Complex Game System: Assessment Brief

By Ben Scott

**Design Stage**

My system will be a Procedural Dungeon Generator, being modular the developer will be able to edit the frequency, width and Length of hallways, rooms and textures.

The purpose of this system is to create easier map generation tool for a developer who wants assistance in creating random dungeon rooms. think of the game ‘binding of Isaac’ where each room is different every time you play. I’m wanting to create it easier for the developer to have the player enjoy a random experience each time they play.

Third party libraries that my system relies on: **(delete if empty later).**

**What is involved**

The mathematical operations involved with my procedural generations will be Binary Space partitioning (BSP). Simply put, BPS recursively subdivides a space into two convex sets in a tree like data structure also known as a BSP Tree. BSP is more commonly used to gather spatial information about objects in a scene and assist in rendering such as being able to move objects around and rotated based on the player’s position. I’m using BSP to find information about the world space, using the tree like data to divide the world into parts, each part being a node of the tree, it would be given random value from unity’s Random.Range() functions inside a limit the developer will set to determine the size and amount generated.

Chart, box and whisker chart

Description automatically generated

The algorithm that is required to create the procedural generator is a BSP Tree. Using BSP means you can guarantee the generation will be evenly spaced and that they will not overlap. This makes it easier to connect the rooms with hallways.

**Step 1** - To construct the BSP Tree you need to create a ‘leaf’ class which in the game will be represented by a pair of rooms with a separator line.

**Step 2** - After the leaf’s have been defined, we can create the rooms by starting from the biggest root and traversing down to the smallest leaf that has no children.

**Step 3** -We then remove the separator lines so we can visibly see the generated rooms. we continue doing this to create each room.

**Step 4** - To create the hallways, we pick a random point inside both rooms by getting a random value from the range of with width and height of the room by using Random.Range() function inside unity or both siblings inside the tree and generate the hallway.

**Algorithm To Construct the tree**

* Start with an initial dungeon size, this is the Container, example 64x64 tiles.
* Do dot point 3 below to get a predefined N of iterations
* For each leaf of the current container:
  + - Using Random.Range() from 1-2 to choose a direction: vertical or horizontal
    - Cut the current container in half in the selected direction.
* For each leaf container, pick a smaller container with a random smaller area, this smaller area inside will be the room.
* Connect the centres of the non- leaf containers.
* Generate each tile/room and walls accordingly.

Graphical user interface

Description automatically generated

**Designed to be Modular**

The project is planned to be easy for another developer to use. I’ll use multiple scripts containing each system rather than one giant script containing everything for example, if I had a player script, I wouldn’t put the movement and abilities all in the one script. I would use a script for each system like a keyboard movement script, taking damage script. Doing it this way gives developer has full control and easier readability. All scripts will contain public values rather than hard coded values, this means the developer can change almost every value from dungeon heights to widths and even textures.

[insert class diagram of the BSP/scripts]

**Packaging and integration**

I’ll be using unity’s UMP package manager, I will be following the unity package convention and hosting it on GitHub as a compressed file. you can acquire the package using the package manager in unity. There is an option inside the unity’s pack manger where you can use a GitHub URL to download the package. The only downside of doing it this way is the person has to have git installed on their computer.

Creating a custom package in unity means I can distribute new features and updates to existing features quickly and easily, doing this provides a platform for developers to discover and share reusable components.

To test if it works, I’ll be creating a brand-new project, importing my dungeon generator package and testing all scripts and demo scenes to make sure its modular and in working condition.

I will implement a demo scene that shows what it does and how to use it with read me files and text on the demo scene.