



StyleDEM: a Versatile Model for Authoring Terrains

Simon Perche, Adrien Peytavie, Bedrich Benes, Eric Galin, Eric Guérin

► To cite this version:

Simon Perche, Adrien Peytavie, Bedrich Benes, Eric Galin, Eric Guérin. StyleDEM: a Versatile Model for Authoring Terrains. 2023. hal-04334315

HAL Id: hal-04334315

<https://hal.science/hal-04334315>

Preprint submitted on 10 Dec 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

StyleDEM: a Versatile Model for Authoring Terrains

Simon Perche, Adrien Peytavie, Bedrich Benes, Eric Galin & Eric Guérin

Many terrain modelling methods have been proposed for the past decades, providing efficient and often interactive authoring tools. However, they generally do not include any notion of style, which is a critical aspect for designers in the entertainment industry. We introduce StyleDEM, a new generative adversarial network method for terrain synthesis and authoring, with a versatile toolbox of authoring methods with style. This method starts from an input sketch or an existing terrain. It outputs a terrain with features that can be authored using interactive brushes and enhanced with additional tools such as style manipulation or super-resolution. The strength of our approach resides in the versatility and interoperability of the toolbox.

I. Introduction

Realistic and controllable terrain models are necessary for creating virtual worlds. The existing approaches encompass a variety of methods, including procedural generation, physically-based erosion simulations, and example-based synthesis. Existing methods provide a varying level of control, allowing the user to generate, edit, or modify synthetic terrains. However, they generally do not include any notion of style, which is a critical demand from designers in the entertainment industry.

We define style intuitively as related to the user perception of the overall quality and common properties of a terrain or a specific category with similar elevation and landmark characteristics. A key observation is that style has a tremendous impact on the perception of terrains: a highly irregular mountainous landscape from the Karakoram looks radically different from smooth hills in the Appalachians. Style is not only present at a large scale like in the orometry-based method proposed in [1], but it affects all ranges of scales, from large-scale geographic structures of hundreds of kilometres to landforms of a few meters. This diversity in style is the consequence of complex natural processes acting over varying temporal and spatial scales, including tectonics, stratification, aeolian and hydraulic erosion. Modelling such complex phenomena is intricate and comes at the price of computationally intensive, involved, and hard to understand and control simulations. Most erosion algorithms focus on restricted temporal and spatial scales. In effect, they only account for limited phenomena and simple materials, thus allowing only for the simulation of a limited range of landforms.

Another crucial observation is that artists prefer an interactive editing process when authoring terrains and therefore favour techniques that allow user control. While methods exist that allow for interactive modifications of landform features [2] or local style transfer for virtual worlds [3], little effort has been dedicated to combining the concept of style with an interactive edition framework. We address these limitations by proposing a novel approach that presents versatile interactive tools for editing terrains, including sketching, copy-and-paste sequence, and super-resolution (adding details) while providing ways for the designer to define or impose a given style at different levels (Figure 1).

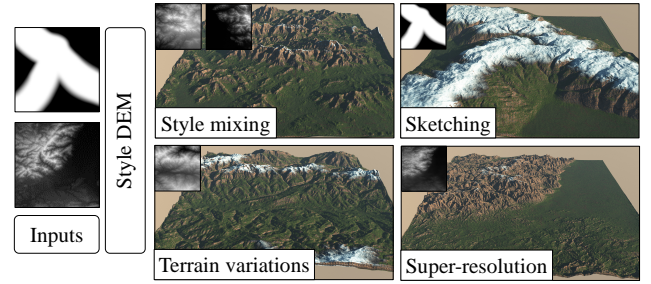


Figure 1. StyleDEM is a deep-learning model that offers versatile authoring possibilities. A range of tools is offered to users so they can easily author high-quality terrains.

More precisely, our contributions are: 1) a novel terrain model employing a StyleGAN architecture that describes and generates high-quality digital elevation models; 2) a variety of tools adapted to the model that are central in terrain authoring; 3) the introduction of terrain style at different scales, which allows for rich context-aware content generation.

II. Related work

Terrain generation methods in Computer Graphics can be classified as procedural generation, physically-based (erosion) simulation and synthesis from exemplars, which includes deep-learning algorithms. Given the identified goal of control, style, and synthesis from real digital elevation models, we briefly review the first two categories and focus on authoring frameworks that evidence interactivity. We refer the reader to [4] for a complete overview of terrain generation techniques.

Procedural terrain modelling relies on procedural noise, often combined with river network carving, to algorithmically reproduce the self-similarity across scales. Most techniques rely on a globally defined sum of noises [5] or assembly of procedurally defined and compactly supported primitives [6]. They are generally computationally efficient and lend themselves to parallel implementation on graphics hardware. Authoring control typi-

cally takes the form of applying noise with circular brushes [7] or matching curve and point constraints using warping [2], or using diffusion curves [8]. Recently, [9] introduced gradient-domain editing for terrain modelling, demonstrating its effectiveness for seamless blending of patches, copy-and-paste operations, and generation from control feature points and curves. Unfortunately, these approaches do not consider style and require careful and tedious editing and parameter tuning to obtain realistic landmarks.

Erosion simulation methods can be broadly classified into surface erosion algorithms and tectonic-based simulations. Surface erosion [5] simulate material detachment, transport, and deposition, possibly considering the strata of the bedrock [10], and enhance relief with sedimentary valleys and small-scale erosion landmarks such as gorges and ravines. These early approaches were improved in several ways by computing the acceleration or deceleration of the fluid to erode the bedrock or deposit sediments [11], combining a shallow water simulation with hydraulic erosion [12], or Smoothed Particle Hydrodynamics [13]. Those methods generate convincing small-scale erosion effects as long as the initial input terrain is sufficiently realistic and supplies large-scale landmarks. Our method addresses this issue by implicitly providing erosion-generated features learned from real-world examples via neural transfer.

Tectonic simulations, in contrast, attempt to reproduce large-scale erosion by taking into account the uplift of the bedrock produced by relative movement of the tectonic plates, balanced by different types of erosion, most often the Stream Power erosion [14]. Contrary to surface erosion, tectonic simulations produce realistic large-scale mountain ranges without needing an initial elevation. However, both methods are difficult to control and computationally intensive, which deters them from interactive authoring. In contrast, our method implicitly addresses the control via deep learning transfer from real-world features.

Example-based methods tackle the terrain realism by combining patches extracted from real-world digital elevation models. Their control is achieved by structure-sensitive warping to match sketched silhouettes [15], use of Conditional Generative Adversarial Networks to learn the correspondence between terrains and the sketch maps corollaries containing ridge and river lines and feature points [16]. Parallel texture-based synthesis [17] modifies the matching process to support style painting, region-based copy-and-paste, and curve and point manipulators. The assembly of terrain patches, even locally geomorphologically correct, is insufficient for generating globally consistent landscapes. Sparse modelling is another efficient way to generate high-resolution terrains from sketches [18] guided by exemplars. Recently, Scott *et al.* [19] proposed a breaching algorithm interlaced in multi-resolution example-based terrain synthesis to improve hydrological consistency. In contrast, our method encodes hydrological consistency at different scales in the latent space.

