Minecraft Structure Generation Model Using Voxel Diffusion

Chenming Zhang
IIIS, Tsinghua University
zcm24@mails.tsinghua.edu.cn

Minghao Zhuang
IIIS, Tsinghua University
zhuangmh24@mails.tsinghua.edu.cn

Yixuan Fu
IIIS, Tsinghua University
fu-yx24@mails.tsinghua.edu.cn

Abstract

Procedural content generation (PCG) in sandbox games like *Minecraft* often struggles with contextual coherence when integrating predefined structures into varied landscapes. This can result in unnatural boundaries and a disjointed player experience. To address this, we propose a conditional Denoising Diffusion Probabilistic Model (DDPM) for 3D voxel-based inpainting. Our model operates on $32\times32\times32$ chunks and is capable of generating context-aware structures that seamlessly blend with their surroundings. The core of our contribution lies in an adaptive block representation strategy. For continuous and gently varying terrains such as grasslands, we employ **Block2vec embeddings** to maintain heightmap coherence. For biomes with complex, discrete objects like forests and villages, we pivot to an **Analog Bits** representation, which proved crucial for generating distinct structures like tree trunks that the embedding-based approach could not capture. Our model, a 3D U-Net, is further enhanced with a **self-conditioning** mechanism to improve generative consistency. Results demonstrate that our method can successfully perform biome-aware repair and generation, advancing beyond traditional rule-based PCG to a learnable and more versatile terrain completion framework.

1 Introduction

The world of *Minecraft* is a near-infinite expanse built from billions of cubic blocks, or "voxels." The game's ability to procedurally generate vast and diverse terrains is central to its appeal. However, a significant challenge in traditional PCG is the seamless integration of complex, predefined structures (like villages or dungeons) into the dynamically generated world. This process, often referred to as "stitching," can create jarring transitions and break the player's sense of immersion.

To overcome this limitation, we move from static, rule-based generation to a dynamic, learning-based approach. We frame the task of terrain generation as a conditional inpainting problem. We introduce a conditional diffusion model designed to intelligently fill a $32\times32\times32$ block volume based on the context provided by its immediate surroundings.

Our primary contribution is a novel framework that adapts its internal data representation based on the target biome. We discovered that a single representation is insufficient for the diverse generative needs of different environments.

- For continuous terrains like **grasslands**, we utilize vector embeddings learned via Block2vec to preserve the smoothness and coherence of the heightmap.

Preprint. Under review.

- For complex structures like **forests and villages**, we adopt an "Analog Bits" strategy, which represents blocks as vectors of binary bits. This proved essential for resolving fine-grained, discrete details such as tree trunks and building walls, a task where the embedding-based method failed.

By combining a 3D U-Net architecture with these adaptive representations and a self-conditioning mechanism, our model demonstrates a robust capacity for high-fidelity, context-aware structure generation.

2 Methodology

Our model is built upon a conditional Denoising Diffusion Probabilistic Model (DDPM) with a 3D U-Net backbone, designed specifically for 3D voxel data. The overall framework is enhanced by several key components, including adaptive data representations, a sophisticated noise scheduling strategy, an advanced attention mechanism, and a self-conditioning loop for improved coherence.

2.1 Adaptive Block Representations

Recognizing that different biomes have fundamentally different structural properties, we employ two distinct methods for representing voxel data.

2.1.1 Block2vec for Continuous Terrains

For biomes characterized by smooth, continuous surfaces like grasslands, we adopt a two-stage approach inspired by word embedding techniques.

1. **Embedding Learning:** Inspired by [1], we first train a Skip-Gram model on our *Minecraft* world data. This model learns to predict a block's neighbors from the block itself, resulting in a low-dimensional vector embedding for each unique block ID as shown in Figure 1.
2. **Diffusion in Embedding Space:** The diffusion model [4] is then trained to operate directly within this continuous embedding space. The 3D U-Net learns to denoise a noisy field of block vectors, effectively learning the statistical patterns of the terrain's structure and height.

