UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS Honours Project Final Paper 2024

Title: Planetary Generation and authoring using diffusion models

Author: Christopher Langebrink

Project Abbreviation: EarthGen

Supervisor(s): Professor James Gain

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	15
System Development and Implementation	*0*	*20*	15
Results, Findings and Conclusions	*10*	*20*	20
Aim Formulation and Background Work	*10*	*15*	10
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	0
Total marks		**80**	**80**

Planetary Generation and Authoring using Diffusion Models

Christopher Langebrink
University of Cape Town
South Africa
lngchr014@myuct.ac.za

ABSTRACT

Generating digital planets is an essential aspect of virtual environment creation with widespread use in media, such as movies and games. Due to algorithmic and computational limitations, digital planet creation often lacks control and diversity. Creating such digital planetary terrains is challenging due to the need for effective user control, interactive feedback, and the model's ability to create perceptually realistic outputs that encompass a variety of land formations. To overcome this, we introduce a machine learning-based planetary sketch-to-terrain system that enhances realism by training models on real-world elevation, satellite imagery, landform classification, and vegetation index data. The terrain-authoring framework is based on an adapted diffusion model for conditional image synthesis. This system allows users to generate terrain ranging from continental scales to detailed landscapes, such as mountains and rivers. However, the challenge of merging terrain tiles without visible seams remains. While the system can generate realistic terrain tiles based on real-world elevation and satellite data, further refinement is needed to seamlessly integrate these tiles into a cohesive planetary surface.

CCS CONCEPTS

• Computing methodologies → Shape modelling; Machine learning;; • Applied computing;

KEYWORDS

Terrain generation, Machine learning, Diffusion Models

1 INTRODUCTION

Virtual planet creation plays an important role in a variety of widespread computer graphics applications, such as gaming, film, and virtual reality[22]. In applications such as film, artists will spend many hours authoring a planet to create a single fly-by shot. Planetary game design will usually use procedural generation algorithms or simulation-based methods to create a variety of planets. These methods, while able to provide an efficient way to generate planets through parameter changes, are solely based on algorithmic generation and lack effective user control, preventing the precise placement of landscape features.

Reproducing naturally occurring terrain presents significant challenges, given the immense variety of landforms such as hills, cliffs, valleys, and grasslands. The complexity of real-world terrain arises from geological processes that have unfolded over millennia, leading to deformation and erosion. For example, natural landforms have been shaped by the forces of weathering, gradual erosion, and tectonic plate movements [20]. Terrain is often represented as discrete heightfields, also known as Digital Elevation Models (DEMs), which are digital representations of Earth's surface topography. They are typically structured as a grid of elevation values or depicted as varying intensities in a greyscale image [3]. Once a virtual planet is created using a blend of heightfield terrains, it is necessary to enhance the surface details to portray elements like mountains, grasslands, vegetation, trees, rivers, and oceans, resulting in a fully realized natural earth [31].

An artist must have a dependable method for generating realistic terrain on demand while retaining the flexibility to incorporate their creative vision in placing landscape features. This process, known as terrain authoring, involves manipulating landforms and is a key aspect of terrain generation[31]. The main issue with large-scale terrain generation is the balance between effective user control, efficient generation, and the ability to create realistic earth-like planets that encapsulate a variety of landscape formations.

Recently, generative machine learning techniques have emerged as a competitive alternative for terrain generation, leveraging deep neural networks to understand the complex patterns and interrelationships between terrain features. Conditional Generative Adversarial Networks (CGANs) [36, 58, 60, 62] and Convolutional Neural Networks (CNNs) [2, 29] are trained on extensive real-world data and are conditioned on user sketches. However, these methods suffer from grid artefacts due to sparse inputs [31].

In this paper, we introduce a novel framework for planetary generation and authoring using a tile-based Denoising Diffusion Probabilistic Model (DDPM) backbone and an interactive blender interface. The system is developed by training and conditioning an image-to-image diffusion model with Classifier-Free Guidance (CFG) to transform low-detail user sketches into highly detailed terrain, represented as DEMs. The model is trained on global DEMs, satellite imagery, and landscape classification acquired from NASA's Blue Marble Next Generation (BMNG) archive [50] and The Global Land Cover by National Mapping Organizations (GLCNMO) [53] . The acquired datasets have a spatial resolution of 15 arc-seconds (approximately 500m per pixel at the equator) which allows for the model to be trained with higher resolution 86400x43200 image input. The model can generate terrain tiles and seamlessly blend them to create a cohesive digital planet.

This paper aims to tackle the complex research challenge of devising a framework for the efficient generation of realistic, planetary-scale terrain, enabling users to directly sketch and modify landscapes on a globe. We propose utilizing a diffusion model to generate realistic Earth-like planets, enhanced by an interactive interface that facilitates users to directly input and modify terrain features on the globe. In developing our approach, we have conducted an

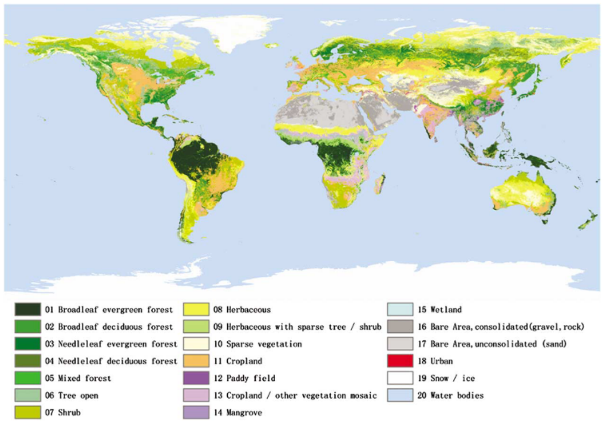
01 Broadleaf evergreen forest
02 Broadleaf deciduous forest
03 Needleleaf evergreen forest
04 Needleleaf deciduous forest
05 Mixed forest
06 Tree open
07 Shrub
08 Herbaceous
09 Herbaceous with sparse tree / shrub
10 Sparse vegetation
11 Cropland
12 Paddy field
13 Cropland / other vegetation mosaic
14 Mangrove
15 Wetland
16 Bare Area, consolidated(gravel, rock)
17 Bare Area, unconsolidated (sand)
18 Urban
19 Snow / ice
20 Water bodies

Figure 1: GLCNMO [50] Global land cover map.

extensive review of existing literature, covering key advancements in machine learning, texture synthesis, and their applications in terrain generation. This review highlights the current limitations in scaling these methods to a global context and identifies the potential of diffusion models in addressing these challenges. Specifically, we aim to explore the following research questions:

(1) Can a diffusion model facilitate the realistic generation of Earth-like planets?
(2) Does a globe-sketching interface enhance user control in the design of Earth-like planets?

Furthermore, we have considered the ethical implications of using real-world data in our project, ensuring that all data sources are appropriately licensed and that the results of our research respect intellectual property rights.

2 RELATED WORK

2.1 Conventional approaches to terrain generation

Traditionally, terrain generation techniques have been classed into three distinct categories: procedural generation, geomorphological simulation-based approaches, and example-based techniques as proposed by Galin *et al.* [20].

