Large-scale terrain authoring through interactive erosion simulation

Hugo Schott, Axel Paris, Lucie Fournier, Eric Guérin, Eric Galin

Large-scale terrain authoring through interactive erosion simulation

HUGO SCHOTT, Univ Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France
AXEL PARIS, Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France
LUCIE FOURNIER, Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France
ERIC GUÉRIN, Univ Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France
ERIC GALIN, Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France

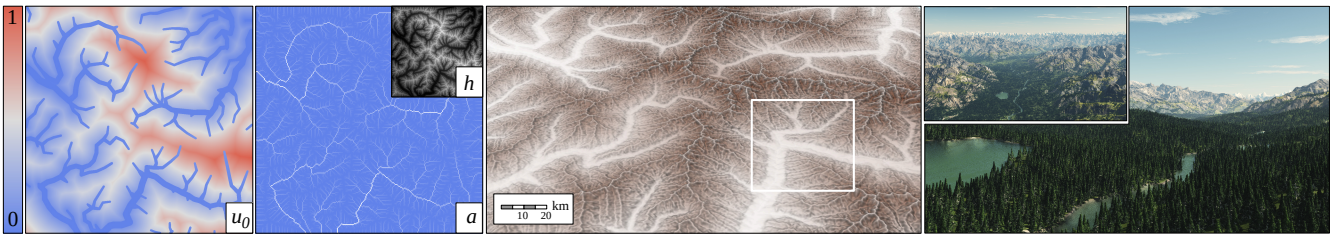

Fig. 1. Given an input uplift field, we automatically generate the large-scale terrain elevation by simulating stream power erosion using a parallel drainage area algorithm inlined in the simulation. The user may define the uplift field, providing ridge and river networks, or using inverse procedural modeling by computing the uplift from an input digital elevation model.

Large-scale terrains are essential in the definition of virtual worlds. Given the diversity of landforms and the geomorphological complexity, there is a need for authoring techniques offering hydrological consistency without sacrificing user control. In this paper, we bridge the gap between large-scale erosion simulation and authoring into an efficient framework. We set aside modeling in the elevation domain in favour of the uplift domain, and compute emerging reliefs by simulating the stream power erosion. Our simulation relies on a fast yet accurate approximation of drainage area and flow routing to compute the erosion interactively, which allows for incremental authoring. Our model provides landscape artists with tools for shaping mountain ranges and valleys, such as copy-and-paste operations; warping for imitating folds and faults; point and curve elevation constraints to precisely sculpt ridges or carve river networks. It also lends itself to inverse procedural modeling by reconstructing the uplift from an input digital elevation model and allows hydrologically consistent blending between terrain patches.

CCS Concepts: • **Computing methodologies** → **Shape modeling**.

Additional Key Words and Phrases: Erosion Simulation, Landscapes

ACM Reference Format:
Hugo Schott, Axel Paris, Lucie Fournier, Eric Guérin, and Eric Galin. 2010. Large-scale terrain authoring through interactive erosion simulation. *ACM Trans. Graph.* 9 (March 2023), 15 pages.

Authors' addresses: Hugo Schott, Univ Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France, hugo.schott@liris.cnrs.fr; Axel Paris, Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France, axel.paris@liris.cnrs.fr; Lucie Fournier, Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France, lucie.fournier@liris.cnrs.fr; Eric Guérin, Univ Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France, eric.guerin@liris.cnrs.fr; Eric Galin, Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France, eric.galin@liris.cnrs.fr.

1 INTRODUCTION

Modeling realistic and controllable large-scale terrain models is essential for creating virtual worlds. The challenge stems not only from the complexity of landforms, the variety of details forming patterns at different scales, the need for geomorphological and hydrological realism, but also from the need to control the shape and location of landforms to follow the designer's intent.

Existing terrain generation approaches encompass a variety of methods. Procedural methods [Ebert et al. 1998] mostly rely on sum of scaled noises to algorithmically reproduce the self-similarity across scales. However, those methods do not take erosion into account and require careful editing from specialists to obtain realistic erosion landmarks. Example-based methods [Zhou et al. 2007], including learning-based approaches [Guérin et al. 2017], tackle realism by using terrain patches extracted from real-world digital elevation models, but do not manage to generate hydrologically consistent large-scale models without significantly affecting terrain landforms. Finally, surface erosion simulations [Musgrave et al. 1989] typically enhance procedurally generated terrain with sedimentary valleys and small-scale erosion landmarks such as gorges and ravines, and generate visually convincing effects as long as the initial input terrain is sufficiently realistic regarding the large-scale landmarks. Such erosion algorithms apply on a limited set of surface phenomena, that are strongly linked to restricted time and spatial scale ranges, which intrinsically prevents their usage for generating large-scale terrains.

A key observation is that landforms are emerging structures resulting from the competition between uplift and erosion. Their diversity is the consequence of complex natural phenomena, including tectonics, stratification, aeolian and hydraulic erosion, which are involved at different spatial and time scales. Modeling those natural phenomena is complex and comes at the price of computationally intensive, involved, and hard to control simulations [Cordonnier

et al. 2016, 2018]. Another crucial observation is that artists require interactive editing processes when authoring terrains, and consequently favor techniques that provide user control. Even though a vast variety of methods exist for controlling landforms – in general through the use of curves and point controllers [Gain et al. 2009; Guerin et al. 2022; Hnaidi et al. 2010], little attention has been dedicated to combining geomorphologically coherent large-scale erosion simulations with interactive editing controls.

We address these limitations by proposing a novel approach for authoring and simulating large-scale terrains. In this work, we aim at bridging the gap between erosion simulation providing realism, and interactive editing giving control. The challenge consists in supplying the user with erosion-based sculpting tools with an interactive feedback of the stream power erosion to guarantee realistic mountain ranges with hydrologically consistent drainage networks [Cordonnier et al. 2018]. Our framework tries to combine the best of both worlds by relying on the computation of the uplift to generate predictable mountain ranges, and on the elevation space by introducing control and feature curves that constrain landscape erosion.

To achieve these goals, we introduce a hierarchical uplift construction tree to control the large scale structures. Taking inspiration from Argudo *et al.* [2019], we propose several novel algorithms based on orometry, *i.e.* the distribution and connectivity between peaks and saddles points. We propose authoring tools compatible with tectonic erosion bringing sketching, copy-and-paste sequences, inverse procedural modeling into large-scale erosion simulations, thus providing ways for the designer to define or impose the location, the orientation and the distribution of specific mountain ranges while preserving the global hydrological consistency of the final terrain (Figure 1). More precisely, our contributions are as follows: 1) We propose a model allowing to author the terrain in the elevation domain or in the uplift domain. 2) We propose an iterative approximation of the drainage area, allowing for a parallel implementation and embedding in the stream power erosion simulation, which produces a coherent drainage computation. 3) We introduce the Uplift Tree, a hierarchical vector-based construction tree taking its inspiration from the implicit models construction trees, but with the particularity that it can represent complex uplift scalar fields with asymmetric profiles or faults.

Images shown throughout the paper as well as the accompanying video depict large-scale terrains ranging from 100×100 km^2 to 1000×1000 km^2, and demonstrate that our method combines interactive authoring and geomorphologically-based erosion simulations in a unified framework.

2 RELATED WORK

A vast variety of terrain generation methods exist and can be sorted into one of the following categories: procedural generation, texture synthesis and physical erosion simulation, as classified in a recent overview [Galin et al. 2019]. Given the identified goals of control, performance, and hydrological realism, we focus this review on authoring frameworks that evidence interactive response rates.

Procedural terrain modeling relies on noise synthesis, often combined with river network carving [Ebert et al. 1998; Kelley et al. 1988; Musgrave et al. 1989], to algorithmically reproduce the self-similarity across scales. They are in general computationally efficient and lend themselves for a parallel implementation on graphics hardware. Control typically takes the form of applying noise with circular brushes [de Carpentier and Bidarra 2009] or matching curve and point constraints using warping [Gain et al. 2009], shortest path traversal [Rusnell et al. 2009], multi-frequency blending [Bradbury et al. 2014], or diffusion [Hnaidi et al. 2010]. Recently, Guérin *et al.* [2022] advocated gradient domain editing for terrain modeling, demonstrating the effectiveness of gradient domain for seamless blending of terrain patches, copy-and-paste operations and generation from control feature points and curves. Unfortunately, those approaches do not take erosion into account and require careful editing from specialists to obtain erosion landmarks.

Example-based methods tackle realism by combining patches extracted from real-world digital elevation models. Here, control is achieved by structure-sensitive warping to match sketched silhouettes [Tasse et al. 2014], or the use of Conditional Generative Adversarial Networks to learn a correspondence between terrains and the sketch maps corollaries [Guérin et al. 2017]. Parallel texture-based terrain synthesis [Gain et al. 2015] 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 not sufficient for generating globally consistent landscapes. Recently, Scott *et al.* [2021] advocated the use of a breaching algorithm interlaced in multi-resolution example-based terrain synthesis to improve hydrological consistency. Although hydrologically consistent, the breaching process still produces visually artificial canyons.

Erosion simulation methods can be broadly classified in two categories: surface erosion algorithms and tectonic-based simulations. Surface erosion algorithms enhance procedurally generated terrain with sedimentary valleys and small-scale erosion landmarks such as gorges and ravines. Musgrave *et al.* [1989] first presented a complete hydraulic model simulating material detachment, transport and deposition between heightfield cells. A geological representation for modeling the strata of the bedrock was introduced by Roudier *et al.* [1993] that considers the characteristics of the different materials during the erosion process. These early approaches were improved in several ways, by computing the acceleration or deceleration of the fluid to erode the bedrock or deposit sediments [Neidhold et al. 2005], combining a shallow water simulation with hydraulic erosion [Benes 2007], or using Smoothed Particle Hydrodynamics [Krištof et al. 2009]. As fluid simulations are computationally intensive, implementations on graphics hardware were proposed to speed-up computations [Mei et al. 2007; Šťava et al. 2008; Vanek et al. 2011].

Surface erosion methods perform the simulations at a limited temporal and spatial scale and then implicitly scale up the results to approximate large scale terrains. Although intrinsically flawed (as the equations of the corresponding phenomena are not linear), those methods generate visually convincing small-scale erosion effects as long as the initial input terrain is sufficiently realistic and supplies large-scale landmarks. Specific landforms can be authored exclusively through the initial terrain, and are often damaged by the simulation.

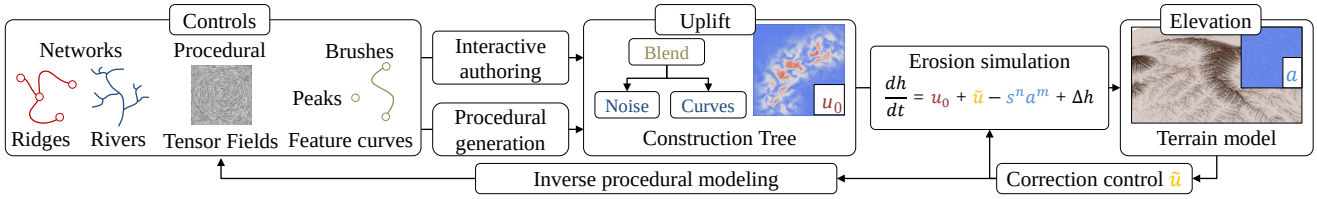

Fig. 2. Our method computes the terrain elevation h from the uplift $u = u_0 + \tilde{u}$ using a modified Stream Power law. Control is achieved by prescribing the static term u_0 from a user-defined or procedurally-generated Uplift Tree. Sparse elevations constraints are handled through the dynamic term $\tilde{u}$.