Deep-learning algorithms are a specific case of example-based approaches. Zhang *et al.* [20] used a modified version of a GAN

with low-resolution maps, global style information, and local style maps as input, and a discriminator capable of classifying different types of terrains. The generator is based on UNet architecture, and patch-based discriminators allow for the local control of style. Another approach, specialised in style embedding, was proposed in [21]. By using a cGAN to insert the embed theme into the terrain, the method can amplify an input low-resolution terrain into a high resolution with style variation. While producing compelling, high-quality results in terms of style transfer, this approach does not provide any edition tools. Recently, [22] trained a Variational Auto Encoder [23] combined with a GAN to generate a heightmap from a low-resolution map coupled to a sketch. While providing authoring tools such as sketching and terrain interpolation, results lack details. We propose a novel method that addresses the blind spot of previous approaches by developing a framework for simultaneous style manipulation and terrain editing.

III. Model

We introduce StyleDEM, a deep neural model that is based on the StyleGAN architecture [24] applied to Digital Elevation Models (DEMs), which are terrains represented as discrete heightfields denoted as \mathcal{T} . StyleGAN takes a large set of input images and encodes their style into its latent space. Inspired by this approach, we use latent space as the primary way to represent and manipulate digital terrains.

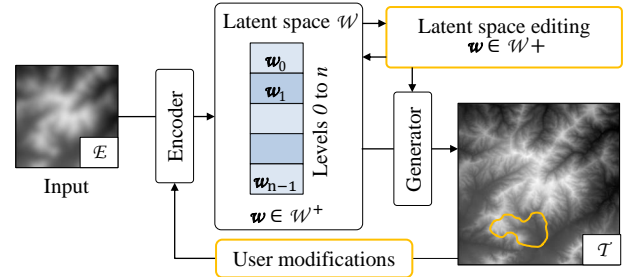


Figure 2. The input heightfield \mathcal{E} (a high or a low-resolution terrain, or even a sketch) is injected into the encoding process to produce a latent vector representation in \mathcal{W}^+ . The generator uses one or multiple \mathcal{W}^+ vectors to synthesise a new terrain \mathcal{T} . The user may modify the latent space to control the generation process, or directly edit \mathcal{T} and inject it again in the generator.

Our work consists of two main parts depicted in Figure 2. The StyleGAN itself is a *generator* trained on a carefully selected and designed dataset of DEMs. The generator synthesises a high-resolution terrain \mathcal{T} from its latent representation \mathcal{W} . Conversely, an inverter called *encoder* takes a latent vector corresponding to the terrain as its input and generates the output DEM. The user can interact at multiple levels of this pipeline, from user inputs to direct latent space modification. At each step, the output is directly rendered for preview or, when the result is satisfactory, streamed to a rendering pipeline.

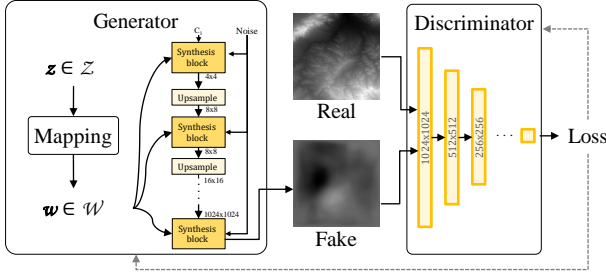


Figure 3. StyleGAN architecture. A vector from the latent space \mathcal{Z} is projected into a second latent space \mathcal{W} by using a mapping network that disentangles latent directions. This \mathbf{w} vector is then fed to the generator at various level of resolution, and the output is sent to a discriminator, which seeks to separate real and fake image. The generator and the discriminator are trained together in a zero-sum game. The results are utilised during the backward propagation phase.

Generator The generation step incrementally builds on the Generative Adversarial Network (GAN) architecture introduced in [25]. The generator denoted as \mathcal{G} creates a high-resolution terrain. During the training process, \mathcal{G} is coupled with another neural network, called the discriminator \mathcal{D} that attempts to detect whether the generated terrain is real or not. Therefore, the generator and the discriminator compete against each other: the generator tries to fool the discriminator by generating images resembling the ones found in the training database, whereas the discriminator tries to distinguish images generated by the generator from those in the training dataset. This architecture allows the network to be trained in an unsupervised way and does not require explicit specification of a loss function.

The introduction of the StyleGAN architecture [24] significantly improved the GAN model and demonstrated its performance by being the first to create photorealistic images while maintaining user control by taking advantage of its latent space \mathcal{W} . Here we extend the scope of StyleGAN to digital terrains. One particularity is the progressive growth of the output image performed by generators at increasing resolutions that take the previous layers as input, driven by the latent representation in \mathcal{W} . Starting at a reduced initial low resolution (in general 4×4), every step increases the resolution of the filters by a factor of two, and the generator employs the latent vector \mathbf{w} to add details. Another layer is specialized to transform these filters into images. Figure 3 illustrates this particular architecture.

Unfortunately, as demonstrated in [26], \mathcal{W} cannot represent every real image. Therefore, \mathcal{W}^+ vectors were introduced to express the necessary variability across models. The StyleGAN generator uses the \mathcal{W}^+ as a concatenation of 18 different 512-dimensional \mathbf{w} vectors for each layer. The vectors in \mathcal{W}^+ control the generation of every layer over the hierarchical process.

Encoder The StyleGAN architecture has a great synthesis power but lacks a so-called *inversion* process: what is the representation of an existing terrain in the latent space? Two main categories of inversion methods exist. Optimisation-based approaches compute

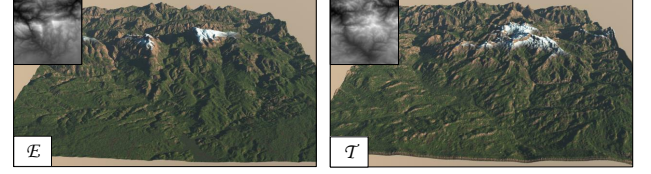


Figure 4. The encoder inverts an input terrain (left) into latent space, before the generator synthesises an approximation (right) using the latent vector representation.

the loss between the generated and target terrain. Starting from an initial random vector in \mathcal{W}^+ and using an optimiser and back-propagation, the system performs an optimisation over \mathcal{W}^+ . Despite its high-quality results, this method is computationally intensive, up to several minutes per image, which is prohibitive for an interactive application. We prefer the encoder-based approach for inverting a terrain, which implements another neural network, denoted as \mathcal{I} , encoding an image into a \mathcal{W}^+ vector. This method necessitates preliminary training and thus operates on pairs of terrain model and latent representations. We refer the reader to a review of GAN inversion from [27] for more details. The encoder option lends itself to the interactive generation pipeline because of its efficiency. Moreover, it allows for generalisation, because it can invert various inputs, including high or low-resolution DEMs or sketches. This property is essential in our model and allows for a range of use cases adapted to authoring. We use the pSp (pixel2style2pixel) architecture proposed in [28] that allows the StyleGAN to produce an output image based on an input image using the latent intermediate representation (See figure 4). In our experiments, we used L2 and LPIPS losses for the encoder that provided the best results. In the remainder of the paper, \mathbf{w} denotes a vector from \mathcal{W}^+ .

