Gradient Terrain Authoring

Eric Guérin[1], Adrien Peytavie[1], Simon Masnou[2], Julie Digne[1], Basile Sauvage[3], James Gain[4], and Eric Galin[1]

[1]Univ Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France
[2]Univ Lyon, UCBL, CNRS UMR 5208, Institut Camille Jordan, F-69622 Villeurbanne, France
[3]Université de Strasbourg, CNRS, ICube UMR 7357, France [4]University of Cape Town

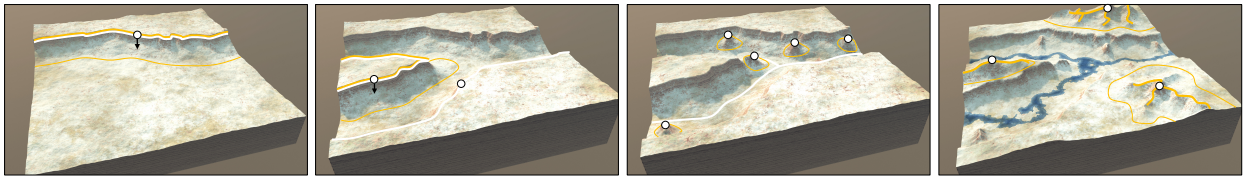

Figure 1: *Our framework allows the user to quickly sketch terrains using a combination of elevation and gradient constraints. This example was authored in a few minutes by a non-artist. The ridges of the canyon were created by placing elevation (white) and gradient (yellow) constraints along feature curves, whereas hills and crest lines where created with gradient constraints over a user-controlled region.*

Abstract

Digital terrains are a foundational element in the computer-generated depiction of natural scenes. Given the variety and complexity of real-world landforms, there is a need for authoring solutions that achieve perceptually realistic outcomes without sacrificing artistic control. In this paper, we propose setting aside the elevation domain in favour of modelling in the gradient domain. Such a slope-based representation is height independent and allows a seamless blending of disparate landforms from procedural, simulation, and real-world sources. For output, an elevation model can always be recovered using Poisson reconstruction, which can include Dirichlet conditions to constrain the elevation of points and curves.

In terms of authoring our approach has numerous benefits. It provides artists with a complete toolbox, including: cut-and-paste operations that support warping as needed to fit the destination terrain, brushes to modify region characteristics, and sketching to provide point and curve constraints on both elevation and gradient. It is also a unifying representation that enables the inclusion of tools from the spectrum of existing procedural and simulation methods, such as painting localised high-frequency noise or hydraulic erosion, without breaking the formalism. Finally, our constrained reconstruction is GPU optimized and executes in real-time, which promotes productive cycles of iterative authoring.

CCS Concepts

• ***Computing methodologies*** → ***Shape modeling;***

1. Introduction

Digital terrains, usually encoded as a heightfield of elevation values, contribute to the realistic portrayal of natural environments and serve an important role in a range of applications, including games, film, simulation and training. Nevertheless, the effective authoring of terrains remains in many ways an unsolved problem. This is primarily due to the delicate balance between artistic control, which requires a range of tools at different levels of abstraction enabling artists to concisely match their design intent, computational performance, so that the design loop between specification and realisation is as fast and productive as possible, and plausibility, in that the resulting terrains are perceived as realistic.

To meet this confluence of requirements we depart from the conventional elevation-centric approach. Instead, we undertake authoring in the gradient domain and map back to the elevation (heightfield) domain when required for rendering and export. Exploiting gradients for terrain synthesis is not without precedent. It has found use for terrain blending along seams between patches using Shepherd interpolation [TGM12] and between multiresolution layers using a Laplacian image pyramid [BCA*14]. These represent rather restricted use cases, but they are indicative of one of the many benefits of a gradient approach, namely the seamless blending of data from varied sources. Other advantages include: the control and expressive power offered by a wide range of painting, sketching, and

copy-paste authoring tools (see Figure 1), and the ease with which previous procedural and simulation methods can be co-opted and transferred to the gradient domain; responsive performance, with real-time updates for terrain resolutions up to 1024×1024, and perceptual realism, as demonstrated by the variety and realism of terrains generated in our validation experiments and user tests.

In terms of implementation, mapping from discretised elevations to gradients involves a simple slope operator. However, the inverse reconstruction is more involved and requires a carefully-optimized multigrid solution of the Poisson equation if real-time mapping is to be attained. To achieve a fully-featured authoring framework requires a variety of sketching and drawing tools, many of which are based on imposing elevation constraints at points and along curves. We achieve this using Dirichlet constraints in the interior of the gradient domain. This approach has the benefit of enforcing hard elevation constraints over soft gradient constraints if there is a conflict between the two forms, which generally matches users' expectations in terms of modelling behaviour. We also incorporate region-based tensor operators, which serve to re-align and scale gradients and offer further modelling flexibility. In summary, the technical contributions of this paper are as follows:

- Introduction of the gradient domain as a unifying formalism for terrain authoring, made possible by GPU-optimized multi-grid reconstruction that supports elevation matching in the form of Dirichlet constraints.
- Development of a complete authoring toolbox, including, but not limited to, sketching tools that impose point and curve constraints on elevation and gradient values; painting tools that modify regions of the terrain to smooth, roughen and align while retaining the underlying geomorphological patterns, and copy-paste operations to transfer section of a source into the destination terrain subject to warping and alignment as needed.
- Extension beyond direct authoring to global synthesis tasks in the terrain production pipeline, such as constrained reconstruction from contour, ridge or river line-maps, super-resolution upsampling of a coarse terrain, and as an improved basis for cGAN training [GDG*17].

2. Related work

Given the identified goals of control, performance, and realism, we choose to focus our review of related work on heightfield authoring frameworks that evidence real-time, or, at worst, interactive response rates. This precludes volumetric terrains [PGMG09, PGP*19], and authoring systems with a non-interactive feedback loop [ZSTR07, CGG*17]. For a broader perspective on digital terrain modelling the reader is referred to the review of Galin *et al.* [GGP*19].

The problem of terrain modelling has historically been tackled with one of three broad strategies: procedural modelling, geomorphological simulation, or example-based derivation.

Procedural terrain modelling uses noise synthesis, often combined with river network carving, to algorithmically reproduce the self-similarity across scales evident in real terrain. Authoring control typically takes the form of applying noise with circular brushes [dCB09] or matching curve and point constraints using warping [GMS09], shortest path traversal [RME09], multi-frequency blending [BCA*14], or diffusion [HGA*10]. These approaches are often a good fit for GPU implementation and tend to be computationally efficient. Unfortunately, exploiting self-similarity only carries realism so far, since it neglects the underlying geomorphology and can lead to an identifiable uniformity of style.

Geomorphological simulations achieve realism by directly encoding the underlying processes, such as tectonic uplift and hydraulic erosion, that give rise to real-world landforms. While such systems can achieve interactive response through GPU optimization [NWD05] and artistic input, for instance by sculpting tectonic uplift [CCB*18] or painting regions of hydraulic erosion [VBHŠ11], their inherent indirection is problematic from an authoring perspective. The effect of additive (uplift) and subtractive (erosion) tools can be hard for a user to predict, often requiring time-consuming trial and error in search of an envisaged outcome.