Procedural generation is a method for synthesizing terrain without relying on real-world data. It employs algorithms and parametric specifications to mimic geomorphological processes. The terrain can be shaped through techniques like subdivision schemes [15, 24], faulting [55], and noise based on fractional Brownian motion [33] (fBm) due to its simplicity [16, 20]. Large-scale procedural generation synthesizes terrain, focusing on fractal properties, and only providing indirect control. This was later extended to encompass entire planets, enabling the generation of large-scale watersheds [17]. Landform procedural generation focuses on specific features such as rivers or hills [20]. Although this method effectively captures terrain phenomenology, it lacks the diversity and complexity of real-world land formations.

Simulation techniques synthesize terrain by simulating the effect of erosion on landforms[20]. Key erosion processes used in these simulations include thermal [35], hydraulic with Eulerian [4] or Lagrangian [28] approaches, and tectonic simulation [7, 8, 34]. This process is carried out over multiple iterations, each corresponding to a real-world time step. Although these simulation techniques provide geological accuracy by leveraging underlying physical principles, they are computationally intensive and do not scale efficiently. Additionally, they often present challenges in terms of user control [20].

Example-based methods generate synthetic terrain based on real-world exemplar data and often rely on scanned heightfield terrains represented as DEMs [20]. Example-based synthesis can involve individual pixels [30], square patches [52, 61], or circular kernels [1]. Building on this foundation, later research focused on improving realism by refining techniques for patch blending and enhancing performance through GPU acceleration [19, 52], and addressing global drainage issues [47, 48]. Gain *et al.* [19] further improved authoring control by implementing painting and sketching tools. However, they often struggle with global consistency, as large terrains are typically composed of smaller, stitched-together features. Using example-based methods for large-scale generation proves to be computationally demanding with increased image resolutions and may suffer from limited user control. Additionally, as with deep learning-based methods, the synthesized terrain is inherently constrained to the input data [20].

However, we place our emphasis on machine learning-based approaches that take advantage of the extensive high-quality scanned real-world terrain data. These methods are particularly appealing as they offer the potential for higher levels of perceptual realism, user control, and performance in terrain generation.

2.2 Machine learning approaches to terrain generation

Machine-learning approaches to terrain generation involve training algorithms on large datasets of real-world terrain to learn the complex relationships and factors that comprise terrain features. Denoising Diffusion Probabilistic Models and Generative Adversarial Networks have demonstrated their effectiveness in generating realistic and complex terrains.

Conditional Generative Adversarial Networks are used in terrain generation by training a generator to create terrain based on specific conditions, such as user-provided sketches or style inputs. The generator's output is then evaluated by a discriminator, which helps refine the terrain until it closely matches real-world data while adhering to the given conditions.[20]. Isola *et al.* [27] showed the effectiveness of using cGANs for image-to-image translation tasks with the release of the pix2pix software. This was improved upon with high-resolution image synthesis and semantic manipulation [56]. Guérin *et al.* [21] proposed a CGAN architecture, drawing inspiration from the pix2pix software for terrain generation through learning the relationship between user sketches and DEMs. This method was further developed to allow for a range of authoring tools such as inpainting and upsampling with erosion detail [54]. However, these methods require highly detailed user sketches where sparse regions can lead to repetition and gridding artefacts.

Denoising Diffusion Probabilistic Models are another type of image-to-image generative model and have proved highly competitive in terrain generation. The DDPM architecture differs from GANs as they utilize a U-net backbone which has an encoder-decoder structure with skip connections that help in retaining spatial information [44]. A Residual Network (ResNet) is used within the U-net architecture to improve performance by introducing residual blocks to help in training very deep networks by addressing the vanishing gradient problem[23]. DDPMs work by gradually adding noise to input data in a forward process and then learning to reverse this process to generate new data from the pure Gaussian noise. They iteratively denoise an image through a series of steps, modelling the data distribution by predicting the noise added at each step. The model allows for new images to be generated based on the data distribution [49].

The inference time for DDPMs is slower than that of GANs because DDPMs require multiple forward and backward passes through the network for each generation, whereas GANs generate data in a single pass. However, despite this, DDPMs have recently demonstrated improvements over GANs in terrain synthesis tasks [11]. Further work has been developed on DDMs to improve image quality and sampling speed [25, 38, 46]. Lochner *et al.* [31] were among the first to develop a DDPM for terrain generation and authoring. Inspired by conditional image generation with high sample diversity [14, 32], they were able to develop a fast, accurate and effective authoring terrain generation framework. They performed a perceptual study which showed the model's ability to outperform CGAN-based terrain synthesis [31].

2.3 Large-scale terrain generation

Terrain generation models are inevitably prone to the computation complexities involved in producing high-quality large-scale accurate terrain. As these models are limited by the terrain size they can produce, adaptations have been made to divide the larger terrain into multiple tiles and combine them later. There are difficulties with this approach as seen with example-based terrain synthesis in creating a single coherent terrain by avoiding repetition and ensuring adjacent tiles are joined seamlessly. Frühstück *et al.* proposed a cGAN architecture that merges outputs from GANs trained on smaller resolutions to generate a large-scale, seamless texture map with minimal boundary artifacts, while also offering a user interface for enhanced artistic control [18]. Cruz *et al.* proposed a patch-based texture synthesis approach through a low-frequency guide, categorization, and map of the distribution of the categories[12]. The methodology utilizes Poison blending [39] and interpolation to generate large-scale terrain. However, limitations show that bending adjacent tiles can produce undesirable results and some results contain visual artefacts. Additionally, detailed user sketches are required to produce desirable results at the same large scale as the output [21]

2.4 Planetary generation

Planetary generation is the largest form of terrain synthesis. Individual tiles are generated and seamlessly merged to form a cohesive planet. Previously, planetary generation has been accomplished using procedural methods as they are computationally efficient for the large-scale task. Derzapf *el al.* [13] sufficiently proposed a procedural planet-generating algorithm that generates realistic river networks and landscapes in seconds and incorporates an interactive fly-through. Cortial *el al.* [9] proposed a procedural approach to generating synthetic tectonic planets by approximating the movement and collision of tectonic plates, without relying on resource-intensive simulations. Users can dynamically control plate movement to create various planetary features like continents and mountain ranges and enhance the model with detailed reliefs using procedural or real-world data. Cortial *el al.* [10] proposed a real-time method for generating highly detailed planets using a procedural hyper-amplification algorithm that subdivides user-defined control maps to produce high-resolution, hydrologically consistent models. Starting with large-scale features, they generate a river network and synthesize detailed landscapes, including tributaries, mountains, and valleys.

3 DATA PREPARATION

Diffusion models for terrain generation require tens of thousands to millions of samples to effectively encapsulate the variety of landforms and train adequately [25]. We obtained a combination of DEM, satellite, landscape classification and vegetation index data at a resolution of 86400x43200 to generate realistic and diverse terrain features. This data enables the model to integrate topographical elevation (DEM), visual surface characteristics (satellite imagery), and utilize detailed landform categorizations, vegetation indices, and low-detail topographical elevation for guidance (classification, vegetation, and low-detail DEM), providing a comprehensive understanding of the Earth's surface for high-fidelity terrain generation in the diffusion model framework. The datasets are gridded at a spatial resolution of 15 arc-seconds (500m approximate spacing at the equator) and use a Plate Carrée, also known as the equirectangular projection, based on equal latitude-longitude grid spacing [50]. The satellite and DEM datasets were acquired from NASA's Bluemarble archive - The Blue Marble Next Generation (BMNG) [50]. They performed temporal adjustments to the data using a discrete Fourier technique which removed cloud cover disturbances. The landscape classification dataset was acquired from Global Land Cover by National Mapping Organizations (GLCNMO) version 3 and classifies global land cover into 20 categories, including forest, shrub, snow, and sand [53]. The Normalized Difference Vegetation Index (NDVI) was acquired from NASA Earth Observations (NEO) which measures vegetation activity on the land surface mapping the values to a range of -0.1 to 0.9 [37].