IV. Terrain authoring with style

We developed and studied a variety of authoring tools that benefit from the latent space representation and the generalisation possibilities of the encoder. Since the information given to the encoder uses the same format as the generator output, it intrinsically allows the interactive edition of the output and iterating through the process.

A. Versatility

We trained the encoder (Section V) using DEMs as input and $\mathbf{w} \in \mathcal{W}^+$ latent vector as output. Every vector \mathbf{w} encodes a high-resolution topography in the high-dimensional latent space of the generator. An immediate consequence is that the output of StyleDEM inherently retains consistent geomorphological properties. We exploit the inference capabilities of the encoder and feed it with new inputs that are different from those used during the training phase. This generalisation is beneficial for our approach as it allows the user to sketch low-resolution maps and edit existing DEMs while keeping consistency and generating necessary details.

Figure 5 demonstrates that StyleDEM can be employed in a

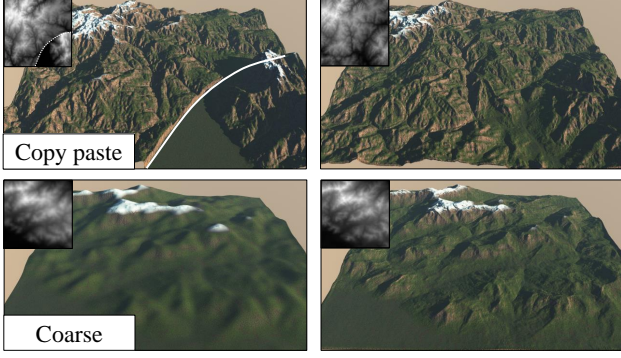


Figure 5. Different categories of input terrains (left) and the corresponding topographies produced by the full StyleDEM encoder and generator process (right): synthesised models exhibit more small-scale details and landforms while maintaining a global consistency.

variety of use cases: a DEM from a real terrain, a copy-and-paste editing, a user-defined sketch, and a low-resolution DEM. While a common strategy consists in utilizing an existing terrain before progressively modifying it, our method also allows using any of the different input data types designed by an artist while keeping the same format. In previous work, copy-and-paste operations either introduce seams (unless performed in the gradient domain as proposed in [9]) or require the use of a blending region to smoothly interpolate the elevations of argument terrain patches [6] which often results in inconsistent characteristics or cross fade of styles. In contrast, StyleDEM produces elevations that conform to the essence of the argument terrain patches. Moreover, artists often start with sketches depicting prominent features, such as mountain landmarks. In this configuration, the networks naturally synthesise terrains that follow the user design approach. By resembling the input sketch at every step, it provides consistent and realistic geomorphologic features. Eventually, when fed with a low-resolution model, the networks automatically synthesise a terrain approximating the input and enhanced with details. While in spirit similar to augmenting a terrain with small-scale features using sparse modelling [18] or procedural noise, the output preserves the style encoded in the latent space.

B. Style mixing

Inherited from the StyleGAN architecture, the generator is structured in 18 layers controlled by the latent vector. While the same latent vector controls the different layers during the training phase, we can also use different ones, which is the purpose of extending the latent space to 18×512 elements. This is particularly true when using the encoder that produces an extended latent vector $\mathbf{w} \in \mathcal{W}^+$ to increase the expressivity (see Section III). This delivers style mixing capacities: the large-scale structure and landmarks of one terrain can be mixed with the details of another one. The structure is represented in the upper layers of the latent vector, whereas details are connected to the lower ones.

We designed a tool that combines the global structure layers from

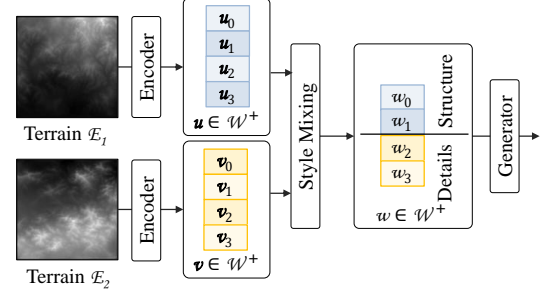


Figure 6. The style mixing proceeds as follows: two latent vectors \mathbf{u} and \mathbf{v} are computed using the encoder and then combined into \mathbf{w} to be finally fed to the generator. The upper part \mathbf{u}_0 to \mathbf{u}_{i-1} is combined to \mathbf{v}_i to \mathbf{v}_{n-1} .

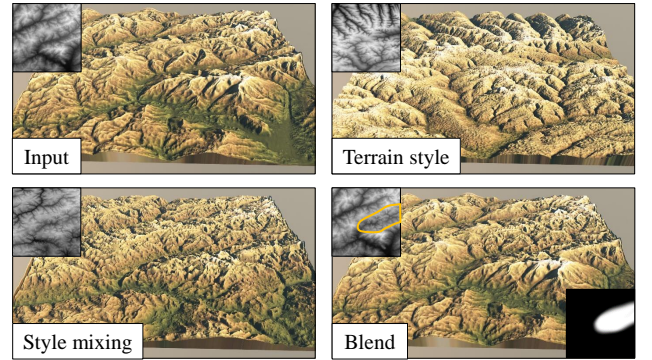


Figure 7. To modify the style over a region Ω of an input terrain \mathcal{T} , we first apply style mixing over the entire domain to obtain a new terrain \mathcal{T} , and then blend arguments with a mask derived from Ω .

a given terrain with the remaining layers of another terrain. Thus, we build an extended latent vector by merging two different parts of latent vectors (see Figure 6). As shown in [24], the network’s first layers (low-resolution) contain large-scale features, and the scale of the features decreases with the successive layers. This is a consequence of the growth of characteristics of the generator. Since small resolutions are generated in the first layers and then upsampled, only large features could be expressed. Therefore, the latent vector \mathbf{w} provided at a given resolution controls the style at this corresponding scale. For terrains, it corresponds to decreasing spatial features from mountains and valleys that define the global geomorphology to small-scale details erosion landmarks. This inherent style embedding of the model allows quick prototyping of terrains by changing details according to an input style, potentially resulting in a completely different visual perception of the initial terrain. The user selects the number of layers needed to apply the desired effect on the terrain, which allows for a balance between global and local control. Figure 8 shows the impact of the number of layers. The generator is fast enough to perform those operations interactively.