Example-based methods offer the promise of perceptual realism by exploiting real-world heightfield data obtained from scanning campaigns. Here, control is achieved by structure-sensitive warping to match sketched silhouettes [TEC*14, KRS15], use of Conditional Generative Adversarial Networks to learn a correspondence between terrains and their sketch map corollaries containing ridge and river lines and feature points [GDG*17], or modification of the matching process in parallel texture synthesis to support style painting, region-based copy-and-paste, and curve and point manipulators [GMM15].

Parallel texture-based terrain synthesis [GMM15] represents the strongest competitor in terms of realism, efficiency and user control. Nevertheless, gradient authoring has several significant advantages. First, copy-paste operations in texture synthesis require that the source terrain be indexed in an acceleration database: a pre-processing step that can take several minutes. Second, any operation outside the texture synthesis toolchain effectively puts a halt to further texture edits, making it difficult to switch between paradigms. Finally, there are issues of extensibility. For instance, our approach allows gradient constraints to be set independently of elevation (see Figure 4). An equivalent tool does not exist in [GMM15]'s system and would require a redesign of the underlying texture synthesis process. In contrast, gradient authoring benefits from ease of toolset configurability and seamless interoperation with other authoring paradigms.

Table 2 (see Appendix) provides a feature-wise comparison of existing terrain authoring systems, including our approach. In addition to distinguishing between the types of control, we also report on whether a given technique is capable of interactive ($1 - 10$ Hz) or real-time (> 10 Hz) update rates for modelling operations applied to heightfields of reasonable resolution (512×512 or larger).

In a broader context, gradient-domain representations have been successfully used for processing images and textures. Solving Poisson equations allows for various operations, including image editing [PGB03], image synthesis and blending [DBP*15], and texture stitching [DSWH14]. Fundamental properties include invariance by offsetting, incorporating constraints, and generating smooth transitions. We leverage these properties extensively for terrain editing.

Furthermore, editing surfaces by interacting with derivatives has proven effective in approaches such as Laplacian Surface Editing [SCOL*04]: some mesh vertices undergo a user-defined deformation, the displacement of the remaining vertices is retrieved by inverting a sparse linear system. Using Laplacian coordinates (the simplest form of differential coordinates) allows encoding and preservation of local detail throughout the editing process. Gradient field manipulation of surfaces has also been studied [YZX*04]: triangles experience some deformation (*e.g.*, rotation and scaling), possibly disconnecting them, which yields a new vector field that leads to new vertex positions by solving a Poisson equation. Our approach is different since we design tools operating directly on the gradient field without explicit triangle manipulation.

3. Gradient terrain model

Our authoring framework (see Figure 2) rests on the twin foundations of a gradient field $\mathbf{g}$ and associated elevation constraints c. If initial elevation data h is provided as a starting point this can easily be converted to its gradient counterpart via a once-off gradient extraction process. Once in the gradient domain, authoring is accomplished using tools that generate a combination of hard elevation constraints, which enforce user-specified elevations over points, curves, and regions, and soft slope constraints, achieved by directly modifying the gradient field. In cases of conflict, hard elevation constraints automatically take precedence over soft gradient constraints. In this process the artist iteratively fashions a revised gradient model $\widetilde{\mathbf{g}}$ and a set of elevation constraints c. For visualization and export purposes this can be converted back to an elevation representation $\widetilde{h}$ using a reconstruction process.

Many extant modelling tools can be easily transferred to the gradient domain. For example, high frequency noise modulation applies equally well to both elevations and gradients. Where this is not the case, it is possible to dip into the elevation domain (via reconstruction), and apply an elevation tool, such as hydraulic erosion, before returning to the gradient domain (via extraction).

We turn now to providing a mathematical formalism for this approach. Let the elevation function mapping planar coordinates to height be denoted as $h : \mathbb{R}^2 \to \mathbb{R}$. The corresponding gradient is a two-dimensional vector field defined by:

$$\mathbf{g} = \nabla h = \left(\frac{\partial h}{\partial x}, \frac{\partial h}{\partial y} \right)$$

For a regular gridded sampling: given a discrete heightfield representation h_{ij}, $(i, j) \in \{0, \ldots, n\}^2$ with $n \times n$ grid points and a grid spacing of δ over the square domain $\mathcal{B}$, the associated discrete gradient is:

$$\mathbf{g}_{ij} = \left(h_{i+1,j} - h_{i-1,j}, h_{i,j+1} - h_{i,j-1} \right) / 2\delta$$

In fact, gradient extraction is simply the application of this equation. Finally, strict elevation constraints are realised by Dirichlet conditions prescribed by an elevation function $c : \Omega \to \mathbb{R}$ on the subdomain $\Omega \subset \mathcal{B}$. This formulation is surprisingly flexible, in that Ω can incorporate any combination of points, curves and regions.

Reconstruction. Deriving an elevation heightfield $\widetilde{h}$ from its discrete gradient representation $\widetilde{\mathbf{g}}$ over a domain Ω is an over-constrained problem that requires solving a linear system with twice as many constraints as unknowns. One common strategy is to employ a Poisson equation formulation by taking the divergence of the gradient vector field $f = \nabla \cdot \widetilde{\mathbf{g}}$ and finding a function $\widetilde{h}$ that satisfies $\Delta \widetilde{h} = f$.

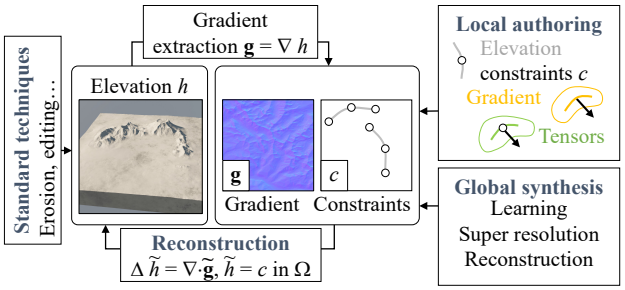

Figure 2: *Gradient terrain authoring involves extracting gradients from source elevations, applying local authoring or global synthesis tools to modify the resulting gradient field and impose elevation constraints, and reconstructing the resulting elevation changes. Such a framework also accommodates standard tools that operate only in the elevation domain. (In our figures, gradient and elevation constraints are depicted in yellow and white, respectively, whereas tensor operators are shown in green).*

Fast Poisson solvers are typically based either on analytic solutions of the Fourier transform, or numeric multigrid methods. While the former are computationally efficient, their utility is limited because they cannot handle general diffusion [HGA*10] or the combination of the Poisson equation with Dirichlet constraints on the domain interior.

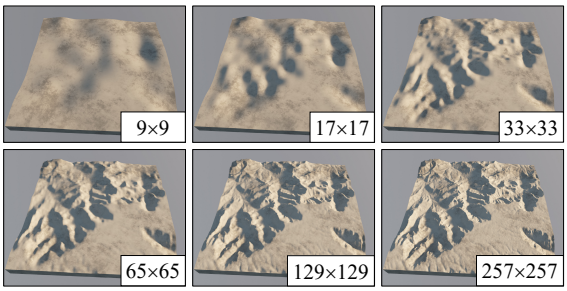

Figure 3: *Intermediate levels of the multi-resolution solver.*

Multigrid methods are thus more suited to our context. They are also backed by a theoretical proof of rapid convergence. When operating on grids the choice is between a black-box [Den82, BD96] or geometric [Sha08] strategy. Black-box methods are usually employed when the set of partial differential equations and the corresponding set of coefficients are not known beforehand. This is not our case, and we thus prefer the latter and implement a geometric version of the half V-cycle multigrid that begins at the coarsest resolution and applies iterative refinement level-by-level until the finest resolution is attained (see Figure 3). On the one hand, this is slower because it does not benefit from residual back-propagation, but, on the other, it is significantly simpler to implement since only the

(bilinear) prolongation operator and systems at different resolution layers are required. To compute the system at lower resolutions, we adopt a fusion strategy that accounts for constraint priority: hard constraints have the highest priority and automatically apply at the lowest resolution. We also average multiple constraints of the same type. In practice, convergence is fast enough to support real-time interaction (see Section 6).