We performed data augmentation on the dataset to ensure the diffusion model was adequately trained. This entailed removing the ocean from the classification, DEM, satellite imagery, and vegetation index. The data was then split into 2 grids, one 1350x675 grid of 64x64 pixels resulting in 318,000 tiles, and another 337x168 grid of 256x256 pixels resulting in 20,000 tiles. The grids were used to train

2 separate models. Each tile had a 90% landmass retention ensuring that a variety of landmass features were encapsulated. A caveat to using the BMNG dataset is that it shows certain inaccuracies in the spatial projection, therefore the top 4000 pixels of the image needed to be removed. This ensured a clean and accurate dataset without a distorted appearance in higher latitudes [50]. The dataset was free from noticeable errors, such as artefacts or noise, which could have been problematic for training purposes. Additionally, artificial human structures such as buildings and cropland, which would normally be problematic in diffusion model terrain generation, as mentioned by Lochner *et al.*[31], are insignificant in the scale of global terrain generation.

We trained two separate models, referred to as DDPM small and DDPM big. Due to the constraints of GPU memory and the underlying architecture of DDPM small, we limited our image size to 64x64 pixels and chose an area of approximately 32×32 km². In contrast, DDPM big benefited from superior GPU memory optimizations, therefore the model was able to produce images at a size of 256x256 pixels and chose an area of about 128×128 km². This necessitates a balance between the area covered and the resolution of the terrain and is large enough to capture significant geographical features like mountains and large rivers while still providing a diverse range of terrain types. This size is ideal for large-scale terrain generation, balancing the need for detail with the available computational resources. Future work will focus on incorporating additional data augmentations by applying a combination of rotations, and horizontal and vertical flips to enlarge the dataset, enabling the model to learn more effectively. However, the set of generated images was sufficient to demonstrate the effectiveness of using a diffusion model for terrain generation.

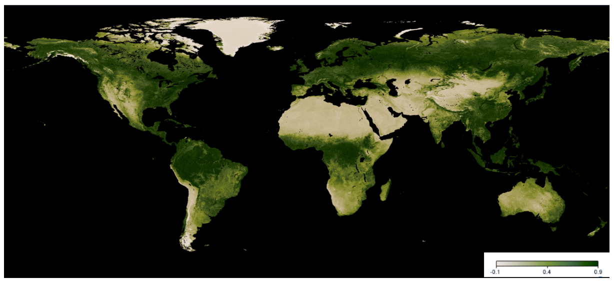

Figure 2: Normalized Difference Vegetation Index (NDVI) [37].

To allow for planet authoring, users need to be given specific control over the placement of landscape features. In contrast to procedural methods with parametric specification for generation, user sketches allow for versatile artistic control in specifying terrain characteristics[21]. User sketches are usually described as a network of river and ridge lines [31]. Our methodology is adapted from this and tested with two different types of sketching methodologies. One method involved using predefined classifications, where users could select a landscape type from 20 predefined categories, as illustrated in Figure 1. The second method utilized the vegetation index (NDVI) and low-detail elevation data, allowing users to draw terrain features. Users could select a vegetation index value from a colour table, as shown in Figure 2, and a grey-scale colour table to specify elevation preferences.

To generate the low-detail grey-scale elevation classification tiles, we performed a combination of downsizing and upsizing using nearest-neighbour interpolation on the high-quality input elevation data. Additionally, to ensure consistency in the model training, the grey-scale NDVI classification data had to be inverted to align with the elevation classification tiles. Each of these two methodologies independently guides the diffusion model in generating terrain that aligns with the user's input. The predefined classification method informs the model about the general landscape type by allowing users to select from 20 predefined categories, shaping the broader terrain features. On the other hand, the method using NDVI and elevation data provides a more detailed input, where users can specify vegetation and elevation characteristics to guide the model in generating specific terrain features. Although distinct, both approaches offer structured input that enables the diffusion model to produce terrain consistent with the user's selections.

4 METHODS

4.1 Diffusion Architecture

To achieve diffusion-based terrain synthesis, we adapted two image-to-image open-source conditional diffusion models. DDMP small was adapted from Dominic Rampas [41] and DDPM big was adapted from Matthew Rozanov [45].

DDPM small does not use lower-bound formulation for sampling and implements Classifier-Free-Guidance (CFG). The conditional DDPM is composed of UNet structure with downsampling, upsampling and self-attention layers as described in Section 2.2. The primary distinction from a traditional UNet is that the upsampling and downsampling blocks incorporate an additional timestep parameter during their forward pass. This is achieved by linearly integrating the timestep into the convolutional layers. The model was initially trained on the CIFAR-10 dataset [42] to produce conditional image-to-image generation based on the label classes. DDPM big integrates the same underlying UNet structure as DDPM small, however, the model replaces weight normalization with group normalization which improves stability and convergence more consistent normalization across group channels especially in large-scale training [57]. Additionally, the 256x256 model uses six feature map resolutions and diffusion time t is specified by adding the Transformer sinusoidal position embedding into each residual block [25, 45]. The model was initially trained on the smithsonian-butterflies-subset by HuggingFace for unconditional image generation [26].

We adapted the diffusion process for DDPM small and DDPM big for conditional terrain generation by concatenating the conditional classification images and the progressively noisy image at the current timestep in the forward pass. This allowed the model to learn the correlation between the conditional classifier images, input DEM and satellite image. The models were further augmented to change the number of input and output channels to allow for

Model	Image Size	Batch Size	Optimizer	Learning Rate	Training Steps	Diffusion Steps	Final Loss
DDPM Small	64x64	6	AdamW	1e-3	214000	1000	0.0094
DDPM Big LC	256x256	6	Adam	2e-5 to 1.56e-5	111000	1000	0.0136
DDPM Big VIE	256x256	6	Adam	2e-5 to 1.81e-5	71250	1000	0.0128

Table 1: Training and Hyperparameters for DDPM Models

a multi-channel diffusion process. DDPM small required five input channels (1: binary DEM, 1:Landscape Classification, and 3: Satellite Image) and four output channels were produced (1: binary DEM and 3: Satellite Image). We trained DDPM big twice based on the two types of sketching methodologies described in Section 3. We will refer to these as DDPM big LC (Landscape Classification) and DDPM big VIE (Vegetation index and Elevation). DDPM big LC required the same amount of input channels and produced the same amount of output channels as DDPM small as it was trained on the same conditional landscape classification dataset. DDPM big VIE required six input channels (1: binary DEM, 3: Satellite Image, 1: low-detail elevation classification, and 1: vegetation index) and similarly, four output channels were produced. The different methodologies to fully encapsulate the terrain with elevation and RGB values for each pixel. These models encapsulate the diverse terrain generation tasks by adapting the diffusion process to suit different inputs and classification schemes. DDPM small captures and reproduces terrain using basic landscape classifications, while DDPM big LC expands this capability by leveraging more detailed landscape with a higher input resolution of 256x256. DDPM big VIE, on the other hand, incorporates both vegetation index and elevation data, allowing for a more detailed and morphologically informed terrain generation.

