback was missing whenever they landed on an enemy, be it the final boss, or just random minions.

In order to resolve this issue, a new sprite animation was added into the game. Whenever any enemy deals damage to the player's avatar or the avatar deals damage to an enemy, a small animation, in the form of a lightning bolt is played. Figure 22 Figure 22 | The sprite sheet for the lightning bolt, which is played every time the player's avatar collides with an enemy.displays this lightning bolt. If the player lands on a minion, the lightning bolt is put on top of said minion, whereas if the avatar is hit, it is played on top of him. During the final encounter, if the player manages to deal damage to the boss, the animation is played at the point where the Panda Ninja avatar and the evil brother are colliding. This is due to the sprite of the brother being reasonably bigger than the lightning bolt sprite. Another suitable solution would have been to scale the lightning bolt, although this may have caused it to be visually inconsistent with the rest of the objects in the world.

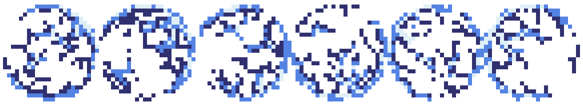


FIGURE 22 | THE SPRITE SHEET FOR THE LIGHTNING BOLT, WHICH IS PLAYED EVERY TIME THE PLAYER'S AVATAR COLLIDES WITH AN ENEMY.

6.5 In Summary

The purpose of the implementation chapter was to present and argue for the game, which has been developed for this project. The two object of most interest is the L-System inspired grammar which had been defined and created, and also the *Eventgine* which hopefully ties the system together. If the test is to be successful, it is detrimental that these two work as intended.

The iterative test approach gave way to a number of improvements, both on the visual side, but also in regards to the actual game play. The idea behind this approach was to weed out annoyances, and possibly improve the narrative if this was necessary. However, as it turned out, only the first was necessary.

7 Test Goals And Procedure

This chapter will cover the initial testing phase of this project. The chapters starts out by outlining the goal of the test of the test, while at same time outlining what is an acceptable success criteria. Secondly, the chapter will cover the three methods for gathering test data, observation, the use of the actantial model and finally, the story summaries which the users will supply.

The test procedure will subsequently be outlined, including some comment and criticism on potential problems that might occur during the testing.

7.1 Test Goals and Success Criteria

At the end of the preliminary analysis, the following problem is set up:

> To what extent can a linearly structured authored narrative be interpreted, if presented in a procedurally generated non-linear random video game world?

The problem formulation implies that in order for the test to provide a successful result, some notion of narrative must be presented, which the user can understand, and relate back to the interviewee.

The formulation is neither inherently verifiable nor falsifiable. The premise of the problem, that it is possible to produce and tell authored narratives in a procedurally generated random world, is however very much verifiable. If any amount of narrative is told, interpreted with the capability of it being retold, the hypothesis, that it is possible to tell narratives in randomized worlds, will be proven true.

The aim of this master thesis is however a bit different. The project has primarily dealt with understanding no whether or not it is possible to tell a story in a procedurally generated world, but rather, why it is so difficult.

Therefore the primary goal of the test is to what extend it is possible for users' to perceive an authored narrative within a procedurally generated random world. In order to do this, some success criteria have been set up.

As the narrative has been divided into smaller elements, dramatic and minor events, it seems prudent to investigate whether or not the users' are capable of identifying this sequencing of events. Secondly, the story features a number of characters, which drives the narrative forward.

As these are instrumental to the progression, the argument can be put forth that users should also be capable of identifying these, while understanding how the interact with the story.

7.2 Methods

In order to answer the problem formulation, three different methods of collecting data was chosen. The users were asked to write down the story, they were asked to fill in an actantial model containing the characters of the story, and lastly, they were being observed throughout the testing session.

7.2.1 Short Story

The idea of having the users write the story down on paper is strongly inspired by the ideas of narrative analysis, in particular both the thematic and the structural methods for narrative analysis. Both methods of analysis are typically carried out orally, but there are potentially some advantages to carrying out test in a written form instead.

Typically, arguing that one method is more time efficient than the other is hardly valid. However, as Riessman points out in her article, one of the caveats with structural analysis is the fact that is extremely resource intensive. As a result, structural analysis may typically only be used on a couple of persons.

Having people writing down the story instead of expressing it vocally may simply be more resource efficient. On the other hand, some of the subtleties that a stenographer might note down during an interview (pauses, tonality) will be lost. This did not seem important for the test, and as a consequence, the written approach was chosen.

7.2.2 Actantial Model

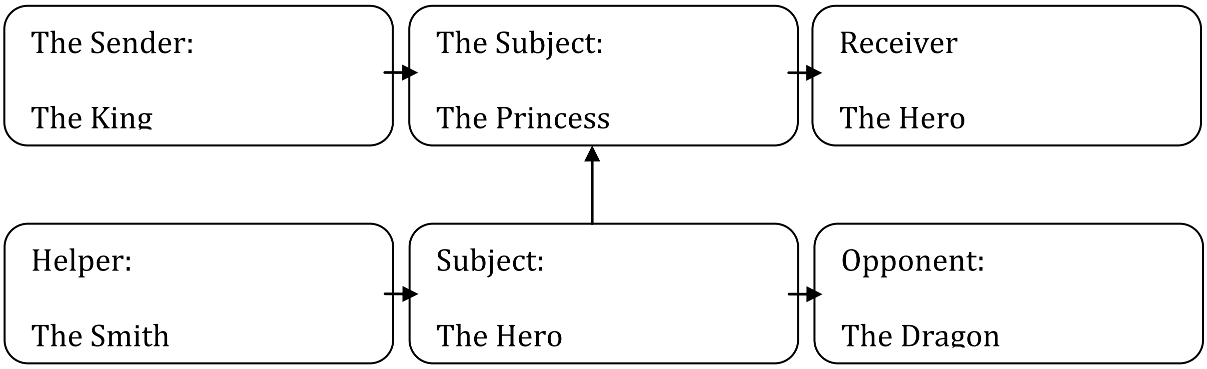
Whereas the objective with the short story is to identify themes and structure, very little focus is given to the characters. The actantial model is a classic analysis model for all kinds of narrative, and it may prove an ideal choice for identifying both characters, but also their relations.

As it is a model used all the way down into the public school, it not only contains the ideal building blocks for analysis, it should also be very much relatable to most test subjects. Unlike the two other test methods, asking people to fill out an actantial model results in a quantitative result, making analysis easier.

The test participants were given a sheet of paper explaining how to fill in the actantial model. In order to explain to them how to fill in the sheet, the following example was given,

> *"In a classic fairy tale, the hero might have to save the princess and slay the dragon. However, he may not be able to slay the dragon before receiving a magic sword from the blacksmith. After having slain the dragon, he returns to the king, who subsequently gives away the princess to the hero, for marriage."*

Along with the text, the following model was provided:



The Sender: The King

The Subject: The Princess

Receiver The Hero

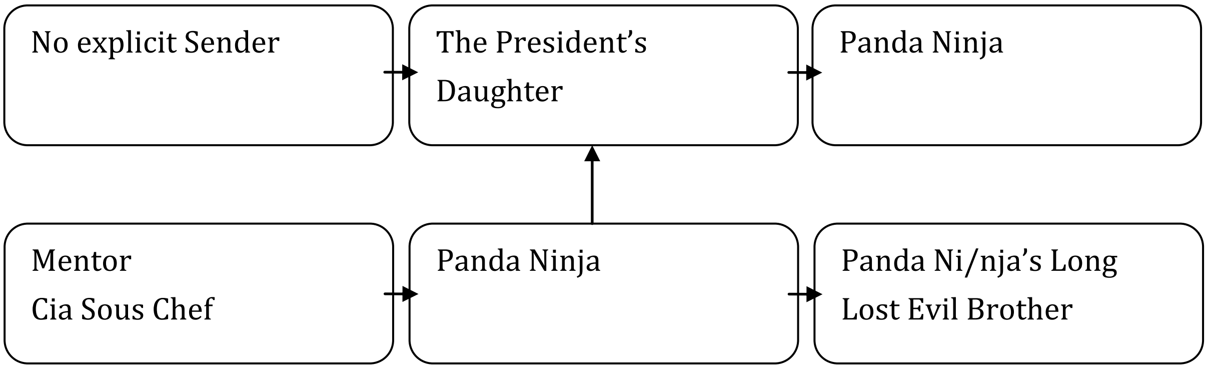
Helper: The Smith

Subject: The Hero

Opponent: The Dragon

They were also told that a story could contain multiple actants, or that a field may not contain one at all.