One limitation of the StyleDEM and StyleGAN architectures is the impossibility of mixing styles of the same levels in the same

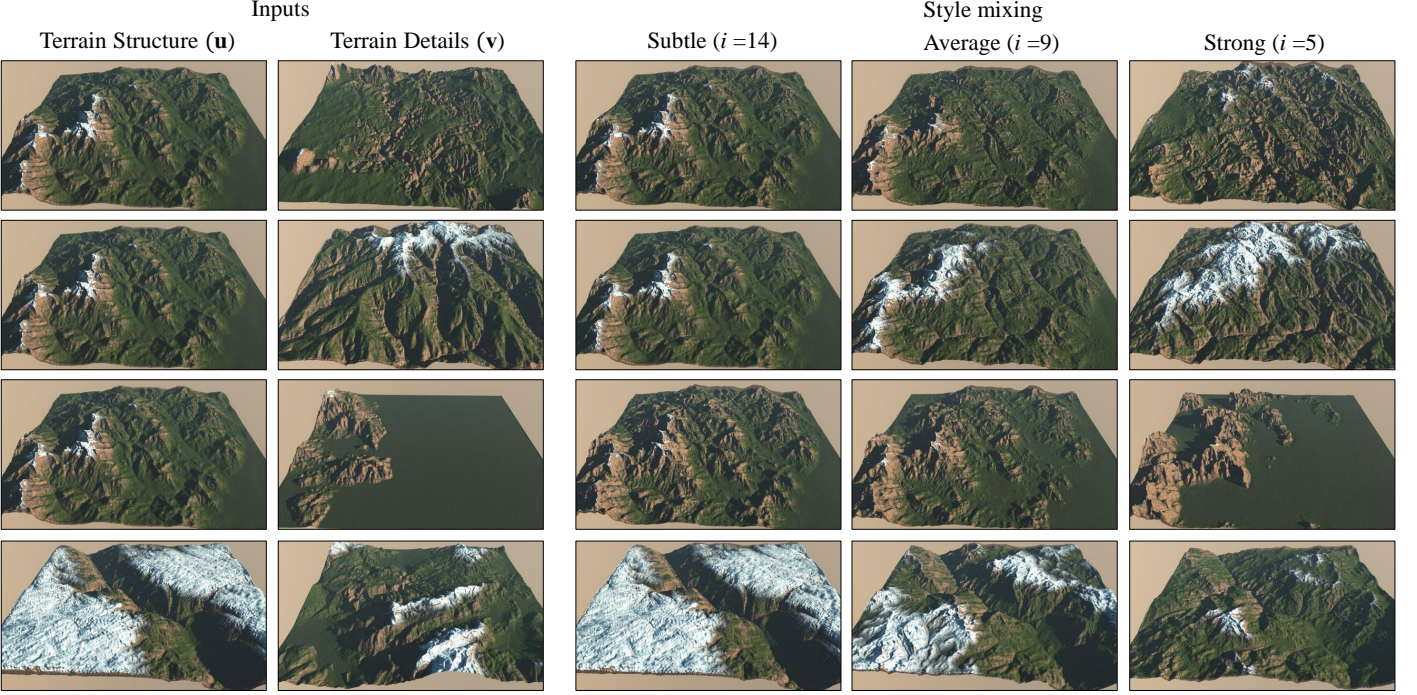


Figure 8. Influence of the i -th level when using style mixing operation. The user can adjust the style, from a subtle effects ($i = 14$) to substantial style enforcing ($i = 5$). The first three rows share the same input to illustrate the impact of different style mixing.

terrain at different locations. To overcome this limitation, we allow the user to generate two different stylised terrains and blend them directly in the altitude domain using an alpha mask defined by a brush (see Figure 7).

C. Interpolation

Another important property of the latent space is the consistency of interpolation as the interpolated latent vector embeds meaningful information about the terrain. Formally, given two latent descriptions \mathbf{u} and \mathbf{v} , we define the interpolated representation as: $\mathbf{w} = (1 - \alpha) \mathbf{u} + \alpha \mathbf{v}$, which interpolates the latent characteristics. When α varies, the resulting terrain $\mathcal{T}(\alpha)$ morphs from the first to the second terrains as shown in Figure 10.

Contrary to the direct elevation-based interpolation methods (such as elevation interpolation or optimal transport-based techniques), this approach generates intermediate terrains $\mathcal{T}(\alpha)$ that exhibit plausible geomorphological features due to the coherence within the latent space caused by the disentangled latent space \mathcal{W} during training.

D. Super-resolution

Super-resolution, or amplification, is crucial in terrain modelling and refers to simultaneous increasing of the resolution while inserting meaningful and consistent details. This step saves a lot of time for artists as it allows for focusing on the main structures and prominent features of a terrain.

We exploit the generalisation capabilities of StyleDEM and

particularly the way details can be generated when the input is a low-resolution map. We can treat two different final precisions based on the two StyleDEM models that have been trained: 30 and 5 meters resolution.

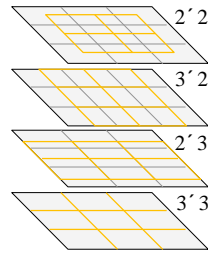


Figure 9. Patch decomposition.

Because the model has a fixed resolution of $1,024 \times 1,024$, generated terrains have a size of about $s \approx 30$ and $s \approx 5$ kilometers respectively. To overcome this limitation, we handle larger resolutions and sizes by dividing them into patches of a size s (see Figure 11).

We decompose input terrains \mathcal{T} of arbitrary resolution, *i.e.*, larger than the $1,024 \times 1,024$ resolution required by the networks, into a grid of k^2 patches. Detailed patches produced by the encoder have a normalised elevation range.

Therefore, we need to retarget each patch elevation to the original patch using a histogram matching, which guarantees recovering of the original elevation range and distribution. This strategy still produces discontinuities at the borders of the independently-generated patches. To overcome this limitation, we use a half-size $s/2$ overlapping and blending. We compute intermediate patches covering the boundaries as illustrated in Figure 9 and combine the three layers of intermediate patches with offset vectors $(s/2) \mathbf{x}$, $(s/2) \mathbf{y}$ and $(s/2) (\mathbf{x} + \mathbf{y})$ respectively. The process yields $(2k - 1)^2$ patches. We finally stitch them together using the minimum error boundary cut from [29].

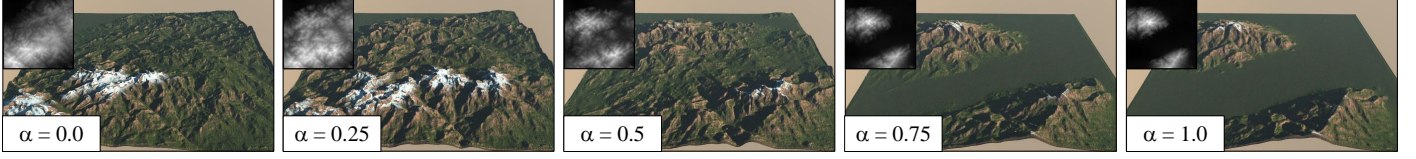


Figure 10. Terrains obtained using the interpolation between two latent space vectors.

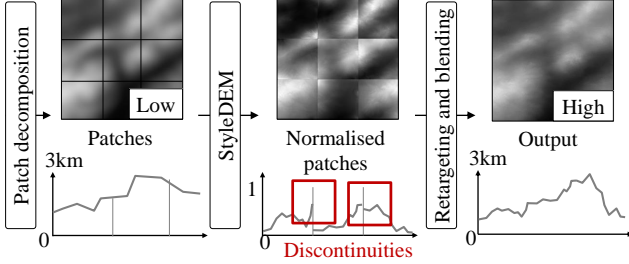


Figure 11. The input terrain is divided into $n \times n$ patches to adapt the size of the trained model. After processing by the networks, we retarget the heights of the (normalised) high-resolution patches and finally blend patches together.

