EXAMEN FINAL COM300: Generador Procedural de Ecosistemas 3D con IA

Por: Ojeda Jose Alex

1. Introducción

1.1 Motivación

La generación procedural de ecosistemas en 3D aborda la necesidad de crear entornos virtuales ricos, diversos y realistas sin intervención manual exhaustiva. En campos como los videojuegos, la simulación científica y la visualización arquitectónica, modelar paisajes naturales y distribuir flora y fauna de manera coherente resulta complejo y consume mucho tiempo. Gracias a los algoritmos de IA no supervisada, como Growing Neural Gas y K-Means, podemos aprender directamente de datos topológicos y espaciales para:

- **Capturar la forma del terreno** (montañas, valles, llanuras) sin definir manualmente cada vértice.
- **Organizar la distribución de la vegetación** y otros elementos biológicos en función de patrones extraídos del propio terreno.
- **Escalar y adaptar** rápidamente la generación a distintos tipos de ecosistemas (bosques, manglares, arrecifes) variando pocos parámetros.

Este enfoque reduce drásticamente el esfuerzo de diseño, permite explorar grandes volumetrías de forma eficiente y abre la puerta a simular procesos ecológicos complejos de forma automática.

1.2 Objetivos

- Diseñar un sistema en Blender que genere paisajes y vegetación automáticamente.
- Aplicar algoritmos de aprendizaje no supervisado (GNG y K-Means) para modelar la topología y diversidad biológica.
- Integrar todo mediante scripts en Python dentro de Blender.

1.3 Recursos Empleados

A continuación, se detallan las herramientas, librerías y técnicas empleadas en el proyecto:

- **Lenguaje de Programación:** Python 3.8.

- **Blender 4.2.3 LTS:** plataforma principal para modelado, scripting y visualización 3D.
- **Bibliotecas Python:**
    - numpy: manejo de arrays y operaciones numéricas.
    - scikit-learn: implementación de K-Means (clustering) y preparación de datos.
    - json: lectura y escritura de archivos JSON para intercambio de datos.
- **Algoritmos de IA No Supervisada:**
    - **Growing Neural Gas (GNG):** para muestreo y aprendizaje de la topología del terreno.
    - **K-Means:** clustering de nodos para simular diversidad biológica.
- **Google Colab (Opcional):** entorno en la nube para ejecución de scripts de generación y clustering.
- **Git y GitHub:** control de versiones y alojamiento del repositorio del proyecto.
- **Assets 3D:** modelos de árboles y plantas descargados de repositorios libres (p. ej. Poly Haven, Free3D).

2. Fundamentos Teóricos

2.1 Growing Neural Gas (GNG)

Definición y características. Growing Neural Gas (GNG) es un algoritmo de aprendizaje no supervisado que construye una red de neuronas (nodos) y conexiones (aristas) que se adaptan gradualmente a la geometría y distribución de un conjunto de datos. Sus características principales incluyen:

- Aprendizaje incremental: los nodos se añaden dinámicamente según la densidad de datos.
- Topología adaptable: mantiene una estructura de grafo que refleja la forma del espacio de entrada.
- Sin presuponer número fijo de nodos: la red crece hasta cubrir adecuadamente la distribución de datos.

Algoritmo paso a paso.

1. Inicialización: crear dos nodos posicionados de forma aleatoria en el espacio de datos.
2. Entrada de datos: por cada iteración, seleccionar un punto de datos (o generar un dato sintético) que actúa como estímulo.
3. Competencia: identificar los dos nodos más cercanos al punto de datos (ganador y sub-ganador).
4. Ajuste de pesos: mover el nodo ganador y sus vecinos directos una pequeña fracción hacia el punto de datos.
5. Agregar aristas: conectar el ganador con el sub-ganador (o renovar la edad de la arista si ya existía).
6. Eliminación de aristas viejas: quitar aquellas aristas cuya edad supere un umbral para evitar conexiones inservibles.
7. Inserción de nuevos nodos: cada cierto número de iteraciones, añadir un nodo en la arista con mayor error acumulado.
8. Disminución de errores: ajustar los errores acumulados de nodos y aristas.
9. Repetición: continuar hasta alcanzar el número deseado de nodos o convergencia.