The test participants were asked to fill in a similar model, but using the video game as the basis for each field. Given that the story was meant to be absolutely unambiguous, the following model can be considered the ideal model, for which the users to aim for, if they manage to interpret and understand the story correctly



7.2.3 Observations

Lastly, the test users were observed throughout the entirety of the testing period, with notes being taken regarding their behavior and general utterances. This is not meant as a replacement for the missing parts of the narrative analysis (the vocal aspect), but rather simply a third mode of data gathering.

It is difficult to create meaning from just observations; instead it should be treated more as a helpful tool for spotting specific (if any) tendencies within the test material.

7.3 Test Procedure

The test procedure can be divided into two different sections, specifically a playing part, and an evaluation part.

The test subjects were first sat down in front the computer and introduced to the game and its mechanics. At first, the game mechanisms were introduced, including controls and basic objective. After the player had gone through the first couple lines of dialogue, the game opened up, and the subjects got their first glimpse of the game world. At this point, the test subjects were introduced to the interface, and were told how the mini map and the health bar worked.

At this point, they were given free hands to do with they pleased within the game, with only minimal supervision. If it seemed as if the subjects were not moving towards the first objective, or having trouble controlling, extra supervision were given in order to make sure in order to make sure the subjects were on the right path.

The subjects were subsequently observed throughout the play session, registering both irregular behavior, but also noting vocal comments, should any be uttered. If at any point, the subject lost the game, he or she was asked to play it again, in order to ensure that the entire narrative was experienced.

Having finished the play-through by completing the game, the test moved to the second phase. Here, the users were asked to fill in provided sheets of paper, containing both the short story assignment, and the actantial model. Once again, they were monitored in order to ensure that the paper was filled out correctly, and that they understood their assignments.

Participants were selected at random, consisting mainly of people from the university campus in Ballerup. Given the qualitative nature of the test, and the work required to interpret the data, only twelve participants were chosen for the test. Of these, one person's answers had to be discarded.

7.3.1 Source Critique

The importance of being critical of one's own test is hard to underestimate. Not only will it make it possible to screen for possible errors and biases in advance, it will also make it possible to readjust success criteria, should any bias or errors occur.

One pre-requisite for being eligible to test was complete was a complete lack of knowledge regarding the project. If test participants had any knowledge regarding the project, they might consciously pay extra attention to the narrative of the game, thus incorrectly skewing the test in favor of a positive result.

Secondly, as outlined in the preliminary analysis, people play video games for a multitude of different reasons, and as such, may play games in different manners. Whereas it is emphasized at the beginning of the test, that the game features a narrative, some players may simply choose to ignore it, leaving it very hard, if not impossible for them to deduce what actually occurred within the game.

8 Test Results

The following section presents the results of the test, with each subsection focusing on its respective test method.

8.1 Actantial Model

This subsection presents the results of the first part of the test, where users were asked to fill in an actantial model.

Every test subject was capable of identifying the protagonist of the story, although not everyone was able to remember his name, Panda Ninja. Two people identified the protagonist as being themselves (using 'me' as a descriptive noun). Four people referred to him as the Panda Ninja, whereas others referred to him as the player or the ninja (in some permutation).

Similarly, everyone managed to identify the object of the story, namely the President's Daughter. Three people curiously identified her as the princess, whereas one other person identified her as the playboy bunny, probably because of her outfit.

In regards to the helper actant, several people identified more than one character. Seven people identified the helper as being the CIA undercover sous chef, whereas seven people similarly identified the Panda Ninja's Mentor. One person failed to identify either of them. Five people guessed both of them. No other answers were registered in this category.

The opponent also had multiple entries in one case. Nine people correctly identified the evil brother as being opponent, with one player (in addition to the brother entry) suggesting also the ninja army. One test subject curiously put 'chef' into the field.

Every test subject save one entered The Panda Ninja as being the receiver of the object. The last one simply left the field blank.

Lastly, eight people suggested that the president could be regarded as the sender of object, with one person leaving it blank, one person entering the mentor, and one person attributing the evil brother.

8.2 Observations

As previously mentioned, some notes were written down during the testing, particular paying attention to the test subjects' behavior and utterances.

One common occurrence was people's lack of attention paid to the mini map, or the health bar, even with introduction. This typically resulted in the test subjects just running around the castle at random, disposing of the ninjas until there was nothing more to do. On the other hand, once the players were once again instructed to look at the map for their next objective, a tendency to look at the map instead of the world seemed to gain ground. One subject started exploring the castle randomly, investigating every nook and cranny. When queried about this behavior afterwards, he noted that he was hoping that some bonuses or extras were hidden in some nooks or crannies.

Two test subjects remarked that they had trouble answering the assignments given afterwards, claiming that the intensity of the game was so high that they did not have a chance to pay attention to the story. In general, the experience was marked by a high level of engagement, and focus on the game experience. One person was giggling loudly at every piece of dialogue presented to her.

8.3 Story Summaries

This section will present, in some detail, the results of the biggest part of the test, the story summaries. In order to present this thoroughly, a twofold emphasis has been chosen. First, some structural considerations will be presented, based on the various summaries, and secondly, a number of themes have been identified, pertaining both to the experience of game, but also the meta-game experience (as defined by the Playscripts article).

As an example of a summary, this was the first supplied answer,

> *"You must save the president's daughter in some kind of old castle. Your useless mentor gets caught and offers some advice to help you find the maiden and bring her back safe. Along the way, you defeat some evil blue ninja "robots" and get some help from the CIA. Towards the end you have a face off against your long lost evil brother and by defeating him you get to save the president's daughter... and have an after-party."*

The rest of this section will aim to identify and define a number of themes present within these summaries.

Division of Events: The game can, as detailed earlier, be divided into eight dramatic events. In the summaries, roughly half of the subjects went through each event, detailing the progression through the game, whereas the other half wrote shorter summaries, not focusing on specific events.

Minor Events: Five people mentioned the one reoccurring minor event within the game, the fights with the ninja robots.

The Control Console: Where most people were capable of identifying and reporting on the majority of the events within the game, the one dramatic event which consistently went unreported the most was the event where the player had to turn off the control console, and then shut down all the ninja robots. While three people mentioned it, only one out of those remembered what the purpose of shutting down the console was.

Identification: Some players identified with the protagonist, the Panda Ninja, more than others. As such, the summaries were written both from the 3rd person perspective, and the first person perspective. One person switches both perspective, describing first his experience with the game, while subsequently describing the adventures of the character. The other subjects who wrote the summary from the first person perspective, assumed the role of Panda Ninja, completely identifying with the character.

Emotive Response: Several of the accounts written from the first person perspective, go into greater detail the relationships between the characters, typically the relationship between the Test subject (Panda Ninja) and the president's daughter. One player writes the following, *"She thanked and told me that I am cute :) (!)"*

Adopting the Language: Some test subjects adopted the language and terminology of the game, more so than others. To exemplify, one subject ends his summary with the following line,

> *"I then rescued the president's daughter and had a "rad" party with her the same night."*

9 Discussion

The primary aim of this chapter is to discuss the outcome of this project. This will be done by first discussing the test results, comparing the various tests which each other, and then subsequently making a qualified assumption as to whether or not the requirements presented by the solution specification were sufficiently met, whether the solution specification was indeed sufficient to answer and the problem formulation, and lastly, whether or not the problem formulation was sufficiently answered.

