**Annotated Bibliography**

**Callum Myers**

**2102549**

**Brackeys (2017) *PERLIN NOISE in Unity - Procedural Generation Tutorial*. 17 May. Available at:** [**https://youtu.be/bG0uEXV6aHQ?si=Pp4PAXs3kxJ4dOBA**](https://youtu.be/bG0uEXV6aHQ?si=Pp4PAXs3kxJ4dOBA) **(Accessed: 1 November 2024).**

This video by the YouTube game development channel “Brackeys” (Asbjørn Thirslund) shows how to implement a Perlin Noise generator in Unity using the engine’s built in noise functions. It is a good source as the channel is reputable, having focused on programming and game development for years now, and Asbjørn also does a good job of clearly and concisely explaining how to implement the code into a project. It is done using an older version of Unity however this should not cause an issue since most of the functions are largely the same, however he is also writing this code with 3D generation in mind which means I may have to find ways to adapt this for the 2D setting of my own project.

**Snodgrass, S. and Ontanon, S. (2021). *Generating Maps Using Markov Chains.* Available at:** [**http://www.fdg2014.org/papers/fdg2014\_paper\_29.pdf**](http://www.fdg2014.org/papers/fdg2014_paper_29.pdf) **(Accessed: 5 November 2024).**

This thesis paper investigates using Markov Chain functions to adapt a classic 2D platformer, Super Mario Bros, to incorporate procedural level generation techniques. The study was motivated by the idea that having many developers manually creating levels can increase both monetary costs and time efficiency in game studios. I personally felt this could be a strong source for my project because, whilst the aims of this are different to my own, the idea of using Markov Chains to add variety into a level generator is a key part of my methodology, and this paper shows a strong knowledge of implementation. A key difference however would lie in the fact this model using a training algorithm to create levels like a classic Super Mario game, whereas I intend to use more of a parameterised system in my own model, where it would be adapted to make decisions based on rules given by the code, as opposed to being based on a pre-made level.

**Chakraborttii, C. and Ferreirra, L. (2024). *Towards Generating Surprising Content in 2D Platform Games.* Available at:** [**https://doi.org/10.1145/3649921.3659848**](https://doi.org/10.1145/3649921.3659848)**. (Accessed: 8 November 2024).**

This thesis paper investigates how the element of surprise and making unpredictable levels can help to improve player engagement in games. It tests this theory by having players play a classic Super Mario level, then applying the “Violation of Expectation, Caught Off Guard, and Learning Model” to create a second level which is different to what is expected. The results here are especially interesting because it showed that often the “change of linearity” in a level can improve player surprise, which will be useful to back up my theory that PCG can improve player engagement, since my generator has a big focus on creating interesting level layouts. It also states enemy, and object density, can be useful, which I hope to be able to prove through my use of allowing players to set their own parameters for these before level generation begins.

**Smith, G., Gan, E., Othenin-Girard, A. and Whitehead, J. (2011). *PCG-Based Game Design.*  Available at:** [**https://doi.org/10.1145/2000919.2000926**](https://doi.org/10.1145/2000919.2000926)**. (Accessed: 4 November 2024).**

This paper centres around creating a PCG-based platformer where the levels are procedurally generated, however the player has a form of choice over what the parameters of the next level will look like through the mechanics of the game. It has a very strong background section which gives a good base for finding research on what games are currently out there which use PCG. Primarily, the idea of games not being either PCG-based or non PCG-based, but instead falling on a spectrum, may be an interesting point to bring up in the background section of my own paper. The paper doesn’t go in-depth of the method of creating the project, so it will mostly be useful for the theory and context sections of my research.

***Ahmed, S. and Pandey, B. (2024). View of Procedural Terrain Generation by Sampling a 2D Monochrom Perlin Noise Map in Unity*. Available at:** [**https://journalajrcos.com/index.php/AJRCOS/article/view/333/662**](https://journalajrcos.com/index.php/AJRCOS/article/view/333/662)**. (Accessed: 2 November 2024).**

This journal article investigates using a 2D Perlin Noise function to generate terrain in Unity. The performance analysis section appears that it could be very useful as it contains two tables which show how the grid size can affect loading times and memory usage. This could be important as it is something I should bear in mind when considering the sizes of levels I should allow in the user parameter settings. It also puts forth an interesting idea, using parallelisation in my level generator, which I may research further if time permits.