Adaptive Procedural Generation in Minecraft

Blake Patterson, Michael Ward

CIS 5603 Final Project, Dr. Pei Wang
Department of Computer & Information Sciences
Temple University
May 3, 2022

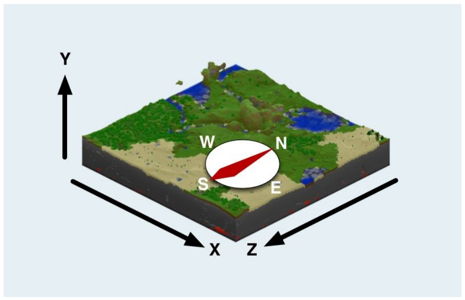
Contents

1 Abstract

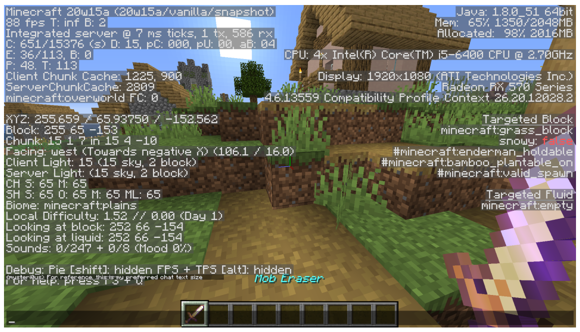
Abstract here

2 Introduction

Minecraft is a sandbox video game, created by Mojang in 2009, where players explore and build in a procedurally generated 3D grid-like world with infinite terrain. The main game-play element of Minecraft consists of collecting various types of materials and using them to build tools and structures. The world is divided into 1x1x1 blocks which can vary in material, spawning location, and usability. Aside from the popularity of the base game, Minecraft has become well known for it's customization possibilities through a variety of open source application program interfaces. These application program interfaces allow players to modify textures and color palettes, add new items, block types and enemies, and more.



(a) Grid



```
Minecraft 20w15a (20w15a/vanilla/snapshot)
88 fps T: inf B: 2
Integrated server @ 7 ms ticks, 1 tx, 586 rx
C: 651/15376 (s) D: 15, pC: 000, pU: 00, aB: 04
E: 36/113, B: 0
P: 48. T: 113
Client Chunk Cache: 1225, 900
ServerChunkCache: 2809
minecraft:overworld FC: 0

XYZ: 255.659 / 65.93750 / -152.562
Block: 255 65 -153
Chunk: 15 1 7 in 15 4 -10
Facing: west (Towards negative X) (106.1 / 16.0)
Client Light: 15 (15 sky, 2 block)
Server Light: (15 sky, 2 block)
CH S: 65 M: 65
SH S: 65 O: 65 M: 65 ML: 65
Biome: minecraft:plains
Local Difficulty: 1.52 // 0.00 (Day 1)
Looking at block: 252 66 -154
Looking at liquid: 252 66 -154
Sounds: 0/247 + 0/8 (Mood 0%)

Debug: Pie [shift]: hidden FPS + TPS [alt]: hidden
<mysteri0us> For reference, this is my preferred chat text size
For help: press F3 + Q
```

```
                        Java: 1.8.0_51 64bit
                     Mem:  65% 1350/2048MB
                     Allocated:  98% 2016MB

    CPU: 4x Intel(R) Core(TM) i5-6400 CPU @ 2.70GHz

          Display: 1920x1080 (ATI Technologies Inc.)
                             Radeon RX 570 Series
4.6.13559 Compatibility Profile Context 26.20.12028.2

                                   Targeted Block
                         minecraft:grass_block
                                     snowy: false
                 #minecraft:enderman_holdable
               #minecraft:bamboo_plantable_on
                        #minecraft:valid_spawn

                                   Targeted Fluid
                                 minecraft:empty
```

(b) Developer Overlay

Figure 1: Observable Minecraft Environment

The open source nature and in-game environment of Minecraft has also caught the attention of artificial intelligence researchers. The environment of Minecraft is ideal for research in AI because of the endless possibilities, from training an agent on simple tasks like searching for a specific object or material, to building complex structures or navigating obstacle courses. Since the environment is divided into a three-dimensional grid of equal sized cubes, it is also easy to measure and evaluate the performance of AI in Minecraft.

This project is focused on the application of AI for Procedural Content Generation (PCG) within Minecraft. PCG is defined as the algorithmic creation of game content with limited or indirect user input [11]. Content in the context of PCG can be described as most of what can be contained within a game including maps, rules, textures, items, quests, music, characters, and more [11]. Many popular games have made use of PCG including Rogue, Dwarf Fortress, Diablo, Spore, Civilization, Spelunky, as well as Minecraft itself [11]. The usage of PCG varies from game to game and can range from fully autonomous game design, to automating routine or common aspects of game design. One major critique of PCG in game design has to do with repetition and functionality; rule-based agents are likely to create good looking and functional content that looks similar, and search-based agents are likely to create more diverse content, but takes more time and resources to ensure that it is functional for the player [5].