9.1 Testing Scenario

As shown, the test revealed a quite few things, most of which were dividable into a number of major themes. Perhaps it is not possible to make any quantifiable statements regarding the tests, given the low number of test participants, but some clear indications towards the answer were most certainly visible.

This part of the discussion will begin with debating the actantial model, and subsequently the various numbers of themes identified from the test subjects' summaries.

The actantial model which the users out was, in most cases, mostly correct. This does at least show that users received a strong understanding of the individual characters within the game world, and their relations toward each other. Perhaps, it can be argued, it also implies some understanding of the structure within the world. Curiously however, the majority of the users identified a character, the President, who was not even present within the story and narrative of the world. The most likely explanation however, is that they used the provided example as a base, which lists a king as the sender of the daughter.

Some users also wrote a princess instead of the President's daughter. Whether this was because they did not pay enough attention to the names of the characters is difficult to say. It may also be that they were capable of relating the structure of the game to Super Mario Bros (which, as described earlier, was an inspiration for the endgame), and for that reason, labeled her the Princess, similarly to how Mario must rescue Princess Peach.

Perhaps a short note should be made regarding the subjects identifying with the main character. As reported in the test result chapter, players who identified with the character wrote notably more emotive summaries, than the players who wrote from a third person perspective. One may hypothesize that the thing to take from this, is that players writing from this perspective simply thought a more emotive style was necessary, given the storyline. Another hypothesis is that players identifying with the main character simply become more emotionally invested. The aim of this project is however not to investigate emotional response, nor immersion so this will be left up in the air for now.

Every test subject was capable of identifying the main elements within the story line. Judging from the provided answers, but also from some remarks given post-playing, some participants were capable of remembering more events than others. One subject did write that he experienced the narrative to be ambiguous[26], but the rest did not seem in doubt.

More specifically, the subjects were mostly able to identify the structure of the game, and it is apparent, looking at the answers of the summaries, that most test subjects even presented the story in the same order as it was presented within the game! This seems to at verify the initial problem statement that narratives can be presented within a completely procedurally generated world can be told, interpreted and retold reasonably well. The question still remains however, to what extent?

It is clear that some narrative elements did not get reported on, as often as others. As reported, every test subject was capable of identifying the opponent within the story. However, not a single person made any note regarding the motivation, and the real reason behind the kidnapping. In the story, the evil brother kidnaps the President's Daughter, merely to demonstrate his powers to Panda Ninja, and make him join him. One of the dialogue lines between Panda Ninja and his brother reads,

> *"All of this. I've planned it, planned for you to come here and see me. Join me, join me brother, and together we shall rule the world."*

The reasons why this went unreported may be multiple, and again, it seems possible to only guess and hypothesize why it may be. One reason may be that it may be well beyond the limits of most users' capability to remember and report upon a video game

[26] It is difficult to however analyze whether he however meant that the story was intentionally ambiguous, or that he was simply not able to deduce the overall narrative. In any case, the answer represented the smallest possible minority, and will thus be disregarded for the remainder of the discussion.

plot. Note that no distinction between procedurally generated worlds, and regularly designed worlds is being made here. This will be accounted for in a forthcoming part of the discussion.

Another reason why this might be the case, is that the antagonist's (The evil brother's) motivation may not have taken up enough space, in comparison with the rest of the plot, being presented merely through one line of dialogue. On the other hand, the game is not exactly heavy on dialogue, containing a total of 35 lines, counting the narration both in the beginning and at the end. The entire event was written to be a parody of the climax in Star Wars episode 5, a scene which such a large cultural impact that most people should be familiar with it. This should have worked in favor of that specific part, but people may (once again, this is merely a guess) treated as just that, a parody. A parody which had only the only purpose of driving the story forward.

One other major event was largely ignored, or not understood by most test subject, the event involving the control console. Its role in the monomythical structure was to be the final test for the main character, before being able to face off with the head antagonist. However, while it was mentioned by some, only one person seemed to realize its purpose. There is always going to be a fine line as to how much exposition and description a narrative must contain, and at what point it becomes superfluous. In this case, there was not enough. A way to remedy this, would be to explain more as the console is being shut down, perhaps with more internal dialogue from the game's Protagonist.

Seemingly very little attention was paid to the environment in which the game took place. It was never explicitly stated anywhere that the game took part in an old castle, and perhaps as a direct consequence thereof, only three people mentioned the location in their summaries. At a glance, this seems to somewhat conflict with some of the theory presented by Henry Jenkins, who regards narrative architecture as being important to the narrative experience

However, there is more to it than that. As pointed out in the analysis, game narratives can be told through evocative spaces, that is, spaces which rely on the users' experience, familiarity, and expectances to form the universe for the user. As the discussion on the past few pages show, this was not always done successfully throughout this game.

Another term, also covered in the analysis is the concepts of spatial stories, and spatial storytelling. This is where the spatial structure of the video game comes in, and where one of the biggest challenges within this project lied.

9.2 Structure Versus Content

If we consider the point just made, that it is possible to differentiate between the structure of a story, spatially, and the methods which are used to tell it, a new hypothesis can suddenly be presented; that if one was to produce a video game with a authored narrative identical to the one presented in this project, but in a world designed by human hand, and not stochastically by machine, the outcome of a potential test would be the same, given that they had access to the same assets.

People would very likely be ignoring the same story elements (outside of one case which will be presented later), just as they would be able to interpret and retell the same amount of story. This is obviously an entirely untested hypothesis, and quite a claim to make at the end of a report, but if true, it opens up and presents a whole new line of challenges within the paradigm of procedural content generation.

As such, the main differentiator between authored worlds and procedurally generated ones becomes the spatial structure of the world. In an authored world, the structure must be decided by hand, whereas in a random world, it can be entirely dynamic. It is not unreasonable to argue then, based on the test results, that the algorithm within the game did an adequate job. People's summaries did at least indicate that they were capable of not only separating the majority of the events, but also structure them in order.

The algorithm used within this project used fixed numbering (room #1, room #4 etc) to distribute and place the events within the game. That is not to say the placements were in any remotely fixed, spatially, within the final world. The strength of the parallel rewriting mechanisms within the grammar, combined with the stochastic approach to just about every aspect in the application (from randomly choosing between rooms and corridors, to corridor lengths to directions) ensured highly random layouts. And while the final encounters with the Evil Brother and the President's Daughter most often lay within the very right side of the map, the way of getting there appeared wildly different in every test.

9.3 The Challenges Of Pacing

There are other obvious challenges that this project did not test for (nor did it intend). One issue detrimental to video games is pacing. As a designer, pacing is an important in order to keep the experience as enjoyable as possible. Some events or scenarios[27] may be perceived as drawn out if they go on for too long, just as people will be left craving for more, or might feel unsatisfied if they are over in an instant.

However within this project, in one instance, the pacing, or lack of pacing, may have created a structural problem, which in turn may have caused interpretative problems. The event (or scenario) debated here, is the previously mentioned control console. Ideally, the scenario the player experiences would be this[28];

> The player is told to turn off the control console in order to turn off all the robots within the castle. He makes his way to the control console (which should be placed somewhere in the middle of world, perhaps a bit towards the end), disposing of some robots as he sees them, jumping past others, and finally reaches the room. The console is turned off, and immediately all the robots vaporize and the player is ready to face off with the villain. The villain is however not in the next room. Instead, he is hiding further into the castle, perhaps requiring some backtracking to reach him. It is as the player travels through the castle that he must also experience it being empty. To experience that he has effectively turned off all the robots.

In the case of the procedurally generated world this may however not always be the case. The worst case scenario presents him as being merely a single room away, which would not allow the player at all to experience the empty castle, but if he is even one or two rooms away, the effect may be present. Two solutions seem apparent. Either a new intelligent algorithm must be introduced, which takes into account the pacing of the game world. In an ideal world, this would be the proper solution. Not only would it make it possible to make the game more structurally coherent, such a potential algorithm could also be capable of monitoring and adjusting the pace to the specific player. This may even be an entire project in itself!