The tension between hard elevation and soft gradient constraints has one important implication. Because the gradient field only acts as a guide, the reconstruction process may result in a mismatch between the source vector-field and resulting elevations. This can be reconciled through a two-pass regularization: first, a reconstruction $\widetilde{\mathbf{g}} \mapsto \widetilde{h}$, followed by an extraction $\widetilde{h} \mapsto \widetilde{\mathbf{g}}'$. This represents a full cycle in Figure 2 starting from the user-edited vector-field and returning to the gradient field. As a consequence, the Dirichlet conditions can subsequently be dropped as they become implicitly incorporated into both the elevation and gradient representations. Note that this operation reduces the curl component of the vector field that may have been introduced during gradient modifications. This is just one of the implications of the gradient formalism. Section 4 explores more details of its application to landform authoring, including the development of copy-paste, tensor field modification, and procedural synthesis operations.

Global synthesis. Most global terrain synthesis operations rely on an image input format. Fortunately, for this purpose gradient maps can be encoded as two-channel (red r_{ij} and green g_{ij}) images, as follows: $(r_{ij}, g_{ij}) = 2^{p-1}(1 + \mathbf{g}_{ij}/m)$, where m is the component-wise maximum of the gradient values and p is the available bits per channel. Although this quantization introduces a loss of precision it allows us to leverage a variety of image processing and synthesis algorithms, as demonstrated in Section 5 for specific terrain-oriented super-resolution and deep learning applications.

4. Terrain authoring

Local gradient map modifications can be combined with Dirichlet constraints on elevation to provide a rich terrain-authoring toolset. The options available to digital artists include editing gradients directly, modelling with tools specialised for sculpting particular landforms, warping gradients by applying tensor maps via matrix multiplication, and even employing tools transplanted from prior work in procedural and simulation-based modelling. All of these mechanisms are implemented with appropriate painting, drawing, and copy-paste tool metaphors.

4.1. Gradient editing

The gradient map, being the first derivative of elevation, captures the steepness and orientation of slopes, which represent quantities that are easily interpreted by users. As a consequence, the direct manipulation of gradients is relatively intuitive.

One option in this regard is to locally amplify or suppress slopes by performing a scalar multiplication of the gradient. Note that this is quite different in both principle and effect from multiplying elevations, which leads to scaling relative to a rather arbitrary sea level value ($h = 0$). Instead, gradient amplitude manipulation supports appropriate modification of a landform's slope dynamics.

A second broadly-applicable manipulation is to copy and paste sections from a gradient source into a gradient target. The pasted elevation blends seamlessly with the original terrain because surrounding elevations are matched across the boundary, as is to be expected based on similar behaviour in Poisson image editors [PGB03]. A live session capture of these tools in action is provided in the accompanying video.

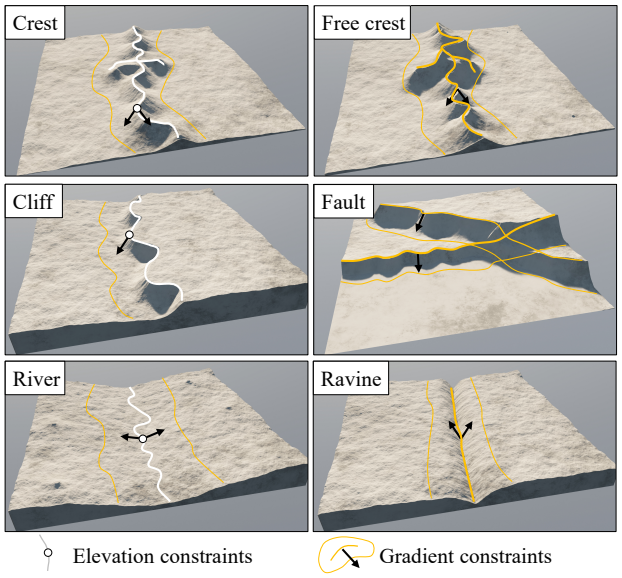

Figure 4: *Using combinations of elevation and gradient constraints, a variety of geomorphological features can be authored.*

4.2. Landform-adapted tools

It is helpful to provide, particularly for non-experts, a suite of tools purpose-built for the creation of specific geological features. In general, these follow the pattern of a sketched curve with an offset radius defining the area of influence of the modification. What varies is the exact combination of elevation and gradient constraints imposed.

Figure 4 shows a catalogue of typical landforms and the required configuration of constraints. The group of crests, cliffs, and rivers rely primarily on elevation constraints. Crests and cliffs incorporate additional steep gradient constraints to enforce the slope on either side, while rivers require at most a gentle upward gradient for the banks. The other group of free crests, faults and ravines are built purely with gradient constraints and require no enforcement of elevation.

It is worth noting that the superposition of gradient constraints is additive, unlike for elevation constraints. This is illustrated by the different behaviour of crests and faults in Figure 4 in areas where their defining curves cross or overlap.

4.3. Tensor-map transformations

If terrain elevations are represented as a scalar map on a 2D domain, then the associated gradients are a vector (first-order tensor) map over the same domain. This means that gradients are amenable to transformation through matrix multiplication with second-order (2×2) tensors. In the most general case, a modified gradient map $\widetilde{\mathbf{g}}$ can be obtained as the matrix product $\widetilde{\mathbf{g}} = \mathbf{A}\,\mathbf{g}$ of a tensor map $\mathbf{A}$ and the original gradient map $\mathbf{g}$. This has a number of ramifications for modelling.

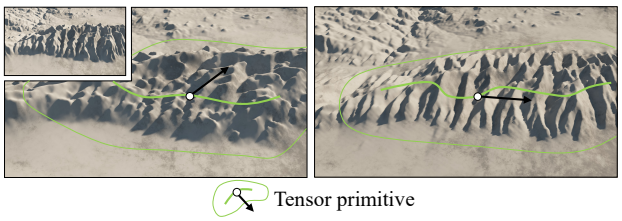

Figure 5: *Applying different directional tensors (see arrows) to a mountain chain leads to various orienting effects.*

Simple slope direction amplification can be obtained by applying a constant direction-oriented tensor:

$$\mathbf{D} = \mathbf{I} + \mu \begin{pmatrix} \cos 2\theta & \sin 2\theta \\ \sin 2\theta & -\cos 2\theta \end{pmatrix},$$

where $\mathbf{I}$ is the identity tensor, θ is an alignment angle, and $\mu \in [0,1]$ is the weight accorded to the modification. This operator acts to locally drag gradients within a defined region of influence Ω towards the direction defined by the angle θ. Figure 6 demonstrates the angle response imposed by a directional tensor with $\theta = 60°$, using two different strengths, $\mu = 0.9$ and $\mu = 0.5$.