Figure 12 shows a terrain that was processed by the super-resolution tool which multiplied the resolution by a factor of 3.

A StyleDEM is used with datasets of a specific resolution s corresponding to the terrain size of the training dataset. Working with different dimensions requires training of other networks. In our experiments, we trained two different models and applied them in cascade to increase the initial resolution of a terrain by several orders of magnitude. The 30-meters resolution model synthesises the structure of large-scale terrains (90 km), whereas the 5-meters networks are devoted to super-resolution. Super-resolution and style-mixing can also be combined in order to guide the details towards an exemplar (see Figure 12).

V. Results and discussion

The source used for creating datasets is composed of publicly available raster images of Digital Elevation Models (DEMs). We trained two different StyleDEM at 5 and 30 meters resolutions and built the datasets accordingly.

We used the IGN RGE ALTI database composed of 5×5 kilometers patches with the five-meter precision. We resampled tiles to $1,024 \times 1,024$ using bi-cubic interpolation to adapt to the network requirements. We downloaded 5,600 DEMs covering France and performed a selection based on their dynamic range so that flat terrains should not be over-represented and over-generated. We created elevation histograms and selected an equal number of representatives for each class. More precisely, every tile was classified into its dynamics rounded to the nearest ten meters. We randomly selected ≈ 20 terrains to maintain the balance between classes, resulting in 1,760 patches. The second dataset uses elevations from NASA SRTM consisting of 344 DEMs covering Europe, cut into 3,096 patches at 30 meters precision. We applied

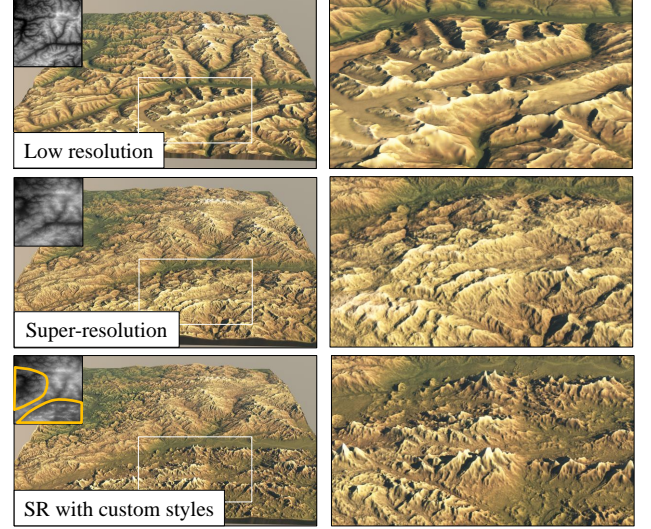


Figure 12. A 90 km terrain of 30 m precision was obtained with the super-resolution tool (center) from an initial low resolution model (top). We controlled the generation by adding iteratively two different styles (bottom).

the same process to handle the dynamic range and produced 1,900 patches.

We modified the StyleGAN to support a 16-bit grayscale precision format for terrain images, which were normalised to unit interval for training purposes. Because the training of a StyleGAN is computationally intensive, we relied on a high-performance computing centre. The generator \mathcal{G} , capable of generating terrains with a latent vector \mathbf{w} as input, was trained on 4 Nvidia V100 GPUs with 16 GB of memory, for 35 hours for the five meters precision and 75 hours for 30 meters precision.

Once the generator-training process was completed, we generated 20,000 synthetic terrains to train the encoder using randomly selected latent vectors, divided into 16,000 images for training and the others for testing. Training was performed on a single Nvidia V100 GPU with 16 GB of memory for 12 hours.

A. Implementation and performance

The scripts for generating the datasets were implemented in Python, and PyTorch was used for machine learning. StyleGAN [30] and the encoder pSp [28] were adapted from the author's implementation. The generator runs at 13 ms on Nvidia GPU hardware (see Table 1), thus providing interactive feedback to the user. All the terrains throughout the paper have been rendered

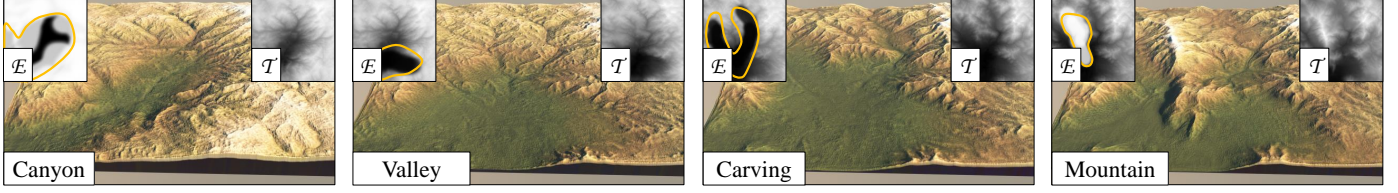


Figure 13. Four steps of a typical 5 minutes editing session. The user first sketched a low-resolution canyon, and then carved valleys and raised mountains by directly drawing low-resolution elevation level sets in the heightfield.

Tool	RTX3090	GTX970	# \mathcal{G}	# \mathcal{I}
Generator	12 ms	42 ms	1	0
Encoder	24 ms	112 ms	0	1
StyleDEM	37 ms	163 ms	1	1
Style mixing	81 ms	348 ms	1	2
Super-resolution 3×3	5.2 s	12.7 s	25	25
Super-resolution 6×6	16.2 s	50.6 s	121	121

Table 1. Performance for different operations. The last two columns correspond respectively to the number of passes of the generator \mathcal{G} and the encoder \mathcal{I} . Note that the super-resolution includes other operations such as blending or retargeting.

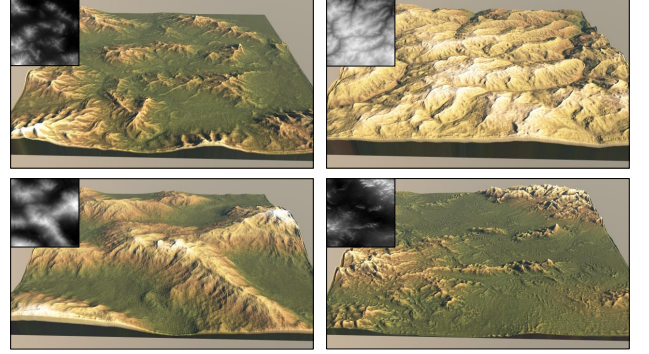


Figure 14. Four terrains obtained by different non-artists users during the qualitative testing process of our model. Only sketching tools have been used during these sessions.

using Eon-Software Vue with two shading types: realistic (see Figure 10) and cartographic (see Figure 12) when we wanted to emphasise the landforms details.

B. Control

We developed a plugin for the open-source modelling software Blender that grants accessibility to our algorithms to unfamiliar users. It integrates all the functionalities and interactive tools described in Section IV and can be used in all the Blender environments: modelling, shaders and render engine. See accompanying video for examples of user interaction.