[27] Admittedly, this term will be made up entirely for the discussion. A scenario within a computer game is a set of isolated dramatic and minor events occurring in succession.
[28] Regardless of the world being authored by hand, or generated by machine.

However reduced to the scope of this project, the solution may be simpler. Removing the event, or replacing it with one which is not as dependent on being structurally well paced may also do the trick. As pointed out a couple of paragraphs ago, the purpose of this project was not to produce a well paced game, and while some narrative elements depend on it, it may not be necessary to include such events.

It seems, at this point, ideal to return and debate story telling techniques, and their merits within the context of this project. Whereas the structure was the element which truly differentiates "Castle of the Flying Panda" from games with hand-made worlds, studying and understanding story telling techniques (evocative spaces, enacted narratives) was very much a necessity within the scope of this project. If the experiment was to have any credibility, it was necessary to introduce and tell the story as convincingly as possible, and as such, the study of story telling techniques was justified. Aside from a few missed elements, it is not unreasonable that this was accomplished in a manner which could be considered reasonably well.

Though this discussion does not determine exactly to what extend it is possible to present narratives in procedurally generated random worlds, it is possible to put forth the argument that given the same story telling tools, assets, and techniques, it may be done equally as well as in any authored world.

Some story telling tools can however be considered considerably stronger in an authored context, so there are still more challenges to consider in the future.

9.4 Perspectives: Building Interesting Spaces

Hand-made worlds and procedurally generated worlds can be differentiated in one more way; through the design of the individual spaces or set pieces. It was attempted to create variation within the various rooms of the game through the introduction of decals. No part of the test really accounted for these, but perhaps at this moment, it is prudent to compare what was achieved with the decals, to what can be achieved in a designed space. One of the aims of designers is to create interesting[29] environments. This can be done with many techniques, for instance, through evocative spaces. No matter how many arguments are put forward, the environments in Castle of the Flying Panda simply are not interesting. The corridors are very much alike, and while some variation do exist between the individual set pieces, it is hard to attribute meaning to any of it. As it stands, it may even be considered a bit soul less?

Perhaps therefore another challenge within the paradigm of procedural content generation will be intelligent generation of interesting spaces?

[29] Admittedly, this term is as vague as it gets, but an interesting space can be considered a space which invokes the curiosity of the player, in an attempt to keep him in the front of the screen

10 Conclusion

In general, it seemed as if the merger of authored narratives and procedurally generated random worlds was to be an uneasy one. As such, no games, at least to the knowledge of the author of this project, had managed to tell a coherent structured narrative in a completely random world. Therefore, the goal of this project became to, not only attempt this merging, but also to investigate exactly to what extend it was possible. The final problem statement became as follows,

> To what extent can a linearly structured authored narrative be interpreted, if presented in a procedurally generated non-linear random video game world?

In order to solve this, a number of different topics, games, and academic debates related to both video game narratives and procedural content generation were studied, both in the preliminary analysis, and the subsequent analysis chapter. The sections on narratives rewarded the project with a number of structures (both narrative structures, but also potential world structures), models (the monomyth, for instance) and theories (evocative spaces, and the gameplay of cutscenes).

Similarly, the study of procedural content generation was equally rewarding. Arguably the most important discovery here came in the form of the L-System grammar, which proved both as a strong inspiration and base for the world builder, the engine from which the world was built. In order to test the problem statement, narrative analysis proved of special interest. Originally developed as tool to create structure and meaning from people experiences, in its modified form it also proved a viable tool to re-interpret the users' experiences of the narrative within the game.

The low number of users tested means it is hard to produce a definite conclusion to the problem. If anything, it is at least not possible to determine to what specific extent an authored narrative can be interpreted. However, that is not to say that the project did not produce a number of results.

The most immediate observation made from the test is that users generally were capable of perceiving the story. If the story was to be divided into structure, events (be it dramatic or minor) and characters, people were capable of identifying the majority.

Some things were more questionable than others, and not every event was perceived as clearly. This may however have been related to how the narrative was structured in the random world.

The obvious comparison in a project like this will always be the hand-made counterpart, and perhaps the most interesting result in this regard is not as much an objective result as it is a new hypothesis, namely that given the same assets, same linear story line in the same setting, the perceived narrative will be the same in each case. A procedurally generated random world can then have the benefit of being new every time the game is loaded, something the hand-made game will not.

11 Bibliography

Avatars of story [Book] / auth. Ryan M.L.. - [s.l.] : Univ Of Minnesota Press, 2006.

Charbitat [Online] / auth. Nitsche Michael. - 2007. - 2 18, 2010. - http://egl.gatech.edu/charbitat/.

Cybertext: perspectives on ergodic literature [Book] / auth. Aarseth Espen. - [s.l.] : Johns Hopkins Univ Pr, 1997.

Designing procedural game spaces: A case study [Conference] / auth. Nitsche M. and Ashmore, C. and Hankinson, W. and Fitzpatrick, R. and Kelly, J. and Margenau, K. // Proceedings of FuturePlay. - [s.l.] : Citeseer, 2006.

Dyson [Online] / auth. Kremers R. and May A. // Dyson - Independent Games Festival Finalist. - dyson-games.com, 2009. - November 23, 2009. - http://www.dyson-game.com.

Gamasutra [Online] / auth. West Mike. - 8 6, 2008. - 4 2, 2010. - http://www.gamasutra.com/view/feature/1648/random_scattering_creating_.php.

Game Design as NArrative Architecture [Article] / auth. Jenkins H. // First person: New media as story, performance, and game. - 2003.

Games Telling Stories [Journal] / auth. Juul Jesper. - 2001. - 1 : Vol. 1.

Glenn R. Wichmann [Online] / auth. Wichmann G. // "A Brief History of Rogue". - 1997. - November 27, 2009. - http://www.wichman.org/roguehistory.html.

Graphical Applications of L-systems [Book] / auth. Prusinkiewicz P. and Science University of Regina Dept. of Computer. - [s.l.] : Citeseer, 1985.

Hamlet on the Holodeck [Book] / auth. Murray Janet. - [s.l.] : MIT Press, 1998.

HEARTS, CLUBS, DIAMONDS, SPADES: PLAYERS WHO SUIT MUDS [Online] / auth. Bartle R.. - August 28th, 1996. - 4 3, 2010. - http://www.mud.co.uk/richard/hcds.htm.

Here Be Dragons: Emergent Procedural Landscapes [Online] / auth. Furmanski T.. - 2005. - November 15, 2009. - http://interactive.usc.edu/members/todd/archives/furmanski_thesis_paper.doc.

In Defense of Cutscenes [Conference] / auth. Klevjer R. // Computer Games and Digital Cultures 2002 Conference Proceedings. - 2002.

Instant Architecture [Journal] / auth. Wonka P. and Wimmer, M. and Sillion, FX. - [s.l.] : ACM transactions on graphics, 2003. - 3 : Vol. 22.

Introduction to Narratology [Online] / auth. Felluga Dino. - July 2002, 2002. - 2 4, 2010. - http://www.cla.purdue.edu/english/theory/narratology/.

Introversion [Online] / auth. Delay Chris // Subversion Blog. - Introversion, January 15, 2007. - December 4th, 2009. - http://www.introversion.co.uk/subversion.

Narrative across media: the languages of storytelling [Book] / auth. Ryan Marie Laurie. - [s.l.] : Univ of Nebraska Pr, 2004.

Narrative analysis [Book] / auth. Riessman C.K. - [s.l.] : Sage Publications, Inc, 1993.

Narrative as virtual reality: Immersion and interactivity in literature and electronic media [Book] / auth. Ryan M.L.. - [s.l.] : Johns Hopkins University Press Baltimore, MD, USA, 2001.