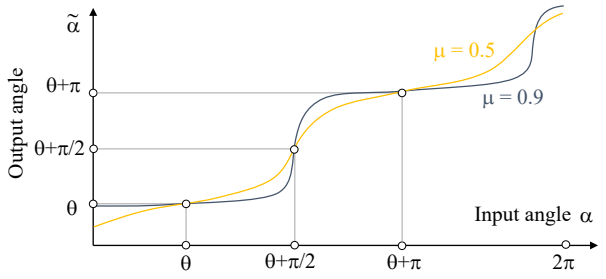

Figure 6: *Direction angle of the gradient before ($\mathbf{g}$) and after ($\widetilde{\mathbf{g}} = \mathbf{D}\,\mathbf{g}$) modification with two directional tensors of different strength.*

Skeleton-controlled tensors allow modification to be guided by user-placed skeleton primitives in the form of points, line segments, curves, and discs. The impact of tensors is strongest at the primitive and tails off with distance. This is accomplished with a composite tensor field $\mathbf{A}(\mathbf{p})$ at a point $\mathbf{p}$, defined by:

$$\mathbf{A}(\mathbf{p}) = \mathbf{I} + \sum_i \mathbf{A}_i(\mathbf{p}),$$

where a constant background identity tensor $\mathbf{I}$ is defined on the entire domain and each primitive i contributes a primitive-specific tensor field $\mathbf{A}_i(\mathbf{p})$ in a neighbourhood of compact support Ω_i. In turn, individual component tensor fields have the form:

$$\mathbf{A}_i(\mathbf{p}) = \mu_i\, \alpha(d(\mathbf{p})/r_i)\mathbf{T}_i(\mathbf{p}, \mathbf{u}(\mathbf{p})).$$

Here, $\mu_i \in [0,1]$ is the strength of the primitive, α is a compactly supported smoothly decreasing C^2 function satisfying $\alpha(0) = 1$ and $\alpha(1) = 0$, $d(\mathbf{p})$ is the (anisotropic) distance from $\mathbf{p}$ to the skeleton, and r_i is the radius of influence of the primitive. These determine the distance-dependent weighting of the tailored tensor $\mathbf{T}_i$, which itself can depend on the field location $\mathbf{p}$ and direction $\mathbf{u}(\mathbf{p})$ pointing towards the closest part of the primitive. This approach is slightly different from the one introduced by [ZHT07] where primitives have an infinite support domain.

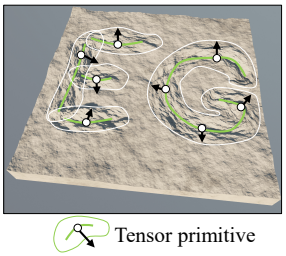

Figure 7: *A tensor-based brush orienting slope in the direction of the stroke.*

For a simple point primitive centered at $\mathbf{c}_i$ with radius r_i this would mean a disc-like region of support, weighted maximally at $\mathbf{p} = \mathbf{c}_i$ and tapering to zero at $d(\mathbf{p}) \geq r_i$, with $\mathbf{u}(\mathbf{p})$ pointing towards $\mathbf{c}_i$.

We have incorporated two variants of skeleton tensors into our toolset. The first (see Figure 5) applies a constant tensor within the area of effect of the primitive, effectively ignoring the parameters of $\mathbf{T}_i$. The second (see Figure 7) is a drawing tool that uses $\mathbf{u}(\mathbf{p})$ to align the tensor to the direction of the user's stroke.

Composite procedural primitives can be constructed by combining elevation and gradient constraints with tensor-based transformations. We illustrate this in Figure 8 with a mountain range brush that creates the main structure of a mountain (left) and mesa (right) with gradient and elevation constraints, respectively, and adds downslope-oriented noise using tensors, thus imitating erosion features.

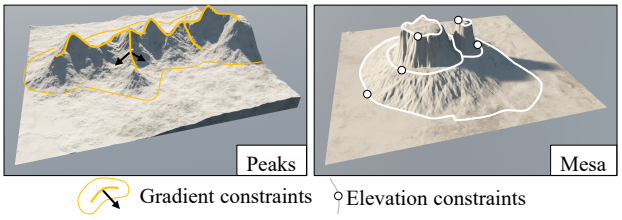

Figure 8: *Composite brush primitives used to generate a mountain range (left) and a mesa (right).*

4.4. Backward compatible tools

Since the gradient and elevation domains exist as duals with an efficient mechanism for mapping between them, it is possible to leverage previous state-of-the-art techniques in heightfield modelling. Notably, this can be achieved while preserving elevation

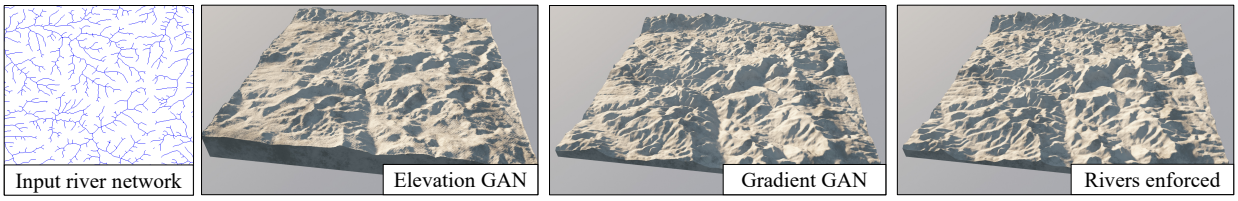

Figure 9: *A comparison between cGAN terrain synthesis operating on elevations and gradients. The gradient version shows sharper landforms, fewer grid artefacts, and enables the enforcement of river networks, guaranteeing a consistent river flow.*

constraints, so that tools taken from prior art fit seamlessly within our authoring framework. There are two strategies for achieving such integrations: either switch temporarily to the elevation domain to perform the tool operation, or transplant the candidate tool entirely to the gradient domain.

Firstly, as an example of the elevation domain strategy, we implemented an hydraulic-erosion brush (see Figure 14) adapted from geomorphological simulation. In doing so, we make the choice to retain previous Dirichlet constraints, thereby enforcing user-designated elevations even while adding eroded features. While not strictly physically correct, this strategy provides designers with greater agency in their control over the final generated terrain.

Secondly, as an example of the gradient domain strategy, we adapted noise-layering brushes to work on gradients (see Figure 14 left). Noise functions are central in terrain authoring and are commonly used to represent high-frequency features not associated with any particular structuring phenomena. With our model, this simply requires adding the gradient of the desired noise n to the gradient field, such that: $\widetilde{\mathbf{g}} = \mathbf{g} + \nabla n$. Figure 10 and 11 show examples of reconstructed terrains that do not contain any noise.

5. Global gradient synthesis

In this section, we explore the utility of a series of global gradient-based synthesis techniques. Although not directly focused on interactive editing, these techniques provide significant improvements in digital terrain generation and amplification compared to their elevation-oriented counterparts.

5.1. Fitting to constraints

The constraint-matching capabilities of our system in both the elevation and gradient domain make the task of fitting terrains to sparse curve descriptions an obvious application. Two particular use-cases are of interest: fitting to ridge and river networks, and contours.

Figure 11 shows an example of a ridge and river fitting process and compares bare diffusion (bottom left) with a more elaborate anisotropic noise enhancement (bottom right), where noise direction is aligned perpendicular to the ridge lines using tensors and noise amplitude is moderated according to distance from the nearest ridge or river.