Ventajas y aplicaciones en generación procedural.

- Cobertura eficiente del terreno: adapta dinámicamente la densidad de nodos donde hay más complejidad (picos o valles).
- Flexibilidad: no requiere parametrizar el número exacto de elementos previo al entrenamiento.
- Simetría natural: la red generada se asemeja a patrones de crecimiento biológico (raíces, ramificaciones).
- Aplicaciones: modelado de paisajes, distribución de vegetación, diseño de redes de ríos o cavernas.

2.2 K-Means

Contexto de clustering no supervisado. K-Means es un algoritmo clásico de agrupamiento que divide un conjunto de datos en K grupos basándose en la proximidad en el espacio de características.

Funcionamiento básico: inicialización, asignación, actualización.

1. Inicialización: elegir aleatoriamente K centroides iniciales (o mediante técnicas como k-means++).

2. Asignación: cada punto de datos se asigna al clúster cuyo centroide esté más cercano.
3. Actualización: recalcular la posición de cada centroide como la media de los puntos asignados.
4. Iteración: repetir los pasos de asignación y actualización hasta que los centroides dejen de moverse significativamente.

Uso para clasificar nodos en tipologías biológicas.

- Entrada: las posiciones (x, y, z) de los nodos generados por GNG.
- Proceso: K-Means etiqueta cada nodo con un entero [0, K−1], generando grupos que luego se asocian a especies o tipos de vegetación (p. ej., coníferas, arbustos, palmeras).
- Ventajas: rápida convergencia, simplicidad de implementación y control directo sobre el número de categorías.

3. Implementación Paso a Paso

3.1 Instalación de Dependencias

- Python: pip install numpy scikit-learn
- Blender: versión 4.2.3 LTS

4.2 Ejecución en Google Colab (Opcional)

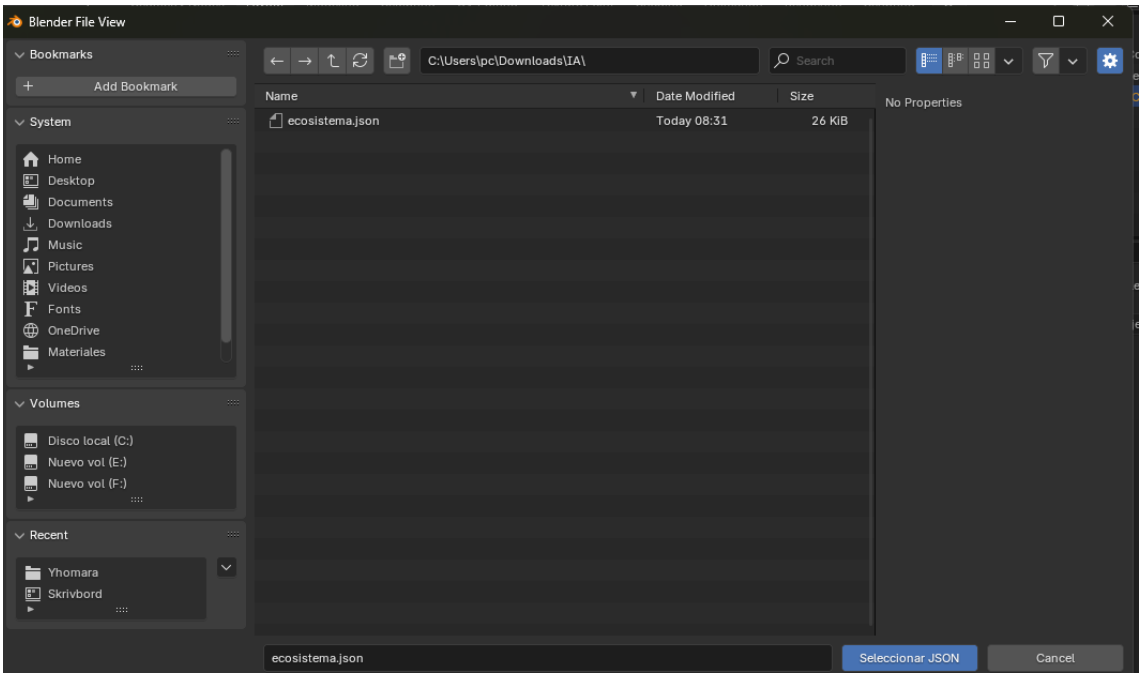
1. Crear notebook.
2. Código de generación de ecosistema.json.
3. Descargar archivo.