Playscripts a new method for analyzing game design and play [Journal] / auth. Aldred J. and Biddle, R. and Eaket, C. and Greenspan, B. and Mastey, D. and Tran, M.Q. and Whitson, J.. - [s.l.] : ACM, 2007.

Procedural Graphics Model and Behavior Generation [Journal] / auth. Hidalgo JL. [et al.]. - [s.l.] : Springer, 2008. - Vol. 5102.

Procedural level design for platform games [Conference] / auth. Compton K. and Mateas, M // Proceedings of the Artificial Intelligence and Interactive Digital Entertainment International Conference (AIIDE). - 2006.

Procedural Modelling Of Buildings [Conference] / auth. Müller P. [et al.] // Proceedings of the 28th annual conference on Computer graphics and interactive techniques. - New York : ACM, 2006.

Procedural Modelling Of Cities [Conference] / auth. Parish, Y.I.H. and Müller P. // Proceedings of the 28th annual conference on Computer graphics and interactive techniques. - New York, USA : ACM New York, NY, USA, 2001.

Quest games as post-narrative discourse [Book Section] / auth. Aarseth Espen // Narrative across media: the languages of storytelling / book auth. Ryan M.L.. - [s.l.] : Univ of Nebraska Pr, 2004.

Rhythm-based level generation for 2d platformers [Conference] / auth. Smith G. and Treanor, M. and Whitehead, J. and Mateas, M. // Proceedings of the 4th International Conference on Foundations of Digital Games. - [s.l.] : ACM, 2009.

Search-based Procedural Content Generation [Journal] / auth. Togelius J. and Yannakakis, G.N. and Stanley, K.O. and Browne, C.. - 2010.

Simulation vs. Narrative: Introduction to Ludology [Journal] / auth. Gonzalo F. - [s.l.] : Estados Unidos: Routledge, 2003.

Synthetic Topiary [Conference] / auth. Prusinkiewicz P. and James, M. and Mech, R. // Proceedings of the 21st annual conference on Computer graphics and interactive technique. - [s.l.] : ACM, 1994.

The algorithmic beauty of plants [Book] / auth. Lindemayer A. and Prusinkiewicz, P. and others. - [s.l.] : Springer-Verlag, 1996.

The Backroom Boys: The secret return of the British Boffin [Book] / auth. Spufford F.. - [s.l.] : Faber, 2003.

The FL-system: a functional L-system for procedural geometric modelin [Journal] / auth. arvie J.E. and Perret, J. and Bouatouch, K.. - The Visual Computer : Springer, 2005. - 5 : Vol. 21.

The gaming situation [Journal] / auth. Eskelinen M.. - 2001. - 1 : Vol. 1.

The Quest in a Generated World [Conference] / auth. Ashmore C. and Nitsche, M. // Proc. 2007 Digital Games Research Assoc.(DiGRA) Conference: Situated Play. - [s.l.] : Citeseer, 2007.

The Uneasy Merging Of Narrative And Gameplay [Online] / auth. Shirinian Ara // Gamasutra. - January 26, 2010. - 5 2010, 12. - http://www.gamasutra.com/view/feature/4253/the_uneasy_merging_of_narrative_.php.

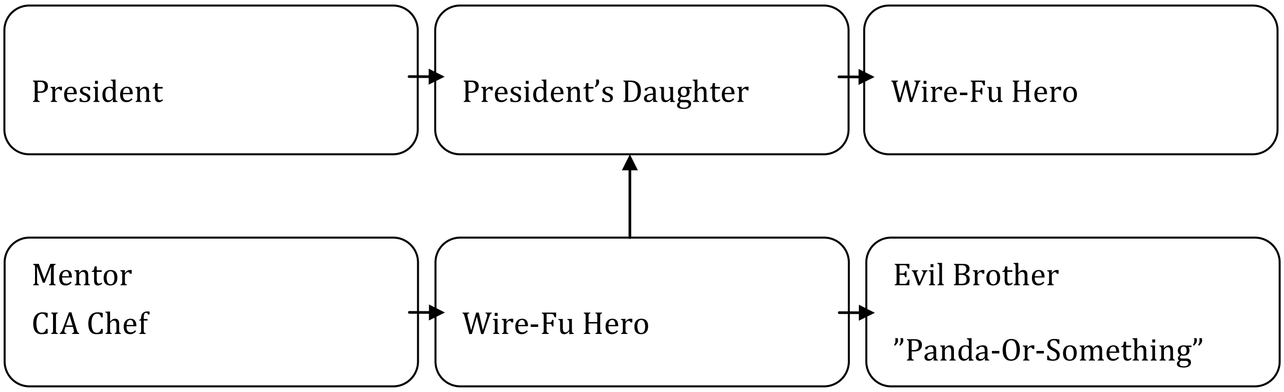
Video Game Spaces [Book] / auth. Nitsche M.. - [s.l.] : Cambridge, Massachusetts: MIT Press, 2008.

Visual models of plants interacting with their environment [Conference] / auth. Mech R. and Prusinkiewicz, P // Proceedings of the 23rd annual conference on Computer graphics and interactive technique. - NY, US : ACM New York, 1996.