While a number of methods for fitting terrains to contours have been proposed [HSS03], the advantage of our framework is a simple and direct correspondence between input contour lines and elevation constraints. Figure 10 shows a typical instance for a canyon reconstruction.

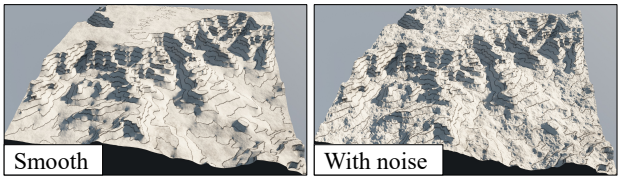

Figure 10: *Fitting to contours is another application of our model, with the possibility of layering additional noise into the gradient map.*

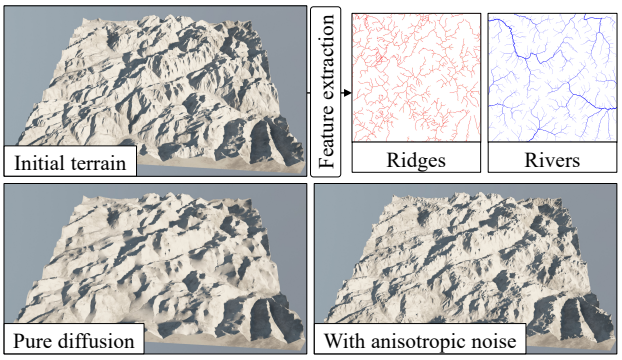

Figure 11: *Fitting to constraints: the initial terrain (top-left) was analyzed to extract ridge and river networks (top-right) and Dirichlet constraints were set accordingly. Reconstructions were performed using a null-Laplacian value (bottom-left) and a more elaborate anisotropic gradient (bottom-right). In both cases the feature networks were well-respected.*

5.2. Example-based cGAN synthesis

While Conditional Generative Adversarial Networks (cGANs) have found success in the example-based synthesis of terrains from sketches [GDG*17], our experiments indicate (see Figure 9) that substantial improvement can be achieved by switching from elevation images to gradient images. Adding the input sketches as constraints emphasizes local relief and leads to even better results. The

overall reduction in cGAN artefacts can be explained, in part, by the elevation invariance of gradient images leading to improved pattern matching. In these experiments, we used the identical set-up of a *pix2pix* cGAN using two datasets: pairs of sketch/elevation maps, and pairs of sketch/gradient maps. We used a total of 135 pairs of size 512×512 cropped to 256×256 resolution to train over 600 epochs. Gradients were encoded in two channels of an RGB image to fit the pipeline.

5.3. Terrain super-resolution

Terrain super-resolution, or amplification, involves increasing resolution by generating plausible details consistent with an atlas of reference exemplars. There has been recent interest in the topic [GDGP16, ACA18, ZLB*19], but it remains far from a solved problem.

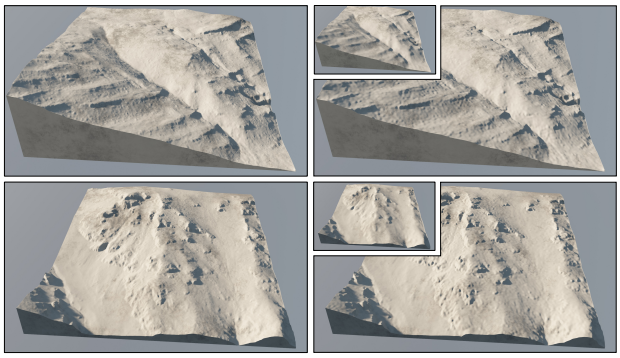

Figure 12: *Super-resolution of an input terrain using a* $4\times$ *amplification factor (increasing precision from* 4 *to* 1 *meter). Low resolution maps are shown as insets.*

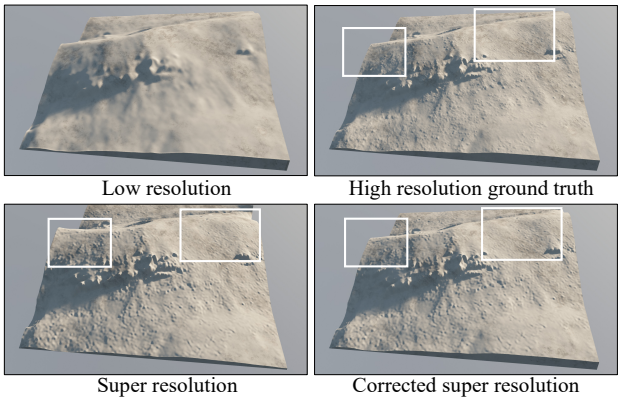

Figure 13: *Adjusting gradients after applying the super-resolution process to the initial low-resolution model removes visual bias.*

In the context of images, state-of-the-art super-resolution (SR) methods using deep convolutional networks increase resolution by up to a factor of 4 with impressive results [LTH*17]. Early attempts to configure this approach for terrains using elevation images were an unmitigated failure. We have found that a markedly more successful approach (as shown in Figure 12) is to apply deep learning to gradient images instead. One concern is that there is no guarantee that the output will respect the initial gradient. So, we remove the introduced bias by adjusting the level of each 4×4 patch of the super-resolution output so that its average matches the initial low-resolution input.

This bias is equal to $\mathbf{g}_\mathrm{l} - \overline{\mathbf{g}_\mathrm{h}}$ where $\mathbf{g}_\mathrm{h}$ and $\mathbf{g}_\mathrm{l}$ are the high- and low-resolution gradients, respectively and $\overline{\mathbf{g}_\mathrm{h}}$ is the mean gradient of the patch. This enables a better correspondence with the initial low-resolution terrain, as can be seen in Figure 13. Training required a dataset of 440 pairs of low resolution 128×128 gradient images and high resolution 512×512 gradient images, a random crop of size 24, a VGG54 perceptual mode, and a total of $200k$ iterations.

6. Implementation and results

We implemented our algorithms in C++ and OpenGL using the compute shader, which enables real-time editing of terrains. Performance was measured on an Intel i7® CPU clocked at 4GHz and equipped with 32GB RAM and an NVidia GeForce® GTX 970 GPU with 4GB RAM. Reconstructed terrains were directly streamed to E-On Vue® to produce the photorealistic renderings shown throughout the paper. The source code for our multigrid solver is available at `https://github.com/eric-guerin/gradient-terrains`.

6.1. Performance

Our half V-cycle multigrid solver reconstructs large terrains by first processing the entire terrain at the coarsest resolution, and then progressively zooming and refining to achieve the target resolution and cropping window. The core iterative solution step is computationally intensive and so relies on a compute shader. For the number of iterations, we settled empirically on $50 + 20\ell$ per level, where ℓ represents the level counting upwards from $\ell = 1$ at the coarsest 9×9 resolution. Although our implementation supports any grid size, for best performance it should be set equal to $2^n + 1$ so that only odd grid sizes occur in the multigrid process. In terms of memory usage our solution requires five buffers for each multigrid level — two alternating buffers for the solution, one for the Laplacian values, one for elevation constraints, and one for representation of the domain Ω.