2.1.2 Analog Bits for Discrete Structures

To generate complex structures like trees and buildings, we employ the **Analog Bits** [2] method:

1. **Integer-to-Bit Conversion:** Each integer block ID is converted into its binary representation of fixed length.
2. **Bit Scaling:** Binary values $\{0, 1\}$ are scaled to $\{-1.0, 1.0\}$ using the transformation $2.0 \times \text{bits} - 1.0$.
3. **Model Objective:** The model is trained to predict clean analog bits from their noisy versions. Outputs are converted back via thresholding (> 0).

This method forces the model to make a "hard" decision for each bit, enabling faithful reconstruction of discrete objects.

2.2 Architectural and Process Enhancements

- **3D U-Net:** The model uses 3D convolutions, residual blocks, and group normalization. For Analog Bits, we use a configurable U-Net with weight-standardized convolutions.
- **Spatiotemporal Attention:** Incorporated at low-resolution stages to fuse context from spatial and temporal axes for complex structures.
- **Self-Conditioning:** At each step t, the model reuses the previous prediction $\tilde{x}_0$ as input by concatenating it with x_t along the channel dimension. This provides temporal consistency. The self-conditioning mechanism is illustrated in Figure 2.

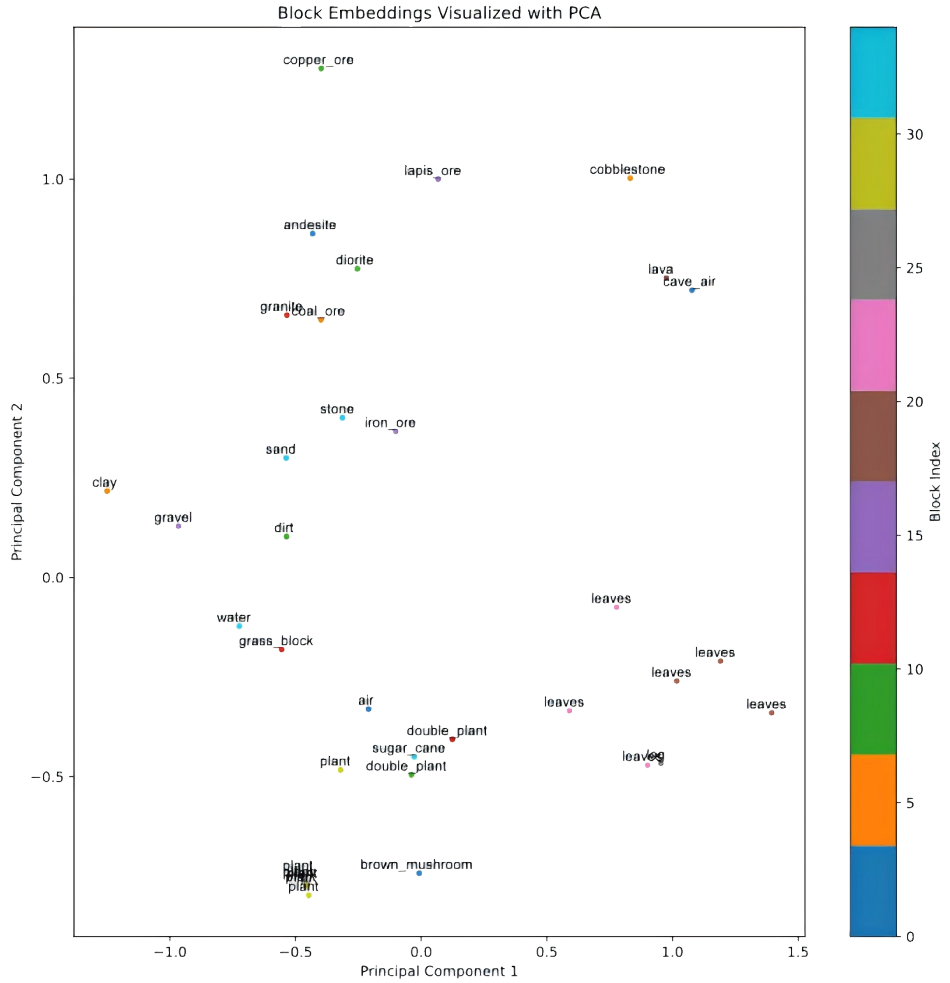
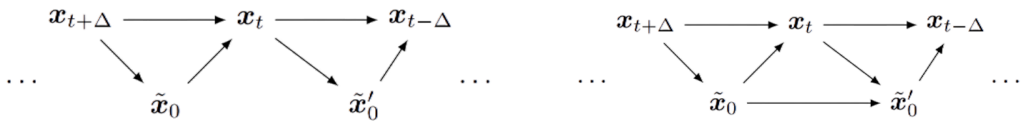


