Tarea 4 - Informe

Integrantes: Romina Villagra y Fernanda Catalán

Generación Procedural de Mapas basado en Stardew Valley usando Binary Space Partitioning

1. Abstract

Este informe presenta el desarrollo de un sistema de generación procedural de mapas, en este caso inspirado en las minas del videojuego Stardew Valley, utilizando el algoritmo Binary Space Partitioning (BSP). El objetivo principal fue replicar la estructura dinámica y rejugable de las minas del juego original, mediante la subdivisión recursiva del espacio en salas conectadas. Se implementaron múltiples mejoras al algoritmo tradicional, incluyendo la distribución temática de recursos, enemigos y objetos según el nivel de profundidad. Además, se ajustaron hiperparámetros clave como la profundidad de partición y el tamaño mínimo de salas para optimizar la variedad y jugabilidad. Los resultados muestran que el sistema genera mapas coherentes, diversos y reproducibles, lo que valida la efectividad del enfoque aplicado para entornos explorables en videojuegos.

2. Introducción

La generación procedural de mapas es una técnica clave en muchos videojuegos actuales. En este proyecto quisimos recrear parte de Stardew Valley, enfocándonos especialmente en la estructura de sus minas, ya que estas presentan un diseño generado dinámicamente. Nuestra meta fue replicar ese sistema utilizando el algoritmo Binary Space Partitioning (BSP).

2.1. Definición y descripción del juego

Stardew Valley es un juego de rol y simulación de vida en una granja, donde el jugador hereda un terreno descuidado y lo transforma con trabajo y dedicación. Más allá del cultivo y la pesca, el juego tiene un sistema de minas el cual se genera de forma procedural, ofreciendo variedad en cada partida.

2.2. Justificación del juego escogido

Se escogió Stardew Valley para recrear por su uso de generación procedural en las minas, promoviendo rejugabilidad y exploración. Por lo que replicar este comportamiento permite estudiar técnicas de diseño de mapas dinámicos.

2.3. Justificación del algoritmo

El algoritmo Binary Space Partitioning (BSP) permite dividir recursivamente un área en regiones más pequeñas donde se pueden colocar salas, conectadas por corredores. Su simplicidad para controlar el tamaño y distribución de áreas es útil para diseñar entornos como minas, esto lo hace ideal para este proyecto. Se puede combinar con ruido Perlin o algoritmos de colocación de objetos para mayor naturalidad.

3. Estado del Arte

Al revisar otros juegos con mapas generados proceduralmente, encontramos varios ejemplos útiles. Uno de ellos fue The Binding of Isaac, que usa una variación de BSP para construir habitaciones conectadas al azar. Otro es Diablo, famoso por sus niveles subterráneos generados dinámicamente. Estos juegos nos sirvieron como referencia para entender distintas formas de implementar generación procedural y justificar nuestra elección del algoritmo BSP.

3.1. Referencias Relevantes

- **The Binding of Isaac**: Usa variantes de BSP para crear habitaciones conectadas de forma aleatoria.
- **Diablo**: Niveles subterráneos generados proceduralmente con pasillos y salas interconectadas.

4. Metodología

El desarrollo del generador procedural de mapas se realizó a través de una implementación en C++ del algoritmo Binary Space Partitioning (BSP), adaptado específicamente para simular los niveles de mina del juego Stardew Valley.

4.1. Implementación del algoritmo

La implementación consiste en dividir un mapa rectangular en subregiones usando particiones recursivas, hasta alcanzar un límite de profundidad. A cada región final se le asigna una habitación rectangular que se conecta a las demás mediante túneles ortogonales en forma de "L". Una vez creado el mapa base, se decoran las habitaciones con elementos del entorno: piedras, minerales, enemigos, barriles, cofres y escaleras.

Los elementos decorativos varían según el nivel de profundidad de la mina:

- Cada 10 niveles se genera un piso especial con un cofre y sin enemigos.
- En los demás niveles, se colocan objetos con distribución aleatoria y condicional.

Los minerales se colocan sobre celdas de piedra y su aparición depende del nivel de la mina, simulando la progresión de dificultad del juego original.