Size	Time (ms)	Memory (MB)
513×513	27	7
1025×1025	68	28
2049×2049	275	112
4097×4097	1180	448
8193×8193	7670	1790

Table 1: *Statistics for reconstructing the elevation* $\widetilde{h}$ *from the gradient* $\widetilde{\mathbf{g}}$ *for different grid sizes: average reconstruction time calculated from* 10 *reconstruction runs, and GPU memory.*

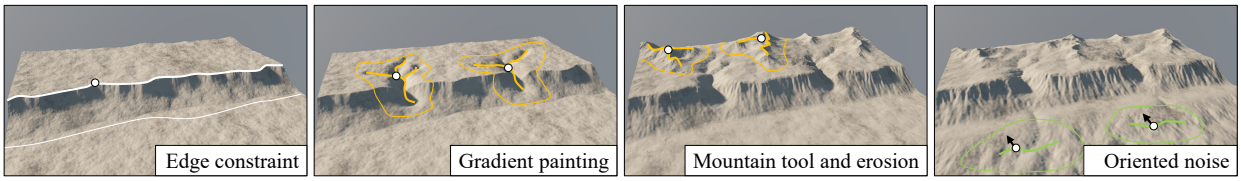

Figure 14: *This cliff was authored with about 50 mouse actions in a timespan of 3 minutes.*

Table 1 reports the reconstruction times and memory usage for various grid sizes. Unsurprisingly, both performance measures are roughly linear with grid resolution. The low memory footprint allows us to process maps with a resolution of up to 8193×8193, given the limits of modern GPU memory. However, computational performance is really the dominant concern. Our implementation could have been further tuned. For instance, certain tasks, such as the prolongation operator, are performed on the CPU, which incurs delays due to CPU-GPU memory transfer. Nevertheless, response is real-time up to a resolution of 1025×1025 at 14.7Hz (68 ms), and thereafter interactive up to resolutions of 2049×2049 at 3.6Hz (275 ms).

6.2. Control

Various aspects of user control are illustrated in Figure 14 with snapshots from a typical authoring session. This demonstrates how a user can pick and choose from among the available tools to best achieve their design intent.

Since strict elevation constraints can be placed anywhere in the domain, one particularly useful tool is a *freeze brush* that prevents further edits in user-painted sections of the terrain (such as the cliff top and base in Figure 14). In general, inexperienced users favour high-level tools because of their simplicity. For example, the complex mountain tool can be utilised to generate a convincing mountain range with a single stroke. This enables the rapid sketching of complex terrains, but provides less precision than the low-level tools favoured by experienced artists.

In either case, complex scenes can be modelled in a matter of minutes. The editing session in Figure 14 had a duration of 3 minutes (see also the accompanying video for other interactive examples), while the scenes in Figure 1 and 18 were each completed in less then 10 minutes by an experienced user.

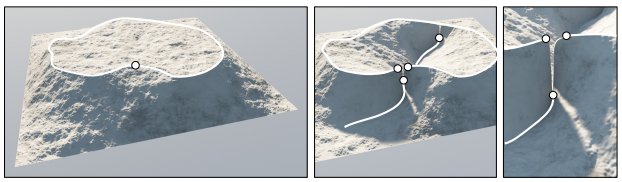

Figure 15: *An instance of conflicting straddling constraints: starting from a prescribed elevation constraint defining a plateau, the user naively attempts to create a canyon by adding a transecting constraint, which produces an unintended narrow pass.*

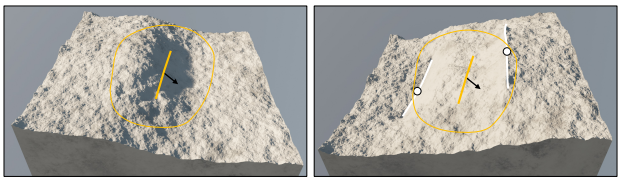

Figure 16: *Whenever user-prescribed hard and soft constraints are discordant, hard ones always prevail.*

One limitation in terms of control is that intersecting or grazing constraints may cause conflicts in the reconstruction process that yield unexpected outcomes, particularly if users are not aware of the specifics of the underlying mechanism. Figure 15 shows a case where crossing constraints generate near vertical edges. Another typical case occurs when soft constraints conflict with hard ones. In these situations, hard constraints will always prevail, which can sometimes make it difficult for users to anticipate the shape of resulting landforms (see Figure 16).

6.3. Validation

To demonstrate the effectiveness of our gradient-based model and its authoring potential we performed two sets of user tests. In the first, we asked two 3D artists specialized in terrain authoring to test our modelling tools, comment on their effectiveness, and compare them to standard terrain editing systems. Both artists author terrains using software such as World machine and Houdini, which combine interactive editing and node-based procedural design. This approach allowed us to iteratively improve our toolset by taking expert feedback into account. Apart from some interface-specific comments, the artists found gradient-based authoring intuitive and efficient.

An interesting aspect of their feedback was a request that dimensional information on the terrain be accessible at all times, specifically measures of terrain elevation (in meters), and average and steepest slope (in degrees). This demonstrates the value of dimensional elevation data as compared to greyscale images.

The second set of qualitative user tests was conducted with 7 non-artists. They were tasked with producing three scenes based on short textual descriptions, each within a 10 minute time limit. The scenes, assigned in random order so as to minimize any learning effects, were: an open volcano with a breach in the caldera, a Y-shaped canyon, and a mountain range with two prominent peaks.

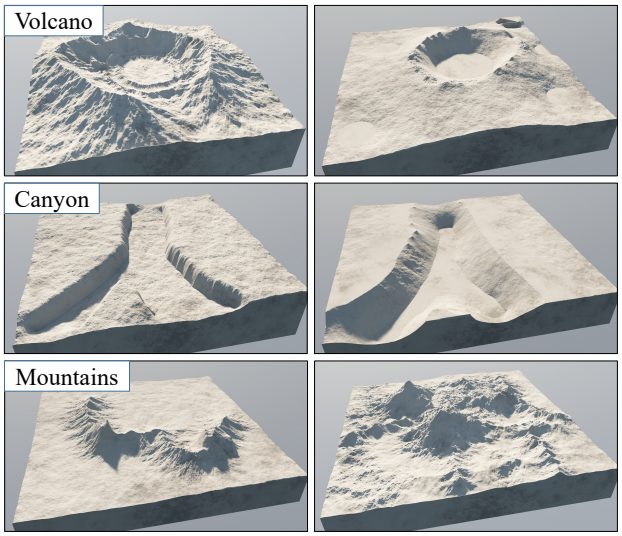

Figure 17: *Examples of terrains authored by non-artist users.*

Demonstration videos were embedded with the application to explain the controls, and an expert was on hand to provide answers and help where required. For all experiments, we collected both the participants' final terrains (see Figure 17) and qualitative feedback regarding the strengths and weaknesses of the different tools and their general impressions. The average time per session was 6 minutes, with an average of 38 user actions. The most effectively used tool was the placement of hard constraints (used 53% of the time on average), then the mountain tool (10%), positioning elevation and gradient constraints (10%), adding gradient (7%), moving elevation constraints (6%) and freezing results (6%). All the other tools were seldom used ($\leq 2\%$).

In terms of feedback, a recurrent positive was the ease of use of the mountain range tool. Conversely, users found elevation constraints challenging. In general, our non-expert participants had little difficulty mastering each tool independently, but struggled to employ them in combination, possibly due to the sheer number of tools and time pressure. This points to a benefit in improving the structure of the toolset as part of future work. Finally, in their general comments six of the seven participants reported that the system was both easy to learn and intuitive.