Tectonic simulations in contrast, attempt to reproduce large-scale erosion effects by taking into account the uplift of the bedrock balanced by different types of erosion described in geology. Contrary to previously described surface erosion methods, they produce realistic large scale mountain ranges with dendritic river networks without the need for an initial elevation. Cordonnier *et al.* [2016] introduced the Stream Power erosion law from geomorphology [Whipple and Tucker 1999] to computer graphics. This work was extended [Cordonnier et al. 2018] with a geologically-coherent model taking into account folds and faults and the characteristics of the different bedrock strata from the relative movement of the tectonic plates. Although the simulation can be guided by moving the tectonic plates, controlling the generation of the small-scale features remains difficult. Moreover, large-scale erosion simulations remain computationally intensive and challenging to control, which deters them from production environments.

In contrast, our method bridges the gap between tectonic erosion and interactive authoring and provides a unified framework combining the automatic generation of large scale landforms and user-constrained peaks, ridges, valleys or other landmarks. It combines the best of both worlds, allowing the user to control the shape of landforms with painting, region-based copy-and-paste, and curve and point manipulators while relying on a coherent stream power erosion to generate realistic and hydrologically consistent dendritic mountainous landforms.

3 OVERVIEW

The elevation of the terrain results from an equilibrium between uplift and stream power erosion (also called fluvial erosion). We depart from tectonic-based computation and propose to rely on a procedurally defined construction tree to control the uplift that indirectly defines the shape and elevation of the mountain ranges. Both representations are tightly coupled, which allows to intuitively author elevations by the mean of the uplift, while enforcing altitudes using sparse constraints defined in the elevation domain. Our authoring framework rests on the foundations of a fast erosion simulation based on a incremental and parallel drainage area approximation. Figure 2 presents a synthetic overview of the workflow.

In a typical workflow, the user constructs the static uplift u_0 using brushes (that are combinations of skeletal primitives and operators) and the erosion simulation carves the mountain ranges. At any time, the user can modify u_0, prescribe elevation constraints along crest lines, or directly modify the terrain in the elevation domain. The stream power erosion simulation ensures that the resulting elevation is coherent and flows towards the borders of the map.

Erosion

The shaping of mountainous landscapes by the Stream Power law and hill slope processes (linear diffusion) has been extensively studied in geomorphology [Howard and Kerby 1983; Whipple and Tucker 1999]. The simplified governing equation can be written as:

$$\frac{\partial h}{\partial t} = u - s^n\, a^m + \Delta h \tag{1}$$

The term $u - s^n\, a^m$ states that the rate of change of surface topography h is controlled by the balance between the uplift u and the fluvial erosion, which is a function of the local slope s and the drainage area a that represents the amount of water flowing through a point (see Section 4.1). The constant exponents n and m depend on the type of terrain; geomorphology studies [Theodoratos and Kirchner 2020; Whipple and Tucker 1999] agree that only the ratio m/n is relevant and should be set to $\approx 1/2$. The hillslope term Δh smoothes the terrain by elevating concavities and lowering convexities by using the Laplacian. A crucial assumption of the stream power equation is that the sediments are transported by rivers and not deposited on the simulated domain. More complex models exist that account for sediment deposition [Yuan et al. 2019] but are beyond the scope of this work.

The drainage area a of a point $\mathbf{p}$ represents the amount of water that flows through $\mathbf{p}$ (see Figure 3). Computing a is the most computationally intensive part of the stream power erosion, as it requires processing cells in descending elevation order. Our work relies on an efficient and accurate approximation of the drainage area (Section 4.2).

The uplift defines the speed at which the ground is elevating due to tectonic processes. One originality of this work is to define the uplift as a sum of two terms $u = u_0 + \tilde{u}$, where u_0 denotes the static part and $\tilde{u}$ a dynamic correction term used to prescribe precise elevations (see Section 5). Those terms play a crucial part in the shape of large-scale mountain ranges, and our framework provides different strategies to construct them.

The static term u_0 is defined using a hierarchical construction tree, the *Uplift Tree*, an efficient vector-based model for constructing and interactively authoring uplift maps. Our model takes its inspiration from the construction tree used for terrain modeling [Génevaux et al. 2015], but notably differs as we model the uplift instead of the elevation. We detail several primitives adapted for the representation of asymmetric cross-section profiles observed in geomorphology.

Our model departs from elevation models as it does not require continuity: discontinuities account for an approximation of faults and folds and produce realistic effects after erosion.

Control

User-control is central to our framework and involves a variety of tools for authoring complex large-scale terrains (see Figure 2).

In the uplift domain, control is achieved by directly modifying u_0 using brushes that operate on the underlying Uplift Tree model. We also propose an inverse procedural modeling approach to infer u_0 from an input elevation map by detecting the ridge and river networks (Section 6.1 and 6.2). This reconstruction process yields an uplift u_0 that, under the effect of fluvial erosion, only approximates the input elevation. Finally, we also present an original procedural technique to generate uplift from a procedurally-defined tensor field representing the crust folds (Section 6.3).

In the elevation domain, sparse constraints enforce the elevation at user-given points or along curves (Section 6.4), in the spirit of features curves proposed by Hnaidi *et al.* [2010].

4 INTERACTIVE EROSION SIMULATION

We address the incremental and interactive resolution of the stream power equation. One particularity of this work compared to most simulations in geomorphology is that the uplift $u = u_0 + \tilde{u}$ is not constant and can be modified at any time by the designer. Instead of using a triangulated irregular network combined with an implicit scheme [Cordonnier et al. 2016, 2018], we propose a framework operating on a regular grid and using an explicit integration scheme.

Let h_i, a_i, u_i, s_i denote the elevation, drainage area, uplift and steepest slope of a cell c_i respectively, with i denoting the linear index in the grid. Let Δt denote the time step, at iteration $k+1$, we have:

$$h_i(k+1) = h_i(k) + \Delta t \left(u_i(k) - s_i^n(k)\, a_i^m(k) + \Delta h_i(k)\right) \qquad (2)$$

Instead of relying on the steepest slope for evaluating the drainage area, we use an $\mathcal{L}^p$ metric (Section 4.1) that balances diffusive flow routing and steepest slope based algorithms. We then propose an approximation to the computation of the drainage area (Section 4.2) which can be directly integrated in the explicit integration scheme of equation 2.

4.1 Flow routing

The upstream drainage area a of point $\mathbf{p}$ is the amount of water that flows through $\mathbf{p}$. In the case of constant precipitation, a is proportional to the area of the surface where every downstream route passes through $\mathbf{p}$.

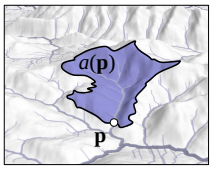

Fig. 3. Drainage area of a point and its highlighted watershed.

The efficient computation of the drainage area, based on flow routing between cells, is crucial in the stream power erosion simulation. Several algorithms [Holmgren. 1994; Seibert and McGlynn. 2007] have been proposed. Let $\mathcal{V}_i$ denote the set of 8-neighboring cells of c_i. Let $\mathcal{A}_i = \{j \in \mathcal{V}_i \,|\, h_j > h_i\}$ and $\mathcal{B}_i = \{j \in \mathcal{V}_i \,|\, h_j < h_i\}$ denote the set of the indexes of the neighboring cells that are respectively above and below c_i. Whenever all neighboring cells are above or of equal height, then the cell is considered as a pit by the drainage area algorithm, and the flow propagation stops. In practice, such pits are progressively removed by the stream power erosion process. Overall, in the discrete case, the drainage area a_i of a cell c_i can be recursively defined as the weighted sum of the drainage area a_j of the cells $\mathcal{A}_i$. In addition, the precipitation on the cell (proportional to the cell area, but here taken constantly as 1 for simplification) is also summed:

$$a_i = 1 + \sum_{j \in \mathcal{A}_i} w_{ij}\, a_j \qquad (3)$$

Let $p \in [1, +\infty[$ denote a chosen exponent corresponding to a $\mathcal{L}^p$ metric. Let d_{ij} denote the Euclidean distance between c_i and c_j, and $s_{ij} = (h_i - h_j)/d_{ij}$ the slope between c_i and c_j. The weighting coefficients w_{ij} define how the water in a cell c_j flows down to its lower neighbours:

$$w_{ij} = \frac{s_{ji}^p}{\sum_{k \in \mathcal{B}_j} s_{jk}^p} \qquad p \in [1, +\infty(\qquad (4)$$

Setting $p = \infty$ simulates that the flow follows the steepest slope, whereas setting $p = 1$ produces a uniform diffusion proportionally to the directional slope of the terrain. Figure 4 illustrates the computation of the drainage area for those exponents. We implemented the general case, while using specific cases for $p = 1$ and $p = \infty$ avoid the computation of a series of computationally-intensive power functions. However, according to Holmgren [1994], the most accurate values are obtained with $4 \leq p \leq 6$. Throughout the remainder of the paper, the exponent p is set to 4 unless stated otherwise.

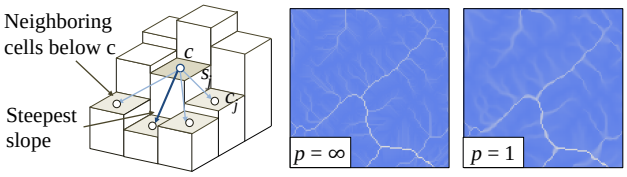

Fig. 4. Distribution of flow among neighboring cells, $p = \infty$ corresponds to routing the flow along the steepest slope (bold arrow), whereas $p = 1$ simulates a diffusion proportionally to the terrain slope, which leads to a smoother drainage area map.

4.2 Parallel approximation of the drainage area

In the general case, the computation of the drainage area proceeds as follows. Cells are first sorted according to their elevation and assigned a reference drainage area value of 1 corresponding to a uniform amount of rain. Cells are then processed in descending order, incrementally computing the drainage area by propagating the values to lower cells according to the flow routing equation. This algorithm has $O(n^2 \ln n)$ complexity (with n the width of the terrain in cells) and cannot be easily parallelized. Improved techniques for cells connected by a graph perform in linear time [Cordonnier et al.

2019]. However, those methods do not lend themselves to parallel implementation on graphics hardware.