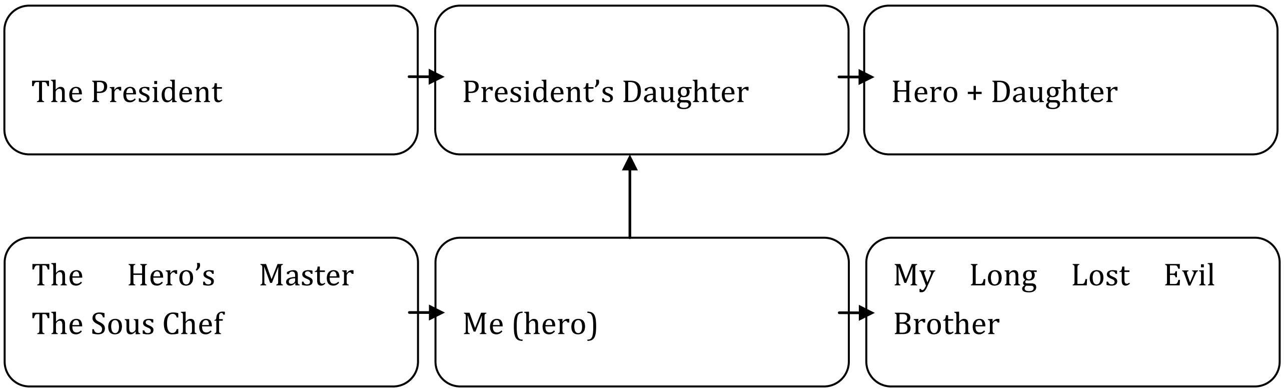
12 Appendix

A. Actiantal Models

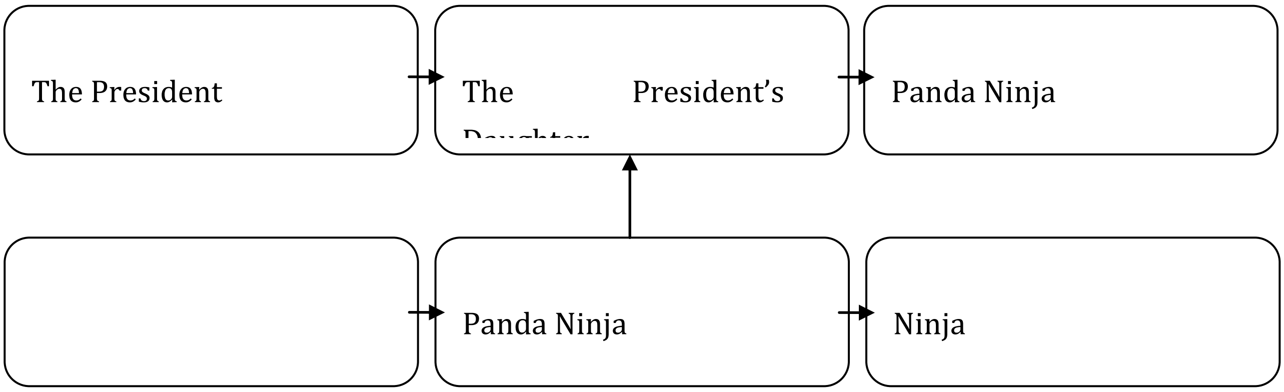
Participant #: 1



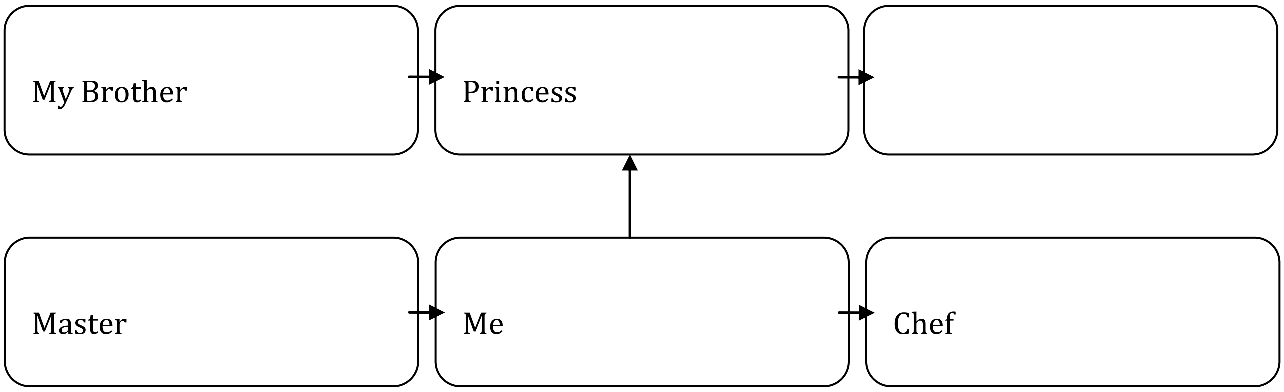
Participant #: 2



Participant #: 3



Participant #: 4



My Brother

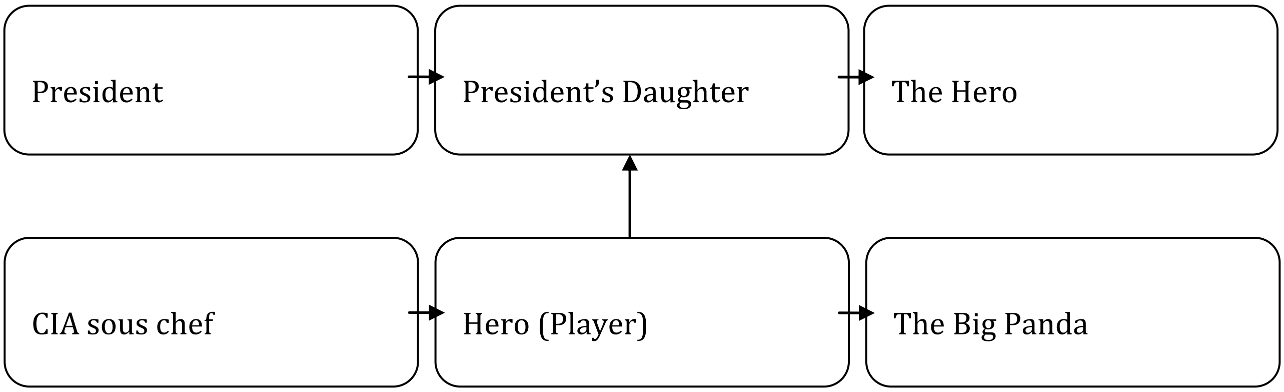
Princess

Master

Me

Chef

Participant #: 5



President

President’s Daughter

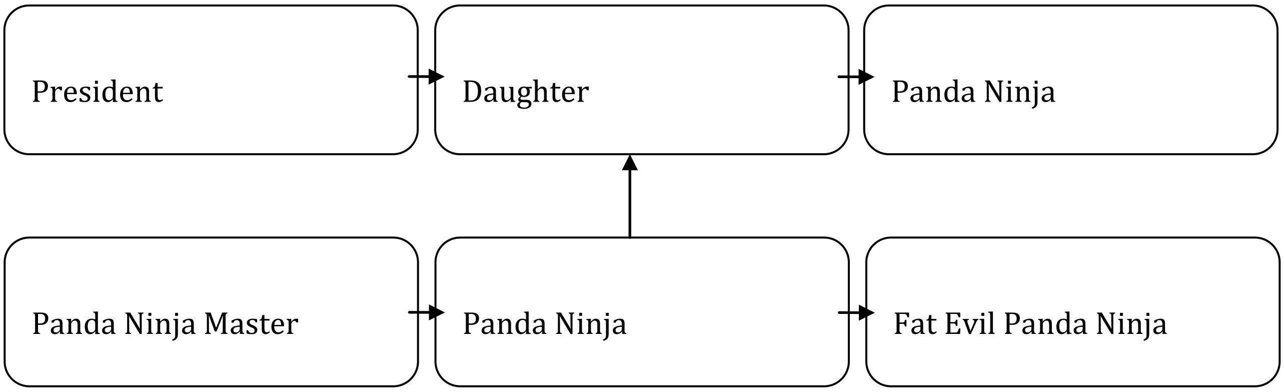
The Hero

CIA sous chef

Hero (Player)