6.4. Comparison

It is worth undertaking a more detailed comparison with competing modelling approaches, particularly the feature curve model and example-based methods.

Feature curves. The diffusion-based feature curve model [HGA*10] is the closest to our approach in its extensive use of gradients. However, with feature curves, gradients have a dependence on curves at one pixel size, which can give rise to sharp-edged landforms. The authors ameliorate these artefacts with a gradient weighting scheme and by blurring with controlled noise as a post process. Instead, we rely on Dirichlet conditions to enforce elevations and the Poisson equation, which uses Laplacian values drawn from the gradients. While similar in computational performance, based on our re-implementation, feature curves require about twice as much memory, making them less suited to more expansive terrains.

A key aspect of feature curves is their compact vector-based encoding. Although beyond the scope of this work, such a formulation is also possible with our method. The main difference lies in the way constraints are expressed in the solver. Feature curves require that the gradient constraint be linked to a source (a crest or ravine) at a predefined elevation. Our approach encompasses and extends this modelling scope as gradients can be signified regardless of source. Furthermore, noise layering, while no longer needed to disguise artefacts remains useful, and can be added to gradients (modified by tensors to produce anisotropic features) or directly to elevations, with no requirement for post-processing.

Figure 18 presents a side-by-side comparison of a volcanic island modelled originally by Hnaidi *et al.* [HGA*10] and now reproduced with our approach. Ours took an experienced user 5 minutes to model by laying constraints at the rim of the volcano, along an island contour, and on the seabed. In constrast, Hnaidi *et al.* report using 48 constraint lines carefully specified over the course of 45 minutes to define the feature elevations. Note that the use case was not the same: scene authoring for Hnaidi *et al. vs* scene reproduction in our case.

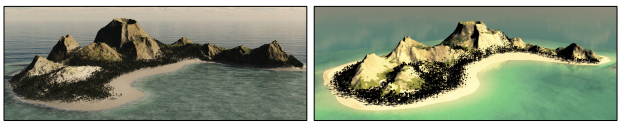

Figure 18: *Gradient terrain authoring (right) can match the visual characteristics of diffusion-based feature curves (left), but with fewer strokes and with markedly less authoring time.*

Synthesis from examples. Another useful point of comparison is with example-based schemes [GMM15, GDGP16, GDG*17]. At their most powerful these ingest scanned exemplar data into derivation structures, such as neighbourhood-matching tables [GMM15] or convolutional neural networks [GDG*17], that then allow plausible detail to be synthesised while simultaneously meeting authored constraints. However, this philosophy of guiding rather than dictating detail represents both a blessing and a curse, in that users sacrifice fine control for rapid synthesis. In contrast, our authoring system requires users to decide on the source and placement of detail, be it cutting, warping, and pasting from an exemplar or painting gradient noise over a region, but this affords greater structural and semantic control. Furthermore, example-based methods lack flexibility in the sense that their authoring toolsets are curated and domain specific. For example, hydraulic erosion in the elevation domain could only be applied as a post-process in Gain et al.'s method and it would be impossible to subsequently recover authored point, curve and brush constraints and resume editing, unlike in our framework.

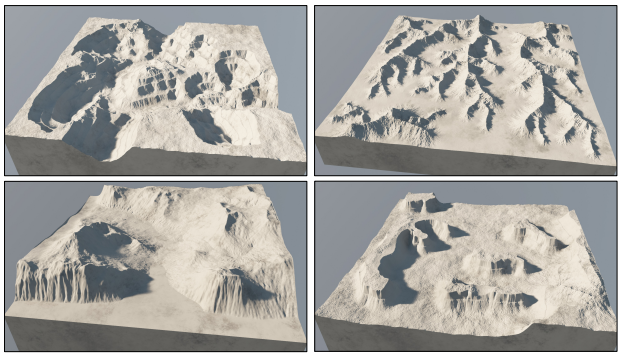

Figure 19: *Our framework is expressive and allows a variety of different landforms to be authored.*

7. Conclusion

The gradient-domain forms a strong basis for authoring terrains. Not only does it accommodate the seamless blending of terrain fragments, but it also enables a broad spectrum of modelling tools with a basis in the disciplines of painting, sketching, and collage. Furthermore, we provide an efficient bijective mapping between the elevation and gradient domains, which makes it possible to incorporate a variety of existing procedural and simulation-based modelling tools.

Adopting a gradient-domain formalism also improves the results for a range of global terrain synthesis tasks. These include reconstruction from sparse descriptors, such as river and ridge networks, and isolines; super-resolution upsampling of a coarse terrain with plausible detail, achieving up to a fourfold increase in resolution, and a reduction in artefacts when training cGANs for example-based terrain generation.

Acknowledgments

This work was part of the project AMPLI ANR-20-CE23-0001, supported by Agence Nationale de la Recherche Française, and also by the National Research Foundation of South Africa (Grant Number: 129257). We thank Matthieu Beaud for the early developments of the editing interface. We also credit Benoît Martinez and Mathieu Gasperin from Ubisoft for their useful feedback on the user interface.

Appendix A: Classification of methods

Table 2 presents a comparison of terrain authoring approaches. The different forms of control are: area-based ($\mathcal{A}$), such as painting or lasso tools that adjust aspects of style and roughness, vector-based ($\mathcal{V}$), which represent sketching tools that generate point or curve constraints, and copy-paste operations ($\mathcal{C}$), where regions are marked out in a source and transferred to a destination.

For vector-based controls specifically, there is further delineation into: planar ($\mathcal{P}$), involving drawing top-down 2D curves, elevation ($\mathcal{E}$), representing $2\frac{1}{2}$ D curves, and gradient ($\mathcal{G}$), where slope profiles can be specified to either side of a curve or around a point.

The initial starting point for authoring is either a blank slate ($\mathcal{B}$), typically a flat or relatively featureless plane, or an existing terrain ($\mathcal{T}$), perhaps imported from scanned real-world sources.

References

[ACA18] ARGUDO O., CHICA A., ANDUJAR C.: Terrain Super-resolution through Aerial Imagery and Fully Convolutional Networks. *Computer Graphics Forum 37*, 2 (2018), 101–110. 7

[BCA*14] BRADBURY G., CHOI I., AMATI C., MITCHELL K., WEYRICH T.: Frequency-based creation and editing of virtual terrain. In *Proceedings of European Conference on Visual Media Production* (London, UK, 2014). 1, 2, 11

[BD96] BANDY V. A., DENDY J. E.: *Black Box Multigrid for Convection-Diffusion Equations on Advanced Computers*. PhD thesis, USA, 1996. 3

[CCB*18] CORDONNIER G., CANI M.-P., BENES B., BRAUN J., GALIN E.: Sculpting mountains: Interactive terrain modeling based on subsurface geology. *IEEE Transactions on Visualization and Computer Graphics 24*, 5 (2018), 1756–1769. 2, 11

[CGG*17] CORDONNIER G., GALIN E., GAIN J., BENES B., GUÉRIN E., PEYTAVIE A., CANI M.-P.: Authoring Landscapes by Combining Ecosystem and Terrain Erosion Simulation. *ACM Transactions on Graphics 36*, 4 (2017), 134:1–134:12. 2

[DBP*15] DIAMANTI O., BARNES C., PARIS S., SHECHTMAN E., SORKINE-HORNUNG O.: Synthesis of complex image appearance from limited exemplars. *ACM Transactions on Graphics 34*, 2 (2015), 22:1–22:14. 2