One particular challenge in accelerating the erosion simulation is to compute the drainage area in parallel [Barnes 2019]. Our solution consists of an iterative approximation of the drainage area that progressively converges to the exact value and can be used in the parallel erosion algorithm. At every step, we compute the flow routing for the 8 neighboring cells $\mathcal{V}_i$ according to equation 4. We then evaluate the fraction w_{ji} of the drainage area of each neighbor flowing into the cell c_i, and update the value $\tilde{a}_i$. Therefore, for every cell c_i, we analyse 5×5 cells (see Figure 5). The values $\tilde{a}_i(k)$ obtained at iteration k are used to compute the flow contributions at the next iteration $k+1$:

$$\tilde{a}_i(k+1) = 1 + \sum_{j \in \mathcal{A}_i} w_{ji}(k)\, \tilde{a}_j(k) \tag{5}$$

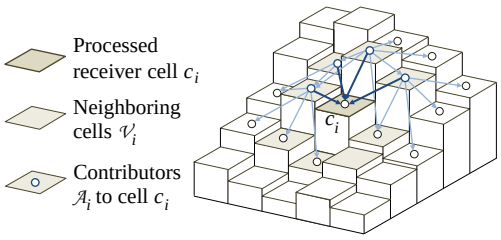

Fig. 5. Approximation of the drainage area $\tilde{a}$ using the direct neighboring cells $\mathcal{A}_i$ above c_i.

The convergence rate of this approximation depends on the length of the maximum flow path in the terrain. Theoretically, the optimal case arises when the terrain has a pyramidal shape with its peak at the centre, with a maximum flow path length equal to $n/2$, where n denotes the grid resolution. In the worst case, a flow path is meandering through the terrain [Hooshyar et al. 2020], producing flow paths with $O(n^2)$ length. The drainage area network progressively improves through the first iterations and becomes rapidly accurate over the entire domain. Figure 6 shows the evolution of the drainage network on a 256 × 256 terrain.

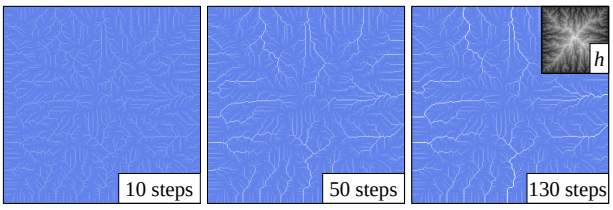

Fig. 6. Approximation of the drainage area $\tilde{a}$ on a 256 × 256 resolution terrain; the reference is reached at 130 iterations.

We performed a series of experiments on real-terrain digital elevation maps and synthetic terrains to evaluate the convergence rate and compare it to a reference implementation. Results show that our method yields the same drainage area after several iterations ranging between n and $4\,n$, with an average of $1.5n$.

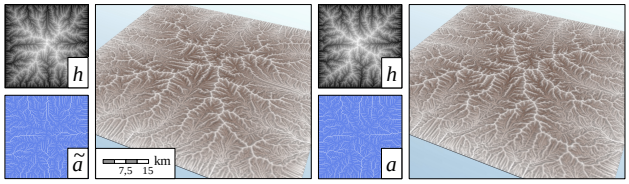

Fig. 7. Comparison between the stream power erosion with the approximation $\tilde{a}$ (left) and with the reference computation a (right). The two methods produce highly similar results.

Using an iterative approximate solver rather than computing the exact solution is compensated by the parallel execution. Moreover, the approximation proves to be sufficiently accurate, even with only a few steps, to set up the erosion process (see accompanying video). One iteration takes a few milliseconds and the stream power erosion algorithm converges to the final steady-state elevation in a few seconds for medium sized grids, *i.e.* $\leq 1024 \times 1024$ (see Section 7.2).

The consistency of the surface flow of water is a crucial aspect of digital landscape realism. We qualify a river network as consistent when the flow is directed to the boundary of the domain, without pit-cells retaining water. Most terrain synthesis techniques do not guarantee the consistency of the water flow, and require a pit-removal process: most often depression breaching or filling [Barnes et al. 2014]. In contrast, our erosion simulation based on the stream power equation converges to a consistent water flow and a more natural river network.

Figure 7 shows a comparison between the stream power erosion with the drainage area approximation $\tilde{a}$ and the reference implementation. Both terrains were generated with a uniform uplift u_0 over a square domain. The two methods produce similar ridge and river network patterns: differences resulting from our approximations remain subtle and not crucial in our context.The simulation converges to a consistent drainage area and forms the classic dendritic cross patterns described by Bonetti *et al.* [2020] and Hooshyar *et al.* [2020]: results conform to the reference implementation, even with the use of the approximate drainage area $\tilde{a}$.

5 UPLIFT CONSTRUCTION TREE

Recall that the uplift defines the speed at which the ground is elevating, for each point of the domain. To build uplift maps efficiently, we introduce a hierarchical construction tree, the *Uplift Tree*, denoted as $\mathcal{T}$. We introduce asymmetric profile-based primitives for generating asymmetric large-scale mountain ranges and discontinues primitives and operators to approximate faults. Our algorithms synthesize and update the tree structure interactively. The construction tree can be queried at a specific point $u(\mathbf{p})$ or converted into a discrete map and directly streamed to the GPU for the erosion simulation.

Figure 8 depicts a typical Uplift Tree structure, the corresponding map u_0, and the generated relief h, and demonstrates its part in the creation of the terrain's main features. The leaves of the tree are primitives defined over a compact region in $\mathbb{R}^2$. The nodes are either unary operators (deformations or affine transformations) or binary operators (minimum, maximum, blending) that combine

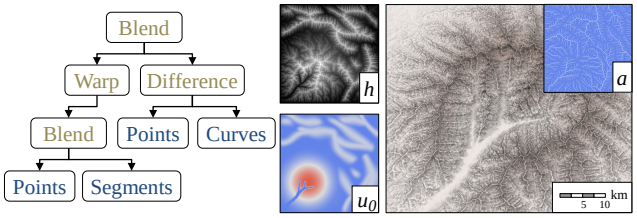

Fig. 8. Example of an uplift scalar field u_0 authored with the Uplift Tree. Curve primitives define the mountain ranges in the north-east, whereas a large disc-based primitives create the south-west mountain range. The river network was carved using segment primitives.

values calculated by their sub-trees. The value at a given point is evaluated by recursively traversing the tree.

5.1 Uplift primitives

The function of a primitive, denoted as $u_{\mathcal{P}}$, combines the distance d to a skeleton $\mathcal{S}$, with a falloff function g parameterized by a radius r and a maximum value located on the skeleton $u_{\mathcal{S}}$:

$$u_{\mathcal{P}}(\mathbf{p}) = g \circ d(\mathbf{p}) \qquad g(d) = \begin{cases} u_{\mathcal{S}}\,(1 - d^2/r^2)^3 & \text{if } d \le r \\ 0 \text{ otherwise} \end{cases} \tag{6}$$

Mountain ranges are typically asymmetrical, which is the consequence of tectonics [Willett et al. 1993]. On one side of the mountains, the strong gradient leads to elevated topography and steep surface slopes, whereas the other side features gentle slopes. Geomorphology demonstrates that this overall asymmetry comes from several factors, including crust folding, faulting and alignment orthogonal to the compression direction. Instead of taking those complex crust and layered strata phenomena into account as done by Cordonnier *et al.* [2018], we developed computationally efficient asymmetric primitives, taking inspiration from the implicit star-shaped surfaces of Crespin [1996].

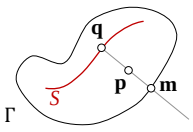

Fig. 9. Asymmetric primitives construction.

To introduce asymmetry in the classic skeleton-based primitives, we replace the Euclidean distance with an anisotropic pseudo distance $\tilde{d}$. It is computed from a closed boundary curve Γ that encompasses a curve skeleton $\mathcal{S}$ (see Figure 9). In our framework, Γ is a piecewise quadric Bezier curve, which permits efficient nearest point computation and intersection with a segment. For a point $\mathbf{p}$, we first compute its orthogonal projection $\mathbf{q}$ on $\mathcal{S}$, then define $\mathbf{m}$ as the first intersection between $[\mathbf{q}, \mathbf{p})$ and Γ. Finally, the pseudo distance $\tilde{d}$ is:

$$\tilde{d}(\mathbf{p}) = \frac{\|\mathbf{q} - \mathbf{p}\|}{\|\mathbf{q} - \mathbf{m}\|} \tag{7}$$

$\mathcal{S}$ defines the set of points where the uplift is maximum, *i.e.*, the crest line in the elevation domain, whereas the curve Γ determines the boundary of the compact support, *i.e.* the extent of the mountain range.

Figure 10 illustrates the effectiveness of star-shaped primitives for large-scale terrain authoring. The east-west asymmetry of the mountain ranges results from the asymmetry of the uplift. We observed that regions with a high gradient always create a high gradient in the elevation, *i.e.* a steep slope flank.

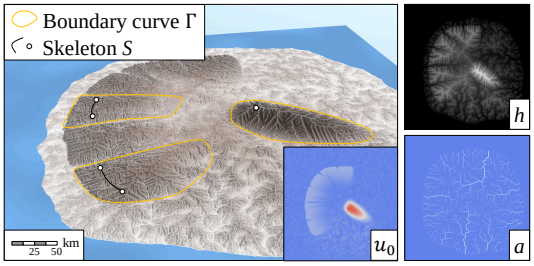

Fig. 10. Asymmetrical star-shaped primitives used to create a large mountain system near the sea.

As more complex examples, we describe here two typical primitives useful for creating large-scale landscapes: the first used for mountain range modeling (see Section 6.3), the second designed to carve valleys (see Section 6.2).

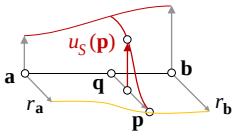

Fig. 11. Mountain range uplift profile.

The segment mountain range primitive is based on segment-skeleton $\mathcal{S} = [\mathbf{a}, \mathbf{b}]$ associated with a varying radius $r : [0, 1] \to \mathbb{R}_+$ and profile $u_{\mathcal{S}} : [0, 1] \to \mathbb{R}_+$ along the skeleton.For any point $\mathbf{p}$ in the plane, we compute its projection $\mathbf{q}$ on the segment. Let $x = \|\mathbf{a}-\mathbf{q}\|/\|\mathbf{a}-\mathbf{b}\|$ clamped between 0 and 1, we compute the corresponding radius $r(x)$ and value on the skeleton $u_{\mathcal{S}}(x)$. We finally compute $u_{\mathcal{P}}(\mathbf{p})$ by evaluating the distance $\|\mathbf{p} - \mathbf{q}\|$ and applying the falloff function of equation (6) with radius $r(x)$ and skeleton $u_{\mathcal{S}}(x)$.

The valley primitive is also based on a segment-skeleton $\mathcal{S} = [\mathbf{a}, \mathbf{b}]$, whereas the classic falloff is replaced by a *valley profile* curve, with low values in the valley zone, and high values outside (see Figure 12). The main parameter of the profile is the varying width of the valley $w : [0, 1] \to \mathbb{R}_+$ along the segment. The user may edit specific profile curves, keeping in mind that high slopes are required to produce valleys during the simulation.

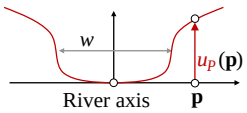

Fig. 12. Valley uplift profile curve.

5.2 Operators

To complete the set of previously presented primitives, operators offer a flexible way to transform a subtree or combine several ones. Blending or warping in the elevation often yields unrealistic artificial-looking terrains with inconsistent drainage network featuring numerous pits, and a loss of the characteristic geometric proportions and power laws of landforms.

Working in the uplift domain alleviates this limitation. Extreme deformations or even discontinuities are correctly handled by the erosion process and do not violate the geomorphological and hydrological coherence of the resultant terrain. We separate operators into two categories: unary operators encompass affine transformations and non-linear deformations, whereas binary operators implement adding, blending, or carving sub-trees. We present a relevant example for each of these two categories.