The Big Panda

Participant #: 6



President

Daughter

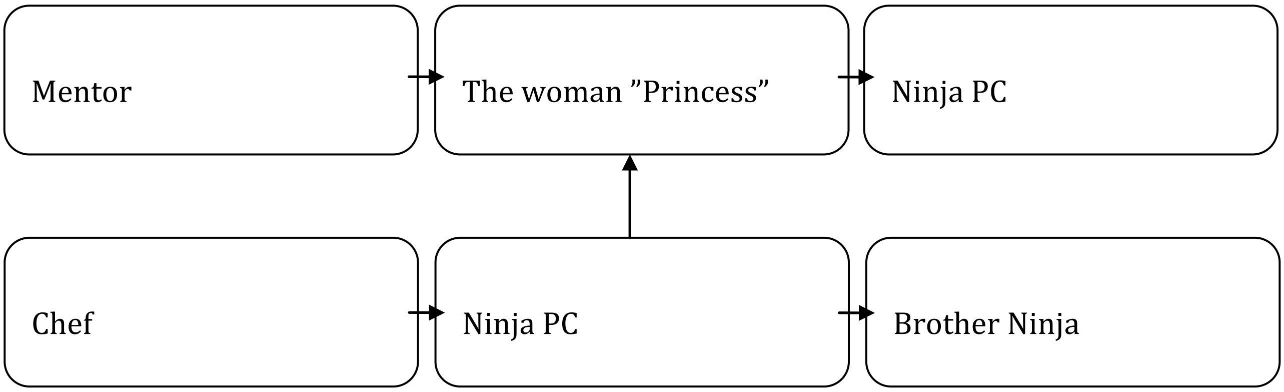
Panda Ninja

Panda Ninja Master

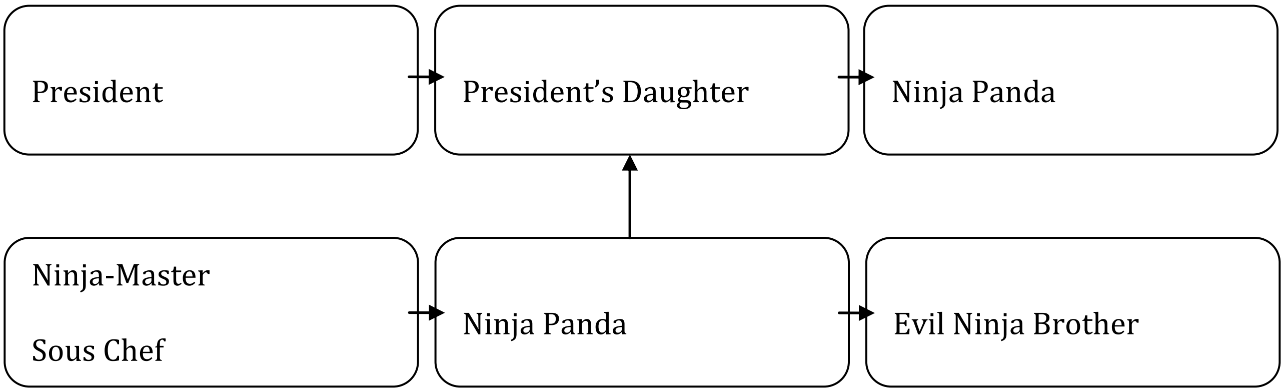
Panda Ninja

Fat Evil Panda Ninja

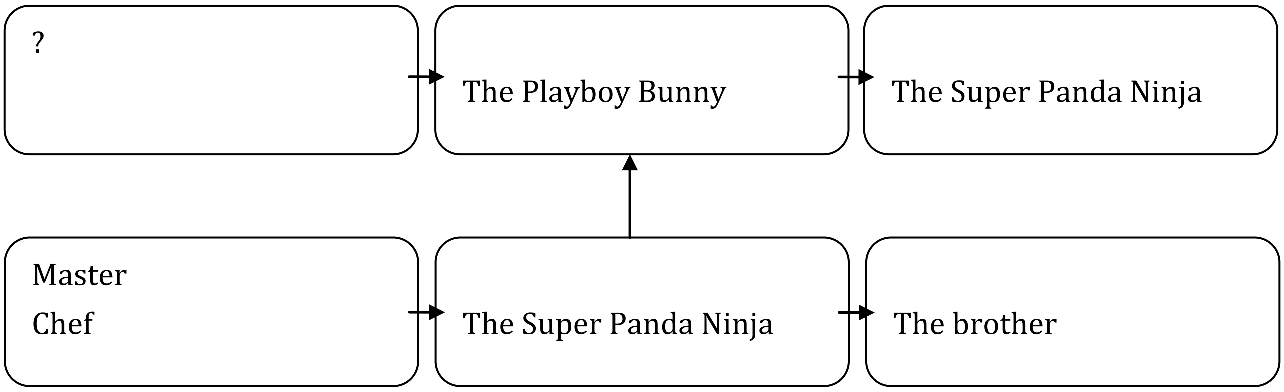
Participant #: 7



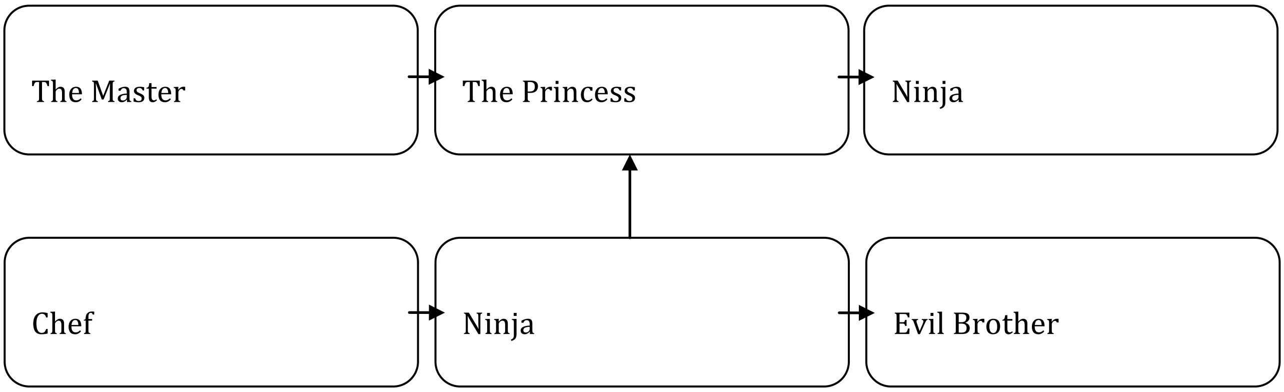
Participant #: 8



Participant #: 9



Participant #: 10



The Master

The Princess

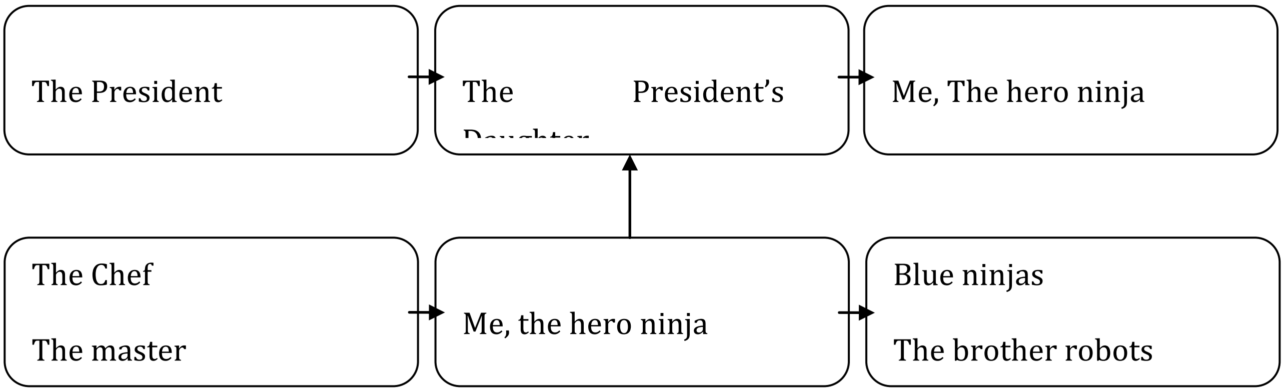
Ninja

Chef

Ninja

Evil Brother

Participant #: 11



The President

The President's Daughter

Me, The hero ninja

The Chef

The master

Me, the hero ninja

Blue ninjas

The brother robots

B. Story Summaries

Summary #: 1
Seed #: 1

Story Summary: "You must save the president's daughter in some kind of old castle.
Your useless mentor gets caught and offers some advice to help you find the maiden and bring her back safe.
Along the way, you defeat some evil blue ninja "robots" and get some help from the CIA.
Towards the end you have a face off against your long lost evil brother and by defeating him you get to save the president's daughter... and have an after-party."

Summary #: 2
Seed #: 10

Story Summary: "The president's daughter was kidnapped at a party at the playboy mansion and it was up to me, using my incredible ninja skills, to find her and escort her back to safety"

Summary #: 3
Seed #: 11

Story Summary: “I had a role of panda ninja. I had to rescue the president's daughter; she had been kidnapped by some bad guys after the party.
I decided to go and help her, on my way I faced some bad guys which I had to defeat.
Hopefully I made my way through and I managed to help the daughter of the president. She thanked and told me that I am cute :)"

Summary #: 4
Seed #: 12

Story Summary: "In order to find the cute princess I had to fight blue creatures, my brother, and to find the way through the huge castle.
And all that happened because she disappeared from the playboy mansion. My master (in a cage), my chef (who eeeked me) help me to

find the lovely princess.
Yeah!"

Summary #: 5
Seed #: 18

Story Summary: "I experienced a story which was extremely intense, fast and ambiguous. The play evolves on finding waypoints, and jumping on the enemies or avoiding them.
The main goal is to rescue the daughter of the president. He will start the story by searching for his master. Which will go through first finding a bird then getting information from the CIA sous chef, which will follow with the having to kill a big dog (which is at the end, a bigger panda).
Shut down robots. The final challenge will be to reach the president's daughter. From then the story will end with the two hanging out."