Algorithm 1 Sampling

```
1:  $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$
2:  for $t = T, \ldots, 1$ do
3:      if $t > 1$ then
4:          $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$
5:      else
6:          $\mathbf{z} = 0$
7:      end if
8:      $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
9:  end for
10: return $\mathbf{x}_0$
```

Algorithm 1 above outlines a sampling process for the diffusion models, converting noise into coherent images[25]. It starts by sampling a noise vector $\mathbf{x}_T$ from a normal distribution. Then, it iteratively refines this noise vector backwards from timestep T to 1. At each step t, it updates $\mathbf{x}_{t-1}$ using a combination of the current state $\mathbf{x}_t$, a learned noise prediction $\epsilon_\theta(\mathbf{x}_t, t)$, and additional noise $\mathbf{z}$. The final result, $\mathbf{x}_0$, is the generated output.

Explanation of Variables:

- $\mathbf{x}_T$: Initial noise vector sampled from a normal distribution.
- T: The total number of timesteps.
- $\mathbf{z}$: Additional noise sampled at each timestep.
- α_t, σ_t: Parameters controlling the update rule.
- $\epsilon_\theta(\mathbf{x}_t, t)$: The predicted noise at timestep t.

4.2 Training and Hyperparameters

Due to limited time and computational resources, it is crucial to choose model parameters that balance quality, computational efficiency and interactive authoring. Lochner *et al.* found that during the tuning phase, larger models with increased base dimensions, channel multipliers, and more ResNet blocks can enhance performance, they significantly prolong training and sampling times and may even cause out-of-memory issues[31]. Table 1 illustrates the chosen hyperparameters for the three models.

We found through early iterations of testing that increasing the batch size beyond six caused the model's training time to increase significantly and came at the cost of increased GPU memory consumption. Additionally, decreasing the batch size below six negatively impacted the model's ability to generalize and capture finer details in the terrain, a problem observed across all three models.DDPM small (27 million parameters) was restricted to a maximum resolution of 64x64 pixels, as the U-Net architecture did not support higher resolutions without memory overload. The model was trained for 214,000 steps and achieved a final loss of 0.0094. However, due to its lower resolution and limited capacity, the final loss is not a direct reflection of its ability to generate realistic terrain. In contrast, the DDPM big models integrated a superior U-Net architecture with GPU memory optimizations, allowing them to handle higher resolution images (256x256 pixels) and larger parameter counts (58 million parameters). The DDPM big LC model was trained for 111,000 steps and achieved a final loss of 0.0136, while the DDPM big VIE model required 71,250 steps and achieved a final loss of 0.0128.

A cosine annealing learning rate scheduler was chosen for both DDPM big models as it allowed for progressive refinement of weights, ensuring model stability and preventing overfitting during the later stages of training. The scheduler gradually reduced the learning rate, enabling the models to settle into local minima. This method proved essential in ensuring high-quality terrain generation with fewer artefacts. Training time was also a consideration. While the DDPM small model trained faster due to its smaller parameter count and lower resolution, it could not generate high-quality terrain compared to the DDPM big models. The DDPM big models required more memory and longer training times, but memory optimizations helped mitigate this, making them better suited for detailed terrain generation. While the loss values for the DDPM big models were higher than that of DDPM small, loss is calculated based on the Mean Square Error (MSE) between predicted and generated noise, which does not fully capture the model's performance in generating terrain. Therefore, other qualitative factors, such as the realism of the generated terrain and perceptual similarity metrics, were considered to assess the model's true capability. These aspects are discussed in greater detail in Section 6.

Training was conducted on Uct's HPC equipped with NVIDIA A100 GPU (40GB), while testing and evaluation were performed on a desktop equipped with an AMD Ryzen 7 5700 (3.70 GHz) and NVIDIA RTX 4070 SUPER (12GB)

4.3 Interface

The planet authoring interface and terrain rendering were implemented in Blender[1]. The python API allowed for integration with the diffusion model framework. Initial testing of an interface using React showed unpromising results with difficulties in vertex calculations, tile dimensions, and projecting elevation values onto a sphere. The issue of generating meshes of a sphere is a well-known problem in 3D computer graphics, moreover generating uniform subdivisions of a sphere is a challenging task [43].
Generating uniform subdivisions of a sphere is necessary for diffusion model planetary generation as a diffusion model's computational complexity limits the size of the image it can generate. Borg implemented an equirectangular projection from a single plane of merged generated terrain tiles to form a cohesive planet[5]. However, this methodology is prone to producing seam-like artefacts towards the poles. The method involves dividing the sphere into segments along its meridians (lines running from pole to pole) and parallels (lines parallel to the equator). This creates a mesh structure, where triangles form a fan shape around the poles, and quadrilateral shapes cover the rest of the sphere [6].

Through evaluation of different types of mesh spheres (UV sphere, Icosphere, Quad sphere, Goldberg Polyhedron) [51], we decided to implement a Quad sphere as this type of mesh sphere suits the tiled application of the diffusion model. This methodology iteratively subdivides a hexahedron producing 6 potential points of distortion, unlike the UV sphere with 2 points of distortion. This is illustrated in Figure 3. Therefore each subdivided tile of the cubic sphere is more uniform than the UV sphere, which produces less overall distortion when mapping generated terrain onto the sphere.

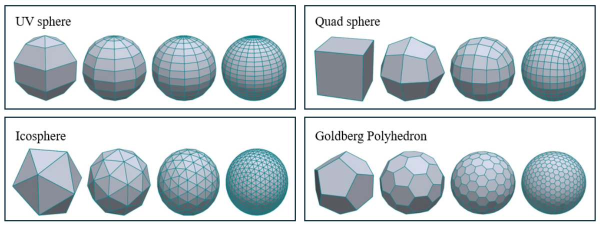

Figure 3: ***A comparison of the different types of mesh spheres where increasing the number of subdivisions is shown from left to right.***

[1]https://www.blender.org/

Representing changes in elevation for a virtual Earth poses significant challenges, especially when trying to maintain both accuracy and realism. A true-to-scale representation of Earth's topography would appear almost entirely smooth from a distance, given that the height difference between the lowest point (ocean floor) and the highest point (mountain peaks) is negligible compared to the Earth's overall diameter. Visualizing this with absolute precision and accuracy would result in Earth resembling a smooth sphere. In our theoretical analysis, we aimed to preserve the original 86,400x43,200 pixel resolution of the global input data. For the DDPM small model, where each tile measures 64x64 pixels (representing an area of approximately 32×32 km^2), this would require subdividing each of the six faces of the cubic sphere into a grid of at least 228x228 tiles, resulting in approximately 310,000 tiles across the sphere. For the DDPM big models, where each tile measures 256x256 pixels (representing an area of 128×128 km^2), this would require subdividing each face into a grid of 32x32 tiles, yielding around 6,100 tiles across the entire sphere.

This level of subdivision would ensure the accuracy and detail of the original input data across the spherical projection. However, given the memory constraints of our available hardware, where each tile consumes significant GPU memory during processing, the original theoretical subdivision would have resulted in prohibitive computational costs. Instead, we implemented 8x8 subdivisions per face, resulting in a total of 384 tiles across the sphere. Despite reducing the number of tiles, the system was still able to generate visually coherent terrain tiles that preserved important topographical features, though future optimizations may allow for increased tile resolution. This subdivision method was evaluated in terms of performance, scalability, and realism, as discussed in Section 6

The interface we developed enables users to interactively design terrain features by drawing on a sphere displayed on the left side of the interface. Users can select from 20 different landscape feature types, each corresponding to specific terrain characteristics[53]. These user-defined features are then used to generate the corresponding terrain on a sphere displayed on the right side of the interface. The terrain generation occurs on a per-tile basis, leveraging the DDPM big LC diffusion model framework integrated with Blender's [2]Python API. Each tile is produced by first generating a Digital Elevation Model (DEM) and a satellite image through the diffusion model. The resulting DEM is used to apply a displacement to the tile, which is subdivided into a 100x100 grid. Finally, the generated satellite image is applied as a texture over the displacement map, ensuring that the visual representation accurately reflects the intended landscape features.