Warping. We use space deformations, implemented as a unary warping operator, to remove the regular pattern produced by skeleton-based primitives. The deformation is defined by a bijective function $\omega : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, and applied to a sub-tree A. Let $u_{\omega(A)}$ the uplift of the resulting warped sub-tree, we have:

$$u_{\omega(A)} = u_A \circ \omega^{-1} \tag{8}$$

The advantage of warping the uplift instead of the elevation is illustrated in Figure 25 and further discussed in Section 7.1. Faults and folds modeled in the uplift domain effectively approximate the large-scale deformed mountain ranges, preserve the erosion patterns, and avoid distorted elevations.

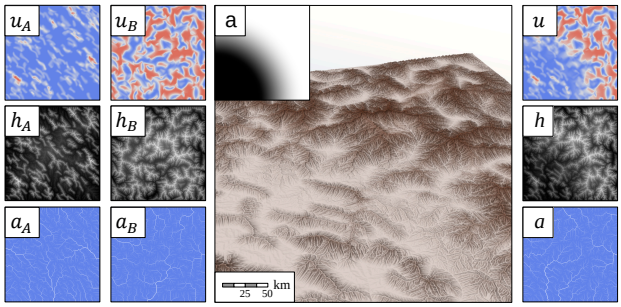

Fig. 13. Example of uplift blending: the generated waterflow remains coherent while features from both input terrains are preserved thanks to the erosion simulation.

Blending. Modeling complex terrains often requires assembling parts of selected models. We adapt the blending operator that smoothly combines terrain patches described by Génevaux *et al.* [2015] in the uplift domain, casting the elevation computation to the simulation. A detailed comparison of the results is given in Section 7.1. The blending operator takes two sub-trees A, B, and a weighting function $\alpha : \mathbb{R}^2 \rightarrow [0, 1]$. The resulting uplift u_{A+B} is defined as:

$$u_{A+B} = (1 - \alpha)u_A + \alpha\, u_B \tag{9}$$

The stream power erosion guarantees the consistency of the drainage when it reaches a steady-state (see illustration in Figure 13) contrary to traditional blending that operates directly on elevation and that produces not only a blurring effect on elevations but also inconsistent drainage.

6 SIMULATION CONTROL

In this section, we show that authoring the base uplift u_0 offers a unified framework for generating large-scale structures and offers many terrain authoring possibilities for controlling the landforms produced by the erosion simulation, as outlined in Figure 2. This model provides landscape artists with tools for shaping mountain ranges and valleys, such as copy-and-paste operations, and warping for imitating folds and faults. Section 6.3 presents a procedural mountain range pattern generation algorithm based on user prescribed folds. Section 6.1 and 6.2 describe methods to build u_0 from graphs representing the ridge lines and river beds of a terrain.

While modeling the base uplift u_0 proves to be efficient for modeling mountain ranges, it remains somewhat limited for prescribing specific elevations for peaks or crest lines. Therefore, Section 6.4 introduces point and feature curve elevation-based constraints that modify the dynamic uplift component $\tilde{u}$ throughout the simulation to retarget the elevation according to the designer prescriptions.

6.1 Ridge networks

In this section, we present a compact vector-based model for characterizing the prevailing landforms of a mountain range.

Argudo *et al.* [2019] provide a method to analyze and extract the representative characteristics of mountainous terrain as a spanning tree, namely a divide tree or ridge tree, connecting significant peaks and saddle points. Although a thorough presentation of the divide tree is beyond the scope of this method, let us briefly recall the definition of peaks and saddles. Peaks are defined as the most significant local maxima of the surface and connected together through a spanning tree that maximizes the minimum elevation of each peak-to-peak path. For each of those paths, the saddle is the point of minimum elevation. Finally, the divide tree is defined as a set of saddle-to-peak paths that we approximate as segments, associated with the elevation information of the points. This structure can either be extracted from DEMs, synthesized to respect a given style of landscape, or sketched by the user.

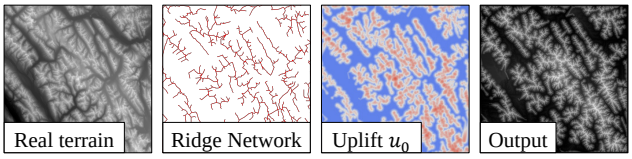

Fig. 14. Illustration of the ridge network constraint: the ridge network is extracted from a real terrain and converted into an Uplift Tree streamed to the simulation.

Let $\mathcal{R} = \{\mathcal{R}_i\}_{0 \leq i \leq n}$ denote the set of segments that compose the divide tree, and $\mathbf{a}_i$, $\mathbf{b}_i$ the end points of the segments. To build the Uplift Tree $\mathcal{T}$, each $\mathcal{R}_i$ is converted into a mountain range primitive (see Section 5). The radii and values are set proportionally to the elevation values of the divide tree. Primitives are finally assembled into a tree structure using the maximum operator.

Using the resulting static uplift u_0, the erosion simulation yields landforms that conform to the structure of the input ridge network and accord to the approximate elevations of the peak and saddle points. This demonstrates the effectiveness of the divide tree for controlling large scale mountain ranges.

The divide tree provides us with complementary strategies for generating terrains. Sketching or procedurally generating a divide tree gives control to the user over the position and shape of the

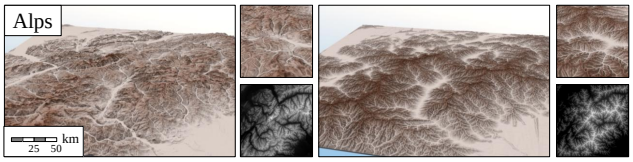

Fig. 15. Comparison between a real digital elevation model of the Alps and a simulation based on the static uplift u_0 built from its ridge network $\mathcal{R}$.

mountain ranges. Another strategy consists in extracting the divide tree from a digital elevation model and using it in an inverse procedural modeling process to generate a static uplift. The stream power erosion process produces a terrain that approximates the original input terrain. Figures 14 and 15 highlight this inverse procedural modeling process and exhibit a lack of variety in the elevation of valleys, that comes from the absence of sediment depositions in the model (see Section 7.5). Actually, the main features of the input terrain are successfully reproduced by the simulation, confirming the inverse procedural modeling capacities of our model.

6.2 River networks

River networks and drainage valleys play a crucial part in shaping the mountain ranges. By limiting the amount of tectonic uplift within river and valley areas, we force the flow routing to those specific zones and guide the erosion process.

The river network can be either sketched or derived by analysing real terrain digital elevation models as exposed by Argudo *et al.* [2019]. In the case of a user-defined network, the graph should contain coherent elevations or drainage area values. Should the input be inconsistent, the simulation sill yields a hydrologically-coherent river network by construction which might not address some of the user's prescriptions (see accompanying video).

Recall that a river tree $\mathcal{R}$ is a directed acyclic graph connecting nodes $\mathcal{R}_k$ defined as geometric points that are characterized by their elevation and drainage area h_k and a_k respectively. Similarly to the case of the ridges, we convert every edge in $\mathcal{R}$ into a *valley* uplift primitive (Section 5.1). The drainage area along the segment is computed by interpolating the drainage values at the corresponding nodes in the graph. The varying valley width w is computed using power law: $w \propto a^{0.4}$ introduced in geomorphology (see the work of Harel *et al.* [2022] for a discussion on the width-area-slope scaling of valleys and channels).

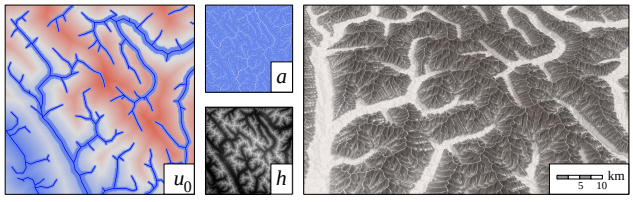

Fig. 16. Example of a static uplift u_0 created by carving a river network with varying valley widths.

Figure 16 shows that the valleys are perfectly shaped according to the river network.

6.3 Procedural generation

A key feature of large scale mountain ranges is the distribution of mountainous clusters following folds and faults produced by the tectonic forces. Instead of simulating the crust folds and faults as presented by Cordonnier *et al.* [2018], we propose an original procedural approach for generating an Uplift Tree controlled by a user-prescribed vector field v representing the crust folds.

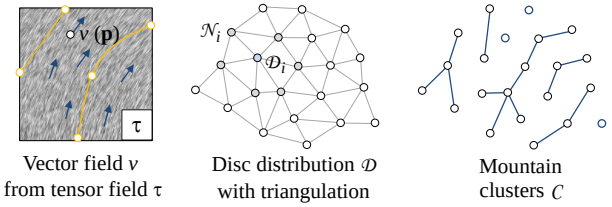

Fig. 17. Principal components of the procedural dendritic ridge cluster generation algorithm.

The algorithm operates in three steps (Figure 17). First, the designer defines a tensor field by interactively combining primitives which, in turn, define a vector field, as advocated by Zhang *et al.* [2007]. The large scale structures of the mountains are then generated by using an anisotropic off-lattice Eden growth model whose growth probability is driven by the direction of the vector field. Finally, the dendritic clusters are converted into primitives and assembled into an Uplift Tree $\mathcal{T}$. Altogether, the cluster-based generation algorithm synthesizes large-scale mountain ranges primitives aligned with user-defined directions.

The vector field v sets the main direction of the folds for every point $\mathbf{p}$ in the domain. It is computed from a tensor field τ as presented by Zhang *et al.* [2007], designed through a construction tree combining primitives in a hierarchical structure, which allows for interactive modifications of the tensor field τ like blending or copy-paste operations, hence the resulting vector field v.

Once the vector field v has been set by the designer, we sample the domain using a Poisson-disc [Lagae and Dutré 2006] distribution with a radius $r_{\mathcal{D}}$ corresponding to the smallest mountainous landmark. Let $\mathcal{D} = \{\mathcal{D}_i\}, 0 \leq i \leq n_{\mathcal{D}}$ denote the set of discs. The discs are then assembled into clusters C_i.

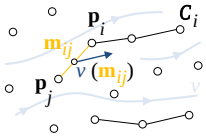

Fig. 18. Anisotropic cluster growth.

We generate clusters by applying a greedy off-lattice Eden growth algorithm [Murray. 1961; Wang et al. 1995] to the set of discs. To optimize Eden growth, we first perform a Delaunay triangulation of the discs to accelerate the nearest neighbor queries required in the growth process. Therefore, we define the neighborhood $\mathcal{N}_k$ of $\mathcal{D}_k$ as the set of discs that share an edge with $\mathcal{D}_k$ in the Delaunay triangulation.

Starting from a yet unlinked disc $\mathcal{D}_i \in \mathcal{D}$, the growth process incrementally aggregates discs by repeating the following steps:

1) Select a random cluster C_i in the set of clusters, and a random disc $\mathcal{D}_j$ in C_i. 2) Select an unlinked disc $\mathcal{D}_k$ in $\mathcal{N}_j$ if it exists, and add it to the cluster C_i by creating an edge between $\mathcal{D}_k$ and $\mathcal{D}_j$ (Figure 18) if the edge accords to the direction of the vector field v. 3) Otherwise, define $\mathcal{D}_j$ as a new cluster, remove it from the set of candidate discs. The creation of new clusters stops when the set of discs $\mathcal{D}$ is empty.