[dCB09] DE CARPENTIER G. J. P., BIDARRA R.: Interactive GPU-based procedural heightfield brushes. In *Proceedings of the International Conference on Foundations of Digital Games* (Orlando, USA, 2009), ACM, pp. 55–62. 2, 11

[Den82] DENDY J.: Black box multigrid. *Journal of Computational Physics 48*, 3 (1982), 366–386. 3

[DSWH14] DESSEIN A., SMITH W. A. P., WILSON R. C., HANCOCK E. R.: Seamless texture stitching on a 3D mesh by Poisson blending in patches. In *ICIP - 21th IEEE International Conference on Image Processing* (Paris, France, 2014). 2

[GDG*17] GUÉRIN E., DIGNE J., GALIN E., PEYTAVIE A., WOLF C., BENES B., MARTINEZ B.: Interactive example-based terrain authoring with conditional generative adversarial networks. *ACM Transactions on Graphics (proceedings of Siggraph Asia 2017) 36*, 6 (2017). 2, 6, 9, 11

[GDGP16] GUÉRIN E., DIGNE J., GALIN E., PEYTAVIE A.: Sparse representation of terrains for procedural modeling. *Computer Graphics Forum (proceedings of Eurographics 2016) 35*, 2 (2016), 177–187. 7, 9

[GGP*19] GALIN E., GUÉRIN E., PEYTAVIE A., CORDONNIER G., CANI M.-P., BENES B., GAIN J.: A Review of Digital Terrain Modeling. *Computer Graphics Forum (proceedings of Eurographics 2019) 38*, 2 (2019), 553–577. 2

[GMM15] GAIN J., MERRY B., MARAIS P.: Parallel, realistic and controllable terrain synthesis. *Computer Graphics Forum 34*, 2 (2015), 105–116. 2, 9, 11

[GMS09] GAIN J. E., MARAIS P., STRASSER W.: Terrain sketching. In *Proceedings of the Symposium on Interactive 3D Graphics and Games* (Boston, USA, 2009), ACM, pp. 31–38. 2, 11

[HGA*10] HNAIDI H., GUÉRIN É., AKKOUCHE S., PEYTAVIE A., GALIN É.: Feature based terrain generation using diffusion equation. *Computer Graphics Forum 29*, 7 (2010), 2179–2186. 2, 3, 9, 11

[HSS03] HORMANN K., SPINELLO S., SCHRÖDER P.: C1-Continuous Terrain Reconstruction from Sparse Contours. In *Proceedings of the Vision, Modeling, and Visualization* (2003). 6

[KRS15] KETABCHI K., RUNIONS A., SAMAVATI F. F.: 3D Maquetter: Sketch-based 3D content modeling for digital Earth. In *2015 International Conference on Cyberworlds* (2015), pp. 98–106. 2, 11

Category	Paper	Method	Control $\mathcal{A}\,\mathcal{V}\,\mathcal{C}$	Constraints $\mathcal{P}\,\mathcal{E}\,\mathcal{G}$	Init. $\mathcal{B}\,\mathcal{T}$	Performance
Procedural	[GMS09]	deformation	● ● ○	● ● ○	● ○	interactive
	[dCB09]	noise brushes	● ○ ○	○ ○ ○	● ○	real-time
	[RME09]	shortest-path	○ ● ○	● ● ○	● ○	interactive
	[HGA*10]	diffusion	○ ● ○	● ● ●	● ○	interactive
	[BCA*14]	noise blending	○ ● ●	● ● ○	● ●	interactive
Simulation	[CCB*18]	uplift sculpting	● ○ ○	○ ○ ○	● ○	interactive
	[VBHŠ11]	erosion brushes	● ○ ○	○ ○ ○	○ ●	interactive
Example-based	[TEC*14]	deformation	○ ● ○	● ● ○	○ ●	interactive
	[KRS15]	deformation	○ ● ○	● ● ○	○ ●	interactive
	[GDG*17]	machine learning	○ ● ○	● ● ○	● ○	interactive
	[GMM15]	texture synthesis	● ● ●	● ● ●	● ●	real-time
Unified	Our method	gradient constraints	● ● ●	● ● ●	● ●	real-time

Table 2: *A comparison of terrain authoring techniques.*

[LTH*17] LEDIG C., THEIS L., HUSZÁR F., CABALLERO J., CUNNINGHAM A., ACOSTA A., AITKEN A., TEJANI A., TOTZ J., WANG Z., SHI W.: Photo-realistic single image super-resolution using a generative adversarial network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 105–114. 7

[NWD05] NEIDHOLD B., WACKER M., DEUSSEN O.: Interactive physically based fluid and erosion simulation. In *Proceedings of the Eurographics Workshop on Natural Phenomena* (Dublin, Ireland, 2005), Eurographics Association, pp. 25–32. 2

[PGB03] PÉREZ P., GANGNET M., BLAKE A.: Poisson image editing. *ACM Transactions on Graphics 22*, 3 (2003), 313–318. 2, 4

[PGMG09] PEYTAVIE A., GALIN É., MÉRILLOU S., GROSJEAN J.: Arches: a framework for modeling complex terrains. *Computer Graphics Forum 28*, 2 (2009), 457–467. 2

[PGP*19] PARIS A., GALIN E., PEYTAVIE A., GUÉRIN E., GAIN J.: Terrain amplification with implicit 3D features. *ACM Transactions on Graphics 38*, 5 (2019), 147:1–15. 2

[RME09] RUSNELL B., MOULD D., ERAMIAN M. G.: Feature-rich distance-based terrain synthesis. *The Visual Computer 25*, 5-7 (2009), 573–579. 2, 11

[SCOL*04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *Proceedings of the EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing* (2004), ACM Press, pp. 179–188. 3

[Sha08] SHAPIRA Y.: *Matrix-Based Multigrid: Theory and Applications*, 2nd ed. Springer Publishing Company, Incorporated, 2008. 3

[TEC*14] TASSE F. P., EMILIEN A., CANI M.-P., HAHMANN S., DODGSON N.: Feature-based terrain editing from complex sketches. *Computers & Graphics 45* (2014), 101–115. 2, 11

[TGM12] TASSE F. P., GAIN J. E., MARAIS P.: Enhanced texture-based terrain synthesis on graphics hardware. *Computer Graphics Forum 31*, 6 (2012), 1959–1972. 1

[VBHŠ11] VANEK J., BENES B., HEROUT A., ŠŤAVA O.: Large-scale physics-based terrain editing using adaptive tiles on the GPU. *Computer Graphics and Applications 31*, 6 (2011), 35–44. 2, 11

[YZX*04] YU Y., ZHOU K., XU D., SHI X., BAO H., GUO B., SHUM H.-Y.: Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics 23*, 3 (2004), 644–651. 3

[ZHT07] ZHANG E., HAYS J., TURK G.: Interactive tensor field design and visualization on surfaces. *IEEE Transactions on Visualization and Computer Graphics 13*, 1 (Jan 2007), 94–107. 5

[ZLB*19] ZHAO Y., LIU H., BOROVIKOV I., BEIRAMI A., SANJABI M., ZAMAN K.: Multi-theme generative adversarial terrain amplification. *ACM Transactions on Graphics 38*, 6 (2019). 7

[ZSTR07] ZHOU H., SUN J., TURK G., REHG J. M.: Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer Graphics 13*, 4 (2007), 834–848. 2