5 EVALUATION

The evaluation of the generated terrain in this paper focuses on four key criteria: scalability, realism, performance, and user control. We provide a direct comparison of the three DDPM models (DDPM small, DDPM big LC, and DDPM big VIE) on several metrics and provide an evaluation of the system as a whole.

[2]https://www.blender.org/

5.1 Scalability

Scalability refers to the model's ability to synthesize terrain in a range of spatial dimensions. It is important to assess scalability as it answers the question of how large of an area the model can generate and in what detail [20]. However, the focus of this paper is large-scale terrain generation so the evaluation will assess the model's ability to generate landscape features such as mountain ranges, continents, rivers, and plateaus that are consistent with real-world scales. By analyzing the generated features in comparison to actual geographical formations, the criteria aim to show a reflection of the model's effectiveness in replicating the scale, proportions, and spatial relationships found in natural landscapes.

Precision refers to the real-world distance measured at each pixel of the generated terrain. This model's ability to accurately reproduce the fine details and specific features of the terrain, such as elevation changes, textures, and boundaries, within each individual pixel of the generated output. Each pixel in the generated Digital Elevation Model (DEM) and corresponding satellite image represents a specific area of the landscape, with the precision of the model being critical for ensuring that these representations closely match real-world terrain. All DDPM models show a precision of 500m per pixel as this is consistent with the precision input data.

Extent refers to the real-world distance measured by the length of a tile of the generated terrain. It is important to consider that computation complexity grows exponentially as the extent of the generated terrain increases. As such our DDPM small model produces 64x64 tiles corresponding to an extent of 32km while the DDPM big models produce 256x256 corresponding to an extent of 128km. Additionally, the extent needs to be measured for planetary generation to ensure the generated earth-like planet is consistent with Earth. DDPM small was unable to produce consistent terrain so we were not able to integrate it into the planetary generation. The total extent of the DDPM big models was 2048km imposed GPU memory constraints and terrain generation time.

5.2 Realism

Realism in terrain generation pertains to the model's capability to generate data that is convincingly authentic. This involves producing high-quality images and digital elevation models that closely resemble the original data. Specifically, for terrain generation, realism encompasses geomorphological accuracy, which includes the diversity of landforms, as well as both local and global consistency, and the overall perceptual believably of the terrain. While there are quantitative metrics available to assess terrain realism, such as PTRM[40] and drainage analysis, these were considered beyond the scope of this experiment and may not be suitable for planetary-scale applications. Machine learning models are generally split into training and test data to evaluate the performance of the model against ground truth data. However, due to limited input data and time constraints, the entire dataset was used for training purposes.

During our evaluation of realism, we used the Learned Perceptual Image Patch similarity (LPIPs) [59] as a metric to evaluate the perceptual similarity between two images, it aims to measure similarity in a way that aligns more closely with human perception

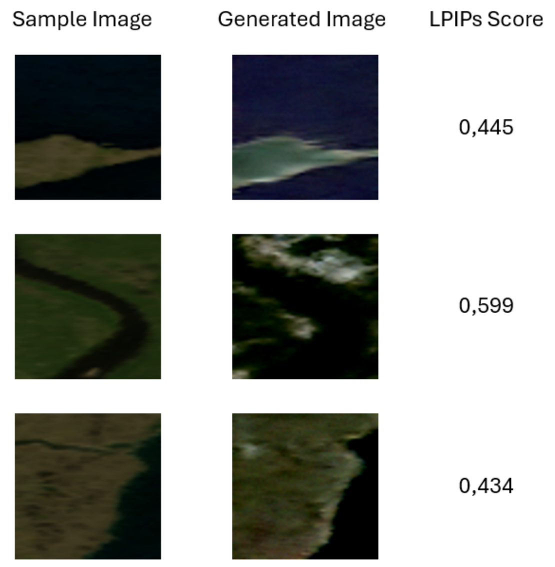

Figure 4: LPIPs metric comparing generated terrain to sample terrain for DDPM small

compared to traditional metrics like Mean Squared Error (MSE) or Peak Signal-to-Noise Ratio (PSNR). LPIPS evaluates the perceptual similarity between two images by leveraging pre-trained deep neural networks, like VGG or AlexNet, to extract and compare high-level features rather than raw pixel values. These features, capturing complex textures and structures, are compared across different network layers, with the differences weighted and summed to produce a final similarity score. A lower LPIPS score indicates that the images are more perceptually similar, aligning closely with human visual perception[59].

Borg *et al.* [5] performed a realism evaluation, comparing PaleoDEMs to the generated planetary terrain, however, this method was unfeasible for our approach as our interface framework requires merging individually generated tiles onto the sphere, instead of generating a large single plane of merged tiles and wrapping that onto the sphere. As such we will compare the generated satellite images and grey scale elevation data encompassing the terrain of the three models using the LPIPs metric on a single tile basis as illustrated in in Figure 4 and Figure 7. Additionally, a qualitative analysis of the realism of a generated planet with an example user sketch will be discussed as illustrated in 8. All generation was performed on 1000 time steps as anything less than this caused the models to produce noisy terrain

5.3 Performance

Evaluating performance for terrain generation is essential as it allows for effective user authoring. Inference speed is the speed at which the diffusion model is capable of generating an output from random noise. It is essential to ensure fast inference speed as this is the difference between the time that the user can draw on the interface and the time it takes the diffusion model to generate the corresponding terrain. Galin *et al.* proposed an evaluation method to categorize inference speed into 4 distinct groups: real-time (< 1/3 seconds or 3Hz), interactive (<3 seconds), seconds (<60 seconds), and minutes (>60 seconds) [20]. We compared our model's ability to generate terrain under these categories with different time steps.

5.4 Control

Control is essential in planet authoring using diffusion models because it enables precise manipulation of the generated landscapes, allowing creators to guide the outcome toward desired characteristics. In planetary generation, especially on a large scale, the ability to control features such as terrain elevation, climate zones, and geographical formations is crucial for achieving both realism and specific aesthetic goals. Diffusion models, while powerful in generating complex patterns, benefit from control mechanisms that allow for targeted modifications, ensuring that the generated planet aligns with the user's vision. We provide a qualitative evaluation of the ease of use of the interface as well as specific control over the placement and shaping of landscape features.

6 DISCUSSION

6.1 Scalability

Our approach enables a level of precision of 500m per pixel, resulting in an accurate reproduction of the fine details and specific features of the terrain as illustrated by the generated DEMs in Figure 7. However, our approach is limited in extent, producing earth-like planets at an extent of 2048km. While procedural and simulation-based techniques can achieve larger extents with even finer precision, they lack the degree of user control provided by our system. The lack of a tile-merging mechanism in the current framework leads to seam artefacts between adjacent tiles, restricting its ability to scale seamlessly. Incorporating methods like linear interpolation or graph cuts could address this issue, allowing for smoother transitions between tiles. With higher-resolution data, our system could be extended to match or exceed the precision of procedural methods. Further improvements, such as increasing the number of tiles, will be necessary to generate full Earth-sized planets without losing detail or control.