The condition for creating an edge in the cluster growth process is based on the underlying direction of the vector field v. Aggregation depends on the angle between the edge and the direction of v: a maximum angle α is used. Let $\mathbf{p}_i$ and $\mathbf{p}_k$ centres of the discs $\mathcal{D}_k$ and $\mathcal{D}_i$, and $\mathbf{m}_{ik}$ the mid-point. We test the scalar product between the unit vector of the direction of the edge connecting $\mathcal{D}_k$ and $\mathcal{D}_i$ and the vector field:

$$|(\mathbf{p}_k - \mathbf{p}_i) \cdot v(\mathbf{m}_{ik})| \,/ \|\mathbf{p}_k - \mathbf{q}_i\| \leq \cos(\alpha)$$

The Eden clusters are converted into an Uplift Tree model by parsing the edges of the clusters' graphs and generating segment-based primitives. At this stage, a variety of user-defined control maps modulate the construction process.

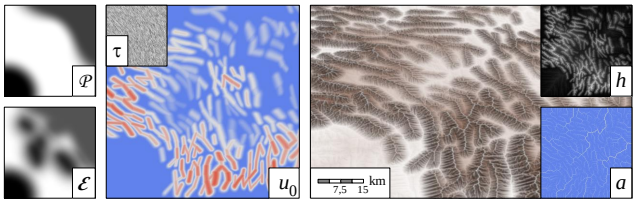

Fig. 19. Uplift map obtained from the tensor field τ and elevation and density control maps $\mathcal{E}$ and $\mathcal{P}$, the resulting terrain shows dendritic mountain ranges aligned with the vector field.

In our model, we rely on two low-resolution density maps: $\mathcal{P}$ representing the probability of the presence of a mountain range, and $\mathcal{E}$ defining the target global elevation of mountain ranges. We remove segments according to $\mathcal{P}$ and parameterize the maximum uplift value u_S and radii of the primitives according to $\mathcal{E}$. Eventually, procedurally defined low-frequency deformation operators are inserted in the Uplift Tree $\mathcal{T}$ to warp crest lines. Figure 19 depicts the generation process with slightly warped primitives aligned with the direction of the user-defined vector field v.

6.4 Elevation constraints

The Uplift Tree $\mathcal{T}$ defining u_0 provides a global control over the major landforms. However, the precise control of the elevation of characteristic landforms such as peaks, ridges or valleys can be difficult: the steady-state elevation reached at a point $\mathbf{p}$ does not only depend on $u_0(\mathbf{p})$ but is a complex process ruled by the stream power erosion. This non-linearity can make the proper control of altitudes a challenging task, and controlling the altitude of specific points in the terrain remains difficult.

We define elevation constraints by introducing a dynamic uplift component $\tilde{u}$ to adapt the values of u during the simulation. A fundamental aspect of elevation constraints is their compatibility with the global erosion simulation. The value depends on the measured error $e(t) = h_0 - h(t)$ between the target elevation h_0, and the current one $h(t)$ in the simulation. Depending on the sign of $e(t)$, the uplift needs to increase or decrease by an unknown amount. This task is performed using a *proportional–integral–derivative* (PID) controller:

$$\tilde{u}(t) = K_p e(t) + K_i \int_{t_0}^{t} e(t)dt + K_d \frac{de(t)}{dt}$$

The coefficient K_p adjusts the uplift proportionally to the error. The integral coefficient K_i increases the response according to the persistence of the error: the longer a small error persists, the higher the uplift is pushed to make it disappear. Finally, the derivative coefficient K_d controls the influence of the variation of the error: the faster the error is rising, the higher the feedback to stabilize it.

In the discrete case, the dynamic uplift update at iteration $k+1$ is defined as follows:

$$\tilde{u}(k+1) = K_p\, e(k) + K_i \sum_{j=0}^{k} e(j) + K_d\ (e(k) - e(k-1)) \qquad (10)$$

Updating $\tilde{u}$ and the controller state at every iteration is not mandatory and can be performed only at a user-given regular interval. The parameters were set experimentally to $K_p = 0.01$, $K_d = 0.01$ and $K_i = 0.001$, for an update every 10 iterations.

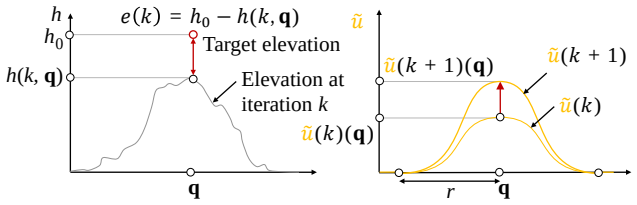

Fig. 20. Dynamic update of $\tilde{u}$ in response to the elevation error e.

Point-based constraints are useful for prescribing the elevation of peaks or other points of interest. Let $\mathcal{P}$ denote a point constraint located at $\mathbf{q} \in \mathbb{R}^2$, and h_0 its prescribed elevation. We define a corresponding dynamic uplift primitive $\tilde{u}_\mathcal{P}$ defined as a compactly supported smooth step function centred at $\mathbf{q}$. The radius r is set by the user to define the influence of the constraint, and $\tilde{u}_\mathcal{P}(\mathbf{q}, 0)$ is initialized to 0. A controller is then assigned to this constraint with the error function $e(k, \mathbf{q}) = h_0 - h(k, \mathbf{q})$. At every iteration k, we compute the correcting term $\tilde{u}(k+1)$ for the primitive (see Figure 20).

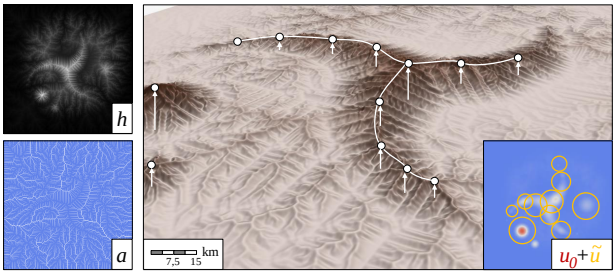

Fig. 21. Example of peak and ridges produced by elevations constraints.

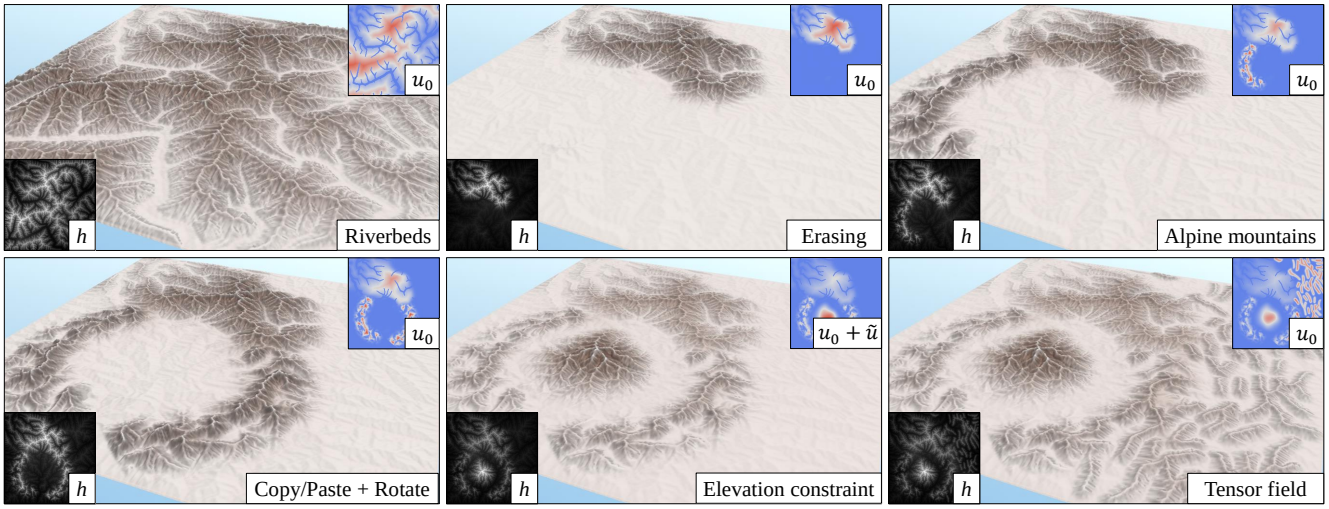

Fig. 22. Example of an interactive editing session. The designer first generated an Uplift Tree from a river network and let the erosion simulation converge. She then erased the uplift over an entire region, which removed mountains and instead used an Alpine brush to create a half-circle shaped mountain range. The circular mountain range was obtained by selecting, rotating and copying the previously defined Alpine uplift. Finally, the designer specified some explicit elevation constraints, before filling the flatlands with smaller mountain ranges (along a main direction, in a similar way to the Iranian Zagros range) produced by our procedural tensor field generation algorithm.

The uplift value corresponding to the target elevation h_0 is found when an elevation controller reaches a steady-state. The controllers are deactivated to freeze the corresponding dynamic uplift component $\tilde{u}$ and integrate it into the static term u_0. The simulation finally converges while conforming to the elevation constraints with a coherent flow discharge.

Figure 21 illustrates curve constraints to set ridges elevation. Given a user-defined target ridge curve and its corresponding elevation, we sample the curve to place multiple point constraints that will form a continuous ridge; limitations are discussed in Section 7.5.

7 RESULTS

We implemented our method in C++, the drainage area and stream power erosion algorithms were coded as GLSL compute shaders. Experiments were performed on a desktop computer equipped with Intel® Core *i*7, clocked at 4 GHz with 16 GB of RAM, and an NVidia GTX 1080ti graphics card. The generated terrains were directly rendered in real-time with a GLSL shader using the optimized Sphere Tracing algorithm for terrains [Génevaux et al. 2015; Hart 1996] to provide immediate feedback to the user or streamed into Eon-Software Vue to produce photorealistic landscapes (Figures 1 and 29). The erosion simulation code is available at https://github.com/H-Schott/StreamPowerErosion.

7.1 Control

Figure 22 illustrates various aspects of user control with screenshots from a typical authoring session. In general, inexperienced users favour high-level tools such as copy-and-paste or predefined uplift maps computed from digital elevation models that allow efficient creation and duplication of complex mountain ranges. A variety of uplift tools generate realistic mountain ranges in a single stroke, which allows for rapid sketching of large-scale terrains but provides less precision than the low-level tools favoured by experienced artists. Elevation constraints lend themselves to specifying the elevation of peaks.