Summary #: 6
Seed #: 22

Story Summary: "Presiden't daughter kidnapped from playboy mansion.
No-one is here.
Get to the chef.
Get to the panda Ninja Master.
Unlock Ninja Master or something. (press button)
Find out who’s behind it all.
"Oh it's my evil fat ninja brother."
Kill fat ninja and rescue the princess :)"

Summary #: 7
Seed #: 1

Story Summary: "The player as a ninja was tasked to rescue the "princess". First to get an overview the master had to be found but he was imprisoned! He told the player to go find an undercover chef. He told the player to go find the evil boss-ninja who turns out to be the player's brother. He gets his ass kicked!"

Summary #: 8

Seed #: 10

Story Summary: "This little panda ninja-guy was summoned to save the president's daughter whom had been captured by the evil ninja-brother.

The ninja-master and sous chef helps the ninja to find the president's daughter.

In the end she is saved (after beating the evil brother) and they go to a kick ass party!"

Summary #: 9

Seed #: 11

Story Summary: "A super panda ninja is sent to rescue some playboy bunny thing. He has to free her from some unknown evil which after some investigation turns out to be the super panda ninja's own brother.

The panda is forced to fight his brother and by doing so, he defeats the brother and rescues the playboy bunny and all is well.

The end."

Summary #: 10

Seed #: 12

Story Summary: "It was year 2012 or something and the princess had been kidnapped! From the playboy mansion.

As a ninja I had to rescue her. First I had to look around and return to my master who told me to go to the chef who was a bit further in the castle. He gave me a key, which I used for something I can't remember and then I had to find the boss, who was my brother and fight him. Afterwards I rescued the princess."

Summary #: 11

Seed #: 18

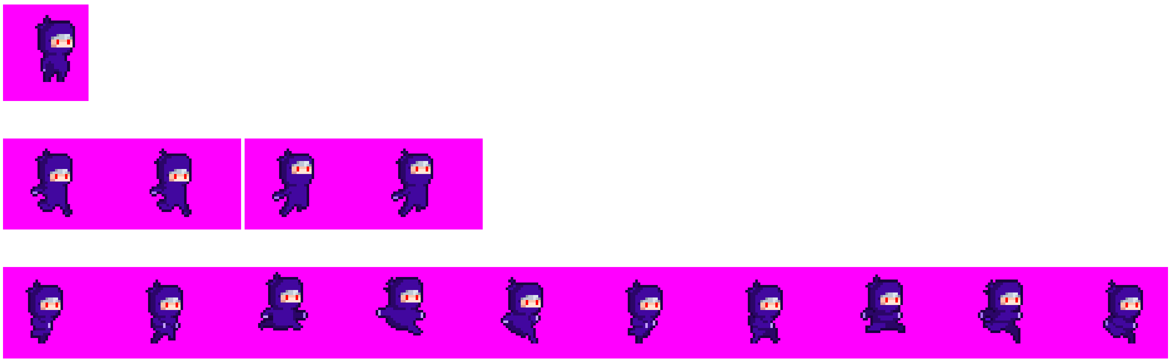
Story Summary: "I was a ninja on a mission to save the president’s daughter who was being held captive inside a castle.

The first task was to find my master who was also held captive. A chef then had more information telling me to find a button to shut down the robots patrolling the castle.

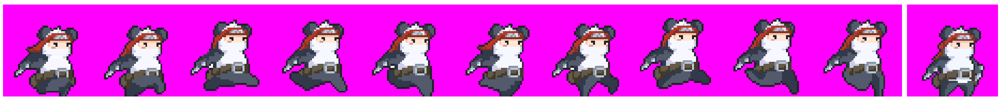
The task for me was now to find out who was behind the kidnapping of the president's daughter. In the end it turned out to be my own brother, whom I then fought to the death (winning of course).
I then rescued the president’s daughter and had a "rad" party with her the same night."

C. Character Sprite Sheets

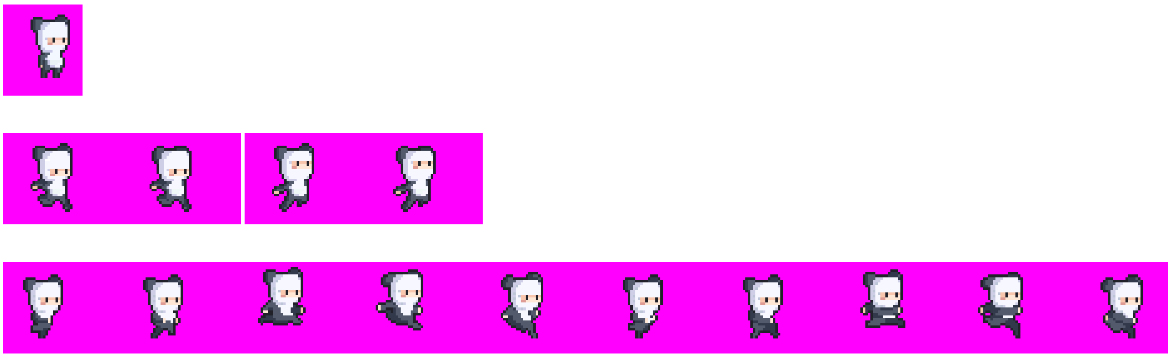
Ninja Robot Minion



Panda Ninja's Brother



Panda Ninja



D. Script

Event No#: 1

Participants: None

Dialogue: "In the year 2021 AD… The president's daughter has been kidnapped at a party at the Playboy Mansion!!! As a student and practitioner of the elusive and completely useless martial art, Wire-Fu you and your master must rescue her from her mysterious kidnappers."

Other Remarks: The screen is black as this event plays. Once it is over, the game appears.

Event No#: 2

Participants: Panda Ninja

Dialogue: "Noone in here... Man, this castle is big and vast. I better find my master, before I do anything else."

Event No#: 3

Participants: Panda Ninja, Panda Ninja's Mentor

Dialogue:

Panda Ninja, "Master?"

Mentor, "I did not want you to see me this way."

Panda Ninja, "What happened?"

Mentor, "They caught me. Hurry, there isn't much tim- "

Panda Ninja, "Let me see if I can free you from this cage."

Mentor, "It is too late for me. You have more important tasks at hand !!"

Panda Ninja, "I'll come back for you once I find out who is behind all of this"

Mentor, "We have a man in the inside. Find the chef. He will know what to do."

Event No#: 4

Participants: Panda Ninja, Undercover CIA Sous Chef

Dialogue:

Sous Chef, "eeek!"

Panda Ninja, "..."

Sous Chef, “ oh, it's you."

Panda Ninja, "Master told me to find you."

Sous Chef, "If you want to fight the big dog, you have to shut down the robots! The robots man, it's your only chance!"

Panda Ninja, "And how do you presume I do this? I can't possibly fight all of them, there are way too many of them"

Sous Chef, "Here. Take this key, and find the robot control room. It's somewhere deeper inside the castle."

Event No#: 5

Participants: Panda Ninja

Dialogue: "Great, that should do it. ...now I just have to find whoever is behind this mess!"

Other Remarks: A console (controlling the robots) is turned off as this event is played out.

Event No#: 6

Participants: Panda Ninja, Panda Ninja’s nefariously evil long lost Brother

Dialogue:	
Panda Ninja,	"noooOOOOOOOooooOOOOOOOooo!";
Brother,	"Surprised to see me?";
Panda Ninja,	"But.. but.. you're my brother!";
Brother,	"All of this. I've planned it, planned for you to come here and see me. Join me, join me brother, and together we shall rule the world.";
Panda Ninja,	"I think I'm just gonna have to kick your ass, Super Mario Bros Style(TM) instead!"

Event No#:	7
Participants:	Panda Ninja, The President's Daughter
Dialogue:	
Daughter,	"You haven't come to torture me, have you?";
Panda Ninja,	"...";
Daughter,	"Teehee, you're cute.";
Panda Ninja,	"...";
Daughter,	"Well, what are you waiting for? Aren't you going to escort me out?";

Event No#:	8
Participants:	None
Dialogue:	"Returning to the playboy mansion with the president's daughter, the world was once again at peace." "Also, they had a totally kickin' rad party that same night." "The End."
Other Remarks:	Screen fades to black once this event is activated.