To demonstrate the effectiveness of our approach, we conducted a qualitative study of our model with 3 non-artists who were asked to evaluate the modelling tools and comment on their effectiveness. Figure 14 shows examples of terrains produced by untrained users after a ≈ 5 minutes editing session only, starting from scratch. Users reported that they managed to author realistic terrains following their intent with different styles. Figure 13 shows a typical editing session performed by an experienced designer.

Figure 15 shows that our model adapts to different terrain sizes: the same sketch can generate different landforms corresponding to the target scale.

C. Comparison

Our approach lends itself to terrain authoring and amplification, and compares favourably to state-of-the-art methods with similar goals. Recently, the Generative Adversarial Terrain amplification (GATA) [21] offered a GAN architecture for style embedding

and amplification that is considered the state-of-the-art method for terrain amplification. While producing high-quality results by amplifying a low-resolution terrain with a specific input style, it lacks interactive editing tools. The pipeline not only offers style transfer and super-resolution but also proposes multiple authoring tools assembled into a unified framework for building new terrains or modifying existing ones. Moreover, the latent vector representation allows for transfer style at any scale.

Amplification and authoring have also been explored by [18] and [31] by using sparse modelling. However, these approaches do not provide terrain generalisation since patches are selected from exemplars without modification. Combining multiple styles is achievable at the expense of numerous exemplars, which intrinsically limits the performance of the selection in the dictionary. Figure 16, shows that [18] introduces repetitions artefacts on planar sketch surfaces, whereas our method manages to generate a palette of consistent landforms.

A recent method [20] implemented a style transfer approach using GANs to encode global and local styles independently. The GAN takes a level set as input and generates a terrain by combining two levels of details. This technique generates convincing variants of different styles. The main limitation comes from the restricted number of styles available, and styles are learned from specific hand-made datasets with evident generalisation limitations. In contrast, our method delivers a large variability by detecting the characteristics of a terrain that was never provided during training.

Finally, we compare our method to [16], which was, to the best of our knowledge, the first technique proposing a deep-learning

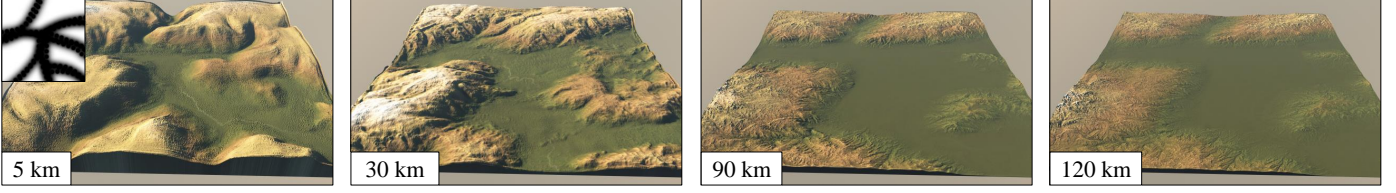


Figure 15. Our framework can adapt to any terrain size by using one tile of the two StyleDEM at 5 km or 30 km, or by composing multiple tiles for larger terrains.

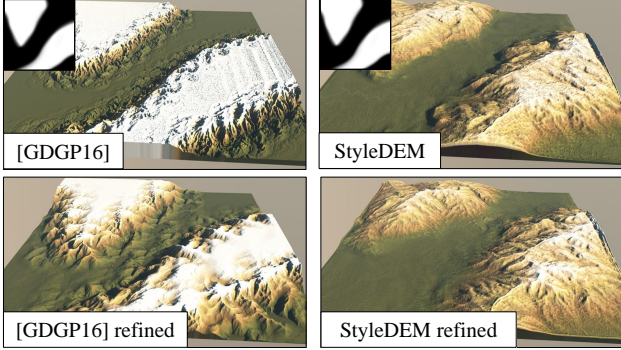


Figure 16. Comparison between sparse modelling [18] (left) and StyleDEM (right): our model allows details to be added while keeping global features. The refined row corresponds to the sketch going through StyleDEM model twice.

framework to terrain generation. Authors adopted a conditional GANs by giving sketches of ridges and rivers and generating high-resolution heightfields. In contrast, we propose a different and more versatile approach for authoring terrains. Style mixing allows fast prototyping using the same input, which would be difficult using the previous method since it requires training another network with a different dataset.

One crucial facet of our method is the versatility of tools. A single pipeline, with one training, offers extensive possibilities to artists. Table 2 shows a comparison of the various features available in previous works. We did not include erosion-simulation-based methods that are not relevant in this comparison.

D. Validation

We needed to preprocess the dataset to balance the presence of every class. As shown in Figure 17, balancing the dataset significantly reduces the number of failure cases and generates more stable results..

We choose an encoder approach to retrieve the latent vector w inside the StyleGAN latent space. An alternative method consists in iteratively optimising a random latent vector to match the input data using a standard loss based on the difference between input and the produced terrain. Theoretically, this method converts a real terrain into the latent space \mathcal{W} more precisely and therefore lends itself for style mixing with high-resolution models. In the case of low-resolution inputs or sketches, the optimisation faithfully

Article		Amplification	Sketches	Interpolation	Style-mixing	Style brush	Curve Constraints	Copy-paste	Code / Dataset
Example-based	Ours	•	•	•	•	•		•	• / •
	[20]		•	•	•	•			• /
	[21]	•		•	•				• /
	[16]	•	•					•	/
	[18]	•	•		•		•		• /
	[17]		•		•	•	•	•	/ -
Procedural	[9]	•	•				•	•	• / -
	[32]		•						• / -
	[8]						•		/ -
	[6]						•	•	/
	[1]		•		•		•	•	• / •
	[2]						•		/ -
	[15]						•		• / -
	[7]		•						/ -
	[33]		•				•		/ -

Table 2. Comparison of tools available in our method and other procedural and example-based systems. Dataset release and trained models are only compatible for machine learning methods, '-' means not applicable for this method.

reproduces the input terrain without introducing any learned landforms. By default, the optimisation uses mean squared error on VGG16 features. Adding an L2 loss directly on images yields better results since features such as mountains are located more precisely. We also found that optimisation is highly dependent on the initialisation. In contrast, the encoder trained with heightfields allows for a broader range of applications, as exemplified in Section IV.A. The optimiser takes ≈ 30 s to converge to a solution, which is to be compared to ≈ 37 ms using the encoder. Only the latter is compatible with interactive feedback. Figure 18 illustrates a comparison of these approaches.