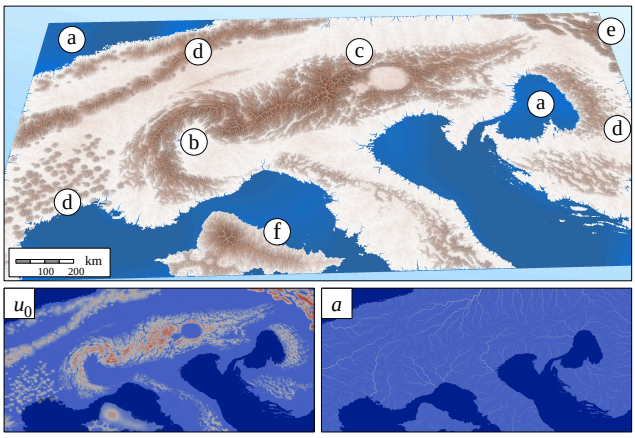

Fig. 23. Authoring of a large terrain (about 1200 × 600 km), starting from an inverse procedural modeling of the Alps, and using several high-level brushes to edit existing features and add new ones: (a) - sea editing; (b) - twist deformation; (c) - erase brush to create craters; (d) - point primitives distribution; (e) - low frequency warping; (f) - sketching of a replica of O'ahu island in the Mediterranean sea.

Users can model large-scale terrains featuring different types of valleys and landforms in a few minutes. The editing session in Figure 22 took 3 minutes (see the accompanying video), whereas

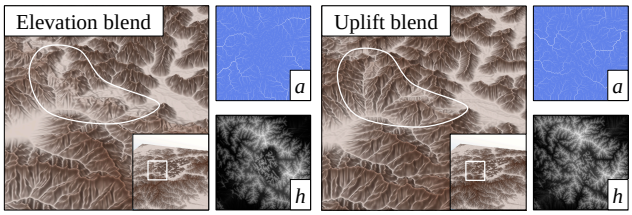

Fig. 24. Two terrains obtained from the erosion simulation are blended together using their elevation or their uplift. The loss of natural features such as dendrites is emphasized in white, while the drainage areas show that the second method preserves a consistent flow.

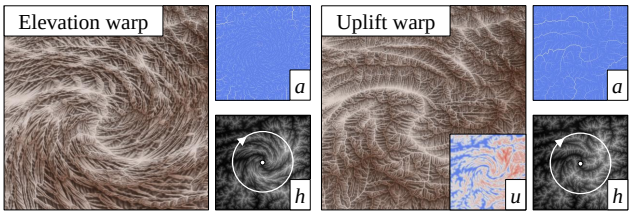

Fig. 25. A terrain obtained from the erosion simulation is warped by twisting its elevation or its uplift. While the main lines are following the warping as intended in both cases, we observe that, with the elevation twist, the dendritic features are also inevitably distorted.

the landscape depicted in Figure 23 was completed in less than 10 minutes by an experienced user.

Editing brushes are based on well-designed primitives adapted to authoring. Although indirect, the Uplift Tree provides control over the location and the average elevation of mountains and valleys; and over the trajectories of ridges and rivers. The designer can enforce a given dendritic ridge or river network using either divide trees or river trees as advocated by Argudo *et al.* [2019] or by sketching low uplift valleys.

Uplift operators. Here we advocate uplift operators as more suitable modeling tools than direct elevation combinations and transformations. As stated in the Section 5.2, blending terrain parts or applying deformations to elevation often yields unrealistic results with over-distorted reliefs, blurred landforms, and inconsistent drainage networks. In contrast, performing interpolation (Figure 24) or warping in the uplift domain (see Section 5.2 and Figure 25) allows complex editing processes while preserving the overall hydrology consistency provided by the stream power erosion simulation.

Inverse procedural modeling. The vector-based representation of the Uplift Tree delivers valuable properties: it does not depend on the underlying grid resolution, and it is compact in memory, which qualifies it for inverse procedural modeling methods. Our orometry-based approach [Argudo et al. 2019] can take any digital elevation model as input, extract the ridge and river networks, convert the graphs to uplift primitives and generate a compact Uplift Tree representation completed with elevation constraints for prominent peaks. The only information needed to reconstruct an approximation of the input terrain is the compact Uplift Tree. Although the synthesized terrain is not strictly identical to the input, the stream power erosion generates landforms with the same major ridges and valleys, as illustrated in Figures 15, 16, and 27.

7.2 Performance

Erosion simulations are a key modeling tool for virtual terrains and achieving interactive visualisation helps artists and geologists understand the processes and tune parameters accordingly. Another crucial aspect of the stream power erosion simulation is the convergence to a steady-state. We compare our approach to the state-of-the-art erosion methods considering both interactivity and convergence speed.

Size	Steps	t_s (ms)	t_c (s)
256	$5k$	0.1	0.5
512	$5k$	0.3	1.3
1024	$11k$	0.8	9.1
2048	$39k$	3.3	128
4096	$81k$	12.5	1036

Table 1. Performance of the simulation with a constant uniform uplift u_0: grid resolution, number of steps, average time for one step t_s (in milliseconds), total simulation time to converge t_c (in seconds).

Convergence speed. While the exact graph-based drainage area computations of Cordonnier *et al.* [2016] allows for fewer steps, our method still converges up to two orders of magnitude faster, thanks to the parallel implementation on graphics hardware. On a 160k points graph, equivalent to a 400×400 grid, their method requires $\approx$ 200 steps to converge, and each step takes about 1.3 *s* (total time 252 *s*). Our approach needs $\approx$ 5000 steps over a 512×512 grid with 0.3 *ms* per step for a total time of 1.3 *s* (see Table 1). Results demonstrate that the graphics hardware parallelism largely compensates the slower explicit solver.

The convergence to steady-state depends primarily on the grid resolution and the choice of Δt. The number of iterations to convergence increases for large grids (resolution greater than 1024×1024). One solution consists in adapting the time step Δt, which should be large enough to reduce iteration count, and small enough to guarantee stability. We observed that the number of pits-cell $\#\mathcal{P}$, *i.e.* cells with no downward neighbors, is related to the overall stability, since it decreases toward zero through the simulation. Therefore, we implemented a heuristic that adapts the time step Δt to the number of pit-cells by increasing it by 1% if $\#\mathcal{P}$ decreases, and strongly reducing it if the $\#\mathcal{P}$ significantly increases:

$$\Delta t_{i+1} = \begin{cases} 1.01\Delta t_i \text{ if } \#\mathcal{P} \text{ is decreasing} \\ 0.75\Delta t_i \text{ otherwise} \end{cases}$$

While this heuristic allows the process to converge in most cases, more complex integration schemes or multigrid techniques would undoubtedly improve the convergence of the simulation. Table 1 shows the time required to converge to steady-state under uniform uplift conditions using an adaptive time step simulation.

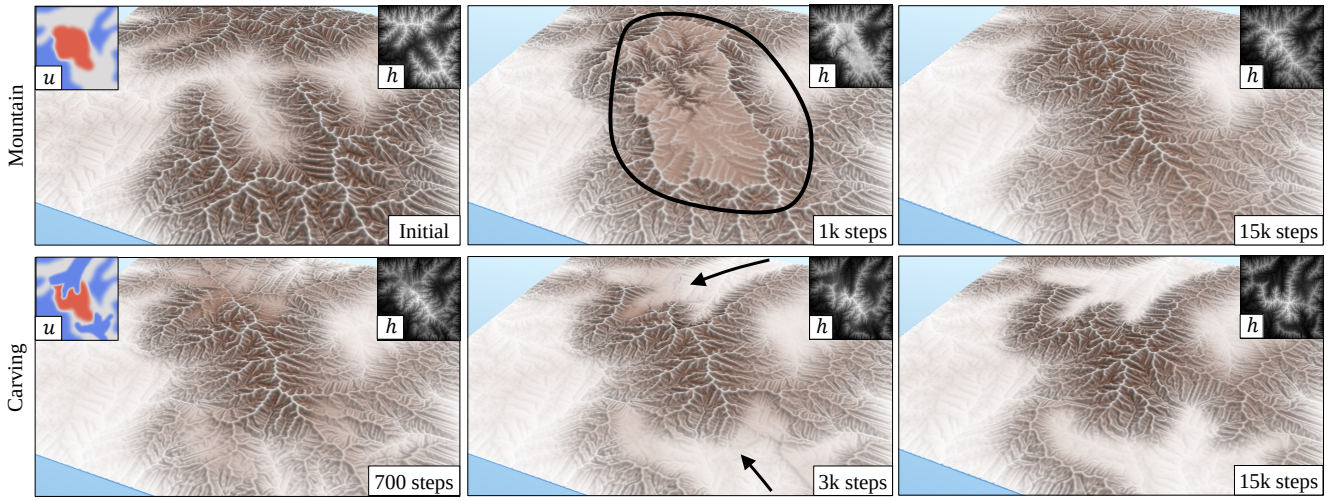

Fig. 26. Evolution of the terrain under two editing scenarios in the uplift domain: top row shows the apparition of a large mountain range at the center, and bottom row shows the creation of deep valleys starting from the borders of the domains. In both cases, the erosion rapidly creates the desired landforms (already visible after 1k steps) and then converges to a steady state.

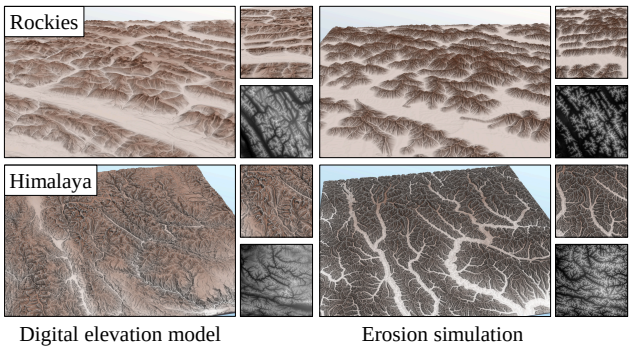

Fig. 27. Comparison between digital elevation models (North American Rockies and Himalaya) and the result of the inverse procedural method based on the erosion simulation (using the divide tree and the river tree respectively).

Interactivity. Cordonnier *et al.* [2018] were able to achieve interactive rates on 100×100 grids. Due to our efficient parallel implementation of the stream power erosion, we achieve interactive frame rates on large grids up to 4096×4096, as shown by the average times for one step in Table 1. The higher the resolution, the more steps are needed to observe the impact and results of an edition operation. In practice, we achieve real-time feedback and editing for 1024×1024 terrains (as shown in the accompanying video).

Furthermore, even though the stream power erosion may require many iterations to converge, the shape of the final terrain is observable after only a few hundred iterations, which allows fast feedback and lends our framework for interactive editing. Figure 26 shows the evolution of a 512×512 terrain after two editing processes: the creation of a large mountain range at the centre, and the carving of deep valleys from the borders. Figure 28 indicates the average elevation difference between the terrain at each step and the final terrain at steady-state using a $\mathcal{L}^1$ metric. Convergence curves show a rapid convergence to a first approximation of the elevation after a few hundred steps, then a plateau where ridge migration processes take place, and finally convergence to the steady-state.

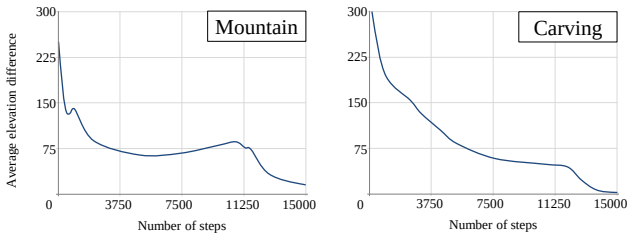

Fig. 28. Convergence curves for the terrains shown in Figure 26 under two editing scenarios using a $\mathcal{L}^1$ metric.