6.2 Realism

Our system demonstrates the ability to generate terrain that aligns with real-world data, though improvements are still needed. Using the LPIPS metric, we assessed perceptual similarity between the generated terrain and sample images. While the model produces terrain with a consistent underlying structure, the results show room for improvement in matching finer details, particularly when generating at lower resolutions (64x64 pixels). Higher-resolution input data (256x256 pixels) should help address this limitation by allowing the model to capture more detailed and accurate features. Although the LPIPS scores were not as close to zero as expected, indicating lower similarity, the generated terrain still shows potential for further refinement with additional training and higher-quality input data. In addition, some generated samples displayed noise and lacked consistency, particularly when fewer time steps were used. This indicates that increasing the number of time steps and training with more robust datasets may improve the overall realism and reduce variability in the outputs.

6.3 Performance

During our performance evaluation, we observed that the model was unable to generate terrain resembling the input image when the number of timesteps was reduced below 1000. At these lower timesteps, the output was dominated by noise, although the inference speed was notably faster.

To evaluate the performance of our models, we measured inference time across varying time steps (1000, 500, 200, 100, 50, 20, 10). As shown in Graph 5 below, all models exhibit a significant reduction in inference time as the number of time steps decreases. At 1000 time steps, the models successfully generated coherent terrain, with inference times of just over a minute. This places them in the "minutes" category, as defined by Galin *et al.* [**?**]Galin2019}. For example, DDPM Big VIE took approximately 41.8 seconds, while DDPM Small completed in 24.8 seconds due to its lower complexity. As the number of time steps was reduced to 100, the inference time dropped significantly to around 6 seconds for all models. Further reducing the time steps to 50 resulted in inference times of about 3 seconds, placing the models in the "interactive" category. At 10 time steps, the inference times for all models were under 1 second, with DDPM Small completing in just 0.21 seconds.

These results highlight a trade-off between inference speed and model complexity. The DDPM Small model consistently outperforms the larger models in terms of speed, making it more suitable for rapid generation. However, this comes at the cost of detail and accuracy, which DDPM Big VIE and DDPM Big LC provide at the expense of longer inference times.

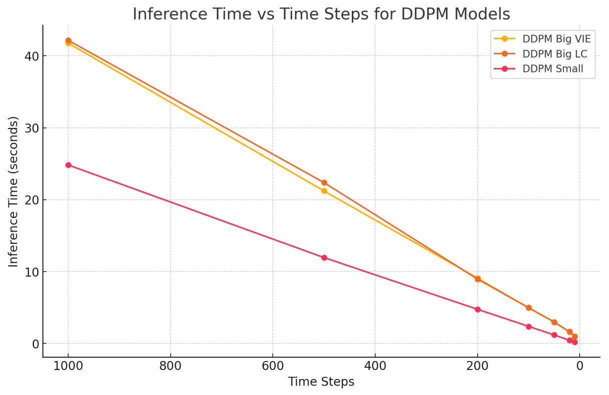

Figure 5: Graph showing the inference time for each of the three DDPM models (Big VIE, Big LC, and Small) across different time steps. The time steps are plotted on the x-axis (with larger time steps on the left), and the inference time in seconds is on the y-axis. Each model's performance is represented by a distinct line

6.4 Control

Control is a crucial aspect of terrain generation, particularly in a system that seeks to balance user input with realistic outputs. Our globe sketching interface (as shown in Figure 6) allows users to draw directly on a spherical canvas, offering intuitive control over the placement and shaping of landscape features. Users can specify conditional classification types, such as broadleaf evergreen forest, water bodies, sand, or shrub, and the diffusion model generates corresponding detailed terrain based on these inputs.

Figure 8 illustrates the comparison between a user's sketch (left) and the generated planet (right), demonstrating the system's ability to translate rough inputs into terrain. While the interface successfully provides broad control over major landforms, some limitations remain in achieving precise, fine-grained control, particularly for intricate features like small rivers or detailed elevation changes. Additionally, due to the model's limited ability to produce consistent terrain, individual tiles do not correspond well to each other.

Further refinements to the interface, such as more granular input tools or the ability to control specific parameters like elevation range and vegetation type, could enhance user control and improve the system's flexibility. Nonetheless, the current implementation already offers a balance between artistic input and automated terrain generation, allowing users to create planets from simple sketches.

6.5 limitations

- **Tile Seam and Blending**: Even though the DDPM Big models (256x256) offer better resolution and morphological detail, they still suffer from visible seam artefacts when multiple tiles are combined. The absence of advanced tile-merging mechanisms results in noticeable discontinuities, which can disrupt the visual coherence of planetary-scale terrains. This limitation is particularly important for applications where seamless large-scale terrain is crucial.
- **Inference Speed and Computational Overhead**: While the 256x256 models provide more detailed outputs, they also require significant computational resources. The larger DDPM models take longer to generate terrains due to the increased number of time steps required to avoid noisy results. Although improvements were made to manage memory through optimizations, the inference time remains in the "minutes" category when using 1000 time steps. For real-time applications, such as interactive planetary generation, this limits the system's usability.
- **Resolution Versus Extent**: Although the resolution of 256x256 tiles (representing 128x128 km^2) improves the detail of individual terrains, the total extent that can be generated (2048 km across multiple tiles) is still a limitation. The models struggle to cover vast areas with consistent terrain without running into GPU memory constraints, requiring more efficient techniques for scaling to planet-sized terrains.
- **Control Precision and Usability**: The integration of vegetation index (NDVI) and detailed landscape classification in the DDPM Big models enhances user control, but fine-grained control remains limited. Users may find it difficult to manipulate smaller-scale features like detailed rivers or subtle elevation gradients. Additionally, the interface's ability to reflect precise user inputs in the generated terrains, particularly at smaller scales, could be improved with more granular input methods or better interpolation techniques.
- **Evaluation Metrics**: While the LPIPS metric was used to evaluate perceptual similarity, the realism of generated terrains needs further qualitative and quantitative evaluation. The current testing environment did not include a comprehensive comparison with other terrain generation techniques, such as GAN-based or procedural methods. As a result, it is difficult to fully assess the competitive advantages of diffusion models without more extensive testing.
- **Generalization Limits Due to Dataset**: The DDPM Big models were trained on specific datasets (BMNG, GLCNMO, and NDVI data) with limited geographical diversity, potentially restricting the models' ability to generalize to novel or unseen terrain types. Future work could focus on expanding the dataset to include more diverse and complex terrains or augmenting training data to simulate different environmental conditions.

7 CONCLUSIONS

This research presents a novel approach to planetary terrain generation using Denoising Diffusion Probabilistic Models (DDPMs), integrated with a user-friendly interface in Blender. By leveraging DEM, satellite imagery, and landscape classifications, the system generates large-scale terrains from low-detail user sketches. The use of two high-resolution (256x256) models, DDPM Big LC and DDPM Big VIE, allows for the placement of landscape features. However, challenges such as tile-merging artefacts, high computational demands, and limited control over finer terrain details remain. Despite these limitations, diffusion models prove effective for creating diverse, realistic terrains compared to traditional methods. Future work will focus on improving tile merging, enhancing scalability, and increasing user control. With further refinement, this system has the potential to be a powerful tool for digital planet creation in fields like gaming, film, and virtual reality.