We benefit from the training dataset of StyleGAN that is only composed of real digital elevation models. Visual inspection

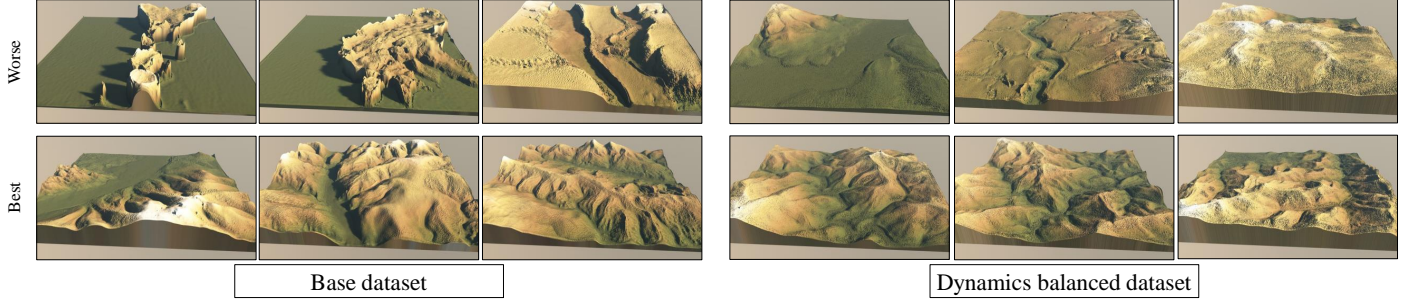


Figure 17. Comparison between unbalanced and balanced dataset. We randomly generated 100 terrains for each model, and picked the 3 (a) worst and (b) best-looking examples.

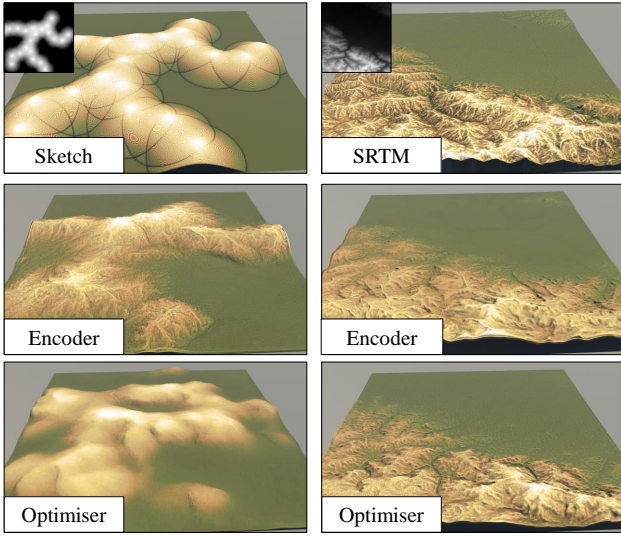


Figure 18. Comparison between encoder and optimiser-based methods for a sketch input (right) and a SRTM DEM (left). The optimiser delivers good results using real terrains and tends to be more accurate in this case. The encoder is a competitor adapted to using sketches and performs better in this configuration by generating features according to the prescribed inputs.

shows that other data-driven methods generally do not deliver outputs as consistent as erosion simulation approaches. Moreover, we performed experiments to estimate hydrological and geomorphological coherence by analysing the drainage properties. We applied a breaching algorithm to guarantee the drainage discharge over the entire terrain. We modified the algorithm to evaluate the volume of bedrock $v_{\mathcal{T}}$ removed to enforce drainage consistency. We compared our technique to other standard methods, and a real digital elevation model in the training dataset, reported in Table 3. Except [34] which is a simulation-based method, results demonstrate that $v_{\mathcal{T}}$ has perform better than other existing methods. This observation is confirmed by the visual inspection: the generator often breaches circular mountain ranges so that water can flow out of them. Figure 19 shows an example of a sketch featuring an endorheic basin breached by the encoder.

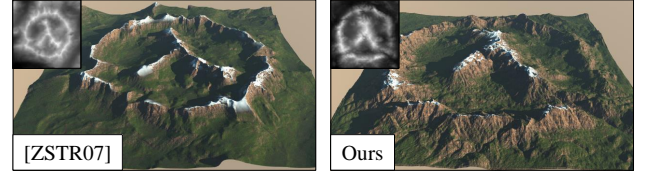


Figure 19. Comparison between texture synthesis [35] (left) and StyleDEM (right): our model better preserves landforms at different resolutions.

E. Limitations

Our approach has several limitations. One comes from the specific size of landforms used to train the generator. A specific encoder and generator need to be trained for every resolution, which requires intensive learning and an increasing amount of exemplar.

Moreover, the encoder does not achieve satisfactory results using small details in the sketch that are not visible in the inversion. After a quantitative study, we found that features smaller than 5% of the image size, around 50 pixels for a 1024×1024 terrain, may not be taken into account. Therefore, our method is useful for rapid prototyping, and obtained terrains can be refined with tools provided classical production pipelines.

StyleGAN works nicely at generalising objects from a given class. Nevertheless, it fails at inferring classes not presented in the training dataset. The patches used while training contains a single class or style because classes are consistent spatially. Thus, the networks cannot generate outputs that combine multiple styles, and the encoder cannot find a corresponding \mathbf{w} for such terrains. While we alleviated this issue by blending outputs of different styles in the elevation domain, this approach remains limited as a post-processing step, which prevents from operating consistently in the latent space.

Furthermore, every class may not be equally represented in the training dataset, thus leading to quality differences between classes. A tedious manual classification as a processing step would be necessary to conform to this equilibrium.

Method	$v_{\mathcal{T}}$
[35]	440
[34]	1
[16]	178
SRTM	57
Generator \mathcal{G}	117
Encoder \mathcal{I}	97

Table 3. The volume of bedrock removed by applying a complete breaching algorithm.

VI. Conclusion

We introduced StyleDEM, a versatile deep neural model for terrain authoring that allows designers to perform many editing tasks in the latent space while respecting the overall style of the terrain. From sketch-based authoring to style-transfer, by way of interpolation and super-resolution, the model embeds a description of terrains that inherently encompasses its geomorphological characteristics and guarantees consistency during generation. Experiments and a small-scale user study demonstrate its effectiveness. We provide a complete implementation of the model, along with datasets and a Blender addon that demonstrates the capabilities of this novel representation.

Acknowledgments

This work is funded by the project AMPLI ANR-20-CE23-0001, supported by Agence Nationale de la Recherche. This work was granted access to the HPC resources of IDRIS under the allocation 20XX-AD011013212 made by GENCI.