7.3 Validation

In addition to control tools and performance, an authoring framework is also characterized by the variety of the terrains it can produce. Figure 29 shows four different outputs of the erosion simulation and demonstrates that our model is capable of producing various types of large scale landscapes.

Evaluating the plausibility of a generated terrain is not a simple assignment. One solution consists in assessing the geomorphological coherency by verifying if observations made by geologists remain valid for synthesized models. We rely on Hack's river scaling power law observed in geomorphology [Hack 1957]. For a given point on the terrain, Hack's law correlates the area of its watershed (fundamentally its drainage area) a to the length of its longest upstream river l as follows:

$$l = c\, a^n \tag{11}$$

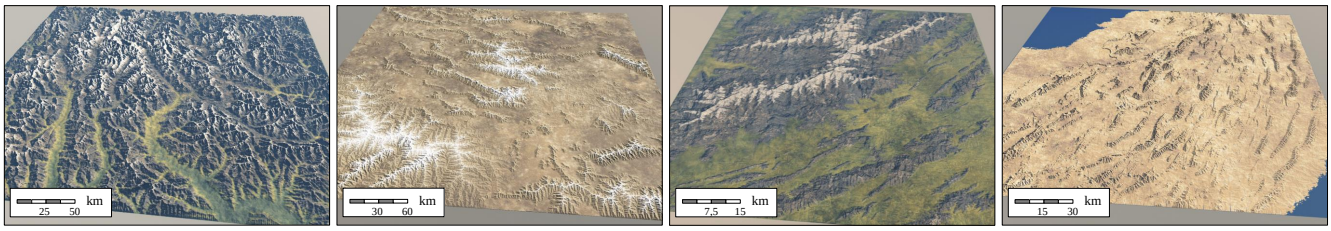

Fig. 29. Various large scale landscapes obtained through the erosion simulation, inspired from Himalayas, Nevada, Appalachians and Iran (from left to right).

Figure	Size	Cell size	$\#\mathcal{U}$	Memory	Hack	
					c	n
10	300	293	967	76	1.27	0.542
13	300	293	911	89	1.19	0.556
14	80	78	3511	281	1.29	0.581
15	150	146	7899	632	1.25	0.561
16	80	78	653	55	1.38	0.553
19	300	293	540	41	1.10	0.575
23	1200	600	≈ 35000	≈ 2800	1.00	0.584
24	150	146	11411	914	1.22	0.568
25	260	254	10128	810	1.15	0.563

Table 2. Size (in kilometers), cell size (in meters), number of nodes $\#\mathcal{U}$ in the Uplift Tree and memory size of the Uplift Tree (in kilobytes), of most of the generated terrains presented in the figures. All terrains are modeled by a 1024×1024 grid (except for Figure 23, which is 2000×1200).

The terms c and n are constants referred to as Hack's coefficient and Hack's exponent, studies [Sassolas-Serrayet et al. 2018] report a range of $[1, 6]$ for c and $[0.45, 0.7]$ for n.

Figure 30 reports statistics for two different terrains generated using the stream power erosion simulation. The obtained Hack's coefficients c and n lie in a valid range which demonstrates that our model conforms to this power law. Small watersheds ($a < 10\,\text{km}^2$) were removed from the plot to avoid artefacts to the grid resolution. The red line was fitted on the log-scaled terrain data using a least square method.

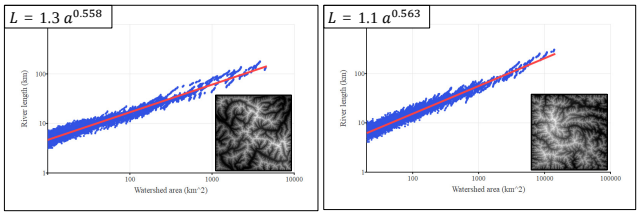

Fig. 30. Hack's Law validation for two terrains generated with the erosion framework. The obtained coefficients fit the ranges given in literature.

We tested all our generated models and the results indicate that they are validated by Hack's law. Table 2 reports the computed coefficients, which stand in the range described in geomorphology.

7.4 Comparison with other techniques

It is worth essaying a detailed comparison with other modeling approaches, particularly 1) hydrologically-based procedural models, 2) feature curves oriented models, and 3) surface erosion and tectonic simulation methods.

Procedural techniques. Procedural techniques and example-based techniques [Gain et al. 2015; Zhou et al. 2007] do not generate terrains that are hydrologically consistent. Scott and Dodgson [2021] propose to use a breaching algorithm [Lindsay. 2016] interlaced in multi-resolution example-based terrain synthesis to improve hydrological consistency, which only partially alleviates the problem as breaching still produces artificial canyons. In contrast, the stream power erosion algorithm always converges to hydrologically coherent terrains without pits, which allows the extraction and generation of river networks easier. Moreover, the Uplift Tree primitives can prescribe the width of valleys while preserving the overall drainage network. Génevaux *et al.* [2013] proposed a specific model focusing on the generation of a hydrologically correct river network. An essential limitation of this approach is that the procedurally generated mountain ranges may not be hydrologically correct, and may not exhibit characteristic dendritic shapes.

Feature curve models. Point and curve elevation constraints have already been introduced in other authoring contexts. Hnaidi *et al.* [2010] use control points on Bézier curves, whereas Guérin *et al.* [2022] relie on a user-defined gradient vector field with elevation and gradient constraints along curves and a tensor field to freely warp it. Gradient-based terrain editing based on control [Guerin et al. 2022] provide great control to the user but they do not create hydrologically-correct terrains, and using breaching [Lindsay. 2016] only partially alleviate the problems as visual artefacts can be created.

Hydrological realism comes from the stream power erosion simulation that carves the uplift-elevated terrain. Therefore, we neither rely on user-prescribed oriented noise to create dendritic features, nor need gradient constraints to assess coherent ridges or river networks. Nonetheless, we still need to set the elevation of the peaks or ridges, which is achieved by introducing elevation constraints with a varying uplift parameter $\tilde{u}$ (see Section 6.4).

Erosion simulation. Although producing visually convincing small-scale landmarks such as ravines or sedimentary valleys, surface erosion algorithms do not lend themselves to creating large-scale terrains. Most notably, those methods do not guarantee hydrologically correct reliefs and often produce pits in valleys, requiring

further post-processing using breaching or depression filling [Lindsay. 2016]. Closer to our technique is the method from Cordonnier *et al.* [2018]. While controlling the simulation of the compression and folding of tectonic plates using displacement motions proves to be a good approximation for producing realistic large-scale landforms, this indirect control does not qualify this approach for authoring. Moreover, the stream power erosion simulation relies on a computationally intensive drainage area computation, which drastically limits the terrain resolution. In contrast, our approach broadens the scope of stream power erosion to interactive authoring. We provide different editing tools and allowing to author and control large-scale terrains interactively by combining brushes (Section 6.4), procedural generation (Section 6.3), or using an inverse procedural modeling approach by generating uplift from ridge or river networks extracted from digital elevation models (see Section 6.1 and Section 6.2).

The coherence of the surface flow of water is a crucial aspect of digital landscape generation. We qualify a river network as coherent when the flow is directed to the boundary of the domain, without pit-cells retaining water. Most terrain synthesis techniques do not guarantee the coherence of the water flow, and require a pit-removal process. This extra step can take various forms: Scott and Dodgson [2021] remove the depressions in parallel with the texture synthesis algorithm, whereas Cordonnier *et al.* [2016] uses the flow routing algorithm to connect depressions forming lakes and preserve the consistency of the river network. While those methods gives great results, they cannot be performed on a GPU. In contrast, the erosion simulation based on the stream power equation consistently converges to a coherent river flow. While drainage area computations based on a hydrologically correct water flow require fewer steps, our iterative approach requires more steps but allows for a parallel implementation on graphics hardware.

7.5 Limitations

Elevation constraints are a powerful way to prescribe mountainous features through control points or curves describing peaks, crests lines, or river valleys. However, they do not come without limitations. First, the designer should handle constraints carefully by adapting them with either disjoint supports or compatible elevations. Placing several neighboring peak constraints with incompatible elevations and overlapping influence regions may lead to instabilities. The rationale is that each controller tries to satisfy different target constraints.

In the current implementation, the continuous update of the uplift allows for precise elevation constraints but slows down the overall simulation as the correction term $\tilde{u}$ is computed on the CPU. A straightforward acceleration consisted in updating $u = u_0 + \tilde{u}$ every ≈ 10 iterations, which proved to be sufficiently accurate. Another solution would consist of computing the dynamic term $\tilde{u}$ on graphics hardware, which is beyond the scope of the paper.

The resolution of the cells in the stream power erosion simulation ranges from 80 to 600 meters. Consequently, our synthesized terrains need to be post-processed using amplification techniques to reach a higher resolution, where details of a smaller scale can be added, like rivers with varying width [Peytavie et al. 2019], vegetation, or even roads. Our method produces standard heightfields and is therefore compatible with standard amplification techniques such as sparse modeling [Guérin et al. 2016], or learning-based approaches [Zhao et al. 2019]. Figure 1 (right) shows a synthesized terrain obtained after a $\times 8$ sparse amplification.

Our implementation of the stream power simulation ensures that the water flows outside the terrain. Although this gives realistic large-scale results, it does not allow for the placement of lakes. The temporal and spatial scale of the results also makes the conservation of matter challenging: uplift continuously elevates bedrock, whereas the stream power erosion removes material. Mass conservation is not guaranteed as the method considers that eroded material is transported and discharged out of the domain without any sediment deposition mechanism. Such processes could be taken into account [Yuan et al. 2019] at the expense of more intensive computation, or added as a post-processing step.

8 CONCLUSION

Combining the stream power erosion simulation with the uplift construction tree and user-prescribed sparse constraints provides foundations for interactive large-scale terrains authoring. Our method incorporates painting, sketching, and copy-and-paste of terrain fragments with automatic generation of hydrologically consistent reliefs. The controlled erosion formalism bridges the gap between simulation and authoring. Our fast parallel drainage area computation algorithm allows designers to interactively compose the large-scale shape of complex landforms with a broad spectrum of procedural and modeling tools, including landform feature primitives and ridge or river networks. Specific elevations prescribed by local controllers directly influence the erosion equation by dynamically retargeting the uplift.

While the erosion simulation and the orometry-based reconstruction do not provide a bijective mapping between the elevation and uplift domains, their relationship is intuitive and allows a variety of existing procedural and simulation-based modeling tools. Moreover, it lends itself to inverse procedural modeling by allowing the definition of the uplift from an input digital elevation model and capturing its essential landmarks.

This work opens avenues for future research. In particular, improving the stream power erosion model to take into account rock hardness (plateaus, strata), sediments (screes) [Yuan et al. 2019] or even glacial erosion (U-shaped and hanging valleys) [Hergarten. 2021] would be a direction worth investigating for modeling a broader range of landforms.

ACKNOWLEDGMENTS

This work was part of the project AMPLI ANR-20-CE23-0001, supported by Agence Nationale de la Recherche Française. We also thank James Gain (University of Cape Town, South Africa) for fruitful scientific discussions.

REFERENCES

Oscar Argudo, Eric Galin, Adrien Peytavie, Axel Paris, James Gain, and Eric Guérin. 2019. Orometry-Based Terrain Analysis and Synthesis. *ACM Transactions on Graphics* 38, 6, Article 199 (2019), 12 pages.

