Horror Lab by Camila Mattos

System for Procedurally Generated Map

High Concept

Horror Lab is a roguelike top-down turn-based strategy RPG where you play as Yuri Sakamoto, a girl who wakes up from a coma to find herself being used as a guinea pig in the Tukugima Pharmaceutical Labs, where they`ve been experimenting with their new drugs on her and other prisoners for a social experiment. If one feels pain, all of them feel pain. All share the same HP. Now, she must escape with the help of her cellmates, a group of people who would never have helped each other in other circumstances.

Player`s Goal: During the game, players will go through the Tukugima Pharmaceutical Labs and its several **floors** in order to achieve their final goal: escape captivity.

Core Loop: Each **floor** is composed of a **map** made of several interconnected **rooms** the player must traverse to reach the **final room**, where they must face a **boss battle**, and complete the level. After completing a floor, players may come back to the **Prison Floor** (the only floor that isn`t procedurally generated) to release new prisoners who can join them in battle, complete quests, and level up. After that, they may continue to the next floor.

Battle: The player advances through the top-down map by defeating enemies such as guards and mad scientists that want to prevent their escape. The battle happens in a Xcom fashion, where the player assembles a team to fight enemies in a strategic turn-based combat.

Movement: Enemies and Characters can move front, backward, left, or right on a square grid placed on the floor, as shown in the image above.

Procedural Emergence

Goal of the system

As with any roguelike, the game has a procedurally generated map that is created based on a set of rules every time the player enters a new floor. However, since the game also has elements of strategic turn-based combat, where the enemy spawns and how far they appear on the grid from the character's spawn points are also important for the system.

The goal of the system is to create rules that will help generate a unique procedural generated map that can not only create replayability but also fits the genre of the game giving the player meaningful choices when it comes to strategic thinking in combat. This way, the player will never play the same map twice, but will also feel the map always fits the theme and gameplay style because it will always follow a set of guidelines.

Rules of the system

The creation of the map can be achieved by stating the rules that define the size of the map and rooms, how it can be populated, how the rooms connect between themselves, and what triggers the level to end and begin.

First, I`ll start by stating some rules that the system must follow and understand in order for it to work.

- **Map:** The system will create a map. A map is a collection of rooms connected to other rooms.
- **Room:** A room is a rectangular-shaped area where assets, characters, and enemies can spawn. Every room can be connected to other rooms.

Other Tunable Variables**: Enemies, Cover, Platforms, Character Spawn Point.**

Map

First, let's define the size of the map. Since the map is made from a collection of rooms, that can be defined by the number of rooms that make a map. Therefore, I`m creating the following rule: the map minimum size is 10 rooms and its maximum size is 15 rooms.

Rooms

Now let's state the rules of how the rooms work. First, we must establish how they interconnect with each other, as follows:

- **Room connections:** All rooms must be connected to at least one other room and a maximum of four others.

Now, we must establish the size of the rooms. Considering the characters and enemies will occupy a 1x1meter square, we can consider that our smallest measure and define that every room will have a $1m^2$ square grid.

Once our measure has been decided, it is important to decide how many different sizes of rooms the map will have, and the frequency they will appear. I opted for 3 sizes, as shown below.

- **Room Sizes:** Small, Medium, Large.

Name	Small	Medium	Large
X axis (min/max)	15 meters / 25 meters	25 meters /35 meters	35 meters /45 meters
Z axis (min/ max)	15 meters / 25 meters	25 meters /35 meters	35 meters /45 meters
How often they appear	25% of the rooms on a map	Approximately 50% of the rooms on a map	25% of the rooms on a map
Connections	Cannot be connected to another small room	---	Cannot be connected to another large room

OBS: If the exact number can`t be found, make the leftover of the equation a medium room, unless the number of rooms is perfectly divided by 2, then make the leftover room a small room.

Now, the room type must also be defined.

Room Type	Normal Room	Start Room	End Room	Key Room	Treasure Room	Buff Room
Rules		The Floor (level) starts in this room	Always has a boss fight. No other rooms have boss fights.	Has a key that is placed on the opposite side of the spawn points, after the last enemy. The key opens the End Room.	Has a treasure that is placed on the opposite side of the spawn points, after the last enemy.	Gives a buff to the player after all enemies are eliminated.
Size	Can be any size	Small	Always large	Can be any size	Medium or Large	Medium or Large
How often they appear	50% of rooms (After taking out 3 for start, end and key) – If an exact number can't be found, make the leftover room into normal room	1 every floor	1 every floor	1 every floor	25% of rooms (After taking out 3 for start, end and key)	25% of rooms (After taking out 3 for start, end and key)
Connections		At least 5 rooms of distance of the End Room, maximum of 8 rooms of distance.	Can only be opened with a key.	At least 3 rooms of distance of the End Room and the Start Room.	Never connected to a buff room or treasure room.	Never connected to a buff room or treasure room.

Populating the Rooms

Once the room type is defined, we must define how to populate them. Considering what is important in the game, we can create the following rule: every room must contain four Character Spawn Points for the player to place characters, cover for both enemies and playable characters to mitigate damage, hide and shoot through, and platforms for them to climb and shoot with advantage. Also, all rooms must contain enemies.