4.3 Configuración en Blender

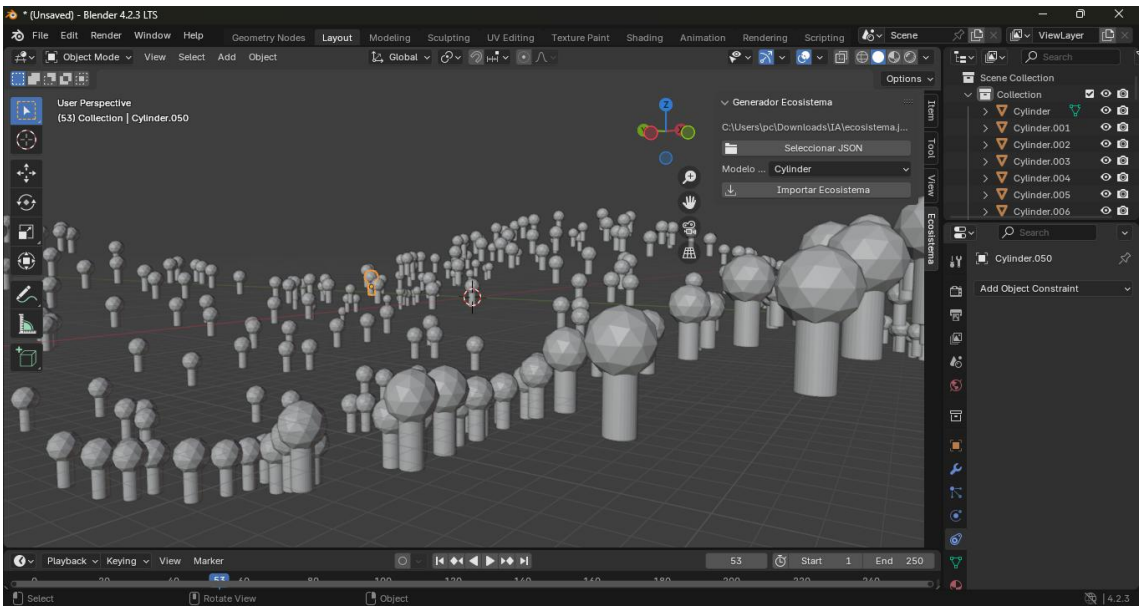
1. Pegar blender_addon.py en el Text Editor.
2. Run Script.
3. Abrir pestaña **Ecosistema**.
4. Seleccionar JSON y Modelo Base.
5. Pulsar **Importar Ecosistema**.

5. Demostraciones y Capturas

- Captura 1: File Browser seleccionando JSON.



- Captura 2: Instanciado de modelos en la escena.



6. Pruebas y Resultados

Parámetro	Valores probados	Observaciones
Iteraciones GNG	100, 300, 500	Mayor de iteraciones = más densidad de nodos.
Número de clusters	2, 3, 5	Afecta diversidad de especies.

Gráficas de tiempo de generación y densidad de nodos.

7. Conclusiones y Extensiones Futuras

7.1 Conclusiones

- La combinación de GNG y K-Means permite simular ecosistemas variados.
- Integración fluida con Blender a través de Python.

7.2 Extensiones

- Añadir simulación de viento con deformaciones dinámicas.
- Integrar fauna procedural con IA reforzada.
- Automatizar importación de múltiples JSONs en batch.