Figure 1: Block2vec embeddings visualized in 3D space. Each point represents a unique block type, with colors indicating different categories (e.g., grass, stone, water). The model learns to generate smooth transitions between these blocks, preserving the natural coherence of the terrain.



(a) Standard reverse diffusion steps.

(b) Self-Conditioning on the previous x_0 estimate.

Figure 2: Self-conditioning mechanism. The model reuses the previous prediction $\tilde{x}_0$ as input at each step t, enhancing temporal coherence and stability in the generated structures. The figure is from [2].

- **Cosine-Based Noise Scheduling:** We use a continuous-time diffusion process with cosine noise decay:

$$\alpha_t = \frac{f(0)}{f(t)}, \quad f(t) = \cos^2\left(\frac{1 + st/T + s}{2\pi}\right)$$

as proposed in [3]. The scheduling process and the comparison with linear noise scheduling are shown in Figure 3.

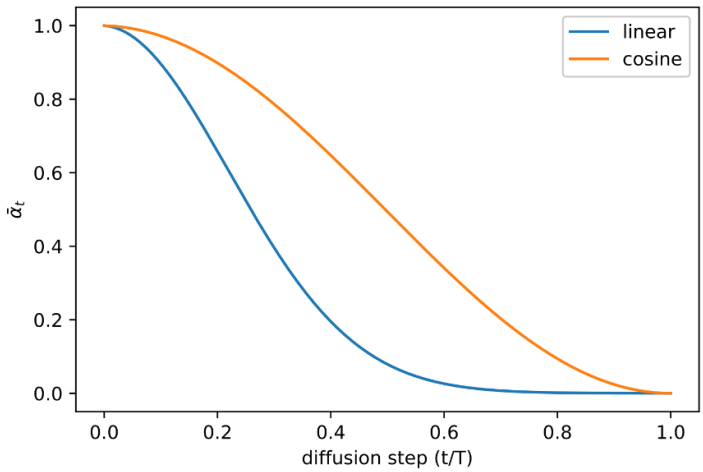
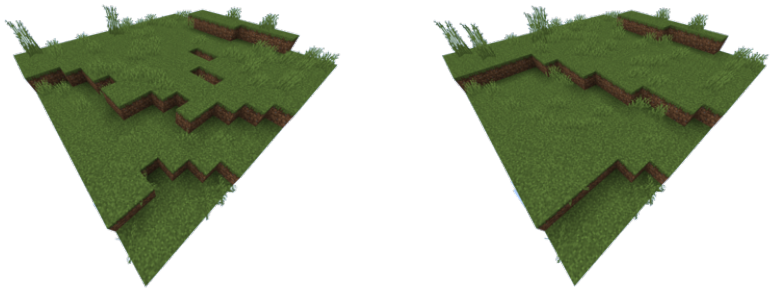


Figure 3: Cosine-based noise scheduling. The noise schedule is designed to provide a smooth transition between the initial and final states, allowing for better control over the denoising process. The figure is from [5].



(a) Raw grassland chunk. (b) Inpainted grassland chunk.

Figure 4: Grassland chunk before and after inpainting. The model successfully filled in missing terrain while preserving the natural heightmap coherence.