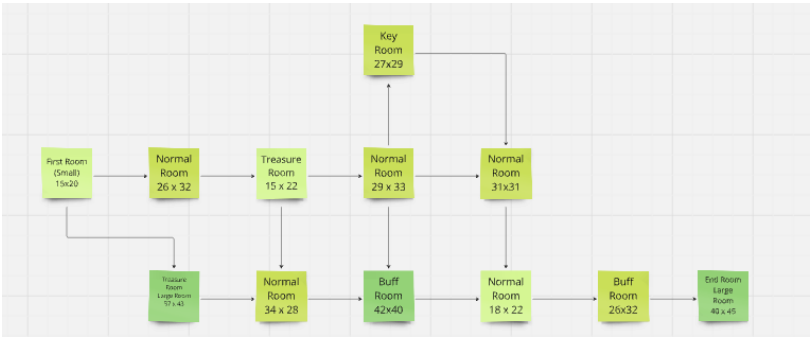
Everything that appears on the level must respect a distance from other objects in order to avoid more than one object spawning on top of each other. So, one general rule must be: Never spawn an asset on the scene if there is another asset there. Other specific rules can be:

Assets	Cover	Platform	Character Spawn Points	Enemies
Rules	Every $5m^2$ has one cover at least All enemies spawn near the cover	**Ladder:** Always have a ladder connected to it. The ladder is always facing the center of the room and has an empty square in front of it, allowing it to be accessible. **Function:** Other objects may spawn on top, except from platforms and character spawn points	They always appear on the bottom side of the room with only 2 squares below them.	At least 1 enemy every $5m^2$ and a maximum of 3 per 5 m^2 in a room 25% chance to spawn in a platform
Size	Min: 1 x 1 meter	Min: 1x3 meters Max: 4x4 meters	1x1 always	1x1 when normal type / 3 x 3 when Boss (only in End Room)

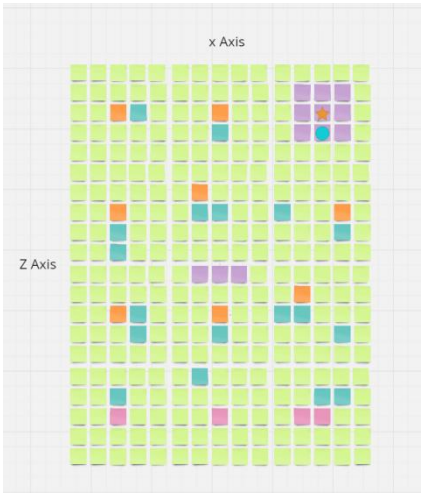
	Max: 1 x 2 meters			
Distance between other objects in the scene	At least 2 squares of distance from another object (NOT including Character Spawning Points, including other covers) **Edges:** min 1 square of distance from the edges.	At least 2 squares of distance from another object (unless the object spawns on top of it) **Edges:** Min 1 square of distance **Other Platforms:** Min 4 squares away	**Character Spawn Points:** A maximum of 2 Spawn Points can be spawned side by side. Others must keep a 2 square distance. **Edges:** Min 2 squares of distance **Platforms:** Min 2 squares of distance. (cannot be spawned on top)	**Edges:** At least 1 square away from the edges **Character Spawn Points:** Min 3 squares away
Amount in each room	The same as the enemies + 4 more	2 / 5	4 always	It takes into consideration the room size: ((z axis– 5) /5) x (x axis/5) = number of enemies in a room If not an integer, ignore the leftovers.

Walkthrough

Map Size: Now let`s go into one example. First things first, once the player enters a floor, a new map loads. Then, a number between 10 and 15 is selected for the number of rooms that will compose the map. In this example, the map has 12 rooms. All maps will have: 1 end room, 1 start room, and 1 key room. That`s a total of 3. In the example, we have 9 other rooms that still need a room type (12 – 3). From the 9 rooms left: 50% (and any leftovers from the equation) will be normal, which means 5 rooms (9/2 = 4,5). The other 25% will be Buff Rooms and the rest (25%) will be Treasure rooms. That means 2 for each. After that, the system selects the room size, of which 50% must be medium, which means 6 rooms out of 12 (0,5 x 12 = 6) in our example. As for small and large, they are 25% of every map, so 3 each (0,25 x 12 = 3). Each room must also select a number for width and length between the max and min for their size type (small, medium, or large). Look at the flowchart below for the numbers generated for each room.



Room Connections: Now, the rooms must connect with each other, starting with the First Room, which in this example is 6 rooms away from the End Room, following the parameters. And the Key Room which is 3 rooms away from both the End Room and the First Room. Looking at the flowchart, you can also observe no buff and treasure rooms are connected to each other. The same goes for small and large rooms.



Populating the Map: Now I`ll show an example of the first room being populated. - Every square in the image on the side is a 1x1 meter square on the grid.

- **Character Spawn Points (Pink on the image):** They appear in the bottom part of the room, 2 squares away from the edges. Only 2 of them appeared together as stated in the rules, others stay at a distance of 3 or four squares.
- **Enemies (Orange):** To determine how many enemies would appear, the following equation was necessary: ((z axis– 5) /5) x (x axis/5) = number of enemies in a room
- X axis = 15 and z axis = 20 ((20– 5) /5) x (15/5) = (15 /5) x 3 = 3 x 3 = 9 enemies
- **Cover (Blue):** The cover is the number of enemies (9) + 4 which is 13. All enemies appear near one.
- **Platform (Purple)** All levels have at least 2 platforms, and that is the case of the example. It also follows the guidelines of keeping at least 2 squares of distance from other assets in the scene.