References

- [1] Argudo, O., Galin, E., Peytavie, A., Paris, A., Gain, J., and Guérin, E., “Orometry-Based Terrain Analysis and Synthesis,” *ACM Transactions on Graphics*, Vol. 38, No. 6, 2019.
- [2] Gain, J. E., Marais, P., and Strasser, W., “Terrain sketching,” *Proceedings of the Symposium on Interactive 3D Graphics and Games*, ACM, Boston, USA, 2009, pp. 31–38.
- [3] Emilien, A., Vimont, U., Cani, M.-P., Poulin, P., and Benes, B., “WorldBrush: Interactive Example-Based Synthesis of Procedural Virtual Worlds,” *ACM Transactions on Graphics*, Vol. 34, No. 4, 2015.
- [4] Galin, E., Guérin, E., Peytavie, A., Cordonnier, G., Cani, M.-P., Benes, B., and Gain, J., “A Review of Digital Terrain Modeling,” *Computer Graphics Forum (Proceedings of Eurographics 2019)*, Vol. 38, No. 2, 2019, pp. 553–577.
- [5] Musgrave, F. K., Kolb, C. E., and Mace, R. S., “The synthesis and rendering of eroded fractal terrains,” *Computer Graphics*, Vol. 23, No. 3, 1989, pp. 41–50.
- [6] Génevaux, J.-D., Galin, É., Peytavie, A., Guérin, É., Briquet, C., Grosbellet, F., and Benes, B., “Terrain Modeling from Feature Primitives,” *Computer Graphics Forum*, Vol. 34, No. 6, 2015, pp. 198–210.
- [7] de Carpentier, G. J. P., and Bidarra, R., “Interactive GPU-based Procedural Heightfield Brushes,” *Proceedings of the International Conference on Foundations of Digital Games*, ACM, Orlando, USA, 2009, pp. 55–62.
- [8] Hnaidi, H., Guérin, É., Akkouche, S., Peytavie, A., and Galin, É., “Feature based terrain generation using diffusion equation,” *Computer Graphics Forum*, Vol. 29, No. 7, 2010, pp. 2179–2186.
- [9] Guerin, E., Peytavie, A., Masnou, S., Digne, J., and James Gain, B. S., and Galin, E., “Gradient Terrain Authoring,” *Computer Graphics Forum (proceedings of Eurographics 2022)*, Vol. 44, No. 2, 2022, pp. 85–95.
- [10] Roudier, P., Peroche, B., and Perrin, M., “Landscapes synthesis achieved through erosion and deposition process simulation,” *Computer Graphics Forum*, Vol. 12, No. 3, 1993, pp. 375–383.
- [11] Neidhold, B., Wacker, M., and Deussen, O., “Interactive physically based Fluid and Erosion Simulation,” *Proceedings of the Eurographics Workshop on Natural Phenomena*, Eurographics Association, Dublin, Ireland, 2005, pp. 25–32.
- [12] Benes, B., “Real-Time Erosion Using Shallow Water Simulation,” *Proceedings of Virtual Reality Interactions and Physical Simulations*, Eurographics Association, Dublin, Ireland, 2007.
- [13] Křištof, P., Benes, B., Křivánek, J., and Šťava, O., “Hydraulic Erosion Using Smoothed Particle Hydrodynamics,” *Computer Graphics Forum*, Vol. 28, No. 2, 2009, pp. 219–228.
- [14] Cordonnier, G., Cani, M.-P., Benes, B., Braun, J., and Galin, E., “Sculpting Mountains: Interactive Terrain Modeling Based on Subsurface Geology,” *IEEE Transactions on Visualization and Computer Graphics*, Vol. 24, No. 5, 2018, pp. 1756–1769.
- [15] Tasse, F. P., Emilien, A., Cani, M.-P., Hahmann, S., and Bernhardt, A., “First Person Sketch-based Terrain Editing,” *Proceedings of Graphics Interface*, Canadian Information Processing Society, Montreal, Canada, 2014, pp. 217–224.
- [16] Guérin, E., Digne, J., Galin, E., Peytavie, A., Wolf, C., Benes, B., and Martinez, B., “Interactive Example-Based Terrain Authoring with Conditional Generative Adversarial Networks,” *ACM Transactions on Graphics (proceedings of Siggraph Asia 2017)*, Vol. 36, No. 6, 2017.
- [17] Gain, J., Merry, B., and Marais, P., “Parallel, Realistic and Controllable Terrain Synthesis,” *Computer Graphics Forum*, Vol. 34, No. 2, 2015, pp. 105–116.
- [18] Guérin, E., Digne, J., Galin, E., and Peytavie, A., “Sparse representation of terrains for procedural modeling,” *Computer Graphics Forum (proceedings of Eurographics 2016)*, Vol. 35, No. 2, 2016, pp. 177–187.
- [19] Scott, J. J., and Dodgson, N. A., “Example-based terrain synthesis with pit removal,” *Computers and Graphics*, Vol. 99, 2021, pp. 43–53.
- [20] Zhang, J., Li, C., Zhou, P., Wang, C., He, G., and Qin, H., “Authoring multi-style terrain with global-to-local control,” *Graphical Models*, Vol. 119, 2022, p. 101122.
- [21] Zhao, Y., Liu, H., Borovikov, I., Beirami, A., Sanjabi, M., and Zaman, K., “Multi-Theme Generative Adversarial Terrain Amplification,” *ACM Trans. Graph.*, Vol. 38, No. 6, 2019.
- [22] Naik, S., Jain, A., Sharma, A., and Rajan, K. S., “Deep Generative Framework for Interactive 3D Terrain Authoring and Manipulation,” ????
- [23] Kingma, D. P., and Welling, M., “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
- [24] Karras, T., Laine, S., and Aila, T., “A Style-Based Generator

- Architecture for Generative Adversarial Networks,” 2019, pp. 4401–4410.
- [25] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., “Generative Adversarial Nets,” *Advances in Neural Information Processing Systems*, Vol. 27, edited by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Curran Associates, Inc., 2014.
 - [26] Abdal, R., Qin, Y., and Wonka, P., “Image2stylegan: How to embed images into the stylegan latent space?” *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4432–4441.
 - [27] Xia, W., Zhang, Y., Yang, Y., Xue, J.-H., Zhou, B., and Yang, M.-H., “GAN Inversion: A Survey,” ????
 - [28] Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., and Cohen-Or, D., “Encoding in Style: a StyleGAN Encoder for Image-to-Image Translation,” *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, ????, pp. 2287–2296. ISSN: 2575-7075.
 - [29] Efros, A. A., and Freeman, W. T., “Image Quilting for Texture Synthesis and Transfer,” 2001, p. 341–346.
 - [30] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T., “Analyzing and Improving the Image Quality of StyleGAN,” *Proc. CVPR*, 2020.
 - [31] Argudo, O., Andujar, C., Chica, A., Guérin, E., Digne, J., Peytavie, A., and Galin, E., “Coherent multi-layer landscape synthesis,” *The Visual Computer*, Vol. 33, No. 6, 2017, pp. 1005–1015.
 - [32] Gaillard, M., Benes, B., Guérin, E., Galin, E., Rohmer, D., and Cani, M.-P., “Dendry: a procedural model for dendritic patterns,” *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, Montreal Quebec Canada, 2019, pp. 1–9.
 - [33] Gènevaux, J.-D., Galin, É., Guérin, É., Peytavie, A., and Benes, B., “Terrain Generation Using Procedural Models Based on Hydrology,” *ACM Transaction on Graphics*, Vol. 32, No. 4, 2013, pp. 143:1–143:13.
 - [34] Cordonnier, G., Galin, E., Gain, J., Benes, B., Guérin, E., Peytavie, A., and Cani, M.-P., “Authoring Landscapes by Combining Ecosystem and Terrain Erosion Simulation,” *ACM Transactions on Graphics*, Vol. 36, No. 4, 2017, p. 134.
 - [35] Zhou, H., Sun, J., Turk, G., and Rehg, J. M., “Terrain Synthesis from Digital Elevation Models,” *IEEE Transactions on Visualization and Computer Graphics*, Vol. 13, No. 4, 2007, pp. 834–848.