# MGAI Assignment 01

Procedural Content Generation

Wei Chen

## 1 Introduction

In this task, a temple (or house) is built (see figure 1) in the game Mine craft, based on GDMC HTTP-API Mod, and the construction is inspired by the style of Aztec architecture, or more precisely Aztec temple.

To cope with game-generated randomness thereby ensuring believability as much as possible, procedural content generation, a method of generating a large amount of content, is introduced.

**Figure 1: Temple in the Aztec architectural style. This temple is constructed algorithmically as opposed to manually.**

## 2 Believability

There are two main algorithms in my code to believably ensure the location of construction.

The first algorithm is to avoid building a temple on any water blocks. It will be called when start choosing a fitting place for construction in a given range of "build area". For example, once the uppermost blocks of a given "build area" are mostly water blocks, it is likely that a place will not be found to build a temple that covers a relatively large area. The idea behind this algorithm is more of a brute-force search that firstly iterates over the coordinates (x and z) of the "build area" and stops until a water block is scanned or the coordinates of all appropriate places are fully stored (for randomness). It is implemented with a nested loop (4 loops) that scans the whole environment and thus takes some time. For example, it can be seen from figure 2 that all of the appropriate locations are on the right shore or at the feet of the current player. After executing a whole run, a temple is perfectly built in the right place (see figure 3), which proves the effectiveness of this algorithm.

The second algorithm is to clear the blocks at a certain distance above the building when building. It serves to allow the temple, including the interior decoration, in the middle of the jungle or mountains not to be completely obscured (see figure 4 and 5). The logic behind the implementation was to replace 1 or 2 block(s) above each block with air and then remove all blocks from its interior in advance before building the top-level temple. Therefore, the terrain of the selected place will not change much.

**Figure 2: The game screen before the selection of the location, where the ocean above is not considered.**

**Figure 3: The game screen after the selection of the location, where one (top right corner of the screen) of the appropriate places is successfully detected.**

## 3 Randomness

Randomness is taken into account, by request. Width along with length will be randomly chosen (they are the same size), while the height of the top-level temple is also randomly generated. In addition, for the first algorithm that detect the fitting places within given "build area", parameter "n_places" can be tuned to shorten the time to scan the environment. For example, $n\_places = 5$ indicates stopping the scanning process once there are 5 appropriate places detected. Then, a place to be built is randomly selected from these five. Therefore, the final location of the temple is random as well.

**Figure 4: Building temple in jungle.**

**Figure 5: Building temple in mountains.**

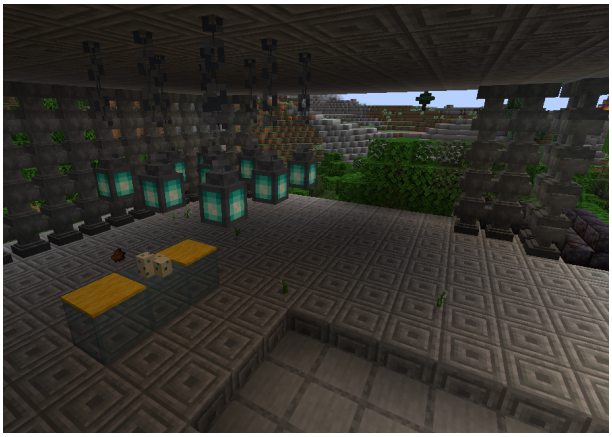

**Figure 6: Interior decoration**

## 4 Interior decoration

See figure 6 for interior decoration.

## 5 Conclusion

The gdmc_http_client_python library (version 4.2) provide many convenient and helpful functions. However, there are still many functions (or methods) that allow for more brilliant ideas (e.g., allow for placing entities, such as blaze man, zombie, etc).

For the second algorithm described above, there are still many improvements to be made. For example, the entrance to the temple will always face the lower slopes of the mountain range so that it will not be hidden in the mountain.

## References