3 Experiments and Results

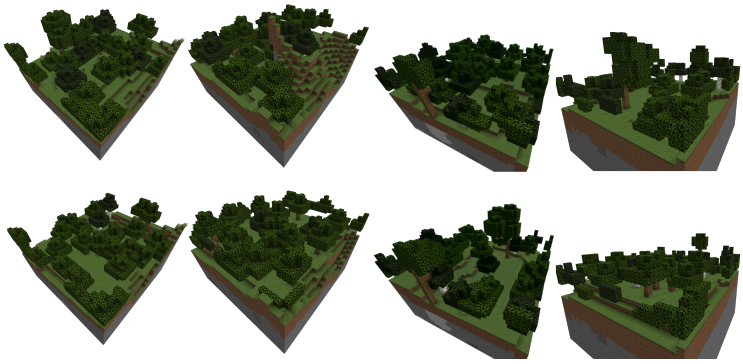
3.1 Data Acquisition

We generated several large-scale *Minecraft* worlds covering different biomes (grasslands, forests, villages). From these, we extracted 3D chunks of size $32\times32\times32$ (and $32\times24\times32$ for villages) for training.

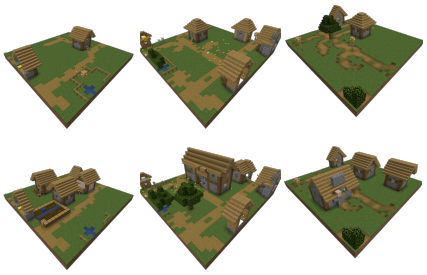
3.2 Experiment

Experiment 1: Grasslands with Block2vec

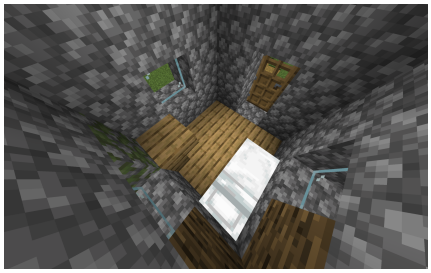
Using Block2vec, our model learned to generate smooth grassland terrain. Figure 4 confirmed that embedding space representation works well for continuous surfaces.



(a) Forest generation results.



(b) Village generation results.



(c) Village interior generation.

Figure 5: Forest and village generation results. The model successfully generated complex structures like trees, buildings, and interior decorations, demonstrating the effectiveness of the Analog Bits representation.

Experiment 2: The Limits of Block2vec in Forests

In forest biomes, the model could generate ground and leaves, but failed to produce tree trunks, revealing the limitation of Block2vec in discrete object generation.

Experiment 3: Success with Analog Bits in Forests and Villages

- **Forest Model:** With 5-bit representation and a 3-stage U-Net, our model generated complete trees.
- **Village Model:** A deeper 4-stage U-Net using 9-bit representation successfully generated buildings, paths, and interiors.

4 Conclusion

We proposed a conditional diffusion model for *Minecraft* structure generation using adaptive data representations. Block2vec excels in continuous terrains, while Analog Bits are superior for discrete structure synthesis. Combined with a self-conditioned 3D U-Net, our method enables biome-aware, high-fidelity procedural generation, setting a new direction for learnable PCG frameworks.

Future work could address current limitations. The model struggles to learn and reproduce very steep terrain features, potentially due to the inherent smoothing effect of diffusion models or the limitations of the current representations in capturing sharp discontinuities. Furthermore, while our adaptive representation strategy is effective, it currently requires separate models for distinct biome types. Integrating these into a single, universally adaptable model that can dynamically switch or blend representations based on local context remains a significant challenge and an important direction for future research.

References

[1] Maren Awiszus, Frederik Schubert, and Bodo Rosenhahn. World-gan: a generative model for minecraft worlds. In *Proceedings of the IEEE Conference on Games*, 2021.

[2] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[3] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[4] Simo Ryu. mindiffusion: A minimal denoising diffusion probabilistic models implementation. `https://github.com/cloneofsimo/minDiffusion`, 2022. GitHub repository.

[5] Lilian Weng. What are diffusion models? `https://github.com/cloneofsimo/minDiffusion`, Jul 2021.