Richard Barnes. 2019. Accelerating a fluvial incision and landscape evolution model with parallelism. *Geomorphology* 330 (2019), 28–39.

Richard Barnes, Clarence Lehman, and David Mulla. 2014. Priority-flood: An optimal depression-filling and watershed-labeling algorithm for digital elevation models. *Computers and Geosciences* 62 (2014), 117–127.

Bedrich Benes. 2007. Real-Time Erosion Using Shallow Water Simulation. In *Proceedings of Virtual Reality Interactions and Physical Simulations*. Eurographics Association, Dublin, Ireland, 43–50.

Sara Bonetti, Milad Hooshyar, Carlo Camporeale, and Amilcare Porporato. 2020. Channelization cascade in landscape evolution. *Proceedings of the National Academy of Sciences* 117, 3 (2020), 1375–1382.

Gwyneth Bradbury, Il Choi, Cristina Amati, Kenny Mitchell, and Tim Weyrich. 2014. Frequency-Based Creation and Editing of Virtual Terrain. In *Proceedings of European Conference on Visual Media Production*. ACM, London, UK, Article 15, 10 pages.

Guillaume Cordonnier, Benoît Bovy, and Jean Braun. 2019. A Versatile, Linear Complexity Algorithm for Flow Routing in Topographies with Depressions. *Earth Surface Dynamics* 7, 2 (2019), 549–562.

Guillaume Cordonnier, Jean Braun, Marie-Paule Cani, Bedrich Benes, Eric Galin, Adrien Peytavie, and Éric Guérin. 2016. Large Scale Terrain Generation from Tectonic Uplift and Fluvial Erosion. *Computer Graphics Forum* 35, 2 (2016), 165–175.

Guillaume Cordonnier, Marie-Paule Cani, Bedrich Benes, Jean Braun, and Eric Galin. 2018. Sculpting Mountains: Interactive Terrain Modeling Based on Subsurface Geology. *IEEE Transactions on Visualization and Computer Graphics* 24, 5 (2018), 1756–1769.

Benoît Crespin, Carole Blanc, and Christophe Schlick. 1996. Implicit Sweep Objects. *Computer Graphics Forum* 15, 3 (1996), 165–174.

Giliam J. P. de Carpentier and Rafael Bidarra. 2009. Interactive GPU-based Procedural Heightfield Brushes. In *Proceedings of the International Conference on Foundations of Digital Games*. ACM, Orlando, USA, 55–62.

David S. Ebert, Forest Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. 1998. *Texturing and Modeling: A Procedural Approach* (3rd ed.). Elsevier.

James Gain, Bruce Merry, and Patrick Marais. 2015. Parallel, Realistic and Controllable Terrain Synthesis. *Computer Graphics Forum* 34, 2 (2015), 105–116.

James E. Gain, Patrick Marais, and Wolfgang Strasser. 2009. Terrain sketching. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*. ACM, Boston, USA, 31–38.

Eric Galin, Eric Guérin, Adrien Peytavie, Guillaume Cordonnier, Marie-Paule Cani, Bedrich Benes, and James Gain. 2019. A Review of Digital Terrain Modeling. *Computer Graphics Forum (Proceedings of Eurographics 2019)* 38, 2 (2019), 553–577.

Jean-David Génevaux, Éric Galin, Éric Guérin, Adrien Peytavie, and Bedřich Beneš. 2013. Terrain Generation Using Procedural Models Based on Hydrology. *ACM Transaction on Graphics* 32, 4 (2013), 143:1–143:13.

Jean-David Génevaux, Éric Galin, Adrien Peytavie, Éric Guérin, Cyril Briquet, François Grosbellet, and Bedrich Benes. 2015. Terrain Modeling from Feature Primitives. *Computer Graphics Forum* 34, 6 (2015), 198–210.

Eric Guérin, Julie Digne, Eric Galin, and Adrien Peytavie. 2016. Sparse representation of terrains for procedural modeling. *Computer Graphics Forum (proceedings of Eurographics 2016)* 35, 2 (2016), 177–187.

Eric Guérin, Julie Digne, Eric Galin, Adrien Peytavie, Christian Wolf, Bedrich Benes, and Benoit Martinez. 2017. Interactive Example-Based Terrain Authoring with Conditional Generative Adversarial Networks. *ACM Transactions on Graphics (proceedings of Siggraph Asia 2017)* 36, 6, Article 228 (2017), 13 pages.

Eric Guerin, Adrien Peytavie, Simon Masnou, Julie Digne, Basile Sauvage, James Gain, and Eric Galin. 2022. Gradient Terrain Authoring. *Computer Graphics Forum (proceedings of Eurographics 2022)* 44, 2 (2022), 85–95.

John T. Hack. 1957. Studies of longitudinal stream profiles in Virginia and Maryland. *U.S Geological Survey Professional Paper* 294-B (1957), 45–97.

Elhanan Harel, Lian Goren, Onn Crouvi, Hanan Ginat, and Eitan Shelef. 2022. Drainage reorganization induces deviations in width-area-slope scaling of valleys and channels. *Earth Surface Dynamics Discussions* 2022 (2022), 1–35.

John C. Hart. 1996. Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. *The Visual Computer* 12, 10 (1996), 527–545.

Stefan Hergarten. 2021. Modeling glacial and fluvial landform evolution at large scales using a stream-power approach. *Earth Surface Dynamics* 9, 4 (2021), 937–952.

Houssam Hnaidi, Éric Guérin, Samir Akkouche, Adrien Peytavie, and Éric Galin. 2010. Feature based terrain generation using diffusion equation. *Computer Graphics Forum* 29, 7 (2010), 2179–2186.

Peter Holmgren. 1994. Multiple flow direction algorithms for runoff modelling in grid based elevation models: An empirical evaluation. *Hydrological Processes* 8 (1994), 327–334.

Milad Hooshyar, Shashank Anand, and Amilcare Porporato. 2020. Variational analysis of landscape elevation and drainage networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 476, 2239 (2020), 20190775.

Alan D. Howard and Gordon Kerby. 1983. Channel changes in badlands. *Geological Society of America Bulletin* 94, 6 (1983), 739–52.

Alex D. Kelley, Michael C. Malin, and Gregory M. Nielson. 1988. Terrain simulation using a model of stream erosion. *Computer Graphics* 22, 4 (1988), 263–268.

Peter Krištof, Bedrich Benes, Jaroslav Křivánek, and Ondřej Šťava. 2009. Hydraulic Erosion Using Smoothed Particle Hydrodynamics. *Computer Graphics Forum* 28, 2 (2009), 219–228.

Ares Lagae and Philip Dutré. 2006. An Alternative for Wang Tiles: Colored Edges versus Colored Corners. *ACM Transactions on Graphics* 25, 4 (2006), 1442–1459.

John B. Lindsay. 2016. Efficient hybrid breaching-filling sink removal methods for flow path enforcement in digital elevation models. *Hydrological Processes* 30, 6 (2016), 846–857.

Xing Mei, Philippe Decaudin, and Baogang Hu. 2007. Fast Hydraulic Erosion Simulation and Visualization on GPU. In *Pacific Graphics*. IEEE, 47–56.

Eden Murray. 1961. A two-dimensional growth process. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 2. University of California Press, 223–239.

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

Benjamin Neidhold, Markus Wacker, and Olivier Deussen. 2005. Interactive physically based Fluid and Erosion Simulation. In *Proceedings of the Eurographics Workshop on Natural Phenomena*. Eurographics Association, Dublin, Ireland, 25–32.

Adrien Peytavie, Thibault Dupont, Eric Guérin, Yann Cortial, Benes Benes, James Gain, and Eric Galin. 2019. Procedural Riverscapes. *Computer Graphics Forum* 38, 7 (2019), 35–46.

Pascale Roudier, Bernard Peroche, and Michel Perrin. 1993. Landscapes synthesis achieved through erosion and deposition process simulation. *Computer Graphics Forum* 12, 3 (1993), 375–383.

Brennan Rusnell, David Mould, and Mark G. Eramian. 2009. Feature-rich distance-based terrain synthesis. *The Visual Computer* 25, 5-7 (2009), 573–579.

Timothée Sassolas-Serrayet, Rodolphe Cattin, and Matthieu Ferry. 2018. The shape of watersheds. *Nature Communications* 9, 3791 (2018).

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

Jan Seibert and Brian L. McGlynn. 2007. A new triangular multiple flow direction algorithm for computing upslope areas from gridded digital elevation models. *Water Resources Research* 43, 4 (2007), 1–8.

Ondřej Šťava, Bedřich Beneš, Matthew Brisbin, and Jaroslav Křivánek. 2008. Interactive Terrain Modeling Using Hydraulic Erosion. In *Proceedings of the Symposium on Computer Animation*. 201–210.

Flora Ponjou Tasse, Arnaud Emilien, Marie-Paule Cani, Stefanie Hahmann, and Adrien Bernhardt. 2014. First Person Sketch-based Terrain Editing. In *Proceedings of Graphics Interface*. Canadian Information Processing Society, Montreal, Canada, 217–224.

Nikos Theodoratos and James W. Kirchner. 2020. Dimensional analysis of a landscape evolution model with incision threshold. *Earth Surface Dynamics* 8, 2 (2020), 505–526.

Juraj Vanek, Bedrich Benes, Adam Herout, and Ondrej Stava. 2011. Large-Scale Physics-Based Terrain Editing Using Adaptive Tiles on the GPU. *IEEE Computer Graphics and Applications* 31, 6 (2011), 35–44.

Chang Y. Wang, Pei-Ling Liu, and James Bassingthwaighte. 1995. Off-lattice Eden-C cluster growth model. *Journal of physics A: Mathematical and general* 28 (1995), 2141–2148.

Kelin X. Whipple and Gregory E. Tucker. 1999. Dynamics of the stream-power river incision model: Implications for height limits of mountain ranges, landscape response timescales, and research needs. *Journal of Geophysical Research: Solid Earth* 104, B8 (1999), 17661–17674.

Sean Willett, Christopher Beaumont, and Philippe Fullsack. 1993. Mechanical model for the tectonics of doubly vergent compressional orogens. *Geology* 21, 4 (1993), 371–374.

Xiaoping P. Yuan, Jean Braun, Laure Guerit, Delphine Rouby, and Guillaume Cordonnier. 2019. A new efficient method to solve the stream power law model taking into account sediment deposition. *Journal of Geophysical Research: Earth Surface* 124, 6 (2019), 1346–1365.

Eugene Zhang, James Hays, and Greg Turk. 2007. Interactive Tensor Field Design and Visualization on Surfaces. *IEEE Transactions on Visualization and Computer Graphics* 13, 1 (2007), 94–107.

Yiwei Zhao, Han Liu, Igor Borovikov, Ahmad Beirami, Maziar Sanjabi, and Kazi Zaman. 2019. Multi-Theme Generative Adversarial Terrain Amplification. *ACM Transactions on Graphics* 38, 6, Article 200 (2019), 14 pages.

Howard Zhou, Jie Sun, Greg Turk, and James M. Rehg. 2007. Terrain Synthesis from Digital Elevation Models. *IEEE Transactions on Visualization and Computer Graphics* 13, 4 (2007), 834–848.