4.2. Parámetros

Se definieron los siguientes hiperparámetros como modificables:

- **Profundidad máxima de partición (maxDepth)**: Controla el número de subdivisiones del mapa. Se utilizó un valor de 4 para asegurar variedad sin generar salas demasiado pequeñas.
- **Tamaño mínimo de subregiones**: Se evita subdividir áreas menores a 10x10 para asegurar que cada sala sea funcional.
- **Nivel de mina** : Determina el contenido de los mapas. Se usa como semilla para generar variación entre niveles.
- **Distribución de minerales**: Depende del nivel. Por ejemplo:
  - Niveles bajos: Más cuarzo y cobre.
  - Niveles intermedios: Predominio de hierro y jade.
  - Niveles altos: Oro, jade y diamante.
- **Porcentajes de elementos**: Se colocan objetos en base al área de la sala:
  - 50 % de las celdas con suelo se reemplazan por piedra.
  - 7 % del área con piedra recibe minerales.
  - 5 % del área total se usa para enemigos.
  - 3 % para barriles.

4.3. Proceso de ajuste de parámetros

Para determinar valores adecuados, se empleó una búsqueda simple basada en múltiples ejecuciones con semillas aleatorias (niveles 1 a 120). Se observaron mapas generados para ajustar los valores de maxDepth y las proporciones de aparición de objetos, manteniendo un equilibrio entre variedad, jugabilidad y estética visual.

4.4. Diferencias con el algoritmo

A diferencia de implementaciones tradicionales del algoritmo BSP, se introdujeron varias adaptaciones:

- Decoración temática de salas según el nivel del juego.
- Uso de objetos adicionales como cofres, barriles y escaleras.
- Distribución condicional de minerales sobre piedras y en función del nivel, simulando progresión.
- Piso especial sin enemigos cada 10 niveles, inspirado en la estructura real del juego.

Estas variaciones buscan aproximarse más a la experiencia de exploración que propone Stardew Valley y contribuyen con una visión más modular y extensible de BSP aplicada a entornos complejos.

5. Resultados

En esta sección se presentan los mapas generados mediante el algoritmo de generación procedural implementado.

El algoritmo procede a dividir el espacio del mapa en particiones verticales u horizontales de forma alternada y recursiva, asegurando que las dimensiones mínimas se respeten para crear salas funcionales. Posteriormente, se conectan estas salas mediante pasillos y puertas generadas aleatoriamente siguiendo las probabilidades establecidas.

A continuación se muestran ejemplos de mapas generados con diferentes ejecuciones del algoritmo, ilustrando la variabilidad que aporta la aleatoriedad en el diseño:

- Coherencia y conectividad del mapa.
- Diversidad espacial y visual.
- Porcentaje de uso del espacio.

5.1. Parametros utilizados:

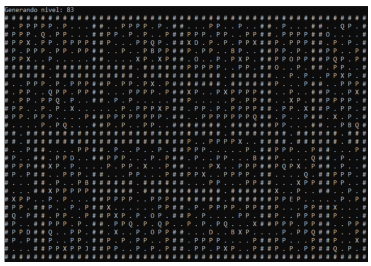

Figura 1: *
Nivel 83

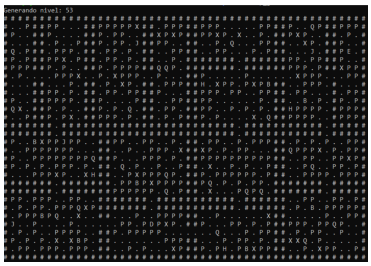

Figura 2: *
Nivel 53

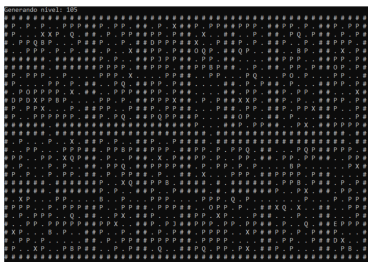

Figura 3: *
Nivel 105

Figura 4: Comparación de niveles con el mismo parámetro de subdivisión. **Weight: 50, Height:30, Depth:4**