8 ACKNOWLEDGEMENTS

We express our gratitude to our supervisor, Professor James Gain, and our second reader, Josiah Chavula, for their invaluable feedback and guidance throughout this project. We also acknowledge the University of Cape Town's ICTS High Performance Computing team for providing the computational resources necessary to complete this work, accessible at hpc.uct.ac.za.

REFERENCES

[1] Oscar Argudo, Carlos Andujar, Antonio Chica, Eric Guérin, Julie Digne, Adrien Peytavie, and Eric Galin. 2017. Coherent multi-layer landscape synthesis. *The Visual Computer* 33 (2017), 1005–1015.
[2] Oscar Argudo, Antoni Chica, and Carlos Andujar. 2018. Terrain super-resolution through aerial imagery and fully convolutional networks. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 101–110.
[3] A Balasubramanian. 2017. Digital elevation model (DEM) in GIS. *University of Mysore* (2017).
[4] Bedřich Beneš, Václav Těšínskỳ, Jan Hornyš, and Sanjiv K Bhatia. 2006. Hydraulic erosion. *Computer Animation and Virtual Worlds* 17, 2 (2006), 99–108.
[5] Oliver Borg. 2023. Planet Authoring with Generative AI. University of Cape Town. https://projects.cs.uct.ac.za/honsproj/cgi-bin/view/2023/borg_brannhitge.zip/assets/Papers/PlanetAI.pdf

[6] Benjamin Coors, Alexandru Paul Condurache, and Andreas Geiger. 2018. Spherenet: Learning spherical representations for detection and classification in omnidirectional images. In *Proceedings of the European conference on computer vision (ECCV)*. 518–533.

[7] Guillaume Cordonnier, Jean Braun, Marie-Paule Cani, Bedrich Benes, Eric Galin, Adrien Peytavie, and Éric Guérin. 2016. Large scale terrain generation from tectonic uplift and fluvial erosion. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 165–175.

[8] Guillaume Cordonnier, Marie-Paule Cani, Bedrich Benes, Jean Braun, and Eric Galin. 2017. Sculpting mountains: Interactive terrain modeling based on subsurface geology. *IEEE transactions on visualization and computer graphics* 24, 5 (2017), 1756–1769.

[9] Yann Cortial, Adrien Peytavie, Eric Galin, and Eric Guérin. 2019. Procedural tectonic planets. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 1–11.

[10] Yann Cortial, Adrien Peytavie, Éric Galin, and Éric Guérin. 2020. Real-time hyper-amplification of planets. *The Visual Computer* 36, 10 (2020), 2273–2284.

[11] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. 2023. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 9 (2023), 10850–10869.

[12] Leandro Cruz, Luiz Velho, Eric Galin, Adrien Peytavie, and Eric Guérin. 2015. Patch-based terrain synthesis. In *International Conference on Computer Graphics Theory and Applications*. 6–pages.

[13] Evgenij Derzapf, Björn Ganster, Michael Guthe, and Reinhard Klein. 2011. River networks for instant procedural planets. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 2031–2040.

[14] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems* 34 (2021), 8780–8794.

[15] AR Dixon, GH Kirby, and Derek PM Wills. 1994. A data structure for artificial terrain generation. In *Computer Graphics Forum*, Vol. 13. Wiley Online Library, 37–48.

[16] David S Ebert, F Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steve Worley. 2002. *Texturing and modeling: a procedural approach.* Elsevier.

[17] Alain Fournier, Don Fussell, and Loren Carpenter. 1982. Computer rendering of stochastic models. *Commun. ACM* 25, 6 (1982), 371–384.

[18] Anna Frühstück, Ibraheem Alhashim, and Peter Wonka. 2019. Tilegan: synthesis of large-scale non-homogeneous textures. *ACM Transactions on graphics (TOG)* 38, 4 (2019), 1–11.

[19] James Gain, Bruce Merry, and Patrick Marais. 2015. Parallel, realistic and controllable terrain synthesis. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 105–116.

[20] Eric Galin, Eric Guérin, Adrien Peytavie, Guillaume Cordonnier, Marie-Paule Cani, Bedrich Benes, and James Gain. 2019. A review of digital terrain modeling. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 553–577.

[21] Éric Guérin, Julie Digne, Eric Galin, Adrien Peytavie, Christian Wolf, Bedrich Benes, and Benoît Martinez. 2017. Interactive example-based terrain authoring with conditional generative adversarial networks. *ACM Trans. Graph.* 36, 6 (2017), 228–1.

[22] Eric Guérin, Adrien Peytavie, Simon Masnou, Julie Digne, Basile Sauvage, James Gain, and Eric Galin. 2022. Gradient terrain authoring. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 85–95.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 770–778.

[24] Houssam Hnaidi, Eric Guérin, Samir Akkouche, Adrien Peytavie, and Eric Galin. 2010. Feature based terrain generation using diffusion equation. In *Computer Graphics Forum*, Vol. 29. Wiley Online Library, 2179–2186.

[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

[26] Hugging Face Huggan and Smithsonian Institution. 2023. Smithsonian Butterflies Subset Dataset. https://huggingface.co/datasets/huggan/smithsonian_butterflies_subset.

[27] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 1125–1134.

[28] Peter Krištof, Bedrich Beneš, Jaroslav Křivánek, and Ondrej Št'ava. 2009. Hydraulic erosion using smoothed particle hydrodynamics. In *Computer graphics forum*, Vol. 28. Wiley Online Library, 219–228.

[29] Ashish A Kubade, Avinash Sharma, and Krishnan Sundara Rajan. 2020. Feedback neural network based super-resolution of dem for generating high fidelity features. In *IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium.* IEEE, 1671–1674.

[30] Qicheng Li, Guoping Wang, Feng Zhou, Xiaohui Tang, and Kun Yang. 2006. Example-based realistic terrain generation. In *International Conference on Artificial Reality and Telexistence.* Springer, 811–818.

[31] Joshua Lochner, James Gain, Simon Perche, Adrien Peytavie, Eric Galin, and Eric Guérin. 2023. Interactive Authoring of Terrain using Diffusion Models. In *Computer Graphics Forum*, Vol. 42. Wiley Online Library, e14941.

[32] Calvin Luo. 2022. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970* (2022).

[33] Benoit B Mandelbrot and John W Van Ness. 1968. Fractional Brownian motions, fractional noises and applications. *SIAM review* 10, 4 (1968), 422–437.

[34] Elie Michel, Arnaud Emilien, and Marie-Paule Cani. 2015. Generation of folded terrains from simple vector maps. In *Eurographics 2015 short paper proceedings.* The Eurographics Association, 4.

[35] F Kenton Musgrave, Craig E Kolb, and Robert S Mace. 1989. The synthesis and rendering of eroded fractal terrains. *ACM Siggraph Computer Graphics* 23, 3 (1989), 41–50.

[36] Shanthika Naik, Aryamaan Jain, Avinash Sharma, and Krishnan Sundara Rajan. 2022. Deep generative framework for interactive 3d terrain authoring and manipulation. In *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium.* IEEE, 6410–6413.

[37] NASA Earth Observing System Data and Information System (EOSDIS). 2024. MODIS Vegetation Index (NDVI). https://neo.gsfc.nasa.gov/view.php?datasetId=MOD_NDVI_M Accessed: 2024-09-07.

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

[39] Patrick Pérez, Michel Gangnet, and Andrew Blake. 2023. Poisson image editing. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2.* 577–582.