Most instances of PCG in video games operate from a 'clean-state' where the generator does not have to consider interaction with preexisting in-game elements [5]. Exploring PCG within Minecraft opens up a new challenge within AI, in which the goal is to produce a functional and believable village settlement that adapts to different environments within a Minecraft map [9]. Instead of generating a village on a clean slate, this problem requires the generator to account for the presence of preexisting game elements and focuses on adaptive generation of artifacts [5]. A map in Minecraft is made up of various biomes which contain different types of terrain, elevation gradients, fauna, and bodies of water. In order for a procedurally generated settlement to be functional and believable, it must be adaptive and able to build on top of and in response to elements that already exist in the Minecraft environment. The Generative Design in Minecraft (GDMC) AI settlement generation competition initially proposed this problem in 2018 [8]. GMDC has ran a yearly open competition for researchers and students to submit their algorithm, which is scored by a panel in terms of the algorithms adaptability and functionality [4].

We propose to develop a Procedural Content Generation AI that is capable of generating a functional and believable Minecraft settlement, which is adaptive to varying environmental factors. Based on our review of literature, it is apparent that developing multiple different algorithms to handle individual pieces of the problem has led to better outcomes in previous research.

3 Related Work

3.1 PCG in Games

Procedural generation of dungeons [14].

Survey of pcg dungeon creations [15].

Search based procedural content generation [13].

Declarative procedural generation of architecture with semantic architectural profiles [1].

Procedural content generation via reinforcement learning [7].

3.2 PCG in Minecraft

Multi agent settlement gen in minecraft [3].

World GAN [2].

Growing artefacts and machines with neural cellular automata in minecraft [12].

Green et al. generate floor plans using a constrained growth algorithm and cellular automata [5].

4 Methodology

4.1 Interface Mod and Python Client

Talk about how mod and client are used

4.2 Terraforming

Talk about terraforming

4.3 Plot Analysis

Talk about plot analysis

4.4 Modified A*

Talk about A*

4.5 House building

Talk about house building

5 Discussion

5.1 Results

5.2 Limitations

6 Acknowledgments

We'd like to thank the contributors and maintainers of both the Minecraft HTTP Interface Mod and Generative Design Python Client: Nils Gawlik, Claus Aranha, Mayank Jain, and Flashing-Blinkenlights. This project would not be possible without the tools they built that allow for interacting with Minecraft via Python scripts.

7 References

[1] Levi van Aanholt and Rafael Bidarra. "Declarative procedural generation of architecture with semantic architectural profiles". In: *2020 IEEE Conference on Games (CoG)*. IEEE. 2020, pp. 351–358.

[2] Maren Awiszus, Frederik Schubert, and Bodo Rosenhahn. "World-GAN: a Generative Model for Minecraft Worlds". In: *2021 IEEE Conference on Games (CoG)*. IEEE. 2021, pp. 1–8.

[3] Albin Esko and Johan Fritiofsson. *Multi-Agent Based Settlement Generation In Minecraft.* 2021.

[4] Marcus Fridh and Fredrik Sy. "Settlement Generation in Minecraft". In: (), p. 55.

[5] Michael Cerny Green, Christoph Salge, and Julian Togelius. "Organic Building Generation in Minecraft". In: *arXiv:1906.05094 [cs]* (June 11, 2019). arXiv: `1906.05094`. URL: `http://arxiv.org/abs/1906.05094` (visited on 02/24/2022).

[6] Jean-Baptiste Hervé and Christoph Salge. "Comparing PCG metrics with Human Evaluation in Minecraft Settlement Generation". In: *arXiv:2107.02457 [cs]* (July 6, 2021). arXiv: `2107.02457`. URL: `http://arxiv.org/abs/2107.02457` (visited on 02/24/2022).

[7] Ahmed Khalifa et al. "Pcgrl: Procedural content generation via reinforcement learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment.* Vol. 16. 1. 2020, pp. 95–101.

[8] Christoph Salge et al. "Generative Design in Minecraft (GDMC), Settlement Generation Competition". In: *Proceedings of the 13th International Conference on the Foundations of Digital Games* (Aug. 7, 2018), pp. 1–10. DOI: `10.1145/3235765.3235814`. arXiv: `1803.09853`. URL: `http://arxiv.org/abs/1803.09853` (visited on 02/24/2022).

[9] Christoph Salge et al. "Generative Design in Minecraft: Chronicle Challenge". In: *arXiv:1905.05888 [cs]* (May 14, 2019). arXiv: `1905.05888`. URL: `http://arxiv.org/abs/1905.05888` (visited on 02/24/2022).

[10] Christoph Salge et al. "The AI Settlement Generation Challenge in Minecraft: First Year Report". In: *KI - Künstliche Intelligenz* 34.1 (Mar. 2020), pp. 19–31. ISSN: 0933-1875, 1610-1987. DOI: `10.1007/s13218-020-00635-0`. URL: `http://link.springer.com/10.1007/s13218-020-00635-0` (visited on 02/24/2022).

[11] Noor Shaker, Julian Togelius, and Mark J Nelson. *Procedural content generation in games.* Springer, 2016.

[12] Shyam Sudhakaran et al. "Growing 3d artefacts and functional machines with neural cellular automata". In: *arXiv preprint arXiv:2103.08737* (2021).

[13] Julian Togelius et al. "Search-based procedural content generation: A taxonomy and survey". In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3 (2011), pp. 172–186.

[14] Roland Van Der Linden, Ricardo Lopes, and Rafael Bidarra. "Procedural generation of dungeons". In: *IEEE Transactions on Computational Intelligence and AI in Games* 6.1 (2013), pp. 78–89.

[15] Breno MF Viana and Selan R dos Santos. "A survey of procedural dungeon generation". In: *2019 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames).* IEEE. 2019, pp. 29–38.