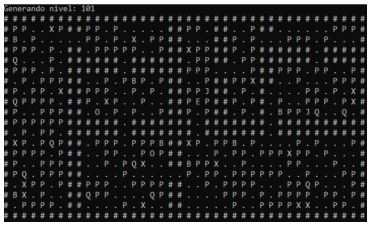

Figura 5: *
Nivel 101

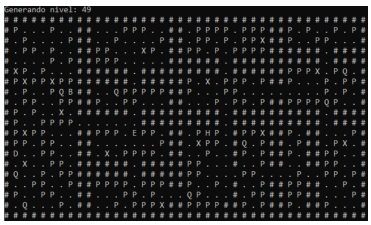

Figura 6: *
Nivel 49

Figura 7: *
Nivel 105

Figura 8: Comparación de niveles con el mismo parámetro de subdivisión. **Weight: 40, Height:20, Depth:4**

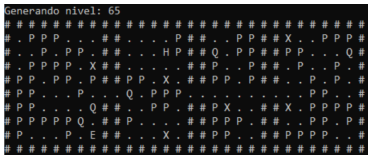

Figura 9: *
Nivel 65

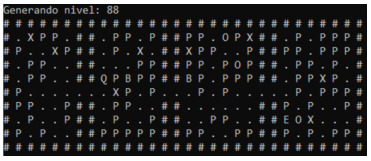

Figura 10: *
Nivel 88

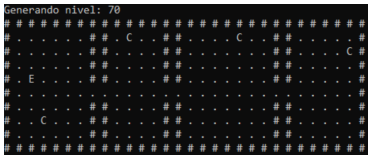

Figura 11: *
Nivel 70

Figura 12: Comparación de niveles con el mismo parámetro de subdivisión. **Weight: 30, Height:10, Depth:4**

Estos resultados evidencian que el algoritmo es capaz de generar diferentes configuraciones de mapas respetando las restricciones de tamaño y conexión, proporcionando diversidad para la experiencia de juego.

Para replicar estos resultados, se recomienda implementar el algoritmo BSP con los parámetros mencionados, utilizando un generador de números aleatorios con control de semilla para reproducir las configuraciones específicas de los mapas.

Elementos del juego y su representación visual:

- Cuarzo = Q.
- Jade = J.
- Diamante = D.
- Cobre = C.
- Hierro = H.
- Oro = O.
- Piedra = P.
- Enemigo = x.
- Escalera = E.
- Barril = B.
- Cofre = C.

6. Conclusión

En este trabajo se implementó un algoritmo de generación procedural basado en Binary Space Partitioning (BSP) que genera mapas compuestos por salas interconectadas mediante túneles. Los mapas cumplen con las restricciones de tamaño mínimo y presentan conexiones que garantizan la transitabilidad.

Los resultados demuestran que, con una correcta parametrización, es posible obtener diversidad en la estructura de los mapas, lo que aporta variabilidad para la experiencia de juego.

Además, la utilización de semillas para controlar el generador de números aleatorios permite reproducir mapas específicos, facilitando la evaluación y ajuste del algoritmo.

En conclusión, el método aplicado es efectivo para generar mapas funcionales y variados, constituyendo una base sólida para futuros desarrollos en generación procedural de niveles.

7. Referencias

- Smith, G. (2015). Procedural Content Generation: An Overview. En S. Rabin (Ed.), Game AI Pro 2: Collected Wisdom of Game AI Professionals (pp. 501–518). CRC Press. (Proporcionado por el profesor Nicolás Barriga).
- GeeksforGeeks. (2020). Binary Space Partitioning. GeeksforGeeks.
  https://www.geeksforgeeks.org/binary-space-partitioning/
- Valve Developer Community. (2024). Binary Space Partitioning.
  https://developer.valvesoftware.com/wiki/Binary_space_partitioning
- monogame. (2020). Procedurally generated rooms (Binding of Isaac style).
  https://community.monogame.net/t/procedurally-generated-rooms-binding-of-isaac-style/16019
- Fandom. (n.d.). Procedural Generation. Diablo Archive Wiki.
  https://diablo-archive.fandom.com/wiki/Procedural_Generation