[40] Suren Deepak Rajasekaran, Hao Kang, Martin Čadík, Eric Galin, Eric Guérin, Adrien Peytavie, Pavel Slavík, and Bedrich Benes. 2022. PTRM: Perceived terrain realism metric. *ACM Transactions on Applied Perceptions (TAP)* 19, 2 (2022), 1–22.

[41] Dominic Rampas. 2022. Diffusion-Models-pytorch. https://github.com/dome272/Diffusion-Models-pytorch.

[42] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2018. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451* (2018).

[43] Anooshe Rezaee Javan, Ting-Uei Lee, and Yi Min Xie. 2022. Dividing a sphere into equal-area and/or equilateral spherical polygons. *Journal of Computational Design and Engineering* 9, 2 (2022), 826–836.

[44] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18.* Springer, 234–241.

[45] Matthew Rozanov. 2023. diffusion-ddpm. https://github.com/mattroz/diffusion-ddpm/tree/main?tab=readme-ov-file.

[46] Tim Salimans and Jonathan Ho. 2022. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512* (2022).

[47] Joshua J Scott and Neil A Dodgson. 2021. Example-based terrain synthesis with pit removal. *Computers & Graphics* 99 (2021), 43–53.

[48] Joshua J Scott and Neil A Dodgson. 2022. Evaluating realism in example-based terrain synthesis. *ACM Transactions on Applied Perceptions (TAP)* 19, 3 (2022), 1–18.

[49] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning.* PMLR, 2256–2265.

[50] Reto Stöckli, Eric Vermote, Nazmi Saleous, Robert Simmon, and David Herring. 2005. The Blue Marble Next Generation-A true color earth dataset including seasonal dynamics from MODIS. *Published by the NASA Earth Observatory* (2005).

[51] Severin Strobl, Arno Formella, and Thorsten Pöschel. 2016. Exact calculation of the overlap volume of spheres and mesh elements. *J. Comput. Phys.* 311 (2016), 158–172.

[52] Flora Ponjou Tasse, James Gain, and Patrick Marais. 2012. Enhanced texture-based terrain synthesis on graphics hardware. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 1959–1972.

[53] Ryutaro Tateishi, Bayaer Uriyangqai, Hussam Al-Bilbisi, Mohamed Aboel Ghar, Javzandulam Tsend-Ayush, Toshiyuki Kobayashi, Alimujiang Kasimu, Nguyen Thanh Hoan, Adel Shalaby, Bayan Alsaaideh, et al. 2011. Production of global land cover data–GLCNMO. *International Journal of Digital Earth* 4, 1 (2011), 22–49.

[54] Luis Oswaldo Valencia-Rosado, Zobeida J Guzman-Zavaleta, and Oleg Starostenko. 2020. Generation of synthetic elevation models and realistic surface images of river deltas and coastal terrains using cGANs. *IEEE Access* 9 (2020), 2975–2985.

[55] Richard F Voss. 1985. Random fractal forgeries. In *Fundamental Algorithms for Computer Graphics: NATO Advanced Study Institute directed by JE Bresenham, RA Earnshaw, MLV Pitteway.* Springer, 805–835.

[56] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision*

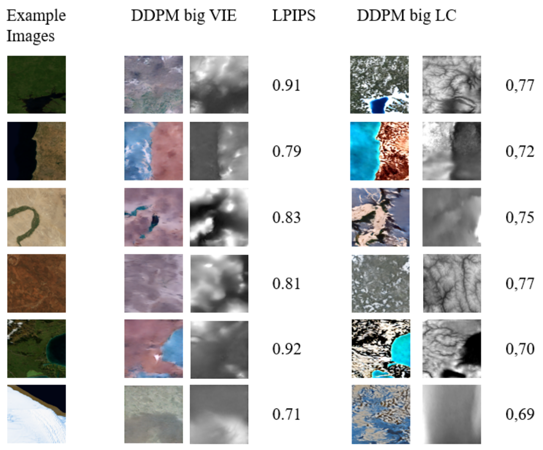

Figure 7: LPIPs metric comparing generated terrain to sample terrain for the DDPM big models. The left-hand image represents the example image and the images on the right are the respectively generated satellite images and grey-scale terrain

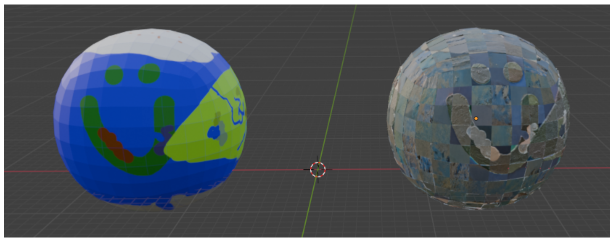

Figure 8: Illustration of the globe sketching interface, comparing the user sketch on the left to the generated globe on the right

and pattern recognition. 8798–8807.

[57] Yuxin Wu and Kaiming He. 2018. Group Normalization. arXiv:1803.08494 [cs.CV] https://arxiv.org/abs/1803.08494

[58] Jian Zhang, Chen Li, Peichi Zhou, Changbo Wang, Gaoqi He, and Hong Qin. 2022. Authoring multi-style terrain with global-to-local control. *Graphical Models* 119 (2022), 101122.

[59] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 586–595.

[60] Yiwei Zhao, Han Liu, Igor Borovikov, Ahmad Beirami, Maziar Sanjabi, and Kazi Zaman. 2019. Multi-theme generative adversarial terrain amplification. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–14.

[61] Howard Zhou, Jie Sun, Greg Turk, and James M Rehg. 2007. Terrain synthesis from digital elevation models. *IEEE transactions on visualization and computer graphics* 13, 4 (2007), 834–848.

[62] Di Zhu, Ximeng Cheng, Fan Zhang, Xin Yao, Yong Gao, and Yu Liu. 2020. Spatial interpolation using conditional generative adversarial neural networks. *International Journal of Geographical Information Science* 34, 4 (2020), 735–758.

A APPENDIX

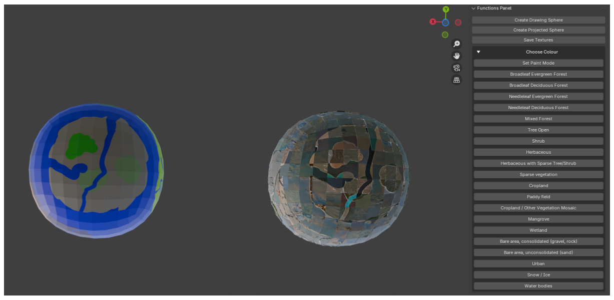
Functions Panel
Create Drawing Sphere
Create Projected Sphere
Save Textures
Choose Colour
Set Paint Mode
Broadleaf Evergreen Forest
Broadleaf Deciduous Forest
Needleleaf Evergreen Forest
Needleleaf Deciduous Forest
Mixed Forest
Tree Open
Shrub
Herbaceous
Herbaceous with Sparse Tree/Shrub
Sparse vegetation
Cropland
Paddy field
Cropland / Other Vegetation Mosaic
Mangrove
Wetland
Bare area, consolidated (gravel, rock)
Bare area, unconsolidated (sand)
Urban
Snow / Ice
Water bodies

Figure 6: Globe sketching interface. The left image shows a low-detail user sketch drawn on the spherical canvas, while the right image depicts the generated terrain based on the sketch. The interface allows users to select from a variety of landscape features, such as forests and water bodies, and apply them to specific regions on the globe. The generated terrain reflects the user's input, translating rough sketches into more detailed terrain.